Advanced Card Systems Ltd.

# ACOS3 Smart Card


Version 3.2 08-2007

Unit 1008, 10th Floor, Hongkong International Trade and Exhibition Centre
1 Trademart Drive, Kowloon Bay, Hong Kong
Tel: +852 2796 7873 Fax: +852 2796 1286 Email: info@acs.com.hk Website: www.acs.com.hk

# CONTENTS

# 1 INTRODUCTION

## *1.1 SCOPE*

The purpose of this document is to describe in detail the features and functions of the **A**CS Smart **C**ard **O**perating **S**ystems Version 3 (ACOS3) developed by Advanced Card Systems Ltd.

## *1.2 Features*

- Full 16 Kbytes of EEPROM memory for application data
- Compliance with ISO 7816 parts 1, 2,3; supporting the T=0 direct protocol
- ISO7816- 2 compliant 6-contact module.
- High-speed transmission possible (9.6 Kbps to 230.4 Kbps) with modifiable ATR.
- DES and MAC capabilities
- 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
- Linear files with fixed record length; record length can be different for different files
- Account data structure for highly secure payment applications as an optional function
- Completely backward compatible with ACOS1 / ACOS2 cards.

## *1.3 Technical Specifications*

### Electrical

- Operating at 5V DC+/-10 %( Class A) and 3V DC +/-10% (Class B)
- Maximum supply current: <10 mA
- ESD protection: ≤ 4 KV

### EEPROM

- Capacity: 16Kbytes
- EEPROM endurance: 100K Erase/Write cycles
- Data retention: 10 years

**Environmental**

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

## *1.4 History of Modification*

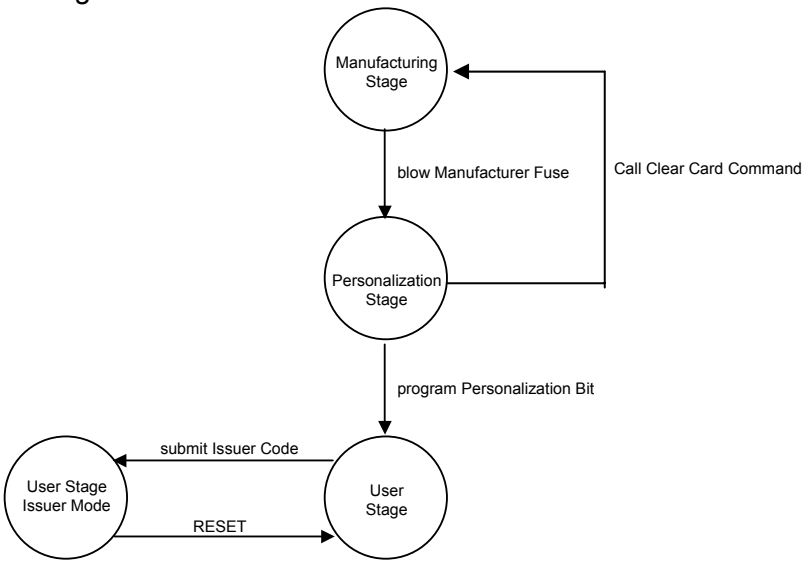
| | |
|---|---|
| December 2005 | ACOS3 revision 1.0 |
| April 2007 | ACOS3 revision 1.01<br>- Modification of the ATR for backward compatibility |
| August 2007 | ACOS3 revision 1.03<br>- Extra ISO7816-3 Guard time added to accommodate more readers<br>- Enhancement added for 230.4 kbps support |

# 2 CHIP LIFE CYCLE

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

- Manufacturing Stage
- Personalization Stage
- User Stage
- User Stage - Issuer Mode

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




The actual chip life cycle stage is determined by the card operating system immediately after a reset. The life cycle stage does not change during the operation of the card. Clear card command can be issued in the personalization stage and manufacture state to clear the card of all data. However, this command will not be able to be used after User Stage.

## *2.1 Manufacturing Stage*

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

**In the Manufacturing Stage, 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 initial IC code is programmed in the ACOS3 microcontroller during the chip manufacturing process.**

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

All card commands are available, although some of the commands, such as AUTHENTICATE will not produce reasonable results as long as the respective data, for example, the keys, have not been programmed in the card.

The following data items are written to the EEPROM memory in the Manufacturing Stage:

- The Manufacturer File, containing **2** records of **8** bytes each associated to the Manufacturing Stage. This file can only be written in the Manufacturing Stage. After programming the Manufacturer Fuse, 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.
- The IC code for the Personalization Stage.
  The IC code must have been presented to the card before the card allows WRITE access to the data files in the *Personalization Stage*, which is applicable immediately after completion of the Manufacturing Stage.
- The Manufacturer Fuse, to irreversibly change the card life cycle from the Manufacturing Stage to the Personalization Stage. The Manufacturer Fuse is one bit in the 16 bytes Manufacturer File.

## 2.2 Personalization Stage

The Personalization Stage is effective from the moment of termination of the Manufacturing Stage until an associated bit in the EEPROM, the so-called *Personalization Bit,* has been programmed.

Once the Personalization Bit has been programmed and the Personalization Stage has thus been terminated, there is no possibility of resetting the card back into the personalization stage.

During card personalization, the card can be reset back to its virgin stage by calling the CLEAR CARD command. This command will physically erase the EEPROM memory and key data so reprogramming is possible.

**In the Personalization Stage, 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 Stage.**

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 Stage.

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

- The Personalization File, containing 3 records of 4 bytes each associated to the Personalization Stage, including the Option Registers. This area can only be written in the Personalization Stage. 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 10 bytes of the Personalization File are transmitted in the Historical Bytes in the Answer-to-Reset.
- Secret Codes and Keys
- The File Definition Blocks of the required User Data Files.
- The Account Data Structure (if enabled by the respective option bit)
- The Personalization Bit to change the card life cycle from the Personalization Stage to the User Stage.

## 2.3 User Stage

*User Stage* designates the 'normal' operating mode of the card. The User Stage is effective from the moment of termination of the Personalization Stage until the so-called Issuer Code has been

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

# 3 EEPROM MEMORY MANAGEMENT

The 16K Bytes EEPROM memory area provided by the card chip is basically segregated in Internal Data Memory and User Data Memory:

- The Internal Data Memory is used for the storage of configuration data and it is used by the card operating system to manage certain functions.
- The User Data Memory stores the data manipulated in the normal use of the card under control of the application.

## *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 composed of data records. A data record is the smallest data unit that can individually be addressed in a data file. Each 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.

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 Stage, the file structure is fixed.

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

**A value of $FF_H$ 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 RECORD or WRITE RECORD access to a file, the file must be opened through the SELECT FILE command. Only one file is selected at any time. The READ RECORD and WRITE RECORD 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:

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

AC x requires Application Code x
PIN requires PIN code
IC requires Issuer Code

A Security Attribute is defined in one byte as follows:

bit 7 bit 0

| IC | PIN | AC5 | AC4 | AC3 | AC2 | AC1 | AC0 |
|---|---|---|---|---|---|---|---|

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.

## *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 stage.

The following Internal Data Files are defined:

| | | **File Security Attributes** | | | **Record Organization** |
|---|---|---|---|---|---|
| **Memory Area** | **Internal File ID** | **Manufacturing Stage** | **Personalization Stage** | **User Stage** | |
| MCU-ID File | FF $00_H$ | R: FREE<br>W: NO ACCESS | R: FREE<br>W: NO ACCESS | R: FREE<br>W: NO ACCESS | 2 x 8 bytes |
| Manufacturer File | FF $01_H$ | R: FREE<br>W: IC | R: FREE<br>W: NO ACCESS | R: FREE<br>W: NO ACCESS | 2 x 8 bytes |
| Personalization File | FF $02_H$ | R: FREE<br>W: IC | R: FREE<br>W: IC | R: FREE<br>W: NO ACCESS | 3 x 4 bytes |
| Security File | FF $03_H$ | R: IC<br>W: IC | R: IC<br>W: IC | R: NO ACCESS<br>W: IC | 12 x 8 bytes |
| User File Management File | FF $04_H$ | R: FREE<br>W: IC | R: FREE<br>W: IC | R: FREE<br>W: IC | N_OF_FILE x 6 bytes |
| Account File | FF $05_H$ | R: FREE<br>W: IC | R: FREE<br>W: IC | R: IC<br>W: IC | 8 x 4 bytes |
| Account Security File | FF $06_H$ | R: FREE<br>W: IC | R: FREE<br>W: IC | R: NO ACCESS<br>W: IC | 4 x 8 bytes |
| ATR File | FF $07_H$ | R: FREE<br>W: IC | R: FREE<br>W: IC | R: FREE<br>W: NO ACCESS | 1 X 36 bytes |
| User File Data Area | file IDs: xx $yy_H$<br>xx ≠ $FF_H$ | According to the file definitions | | | |

### 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 version number – namely ACOS3 Revision 1.01 16Kbyte EEPROM (41 43 4F 53 03 01 01 $10_H$ ).

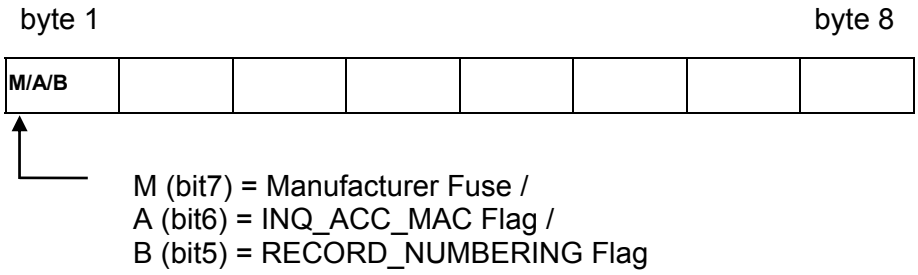
This file is always free for READ access but not WRITE accessible.

### 3.3.2 Manufacturer File

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

The termination of the Manufacturer Stage 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 Stage can never again be entered.

Manufacturer File, first record:




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. Please refer to the section of INQUIRY ACCOUNT for the details.

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).

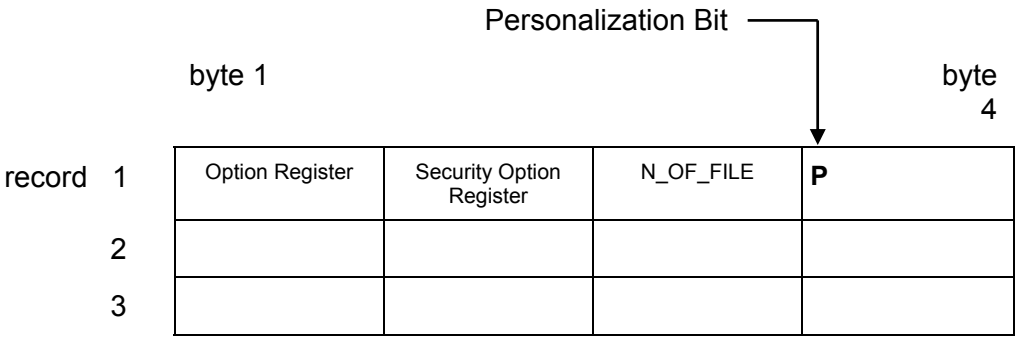
### 3.3.3 Personalization File

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

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

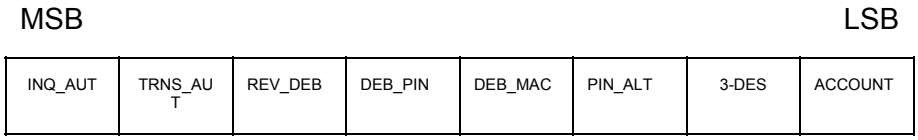
The termination of the Personalization Stage is indicated by writing a '1' into the MSB of byte 4 of the first record in the Personalization File (Personalization Bit). The change of stage will be effective immediately after the next reset of the card.

Personalization File:




The first three 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:

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




ACCOUNT This bit indicates whether 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.

3-DES This bit indicates whether the encryption is DES or 3-DES. If the bit is not set, single DES will only be performed. If the bit is set, triple DES is only supported.

PIN_ALT This bit determines whether 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 whether the DEBIT transaction must be authenticated by a MAC cryptographic checksum (see 'DEBIT'). 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 whether 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 successfully been submitted to the card (see 'DEBIT').

REV_DEB This bit determines whether the card can execute the REVOKE DEBIT command. If the bit is not set, the card will reject the REVOKE DEBIT command. (see ‘REVOKE DEBIT’)

TRNS_AUT This bit determines whether 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.**

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

MSB LSB

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| IC_DES | PIN_DES | AC5_DES | AC4_DES | AC3_DES | AC2_DES | AC1_DES | - |

These bits specify for the corresponding Secret Codes (IC, PIN, AC1...AC5), whether 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.

**For security reasons it is highly recommended that in any application the IC must be submitted in encrypted form in the User Stage!**

**NOTE: 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.**

Byte 3:

N_OF_FILE 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. The maximum number of files allowed in ACOS3 is 31.

Only the Option Registers and the bit of the Personalization Fuse are interpreted by the card operating system.

The first 8 bytes (2 records) of the Personalization File are transmitted in the Historical Bytes in the Answer-To-Reset.

### 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 the 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 Stage and the Personalization Stage of the card life cycle, after presentation of the correct IC.

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

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

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

Security File:

| | byte 1 | byte 8 |
|---|---|---|
| record 1 | Issuer Code IC | |
| 2 | PIN | |
| 3 | Authentication Card Key $K_C$ | |
| 4 | Authentication Terminal Key $K_T$ | |
| 5 | Random Number Seed for $RND_C$ | |
| 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 | Reserved for Future Use (RFU) | |
| 13 | Right half of 3-DES Authentication Card Key $K_C$ | |
| 14 | Right half of 3-DES Authentication Terminal Key $K_T$ | |

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

CNT' xxx = Backup copy of respective counter value

**NOTE: The record #11 storing *CNT xxx* must <u>not</u> be written in the card personalization. Inadvertently writing a wrong value to this record may permanently lock the card and render it useless!**

**NOTE: 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 are not used but present.**

### 3.3.5 Account File

The 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 and the memory space is available for User Data Files.

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

The Account File contents are explained in detail in section 4. *Account Transaction Processing*.

### 3.3.6 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 Stage and the Personalization Stage of the card life cycle.

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

The Account Security File can be written in the Manufacturing Stage and in the Personalization Stage 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 *4. Account Transaction Processing.*

### 3.3.7 User File Management File

The User File Management File consists of N_OF_FILE records of 6 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 Stage of the card life cycle. After termination of the Personalization Stage, 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.

### 3.3.8 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.

### 3.3.9 ATR File

The application developer can change the card's Answer-To-Reset string by setting the customized ATR into this file. The file contains 1 36-byte record. The 1$^{st}$ byte of this record allows the speed of the card to be customized. The second and subsequent bytes allow changes to the historical bytes of the ATR to change. If these fields are found to be invalid, the card OS will then use the default values in the ATR.

Please see Section 5 for more information.

## *3.4 User Data Files*

User Data Files are allocated in the Personalization Stage of the card life cycle. The data stored in a User Data File can be read through the READ RECORD command and updated through the WRITE RECORD 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 Stage. It is not possible to change the number of records of a file once any of the User Data Files has been used.

A User Data File can contain up to 255 records of max. 255 bytes record length each. User will be able to access these records as long as it fits the 16K (16,384) bytes capacity of the card.

Care must be taken by the card issuer 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 beyond the capacity of the card will result in error condition.

### 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, file length and security attributes. Each File Definition Block comprises six (6) bytes:

| byte 1 | byte 2 | byte 3 | byte 4 | byte 5 / 6 |
|---|---|---|---|---|
| Record length | Number of Records | Read Security Attribute | Write Security Attribute | File identifier |

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.

### 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 $02_H$ 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 $04_H$ 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 six bytes of each File Definition Block at once.
5. Now the User Data Files can be selected and read and written.

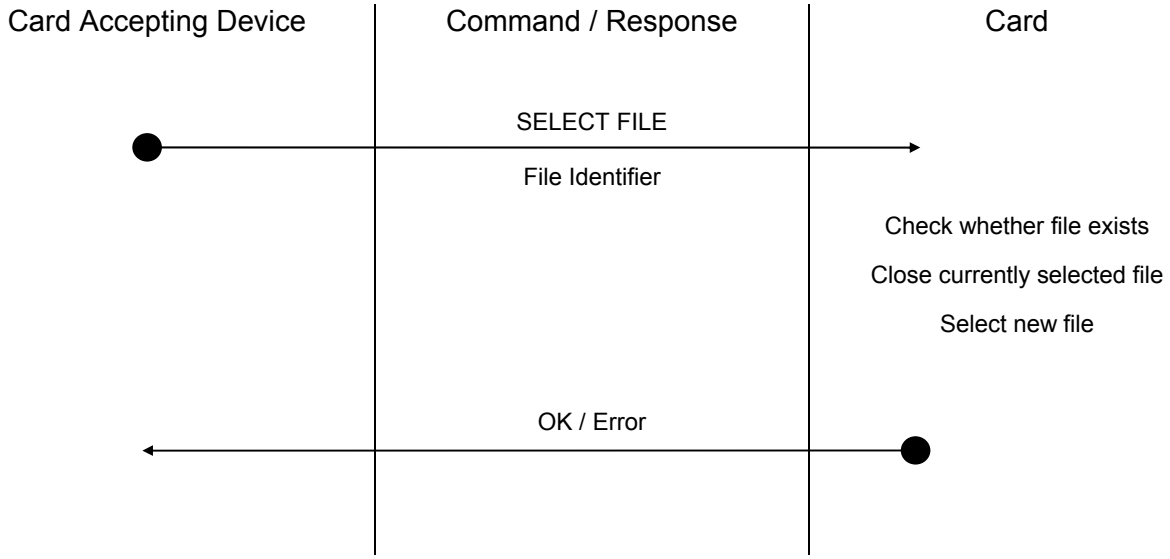
## *3.5 DATA FILE ACCESS*

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:




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

### 3.5.2 READ RECORD

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

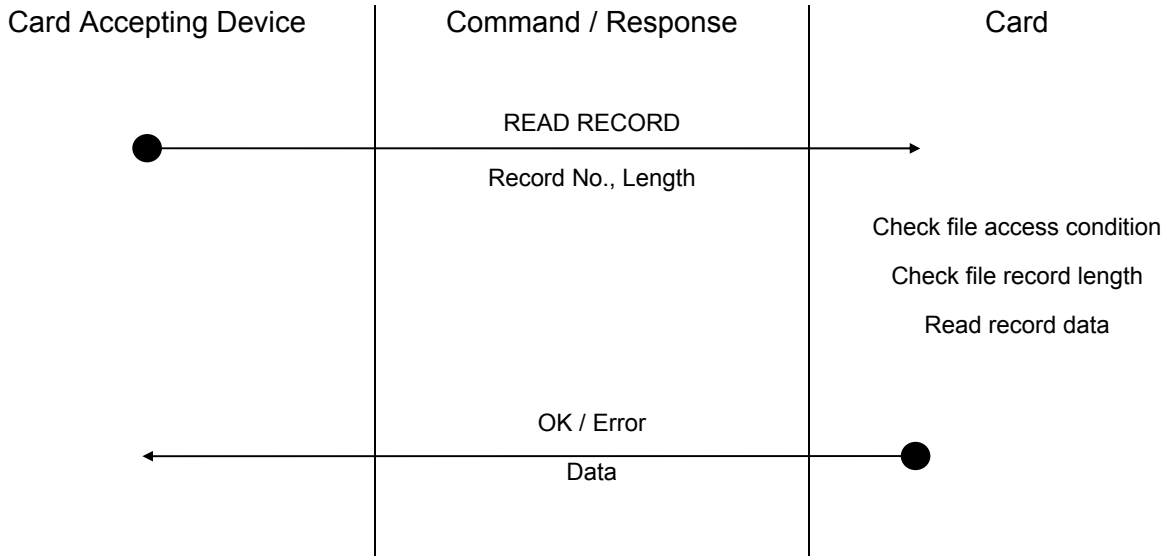
The security conditions associated to the currently selected file are checked prior to the execution of the command by the card. If the security conditions are not fulfilled (i.e., the specified secret codes have not been submitted to the card), the command is rejected by the card.

Data from only one record can be read in each READ RECORD operation. The number of bytes 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:




| | |
|---|---|
| Record No. | One byte logical record number |
| Length | Number of data bytes to be read from the record, max. 255 |
| Data | Record data, *Length* bytes |

### 3.5.3 WRITE RECORD

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

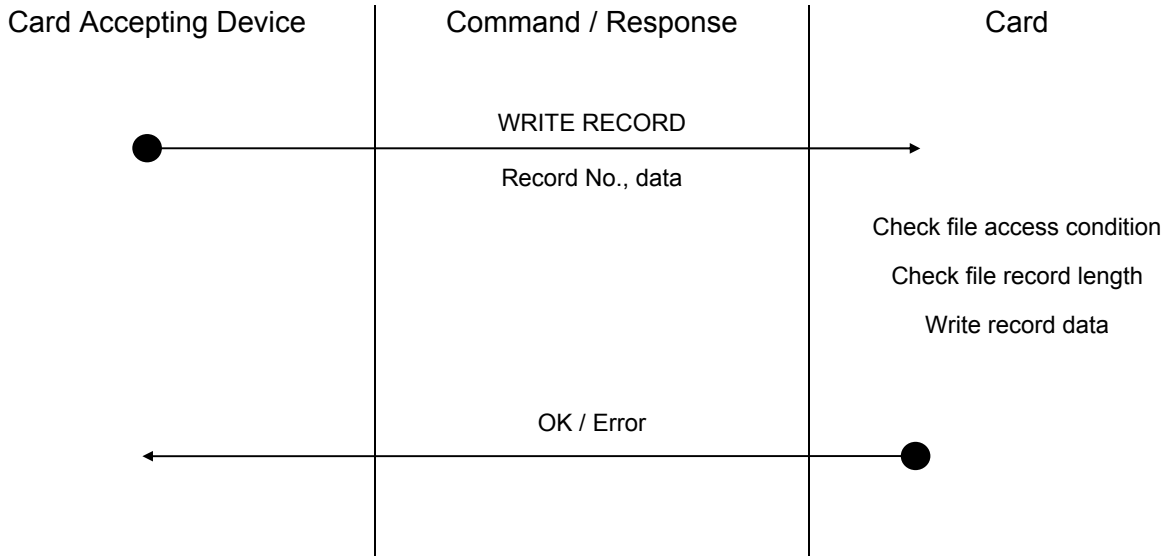
The security conditions associated to the currently selected file are checked prior to the execution of the command by the card. If the security conditions are not fulfilled (i.e., the specified secret codes have not been submitted to the card), the command is rejected by the card.

Data can be written to only one record in each WRITE RECORD operation. The number of bytes to be written in the record is specified in the command.

The maximum number of bytes to be written is equal to the record length.

If the number of bytes to be written (= N) is smaller than the record length, the first N bytes of the record are overwritten with the new data. The remaining bytes in the record are not modified.

The WRITE RECORD command is carried out as follows:




| | |
|---|---|
| Record No. | One byte logical record number |
| Data | Data bytes to be written to the record |

## *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 Stage 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'.

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

| | byte 1 | | | byte 4 |
|---|---|---|---|---|
| record 1 | TRANSTYP 0 | BALANCE 0 | | |
| 2 | ATC 0 | | CHKSUM 0 | 00 |
| 3 | TRANSTYP 1 | BALANCE 1 | | |
| 4 | ATC 1 | | CHKSUM 1 | 00 |
| 5 | MAXBAL | | | 00 |
| 6 | AID | | | |
| 7 | TTREF_C | | | |
| 8 | TTREF_D | | | |

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 ATC 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 $FF_H$), 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:

| byte 1 | byte 4 | byte 5 | byte 6 |
|---|---|---|---|
| AID | | ATC | |

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

TTREF-C The Terminal Transaction Reference - Credit is provided by the Card Accepting Device when a 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 The Terminal Transaction Reference - Debit is provided by the Card Accepting Device 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 as follows:

**Key storage for Single DES**

| Record No | Byte 1 | | | | | | | Byte 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | $K_D$ | | | | | | | |
| 2 | $K_{CR}$ | | | | | | | |
| 3 | $K_{CF}$ | | | | | | | |
| 4 | $K_{RD}$ | | | | | | | |

$K_D$ 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

**Key storage for Triple DES**

| Record No | byte 1 | | | | | | | byte 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | Right half of $K_D$ | | | | | | | |
| 2 | Right half of $K_{CR}$ | | | | | | | |
| 3 | Right half of $K_{CF}$ | | | | | | | |
| 4 | Right half of $K_{RD}$ | | | | | | | |
| 5 | Left half of $K_D$ | | | | | | | |
| 6 | Left half of $K_{CR}$ | | | | | | | |
| 7 | Left half of $K_{CF}$ | | | | | | | |
| 8 | Left half of $K_{RD}$ | | | | | | | |

$K_D$ 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 SECURITY ARCHITECTURE

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

- DES and MAC calculation
- Mutual Authentication and Session Key based on Random Numbers
- Secret Codes
- Secure Account Transaction Processing

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.

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 and MAC calculation are 8 / 16 bytes long depending on Single / Triple DES selection in *Option Register*. The most 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 based on Random Numbers*

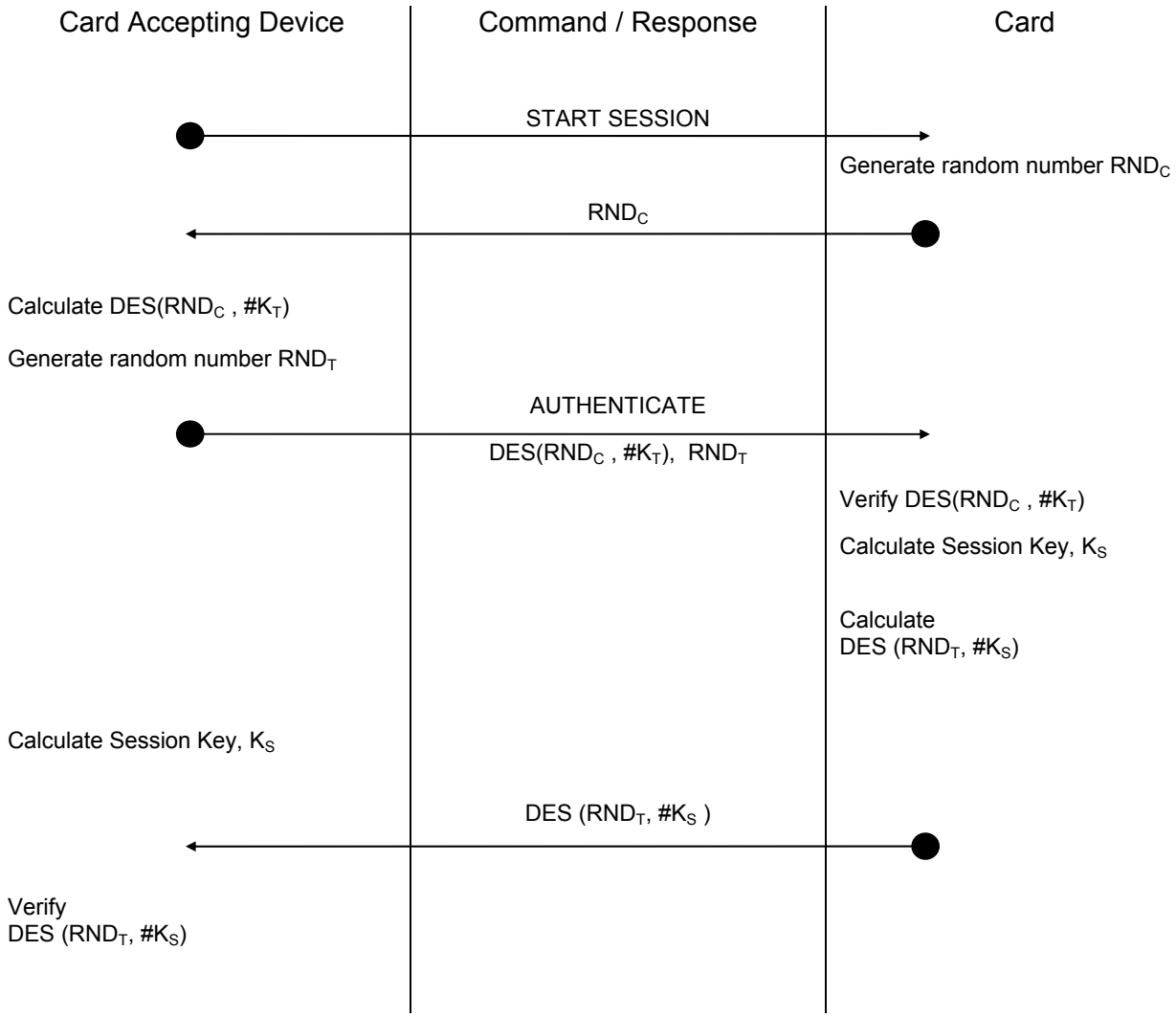
The Mutual Authentication is based on the exchange and mutual verification of secret keys between the Card and the Card Accepting Device. The key exchange is performed in a secure way by use of random numbers and DES data encryption.

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 final result of the Mutual Authentication process.

The Mutual Authentication process is carried out as follows:




NOTE: DES shall be 1-DES or 3-DES depending on *Option*

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

If single DES option has been selected

$K_S$ = 1-DES (1-DES ($RND_C$, $\#K_C$) $\oplus$ $RND_T$, $\#K_T$)

If Triple DES option has been selected,

Left half of $K_S$ = 3-DES (3-DES ($RND_C$, $\#K_C$), $\#K_T$) **&** Right half of $K_S$ = 3-DES ($RND_T$, #(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 |

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 each time the AUTHENTICATE operation fails, i.e., a wrong $K_T$ is presented to the card.
- The error counter is reset when the AUTHENTICATE operation is successful.
- If the error counter reaches a value of 8 (eight), the card will not execute the command AUTHENTICATE any longer. In this case, all related security mechanisms (e.g., the submission of Secret Codes) are blocked. **This condition is irreversible and can render 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.

The 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.

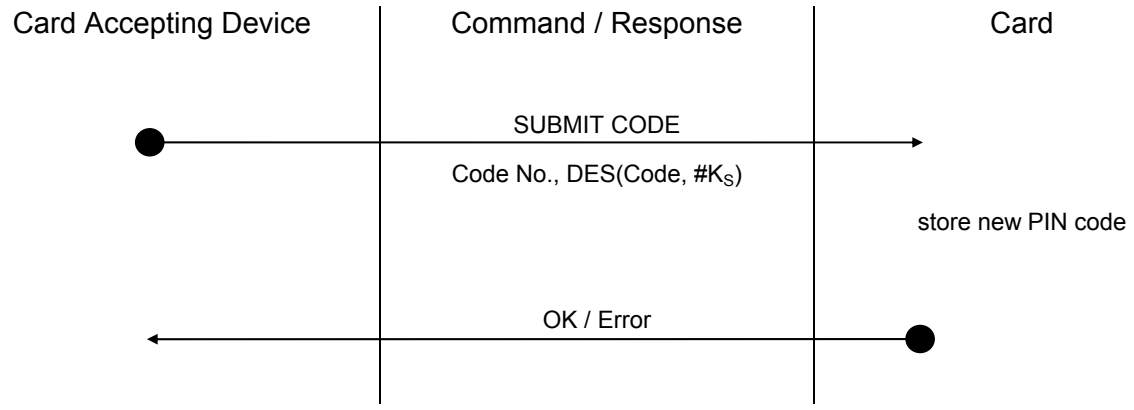
The PIN is eight bytes long. The PIN is presented to the card with the SUBMIT CODE command. Depending on the corresponding option bit PIN_DES in the Security Option Register, the PIN is DES encrypted with the current session key before the presentation to the card, or it is presented in plain.

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:




NOTE: DES shall be 1-DES or 3-DES depending on selected *Option*

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

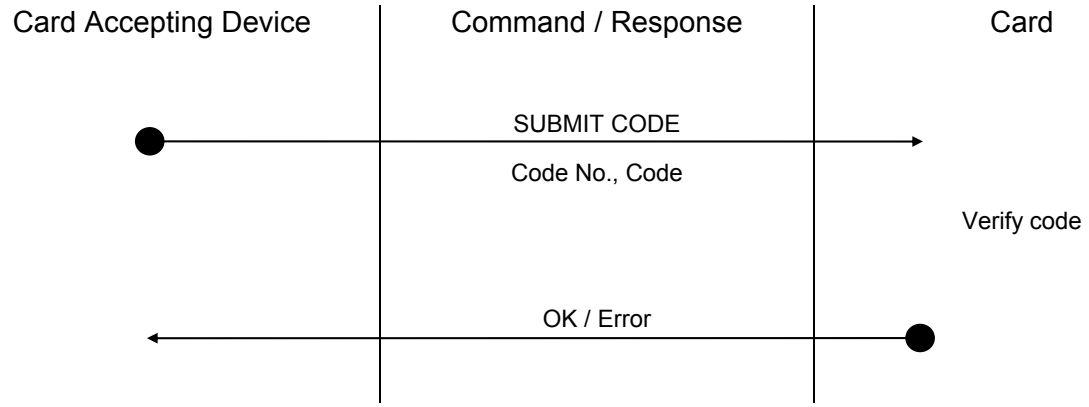
| | | |
|---|---|---|
| 1 ... 5 | = | Application Codes AC1...AC5 |
| 6 | = | PIN |
| 7 | = | Issuer Code IC |

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

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:




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 each time the SUBMIT CODE operation for that code fails, i.e., a wrong secret code is submitted to the card.
- The error counter for a particular secret code is reset 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 command SUBMIT CODE for that code.

The error counters CNT 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.

A backup copy of all error counters is kept in the Security File to prevent a corruption of this important information if an update in progress is interrupted through a reset of the card.

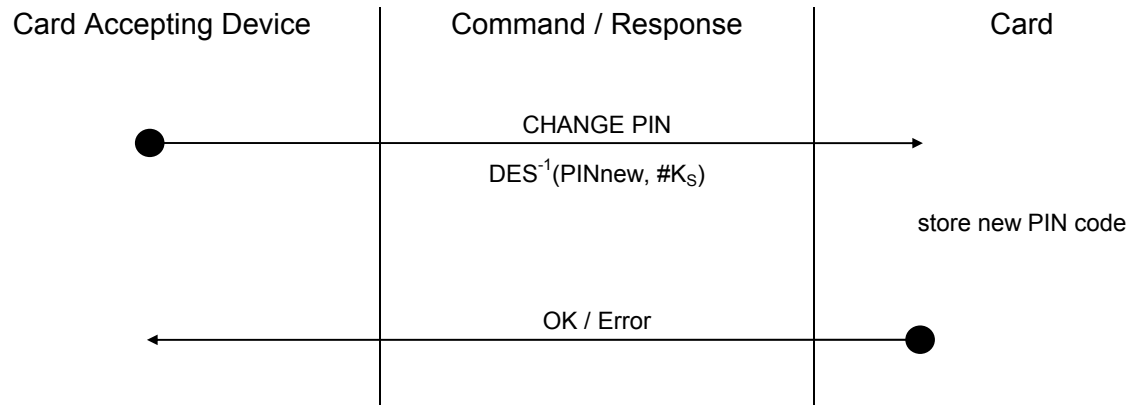
### 4.3.5 Change PIN code

The PIN code can be changed in the User stage with the command CHANGE PIN 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.

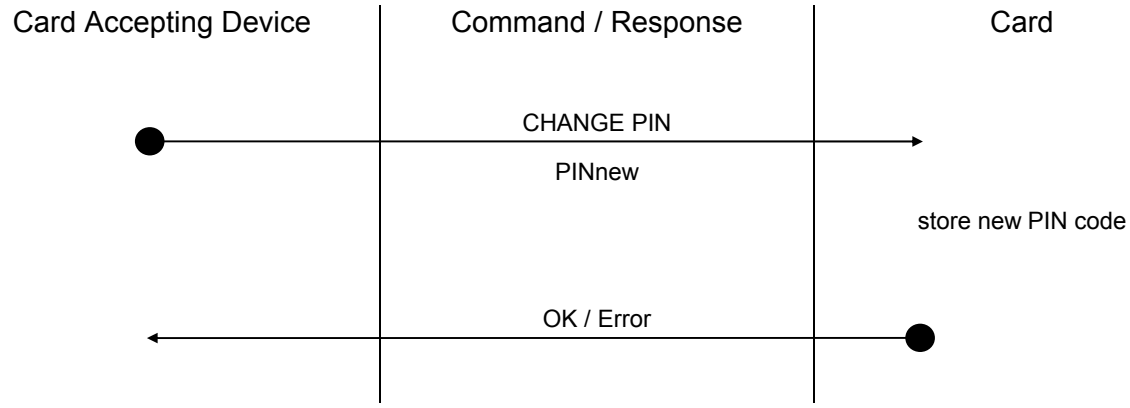
If the option bit PIN_DES is set, the changing of the PIN code is carried out as follows:




NOTE: DES shall be 1-DES or 3-DES depending on selected *Option*

| | |
|---|---|
| PINnew | The new PIN code |
| $K_S$ | The current session key |

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:



## 4.4 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 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 Stage, in the Personalization Stage and in the User Stage after presentation of the Issuer Code IC. In the normal User Stage, 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 security feature for very security critical applications, the option bits TRNS_AUT and INQ_AUT in the Options Register allow to link the Account Processing to the Mutual Authentication.

If the option bit TRNS_AUT is set, the CREDIT, DEBIT and REVOKE DEBIT commands are executed by the card only after a successful completion of the Mutual Authentication. Furthermore, the MAC cryptographic checksum used to prove the authenticity and integrity of the respective command 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 and the MAC cryptographic checksum returned by the card to prove the integrity and authenticity of the account data is encrypted with the current Session Key.

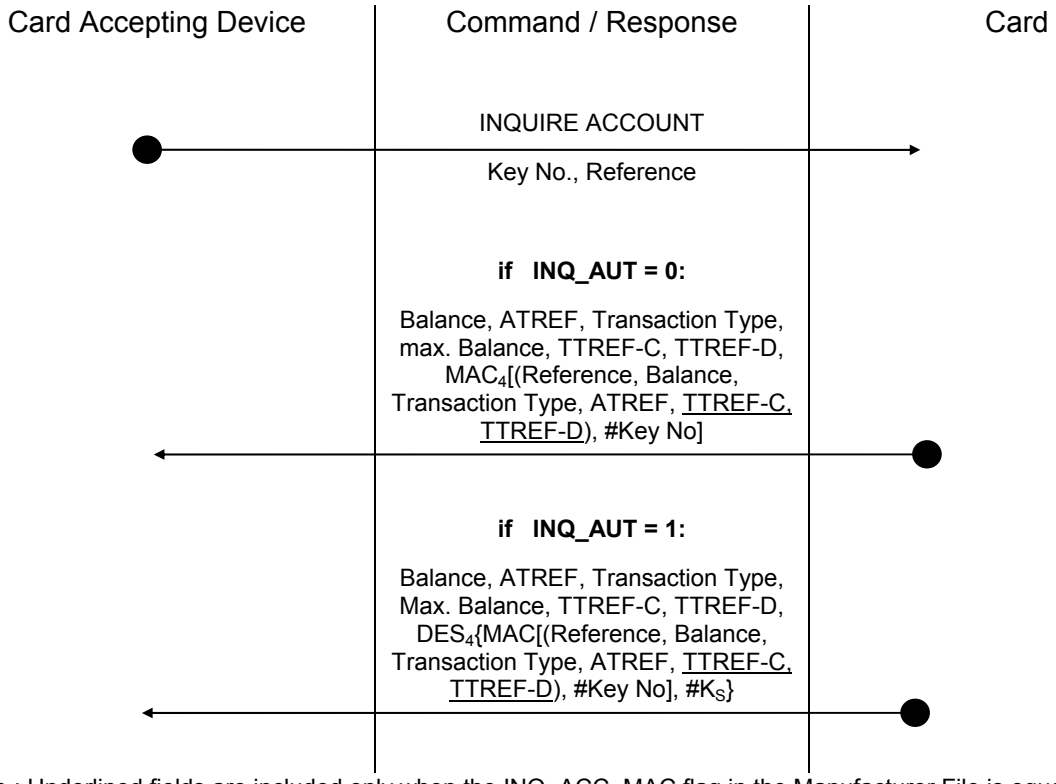
### 4.4.1 INQUIRE ACCOUNT

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:




**Note** : Underlined fields are included only when the INQ_ACC_MAC flag in the Manufacturer File is equal to one.

NOTE: DES shall be 1-DES or 3-DES depending on selected *Option*

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 the Account:

1 = DEBIT
2 = REVOKE DEBIT
3 = CREDIT

Max. Balance — The maximum allowed balance value in the card

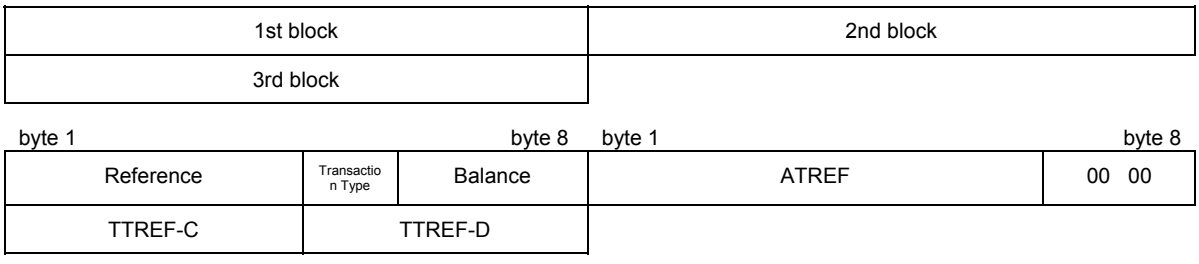
TTREF-C — Terminal Transaction Reference of the last CREDIT transaction

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.*

$DES_4$ — 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.




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

### 4.4.2 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.

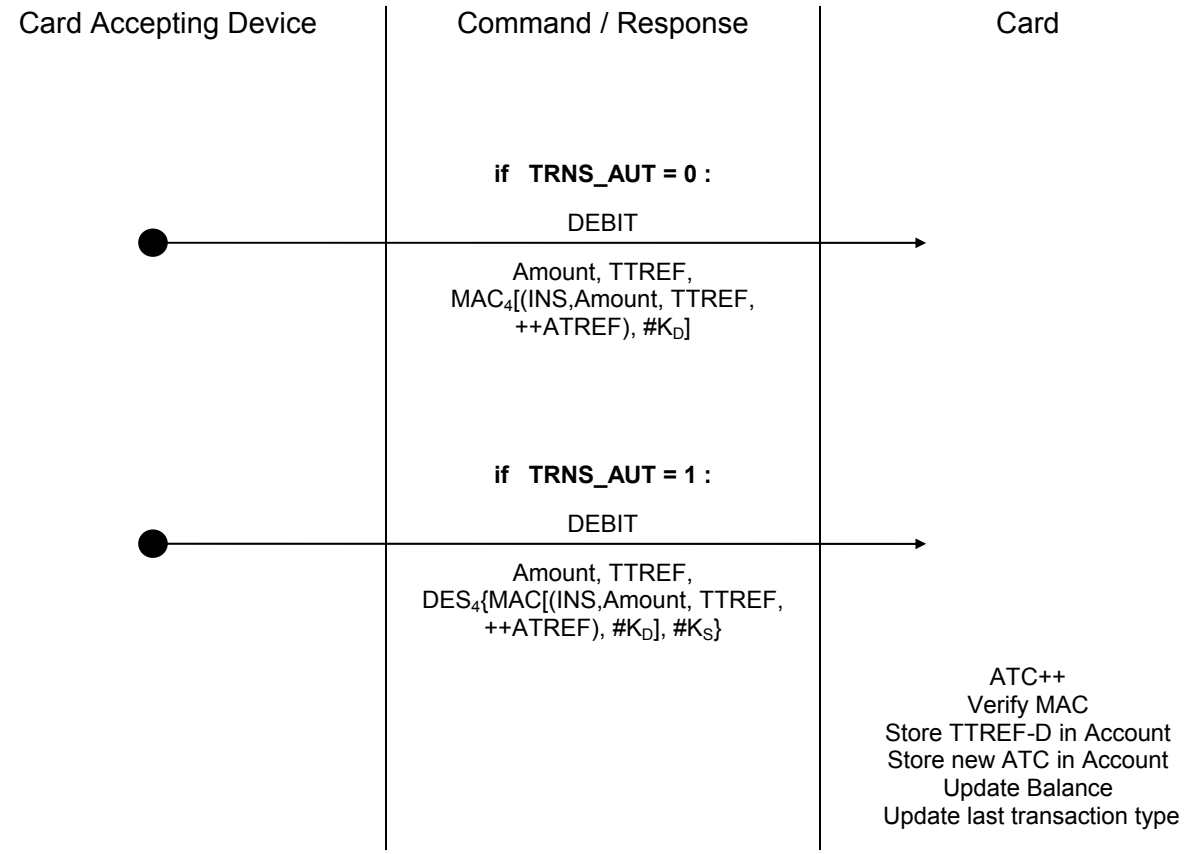
Different security conditions can be specified for the DEBIT transaction to allow for different security requirements. The security conditions for the DEBIT transaction are specified in the DEB_MAC and DEB_PIN option bits in the options register.

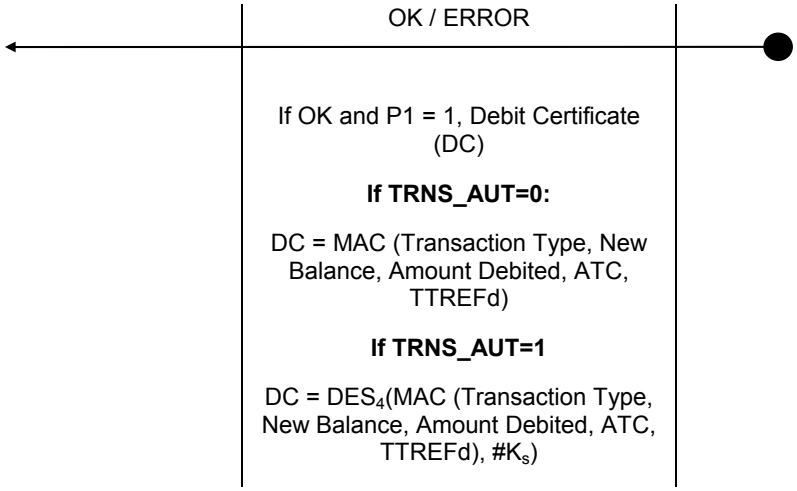
Proper setting of these option bits can specify four different security conditions:

| 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 |

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:






NOTE: DES shall be 1-DES or 3-DES depending on selected *Option*

| | |
|---|---|
| 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. |
| $DES_4$ | 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 incremented before the transaction is being executed!**

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

| 1st block | | | 2nd block | |
|---|---|---|---|---|
| byte 1 | | byte 8 | byte 1 | byte 8 |
| $E6_{HEX}$ | Amount | TTREF-D | ATREF | 00 00 |

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.

**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:

| 1st block | | | 2nd block | | |
|---|---|---|---|---|---|
| byte 1 | | | byte 8 byte 1 | | byte 8 |
| 01 | New Balance | Amount Debited | ATC | TTREFD | 00 00 00 |

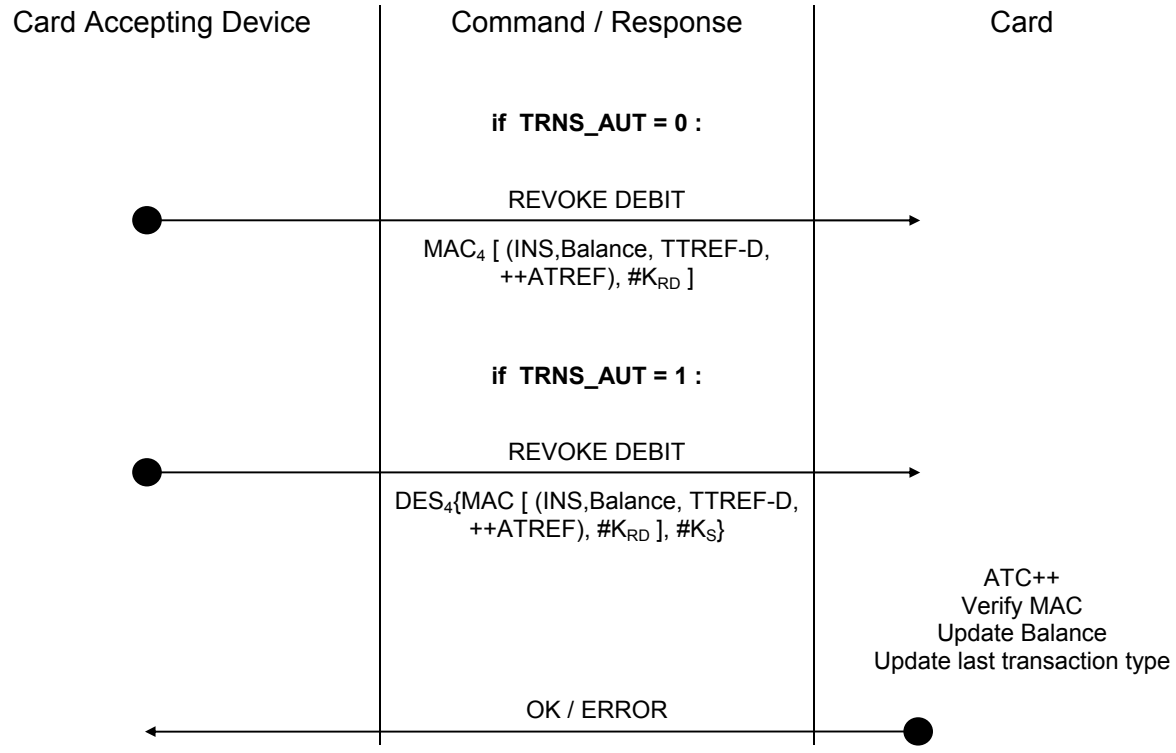
### 4.4.3 REVOKE DEBIT

A 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:




NOTE: DES shall be 1-DES or 3-DES depending on selected *Option*

| | |
|---|---|
| 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.

$DES_4$ 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 incremented before the transaction is being executed!**

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

| 1st block | | | 2nd block | |
|---|---|---|---|---|
| byte 1 | | byte 8 | byte 1 | byte 8 |
| $E8_{HEX}$ | Balance | TTREF-D | ATREF | 00 00 |

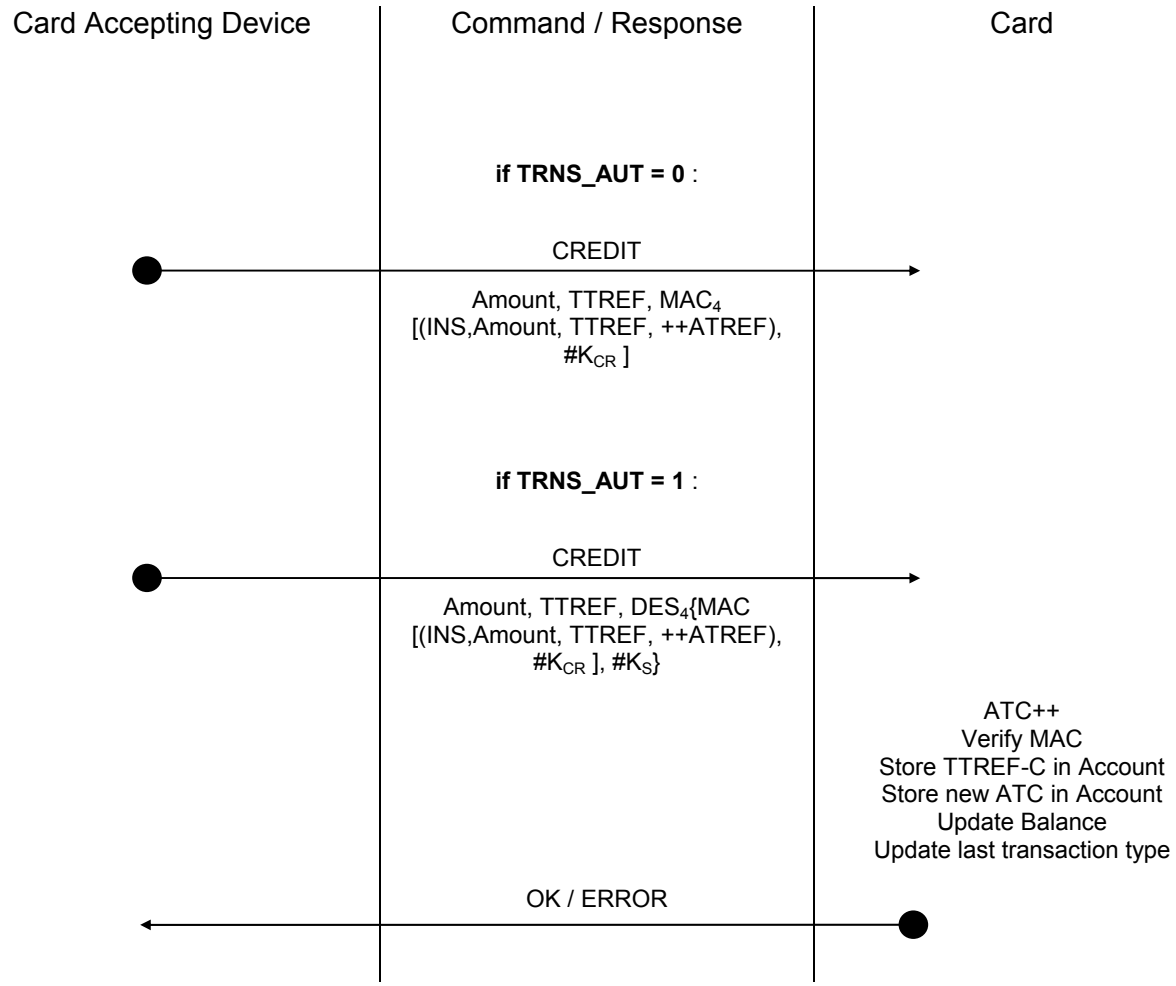
### 4.4.4 CREDIT

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:




NOTE: DES shall be 1-DES or 3-DES depending on selected *Option*

| | |
|---|---|
| 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. |
| $DES_4$ | 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:

| 1st block | | | 2nd block | |
|---|---|---|---|---|
| byte 1 | | byte 8 | byte 1 | byte 8 |
| $E2_{HEX}$ | Amount | TTREF-C | ATREF | 00 00 |

# 5 ISO COMPLIANCE AND ANSWER-TO-RESET

After a hardware reset (e.g., power up), the card transmits an Answer-To-Reset (ATR) 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 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 |
|---|---|---|---|---|---|
| $3B_H$ | $BE_H$ | $11_H$ | $00_H$ | $00_H$ | |

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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $41_H$ | $01_H$ | $10_H$<br>$20_H$<br>$38_H$ | Option registers | | | Personalization File bytes 4 - 8 | | | | | Life-cycle Stage | $90_H$ | $00_H$ |

Lifecycle Stage — Codes the current card lifecycle stage in a single byte
- 0 : User Stage
- 1 : Manufacturing Stage
- 2 : Personalization Stage

Version Bytes — The contents of the version bytes are:
- T1 = $41_H$ = ACOS
- T2 = $01_H$ = Version 1
- T3 = $10_H$ / $20_H$ / $38_H$ = Revision 1.0 / Revision 2.0 / Revision 3.8

Option Registers — The contents of the three bytes option registers:
- T4 = Option Register
- T5 = Security Option Register
- T6 = N_OF_FILE

Personalization File EEPROM memory — 5 bytes following the Option Registers of the Personalization File in the EEPROM memory

**NOTE: For compatibility reasons ACOS3 version bytes will stay the same as ACOS1 revision 3.8.**

## 5.1 Customizing the ATR

There are two ways of customizing ACOS3's ATR. The first way is to add personalization information to the Personalization File FF $02_H$ byte 4 to byte 8. This will be fetched upon power-up in the default ATR's historical bytes stated in the previous section.

In ACOS3, additional capabilities allow for the modification to the card's transmission speed or completely customizing the historical bytes based on the application developer's preference. These ATR modifications are achieved by writing to the internal file FF $07_H$ after submission of the IC code.

As stated in Section 3.3.9, the first byte of the record in FF $07_H$ denotes the customized transmission speed of the card. More specifically, this field states the different clock rate conversion factor and baud rate adjustment factor that directly controls the transmission speed of the card. Note that the transmission rate will depend on the card reader's capability.

The second byte states the length of the modified historical bytes. Notice that after the customizing the historical bytes this way, they will no longer correspond to Options Registers, Personalization File or the Life Cycle Stage.

ACOS3 interprets FF $07_H$ as follows:

| Byte number | Valid Values | Description |
|---|---|---|
| 1 | $96_H$<br>$95_H$<br>$94_H$<br>$93_H$<br>$92_H$<br>$11_H$<br>Any other values | The following are the reference baud rate and their values:<br>- 230,400 bps<br>- 115,200 bps<br>- 57,600 bps<br>- 28,800 bps<br>- 14,400 bps<br>- 9,600 bps (default for backward compatibility reasons)<br>- Use the default speed of 9,600 |
| 2 | $00_H$ – $0F_H$<br>Any other values | - The length of the modified historical bytes.<br>- No modifications are made. The card will use the default historical bytes of the previous section. |
| 3-17 | Any | This field will depend on the length set in byte number 2. This can be used for customized historical bytes dependent on the application developer's preference. |

See ISO7816 Part 3 for more information about ATR formation.

**For example:**

To set the ATR to high speed ($95_H$) and 2 bytes identifier ($1234_H$) in the historical characters, perform the following:

1. Submit IC code (Assuming default IC code): 80 20 07 00 08 41 43 4F 53 54 45 53 $54_H$
2. Select FF $07_H$: 80 A4 00 00 02 FF $07_H$
3. Write 4 bytes to the ATR file including the 1 byte for the speed, 1 byte for the length of the historical characters, and 2 bytes identifier:
   80 D2 00 00 04 95 02 12 $34_H$
4. Reset the card. The ATR is now 3B B2 95 00 00 12 $34_H$.

# 6 COMMANDS

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

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

A summary of the status codes returned by the card is given in *8. STATUS CODES*.

The following commands are provided by ACOS3:

| | |
|---|---|
| START SESSION | Start the Mutual Authentication Process |
| AUTHENTICATE | Authenticate the Card Accepting Device, authenticate the card and compute the Session Key |
| SUBMIT CODE | Submit a secret code |
| CLEAR CARD | Clears the cards back to manufacturer stage |
| SELECT FILE | Select a data file for reading and writing |
| READ RECORD | Read data from a record of the currently open data file |
| WRITE RECORD | Write data to a record of the currently open data file |
| INQUIRE ACCOUNT | Read the balance and other Account information |
| CREDIT | Credit the Account |
| DEBIT | Debit the Account |
| REVOKE DEBIT | Revoke the preceding Debit transaction |
| CHANGE PIN | Change the PIN secret code |
| GET RESPONSE | Get response data available in the card |

## *6.1 START SESSION*

To read a random number from the card and start the mutual authentication process the result of which is the Session Key $K_S$.

**Command:**

| CLA | INS | P1 | P2 | P3 |
|---|---|---|---|---|
| 80 | 84 | 00 | 00 | 08 |

**Response:**

| Data | SW1 SW2 |
|---|---|
| $RND_C$ | Status |

$RND_C$ Eight bytes card random number

**Status Codes:**

| SW1 | SW2 | Meaning |
|---|---|---|
| 69 | 83 | Terminal Authentication Key $K_T$ is locked, authentication process cannot be executed |

## 6.2 AUTHENTICATE

To submit the encrypted random number to the card and initiate the computation of the session key.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA | |
|---|---|---|---|---|---|---|
| 80 | 82 | 00 | 00 | 10 | $DES(RND_C,\#K_T)$ | $RND_T$ |

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

$RND_T$ Eight bytes terminal random number

NOTE: DES shall be 1-DES or 3-DES depending on the selection in *Option* Register

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

| SW1 | SW2 | Meaning |
|---|---|---|
| 69 | 85 | START SESSION not executed immediately before AUTHENTICATE command |
| 63 | Cn | Key $K_T$ not correct; n = remaining number of re-tries |
| 61 | 08 | 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 |
|---|---|---|---|---|
| 80 | C0 | 00 | 00 | 08 |

**Response:**

| Data | SW1 SW2 |
|---|---|
| $DES(RND_T,\#K_S)$ | Status |

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

NOTE: DES shall be 1-DES or 3-DES depending on the selection in *Option* Register

### Specific Status Codes:

| SW1 | SW2 | Meaning |
|---|---|---|
| 69 | 85 | AUTHENTICATE not executed prior to the GET RESPONSE command |

## *6.3 SUBMIT CODE*

To submit a secret code - Application Code, PIN or Issuer Code - to the card.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | 20 | Code No. | 00 | 08 | Code or DES(Code,#$K_S$) |

Code No. Code reference: 1...5 = AC1...AC5
6 = PIN
7 = IC

Code Eight bytes Code

DES(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.**

NOTE: DES shall be 1-DES or 3-DES depending on the selection in *Option* Register

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

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

## *6.4 CLEAR CARD*

To clear the card back to its virgin state. All keys, file structures and data will be erased. This function is only available after successful verification of IC code and when the card is in Manufacturer Stage or Personalization Stage.

**Command:**

| CLA | INS | P1 | P2 | P3 |
|---|---|---|---|---|
| 80 | 30 | 00 | 00 | 00 |

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

| SW1 | SW2 | Meaning |
|---|---|---|
| 69 | 82 | IC code not satisfied or card is in user stage. |

## 6.5 SELECT FILE

To select a data file for subsequent READ RECORD and WRITE RECORD commands.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | A4 | 00 | 00 | 02 | File ID |

File ID Two bytes file identifier

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

| SW1 | SW2 | Meaning |
|---|---|---|
| 6A | 82 | File does not exist |
| 90 | 00 | Internal Data File has been selected |
| 91 | nn | User Data File has been selected.<br>The corresponding File Definition Block is stored in record no. nn in the User File Management File (FF 04 $_H$) |

## 6.6 READ RECORD

To read a number of bytes - up to the record length - from one record in the currently selected file.

**Command:**

| CLA | INS | P1 | P2 | P3 |
|---|---|---|---|---|
| 80 | B2 | Rec No. | 00 | Len |

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

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

**Response:**

| Data | SW1 SW2 |
|---|---|
| Byte 1 ... Byte N | Status |

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

**Specific Status Codes:**

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

## *6.7 WRITE RECORD*

To write a number of bytes - up to the record length - to one record in the currently selected file.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | D2 | Rec No. | 00 | Len | Byte 1 ... Byte N |

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

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

Byte 1 ... Byte N Data bytes to be written to the first *Len* bytes of the record

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

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

## 6.8 CREDIT

To credit the Account.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | E2 | 00 | 00 | 0B | MAC : Amount : TTREF |

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:**

| W1 SW2 |
|---|
| Status |

**Specific Status Codes:**

| SW1 | SW2 | Meaning |
|---|---|---|
| 69 | F0 | Data in account is inconsistent - no access unless in Issuer Mode |
| 6A | 82 | Account does not exist |
| 6F | 10 | Account Transaction Counter at maximum - no more transaction possible |
| 63 | Cn | MAC cryptographic checksum is wrong<br>n = remaining number of retries |
| 6B | 20 | Amount too large |
| 69 | 83 | Credit Key locked |
| 69 | 85 | Mutual Authentication has not been completed |

## *6.9 DEBIT*

To debit the Account.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | E6 | 00<br>01 | 00 | 0B | MAC : Amount : TTREF |

If P1 is 01, 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 bytes Terminal Transaction Reference.

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

| SW1 | SW2 | Meaning |
|---|---|---|
| 69 | F0 | Data in account is inconsistent – no access unless in Issuer Mode |
| 6A | 82 | Account does not exist |
| 6F | 10 | Account Transaction Counter at maximum - no more transaction possible |
| 63 | Cn | MAC cryptographic checksum is wrong<br>n = remaining number of retries |
| 69 | 82 | Security status not satisfied - PIN not submitted |
| 6B | 20 | Amount too large |
| 69 | 82 | PIN not submitted |
| 69 | 83 | Debit Key locked |
| 69 | 85 | Mutual Authentication has not been completed |
| 61 | 04 | Debit successful, issue GET RESPONSE with P3=04 to get Debit Certificate. |

## 6.10 REVOKE DEBIT

To revoke the most recent Debit command.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | E8 | 00 | 00 | 04 | MAC |

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

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

| SW1 | SW2 | Meaning |
|---|---|---|
| 69 | F0 | Data in account is inconsistent - no access unless in Issuer Mode |
| 6A | 82 | Account does not exist |
| 6F | 10 | Account Transaction Counter at maximum - no more transaction possible |
| 63 | Cn | MAC cryptographic checksum is wrong<br>n = remaining number of re-retries |
| 6A | 82 | Account does not exist |
| 69 | F0 | Data in account is inconsistent - no access unless in Issuer Mode |
| 69 | 66 | Command not available (option bit not set) |
| 69 | 83 | Revoke Debit Key locked |

## 6.11 INQUIRE ACCOUNT

To read the relevant information from the Account,

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | E4 | Key No. | 00 | 04 | Reference |

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 | 86 | Key No. invalid |
| 69 | 85 | Mutual Authentication has not been completed |
| 61 | 19 | Issue GET RESPONSE with P3 = 19 |

To get the response, execute the GET RESPONSE command:

**Command:**

| CLA | INS | P1 | P2 | P3 |
|---|---|---|---|---|
| 80 | C0 | 00 | 00 | 19 |

**Response:**

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

MAC4 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 | 85 | No data available; the INQUIRE ACCOUNT command was not executed immediately prior to the GET RESPONSE command |
| 62 | 81 | Account data may be corrupted |

## 6.12 CHANGE PIN

To set a new PIN code in the card.

**Command:**

| CLA | INS | P1 | P2 | P3 | DATA |
|---|---|---|---|---|---|
| 80 | 24 | 00 | 00 | 08 | PIN or $DES^{-1}$(PINnew, #$K_S$) |

PINnew New PIN

Ks Session Key

**NOTE: If the option bit PIN_DES is 0, the PIN code is <u>not</u> DES encrypted with $K_S$!**

NOTE: DES shall be 1-DES or 3-DES depending on the selection in *Option* Register

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

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

## *6.13 GET RESPONSE*

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

**Command:**

| CLA | INS | P1 | P2 | P3 |
|---|---|---|---|---|
| 80 | C0 | 00 | 00 | Len |

Len The expected response data length

**Response:**

| SW1 SW2 |
|---|
| Status |

**Specific Status Codes:**

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

The GET RESPONSE command must be issued **immediately** after the successful execution of any one of the following commands:

| **Command** | **Len** |
|---|---|
| AUTHENTICATE | 8 |
| INQUIRE ACCOUNT | $19_H$ |

# 7 CARD PERSONALIZATION

This section describes the general procedure in the personalization of an ACOS3 smart 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 smart 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 Manufacturer File (File ID = FF $01_H$) and set the related flags in the $1^{st}$ byte of the $1^{st}$ record.

4. Select the Personalization File (File ID = FF $02_H$) 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. Perform a card reset. After the reset, ACOS3 reads the Personalization File and accepts the new value of N_OF_FILE and the option bits stored in the *Option Register,* as well as the new settings in the Manufacturer File.

6. Submit the default Issuer Code IC.

7. Select the User File Management File (File ID = FF $04_H$) and write the File Definition Blocks for the required User Files (WRITE RECORD command) with the security attributes set to 'Free Access'.

8. Select the individual User Files and initialize the data in the files as required (WRITE RECORD command).

9. Select the User File Management File (File ID = FF $04_H$) and write the required security attributes for all User Files (WRITE RECORD command). Verify the contents of the User File Management File (READ RECORD command). **Caution: Do not accidentally change the other parameters in the File Definition Blocks.**

10. If applicable, select the Account File (File ID = FF $05_H$) and initialize the relevant data in the Account File (WRITE RECORD command). Verify the contents of the Account File (READ RECORD command).

11. If applicable, select the Account Security File (File ID = FF $06_H$) and initialize the account processing keys (WRITE RECORD command). Verify the contents of the Account Security File (READ RECORD command).

12. Select the Security File (File ID = FF $03_H$) and initialize all keys and codes (WRITE RECORD command). Verify the contents of the Security File (READ RECORD command)

13. Select the Personalization File (File ID = FF $02_H$) 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.**

14. Perform a card reset. The chip life cycle stage as indicated in the ATR should be 'User Stage'.

15. 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.

# 8 STATUS CODES

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

| SW1 | SW2 | Meaning |
|---|---|---|
| 90 | 00 | Internal Data File has been selected. |
| 91 | nn | User Data File has been selected.<br>The corresponding File Definition Block is stored in record no. nn in the User File Management File (FF 04 $_H$) |
| 61 | nn | O.K. - Issue GET RESPONSE command with $L_e$ = nn to get response data |
| 62 | 81 | Data returned in response to the INQUIRE ACCOUNT command may be incorrect due to corrupted data in the Account Data Structure |
| 63 | Cn | Security related command failed - EXTERNAL AUTHENTICATION failed; incorrect Secret Code submitted; incorrect key used in CREDIT MAC generation;<br>n = number of remaining trials |
| 67 | 00 | Wrong P3 |
| 69 | 66 | Command not available (Manufacturing Stage, option bit not set, etc.) |
| 69 | 82 | Security status not satisfied - Secret Code, Issuer Code or PIN not submitted |
| 69 | 83 | Key or Secret Code is locked - no more verification or submission possible |
| 69 | 85 | 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 |
| 69 | F0 | Account data inconsistent / transaction interrupted - access to account only in privileged mode possible |
| 6A | 82 | File does not exist; account not available |
| 6A | 83 | Record not found - file too short |
| 6A | 86 | P1-P2 incorrect |
| 6B | 20 | Invalid amount in CREDIT/DEBIT command |
| 6C | nn | Issue GET RESPONSE command with P3 = nn to get response data |
| 6D | 00 | Unknown INS |
| 6E | 00 | Invalid CLA |
| 6F | 10 | Account Transaction Counter at maximum - no more DEBIT or CREDIT transaction possible |