Advanced Card Systems Ltd.
Card & Reader Technologies

ACOS3 (Combi)

Reference Manual V3.28

info@acs.com.hk
www.acs.com.hk

Table of Contents

List of Figures

List of Tables

1.0. Introduction

The purpose of this document is to describe in detail the features and functions of the ACOS3 Combi card, a versatile smart card operating system developed by Advanced Card Systems Ltd.

1.1. Features

- Available in the following interface types:
  - Contact and contactless (Combi) interface
  - Contactless only interface
- Full 8 KB of EEPROM for application data
- Compliance with:
  - Contact: ISO 7816 Parts 1, 2, 3
    - Supports T=0 Protocol
  - Contactless: ISO 14443 Parts 1, 2, 3, 4
    - Supports T=CL protocol and fully compatible with ISO 14443 Type A
- High-speed transmission rate:
  - Contact: from 9.6 Kbps to 223.2 Kbps
  - Contactless: from 106 Kbps to 848 Kbps
- DES/Triple DES and MAC capability
- Five secret codes + issuer code
- PIN code that can be updated by card holder
- Key pair for mutual authentication
- Session key based on random numbers
- FIPS 140-2 compliant hardware based random number generator
- Binary files and record files that are available for user data storage
- Secure Messaging function for confidential and authenticated data transfers
- Support for highly secured e-Purse for payment applications

1.2. Technical Specifications

The following are the technical properties of the ACOS3 Combi card:

1.2.1. Electrical

- Operating Voltage: 5 VDC +/-10% (Class A) and 3 VDC +/-10% (Class B)
- Maximum Supply Current: < 10 mA
- ESD Protection: ≤ 4 KV

1.2.2. EEPROM

- Capacity: 8 KB
- EEPROM endurance: 500,000 erase/write cycles
- Data Retention: 20 years

1.2.3. Environmental

- Operating Temperature: -25 °C to 85 °C
- Storage Temperature: -40 °C to 100 °C

1.3. History of Modification

Date	Changes
December 2005	ACOS3-16 revision 1.0
April 2007	ACOS3-16 revision 1.01 • Modification of the ATR for backward compatibility
August 2007	ACOS3-16 revision 1.03 • Extra ISO 7816-3 guard time during transmission for enhanced reader compatibility • Enhancement added for 223.2 kbps communication support
October 2007	ACOS3-16 revision 1.06 • Added read/write record offset feature. • Added maximum code retry counters feature.
November 2007	ACOS3-24 revision 1.07 • Expanded user storage capacity to 24 Kbyte • Added Transparent (Binary) file structure • Added Secure Messaging for user data files
January 2008	ACOS3-64 revision 1.08 • New product option with expanded storage capacity of 64 Kbyte
July 2008	ACOS3-24 revision 1.09 ACOS3-64 revision 1.10 • Enhancement added for ATR communication support • Added get card version information in *GetCardInfo* function
June 2009	ACOS3 revision 1.16 • New product options with user storage capacity of 32 and 72 Kbyte • Allows for 64 files to be stored • Uses FIPS 140-2 compliant Hardware Random Number Generator • Issuing START SESSION clears authentication state for enhanced security
Nov 2010	ACOS3 revision 1.17 • Contact and contactless dual interface ACOS3 version • 8 Kilobyte user storage capacity • Backward compatible except ATR/ATS and its customization
June 2014	ACOS3 revision 1.25 • Modified ATS for quicker response during tearing

2.0. Card Management

This section outlines the card level features and management functions.

2.1. Card Life Cycle States

During the whole life cycle of the chip-card, three phases and two different operating modes can be distinguished:

- Manufacturing State
- Personalization State
- User State
- User State - Issuer Mode

The card is at any moment in one of these four states. The following diagram shows the possible transitions between the four states:

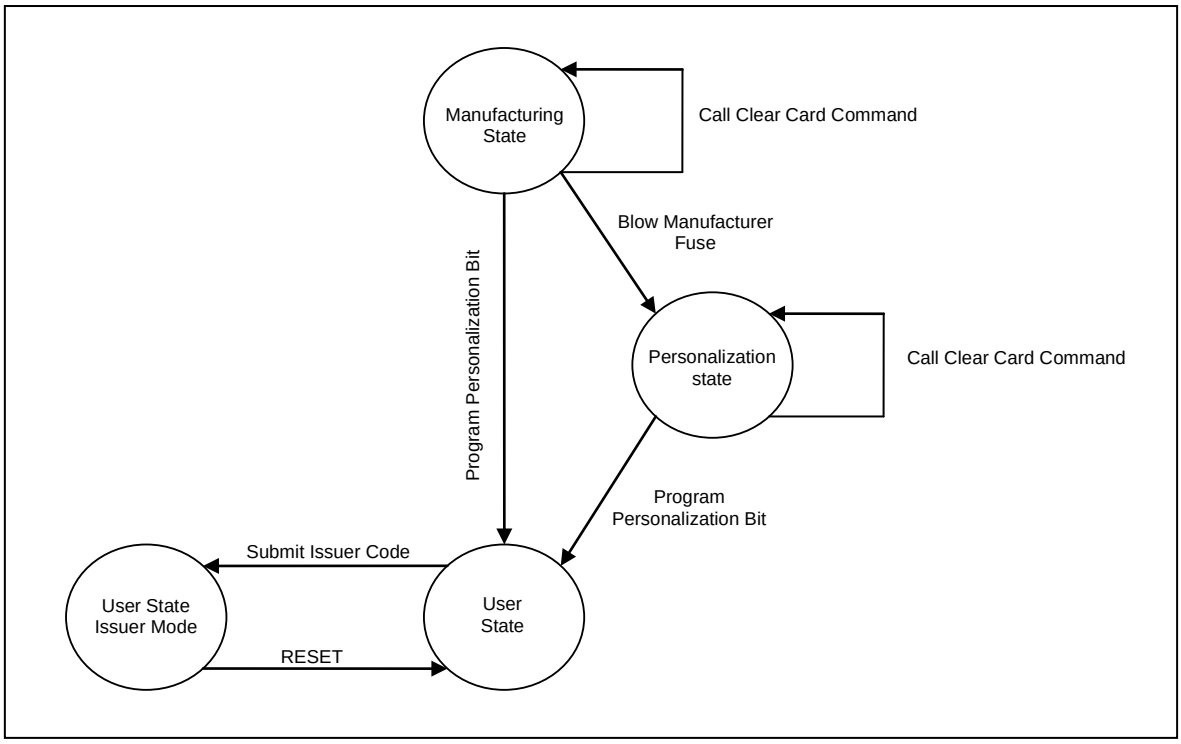


Figure 1: Card Life Cycle Diagram

The actual chip life cycle state is determined by the card operating system immediately after a reset. The life cycle state does not change during the operation of the card. CLEAR CARD command can be issued in the personalization state and manufacture state to clear the card of all data except the manufacturer data and IC code. However, this command will not be able to be used after user state.

2.1.1. Manufacturing State

The **Manufacturing State** is effective from the moment of chip manufacturing until an associated fuse (i.e., a certain bit in the EEPROM), the *Manufacturer Fuse*, has been programmed.

The IC is presented to the card in plain, without encryption.

All commands are available in manufacturer state. In addition, the *Manufacturer File* (FF01h) can only be written in this state.

The manufacturer file contains two (2) records with 8 bytes each associated to the manufacturing state. In this file, it contains the Manufacturer Fuse. After programming the Manufacturer Fuse, the card enters the Personalization State and the Manufacturer File is read-only. Data unique to each card and common card data can be programmed, such as, card manufacturer identification, card serial number, etc. The card does not interpret the data.

In this state, the card's data and keys can be erased by calling the CLEAR CARD command. This command will physically erase the EEPROM memory except for the IC code and manufacturer file.

Once the manufacturer fuse has been blown and the manufacturing state has thus been terminated, there is no possibility of resetting the card back into the manufacturing state.

2.1.2. Personalization State

Personalization State is effective from the moment of termination of the manufacturing state until an associated bit in the EEPROM; the so-called *Personalization Bit* has been programmed.

In this state, the card's data and keys can be erased by calling the CLEAR CARD command. This command will physically erase the EEPROM memory except for the IC code and manufacturer file. Re-personalization of the card is possible.

In the Personalization State, any write access to Internal Data Files, as well as the read access to the Security File is only possible after the presentation of the correct IC code. The card manufacturer writes the IC code in the Manufacturing State.

The IC is presented to the card in plain, without encryption. The authentication process should not be executed prior to programming the correct keys in the Personalization State.

The following data items are written to the memory in the Personalization State:

- The Personalization File, containing three (3) records with 4 bytes each associated to the Personalization State, including the Option Registers. This area can only be written in the Personalization State. After programming the Personalization Fuse, the Personalization File is read-only. Data unique to each card and common card data can be programmed in the Personalization File, such as, card issuer identification, card application code, etc. The first 2 records of the Personalization File are transmitted in the Historical Bytes in the Answer-to-Reset.
- Secret Codes and Keys.
- File Definition Blocks of the required User Data Files.
- Account Data Structure (if enabled by the respective option bit).
- Personalization Bit to change the card life cycle from the Personalization State to the User State.

Once the Personalization Bit has been programmed and the Personalization State has been terminated, it is now impossible to reset a card back to its Personalization State.

2.1.3. User State

User State designates the normal operating mode of the card. There are two types of User States – the User State and the User State - Issuer Mode. The User State is effective from the moment of termination of the personalization state. Most card holder operation should occur in this state.

A submission of the Issuer Code changes the operation mode to Issuer Mode. This privileged card allows access to certain memory areas, which are otherwise not accessible.

2.2. Answer To Reset (ATR)

Contact ATR is in compliance with the standard ISO 7816 Part 3. ACOS3 supports the protocol type T=0. The protocol type selection function is not implemented.

The direct convention is used for the coding of the bits in the communication with the card, i.e., logic level ONE corresponds to the Z state of the I/O line.

Fourteen (14) bytes of data are transmitted in the historical bytes as described below.

The following data are transmitted in the ATR:

TS	T0	TA_1	TB_1	TD_1	14 Historical Bytes
3Bh	BEh	11h	00h	00h	

The 14 bytes string transmitted in the historical bytes is composed as follows:

T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14
41h	01h	25h	00h	00h	00h	00h	00h	00h	00h	00h	00h	90h	00h

2.3. Answer To Select (ATS)

Contactless ATS is in compliance with the standard ISO 14443 Part 4. The following data are transmitted in the ATS:

TL	T0	TA_1	TB_1	TC_1	3 Historical Bytes
08h	78h	33h	B5h	02h	

The three (3) bytes string transmitted in the historical bytes is composed as follows:

T1	T2	T3
41h	01h	25h

2.4. Customizing the ATR/ATS

ATR/ATS can be customized with custom TA1 values and historical bytes. TA1 byte can be customized to increase the baud rate of the card. Historical bytes can also be customized to have unique application identifiers.

Due to difference in firmware architecture of ACOS3 combi/contactless to the ACOS3 contact IC, TA1 value and historical byte must be modified at the ACS production facilities. Please contact your ACS representatives during ordering for custom TA1 and Historical bytes values.

2.4.1. Customize ATR TA1 value

The contact protocol can support the following baud rates:

Valid TA1 Values	Description
	The following are the reference baud rates*:
16h	- 307,200 bps
96h	- 223,200 bps
18h	- 115,200 bps
95h	- 111,600 bps
94h	- 55,800 bps
93h	- 27,900 bps
92h	- 13,950 bps
11h	- 9,600 bps (default for backward compatibility reasons)

Table 1: ACOS3 Interpretation of FF 07h

** Based on an external clock frequency of 3.5712 MHz.*

** Please see ISO 7816 Part 3 for more information about ATR formation.*

Please note that even though ACOS3 combi card can support up to 307,200 bps on the contact interface, the corresponding smart card reader must also support such rates and supports it stably. Therefore, it is imperative that the solutions provider get sample cards of those TA1 values in the card and perform testing with their readers before mass order of an increased TA1 value. For compatibility reasons, the default TA1 value is set to 11h.

2.4.2. Customized ATS TA1 value

The contactless protocol currently has TA1 = 33h as its bit rate capability. This means the card supports 106, 212, 424 kbps for both directions from PICC to PCD and vice versa. This is stated in ISO 14443 Part 4 Section 5.2.4. The ACOS3 combi card can support up to 848 kbps by setting TA1 value to 77h.

Similar to contact TA1 customization, solution provider should ensure that the baud rate works with all their existing contactless smart card readers (PCDs) before a volume order of ACOS3 combi cards. Please contact your ACS representatives for more information.

2.4.3. Customized ATR/ATS for Microsoft Windows Usage

For Windows® 7 and above operating systems: Windows automatically attempts to download the smart card's minidriver whenever a smart card is inserted into the smart card reader. Since ACOS3 is not intended to conform to Windows default usage, such smart card minidriver is not necessary. However, if the ACOS3 is inserted into a Windows system, Windows may search online for the driver and may give a warning that the "device driver was not successfully installed" for the smart card. There are two ways in solving this issue:

1. Disable smart card plug and play and certificate propagation in Windows.
2. Change the ATR so Windows will recognize the ACOS3 smart card to use ACS's Unified Null Driver.

For the first solution, please follow instructions in this Microsoft® support link to disable smart card plug and play. This may have to be done for every computer that will be used in this system. http://support.microsoft.com/kb/976832.

For the second solution, ACS has developed a Unified Null driver for ACOS line of smart cards. The Unified Null driver will satisfy the Windows requirement to have a minidriver for the card, hence the warning from Windows every time the card is inserted will now be removed. The Unified Null Driver can be downloaded automatically from Windows Update. In order for Windows to recognize the ACOS3 Combi smart card to use the Unified Null Driver, the ATR must be customized. The ATR/ATS customization will be done by ACS. Please contact your ACS representative for more information.

In the case of ACOS3 Combi, the ATR and ATS should be:

ATR: 3B BE XX 00 00 41 43 53 5F 41 43 4F 53 33 5F 4E 44 90 00h

ATS: 08 78 XX B5 02 33 4e 44h

The XX is the value of TA1. The TA1 value can be set to the baud rate that the smart card reader used can support. For ACS ACR38 for example, the ATR value of TA1 can be set to 95h and the ATS TA1 value will be 33h.

After the customization, these will be the customized values of ATR and ATS:

ATR: 3B BE 95 00 00 41 43 53 5F 41 43 4F 53 33 5F 4E 44 90 00h

ATS: 08 78 33 B5 02 33 4e 44h

3.0. EEPROM Memory Management

The user EEPROM memory area provided by the card chip is fully-usable for user data storage. There is an additional EEPROM area, which stores internal card configuration data.

- The User Data Memory stores the data of the card under the control of the application.
- The internal card configuration data is used by the card operating system to manage card functionalities.

3.1. Data Files

Access to both the Internal Data Memory area and the User Data Memory area is possible within the scopes of data files and data records. Data files in the Internal Data Memory are referred to as *Internal Data Files*. Data files in the User Data Memory are called *User Data Files*.

Data files are the smallest entity to which individual security attributes can be assigned to control the read and write access to the data stored in the EEPROM.

Data files are of either record type or transparent type.

A record data file contains N data records. The record number must be specified when a record (or data within a record) is read from or written to a file. A data file can contain up to 255 records. The record length can be different for different files but is always fixed within a file. Access to record files is done by issuing READ or WRITE RECORD.

A transparent data file contains a continuous block of EEPROM and it is accessed by providing a length and offset to be read or written within the block. It is also referred to as a binary file. The commands to access this file type are READ or WRITE BINARY.

Internal Data files are of RECORD type only. The file structures of the Internal Data Files (file size, file identifier, record length, security attributes) are defined by the operating system and cannot be changed. The file structure for the User Data Memory is determined in the card personalization. After programming the parameter N_OF_FILE in the Personalization State, the file structure is fixed.

Access to all files is possible only through the READ RECORD/BINARY and WRITE RECORD/BINARY commands. The operating system does not keep track of which records were actually written through the WRITE RECORD/BINARY command. The data returned by the card in response to a READ RECORD/BINARY command are the actual data read from the EEPROM memory, regardless of whether that data were ever been written.

Each file is identified by two bytes File Identifier. The File Identifier is assigned to the file when the file is being defined during the Personalization State. The operating system does not perform any checking on the uniqueness of each File Identifier. If the same identifier has been assigned to more than one file, a malfunction of the card may occur.

Note: *A value of FFh of the first byte of the file identifier is used for Internal Data Files and cannot be used for User Data Files.*

Before any READ or WRITE RECORD/BINARY access to a file, the file must be opened through the SELECT FILE command. Only one file is selected at any time. The READ and WRITE RECORD/ BINARY commands refer to the most recently selected file.

3.2. Data File Access Control

Two security attributes are assigned to each Data File: the Read Security Attribute and the Write Security Attribute. Security attributes define the security conditions that must be fulfilled to allow the respective operation:

A Security Attribute is defined in one byte as follows:

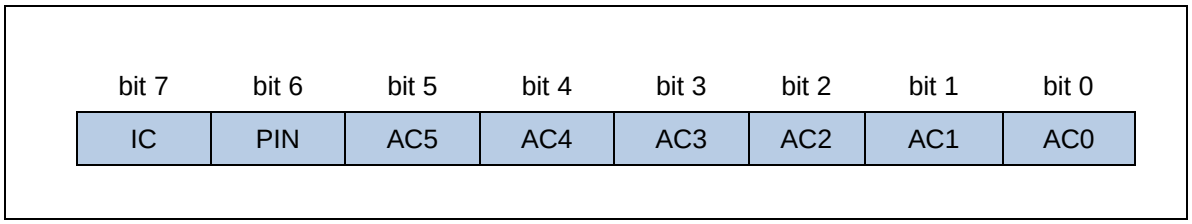


Figure 2: Security Attribute

Each bit of the byte represents a code. If the bit is set to '1', the corresponding code must have been submitted. If the bit is set to'0', the corresponding code is irrelevant for the access condition. Please see the **Section 3.3.4** for the definition of the specific codes.

- The Read Security Attribute controls the read access to the data in a file through the READ RECORD/BINARY command. If the security condition specified in the Read Security Attribute is not fulfilled, the card will reject a READ command to that file.
- The Write Security Attribute controls the write access to the data in a file through the WRITE RECORD/BINARY command. If the security condition specified in the Write Security Attribute is not fulfilled, the card will reject a WRITE command to that file.

The Read Security Attribute and the Write Security Attribute for each data file specify which Application Code, if any, must have been submitted correctly to the card to allow the respective operation, and whether the Issuer Code and/or the PIN code must have been submitted.

A logical OR function applies to the specified Application Codes, AC x, i.e., if more than one Application Code is specified in a security attribute, the security condition is fulfilled if any one of the specified Application Codes has been correctly submitted.

A logical AND function applies to the PIN and the IC code, i.e., if PIN and/or IC are specified in a security attribute, the PIN and/or IC code(s) must have been submitted in addition to the specified Application Codes(s).

Application Code AC0 can be specified in the Security Attribute, but cannot be submitted to the card. It is thus possible, for example, to completely write protect a file by specifying AC0 in the Write Security Attribute of that file.

For Internal Data Files, the security attributes are fixed in the card operating system. For User Data Files, the security attributes of a file are stored in the associated File Definition Block.

The following table lists examples of security conditions that can be specified for User Data Files:

Security Attribute	Security Condition
-	No restriction; free access
AC x	Access only after correct submission of AC x
AC x, AC y, AC z	Access only after correct submission of AC x **or** AC y **or** AC z
IC	Access only after submission of IC
PIN	Access only after submission of PIN
PIN, IC	Access only after submission of PIN **and** IC
AC x, IC	Access only after submission of AC x **and** IC
AC x, PIN, IC	Access only after submission of AC x, **and** PIN **and** IC
AC x, AC y, PIN	Access only after correct submission of AC x **or** AC y, **and** PIN
AC0	No access

Table 2: Sample Security Conditions

AC x requires Application Code x.

PIN requires PIN code.

IC requires Issuer Code.

3.3. Internal Data Files

With exception of the Account Data Structure, which has associated a special set of commands, the memory areas of the Internal Data Memory are processed as data files.

The attributes of the Internal Data Files are defined in the card operating system and cannot be changed. However, the security attributes depend on the card life cycle state.

The following Internal Data Files are defined:

		File Security Attributes			Record Organization
Memory Area	**Internal File ID**	**Manufacturing State**	**Personalization State**	**User State**	
MCU-ID File	FF 00h	R: FREE W: NO ACCESS	R: FREE W: NO ACCESS	R: FREE W: NO ACCESS	2 x 8 bytes
Manufacturer File	FF 01h	R: FREE W: IC	R: FREE W: NO ACCESS	R: FREE W: NO ACCESS	2 x 8 bytes
Personalization File	FF 02h	R: FREE W: IC	R: FREE W: IC	R: FREE W: NO ACCESS	3 x 4 bytes
Security File	FF 03h	R: IC W: IC	R: IC W: IC	R: NO ACCESS W: IC	14 x 8 bytes
User File Management File	FF 04h	R: FREE W: IC	R: FREE W: IC	R: FREE W: IC	N_OF_FILE x 7 bytes
Account File	FF 05h	R: FREE W: IC	R: FREE W: IC	R: IC W: IC	8 x 4 bytes
Account Security File	FF 06h	R: FREE W: IC	R: FREE W: IC	R: NO ACCESS W: IC	4 x 8 bytes
User File Data Area	File IDs: xx yyh xx ≠ FFh	According to the file definitions			

Table 3: Internal Data Files Attributes

3.3.1. MCU ID File

The MCU ID File contains two records of eight bytes each. The contents of this file are determined during the chip manufacturing process and cannot be altered.

The first record contains an 8 byte unique serial number of the chip. The second record contains ACOS3's revision number– namely ACOS3 Revision XX.YY ZZh (41 43 4F 53 03 XX YY ZZh).

XXh is the major version

YYh is the minor version

ZZh is the user EEPROM capacity in kilobytes

This file is always free for READ access but not WRITE accessible.

3.3.2. Manufacturer File

The Manufacturer File comprises of two records of eight bytes each that are written in the Manufacturing State of the card life cycle. After termination of the Manufacturing State, this file is read-only and free for read access.

The termination of the Manufacturer State is indicated by writing a '1' into the MSB of byte 1 of the first record in the Manufacturer File (Manufacturer Fuse). After the next reset of the card, the Manufacturing State can never again be entered.

Manufacturer File, first record:

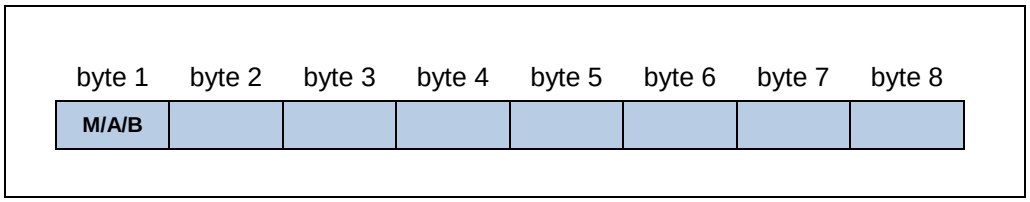


Figure 3: Manufacturer File

M (bit7) = Manufacturer Fuse

A (bit6) = INQ_ACC_MAC Flag

B (bit5) = RECORD_NUMBERING Flag

Only the bits in M, A, B are interpreted by the operating system.

INQ_ACC_MAC flag affects the INQUIRY ACCOUNT command only. A one in this flag makes the composition of the MAC calculation including the credit and debit transaction reference. For more details, please refer to **Section 6.13**.

RECORD_NUMBERING flag affects the record numbering system of the whole card. This flag when one indicates that the records are numbered from 1 to N, a zero in this flag indicates that the records are numbered from 0 to N-1 (where N is the number of records in the file).

3.3.3. Personalization File

Personalization File comprises of 12 bytes, arranged as 3 records of 4 bytes each.

The Personalization File is written during the Personalization State of the card life cycle. After termination of the Personalization State, this file is read-only and free for READ access.

The termination of the Personalization State is indicated by writing a '1' into the MSB of byte 4 of the first record in the Personalization File (Personalization Bit). The change of state will be effective immediately after the next reset of the card.

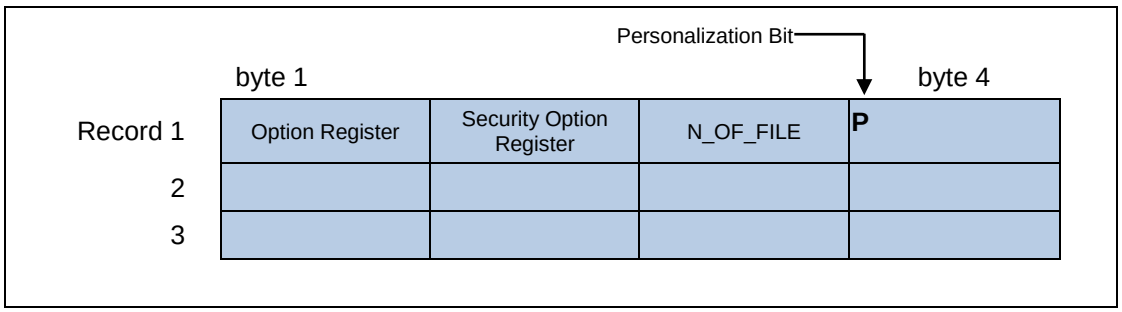


Figure 4: Personalization File

The first three (3) bytes of the first record of the Personalization File are used to set certain parameters and to enable/disable optional features of the card operating system. The following subsections describe each of the 4 bytes:

3.3.3.1. Option Register

Byte 1 is the *Option Register* and contains the following option bits:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
INQ_AUT	TRNS_AUT	REV_DEB	DEB_PIN	DEB_MAC	PIN_ALT	3DES	ACCOUNT

Figure 5: Option Register

ACCOUNT This bit indicates if the Account Data Structure is available in the card. If the bit is not set, indicating that the Account Data Structure is not present, the memory space required for storing the Account Data Structure and the associated security data is available for User Data Files and the Account processing commands cannot be executed.

3DES This bit indicates if the encryption uses DES or 3DES. If the bit is not set, single DES is used. Otherwise, triple DES is used for authentication and secure messaging purposes.

PIN_ALT This bit determines if the PIN code can be changed through the CHANGE PIN command. If the bit is set, the PIN code can be changed after it has successfully been submitted to the card.

DEB_MAC This bit indicates if the debit transaction must be authenticated by a MAC cryptographic checksum (see **Section 4.5.2**). If the bit is not set, the card does not evaluate the data transmitted as MAC checksum in the DEBIT command.

DEB_PIN This bit indicates if the PIN code must be submitted for the DEBIT command. If the bit is set, the DEBIT command is only carried out after the PIN code has been successfully submitted to the card (see **Section 4.5.2**).

REV_DEB This bit determines if the card can execute the REVOKE DEBIT command. If the bit is not set, the card will reject the REVOKE DEBIT command (see **Section 4.5.3**).

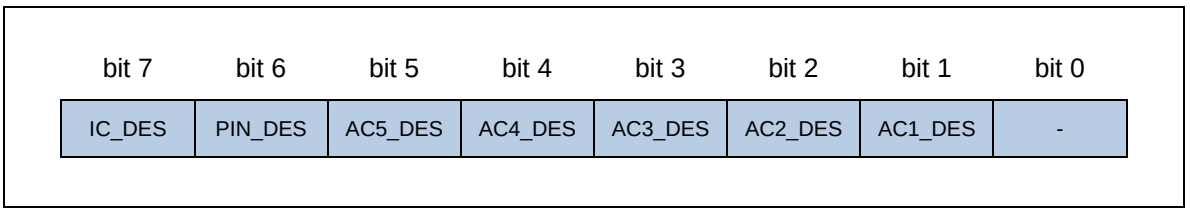
TRNS_AUT This bit determines if the Account Transaction processing requires the previous completion of the mutual authentication process, and the use of the current Session Key in the computation of the MAC cryptographic checksums. If the bit is set, the mutual authentication must have been executed prior to any ACCOUNT TRANSACTION command and the MAC cryptographic checksum must be DES encrypted with the current session key before it is sent to the card.

INQ_AUT This bit determines whether the INQUIRE ACCOUNT command requires the previous completion of the mutual authentication process, and the use of the current Session Key in the computation of the MAC cryptographic checksum returned by the card in response to this command. If the bit is set, the mutual authentication must have been executed prior to the execution of the INQUIRE ACCOUNT command and the MAC cryptographic checksum is DES encrypted with the current session key before it is returned by the card.

Note: *By enabling the options controlled by the bits TRNS_AUT and INQ_AUT, a Unique Key per Transaction scheme can be used with the Account transaction processing. This provides a very high security level.*

3.3.3.2. Security Option Register

Byte 2 is the *Security Option Register* and contains seven option bits:



bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
IC_DES	PIN_DES	AC5_DES	AC4_DES	AC3_DES	AC2_DES	AC1_DES	-

Figure 6: Security Option Register

These bits specify the corresponding Secret Codes (IC, PIN, AC1...AC5), if the codes are presented to the card in plain or encrypted. If a bit is set to '1', the corresponding code submitted in the SUBMIT CODE command must be encrypted with the current session key before it is presented to the card. This means, the Mutual Authentication as described later in this document must have been completed.

If a bit is set to '0', the corresponding code is submitted in plain without encryption.

The bit PIN_DES also determines whether encryption is used with the CHANGE PIN command. If the bit is set, the new PIN code must be encrypted with the current session key before it is submitted in the CHANGE PIN command.

Notes:

1. *For security reasons, it is highly recommended that in any application the IC must be submitted in encrypted form in the User State.*
2. *The Option Register and the Security Option Register are evaluated by the ACOS3 operating system only after a card reset. After changing any option bit during the card personalization, a card reset must be performed in order for the change to take effect.*

3.3.3.3. Number of Files

The third byte is the N_OF_FILE setting. This value specifies the number of data files allocated in the File Data Area. The card operating system expects that accordingly N_OF_FILE File Definition Blocks have been written as records in the User File Management File (FF04h). The maximum number of files allowed in ACOS3 is 64. The operating system will not limit the value that is put in. However, only 64 records will be created in User File Management File (FF04h).

3.3.3.4. Personalization Bit

Byte 4 only has one significant bit - the *Personalization Bit*. Setting this bit will change the card life cycle from Manufacturer State or Personalization State to User State.

Only the first 4 bytes of the Personalization File are interpreted by the card operating system.

The first eight (8) bytes (2 records) of the Personalization File are transmitted in the Historical Bytes in the Answer-to-Reset. The last record of 4 bytes can be written with personalization information and not shown in the ATR.

3.3.4. Security File

The Security File stores the following information:

- The key pair used for card authentication.
- The five Application Codes used for specific file access control.
- The Issuer Code IC.
- The PIN code.

- Error counters for limiting the number of unsuccessful code presentations and authentication.
- The seed for the random number generator.

The Security File can only be read during the Manufacturing State and the Personalization State of the card life cycle, after presentation of the correct IC.

Note: *After termination of the Personalization State, there is NO possibility to read the Security File.*

The Security File can be written in the Manufacturing State and in the Personalization State after presentation of the correct IC, and in the Issuer Mode of the User State.

The Security File comprises of 14 records, 8 bytes in length each and is organized as follows:

	byte 1								byte 8
Record 1	Issuer Code IC								
2	PIN								
3	Authentication Card Key K_C								
4	Authentication Terminal Key K_T								
5	Reserved For Future Use.								
6	Application Code AC1								
7	Application Code AC2								
8	Application Code AC3								
9	Application Code AC4								
10	Application Code AC5								
11	CNT AC1		CNT AC3	CNT AC2	CNT AC5	CNT AC4	CNT IC	CNT PIN		CNT K_T			CNT K_{rd}	CNT K_d		CNT K_{cr}
12	SCN AC1		SCN AC3	SCN AC2	SCN AC5	SCN AC4	SCN IC	SCN PIN		SCN K_T			SCN K_{rd}	SCN K_d		SCN K_{cr}
13	Right half of 3DES Authentication Card Key K_C								
14	Right half of 3DES Authentication Terminal Key K_T								

Figure 7: Security File

CNT xxx = Counter for the successive submission of wrong key/code xxx.

SCN xxx = Starting CNT value (see **Section 4.3.4** for more information).

Notes:

1. *Caution must be taken when writing record number 11 and 12 during card personalization (especially for the IC). Inadvertently writing a wrong value to this record may permanently lock the codes and render it useless.*
2. *Record number 12 is available in ACOS3 version 1.04 or above.*
3. *Record numbers 13 and 14 are available in ACOS1 version 3.0 or above only (It is available in ACOS3). The right half of the authentication keys is stored here. When single DES option is selected, these records are not used but present.*

3.3.5. User File Management File

The User File Management File consists of N_OF_FILE records of 7 bytes each and stores a File Definition Block for an allocated User Data File in each record.

The File Definition Blocks are written during the Personalization State of the card life cycle. After termination of the Personalization State, this file is free for read access and can be written after the Issuer Code has been submitted.

The sequence of File Definition Blocks in the User File Management Area is not relevant. When the SELECT FILE command is issued, the card operating system searches all File Definition Blocks for one whose File Identifier entry matches the value specified in the SELECT FILE command.

The Card Operating System does not provide any error checking on the File Definition Blocks nor does it check the consistency of the number of file definition blocks written with the parameter N_OF_FILE. Any inconsistency of these data can lead to a malfunction of the card.

3.3.6. User File Data Area

The User File Data Area stores the data written to the User Data Files. Security attributes are attached to User Data Files, which control the access to the data in the files.

User Data Files cannot be deleted. Once allocated, the memory space for a User Data File is reserved and cannot be released when the file is no longer used.

3.3.7. Account File

Account File stores the Account Data Structure used for highly secure payment applications.

If the option bit ACCOUNT in the option registers is not set, this file is not processed by the card operating system but the allocated memory for the account files is reserved and therefore cannot be allocated for User Data Files.

The Account File can be written during the Manufacturing and Personalization State of the card life cycle after presentation of the correct IC code. After Termination of the Personalization State, this file can be written after the Issuer Code has been submitted.

The Account File contents are explained in detail in **Section 3.6**.

3.3.8. Account Security File

The Account Security File stores the four secret keys used for the calculation of the MAC cryptographic checksums used in connection with the Account processing commands.

The Account Security File can only be read during the Manufacturing State and the Personalization State of the card life cycle.

Note: *After termination of the Personalization State, there is NO possibility to read the Account Security File.*

The Account Security File can be written in the Manufacturing State and in the Personalization State after presentation of the correct IC code, and in the Issuer Mode.

If the option bit ACCOUNT in the option registers is not set, this file is not processed by the card operating system and the memory space is available for storage User Data Files.

The Account Security File contents are explained in detail in **Section 3.6.1**.

3.4. User Data Files

User Data Files are allocated in the Personalization State of the card life cycle. There are two types of User Data Files, Record and Binary files. Record files are specified by number of records and a fixed record length. Binary files are specified by a file size and accessed via offsetting into the file.

The data stored in a User Data File can be read through the READ RECORD/BINARY command and updated through the WRITE RECORD/BINARY command when the security conditions associated to the data file are fulfilled.

User Data Files are defined by writing the corresponding File Definition Blocks in the records of the User File Management File during the Personalization State. It is not possible to change the number of records of a file once any of the User Data Files has been used. User will be able to access these data as long as it's within the capacity of the card.

A User Data Record File can contain up to 255 records of maximum of 255 bytes record length each. A User Data Binary File can have a specified size up to the length of the card.

Note: *Card issuer must be careful to assure that the memory space allocated for all User Data Files does not exceed the available memory space.*

ACOS3 does not check the available memory space at the time of allocation. Writing and reading beyond the capacity of the card will be rejected by the card.

3.4.1. User File Definition Block

Each User Data File is described in an associated File Definition Block which contains the file identifier, record length/number (or file length) and security attributes and flags. The security attribute bytes are specified in **Section 3.2**. Each File Definition Block comprises 7 bytes:

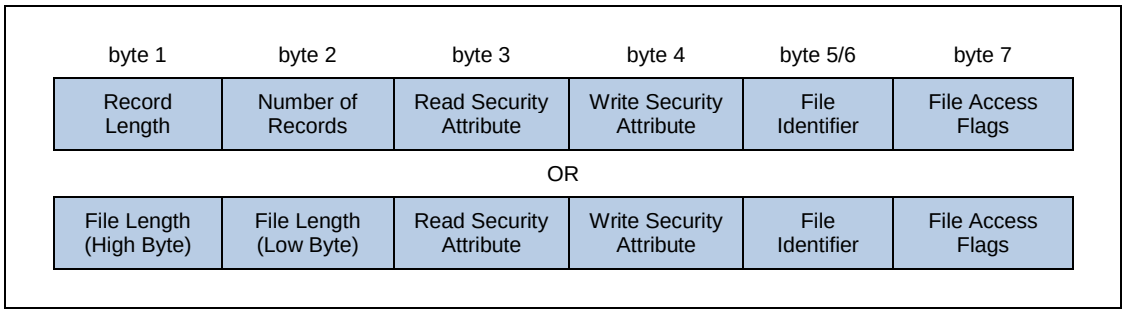


Figure 8: File Definition Block

The two types of User Data File correspond to the Record and Binary file types respectively and it is distinguished by the File Access Flag field described below.

The File Definition Blocks of all files are stored in the User File Management File. They can be read through READ RECORD commands after selection the User File Management File with the SELECT FILE command.

The number of records in the User File Management File is given by the value of the parameter N_OF_FILE in the option register.

The File Access Flag contains the following fields:

b7	b6	b5	b4	b3	b2	b1	b0	Meaning
0								Record File type – byte 1 and 2 of User File Definition Block specifies Record Length and Record Number respectively
1								Binary File type – byte 1 and 2 of User File Definition Block specifies File Length
	1							Read requires Secure Messaging
		1						Write requires Secure Messaging
			0	0	0	0	0	Reserved for future use

Table 4: File Access Flags

On default, File Access Flags will be 00h which will correspond to record file type and no secure messaging required. This will be backward compatible to previous versions of ACOS1/2/3.

3.4.2. User File Allocation

For the allocation of User Data Files in a new card, follow the steps as listed below. It is assumed that the IC has been presented to the card prior to this operation such that the Internal Data Files can be written.

1. Use the SELECT FILE command with file ID = FF 02h to select the Personalization File.
2. Write the number of User Data Files required to the option register N_OF_FILE, which is the third byte of the first record of the Personalization File, to allocate the required space (number of records) in the User File Management File.
3. Use the SELECT FILE command with file ID = FF 04h to select the User File Management File.
4. Write the N_OF_FILE file definition blocks to the User File Management File with the WRITE RECORD command. Write the seven bytes of each File Definition Block at once.
5. Now the User Data Files can be selected and read and written.

3.5. Data File Access

The process of data file access is identical for Internal Data Files and for User Data Files.

3.5.1. SELECT FILE

The SELECT FILE command can be executed any time. The specified file - if existent - will be selected and the previously selected file - if any - will be closed. If the specified file does not exist, the card returns an error code and does not change the status of a currently selected file. The security conditions specified for the newly selected file are not checked in the SELECT FILE processing and the Mutual Authentication need not be completed prior to the execution of the SELECT FILE command. After a card reset, no file is selected.

The SELECT FILE command is carried out as follows:

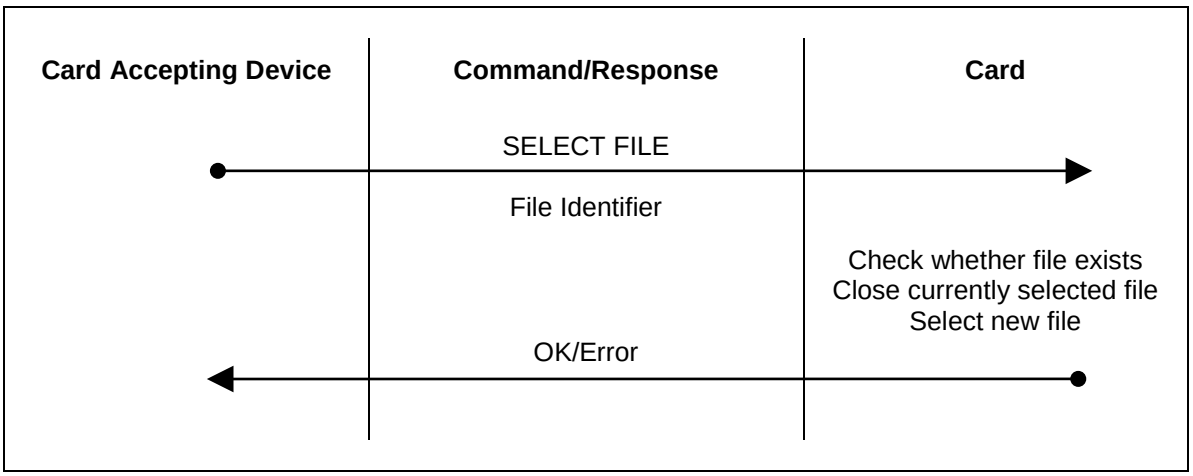


Figure 9: SELECT FILE Command

File Identifier Two bytes file identifier of the file to be selected.

3.5.2. READ RECORD/BINARY

READ RECORD/BINARY command can be executed once a file has been selected through the SELECT FILE command.

The security conditions associated to the currently selected file are checked prior to the execution of the command by the card. If the security conditions are not fulfilled (i.e., the specified secret codes have not been submitted to the card), the command is rejected by the card.

For READ RECORD, data from only one record can be read in each READ RECORD operation. The number of bytes and offset to be read is specified in the command.

For READ BINARY, the number of bytes and offset to be read is specified in the command. The maximum number of bytes to be read is equal to the record length.

If the number of bytes read (= N) is smaller than the record length, the first N bytes of the record are returned by the card.

The READ RECORD command is carried out as follows:

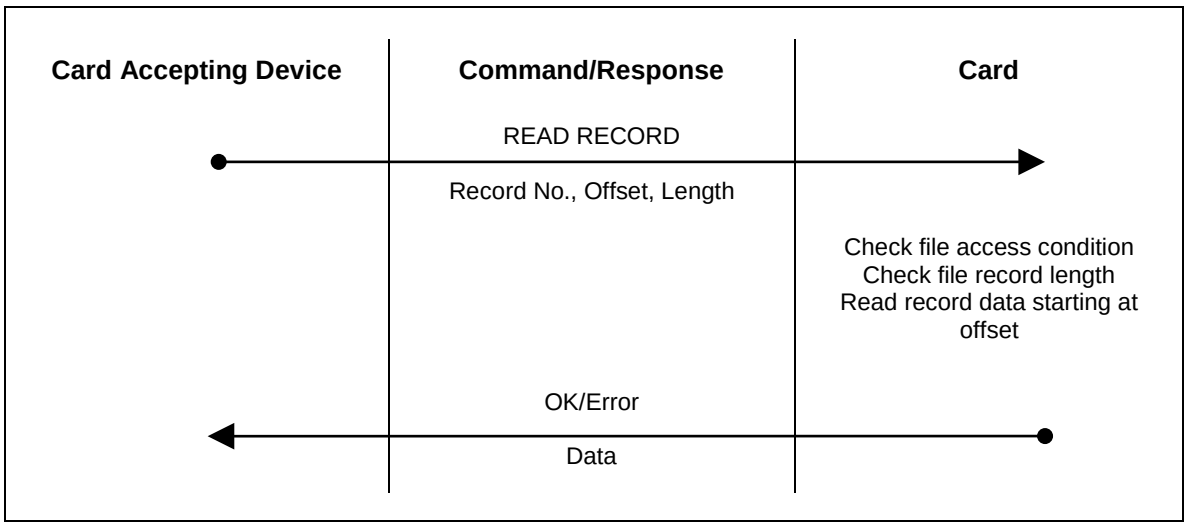


Figure 10: READ RECORD Command

Record No.	One byte logical record number.
Offset	The byte to start reading from within the record.
Length	Number of data bytes to be read from the record (max. 255).
Data	Record data, *Length* bytes.

The READ BINARY command is carried out as follows:

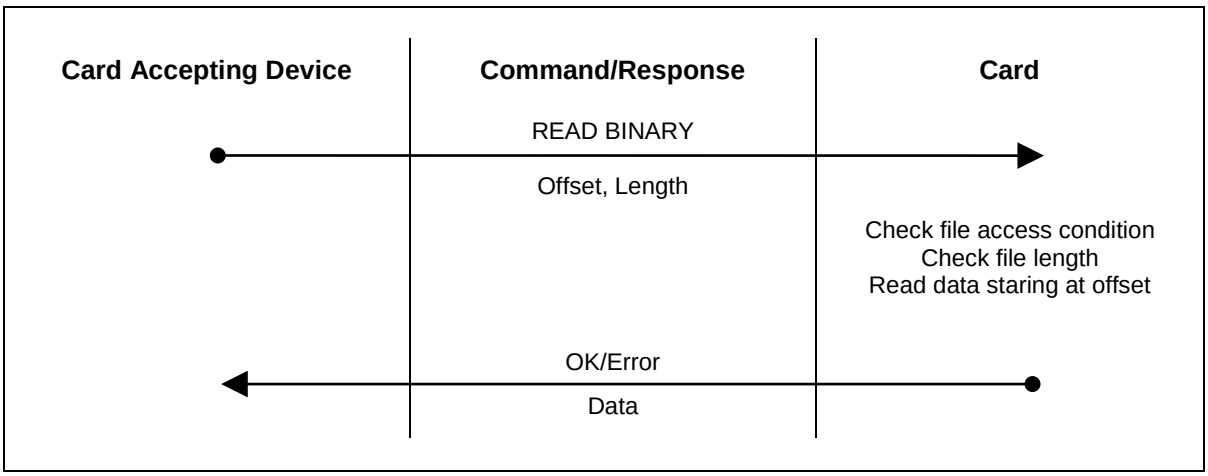


Figure 11: READ BINARY Command

Length	Number of data bytes to be read.
Offset	The byte to start reading from within the file.

Data Record data, *Length* bytes.

3.5.3. WRITE RECORD/BINARY

The WRITE RECORD command can be executed once a file has been selected through the SELECT FILE command.

The security conditions associated to the currently selected file are checked prior to the execution of the command by the card. If the security conditions are not fulfilled (i.e., the specified secret codes have not been submitted to the card), the command is rejected by the card.

For WRITE RECORD, data can be written to only one record in each WRITE RECORD operation. The number of bytes and offset to be written in the record is specified in the command.

For WRITE BINARY, the number of bytes and offset to be written is specified in the command. The maximum number of bytes to be written is equal to the record or file length.

WRITE RECORD command is carried out as follows:

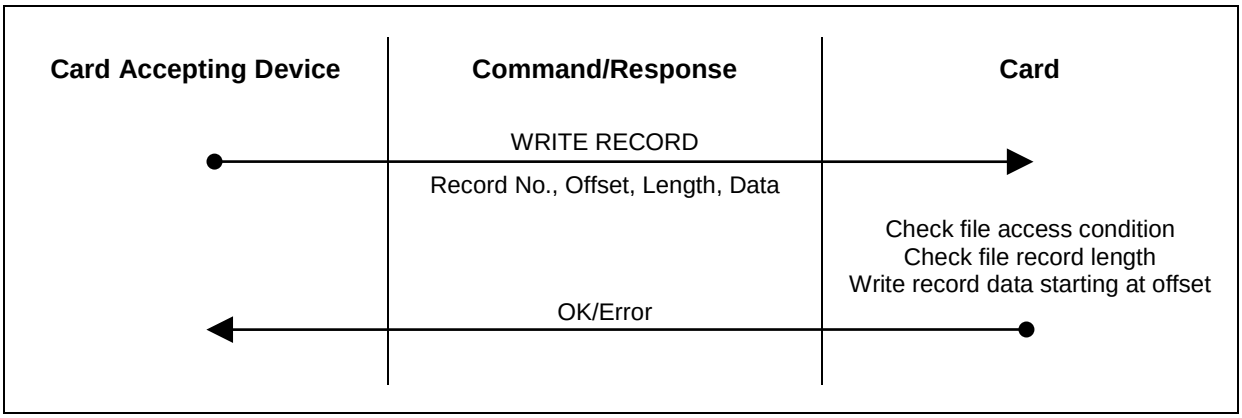


Figure 12: WRITE RECORD Command

Record No.	One byte logical record number.
Offset	Offset in the record to begin writing.
Length	Length of data to write.
Data	Data bytes of Length bytes to be written to the record.

The WRITE BINARY command is carried out as follows:

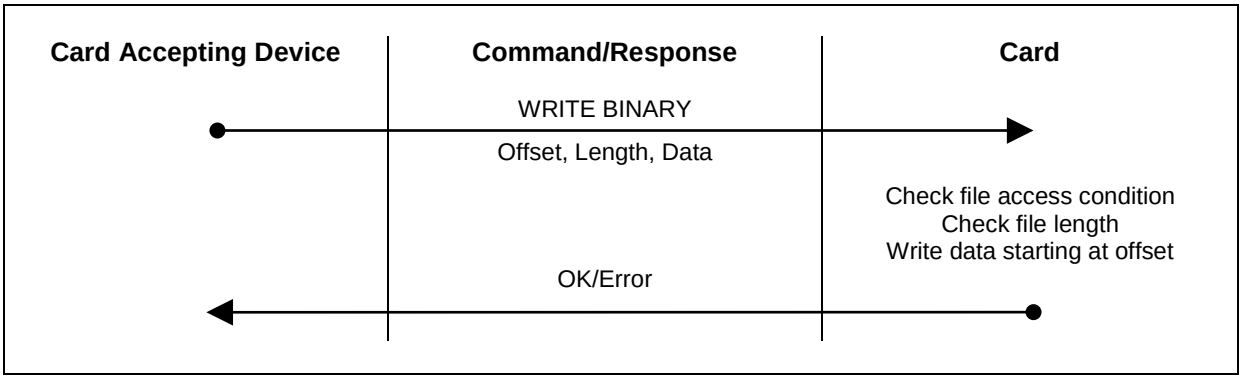


Figure 13: WRITE BINARY Command

Offset	Offset in the file to begin writing.
Length	Length of data to write.
Data	Data bytes of Length bytes to be written to the file.

3.6. Account Data Structure

The Account Data Structure - *Account*, for short - is dedicated for the use in applications in which a numeric value representing some 'amount' must be securely processed. The Account is stored in the Account File.

In the User State of the card life cycle, the data in the Account cannot be manipulated by WRITE instructions like the data in User Data Files. A set of dedicated instructions is available for the processing of the Account, i.e. for adding value to and subtracting value from the balance in the Account and for reading the current balance.

Different access conditions can be specified for adding to, subtracting from and reading the Account.

Critical Account operations, for example, CREDIT, are carried out under strict security control conditions, as explained below in 'Account Transaction Processing'.

Account Data Structure in the Account File has the following form:

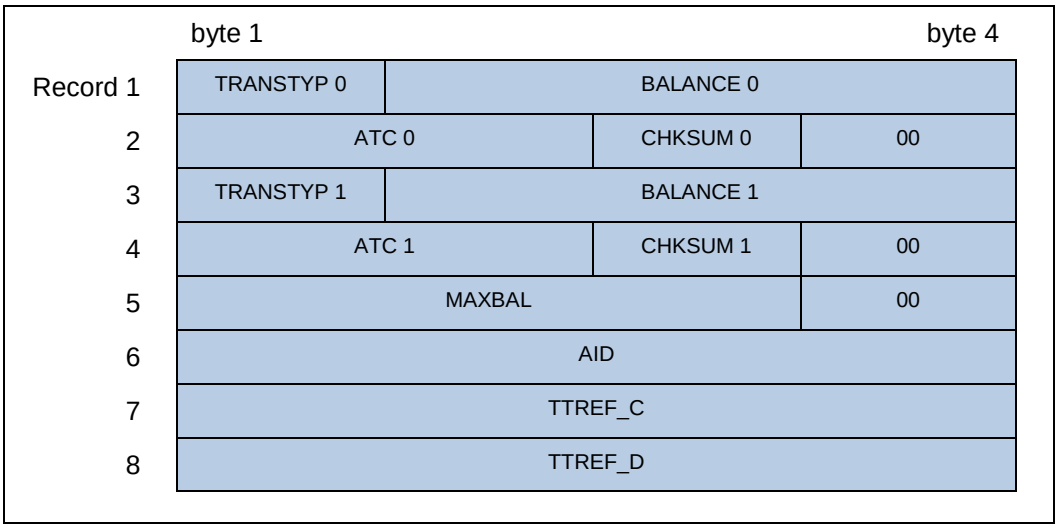


Figure 14: Account Data Structure

TRANSTYP	Together with the balance values is stored the type of transaction that resulted in that balance value. This information is updated when the balance value is updated. The following transaction types are distinguished: CREDIT, DEBIT, and REVOKE DEBIT.
BALANCE	Balance value is three bytes long, can store a value of up to 16.8 Million (2^{24-1}). Only positive integer values are possible for the Balance.
ATC	The Account Transaction Counter is incremented before each transaction to give a unique electronic signature for each transaction. Together with the Account ID AID, the ATC builds the Account Transaction Reference (ATREF), which is used in the calculation of MAC cryptographic checksums to certify the execution of Account related commands by the card. When ATC reaches its maximum value (FF FFh), the operating system does not allow any further transaction.
CHKSUM	The checksum is the least significant byte of the algebraic sum of the bytes of TRANSTYP, BALANCE and ATC, plus one.

MAXBAL The Maximum Balance value is checked by the operating system when a CREDIT transaction is performed. If the sum of current balance plus the amount to be credited exceeded the Maximum Balance value, the card will reject the CREDIT command.

AID The Account Identification is a four bytes value that is combined with the Account Transaction Counter (ATC) to give the six bytes ATREF:

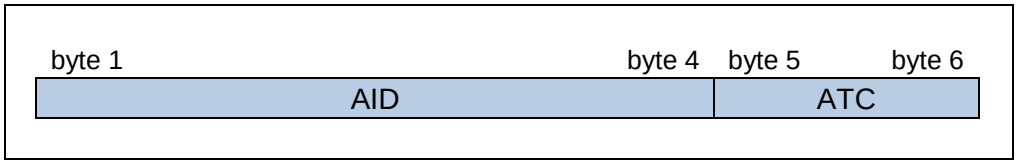


Figure 15: Account Transaction Reference

The AID is written once in the Personalization State of the card life cycle. It is never modified.

TTREF-C Terminal Transaction Reference – the card accepting device provides Credit reference when the CREDIT transaction is executed. It is only stored but not interpreted by the card. The Card Accepting Devices can evaluate this information, for example, to reject a card that has been credited by an unauthorized terminal.

TTREF-D Terminal Transaction Reference - the card accepting device provided the Debit reference when a DEBIT or REVOKE DEBIT transaction is executed. The TTREF-D is stored in the Account when a DEBIT transaction is executed. The REVOKE DEBIT command will only be executed if the TTREF-D submitted with the command is identical with the stored TTREF-D. This identity proves that the same terminal that issued the preceding DEBIT command issued the REVOKE DEBIT command.

TRANSTYP, BALANCE and ATC are stored two times to prevent a loss of this important information when a power-fail or a card reset occurs during a transaction. The larger of the two ATC values in the account indicates the data set used in the most recent transaction.

The checksum is used to verify the integrity of the data in the Account. The checksum is calculated when the account data are updated in a transaction. The checksum is verified by the card operating system before any transaction is executed.

Note: *If the checksum is found incorrect, the card allows the execution of transactions only in the Issuer Mode, i.e., after the submission of the Issuer Code IC.*

3.6.1. Account Processing Keys

The encryption keys used in the computation of MAC cryptographic checksums with the Account processing are stored as records in the Account Security File (FF06h) as follows:

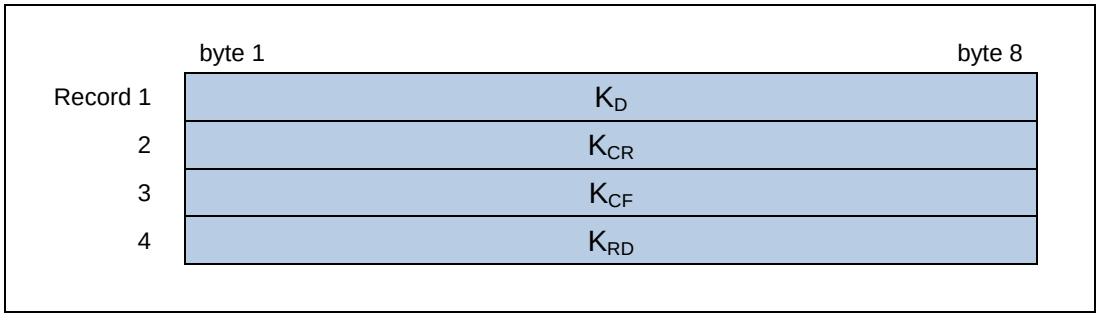


Figure 16: Key Storage for Single DES

K_D The DEBIT key used in the computation of the MAC for the DEBIT command.

K_{CR} The CREDIT key used in the computation of the MAC for the CREDIT command.

K_{CF} The CERTIFY key, used in the computation of the MAC with the INQUIRE ACCOUNT command.

K_{RD} The REVOKE DEBIT key used in the computation of the MAC for the REVOKE DEBIT command.

***Note**: Keys are 8-byte long.*

Figure 17: Key Storage for Triple DES

K_D The DEBIT key used in the computation of the MAC for the DEBIT command.

K_{CR} The CREDIT key used in the computation of the MAC for the CREDIT command.

K_{CF} The CERTIFY key used in the computation of the MAC with the INQUIRE ACCOUNT command.

K_{RD} The REVOKE DEBIT key used in the computation of the MAC for the REVOKE DEBIT command.

***Note:** Keys are 16-byte long.*

4.0. Security Features

The following security mechanisms are provided by the ACOS3 Combi card operating system:

- DES/3DES and MAC Calculation
- Mutual Authentication and Session Key Generation
- Secret Codes
- Secure Messaging for Data Files
- Account Transaction Processing
- Anti-tearing Mechanism

DES refers to the DEA algorithm for data encryption and decryption as specified in the standard ANSI X3.93. MAC refers to the algorithm for the generation of cryptographic checksums (DEA in Cipher Block Chaining mode) as specified in the standard ANSI X3.93.

Mutual Authentication is a process in which both the card and the Card Accepting Device verify that the respective counterpart is genuine. The Session Key is a result of the successful execution of the Mutual Authentication. It is used for data encryption and decryption during a 'session'. A session is defined as the time between the successful execution of a Mutual Authentication procedure and a reset of the card or the execution of another START SESSION command.

Secret Codes and the PIN code are used to selectively enable access to data stored in the card and to features and functions provided by the card, for example, the READ and WRITE commands.

Secure messaging ensures data transmitted between the card and terminal/server is secured and not susceptible to eavesdropping, replay attack and unauthorized modifications. This is achieved by signing the command and response with a MAC and encrypting command and response data.

The Account Transaction Processing provides mechanism for the secure and auditable manipulation of data in the Account Data Structure, in particular, the balance value.

4.1. DES and MAC Calculation

All keys used in DES/3DES and MAC calculation are 8/16 bytes long depending on Single/Triple DES selection in *Option Register*. The least significant bit of each byte of the key is not used in the calculation and is not interpreted by the card operating system.

4.2. Mutual Authentication and Session Key Generation

The Mutual Authentication is based on the exchange and mutual verification of secret keys between the Card and the Card Accepting Device. The key exchange is performed in a secure way by use of random numbers and DES data encryption.

ACOS3 maintains a dedicated pair of data encryption/decryption keys for the Mutual Authentication, K_T, called *Terminal Key*, and K_C, called *Card Key*.

ACOS3 also provides a generator for the random numbers used in the Mutual Authentication process, RND_C, called *Card Random Number*.

The *Session Key* is the result of the Mutual Authentication process and it is based on the random numbers of both card and terminal.

The Mutual Authentication process is carried out as follows:

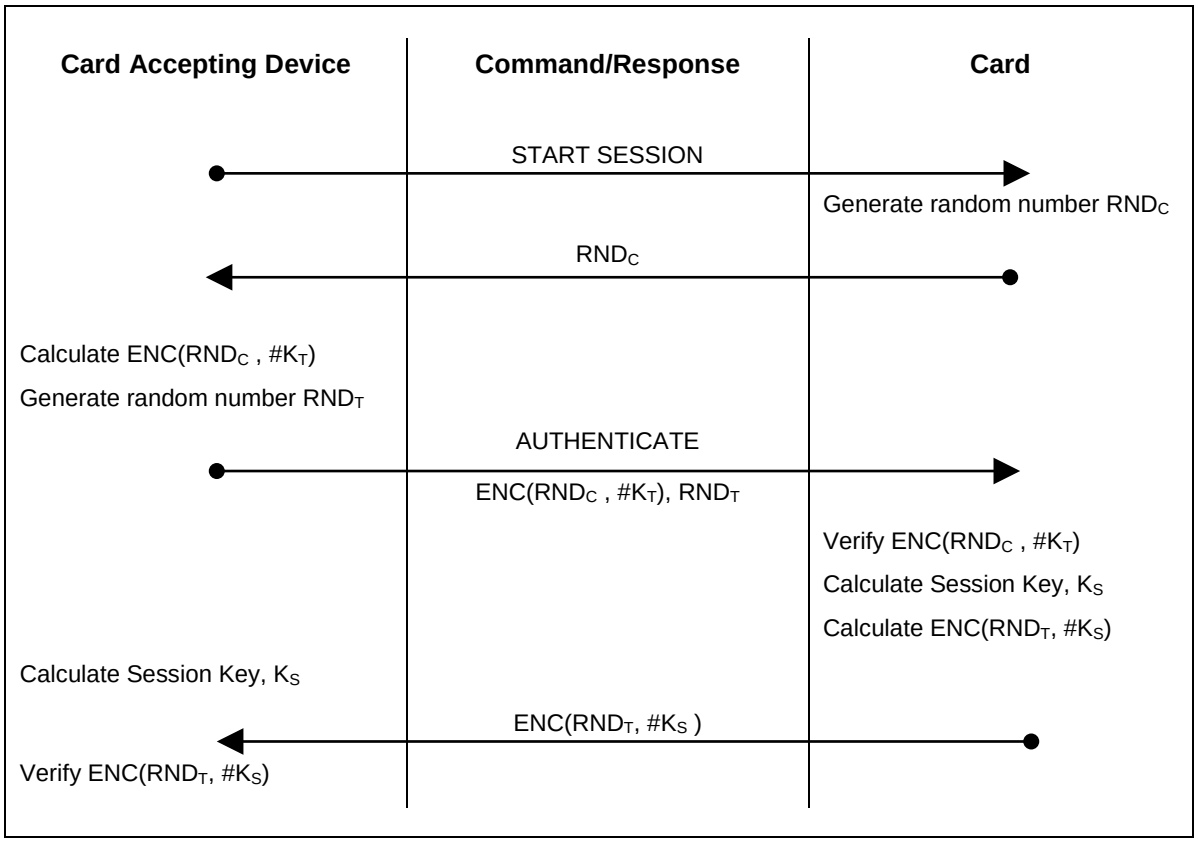


Figure 18: Mutual Authentication Process

Calculation of Session Key K_S also depends on the encryption selected.

If single DES option has been selected

K_S = DES (DES (RND_C, #K_C) ⊕ RND_T, #K_T)

If Triple DES option has been selected,

Left half of K_S = 3DES (3DES (RND_C, #K_C), #K_T)

Right half of K_S = 3DES (RND_T, #(reverse K_T))

Where #(reverse K_T) is obtained by exchanging the Left and Right half of K_T

RND_C Eight bytes random number generated by the Card.

RND_T Eight bytes random number generated by the Card Accepting Device.

K_C Card Key.

K_T Terminal Key.

K_S Session Key.

Note: *ENC shall be DES or 3DES depending on the Option Register.*

The successful completion of the Mutual Authentication is recorded in the card. The resulting *Session Key* (K_S) is used for all data encryption and decryption during the same session.

The Mutual Authentication between Card and Card Accepting Device must be completed in the specified order. If any other card command is sent to the card interrupting the Mutual Authentication procedure as specified above, the card will abort the Mutual Authentication process and erase any intermediate data resulting from the preceding Mutual Authentication commands. The terminal must restart the complete Mutual Authentication procedure from the START SESSION command.

If after a successfully completed Mutual Authentication procedure the card receives the START SESSION command, it erases the previous session key and the complete Mutual Authentication procedure must be repeated to define a new session key. The current security status of the card will be maintained, i.e., Secret Codes that have successfully been submitted to the card need not be submitted again.

The card maintains an error counter CNT K_T to count and limit the number of consecutive unsuccessful executions of the AUTHENTICATE command:

- The error counter is incremented by one (1) each time the authenticate operation fails, e.g. a wrong K_T is presented to the card.
- The error counter is reset to SCN K_T if the authenticate operation is successful.
- If the error counter reaches a value of eight (8), the card will not execute the AUTHENTICATE command any longer. In this case, all related security mechanisms (e.g., the submission of Secret Codes) may be blocked.

  Note: *This condition is irreversible and can make the card unusable.*

The error counter is stored in the Security File. The value of the counter is returned in the card response if a wrong K_T is used in the AUTHENTICATE command.

Card Random Number (RND_C) is derived in a complex non-predictable mathematical process from the Random Number Seed stored in the Security File. The Random Number Seed is internally updated by the operating system after each START TRANSACTION command.

4.3. Secret Codes

Secret Codes stored in the card are used to restrict the access to data stored in user data files and to certain commands provided by the card. Secret codes must be presented to the card in order to be able to read data from or write data to user data files and to execute certain privileged card commands.

ACOS3 provides the following secret codes:

- Five Application Codes (AC)
- One Issuer Code (IC)
- One PIN Code (PIN)

4.3.1. Application Codes

Five Application Codes (AC1…AC5) are available to control the access to the data stored in data files. Each Application Code is eight bytes long.

An option bit in the Security Option Register in the Personalization File specifies for each code whether the code must be submitted to the card in plain or encrypted with the current session key.

4.3.2. Issuer Code

The Issuer Code is provided to control access to data files and to privileged card functions; it is eight bytes long.

An option bit in the Security Option Register in the Personalization file specifies for the IC whether it must be submitted to the card in plain or encrypted with the current session key.

4.3.3. PIN Code

The PIN Code is provided to control access to data files.

The PIN is eight bytes long. The PIN is presented to the card with the SUBMIT CODE command. Depending on the corresponding option bit PIN_DES in the Security Option Register, the PIN is DES encrypted with the current session key before the presentation to the card, or it is presented in plain.

The PIN code can be changed with the CHANGE PIN command if setting the PIN_ALT option bit in the option register has enabled this option. Depending on the option bit PIN_DES, the new code is DES encrypted with the current session key before it is written to the card, or it is written in plain.

4.3.4. Secret Code Submission and Error Counters

Depending on the setting of the corresponding bit in the Security Option Register, a code is submitted to the card in plain or DES-encrypted with the current session key.

If the option bit xx_DES for the code XX is set, the code is presented as follows:

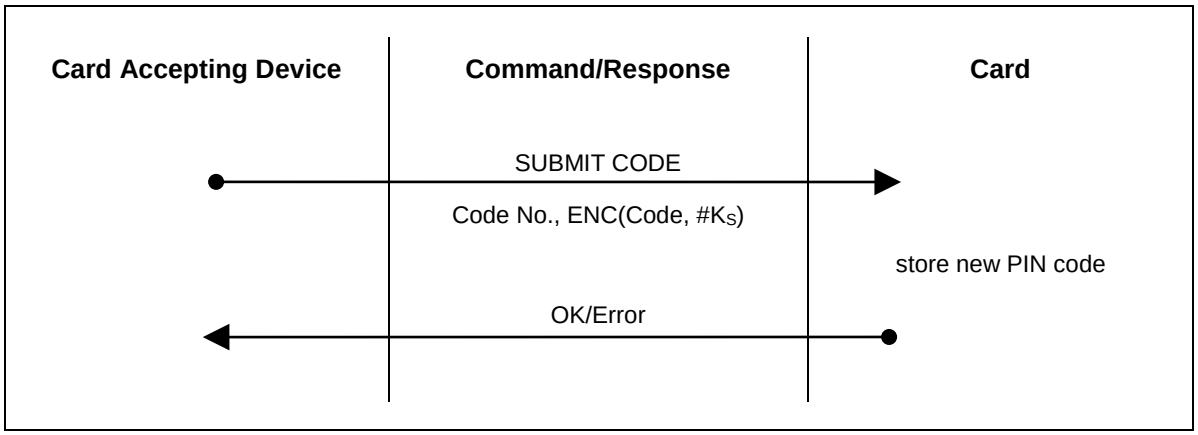


Figure 19: SUBMIT CODE with DES Encryption

Code No. Reference to the particular code that is submitted with the command:

1 ... 5 = Application Codes AC1...AC5

6 = PIN

7 = Issuer Code IC

Other values for *Code No*. are not allowed and will be rejected by the card.

Code The eight bytes secret code to be submitted.

K_S The current session key.

***Note**: ENC shall be DES or 3DES depending on selected Option Register.*

If the option bit xx_DES is not set, the DES encryption of the code is not necessary and the code is submitted in plain:

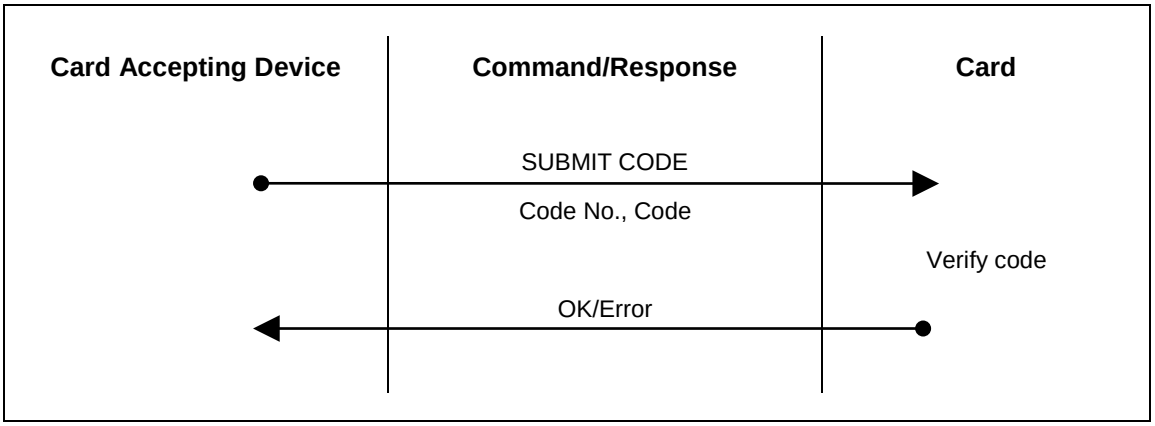


Figure 20: SUBMIT CODE with No Encryption

Code No. Reference to the particular code that is submitted with the command:

1 ... 5 = Application Codes AC1...AC5

6 = PIN

7 = Issuer Code IC

Other values for *Code Number* are not allowed and will be rejected by the card.

Code The eight bytes secret code to be submitted.

The card maintains an error counter CNT xx for each secret code to count and limit the number of consecutive unsuccessful executions of the SUBMIT CODE command:

- The error counter for a particular code is incremented by one (1) each time the Submit Code operation for that code fails, e.g., a wrong secret code is submitted to the card.
- The error counter for a particular secret code is reset to SCN xx when the Submit Code operation for that code has successfully been executed.
- If the error counter reaches a value of eight (8), the card will reject the SUBMIT CODE command for that code.

The error counters CNT xx and starting counters SCN xx are stored in the Security File. The counter value for a particular code is returned in the response by the card to an unsuccessful Submit Code operation.

The terminal key K_T and all codes have default of eight (8) maximum number of retries. This can be customized by modifying the Starting CNT Value (SCN xxx) and the current Error Counter (CNT xxx). For example, to change the maximum number of retries for PIN to 3, the nibble corresponding to PIN in record numbers 11 and 12 can be written with 5.

IMPORTANT: *Care must be taken when writing to record numbers 11 and 12. If any CNT or SCN are written with the nibble 8 or above, that code would be locked. If CNT_{IC} is greater than or equals to 8, the card would become permanently locked.*

4.3.5. Change PIN Code

The PIN code can be changed in the User State with CHANGE PIN command if the option bit PIN_ALT is set.

To program a new PIN code in the card, the current PIN code must have been submitted first.

For security reasons, the CHANGE PIN command can only be executed immediately after a Mutual Authentication process. No other command must have been executed between the Mutual Authentication and the CHANGE PIN command. Otherwise, the command is rejected.

If the option bit PIN_DES is set, the changing of the PIN code is carried out as follows:

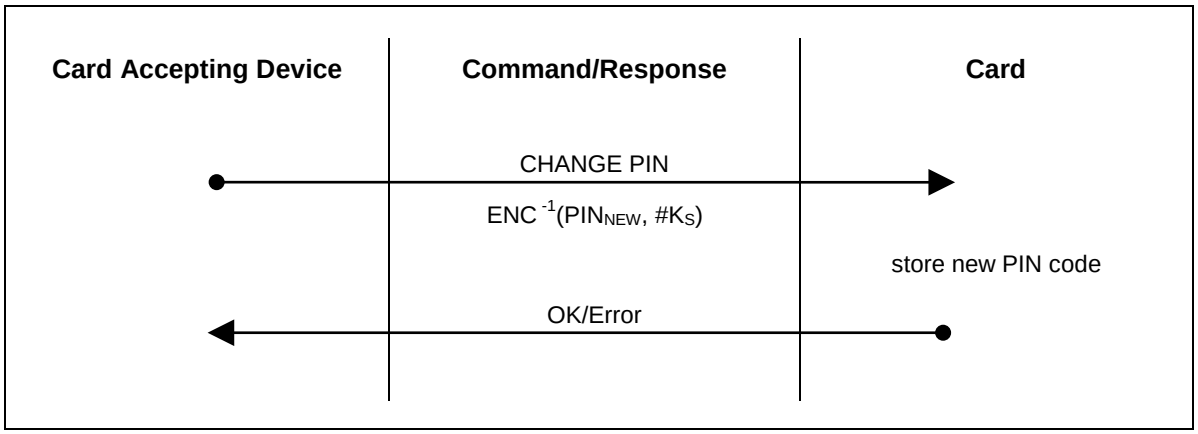


Figure 21: CHANGE PIN Code with set PIN_DES

PIN_{NEW} The new PIN code.

K_S The current session key.

Note: *ENC shall be DES or 3DES depending on selected Option.*

If the option bit PIN_DES is not set, the DES^{-1} encryption of the new PIN is not necessary and the changing of the PIN code is carried out as follows:

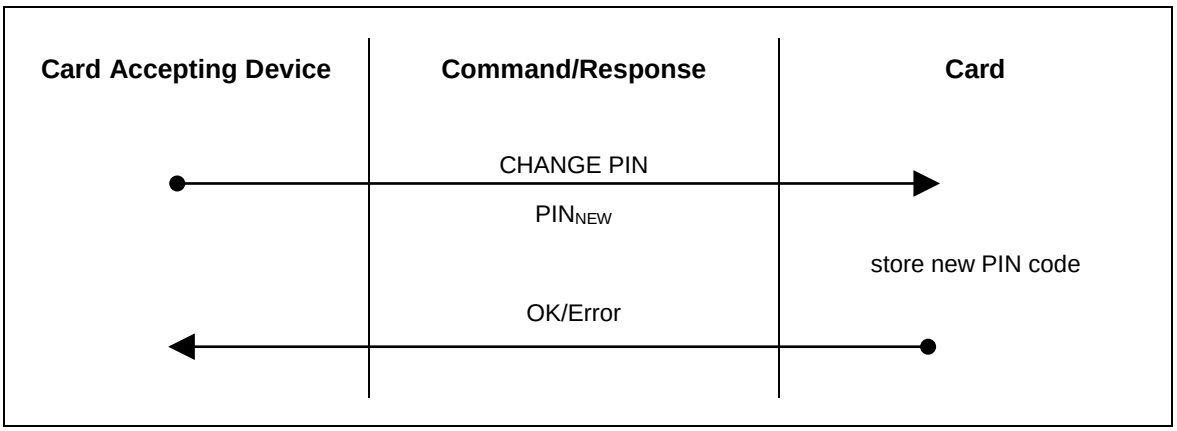


Figure 22: CHANGE PIN Code with no PIN_DES set

4.4. Secure Messaging

ACOS3 Version 1.07 and above support secure messaging (SM) for data files. Secure messaging ensures data transmitted between the card and terminal/server is secured and not susceptible to eavesdropping, replay attack and unauthorized modifications. User data file can be specified (See **Section 3.4.1**) that secure messaging is required for READ/WRITE RECORD/BINARY commands. Almost all the other commands can also use secure messaging initiated by the terminal. The commands that do not accept secure messaging are START SESSION, MUTUAL AUTHENTICATION and GET RESPONSE.

Note: *Issuing the START SESSION command will clear the authentication state. A mutual authentication will have to be performed again to re-establish the session.*

The SM employed in ACOS3 both encrypts and signs the data transmitting into and out of the card. The card will interpret the terminal command is in SM mode if the CLA of the command has the secure messaging bits set.

Sections 4.4.1 to **4.4.7** discuss the constructs used in secure messaging. **Section 4.4.8** details the exact constructs of secure messaging commands and response. **Section 4.4.9** lists the secure messaging return codes.

4.4.1. Notations

To describe the secure messaging adequately, there are some notations to be introduced in this section. Card commands are basically input and output commands – called ISO-IN and ISO-OUT. Some commands have both input and outputs. The terminal needs to call GET RESPONSE to retrieve the outputs for ISO 7816-3 T=0 protocol. This is called an ISO-IN-OUT command. The following are the possible ISO 7816 Part 3 communications protocol command and response pairs.

Commands:

a. With command data:

	CLA	INS	P1	P2	P3 = n	Data in
Bytes	1	1	1	1	1	n

b. Without command data:

	CLA	INS	P1	P2	P3
Bytes	1	1	1	1	1

Responses:

a. Without response data:

	SW1SW2
Bytes	2

b. With response data:

	Response	Sw1Sw2
Bytes	n	2

The class (CLA) byte stated above does not have the secure messaging bits set.

The modified CLA* for secure messaging in **Section 4.4.8** has the secure messaging bits b3 and b2 set to 1. That is, the original CLA OR 0Ch.

P3 is the normal length of the command data.

P3* will be the length of the command data under secure messaging.

The following figure shows the command transformation of a command without data (ISO-OUT) to a secure messaging APDU command:

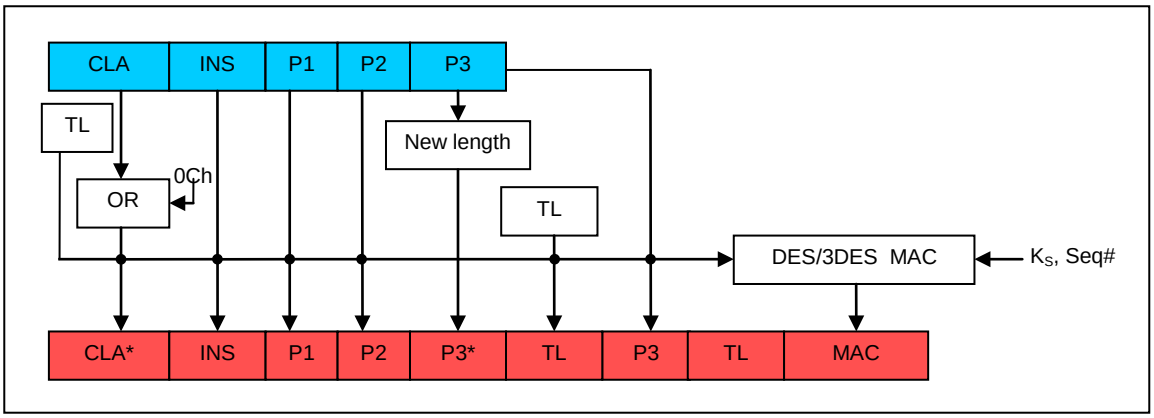


Figure 23: Command Transformation of ISO-OUT

The following figure shows the command transformation of a command with data (ISO-IN) to a secure messaging APDU command:

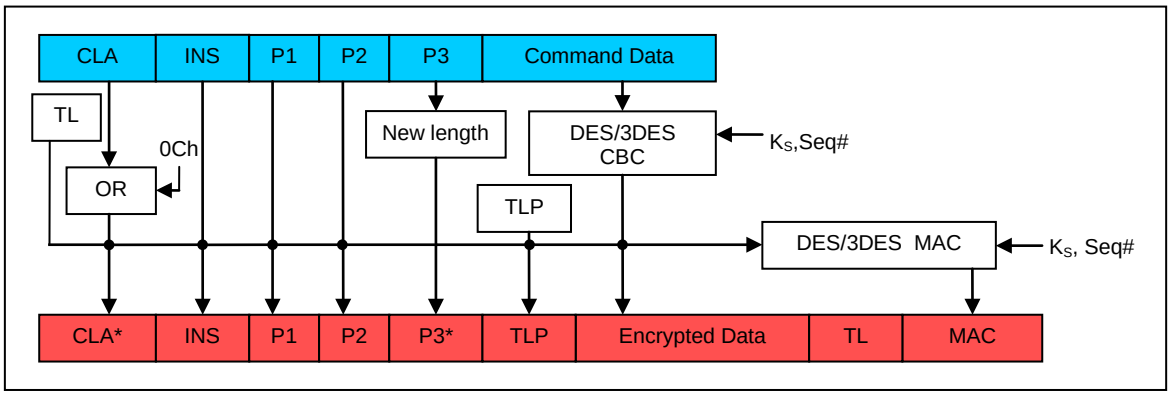


Figure 24: Command Transformation of ISO-IN

The following figure shows the response transformation:

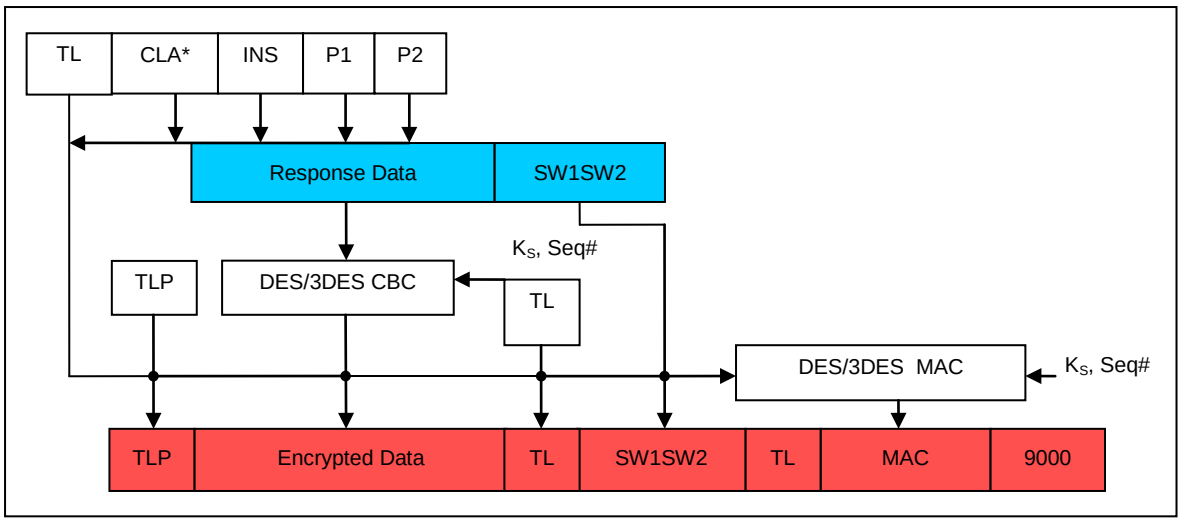


Figure 25: Response Transformation

If there is no response data, that field does not appear in the response output.

The components necessary to compute secure messaging are explained in the sections below. The terms TL and TLP are *Tag-Length* and *Tag-Length-Padding* respectively described in **Section 4.4.7**.

4.4.2. Secure Messaging Key

The key used in secure messaging is the Session Key (K_S) computed in **Section 4.2**. Therefore, mutual authentication between ACOS3 and the terminal must be performed before secure messaging. The secure messaging key is either DES or 3DES depending on the session key generated (which depends on the 3DES bit in **Section 3.3.3.1**).

4.4.3. Sequence Number

The sequence number (seq#) is used as the initial vector in the CBC calculation. The start of the sequence number is the increment of the random number of the last 2 bytes of the START SESSION command padded with 6 NULL bytes. The response data is the increment of the command sequence number.

If a command is sent from the terminal to the card SM'ed with seq#, and secure messaging is successful, the card will reply with a MAC computed with seq# + 1. The next secure messaging command to send will use seq# + 2. When the sequence number reaches 00 00 00 00 00 00 FF FFh, the next number would be 00 00 00 00 00 00 00 00h.

4.4.4. Authentication and Integrity

The COS and terminal will use the SM key and DES/3DES in CBC mode to compute the MAC. For the purpose of this section, this notation would denote MAC computations:

SIGN_CBC (data to sign, Initial Vector = seq#)

Only the first 4 bytes of the resultant MAC are used for the command and response data. When a SM command is sent to the card, the card will first verify the MAC_{cmd} with the command and command data. The COS will execute the command only if the MAC_{cmd} is verified. This will ensure that the data is genuinely from the terminal.

If the MAC_{cmd} verification failed (status word = 6988h), the sequence number *n* would have been incremented. The next command should use *n* + 1. In the case that the sequence number is out of synchronization between the terminal and card, a new session key can be re-established.

4.4.5. Confidentiality

The COS and terminal will use the SM key and DES/3DES in CBC mode to compute the encryption. The following notation denotes the secure messaging encryption:

ENC_CBC (plaintext to encrypt, Initial Vector = seq#)

After verifying the MAC_{cmd} the COS will decrypt the command data encrypted by the terminal. This command data will be written to the card based on the write binary or record command.

4.4.6. Padding

DES and 3DES algorithms have an input block size of eight (8). An appropriate padding is necessary. If the data to sign or encrypt is not in a multiple of eight (8), an 80h byte is appended followed by 0 to 6 00h to make the data to sign to be a multiple of 8 bytes. If the data to sign or encrypt is a multiple of 8, then no padding is applied. For data encryption, a padding-content indicator will indicate how many (0-7) bytes of padding are applied.

4.4.7. Secure Messaging Data Object

Secure Messaging Data Objects (SMDOs) are necessary to describe the data elements unambiguously when transferring in SM mode. The following SMDOs are supported in ACOS3 Secure Messaging:

Tag	Length	Description
87h	Var.	Padding-content indicator byte followed by cryptogram
89h	4	Command header (CLA* INS P1 P2)
8Eh	4	Cryptographic checksum
97h	1	Original P3 in an ISO-out command
99h	2	Processing status word (Sw1 Sw2) of the command.

Table 5: Secure Messaging Data Objects

The exact usage of these SMDOs is stated in the next section with the possible command and response data pair for SM.

4.4.8. Secure Messaging Semantics

In order to send out a secure messaging command, the normal APDU of **Section 4.4.1** will be modified with SMDOs. The following three subsections show how those modifications are to be done.

4.4.8.1. ISO-IN

In commands such as WRITE RECORD and BINARY, the commands are extended with SMDOs as follows:

	CLA*	INS	P1	P2	P3*	87h	L_{87}	Pi	Encrypted Data	8Eh	04h	MAC_{cmd}
Bytes	1	1	1	1	1	1	1	1	n*	1	1	4

CLA* = CLA OR 0Ch = 8Ch

P3* = 3 + n* + 2 + 4

L_{87} = n* + 1 (Length of Tag 87)

Pi = Padding indicator: 00h – No padding is used in encrypted data

01-07h – Number of padding bytes used in encrypted data

n* = P3 + length (padding) = P3 + Pi

Seq# = previous Seq# + 1

Encrypted Data = ENC_CBC (<Command Data> padding, Seq#))

MAC_{cmd} = SIGN_CBC (<89 04h CLA* INS P1 P2> <87h L_{87} Pi Encrypted Data> padding, Seq#)

Since the maximum of P3* must be less than 255, with the MAC, padding and the SMDO, the original P3 must be less than or equal to 240 bytes.

If the command accepts secure messaging, the key has been established and the MAC_{cmd} is correct, the response will be 610Ch. Note that 610Ch may not actually mean that the command is successful. It merely means that the Secure Messaging is successful. A subsequent call to GET RESPONSE will yield the actual Status Words stating success or error. The Get Response must have P3 = 0Ch exactly. Otherwise 6C0Ch will be replied without SM.

The response to GET REPONSE is as follows:

	99h	02h	SW1SW2	8Eh	04h	MAC_{rsp}	90 00h
Bytes	1	1	2	1	1	4	2

Seq# = Seq# of SM command + 1

MAC_{rsp} = SIGN_CBC (<89 04h CLA* INS P1 P2> <99 02h SW1 SW2> padding, Seq#)

The field SW1SW2 is the actual status word returned for the command. The last status word 90 00h states that the GET RESPONSE command is successful.

4.4.8.2. ISO-OUT

The ISO-OUT commands effectively become an ISO-IN command when the SM field is set.

	CLA*	INS	P1	P2	P3*	97h	01h	P3	8Eh	04h	MAC_{cmd}
Bytes	1	1	1	1	1	1	1	1	1	1	4

CLA* = CLA OR 0Ch = 8Ch

P3* = 9

Seq# = previous Seq# + 1

MAC_{cmd} = SIGN_CBC (<89 04h CLA* INS P1 P2> <97 01h P3> padding, Seq#)

If the command accepts secure messaging, the key has been established and the MAC_{cmd} is correct, the response will be 61xx. Same as ISO-IN, the status word of 61xx only means that SM on the command is successful. The actual success of the overall command will depend on the SW1SW2 data object when a subsequent GET RESPONSE is called with P3 = xx, where xx can be 15-FDh.

Note the get response must be called with P3 = xx exactly, else 6Cxx will be returned. The original data out *n* must be less than or equal to 240 bytes. Else, error status 6700h will be returned.

The response is as follows:

	87h	L_{87}	Pi	Encrypted data	99h	02h	SW1SW2	8Eh	04h	MAC_{rsp}	90 00h
Bytes	1	1	1	n*	1	1	2	1	1	4	2

L_{87} = n* + 1 (Length of Tag 87h)

Pi = Padding indicator: 00 – No padding is used in encrypted data

01-07h – Number of padding bytes used in encrypted data

Seq# = Seq# of SM command + 1

Encrypted Data = ENC_CBC (<Response Data> padding, Seq#))

n* = P3 + length (padding) = P3 + Pi

MAC_{rsp} = SIGN_CBC (<89 04h CLA* INS P1 P2> <87h L_{87} Pi Encrypted Data> <99 02h SW1 SW2> padding, Seq#)

4.4.8.3. ISO-IN-OUT

In commands such as INQUIRE ACCOUNT and DEBIT, the commands are extended with SMDOs as follows:

	CLA*	INS	P1	P2	P3*	87h	L_{87}	Pi	Encrypted Data	8Eh	04h	MAC_{cmd}
Bytes	1	1	1	1	1	1	1	1	N*	1	1	4

CLA* = CLA OR 0Ch = 8Ch

P3* = 3 + n* + 2 + 4

L_{87} = n* + 1 (Length of Tag 87h)

Pi = Padding indicator: 00h – No padding is used in encrypted data

01-07h – Number of padding bytes used in encrypted data

n* = P3 + length (padding) = P3 + Pi

Seq# = previous Seq# + 1

Encrypted Data = ENC_CBC (<Command Data> padding, Seq#))

MAC_{cmd} = SIGN_CBC (<89 04h CLA* INS P1 P2> <87h L_{87} Pi Encrypted Data> padding, Seq#)

Since the maximum of P3* must be less than 255, with the MAC, padding and the SMDO, the original P3 must be less than or equal to 240 bytes.

If the command accepts secure messaging, the key has been established and the MAC_{cmd} is correct, the response will be 61xx. Same as ISO-IN, the status word of 61xxh only means that SM on the command is successful. The actual success of the overall command will depend on the SW1SW2 data object when a subsequent Get Response is called with P3 = xx, where xx can be 15-FDh.

Note the Get response must be called with P3 = xx exactly, else 6Cxxh will be returned. The original data out *n* must be less than or equal to 240 bytes. Else, error status 6700h will be returned.

The response is as follows:

	87h	L_{87}	Pi	Encrypted data	99h	02h	SW1SW2	8Eh	04h	MAC_{rsp}	90 00h
Bytes	1	1	1	n*	1	1	2	1	1	4	2

L_{87} = n* + 1 (Length of Tag 87h)

Pi = Padding indicator: 00h – No padding is used in encrypted data

01-07h – Number of padding bytes used in encrypted data

Seq# = Seq# of SM command + 1

Encrypted Data = ENC_CBC (<Response Data> padding, Seq#))

n* = P3 + length (padding) = P3 + Pi

MAC_{rsp} = SIGN_CBC (<89 04h CLA* INS P1 P2> <87h L_{87} Pi Encrypted Data> <99 02h SW1 SW2> padding, Seq #)

4.4.9. Secure Messaging Specific Return Codes

The following table lists the specific SM return codes:

SW1	SW2	Meaning
61	xxh	SM successful, call GET RESPONSE to retrieve xx bytes.
67	00h	The original P3 is greater than 240 bytes. SMDO overflow.
68	82h	Secure messaging not allowed.
69	82h	Condition of use not satisfied – Secure Messaging required but not present.
69	85h	The Session Key has not been established.
69	87h	Expected secure messaging data objects missing.
69	88h	The MAC_{cmd} does not match the data.
6C	xxh	Get Response with xx byte as P3 is expected. Please re-issue Get Response with P3=xx.

Table 6: Secure Messaging Specific Return Codes

4.5. Account Transaction Processing

Associated to the *Account* are four keys:

- The Credit Key (K_{CR})
- The Debit Key (K_D)
- The Certify Key (K_{CF})
- The Revoke Debit Key (K_{RD})

The keys are stored in the Account Security File.

The keys are used in the calculation and verification of MAC cryptographic checksums on commands and data exchanged between the card and the Card Accepting Device in the Account processing.

All keys are 8 bytes long. The least significant bit of each byte of the keys is not used in the calculation and not interpreted by the card operating system.

Debit Key, Credit Key and Revoke Debit Key have each associated an error counter CNT K_{xx} to count and limit the number of consecutive unsuccessful executions of the transaction commands:

The error counter for a key is incremented by one each time a command using the key fails due to a wrong key used by the Card Accepting Device.

The error counter is reset to CNT K_{xx} when a command using the key is successful.

If the error counter of a command reaches a value of eight (8), the card will reject any further commands using that key.

The error counters CNT K_{xx} for the transaction processing keys are stored in the normal Security File. Anti-tearing protection is done on the update to prevent a loss of this important information during update.

Four different transaction types can be executed on the Account Data Structure under security conditions:

- INQUIRE ACCOUNT
- DEBIT
- REVOKE DEBIT
- CREDIT

The Account Data Structure can be read as a record oriented file in the Manufacturing State, in the Personalization State and in the User State after presentation of the Issuer Code IC. In the normal User State, a WRITE access to the account is possible only through the special account processing commands. WRITE RECORD access is possible after presentation of the Issuer Code (IC).

As an additional feature for very security critical applications, the option bits TRNS_AUT and INQ_AUT in the *Option Register* will enforce Mutual Authentication to be carried out prior to Account Processing.

If the option bit TRNS_AUT is set, the CREDIT, DEBIT and REVOKE DEBIT commands can only be executed by the card after a successful completion of Mutual Authentication. The MAC cryptographic checksum is encrypted with the current Session Key before it is transmitted to the card.

If the option bit INQ_AUT is set, the INQUIRE ACCOUNT command can be executed only after a successful completion of the Mutual Authentication. The MAC cryptographic checksum returned by the card is encrypted with the current Session Key.

4.5.1. INQUIRE ACCOUNT

In the INQUIRE ACCOUNT transaction, the card returns the current balance value together with other relevant account information and a MAC cryptographic checksum on the relevant data. This signature can be regarded as a certificate issued by the card on the current balance and on the immediately preceding transaction. The key to be used in the generation of the MAC cryptographic checksum can be specified.

To prevent a replay of the response from a previous INQUIRE ACCOUNT command, the card-accepting device can pass a reference value to the card to be included in the MAC calculation.

If the option bit INQ_AUT is set, the Mutual Authentication process must have been completed prior to the execution of the INQUIRE ACCOUNT command.

The Inquire Account transaction is carried out as follows:

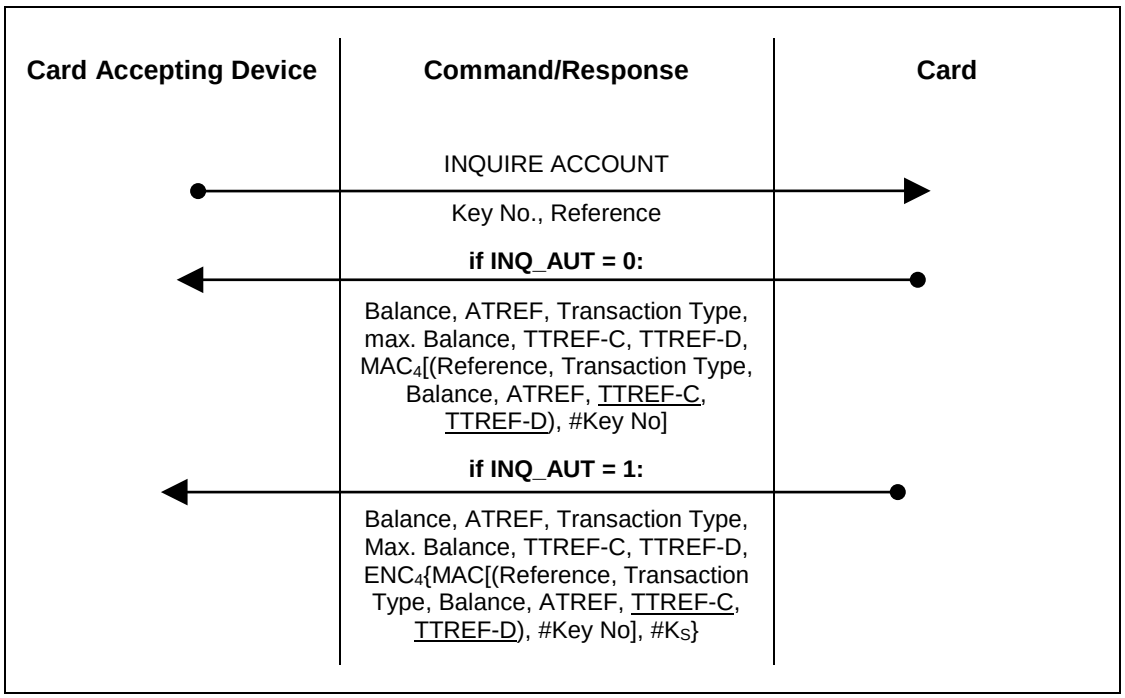


Figure 26: INQUIRE ACCOUNT Transaction

Notes:

1. *Underlined fields are included only when the INQ_ACC_MAC flag in the Manufacturer File is equal to one.*
2. *ENC shall be DES or 3DES depending on selected Option Register.*

Reference	Four (4) bytes reference value supplied by the card-accepting device to be included in the calculation of the MAC cryptographic checksum
Key No.	Reference to the Account key to be used in the calculation of the MAC cryptographic checksum: 0 = Debit Key K_{D1} 1 = Credit Key K_{CR} 2 = Certify Key K_{CF} 3 = Revoke Debit Key K_{RD} Other values are not permitted and will be rejected by the card.
Balance	Current balance value
ATREF	Account Transaction Reference of last transaction
Transaction Type	One byte specifying the type of the last transaction performed on card: 1 = DEBIT 2 = REVOKE DEBIT 3 = CREDIT
Max. Balance	The maximum allowed balance value in the card
TTREF-C	Terminal Transaction Reference of the last Credit transaction
TTREF-D	Terminal Transaction Reference of the last Debit transaction
MAC_4	The first 4 bytes of MAC cryptographic checksum using the key specified by *Key No.*
ENC_4	The first 4 bytes of the MAC cryptographic checksum using the key specified by *Key No.* encrypted with the current Session Key (K_S)

If INQ_ACC_MAC flag in Manufacturer file is zero, the first two blocks (16 bytes) will be used to calculate the MAC cryptographic checksum. If INQ_ACC_MAC flag is one, all three blocks (24 bytes) will be used to calculate the MAC cryptographic checksum.

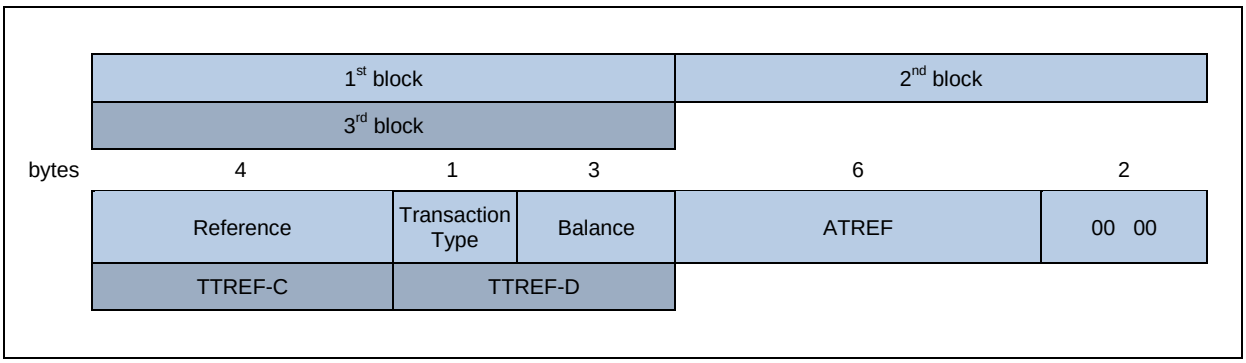


Figure 27: INQUIRE ACCOUNT MAC Data Block

Note: *ACS writes the INQ_ACC_MAC flag before the devices are being shipped. It is not changeable by the card issuer.*

4.5.2. DEBIT

In a DEBIT transaction, the balance in the Account is decreased by the specified amount. The maximum amount that can be debited to the Account is the current balance value. Negative balance values are not allowed.

Different security conditions can be specified for the Debit transaction to allow for different security requirements. The security conditions for the Debit transaction are specified in the DEB_MAC and DEB_PIN option bits in the options register.

Proper setting of these option bits can specify four different security conditions:

DEB_MAC	DEB_PIN	Security Condition
0	0	No security checking; the DEBIT transaction can always be executed.
0	1	The PIN code must have been submitted to the card prior to the execution of the DEBIT transaction.
1	0	The MAC cryptographic checksum is required with the DEBIT transaction.
1	1	The MAC cryptographic checksum with the DEBIT transaction is required **and** the PIN code must have been submitted to the card.

Table 7: Security Conditions in Debit Transaction

If the option bit TRNS_AUT is set, the Mutual Authentication process must have been completed prior to the execution of the DEBIT command.

The DEBIT transaction is carried out as follows:

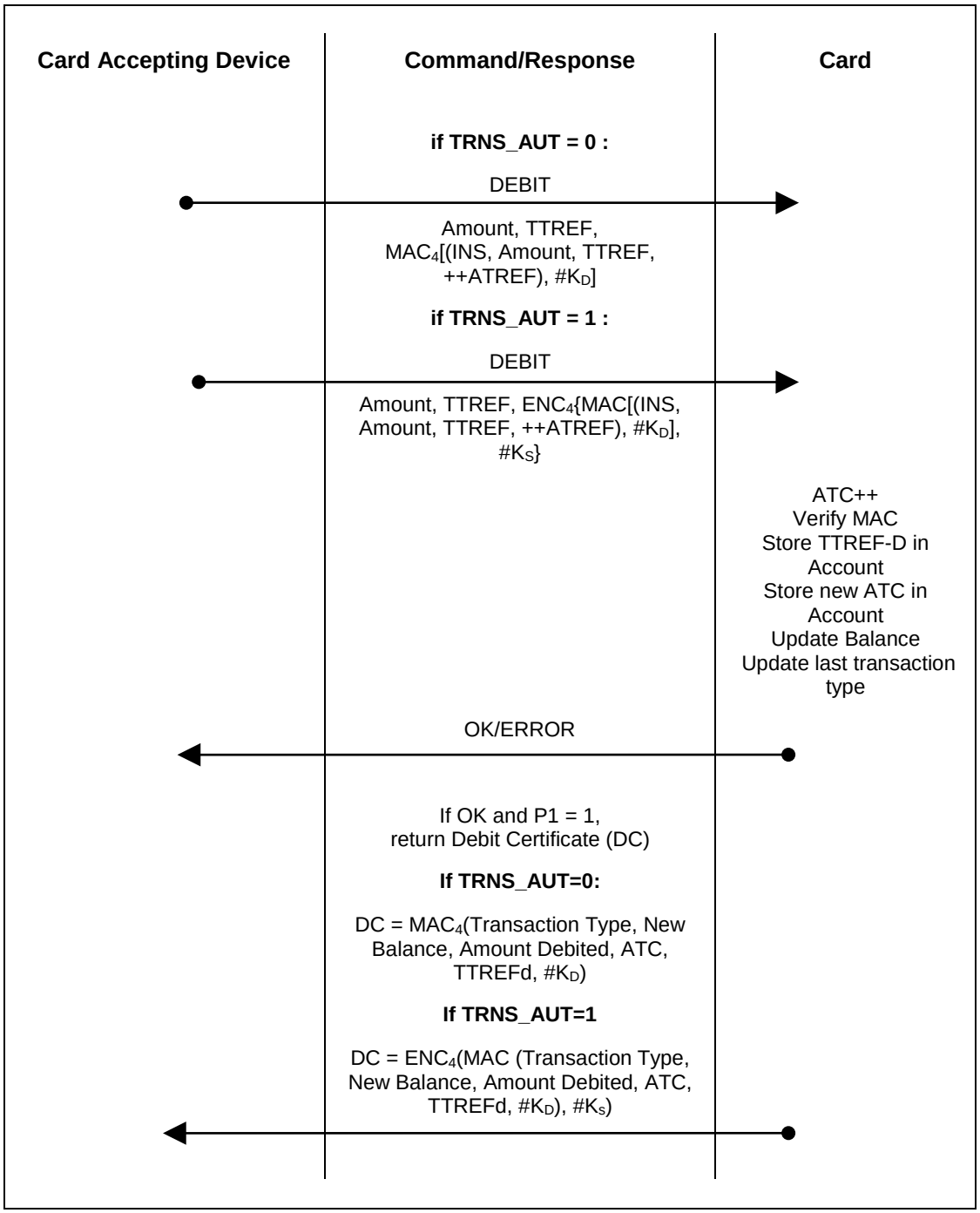


Figure 28: DEBIT Transaction

***Note**: ENC shall be DES or 3DES depending on selected Option Register.*

TTREF	Terminal Transaction Reference for this DEBIT transaction.
++ATREF	Account Transaction Reference for this transaction.
INS	ACOS3 instruction code for DEBIT command.
Amount	Amount to be debited to the Account.

K_D	Debit Key.
MAC_4	The first 4 bytes of a MAC cryptographic checksum using K_D as the key.
ENC_4	The first 4 bytes of MAC cryptographic checksum using K_D encrypted with the current Session Key (K_S).

***Note**: The transaction counter in the card is <u>incremented before</u> the transaction is being executed.*

The sixteen bytes data string on which the MAC cryptographic checksum is calculated is composed as follows:

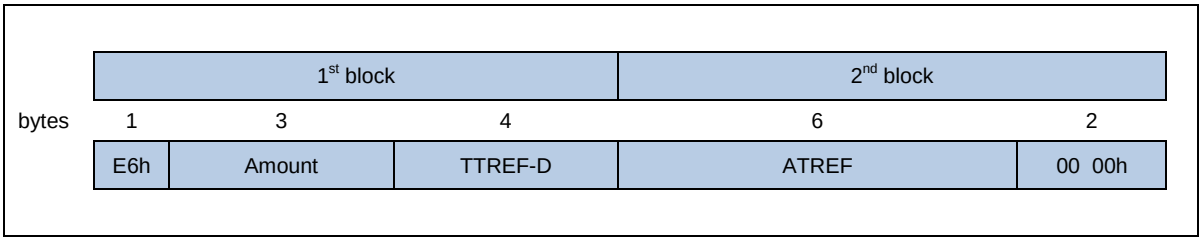


Figure 29: DEBIT MAC Data Block

If the option bit DEB_MAC is not set, the 4 bytes of MAC cryptographic checksum must be transmitted in the command to the card but they are not evaluated by the operating system. The card will accept any value transmitted.

4.5.2.1. Debit Certificate

To make sure that the DEBIT command is indeed executed, the terminal may request for a DEBIT CERTIFICATE from the card. The DEBIT Certificate is a MAC cryptographic checksum computed by the DEBIT KEY and the following data block:

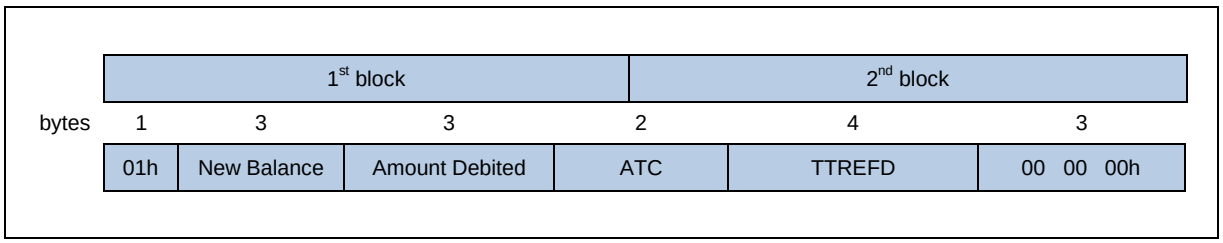


Figure 30: DEBIT Certificate MAC Data Block

4.5.3. REVOKE DEBIT

REVOKE DEBIT is only possible after a Debit transaction and applies always to the immediately preceding Debit transaction. The Revoke Debit transaction can be executed to annul a Debit transaction, for example, if the amount debited was found wrong later on. As a result of the transaction, the balance value that was valid before the last Debit transaction is restored.

The Revoke Debit transaction is enabled and disabled by the option bit REV_DEB in the option register.

If the option bit TRNS_AUT is set, the Mutual Authentication process must have been completed prior to the execution of the REVOKE DEBIT command.

The REVOKE DEBIT transaction is carried out as follows:

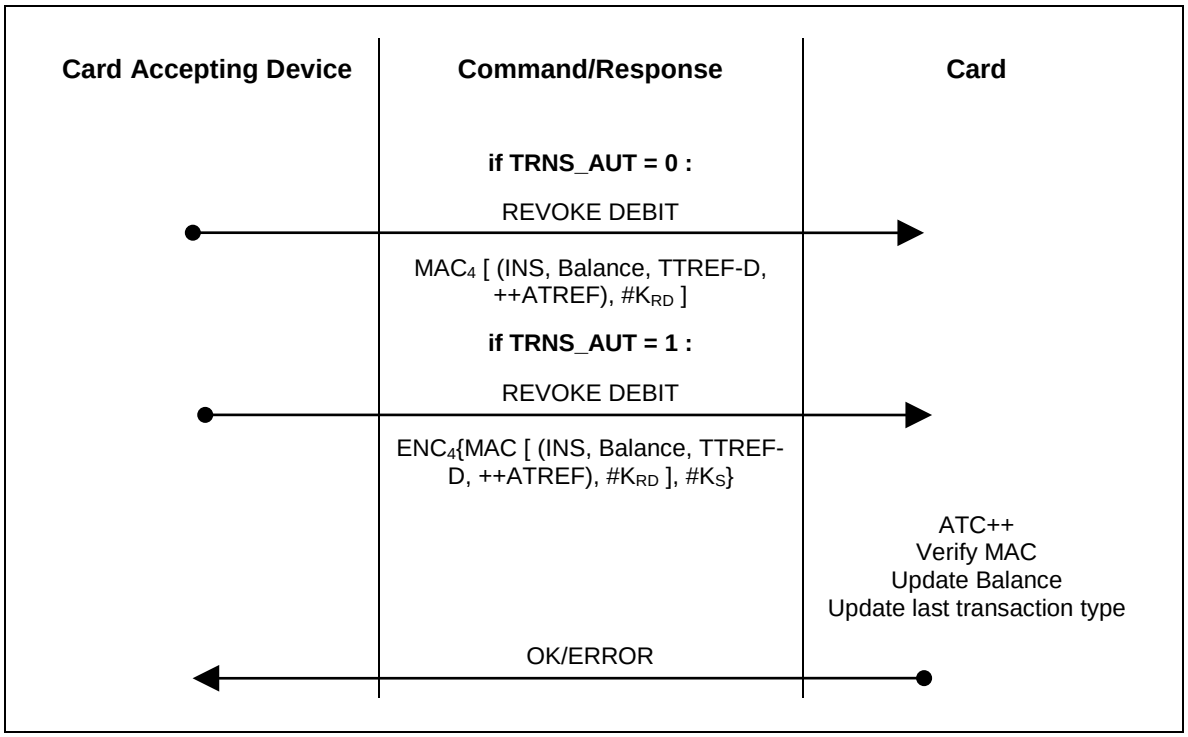


Figure 31: REVOKE DEBIT Transaction

***Note**: ENC shall be DES or 3DES depending on selected Option Register.*

INS	ACOS3 instruction code for REVOKE DEBIT command.
Balance	Balance value to be restored (= balance before the preceding DEBIT transaction).
TTREF-D	Terminal Transaction Reference used in the preceding DEBIT transaction.
++ATREF	Account Transaction Reference for this transaction.
K_{RD}	REVOKE DEBIT key.
MAC_4	The first 4 bytes of a MAC cryptographic checksum using K_{RD} as the key.
ENC_4	The first 4 bytes of MAC cryptographic checksum using K_{RD} encrypted with the current Session Key (K_S).

***Note**: The transaction counter in the card is <u>incremented before</u> the transaction is being executed.*

The sixteen bytes data string on which the MAC cryptographic checksum is calculated is composed as follows:

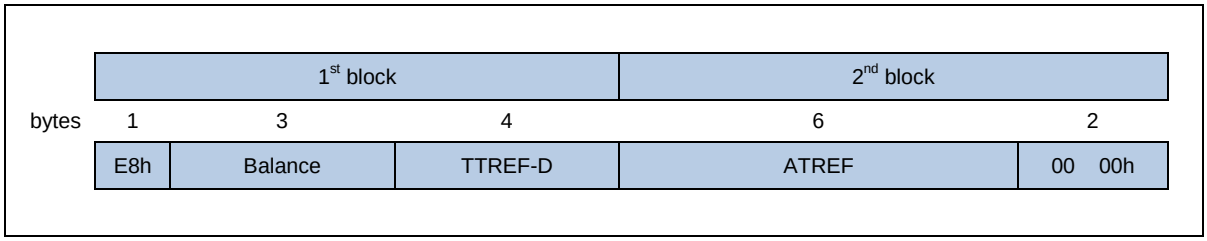


Figure 32: REVOKE DEBIT MAC Data Block

4.5.4. CREDIT

In a CREDIT transaction, the balance in the Account is increased by the specified amount. The maximum allowed the new balance must not exceed balance value MAXBAL as stored in the Account Data Structure. Otherwise, the card will reject the CREDIT command.

The Credit transaction is always carried out under high security processing.

If the option bit TRNS_AUT is set, the Mutual Authentication process must have been completed prior to the execution of the CREDIT command.

The CREDIT transaction is carried out as follows:

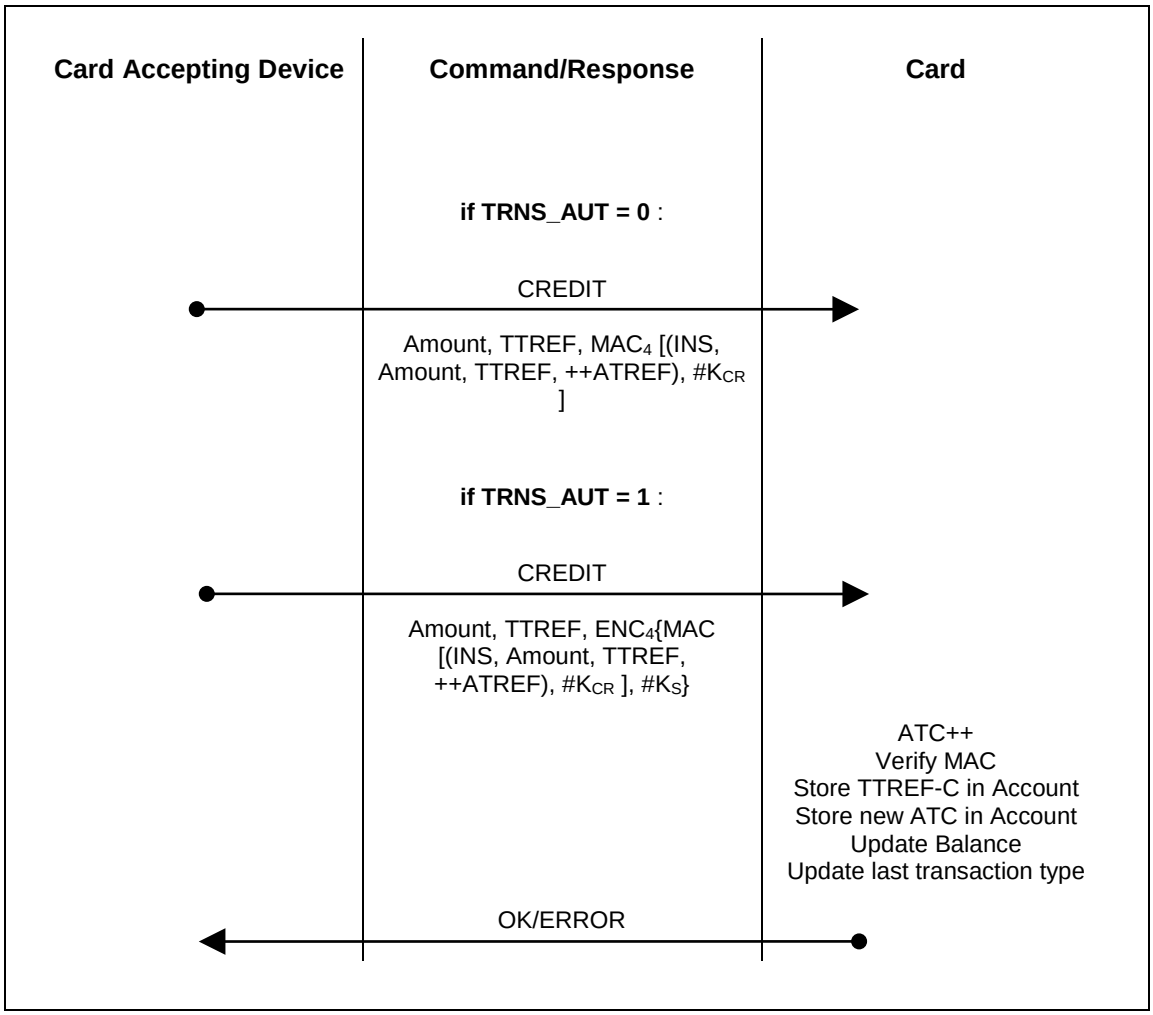


Figure 33: CREDIT Transaction

Note: *ENC shall be DES or 3DES depending on selected Option Register.*

TTREF	Terminal Transaction Reference for this CREDIT transaction.
++ATREF	Account Transaction Reference for this transaction.
INS	ACOS3 instruction code for CREDIT command.
Amount	Amount to be credited to the Account.
K_{CR}	CREDIT key.
MAC_4	The first 4 bytes of a MAC cryptographic checksum using K_{CR} as the key.

ENC_4 First four bytes of MAC cryptographic checksum using K_{CR} encrypted with the current Session Key K_S.

Note: *The transaction counter in the card is incremented before the transaction is being executed.*

The sixteen bytes data string on which the MAC cryptographic checksum is calculated is composed as follows:

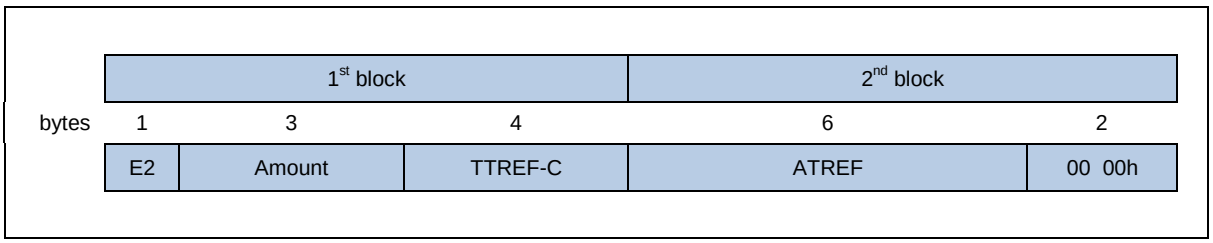


Figure 34: CREDIT MAC Data Block

4.6. Anti-tearing Mechanism

Anti-tearing mechanism help protects card data and security in the event that the card is suddenly powered down or pulled out during a card operation.

When writing user data into the card, ACOS3's anti-tearing mechanism ensures the operation is performed automatically. That is, data is either completely written or the target writing area is left at its previous state before the write operation. The account data file is protected similarly when performing CREDIT/DEBIT/REVOKE DEBIT commands.

When verifying secret codes and performing mutual authentication, the card ensures that, the error counter is decremented before the comparison is performed (and reverts to the maximum value if the verification/authentication is successful). This helps protect the card from side channel attacks. Please note that while this method provides a strong protection against the card's secret code and keys, it does require additional erase/write cycles of the EEPROM even if the verification/authentication is successful. Hence, care should be taken to treat code verification or mutual authentication as an EEPROM write operation.

5.0. Commands

This section describes in detail the format of all ACOS3 commands and the possible responses.

The command descriptions use the APDU representation. All numeric values are given in HEX.

A summary of the status codes returned by the card is given in **Appendix A**. Secure messaging can be added to any commands except START SESSION, MUTUAL AUTHETNCATION and GET RESPONSE. To use SM on any other commands, please see **Section 4.4** for more information.

Command	INS	Description
START SESSION	84h	Start the mutual authentication process.
AUTHENTICATE	82h	Performs mutual authentication between card and terminal and compute a session key.
GET RESPONSE	C0h	Get response data available in the card.
SUBMIT CODE	20h	Submit a secret code.
CHANGE PIN	24h	Change the PIN secret code.
GET CARD INFO	14h	Get card's serial number.
CLEAR CARD	30h	Clears the card back to personalization state.
SELECT FILE	A4h	Select a data file for reading and writing.
READ RECORD	B2h	Read data from a record of the currently selected record file.
WRITE RECORD	D2h	Write data to a record of the currently selected record file.
READ BINARY	B0h	Read data from currently selected binary file.
WRITE BINARY	D0h	Write data to the currently selected binary file.
INQUIRE ACCOUNT	E4h	Read the balance and other account information.
CREDIT	E2h	Credit the account.
DEBIT	E6h	Debit the account.
REVOKE DEBIT	E8h	Revoke the preceding debit transaction.

Table 8: Command Set

5.1. START SESSION

This command is used to get a random number from the card and start the mutual authentication process to produce the Session Key (K_S).

Command

CLA	INS	P1	P2	P3
80h	84h	00h	00h	08h

Response

Data	SW1 SW2
RND_C	Status

Where:

RND_C Eight bytes card random number.

Status Codes

SW1	SW2	Meaning
69	83h	Terminal Authentication Key K_T is locked, authentication process cannot be executed.

5.2. AUTHENTICATE

5.2.1. AUTHENTICATE For Contact

This command is used to submit the encrypted random number to the card and initiate the computation of the session key. Please see **Section 4.2** for more information.

Command

CLA	INS	P1	P2	P3	DATA	
80h	82h	00h	00h	10h	$ENC(RND_C,\#K_T)$	RND_T

Where:

$ENC(RND_C,\#K_T)$ Eight bytes card random number encrypted with Terminal Key K_T.

RND_T Eight bytes terminal random number.

Note: *ENC shall be DES or 3DES depending on the selection in Option Register.*

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
69 85h	START SESSION not executed immediately before AUTHENTICATE command.
63 Cnh	Key K_T not correct; n = remaining number of re-tries
61 08h	Issue GET RESPONSE with P3 = 8 to get the encrypted terminal random number.

To get the response, execute the GET RESPONSE command:

Command

CLA	INS	P1	P2	P3
80h	C0h	00h	00h	08h

Response

Data	SW1 SW2
$ENC(RND_T,\#K_S)$	Status

$ENC(RND_T,\#K_S)$ Eight bytes terminal random number RND_T DES-encrypted with Session Key (K_S).

Note: *ENC shall be DES or 3DES depending on the selection in Option Register.*

Specific Status Codes

SW1 SW2	Meaning
69 85h	AUTHENTICATE not executed prior to the GET RESPONSE command.

5.2.2. AUTHENTICATE For Contactless

This command is used to submit the encrypted random number to the card and initiate the computation of the session key. Please see **Section 4.2** for more information.

Command

CLA	INS	P1	P2	P3	DATA	
80h	82h	00h	00h	10h	$ENC(RND_C,\#K_T)$	RND_T

Where:

$ENC(RND_C,\#K_T)$ Eight bytes card random number encrypted with Terminal Key K_T.

RND_T Eight bytes terminal random number.

Note: *ENC shall be DES or 3DES depending on the selection in Option Register.*

Response

Data	SW1 SW2
$ENC(RND_T,\#K_S)$	Status

$ENC(RND_T,\#K_S)$ Eight bytes terminal random number RND_T DES-encrypted with Session Key (K_S).

Note: *ENC shall be DES or 3DES depending on the selection in Option Register.*

Specific Status Codes

SW1 SW2	Meaning
69 85h	START SESSION not executed immediately before AUTHENTICATE command.
63 Cnh	Key K_T not correct; n = remaining number of re-tries
90 00h	Command executed successfully

5.3. GET RESPONSE

This command is used to retrieve the response data to an APDU case 4 command (incoming and outgoing data).

Command

CLA	INS	P1	P2	P3
80h	C0h	00h	00h	Len

Where:

Len The expected response data length.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
90 00h	OK.
6C nnh	Wrong expected data length - issue command again with P3 = nnh
69 85h	No data available.
62 81h	Part of the data may be corrupted.

The GET RESPONSE command must be issued **immediately** after the receiving 61 nnh from a previous command.

5.4. SUBMIT CODE

This command is used to submit a secret code – Application Code, PIN or Issuer Code – to the card.

Command

CLA	INS	P1	P2	P3	DATA
80h	20h	Code No.	00h	08h	Code or ENC(Code,#K_S)

Where:

Code No. Code reference:

1...5 = AC1...AC5

6 = PIN

7 = IC

Code Eight bytes Code.

ENC(Code,#K_S) Eight bytes Code encrypted with Session Key.

Note: *If the corresponding option bit xx_DES in the Security Option Register is set, code XX is submitted DES encrypted. If the option bit is not set, the code is submitted in plain without encryption.*

ENC shall be DES or 3DES depending on the selection in Option Register.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
63 Cnh	Wrong Code; n = remaining number of re-tries
69 83h	The specified Code is locked.
69 85h	Mutual Authentication not successfully completed prior to the SUBMIT CODE command.

5.5. CHANGE PIN

This command is used to set a new PIN code in the card.

Command

CLA	INS	P1	P2	P3	DATA
80h	24h	00h	00h	08h	PIN or $ENC^{-1}(PIN_{NEW}, \#K_S)$

Where:

PIN_{NEW} New PIN.

Ks Session Key.

Notes: *If the option bit PIN_DES is 0, the PIN code is not DES encrypted with K_S. ENC shall be DES or 3DES depending on the selection in Option Register.*

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
69 82h	PIN not submitted prior to issuing this command.
69 85h	Mutual Authentication not completed immediately prior to this command.
69 66h	Command not available; option bit not set.

5.6. GET CARD INFO

This command is used to get the card's serial number, size of user EEPROM or the revision information. These data are also available in MCD-ID file, FF 00h.

Command

CLA	INS	P1	P2	P3
80h	14h	00h 05h 06h	00h	08h 00h 08h

Response

If P1 is 00h, the card's serial number is returned:

Data	SW1 SW2
8 byte card's serial number	Status

The card's serial number is the equivalent to the first record of MCU ID File described in **Section 3.3.1**.

If P1 is 05h, the card's size of user EEPROM is returned:

SW1 SW2
Status

If P1 is 06h, the ACOS3's revision number is returned:

Data	SW1 SW2
8 byte card's revision number	Status

The card's revision number is the equivalent to the second record of MCU ID File described in **Section 3.3.1**.

Specific Status Codes

SW1	SW2	Meaning
67	00h	Wrong P3.
6F	00h	P1 or P2 invalid.
90	XXh	If P1 is 05h, then 90 XXh is returned where XX is the size of user EEPROM in hexadecimal.

5.7. CLEAR CARD

This command is used to clear the card's data, secret codes and authentication keys, except the manufacturer file and IC code. This function is available only after a successful verification of IC code and when the card is in Manufacturing State or Personalization State.

Command

CLA	INS	P1	P2	P3
80h	30h	00h	00h	00h

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
69 82h	IC code not satisfied or card is in user state.

5.8. SELECT FILE

This command is used to select a data file for subsequent READ/WRITE RECORD/BINARY commands.

Command

CLA	INS	P1	P2	P3	DATA
80h	A4h	00h	00h	02h	File ID

Where:

File ID Two bytes file identifier.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
6A 82h	File does not exist.
90 00h	Internal Data File has been selected.
91 nnh	User Data File has been selected. The corresponding File Definition Block is stored in record no. nnh in the User File Management File (FF 04h).

5.9. READ RECORD

This command is used to read a number of bytes – from an offset up to the record length – from one record in the currently selected file.

Command

CLA	INS	P1	P2	P3
80h	B2h	Rec No.	Offset	Len

Where:

Rec No. Logical record number to be read.

0..N-1 if RECORD_NUMBERING flag in Manufacturer file is zero.

1..N if RECORD_NUMBERING flag in Manufacturer file is one.

Offset Offset from that record to start reading from.

Len Number of data bytes to be read.

Response

Data	SW1 SW2
Byte 1 ... Byte N	Status

Where:

Byte 1 ... Byte N Data bytes read from the record.

Specific Status Codes

SW1 SW2	Meaning
67 00h	Specified Len plus Offset is larger than record length.
69 81h	Command incompatible with file structure.
69 82h	Security status not satisfied – Secret code(s) not submitted.
69 85h	No file selected.
6A 83h	Record not found – file too short.
6F 00h	I/O error; data to be accessed resides in invalid address.

5.10. WRITE RECORD

This command is used to write a number of bytes – from an offset up to the record length – to one record in the currently selected file.

Command

CLA	INS	P1	P2	P3	DATA
80h	D2h	Rec No.	Offset	Len	Byte 1 ... Byte N

Where:

Rec No. Logical record number to be read.

0..N-1 if RECORD_NUMBERING flag in Manufacturer file is zero.

1..N if RECORD_NUMBERING flag in Manufacture file is one.

Offset Offset from that record to start writing from.

Len Number of data bytes to be written to the record Rec No.

Byte 1… Byte N Data bytes to be written.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
67 00h	Specified Len plus Offset is larger than record length.
69 81h	Command incompatible with file structure.
69 82h	Security status not satisfied – Secret code(s) not submitted.
69 85h	No file selected.
6A 83h	Record not found – file too short.
6F 00h	I/O error; data to be accessed resides in invalid address.

5.11. READ BINARY

This command is used to read a number of bytes – from an offset up to the record length – from one record in the currently selected file.

Command

CLA	INS	P1	P2	P3
80h	B0h	Offset High	Offset Low	Len

Where:

Offset Offset from the start of file to start reading from. High is the most significant byte, while Low is the least significant.

Len Number of data bytes to be read.

Response

Data	SW1 SW2
Byte 1 ... Byte N	Status

Where:

Byte 1 ... Byte N Data bytes read from the record.

Specific Status Codes

SW1 SW2	Meaning
67 00h	Specified Len plus Offset is larger than file length.
69 81h	Command incompatible with file structure.
69 82h	Security status not satisfied – Secret code(s) not submitted.
69 85h	No file selected.
6A 83h	File too short – Offset is larger than the file length.
6F 00h	I/O error; data to be accessed resides in invalid address.

5.12. WRITE BINARY

This command is used to write a number of bytes – from an offset up to the file length – to the currently selected file.

Command

CLA	INS	P1	P2	P3	DATA
80h	D0h	Offset High	Offset Low	Len	Byte 1 ... Byte N

Where:

Offset Offset from the start of file to start writing from. High is the most significant byte while Low is the least significant.

Len Number of data bytes to be written.

Byte 1 ... Byte N Data bytes to be written.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
67 00h	Specified Len plus Offset is larger than file length.
69 81h	Command incompatible with file structure.
69 82h	Security status not satisfied – Secret code(s) not submitted.
69 85h	No file selected.
6A 83h	File too short – Offset is larger than the file length.
6F 00h	I/O error; data to be accessed resides in invalid address.

5.13. INQUIRE ACCOUNT

5.13.1. INQUIRE ACCOUNT For Contact

This command is used to read the relevant information from the *Account*.

Command

CLA	INS	P1	P2	P3	DATA
80h	E4h	Key No.	00h	04h	Reference

Where:

Key No. Reference to the key to be used in the calculation of the MAC cryptographic checksum.

Reference Four bytes arbitrary reference data.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
6A 86h	Key No. is invalid.
69 85h	Mutual Authentication has not been completed.
61 19h	Issue GET RESPONSE with P3 = 19h

To get the response, execute the GET RESPONSE command:

Command

CLA	INS	P1	P2	P3
80h	C0h	00h	00h	19h

Response

Data							SW1 SW2
MAC4	Trans. Type	Balance	ATREF	Max. Balance	TTREF-C	TTREF-D	Status

Table 9: Inquire Account Response

Where:

MAC_4 First 4 bytes of MAC cryptographic checksum on the account data and the reference.

Trans. Type One byte coding the type of the most recent transaction.

Balance Three bytes current balance value.

ATREF Six bytes Account Transaction Reference.

Max. Balance	Three bytes maximum allowed balance value.
TTREF-C	Four bytes Terminal Transaction Reference – Credit.
TTREF-D	Four bytes Terminal Transaction Reference – Debit.

Specific Status Codes

SW1 SW2	Meaning
69 85h	No data available; the INQUIRE ACCOUNT command was not executed immediately prior to the GET RESPONSE command.
62 81h	Account data may be corrupted.

5.13.2. INQUIRE ACCOUNT For Contactless

This command is used to read the relevant information from the *Account*.

Command

CLA	INS	P1	P2	P3	DATA
80h	E4h	Key No.	00h	04h	Reference

Where:

Key No.	Reference to the key to be used in the calculation of the MAC cryptographic checksum.
Reference	Four bytes arbitrary reference data.

Response

Data							SW1 SW2
MAC4	Trans. Type	Balance	ATREF	Max. Balance	TTREF-C	TTREF-D	Status

Table 10: Inquire Account Response

Where:

MAC_4	First 4 bytes of MAC cryptographic checksum on the account data and the reference.
Trans. Type	One byte coding the type of the most recent transaction.
Balance	Three bytes current balance value.
ATREF	Six bytes Account Transaction Reference.
Max. Balance	Three bytes maximum allowed balance value.
TTREF-C	Four bytes Terminal Transaction Reference – Credit.
TTREF-D	Four bytes Terminal Transaction Reference – Debit.

Specific Status Codes

SW1 SW2	Meaning
6A 86h	Key No. is invalid.
69 85h	Mutual Authentication has not been completed.

SW1 SW2	Meaning
90 00h	Command executed successfully

5.14. CREDIT

This command is used to apply credit to the *Account*.

Command

CLA	INS	P1	P2	P3	DATA
80h	E2h	00h	00h	0Bh	MAC : Amount : TTREF

Where:

MAC Four bytes MAC cryptographic checksum on the command.

Amount Three bytes value of amount to be credited.

TTREF Four bytes Terminal Transaction Reference.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
69 F0h	Data in account is inconsistent – no access unless in Issuer Mode.
6A 82h	Account does not exist.
6F 10h	Account Transaction Counter at maximum – no more transaction is possible.
63 Cnh	MAC cryptographic checksum is wrong; n = remaining number of retries
6B 20h	Amount is too large.
69 83h	Credit Key is locked.
69 85h	Mutual Authentication has not been completed.

5.15. DEBIT

This command is used to debit the *Account*.

Command

CLA	INS	P1	P2	P3	DATA
80h	E6h	00h 01h	00h	0Bh	MAC : Amount : TTREF

If P1 is 01h, ACOS3 will return a 4-byte Debit Certificate.

MAC	Four bytes MAC cryptographic checksum on the command.
Amount	Three bytes value of amount to be debited.
TTREF	Four (4) bytes Terminal Transaction Reference.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
69 F0h	Data in account is inconsistent – no access unless in Issuer Mode.
6A 82h	Account does not exist.
6F 10h	Account Transaction Counter at maximum – no more is transaction possible.
63 Cnh	MAC cryptographic checksum is wrong; n = remaining number of retries
69 82h	Security status not satisfied – PIN not submitted.
6B 20h	Amount too large.
69 82h	PIN is not submitted.
69 83h	Debit Key is locked.
69 85h	Mutual Authentication has not been completed.
61 04h	Debit successful, issue GET RESPONSE with P3=04h to get Debit Certificate.

5.16. REVOKE DEBIT

This command is used to revoke the most recent DEBIT command.

Command

CLA	INS	P1	P2	P3	DATA
80h	E8h	00h	00h	04h	MACh

Where:

MAC Four bytes MAC cryptographic checksum on Balance, TTREF-D, ATREF.

Response

SW1 SW2
Status

Specific Status Codes

SW1 SW2	Meaning
69 F0h	Data in account is inconsistent – no access unless in Issuer Mode.
6A 82h	Account does not exist.
6F10h	Account Transaction Counter at maximum – no more transaction is possible.
63 Cnh	MAC cryptographic checksum is wrong; n = remaining number of re-retries
69 F0h	Data in account is inconsistent – no access unless in Issuer Mode.
69 66h	Command not available (option bit not set).
69 83h	Revoke Debit Key is locked.

6.0. Card Personalization

This section describes the general procedure in the personalization of an ACOS3 Combi card. While the card personalization may be carried out in separate processing steps, the personalization process generally requires the execution of the steps described below.

The personalization of a new ACOS3 Combi card is suggested to be carried out according to the following sequence:

1. Power up and reset the card
2. Submit the default Issuer Code IC (the code is communicated to the card issuer by ACS; the code may be different for different batches of cards supplied)
3. Select the Security File (File ID = FF 03h) and write the required secret codes and authentication keys.

   Caution: *It is important to not lose these codes and keys. If the codes and keys (especially the issuer code) cannot be verified, the card may become locked.*
4. Select the Personalization File (File ID = FF 02h) and write the required settings to the Option Register and the parameter N_OF_FILE.

   Caution: *Do not accidentally set the Personalization Bit and do not change the Security Option Register at this stage.*
5. Select the User File Management File (File ID = FF 04h) and write the File Definition Blocks for the required User Files (WRITE RECORD command) with the required security attributes.
6. Select the individual User Files, submit the required code verification, and initialize the data in the files as required (WRITE RECORD command).
7. If applicable, select the Account File (File ID = FF 05h) and initialize the relevant data in the Account File (WRITE RECORD command). Verify the contents of the Account File (READ RECORD command).
8. If applicable, select the Account Security File (File ID = FF 06h) and initialize the account processing keys (WRITE RECORD command). Verify the contents of the Account Security File (READ RECORD command).
9. Select the Security File (File ID = FF 03h) and initialize all keys and codes (WRITE RECORD command). Verify the contents of the Security File (READ RECORD command)
10. Select the Personalization File (File ID = FF 02h) and initialize the Security Option Register and the remaining bytes of the Personalization File. Set the Personalization Bit (WRITE RECORD command). Verify the contents of the Personalization File (READ RECORD command).

    Caution: *Do not accidentally change the value of the Option Register and N_OF_FILE.*
11. Perform a card reset. The chip life cycle state as indicated in the ATR should be ‘User State.’
12. The correct personalization can be verified by submitting the secret codes and keys programmed in the card (AUTHENTICATE, SUBMIT CODE commands) and reading/writing the allocated data files and executing the Account commands.

6.1. Sample Card Personalization

The following is an example of a simple card personalization. Depending on the design of the application, the personalization may be different. It may be easier to build customized personalization using the Card Tools application provided with ACS software development kits.

This simple card personalization creates:

- One record file 41 00h of 10 records and 255 bytes in length with free read/AC1 write access
- One binary file 41 01h of 1 kilobyte in length with PIN read/write access
- Activate the purse file with 0 value

```
; Reset and get ATR
3B BE 11 00 00 41 01 38 00 00 00 00 00 00 00 00 01 90 00

; submit default IC code
80 20 07 00 08 41 43 4F 53 54 45 53 54 (9000)

; Select and write the security codes
80 A4 00 00 02 FF 03 (9000)
; Write secret issuer code
80 D2 00 00 08 <8-byte IC>(9000)
; Write secret PIN code
80 D2 01 00 08 <8-byte PIN> (9000)
; Write secret application code 1 (AC1)
80 D2 05 00 08 <8-byte AC1> (9000)

;Select and write personalization file
80 A4 00 00 02 FF 02 (9000)
; Write 2 for N_OF_FILES
80 D2 00 02 01 02 (9000)

; Select and write user file management file
80 A4 00 00 02 FF 04 (9000)
; Write the two files
80 D2 00 00 07 FF 0A 00 02 41 00 00 (9000)
80 D2 01 00 07 04 00 60 60 41 01 80 (9000)

; Select and write personalization data to file 4100H
80 A4 00 00 02 41 00 (9100)
; Submit AC1
80 20 01 00 08 <8-byte AC1> (9000)
; Write some personalization data to the records
80 D2 00 00 XX <Some data> (9000)

; Initialize the purse file
80 A4 00 00 02 FF 02 (9000)
80 D2 00 00 01 01 (9000)

80 A4 00 00 02 FF 05 (9000)
80 D2 00 00 04 00 00 00 00 (9000)
80 D2 01 00 04 00 00 01 00 (9000)
80 D2 02 00 04 00 00 00 00 (9000)
80 D2 03 00 04 00 00 01 00 (9000)
80 D2 04 00 04 00 03 E8 00 (9000)

; Initialize the purse key file
80 A4 00 00 02 FF 06 (9000)
80 D2 00 00 08 <8-byte debit key> (9000)
80 D2 01 00 08 <8-byte credit key> (9000)
80 D2 02 00 08 <8-byte certify key> (9000)
```

```
80 D2 03 00 08 <8-byte revoke debit key> (9000)

; After full testing, lock the card to user state
80 A4 00 00 02 FF 02 (9000)
80 D2 00 03 01 80 (9000)
```

Appendix A. Response Status Codes

The following is a summary of the status codes returned by the card.

SW1 SW2	Meaning
90 00h	Command executed successfully.
91 nn	User Data File has been selected. The corresponding File Definition Block is stored in record no. nn in the User File Management File (FF 04h).
61 nn	Command Successful – Issue GET RESPONSE command with P3 = nn to get response data.
62 81h	Data returned in response to the INQUIRE ACCOUNT command may be incorrect due to corrupted data in the Account Data Structure.
63 Cnh	Security related command failed – EXTERNAL AUTHENTICATION failed; incorrect Secret Code submitted; incorrect key used in CREDIT MAC generation; n = number of remaining trials
67 00h	Wrong P3.
68 82h	Secure messaging not allowed.
69 66h	Command not available (Manufacturing State, option bit not set, etc.).
69 81h	Command incompatible with file structure.
69 82h	Security status not satisfied. • Secret Code, Issuer Code or PIN not submitted. • Secure Messaging required but not present.
69 83h	Key or Secret Code is locked - no more verification or submission possible.
69 85h	Conditions of use not satisfied - no data for GET RESPONSE command available; CREDIT/DEBIT command executed without previous START TRANSACTION; Mutual Authentication not completed; no file selected; Session Key not established for Secure Messaging; Secure Messaging Command expected.
69 87h	Expected secure messaging data objects missing.
69 88h	The Secure Messaging MAC_{cmd} does not match the data.
69 F0h	Account data inconsistent/transaction interrupted - access to account only in privileged mode possible.
6A 82h	File does not exist; account not available.
6A 83h	Record not found - file too short.
6A 86h	P1-P2 is incorrect.
6B 20h	Invalid amount in CREDIT/DEBIT command.
6C nn	Issue GET RESPONSE command with P3 = nn to get response data.
6D 00h	Unknown INS.
6E 00h	Invalid CLA.
6F 00h	I/O error; data to be accessed resides in invalid address.

SW1 SW2	Meaning
6Fh	Account Transaction Counter at maximum - no more DEBIT or CREDIT transaction is possible.

Table 11: Response Status Codes

Appendix B. Creating User File

Example 1: Create 1 user file data with a File ID of 10 00 containing 5 records. Each record has a length of 4 bytes and has Free R/W access.

Procedure	APDU Command
Submit Issuer Code (IC). Default IC of ACOS3 Contact card is 41 43 4F 53 54 45 53 54 (ACOSTEST).	80 20 07 00 08 41 43 4F 53 54 45 53 54h
Select Personalization File (FF 02)	80 A4 00 00 02 FF 02h
Set N_OF_FILE byte in the personalization file (Refer to **Figure 4**). This value specifies the number of user data file the will be created. The maximum number of files allowed in ACOS3 is 64.	80 D2 00 00 04 00 00 01 00h
Select User File Management File (FF 04)	80 A4 00 00 02 FF 04h
Set the following bytes in the User File Definition Block: (Refer to **Figure 8**) Byte 1 = 04h(Record Length) Byte 2 =05h (Number of Records) Byte 3 =00h (Read Security Attributes. Refer to **Figure 2**) Byte 4 =00h (Write Security Attribute. Refer to **Figure 2**) Byte 5 =10h (File ID MSB) Byte 6 =00h (File ID LSB) Byte 7=00h (File Access Flag Refer to **Table 4)**	80 D2 00 00 07 04 05 00 00 10 00 00h

Table 12: Commands in creating User File (Example 1)

Example 2: Add another user file data with a File ID of 20 00 containing a 128-byte Binary File. Read Access: PIN, Write Access: IC. (Note: Previous user file data must not be overwritten).

Procedure	APDU Command
Submit Issuer Code (IC). Default IC of ACOS3 Contact card is 41 43 4F 53 54 45 53 54 (ACOSTEST).	80 20 07 00 08 41 43 4F 53 54 45 53 54h
Select Personalization File (FF 02)	80 A4 00 00 02 FF 02h

Procedure	APDU Command
Set N_OF_FILE byte in the personalization file (Refer to **Figure 4**). This value specifies the number of user data file the will be created. The maximum number of files allowed in ACOS3 is 64.	80 D2 00 00 04 00 00 02 00h
Select User File Management File (FF 04)	80 A4 00 00 02 FF 04h
Set the following bytes in the User File Definition Block: (Refer to **Figure 8**) Byte 1 = 00h(File Length High Byte) Byte 2 =08h(File Length Low Byte) Byte 3 =40h (Read Security Attribute Refer to **Figure 2**) Byte 4 =80h (Write Security Attribute Refer to **Figure 2**) Byte 5 =20h (File ID MSB) Byte 6 =00h (File ID LSB) Byte 7=80h (File Access Flag) Refer to **Table 3**	80 D2 01 00 07 00 80 40 80 20 00 80h

Table 13: Commands in creating User File (Example 2)

Appendix C. Creating Account

Example 1: Create an account with the following feature: PIN change, 3DES Encryption, MAC required before Debit transaction, Revoke Debit. Set the PIN to 123456(ASCII). Initialize Check Sum. SetMaximum Balance to 150,000.

Procedure	APDU Command
Submit Issuer Code (IC). Default IC of ACOS3 Contact card is 41 43 4F 53 54 45 53 54 (ACOSTEST).	80 20 07 00 08 41 43 4F 53 54 45 53 54h
Select Personalization File (FF 02)	80 A4 00 00 02 FF 02h
Set the Option Register byte in the personalization file (Refer to **Figure 4**). The ACCOUNT bit must be set to HIGH to enable account feature. Set all the option register bits required to HIGH. Bit 0 = ACCOUNT Bit 1 = 3DES Bit 2 =PIN_ALT Bit 3 = DEB_MAC Bit 4 = DEB_PIN Bit 5 = REV_DEB Bit 6 = TRNS_AUT Bit 7 = INQ_AUT (Refer to **Figure 5)**	80 D2 00 00 04 2F 00 00 00h
Select Security File (FF 03)	80 A4 00 00 02 FF 03h
Submit Issuer Code (IC). Default IC of ACOS3 Contact card is 41 43 4F 53 54 45 53 54 (ACOSTEST).	80 20 07 00 08 41 43 4F 53 54 45 53 54h
Write 123456(ASCII) in record 1	80 D2 01 00 08 31 32 33 34 35 36 00 00h
Select Account File (FF 05)	80 A4 00 00 02 FF 05h
Submit Issuer Code (IC). Default IC of ACOS3 Contact card is 41 43 4F 53 54 45 53 54 (ACOSTEST).	80 20 07 00 08 41 43 4F 53 54 45 53 54h
Initialize Checksum in records1 and 3. Write 01h in byte 3.(Initial Checksum should be 01h)	80 D2 01 00 04 00 00 01 00h 80 D2 03 00 04 00 00 01 00h
Set Maximum balance to 150,000. Write 00 02 49 F0 (150,000 in HEX) in record 4.	80 D2 04 00 04 00 02 49 F0h

Table 14: Commands in creating an Account

Appendix D. Calculation of MAC

Procedure	APDU Command
Issue Inquire Account command using certify key. (02h)	80 E4 02 00 04 00 00 00 00h
Issue Get Response command to get all relevant account information needed. For example, getting the current balance will let you know the value that will be DEDUCTED for the debit command is valid.	80 C0 02 00 19h
Retrieve and interpret the received bytes. (Refer to **Table 9**)	2A C6 83 36 01 00 03 3D 00 00 00 00 00 01 02 49 F0 00 00 00 00 00 00 00 00h (Example Only)

Table 15: Commands in calculating the MAC

Account File Information	Response Data
MAC4	2A C6 83 36h
Trans. Type	01h
Balance	00 03 3Dh
ATREF	00 00 00 00 00 01h
Max. Balance	02 49 F0h
TTREF-C	00 00 00 00h
TTREF-D	00 00 00 00h

Table 16: Inquire Account Response Data

Example 1: Credit MAC Calculation. The sixteen bytes data string on which the MAC cryptographic checksum is calculated is composed as follows:

1st Block			2nd Block		
E2	Amount	TTREF-C	ATREF + 01	00	00

Example 2: Debit MAC Calculation. The sixteen bytes data string on which the MAC cryptographic checksum is calculated is composed as follows:

1st Block			2nd Block		
E6	Amount	TTREF-C	ATREF + 01	00	00

Example 3: Revoke Debit MAC Calculation. The sixteen bytes data string on which the MAC cryptographic checksum is calculated is composed as follows:

1st Block			2nd Block		
E8	Amount	TTREF-C	ATREF + 01	00	00

(Same process for example 1, 2 and 3)

Prepare MAC data block: E2+ Amount + TTREF-C + ATREF + 00 + 00.

Store MAC data block intmpArray[32].

Store the Credit Key/Debit key in tmpKey[8] (if DES operation) or tmpKey[16] (if 3DES operation).

(Same process for example 1, 2 and 3)

Prepare MAC data block: **E2+ Amount + TTREF-C + ATREF + 00 + 00.**

Store MAC data block in**tmpArray[32].**

Store the Credit Key/Debit key in **tmpKey[8]** (if DES operation) or **tmpKey[16]** (if 3DES operation).

Use the following code for the encryption process of MAC. Use tmpArray as the data block when executing **Credit / Debit / Revoke Debit** command.

```
//DES Encryption

DES (tmpArray, tmpKey); //Encryption Process

        for (indx = 0;indx<8;indx++)
        {
        .............tmpArray[indx] = tmpArray[indx] ^= tmpArray[indx + 8];
        }

DES (tmpArray, tmpKey); //Encryption Process


//3DES Encryption

DES3 (tmpArray, tmpKey); //Encryption Process

        for (indx = 0;indx<16;indx++)
        {
        .............tmpArray[indx] = tmpArray[indx] ^= tmpArray[indx + 8];
        }

DES3 (tmpArray, tmpKey); //Encryption Process
```

Appendix E. Executing Credit Command

Example 1: Add 1,500Load. New balance should be 1,500. Use 3DES encryption to calculate the MAC needed for the operation.

Procedure	APDU Command
Issue Inquire Account command using certify key. (02h)	80 E4 02 00 04 00 00 00 00h
Issue Get Response command to get all relevant account information needed. For example, getting the current balance will let you know the value that will be ADDED for the credit command is valid.	80 C0 02 00 19h
Issue Credit command assuming that one knows the correct set/s of data to be sent CREDIT MAC: AMOUNT: TTREF (Refer to **Appendix D** for MAC calculation).For this example, the MAC computed is 5F FE A4 14. Convert 1,500 to 3 bytes HEX (00 05 DCh). Default TTREF is 00 00 00 00h.	80 E2 00 00 0B 5F FE A4 14 00 05 DC 00 00 00 00h
Issue Inquire Account command using certify key. (02h)	80 E4 02 00 04 00 00 00 00h
Issue Get Response command to get all relevant account information needed. For example, getting the current balance will let you know the value that will be ADDED for the credit command is valid.	80 C0 02 00 19h
Retrieve and interpret the received bytes. (Refer to **Table 9**) Current Balance: 00 05 DCh (1,500)	20 AF 12 56 03 00 05 DC 00 00 00 00 00 01 02 49 F0 00 00 00 00 00 00 00 00h

Table 17: Executing Credit Commands

Appendix F. Executing Debit Command

Example 1: Deduct 671Load. New balance should be 829.Use 3DES encryption to calculate the MAC needed for the operation.

Procedure	APDU Command
Issue Inquire Account command using certify key. (02h)	80 E4 02 00 04 00 00 00 00h
Issue Get Response command to get all relevant account information needed. For example, getting the current balance will let you know the value that will be DEDUCTED for the debit command is valid.	80 C0 02 00 19h
Issue Debit command assuming that one knows the correct set of data to be sent DEBIT MAC: AMOUNT: TTREF (Refer to **Appendix D** for MAC calculation). For this example, the MAC computed is E5 A4 D536. Convert 671 to 3 bytes HEX (00 02 9Fh). Default TTREF is 00 00 00 00h.	80 E6 00 00 0B E5 A4 D5 36 00 02 9F 00 00 00 00h
Issue Inquire Account command using certify key. (02h)	80 E4 02 00 04 00 00 00 00h
Issue Get Response command to get all relevant account information needed. For example, getting the current balance will let you know the value that will be ADDED for the credit command is valid.	80 C0 02 00 19h
Retrieve and interpret the received bytes. (**Table 9**) Current Balance: 00 03 3Dh (829)	2A C6 83 36 01 00 03 3D 00 00 00 00 00 01 02 49 F0 00 00 00 00 00 00 00 00h

Table 18: Executing Debit Commands

Appendix G. Executing Revoke Debit Command

Example 1: Execute revoke debit command. Calculate the MAC to be submitted. The amount needed for the MAC Calculation should be equal to the last valid balance on ACOS3 card which is 1,500 (00 05 DC). Use 3DES encryption to calculate the MAC needed for the operation.

Procedure	APDU Command
Issue RevokeDebit command assuming that one knows the correct set of data to be sent REVOKE DEBIT MAC (Refer to **Appendix D** for MAC calculation). For this example, the MAC computed is 4C 54 59 F3h.	80 E8 00 00 04 4C 54 59 F3h
Issue Inquire Account command using certify key. (02h)	80 E4 02 00 04 00 00 00 00h
Retrieve and interpret the received bytes. (Refer to **Table 9**) Current Balance: 00 05 DCh (1,500)	3F 60 A9 B5 02 00 05 DC 00 00 00 00 00 01 02 49 F0 00 00 00 00 00 00 00 00h

Table 19: Executing Revoke Debit Commands

Appendix H. Checking e-Purse Balance

Example 1: Read Current Balance and Maximum Balance.

Procedure	APDU Command
Issue Inquire Account command using certify key. (02h)	80 E4 02 00 04 00 00 00 00h
Issue Get Response command to get all relevant account information needed.	80 C0 02 00 19h
Retrieve and interpret the received bytes. (Refer to **Table 9**) Current Balance: 00 03 3Dh Maximum Balance: 02 49 F0h	2A C6 83 36 01 00 03 3D 00 00 00 00 00 01 02 49 F0 00 00 00 00 00 00 00 00h (Example Only)

Table 20: Checking e-Purse Balance Commands