

DATA SHEET

mifare DESFire

**Contactless Multi-Application IC
with DES and 3DES Security**

MF3 IC D40

Objective Specification

January 2003

Revision 1.1

CONFIDENTIAL

Functional Specification

mifare DESFire MF3 IC D40

CONTENTS

1	FEATURES	4
1.1	RF Interface: ISO 14443 Type A	4
1.2	Non - Volatile Memory	4
1.3	NV-Memory Organisation	4
1.4	Security	4
2	GENERAL DESCRIPTION	5
2.1	Contactless Energy and Data Transfer	5
2.2	Delivery Types	5
2.3	Anticollision	6
2.4	UID / Serial Number	6
2.5	Memory Organisation	6
2.6	Security	7
3	MF3 IC D40 – CODING OF SECURITY-, APPLICATION- AND FILE- RELATED FEATURES	8
3.1	Coding of File Types	8
3.2	Coding of Communication Settings – Encryption Levels	8
3.3	Coding of Access Rights	9
3.4	Coding of Status- and Error Codes	10
4	DESFire COMMAND SET	11
4.1	Command Set ISO 14443-3:	11
4.1.1	Request Type A (REQA)	11
4.1.2	Wake-Up (WUPA)	11
4.1.3	ANTICOLLISION and SELECT of Cascade Level 1	12
4.1.4	ANTICOLLISION and SELECT of Cascade Level 2	13
4.2	Command Set ISO 14443-4:	14
4.2.1	Request for Answer To Select (RATS)	14
4.2.2	Protocol and Parameter Selection Request (PPS)	15
4.2.3	Frame Waiting Extensions (WTX)	16
4.3	MF3 IC D40 Command Set – Security Related Commands:	17
4.3.1	Authenticate	17
4.3.2	ChangeKeySettings	21
4.3.3	GetKeySettings	24
4.3.4	ChangeKey	25
4.3.5	GetKeyVersion	27
4.3.6	FormatPICC	33

Functional Specification

mifare DESFire MF3 IC D40

4.4	MF3 IC D40 Command Set – PICC Level Commands:	29
4.4.1	CreateApplication	29
4.4.2	DeleteApplication	30
4.4.3	GetApplicationIDs.....	31
4.4.4	SelectApplication	32
4.5	MF3 IC D40 Command Set – Application Level Commands:	36
4.5.1	GetFileIDs.....	36
4.5.2	GetFileSettings	37
4.5.3	ChangeFileSettings	39
4.5.4	CreateStdDataFile	40
4.5.5	CreateBackupDataFile	41
4.5.6	CreateValueFile.....	42
4.5.7	CreateLinearRecordFile	43
4.5.8	CreateCyclicRecordFile.....	44
4.5.9	DeleteFile	45
4.6	MF3 IC D40 Command Set – File Level Commands	46
4.6.1	ReadData	46
4.6.2	WriteData.....	48
4.6.3	GetValue.....	50
4.6.4	Credit	51
4.6.5	Debit	52
4.6.6	LimitedCredit	53
4.6.7	WriteRecord.....	54
4.6.8	ReadRecords	56
4.6.9	ClearRecordFile	58
4.6.10	CommitTransaction	59
4.6.11	AbortTransaction	60
4.6.12	GetVersion.....	33
5	DEFINITIONS.....	61
6	LIFE SUPPORT APPLICATIONS	61
7	REVISION HISTORY	62
	Contact Information	64

Functional Specification

mifare DESFire MF3 IC D40

1 FEATURES

1.1 RF Interface: ISO 14443 Type A

- Contactless transmission of data and powered by the RF-field (no battery needed)
- Operating distance: Up to 100 mm (depending on antenna geometry)
- Operating frequency: 13.56 MHz
- Fast data transfer: 106 kbit/s, 212 kbit/s, 424 kbit/s
- High data integrity: 4 Byte MAC, 16 Bit CRC, parity, bit coding, bit counting
- True deterministic anticollision
- 7 byte unique identifier (cascade level two according to ISO 14443-3)
- Uses ISO 14443-4 transport protocol

1.2 Non - Volatile Memory

- 4 kbyte NV-Memory
- NV-Memory write time 2 ms (1 ms erase, 1 ms program)
- Data retention of 10 years
- Write endurance 100 000 cycles

1.3 NV-Memory Organisation

- Flexible file system
- Up to 28 applications simultaneously on one PICC
- Up to 16 files in each application

1.4 Security

- Unique 7 Byte serial number for each device
- Mutual three pass authentication
- Hardware DES/3DES Data encryption on RF-channel with replay attack protection
- Data Authenticity by 4 Byte MAC
- Authentication on Application level

Functional Specification

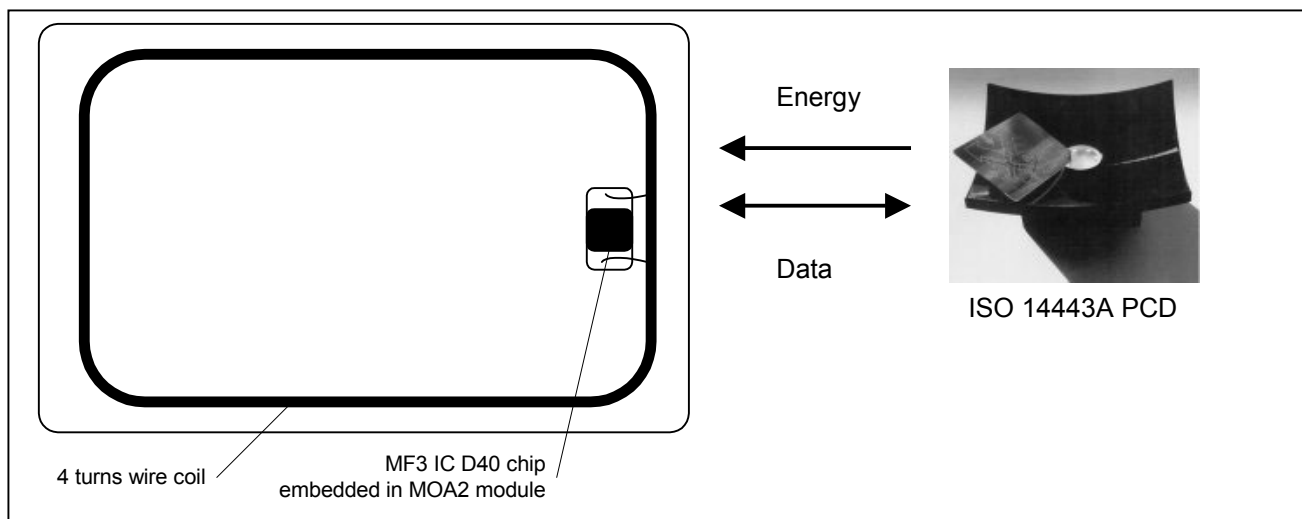
mifare DESFire MF3 IC D40

2 GENERAL DESCRIPTION

Philips has developed the MIFARE® DESFire (MF3 IC D40) to be used with Proximity Coupling Devices (PCDs) according to ISO14443 Type A. The transport protocol complies to part ISO 14443-4. The MF3 IC D40 is primarily designed for secure contactless transport applications and related loyalty programs.

2.1 Contactless Energy and Data Transfer

In the MIFARE® system, the MF3 IC D40 is connected to a coil consisting of a few turns embedded in a standard ISO smart card. No battery is needed. When the card is positioned in the proximity of the PCD antenna, the high speed RF communication interface allows to transmit data with up to 424 kbit/s.



2.2 Delivery Types

- Dies on 8" wafer, sawn on FFC, 150µm thickness
- Au-Bumped Dies on 8" wafer, sawn on FFC, 150µm thickness
- MOA2 Contactless Chip Card Module

Functional Specification

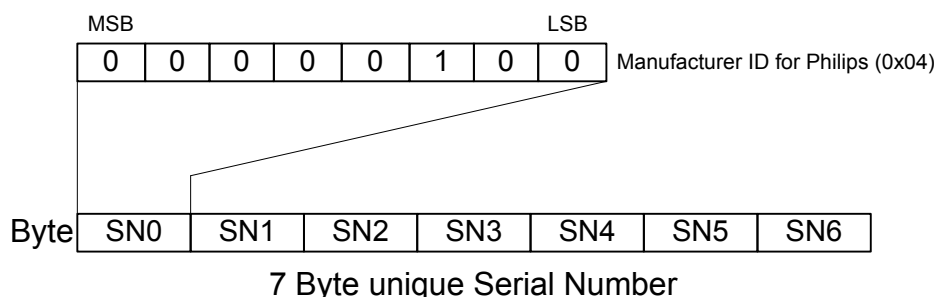
mifare DESFire MF3 IC D40

2.3 Anticollision

An intelligent anticollision mechanism allows to handle more than one PICC in the field simultaneously. The anticollision algorithm selects each PICC individually and ensures that the execution of a transaction with a selected PICC is performed correctly without data corruption resulting from other PICCs in the field.

2.4 UID / Serial Number

The unique 7 byte serial number (UID) is programmed into a locked part of the NV-memory which is reserved for the manufacturer. Due to security and system requirements these bytes are write-protected after having been programmed by the IC manufacturer at production time.



According to ISO14443-3 the first anticollision loop, see chapter 4.1.3, will return the cascade tag 0x88 and the first 3 bytes of the UID, SN0 to SN2 and BCC. The second anticollision loop, see chapter 4.1.4, will return bytes SN3 to SN6 and BCC.

SN0 holds the Manufacturer ID for Philips (04h) according to ISO14443-3 and ISO 7816-6 AM1.

2.5 Memory Organisation

The 4 kbyte NV-memory is organised using a flexible file system. This file system allows a maximum of 28 different applications on one single PICC. Each application provides up to 16 files. Every application is represented by its 3 bytes **A**pplication **I**Dentifier, **AID**.

Five different file types are supported, see chapter 3.1.

Each file can be created either at PICC initialisation (card production / card printing), at PICC personalisation (vending machine) or in the field.

If a file or application becomes obsolete in operation, it can be permanently invalidated.

Commands which have impact on the file structure itself (e.g. creation or deletion of applications, change of keys...) activate an automatic rollback mechanism, which protects the file structure from getting corrupted.

If this rollback is necessary, it is done without user interaction before carrying out further commands.

To ensure data integrity on application level, a transaction oriented backup is implemented for all file types with backup. It is possible to mix file types with and without backup within one application, whereby backup is possible only for files 0 .. 7, files 8 .. 15 do not feature backup mechanisms.

Functional Specification

mifare DESFire MF3 IC D40

2.6 Security

The 7 byte UID is unchangeably programmed into each device during production. It cannot be altered and ensures the uniqueness of each device.

The UID may be used to derive diversified keys for each ticket. Diversified PICC keys contribute to gain an effective anti-cloning mechanism.

Prior to data transmission a mutual three pass authentication can be done between PICC and PCD depending on the configuration employing either DES or 3DES.

Data Transmission between PICC and PCD can be done on three levels of security:

- Plain data transfer
- Plain data transfer with DES/3DES cryptographic checksum (MAC)
- DES/3DES encrypted data transfer (secured by CRC before encryption)

Access to user data is granted on application level. For each application up to 14 different user definable keys can be assigned to control access to data stored in the PICC.

Functional Specification

mifare DESFire MF3 IC D40

3 MF3 IC D40 – CODING OF SECURITY-, APPLICATION- AND FILE- RELATED FEATURES

3.1 Coding of File Types

The files within an application can be of different types as:

- Standard Data Files (coded as 0x00)
- Backup Data Files (coded as 0x01)
- Value Files with Backup (coded as 0x02)
- Linear Record Files with Backup (coded as 0x03)
- Cyclic Record Files with Backup (coded as 0x04)

3.2 Coding of Communication Settings – Encryption Levels

The Communication Settings define the level of security for the communication between PCD and PICC. Communication Settings always apply on file-level.

The Settings are coded into one byte which needs to be set to

Communication Mode	bit 7 – bit 2	bit 1	bit 0
Plain communication	RFU = 0	ignored	0
Plain communication secured by DES/3DES MACing	RFU = 0	0	1
Fully DES/3DES enciphered communication	RFU = 0	1	1

Both DES and 3DES keys are stored in strings consisting of 16 bytes:

If the 2nd half of the key string is equal to the 1st half, the key is handled as a single DES key by the PICC.

If the 2nd half of the key string is **not** equal to the 1st half, the key is handled as a 3DES key.

All operations based on the key (authentication, MACing, encipherment ...) are further on handled with the respective method DES or 3DES.

Examples for single DES keys: 0x00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
 0x00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 *

Examples for 3DES keys: 0x00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF
 0x00 11 22 33 44 55 66 77 00 00 00 00 00 00 01 00
 0x00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00

* All bytes 0x00 is the default key of the MF3 IC D40 IC, and defines single DES operation as default.

DESFire features key versioning, please refer to chapter 4.3.5 for details.

Functional Specification

mifare DESFire MF3 IC D40

3.3 Coding of Access Rights

There are four different Access Rights (2 bytes for each file) stored for each file within each application:

- Read Access (GetValue, Debit for Value files)
- Write Access (GetValue, Debit, LimitedCredit for Value files)
- Read&Write Access (GetValue, Debit, LimitedCredit, Credit for Value files)
- ChangeAccessRights

Each of the Access Rights is coded in 4 bits, one nibble. Each nibble represents a link to one of the keys stored within the respective application's key file.

One nibble allows to code 16 different values. If such a value is set to a number between 0 and 13 (max. 14 keys), this references a certain key within the application's key file, provided that the key exists (configuring a non-existing key is not allowed).

If the number is coded as 14 (0xE) this means "free" access. Thus the regarding access is granted always with and without a preceding authentication, directly after the selection of the respective application.

If only one of the keys for "Read" and "Read&Write" access (or "Write" and "Read&Write" access) is set to 0xE, the other key is different from 0xE, communication is done MACed/enciphered in case of a valid authentication and done in plain in case of no valid authentication. In the second case the communication settings, see chapter 3.2, are ignored by the PICC.

The number 15 (0xF) defines the opposite of "free" access and has the meaning "never" access. Therefore the respective linked Access Rights is always denied.

The most significant 4 bits of the two bytes parameter code the reference number of the key which is necessary to know for getting Read Access (in case of Value files: for the GetValue and the Debit Command).

The next 4 bits hold the number of the key for getting Write Access (in case of Value files: GetValue, Debit and LimitedCredit Command).

The upper nibble of the lower byte holds the key number for getting Read&Write Access, in Value files this right allows full access (GetValue, Debit, LimitedCredit and Credit Command).

The least significant nibble holds the reference number of the key, which is necessary to be authenticated with in order to change the Access Rights for the file and to link each Access Right to key numbers.

Access rights are always packed as a WORD as follows:

15	12	11	8	7	4	3	0		
Read Access				Write Access		Read&Write Access		Change Config	
MS Bit								LS Bit	

Read is possible with Read Access and Read&Write Access.

Write is possible with Write Access and Read&Write Access.

If a file is accessed without valid authentication but free access (0xE) is possible through at least one relevant access right, the communication mode is forced to plain.

Functional Specification

mifare DESFire MF3 IC D40

3.4 Coding of Status- and Error Codes

Hex code	Status	Description
0x00	OPERATION_OK	Successful operation
0x0C	NO_CHANGES	No changes done to backup files, CommitTransaction / AbortTransaction not necessary
0x0E	OUT_OF_EEPROM_ERROR	Insufficient NV-Memory to complete command
0x1C	ILLEGAL_COMMAND_CODE	Command code not supported
0x1E	INTEGRITY_ERROR	CRC or MAC does not match data Padding bytes not valid
0x40	NO_SUCH_KEY	Invalid key number specified
0x7E	LENGTH_ERROR	Length of command string invalid
0x9D	PERMISSION_DENIED	Current configuration / status does not allow the requested command
0x9E	PARAMETER_ERROR	Value of the parameter(s) invalid
0xA0	APPLICATION_NOT_FOUND	Requested AID not present on PICC
0xA1	APPL_INTEGRITY_ERROR	Unrecoverable error within application, application will be disabled *
0xAE	AUTHENTICATION_ERROR	Current authentication status does not allow the requested command
0xAF	ADDITIONAL_FRAME	Additional data frame is expected to be sent
0xBE	BOUNDARY_ERROR	Attempt to read/write data from/to beyond the file's/record's limits. Attempt to exceed the limits of a value file.
0xC1	PICC_INTEGRITY_ERROR	Unrecoverable error within PICC, PICC will be disabled *
0xCD	PICC_DISABLED_ERROR	PICC was disabled by an unrecoverable error *
0xCE	COUNT_ERROR	Number of Applications limited to 28, no additional CreateApplication possible
0xDE	DUPLICATE_ERROR	Creation of file/application failed because file/application with same number already exists
0xEE	EEPROM_ERROR	Could not complete NV-write operation due to loss of power, internal backup/rollback mechanism activated *
0xF0	FILE_NOT_FOUND	Specified file number does not exist
0xF1	FILE_INTEGRITY_ERROR	Unrecoverable error within file, file will be disabled *

* These errors are not expected to appear during normal operation.

Functional Specification

mifare DESFire MF3 IC D40

4 DESFIRE COMMAND SET

4.1 Command Set ISO 14443-3:

The MF3 IC D40 provides the following command set according to ISO 14443-3 activation:

4.1.1 Request Type A (REQA)

Code	Parameter	Data	Integrity mechanism	Response
0x26 (7 Bit)	-	-	-	0x0344

Description: The MF3 IC D40 accepts the REQA command in Idle state only. The response is the 2-byte ATQA (0x0344). REQA and ATQA are implemented fully according to ISO14443-3.

REQA:

Note: Times units are not drawn to scale!

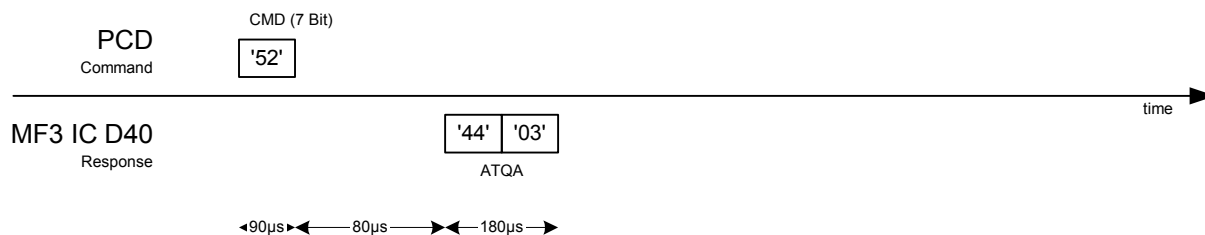


4.1.2 Wake-Up (WUPA)

Code	Parameter	Data	Integrity mechanism	Response
0x52 (7Bit)	-	-	-	0x0344

Description: The MF3 IC D40 accepts the WAKE-UP command in the Idle and Halt state only. The response is the 2-byte ATQA (0x0344). WAKE-UP is implemented fully according to ISO14443-3.

WAKE-UP



Functional Specification

mifare DESFire MF3 IC D40

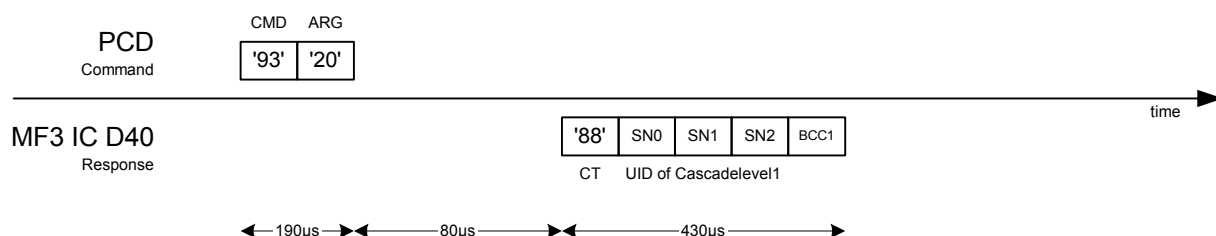
4.1.3 ANTICOLLISION and SELECT of Cascade Level 1

Code	Parameter	Data	Integrity mechanism	Response
Anticollision: 0x93 Select: 0x93	0x20 – 0x67 0x70	Part of the UID Cascade Tag, First 3 bytes of UID	Parity Parity, BCC, CRC	Parts of UID SAK (0x24)

Description: The ANTICOLLISION and SELECT commands are based on the same command code. They differ only in the Parameter byte. This byte is per definition 0x70 in case of SELECT. The MF3 IC D40 accepts these commands in the Ready1 state only. The response is part 1 of the UID.

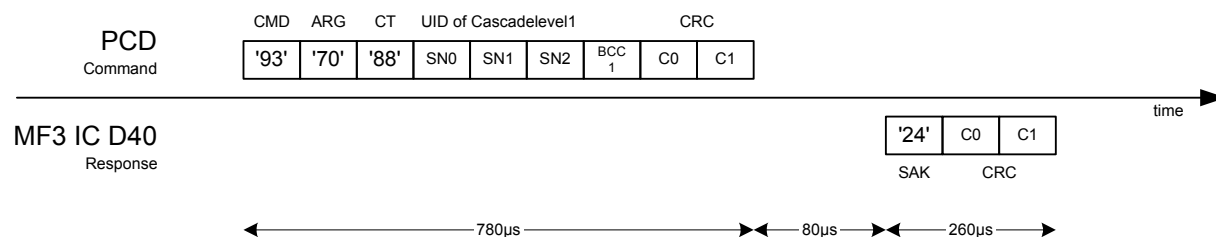
ANTICOLLISION of Cascade Level1

Note: Times units are not drawn to scale!



SELECT of Cascade Level1

Note: Times units are not drawn to scale!



Functional Specification

mifare DESFire MF3 IC D40

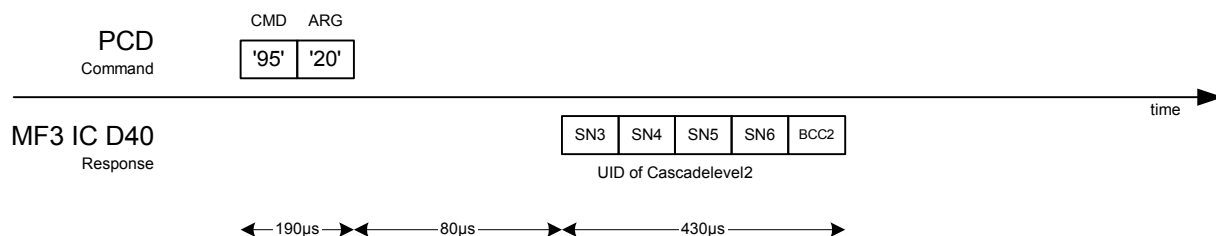
4.1.4 ANTICOLLISION and SELECT of Cascade Level 2

Code	Parameter	Data	Integrity mechanism	Response
Anticollision: 0x95	0x20 – 0x67	Part of the UID	Parity	Parts of UID
Select: 0x95	0x70	Second 4 bytes of UID	Parity, BCC, CRC	SAK (0x20)

Description: The ANTICOLLISION and SELECT command are based on the same command code. They differ only in the parameter byte. This byte is per definition 0x70 in case of SELECT. The MF3 IC D40 accepts these commands in the Ready2 state only. The response is part 2 of the UID.

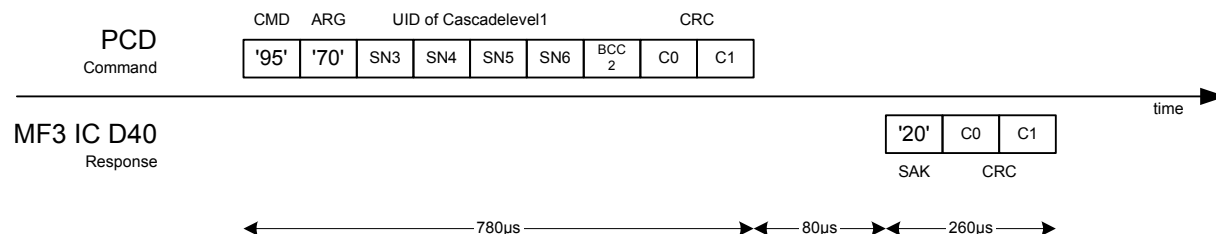
ANTICOLLISION of Cascade Level2

Note: Times units are not drawn to scale!



SELECT of Cascade Level2

Note: Times units are not drawn to scale!



Functional Specification

mifare DESFire MF3 IC D40

4.2 Command Set ISO 14443-4:

The MF3 IC D40 provides the following command set according to ISO 14443-4:

4.2.1 Request for Answer To Select (RATS)

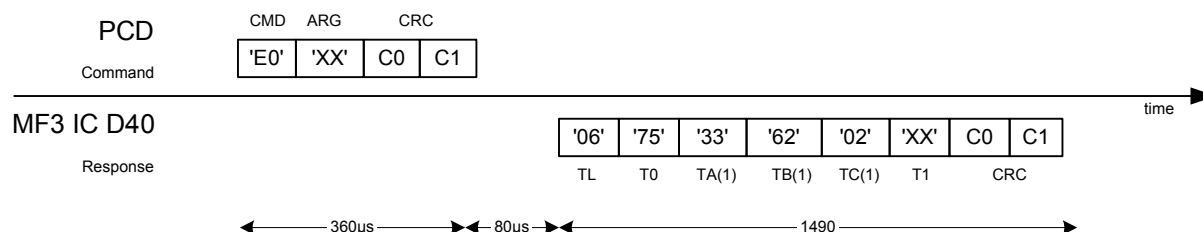
Code	Parameter	Data	Integrity mechanism	Response
0xE0	High Nibble: FSDI Low Nibble: CID	-	CRC	16 Byte Data

Description: The response to the RATS command identifies the PICC type to the PCD. The parameter byte codes two different parameters for communication:

- FSDI: The High Nibble of the parameter byte codes the maximum frame size supported by the PCD for communication with the PICC.
- CID: The Low Nibble of the parameter byte codes the logical number of the addressed PICC. This logical number is in the range from 0x00 to 0x0E. This CID is used to distinguish several PICCs simultaneously selected by a single PCD. The DESFire PICC fully supports CIDs, which is coded in TC(1).

RATS

Note: Times units are not drawn to scale!



The response of the MF3 IC D40 to the RATS is the 'Answer to Select', ATS. This ATS consists of the following bytes:

TL: 'Length Byte', the length byte TL specifies the length of the transmitted ATS including itself. The two CRC bytes are not included in TL. For the MF3 IC D40 device TL is set to 0x06.

T0: 'Format Byte', the format bytes defines the presence of the subsequent bytes TA(1), TB(1) and TC(1). All three are present, resulting is 0x7 for the higher nibble. The lower nibble (FSCI) specifies the maximum size of a frame accepted by the MF3 IC D40 which is 64 bytes, coded as 0x5. The format byte therefore is 0x75.

TA(1): The 'Interface byte TA(1)' codes the maximum possible data rates supported by the PICC. As the MF3 IC D40 supports data rates up to 424 kbaud in both directions (PICC to PCD and PCD to PICC), the TA(1) byte is set to 0x33.

TB(1): The higher nibble codes the Frame Waiting Time (FWT), which is set to 0x6 for DESFire, indicating 19.33 ms. The lower nibble codes the start-up frame guard time (SFGT). It is set to 0x2, indicating 1.21ms. The PCD will therefore receive 0x62.

TC(1): CID is supported, NAD is not supported, therefore TC(1) is set to 0x02.

T1: The DESFire PICC sends one byte as historical character which should be ignored by the application software.

Functional Specification

mifare DESFire MF3 IC D40

4.2.2 Protocol and Parameter Selection Request (PPS)

Code	Parameter	Data	Integrity mechanism	Response
'PPSS: 'DX	PPS0 PPS1	-	CRC	PPSS: 'D0'

Description: The PPS command allows to individually select the communication baud rate between PCD and PICC. For DESFire it is possible to individually set the communication baud rate independently for both directions i.e. DESFire allows a non-symmetrical information interchange speed.

- PPSS: The higher nibble of the PPSS byte is RFU and therefore needs to be set to 'D'. The lower nibble indicates the CID of the selected PICC in the range of 0x00 and 0x0E.
- PPS0: The PPS Parameter 0 byte indicates the presence of PPS1 (which is valid for DESFire) and therefore has to be set to 0x11.
- PPS1: The PPS Parameter 1 byte defines the divisor integer for timings between PCD and PICC which directly defines the baud rate in each direction.

The higher nibble of the PPS1 byte is RFU and has to be set to '0'. Bits b3 and b2 code the divisor integer from PICC to PCD and are called DSI. Bits b1 and b0 code the divisor integer from PCD to PICC and are called DRI.

The coding of DRI and DSI is done as specified below:

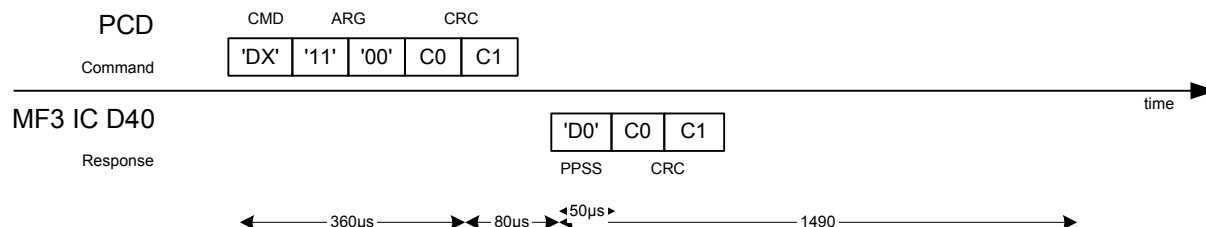
DRI, DSI bit coding	00	01	10
Divisor	1	2	4
Resulting Baud Rate	106 kbaud	212 kbaud	424 kbaud

DESFire supports baud rates up to 424 kbaud in both directions. please note that it is possible to independently set the communication speed in both directions, which allows for example to use 106 kbaud for communication from PCD to PICC (DRI=00) and 424 kbaud from PICC to PCD (DSI=10).

For communication with 106 kbaud in both directions the value of PPS1 has to be '00', which is the default if no PPS command is sent to the PICC.

PPS

Note: Times units are not drawn to scale!



The response of the MF3 IC D40 to the PSS command is the PPS Start byte (PPSS, 0xD0). Invalid PPS requests are ignored (as defined in ISO).

Functional Specification**mifare DESFire MF3 IC D40**

4.2.3 Frame Waiting Extensions (WTX)

If the PICC needs more time than the defined FWT to respond to a PCD command it will send a request for a waiting time extension.

A 14443-4 S-block is sent by the PICC. According to ISO the INF field will contain the value 0x01, requesting another FWT interval.

The PCD has to confirm the request by sending another S-block either confirming 0x01 in the INF field.

Functional Specification

mifare DESFire MF3 IC D40

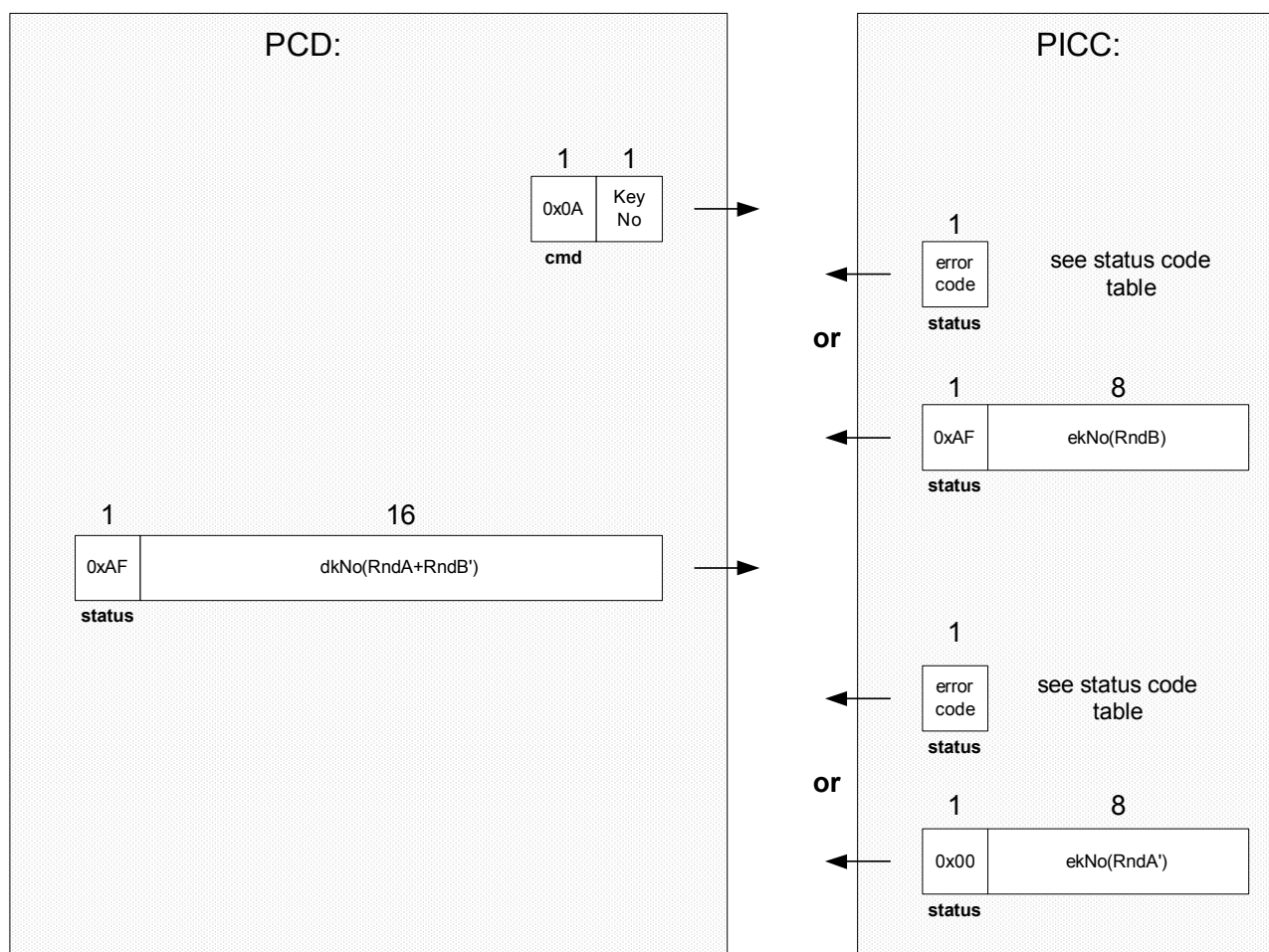
4.3 MF3 IC D40 Command Set – Security Related Commands:

The MF3 IC D40 provides the following command set for security related functions:

4.3.1 Authenticate

In this procedure both, the PICC as well as the reader device, show in an encrypted way that they possess the same secret which especially means the same key. This procedure not only confirms that both entities are permitted to do operations on each other but also creates a session key which can be used to keep the further communication path secure. As the name “session key” implicitly indicates, each time a new authentication procedure is successfully completed a new key for further cryptographic operations is obtained.

Authenticate(keyNo) [2bytes]



Functional Specification

mifare DESFire MF3 IC D40

Below it is described in more detail how the mutual 3-pass authentication procedure is done and the session key is generated:

#	PCD	Data exchanged	PICC
1	<p>The reader device is always the entity which is starting an authentication procedure. This is done by sending the command Authenticate. As parameter the key number is passed to the PICC in order to reference a certain key stored in its NV-memory (up to 14 different keys per application). If the key number does not reflect a valid key in the PICC memory, an error code is sent by the PICC in response.</p> <p>Depending on the previously selected AID on the PICC, the subsequent authentication procedure is done for this specific AID.</p> <p>If the previously selected AID is 0x00, then the authentication is done using the PICC Master Key. In this case, the parameter key number has to be set to 0x00, too. The possibilities and usage of the PICC Master Key is described later on. After power up of the PICC the AID 0x00 is implicitly selected, which means that an Authenticate command after power-up always references the PICC Master Key.</p>	<p style="text-align: center;">→</p> <p style="text-align: center;">Authenticate (KeyNo)</p>	
2		<p style="text-align: center;">←</p> <p style="text-align: center;">8 bytes $ek_{keyNo}(RndB)$</p>	<p>After a specific key is selected, the PICC generates an 8 byte random number <i>RndB</i>. This random number is DES/3DES enciphered with the selected key, denoted by $ek_{keyNo}(RndB)$, and is then transmitted to the PCD.</p>
3	<p>The PCD runs a DES/3DES deciphering operation on the received $ek_{keyNo}(RndB)$ and thus gains <i>RndB</i>. The used key for the deciphering obviously has to be the same as for the previous enciphering by the PICC.</p> <p>In the next step the deciphered <i>RndB</i> is rotated left by 8 bits (first byte is moved to the end of <i>RndB</i>), resulting in <i>RndB'</i>.</p> <p>Now the PCD itself generates an 8 byte random number <i>RndA</i>. This <i>RndA</i> is concatenated with <i>RndB'</i> and deciphered using DES/3DES (The decryption of the</p>	<p style="text-align: center;">→</p> <p style="text-align: center;">16 bytes $dk_{keyNo}(RndA + RndB')$</p>	

Functional Specification

mifare DESFire MF3 IC D40

#	PCD	Data exchanged	PICC
	two blocks is chained using the Cipher Block Chaining (CBC) send mode). This token $dk_{keyNo}(RndA + RndB')$ is sent to the PICC.		
4		<p style="text-align: center;">←</p> <p style="text-align: center;">8 bytes</p> <p style="text-align: center;">$ek_{keyNo}(RndA')$</p>	<p>The PICC runs an DES/3DES encipherment on the received token and thus gains $RndA + RndB'$. The PICC can now verify the sent $RndB'$ by comparing it with the $RndB'$ obtained by rotating the original $RndB$ left by 8 bits internally.</p> <p>A successful verification proves to the PICC that the PICC and the PCD posses the same secret (key).</p> <p>If the verification fails, the PICC stops the authentication procedure and returns an error message.</p> <p>As the PICC also received the random number $RndA$, generated by the PCD, it can perform a rotate left operation by 8 bits on $RndA$ to gain $RndA'$, which is enciphered again, resulting in $ek_{keyNo}(RndA')$. This token is sent to the PCD.</p>
5	<p>The PCD runs a DES/3DES decipherment on the received $ek_{keyNo}(RndA')$ and thus gains $RndA'$ for comparison with the PCD-internally rotated $RndA'$.</p> <p>If the comparison fails, the PCD exits the procedure and may halt the PICC.</p>		
6			The PICC sets the authentication state for the currently selected application or the PICC itself (in case of AID=0x00).
7	<p>If all comparisons are successful, the 16 byte session key is obtained by employing $RndA$ and $RndB$. The session key is gained by combining them according to the following rule:</p> <p style="text-align: center;">session key := $RndA_{1st\ half} + RndB_{1st\ half} + RndA_{2nd\ half} + RndB_{2nd\ half}$</p> <p>This scrambling of $RndA$ and $RndB$ is done to avoid that a malicious PCD could degenerate 3DES cryptography to single DES operation by forcing $RndA = RndB$.</p> <p>In case of WANTED single DES operation (leading 8 bytes of the secret key are identical to the trailing 8 bytes), only the first 8 bytes of the session key ($RndA_{1st\ half} + RndB_{1st\ half}$) are used for further cryptographic operations, the trailing 8 bytes must not be used.</p> <p>With the generation of the session key the mutual 3-pass authentication is successfully completed.</p>		

Functional Specification

mifare DESFire MF3 IC D40

Depending on the configuration of the application (represented by its AID), an authentication has to be done to perform specific operations:

- Gather information about the application
- Change the keys of the application
- Create and delete files within the application
- Change access rights
- Access data files in the authenticated application

Depending on the security configuration of the PICC, the following commands may require an authentication with the PICC master keys:

- Gather information about the applications on the PICC
- Change the PICC master key itself
- Change the PICC key settings
- Create a new application
- Delete an existing application

The authentication state is invalidated by

- Selecting an application
- Changing the key which was used for reaching the currently valid authentication status
- A failed authentication

Please note: Master keys are identified by their key number 0x00. This is valid on PICC level (selected AID = 0x00) and on Application level (selected AID ≠ 0x00).

Functional Specification

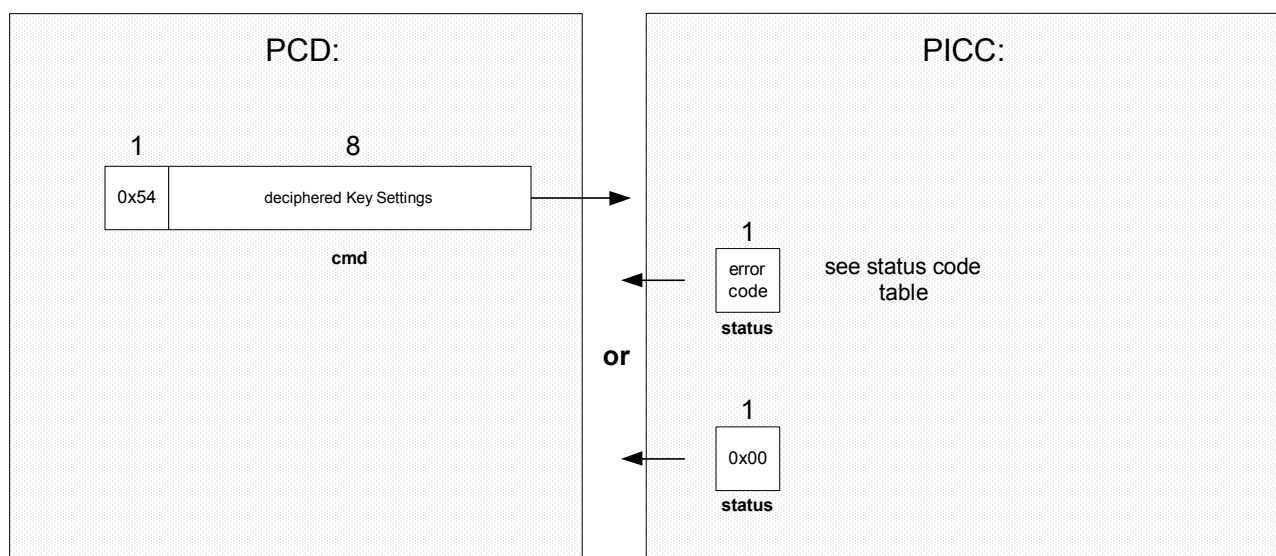
mifare DESFire MF3 IC D40

4.3.2 ChangeKeySettings

This command changes the master key settings dependent on the currently selected AID.

If AID = 0x00 has been selected in advance, the change applies to the PICC key settings, otherwise (AID ≠ 0x00) it applies to the application key settings of the currently selected application.

ChangeKeySettings(KeySettings) [2 bytes]



This command only succeeds if the “configuration changeable” bit, see below, of the current key settings was not cleared before.

Additionally a successful preceding authentication with the master key is required (PICC master key if AID = 0x00, else with application master key).

This command takes one byte as parameter which codes the new master key settings.

To guarantee that the ChangeKeySettings command is sent by the same PCD which did the preceding Authentication command, it is necessary to apply the same security mechanism as for the ChangeKey command, see chapter 4.3.4.

A CRC is calculated on the parameter byte and appended at its end. As this modified data stream now is of three bytes length, five zero bytes 0x00 are appended to get an 8 bytes long data stream suitable for the DES/3DES decipherment operation.

The PICC now is capable of proving the authenticity of the received data by running a DES/3DES encipherment and checking the recovered CRC and padding bytes of the plain data.

Functional Specification

mifare DESFire MF3 IC D40

- PICC Master Key Settings:

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
RFU	RFU	RFU	RFU	configuration changeable	free create / delete without master key	free directory list access without master key	allow change master key

On PICC Level (selected AID = 0x00) the coding is interpreted as:

Bit7-Bit 4: RFU, has to be set to 0.

Bit3: codes whether a change of the PICC master key settings is allowed:

- 0: configuration not changeable anymore (frozen).
- 1: this configuration is changeable if authenticated with PICC master key (default setting).

Bit2: codes whether master key authentication is needed before CreateApplication / DeleteApplication

- 0: CreateApplication / DeleteApplication is permitted only with PICC master key authentication.
- 1: CreateApplication is permitted without PICC master key authentication (default setting).

DeleteApplication requires an authentication with application master key or PICC master key.

Bit1: codes whether PICC master key authentication is needed for application directory access:

- 0: Successful PICC master key authentication is required for executing the GetApplicationIDs and GetKeySettings commands.
- 1: GetApplicationIDs and GetKeySettings commands succeed independently of a preceding PICC master key authentication (default setting).

Bit0: codes whether the PICC master key is changeable:

- 0: PICC Master key is not changeable (frozen).
- 1: PICC Master key is changeable (authentication with the current PICC master key necessary, default setting).

Functional Specification

mifare DESFire MF3 IC D40

- Application Master Key Settings:

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ChangeKey Access Rights Bit3	ChangeKey Access Rights Bit2	ChangeKey Access Rights Bit1	ChangeKey Access Rights Bit0	configuration changeable	free create / delete without master key	free directory list access without master key	allow change master key

On Application Level (selected AID ≠ 0x00) the coding is interpreted as:

Bit7-Bit4: hold the Access Rights for changing application keys (ChangeKey command).

- 0x0: Application master key authentication is necessary to change any key (default).
- 0x1 .. 0xD: Authentication with the specified key is necessary to change any key.
- 0xE: Authentication with the key to be changed (same KeyNo) is necessary to change a key.
- 0xF: All Keys (except application master key, see Bit0) within this application are frozen.

Bit3: codes whether a change of the application master key settings is allowed:

- 0: configuration not changeable anymore (frozen).
- 1: this configuration is changeable if authenticated with the application master key (default setting).

Bit2: codes whether application master key authentication is needed before CreateFile / DeleteFile

- 0: CreateFile / DeleteFile is permitted only with application master key authentication.
- 1: CreateFile / DeleteFile is permitted also without application master key authentication (default setting).

Bit1: codes whether application master key authentication is needed for file directory access:

- 0: Successful application master key authentication is required for executing the GetFileIDs and GetKeySettings commands.
- 1: GetFileIDs and GetKeySettings commands succeed independently of a preceding application master key authentication (default setting).

Bit0: codes whether the application master key is changeable:

- 0: Application master key is not changeable (frozen).
- 1: Application master key is changeable (authentication with the current application master key necessary, default setting).

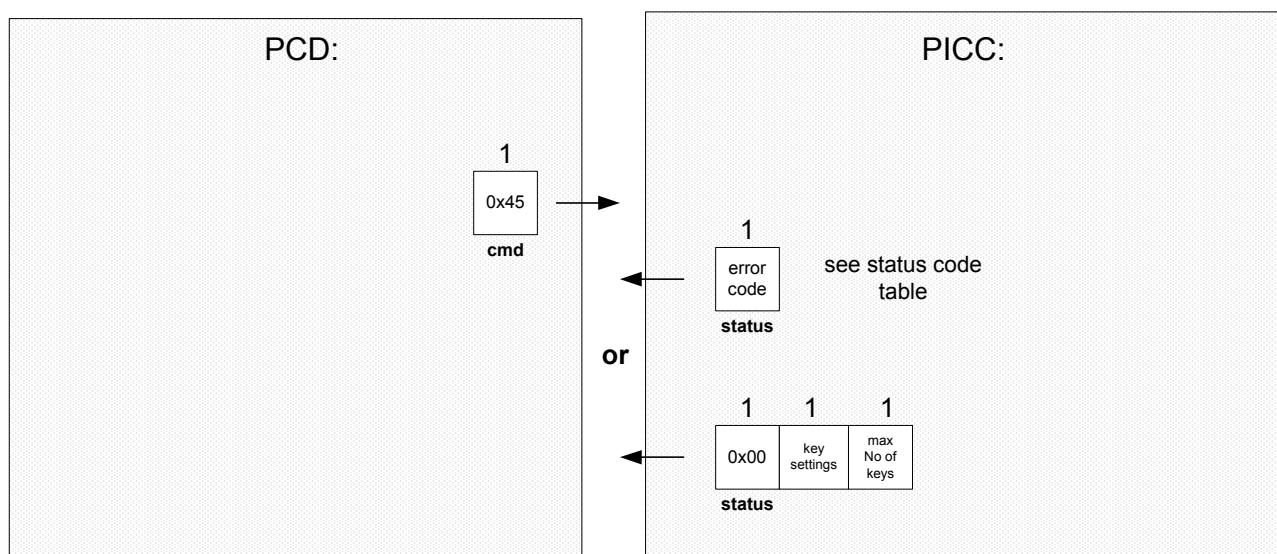
Functional Specification

mifare DESFire MF3 IC D40

4.3.3 GetKeySettings

The GetKeySettings command allows to get information on the PICC and application master key settings as described in chapter 4.3.2. In addition it returns the maximum number of keys which can be stored within the selected application.

GetKeySettings() [1 byte]



No parameter is passed with this command.

Depending on the master key settings (see chapter 4.3.2), a preceding authentication with the master key is required.

If the PICC master key settings are queried (currently selected AID = 0x00), the number of keys is returned as 0x01, as only one PICC master key exists on a PICC.

Functional Specification

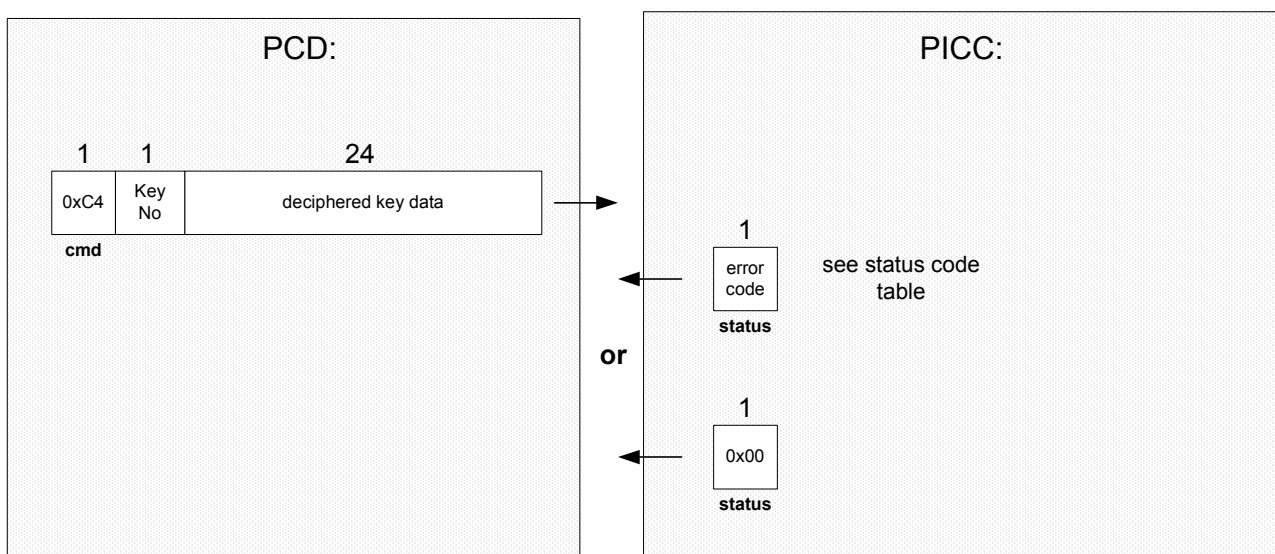
mifare DESFire MF3 IC D40

4.3.4 ChangeKey

This command allows to change any key stored on the PICC.

If AID = 0x00 is selected, the change applies to the PICC master key and therefore only KeyNo = 0x00 is valid (only one PICC master key is present on a PICC). In all other cases (AID ≠ 0x00) the change applies to the specified KeyNo within the currently selected application (represented by it's AID).

ChangeKey(KeyNo) [26 bytes]



As a parameter this command takes the KeyNo which is of one byte length and has to be in the range from 0x00 to number of application keys - 1.

The second parameter holds the information about the new key packed in a cryptogram.

The respective attribute settings (see chapter 4.3.2) define whether a change of keys is permitted or not, additionally they show which key is needed for Authentication before the ChangeKey Command.

- If the ChangeKey Key is set to 0x0 .. 0xD, authentication with the KeyNo defined in the ChangeKey Key settings is necessary to change any key (except Master Key and the ChangeKey Key itself) within the application.

In this case the PCD generates the data frame “deciphered key data” in the following way:

The new key and the current key are bit-wise XORed (16 byte). A CRC (2 byte) is calculated over the XORed data and appended at the end. Additionally a CRC (2 byte) of the new key is appended. After this padding of zeros (4 byte) is applied to reach an adequate frame size of multiples of 8 (24 byte overall). Finally a DES/3DES deciphering operation (using the current session key derived from ChangeKey key) is performed on the whole key data field. The three cryptogram blocks are chained using the CBC send mode.

The same method must be used to change the ChangeKey Key itself: In this case a preceding authentication with the application master key is necessary.

Functional Specification

mifare DESFire MF3 IC D40

- If the ChangeKey Key is set to 0xE, authentication with the KeyNo which will be changed is necessary. This mode below is also employed for changing a Master Key.

In this case the PCD generates the data frame “deciphered key data” in the following way:

A CRC (2 bytes) is calculated over the new key data (16 bytes) and appended at the end. After this padding of zeros (6 bytes) is applied to reach an adequate frame size of multiples of 8 (24 byte overall). Finally a DES/3DES deciphering operation (using the current session key) is performed on the whole key data field. The three cryptogram blocks are chained using the CBC send mode.

- If the ChangeKey Key is set to 0xF (“never”), all keys except the Master Key (see chapter 4.3.2, Bit0) are frozen. The ChangeKey command therefore will return an error code when attempting to change a key different from the master key.

In order to support key versioning, DESFire allows to store a one byte (8 bit) key version within the key. It is an inherent property of DES and 3DES to use only 7 bits of every key byte as significant key information. The 8th bit (which is the LSBit of each key byte) holds parity information only. As the DESFire crypto hardware does not use this parity information, it can be used to store a key version. Every parity bit of the first 8 byte of the DES / 3DES key is used for this purpose.

To store a key version to the PICC, the new version number needs to be coded by the application software into the key information before issuing the ChangeKey command. The key version information is NOT checked by the PICC in any respect.

To read out the version of the current key from a PICC the GetKeyVersion command, see chapter 4.3.5, is used.

Note: After a successful change of the key used to reach the current authentication status, this authentication is invalidated i.e. an authentication with the new key is necessary for subsequent operations.

Functional Specification

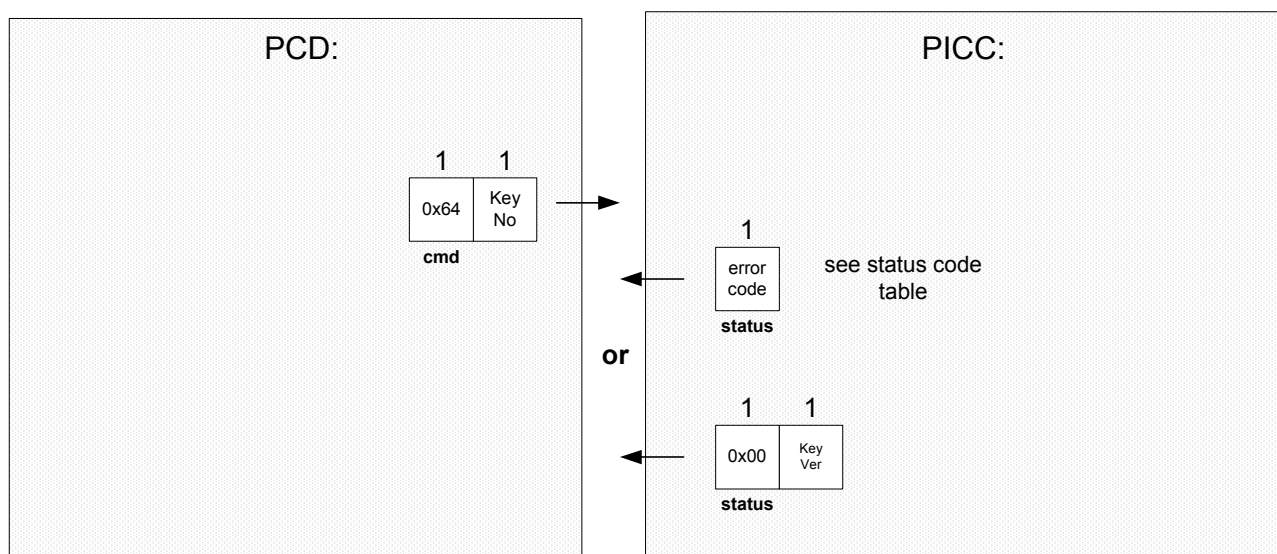
mifare DESFire MF3 IC D40

4.3.5 GetKeyVersion

The GetKeyVersion command allows to read out the current key version of any key stored on the PICC.

If AID = 0x00 is selected, the command returns the version of the PICC master key and therefore only KeyNo = 0x00 is valid (only one PICC master key is present on a PICC). In all other cases (AID ≠ 0x00) the version of the specified KeyNo within the currently selected application (represented by it's AID) is returned.

GetKeyVersion(KeyNo) [2 bytes]



One parameter is passed with this command which codes the key number.

The command returns the current version of the specified key as an unsigned byte.

To change the key version of any key, the ChangeKey command, see chapter 4.3.4, is used.

This command can be issued without valid authentication.

Functional Specification

mifare DESFire MF3 IC D40

Functional Specification

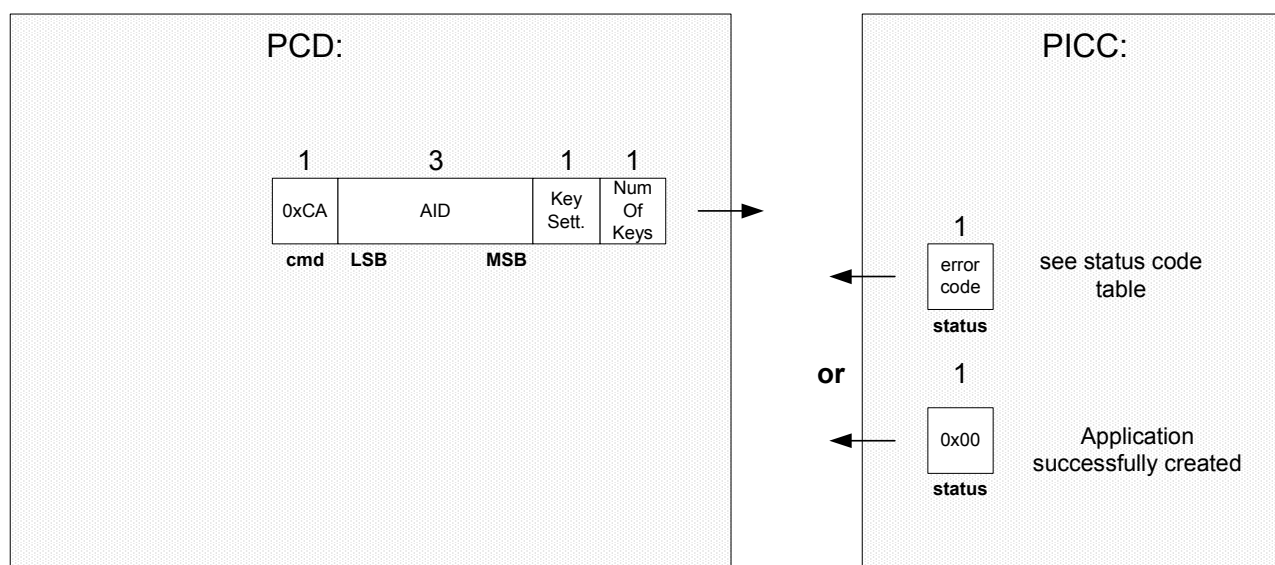
mifare DESFire MF3 IC D40

4.4 MF3 IC D40 Command Set – PICC Level Commands:

4.4.1 CreateApplication

The CreateApplication command allows to create new applications on the PICC.

CreateApplication(AID,KeySettings,NumOfKeys) [6 bytes]



An application is identified by an 'Application Identifier', AID, which is implemented as a 24 bit number. Application Identifier 0x00 00 00 is reserved as a reference to the PICC itself.

Depending on the PICC master key settings, see chapter 4.3.2, a preceding PICC master key authentication may be required.

This command requires that the currently selected AID is 0x00 00 00 which references the card level.

One PICC can hold up to 28 Applications. Each application is linked to a set of up to 14 different user definable access keys. To store data in an application, it is necessary to create so called files within that application, see chapters 4.5.4 to 4.5.8. Up to 16 files of different size and type can be created within each application. Different levels of access rights for each single file can be linked to the keys of the application.

The 24 bit AID is the first parameter of the command.

The second parameter is the Application Master Key Settings as defined in chapter 4.3.2.

The last parameter 'Number of Keys' defines how many keys can be stored within the application for cryptographic purposes.

All keys are initialised with a string consisting of sixteen 0x00 bytes and therefore are single DES keys by definition, see chapter 3.2.

Note: These keys **must** be personalised latest at card issuing using the command ChangeKey.

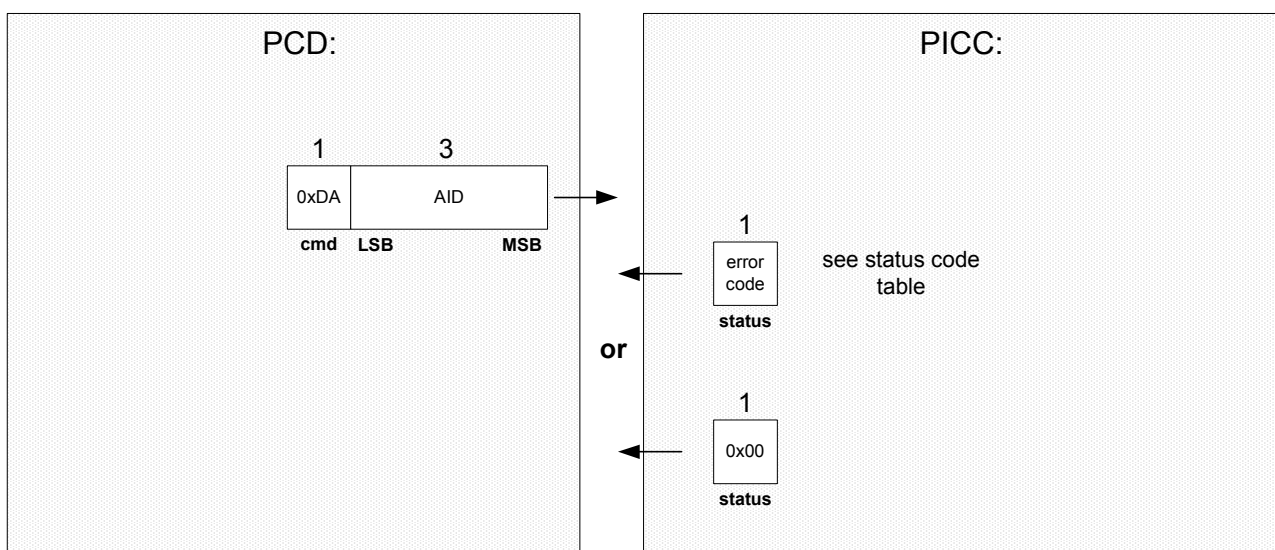
Functional Specification

mifare DESFire MF3 IC D40

4.4.2 DeleteApplication

The DeleteApplication command allows to permanently deactivate applications on the PICC.

DeleteApplication(AID) [4 bytes]



The application which will be deleted is represented by its Application Identifier, AID, which is the only parameter of this command.

Depending on the PICC master key settings, see chapter 4.3.2, either a preceding PICC master key authentication or an application master key authentication is required.

In the latter case, it has to be the master key authentication for the application which shall be deleted by this command.

The AID allocation is removed, therefore it is possible to create a new application with the deleted application's AID. However, the deleted memory blocks can only be recovered by using the FormatPICC command (see chapter 4.4.5) which erases the full user memory of the PICC.

Note: Even if the PICC master key contains the default value 0 and the bit "free create / delete without master key" is set, it is necessary to be either authenticated with the zero PICC master key or the respective application master key.

Note: If the currently selected application is deleted, this command automatically selects the PICC level, selected AID = 0x00 00 00.

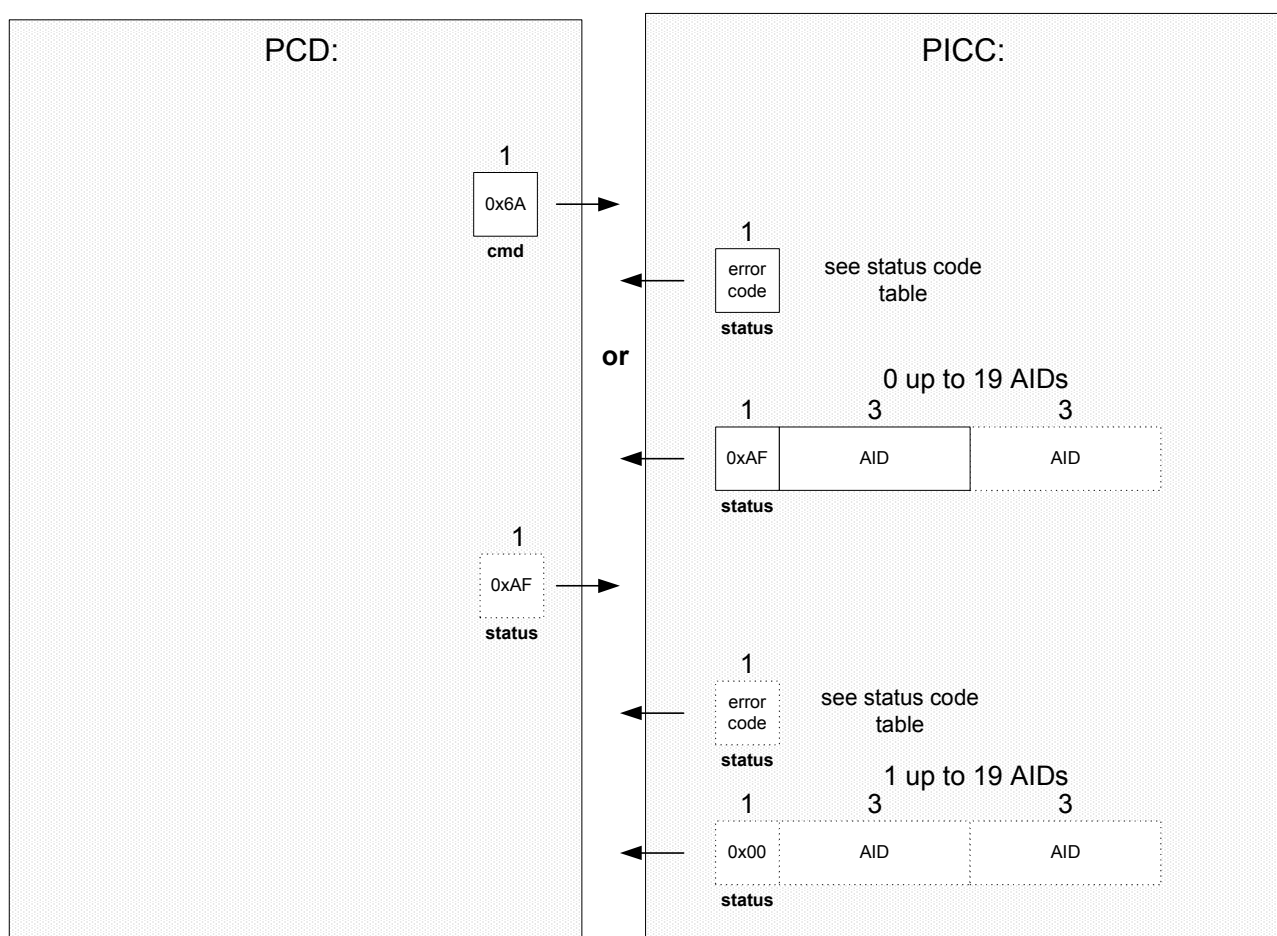
Functional Specification

mifare DESFire MF3 IC D40

4.4.3 GetApplicationIDs

The GetApplicationIDs command returns the Application IDentifiers of all active applications on a PICC.

GetApplicationIDs() [1 byte]



This command does not accept any parameters.

Depending on the PICC master key settings (see chapter 4.3.2) a successful authentication with the PICC master key might be required to execute this command.

This command requires that the currently selected AID is 0x00 00 00 which references the card level.

As response the PICC sends a sequence of all installed AIDs. If this sequence does not fit into one single frame, an additional frame is set by the PICC, indicated by a 0xAF in the status byte of all frames which will be continued.

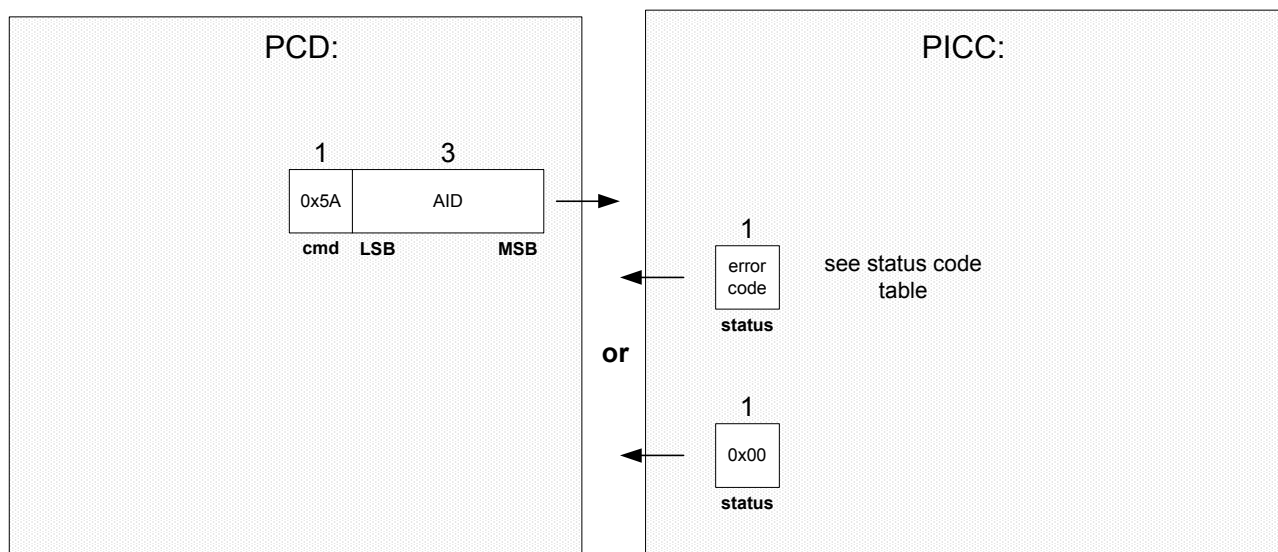
Functional Specification

mifare DESFire MF3 IC D40

4.4.4 SelectApplication

The SelectApplication command allows to select one specific application for further access. In normal operation typically this would be a subsequent Authenticate command.

SelectApplication(AID) [4 bytes]



As parameter this command takes three bytes coding the AID.

If this parameter is 0x00 00 00, the PICC level is selected and any further operations (typically commands like CreateApplication, DeleteApplication...) are related to this level.

If an application with the specified AID is found in the application directory of the PICC, the subsequent commands interact with this application.

As mentioned in the description of the Authenticate command (see chapter 4.3.1), each SelectApplication command invalidates the current authentication status.

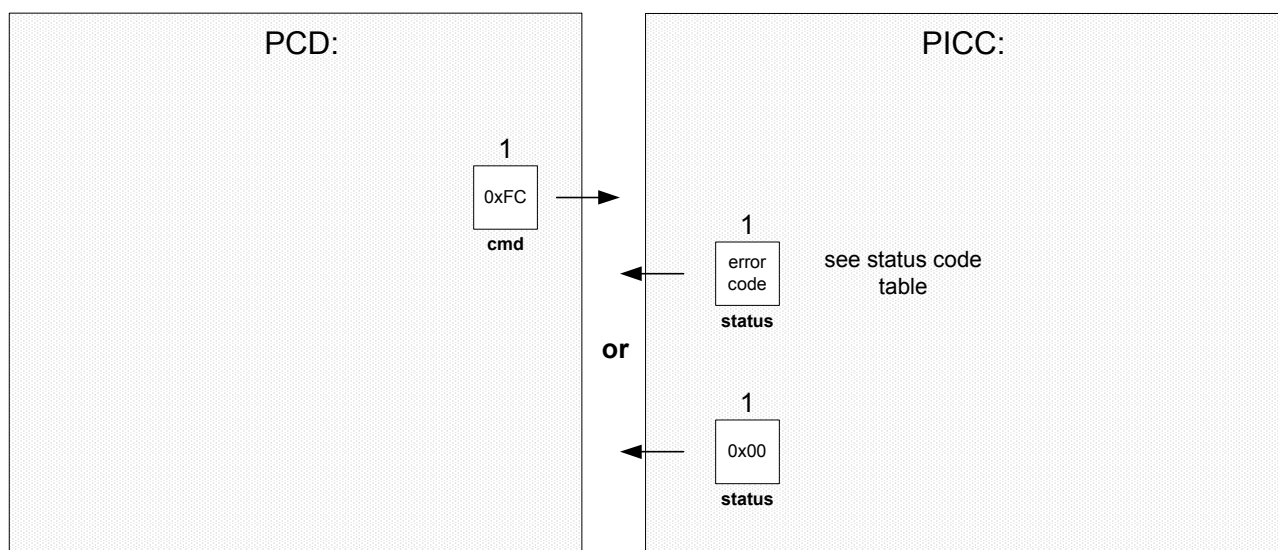
Functional Specification

mifare DESFire MF3 IC D40

4.4.5 FormatPICC

This command releases the PICC user memory.

FormatPICC() [1 byte]



No parameters are passed with this command.

The FormatPICC Command releases all allocated user memory on the PICC.

All applications are deleted and all files within those applications are deleted.

The PICC master key and the PICC master key settings keep their currently set values, they are not influenced by this command.

This command always requires a preceding authentication with the PICC master key.

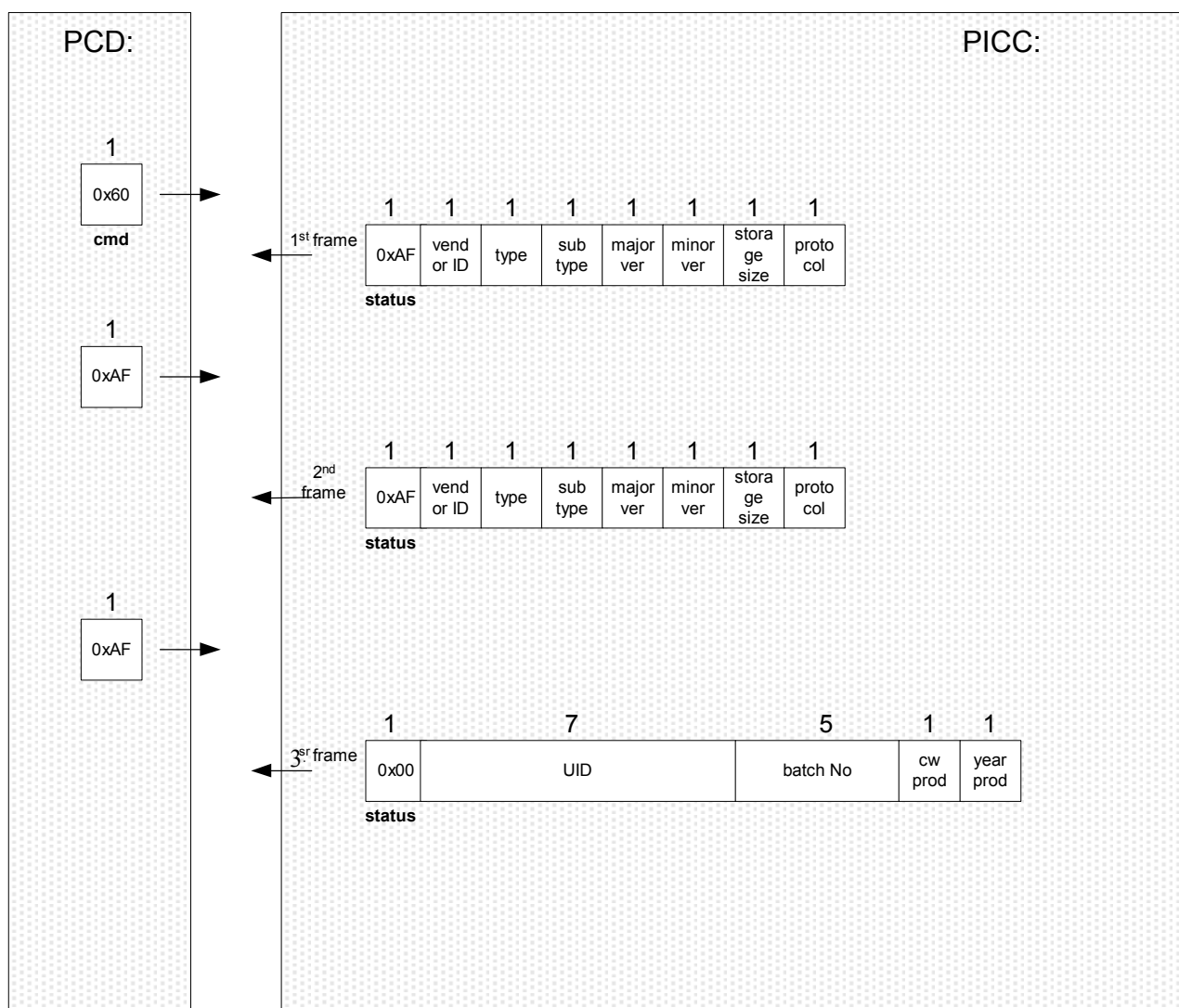
Functional Specification

mifare DESFire MF3 IC D40

4.4.6 GetVersion

The GetVersion command returns manufacturing related data of the PICC.

GetVersion [1 byte]



This command takes no parameter.

Functional Specification

mifare DESFire MF3 IC D40

Three frames of manufacturing related data are returned by the PICC:

The 1st frame: contains hardware related information:

- byte1: codes the vendor ID (0x04 for PHILIPS)
- byte2: codes the type (here 0x01)
- byte3: codes the subtype (here 0x01)
- byte4: codes the major version number (here 0x00)
- byte5: codes the minor version number (here 0x00)
- byte6: codes the storage size* (here 0x18 = 4096 bytes)
- byte7: codes the communication protocol type (here 0x05 meaning ISO 14443-4)

The 2nd frame contains software related information:

- byte1: codes the vendor ID (here 0x04 for PHILIPS)
- byte2: codes the type (here 0x01)
- byte3: codes the subtype (here 0x01)
- byte4: codes the major version (here 0x00)
- byte5: codes the minor version (here 0x00)
- byte6: codes the storage size* (here 0x18 = 4096 bytes)
- byte7: codes the communication protocol type (here 0x05 meaning ISO 14443-4)

The 3rd frame returns the unique serial number, batch number, year and calendar week of production:

- byte1 to byte7: code the unique serial number
- byte8 to byte12: code the production batch number
- byte13: codes the calendar week of production
- byte14: codes the year of production

* The 7 MSBits (= n) code the storage size itself based on 2^n , the LSBit is set to '0' if the size is exactly 2^n and set to '1' if the storage size is between 2^n and $2^{(n+1)}$. For this version of DESFire the 7 MSBits are set to 0x0C ($2^{12} = 4096$) and the LSBit is '0'.

Functional Specification

mifare DESFire MF3 IC D40

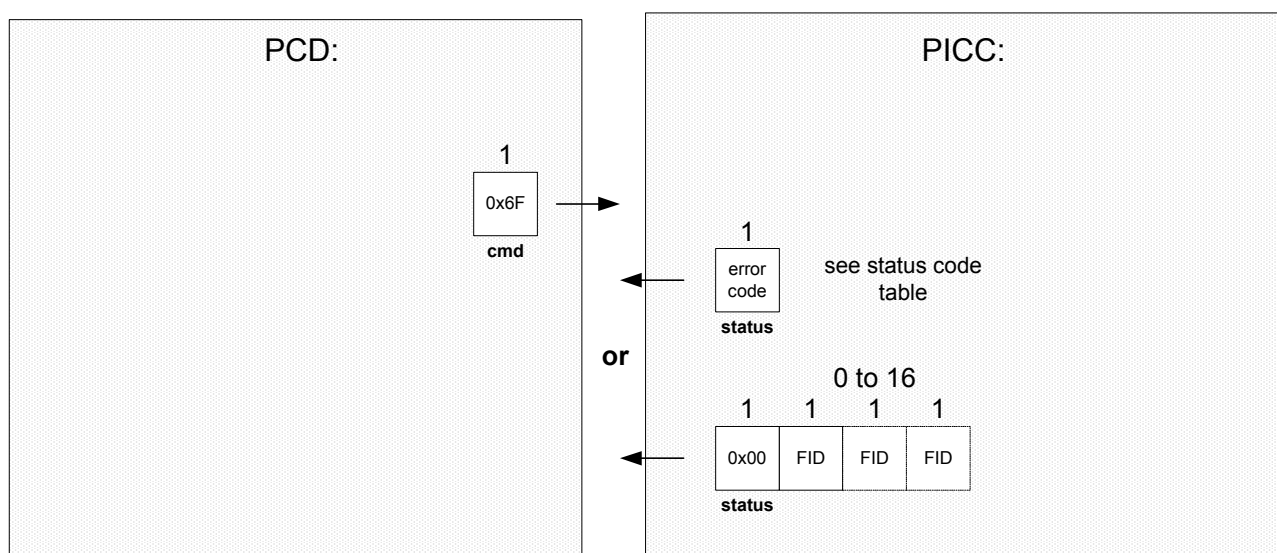
4.5 MF3 IC D40 Command Set – Application Level Commands:

The MF3 IC D40 provides the following command set for Application level functions:

4.5.1 GetFileIDs

The GetFileIDs command returns the File IDentifiers of all active files within the currently selected application.

GetFileIDs() [1 byte]



This command takes no parameters.

Depending on the application master key settings (see chapter 4.3.2), a preceding authentication with the application master key might be required.

Each File ID is coded in one byte and is in the range from 0x00 to 0x0F.

Duplicate values are not possible as each file must have an unambiguous identifier.

As the number of files is limited to eight within one application, the response always fits into one single data frame.

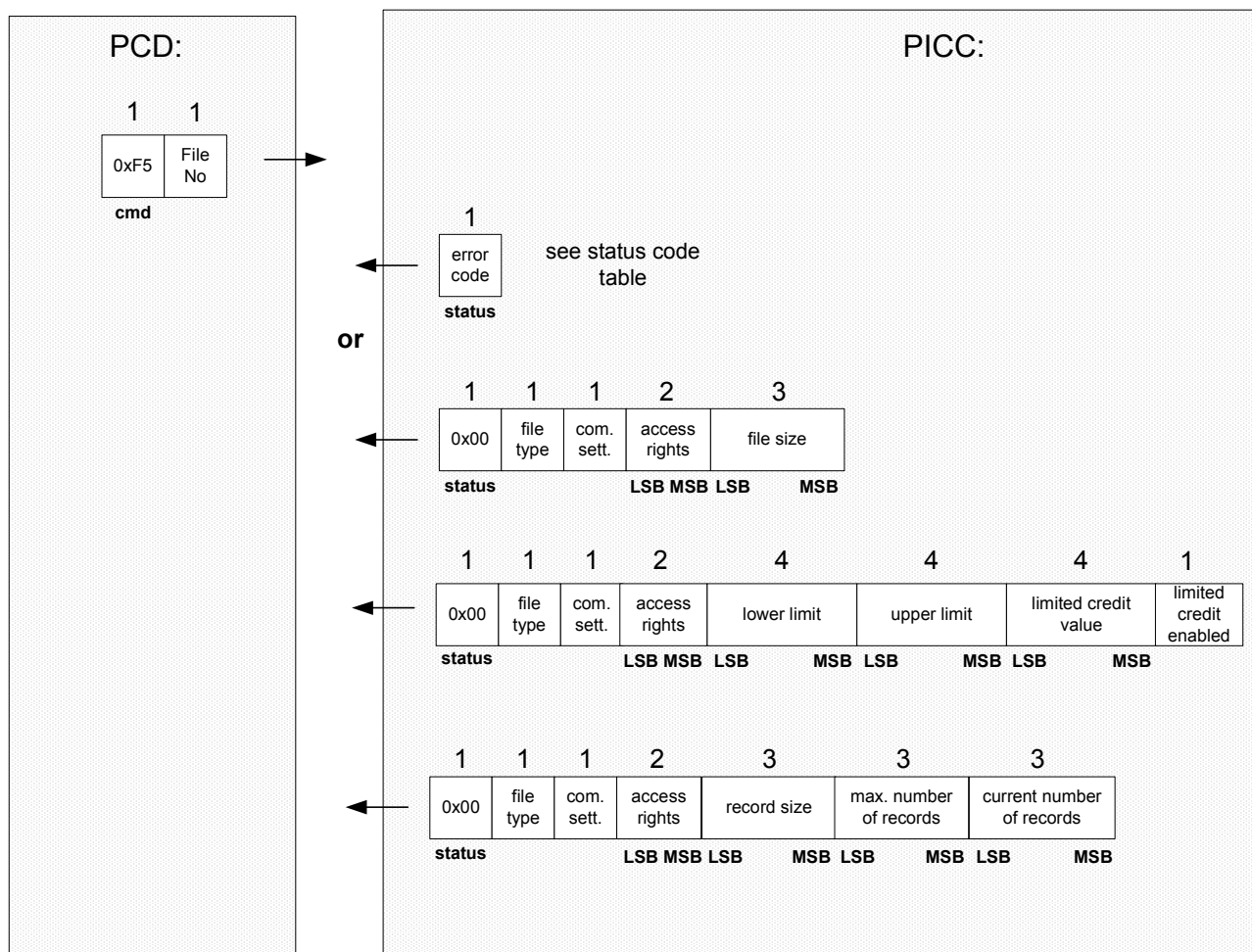
Functional Specification

mifare DESFire MF3 IC D40

4.5.2 GetFileSettings

The GetFileSettings command allows to get information on the properties of a specific file. The information provided by this command depends on the type of the file which is queried.

GetFileSettings() [2 byte]



The GetFileSettings command takes one parameter, coding the file number of the file to be queried within the currently selected application. This file number must be in the range between 0x00 and 0x0F.

Depending on the application master key settings (see chapter 4.3.2), a preceding authentication with the application master key might be required.

After updating a value file's value but before issuing the CommitTransaction command, the GetFileSettings command will always retrieve the old, unchanged limit for the limited credit value.

Functional Specification

mifare DESFire MF3 IC D40

The first part of the returned message is the same for all file types:

The first byte indicates the file's type, see chapter 3.1.

The next byte provides information on the file's communication settings (plain/MACed/Enciphered), see chapter 3.2.

This information is followed by the 2 byte file Access Rights field, see chapter 3.3.

All subsequent bytes in the response have a special meaning depending on the file type:

- Standard Data Files and Backup Data Files:

One field of three bytes length returns the user file size in bytes.

- Value Files:

Three fields, each of four bytes length, are returned whereby the first field returns the "lower limit" of the file (as defined at file creation), the second field returns the "upper limit" (as defined at file creation) and the next field returns the current maximum "limited credit" value, see chapter 4.6.5. If the limited credit functionality is not in use, the last field contains all zeros. The last byte codes, if the LimitedCredit command is allowed for this file (0x00 for disabled, 0x01 for enabled).

- Linear Record Files and Cyclic Record Files:

Three fields, each of three bytes length, are returned whereby the first field codes the size of one single record (as defined at file creation), the second field codes the maximum number of records within the record file (as defined at file creation) and the last field returns the current number of records within the record file. This number equals the maximum number of records which currently can be read, see chapter 4.6.8.

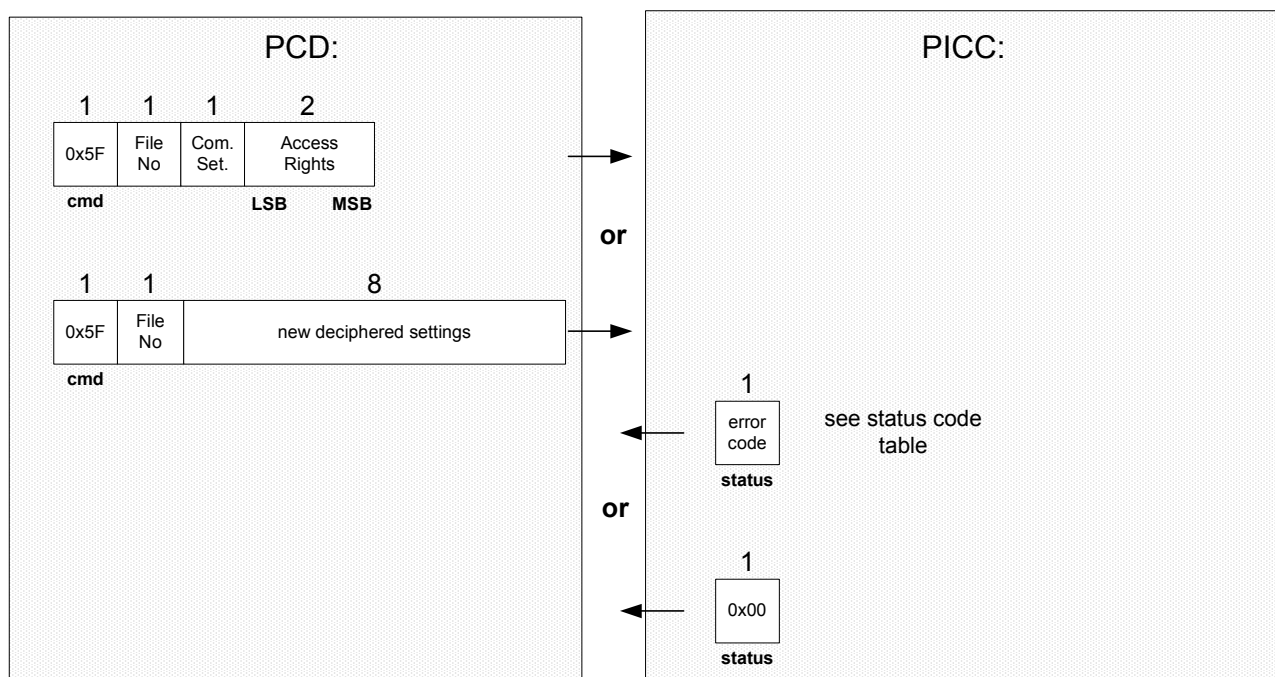
Functional Specification

mifare DESFire MF3 IC D40

4.5.3 ChangeFileSettings

This command changes the access parameters of an existing file.

ChangeFileSettings(FileNo,Com.Set.,AccessRights) [5/10 bytes]



As first parameter one byte is to be passed which codes the file number within the currently selected application.

The next byte defines the new communication settings, see chapter 3.2.

Finally a two byte field defines the new Access Rights, see chapter 3.3.

This change only succeeds if the current Access Rights for "Change Access Rights" is different from "never", see also chapter 3.3.

To guarantee that the ChangeFileSettings command is coming from the same party which did the preceding authentication, it is necessary to apply basically the same security mechanism as used with the ChangeKey command, see chapter 4.3.4:

A CRC is calculated over the new three byte settings and appended at the end. As this modified data stream now is of five bytes length, three bytes, all 0x00, are appended to get an eight bytes long data stream suitable for DES/3DES operation. Finally a DES/3DES decipherment is done on this data on PCD side.

The PICC now is capable of proving the authenticity of the received data by running a DES/3DES encipherment and checking the recovered CRC on the plain data.

However, if the ChangeAccessRights Access Rights is set with the value "ever", no security mechanism is necessary and therefore the data is sent as plain text (5 byte overall length).

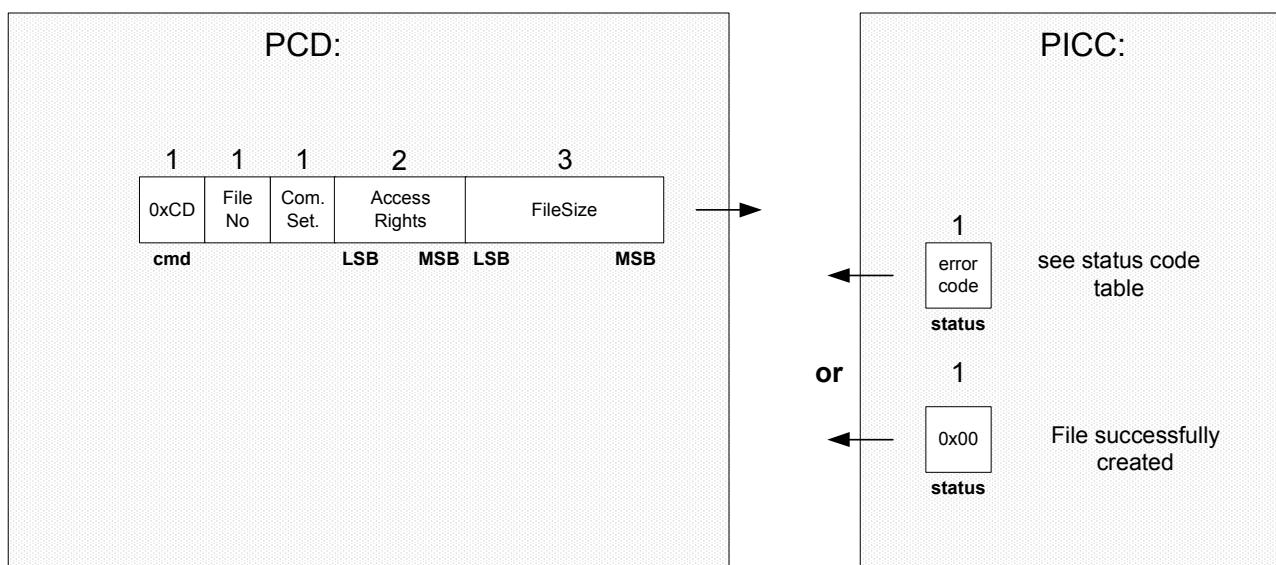
Functional Specification

mifare DESFire MF3 IC D40

4.5.4 CreateStdDataFile

The CreateStdDataFile command is used to create files for the storage of plain unformatted user data within an existing application on the PICC.

CreateStdDataFile(FileNo,Com.Set.,AccessRights,FileSize)
[8bytes]



As first parameter this command needs the file number of the new file within the range 0x00 to 0x0F. The file will be created in the currently selected application. It is not necessary to create the files within the application in a special order. If a file with the specified number already exists within the currently selected application, an error code is returned.

The next byte defines the communication settings, see chapter 3.2.

Then a two byte field defines the Access Rights for the new file, see chapter 3.3.

The last parameter of three byte length specifies the size of the file in bytes.

Note: The MF3 IC D40 internally allocates NV-memory in multiples of 32 bytes. Therefore a file creation command with FileSize parameter 0x00 00 01 (1 byte file size) will internally consume the same amount of NV-memory as a 0x00 00 20 (32 byte file size), namely 32 bytes.

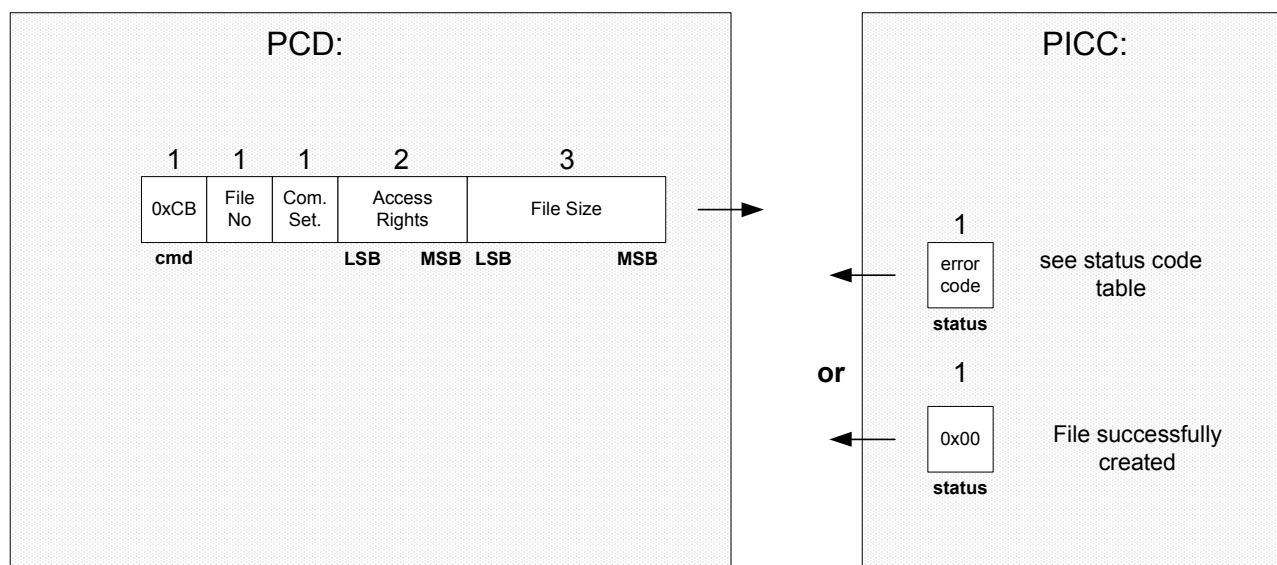
Functional Specification

mifare DESFire MF3 IC D40

4.5.5 CreateBackupDataFile

The CreateBackupDataFile command is used to create files for the storage of plain unformatted user data within an existing application on the PICC, additionally supporting the feature of an integrated backup mechanism.

CreateBackupDataFile(FileNo,Com.Set.,AccessRights,FileSize)
[8bytes]



As the name "BackupDataFile" implies, files of this type feature an integrated backup mechanism.

Every Write command is done in a independent mirror image of this file. To validate a write access to this file type, it is necessary to confirm it with a CommitTransaction command, see chapter 4.6.10. If no CommitTransaction command is send by the PCD, only the mirror image is changed, the original data remains unchanged and valid.

All parameters have the same format as for the CreateStdDataFile command, see chapter 4.5.4, except the parameter FileNo. As only the first 8 bytes within an application feature the integrated backup mechanism, only FileNo 0x00 to 0x07 is allowed.

Due to the mirror image a BackupDataFile always consumes DOUBLE the NV-memory on the PICC compared to a StdDataFile with the same specified FileSize.

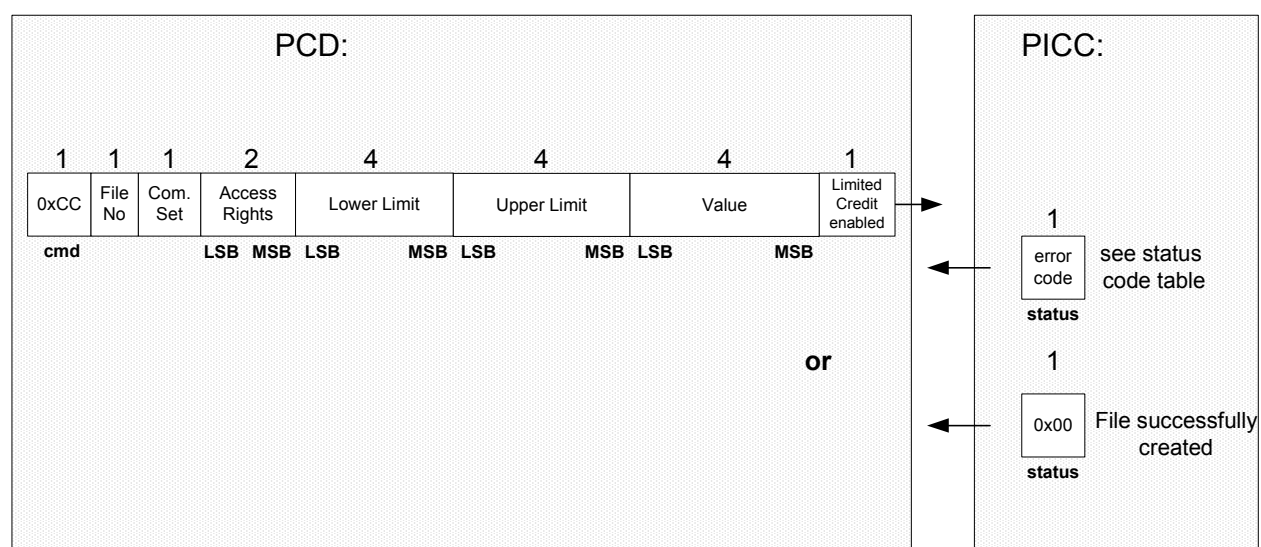
Functional Specification

mifare DESFire MF3 IC D40

4.5.6 CreateValueFile

The CreateValueFile command is used to create files for the storage and manipulation of 32bit signed integer values within an existing application on the PICC.

**CreateValueFile(FileNo,Com.Set.,AccessRights,LowerLimit,
UpperLimit,Value,LimitedCreditEnabled)**
[18 bytes]



As first parameter one byte codes the file number in the range 0x00 to 0x07 which the new created file should get within the currently selected application.

The next byte defines the communication settings, see chapter 3.2

Then a two byte field defines the Access Rights for the new file, see chapter 3.3.

The next parameter is of 4 byte length and codes the lower limit which is valid for this file. The lower limit marks the boundary which must not be passed by a Debit calculation on the current value, see chapter 4.6.5. The lower limit is a 4 byte signed integer and thus may be negative too.

After this again 4 bytes are used to code the upper limit which sets the boundary in the same manner but for the Credit operation, see chapter 4.6.4. This parameter is also a 4 byte signed integer.

The upper limit has to be \geq lower limit, otherwise an error message would be sent by the PICC and thus the file would not be created.

The next parameter is a 4 byte signed integer again and specifies the initial value of the value file. The upper and lower limit is checked by the PICC, in case of inconsistency the file is not created and an error message is sent by the PICC.

The last bytes codes the activation of the LimitedCredit feature, see chapter 4.6.6. Here 0x00 means that LimitedCredit is disabled and 0x01 enables this feature.

ValueFiles feature always the integrated backup mechanism. Therefore every access changing the value needs to be validated using the CommitTransaction command, see chapter 4.6.10.

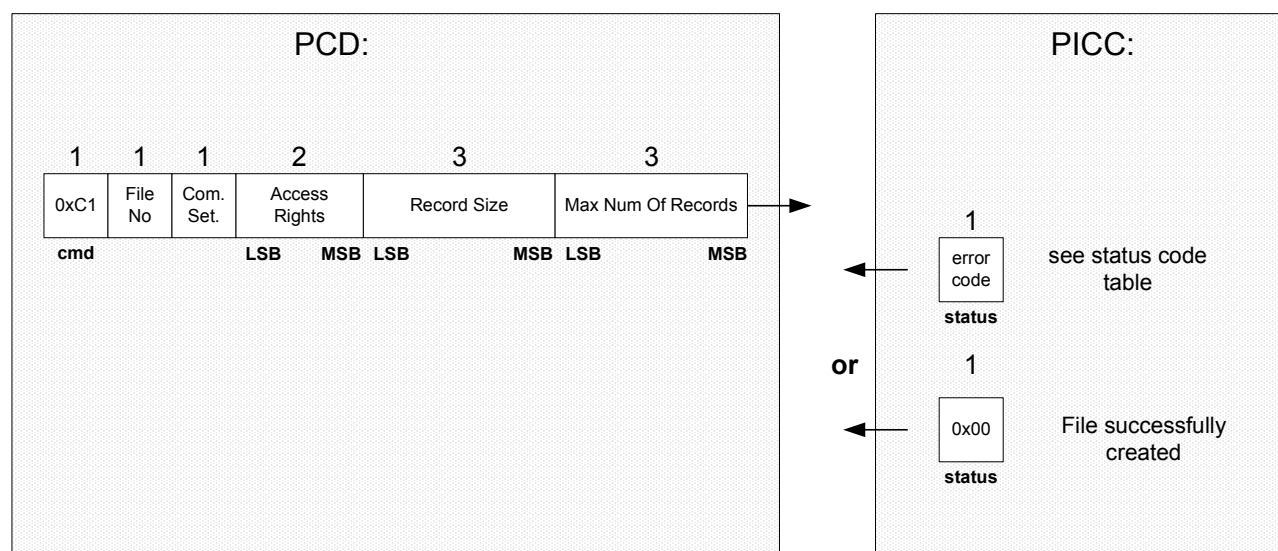
Functional Specification

mifare DESFire MF3 IC D40

4.5.7 CreateLinearRecordFile

The CreateLinearRecordFile command is used to create files for multiple storage of structural similar data, for example for loyalty programs, within an existing application on the PICC. Once the file is filled completely with data records, further writing to the file is not possible unless it is cleared, see command ClearRecordFile, chapter 4.6.9.

**CreateLinearRecordFile(FileNo,Com.Set.,AccessRights,Rec.Size,
MaxNumOfRecords) [11 bytes]**



As first parameter one byte codes the file number in the range 0x00 to 0x07 which the new created file should get within the currently selected application.

The next byte defines the communication settings, see chapter 3.2.

Then a two byte field defines the Access Rights for the new file, see chapter 3.3.

The next parameter is of three bytes length and codes the size of one single record in bytes. This parameter has to be in the range from 0x00 00 01 to 0xFF FF FF.

The last parameter is also of three bytes length and codes the number of records. This parameter has to be in the range from 0x00 00 01 to 0xFF FF FF, too.

Thus the entire file size in the PICC NV-memory is given by $\text{RecordSize} \times \text{MaxNumOfRecords}$.

Linear Record Files feature always the integrated backup mechanism. Therefore every access appending a record needs to be validated using the CommitTransaction command, see chapter 4.6.10.

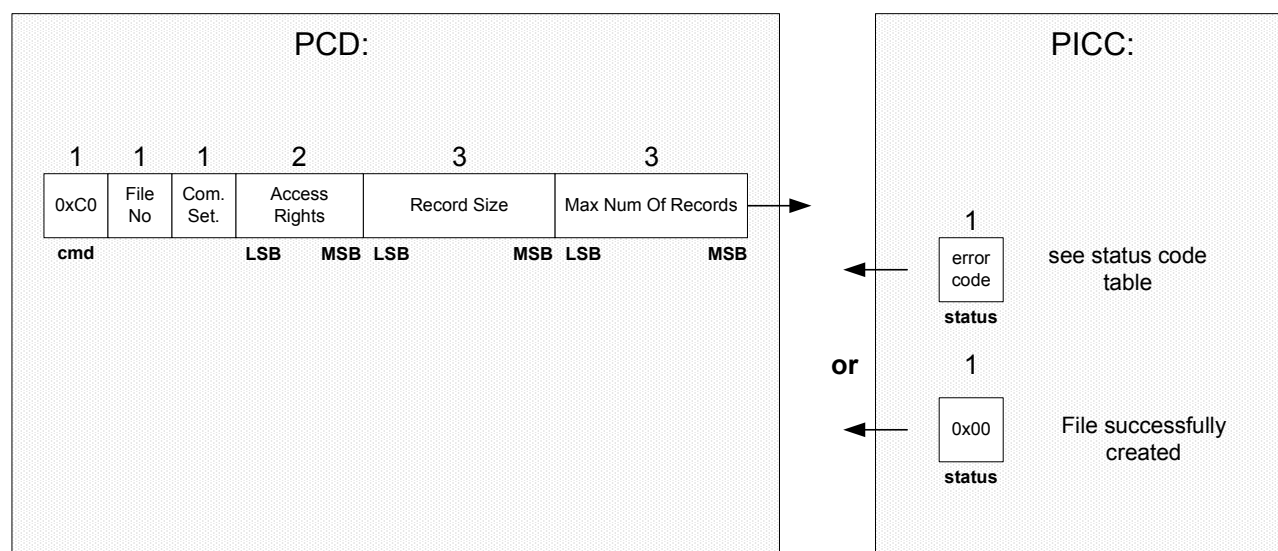
Functional Specification

mifare DESFire MF3 IC D40

4.5.8 CreateCyclicRecordFile

The CreateCyclicRecordFile command is used to create files for multiple storage of structural similar data, for example for logging transactions, within an existing application on the PICC. Once the file is filled completely with data records, the PICC automatically overwrites the oldest record with the latest written one. This wrap is fully transparent for the PCD.

CreateCyclicRecordFile(FileNo, Com.Set., AccessRights, Rec.Size, MaxNumOfRecords) [11 bytes]



As first parameter one byte codes the file number in the range 0x00 to 0x07 which the new created file should get within the currently selected application.

The next byte defines the communication settings, see chapter 3.2.

Then a two byte field defines the Access Rights for the new file, see chapter 3.3.

The next parameter is of three bytes length and codes the size of one single record in bytes. This parameter has to be in the range from 0x00 00 01 to 0xFF FF FF.

The last parameter is also of three bytes length and codes the number of records. This parameter has to be in the range from 0x00 00 02 to 0xFF FF FF, too.

Thus the entire file size in the PICC NV-memory is given by $\text{RecordSize} * \text{MaxNumOfRecords}$.

Cyclic Record Files feature always the integrated backup mechanism. Therefore every access appending a record needs to be validated using the CommitTransaction command, see chapter 4.6.10.

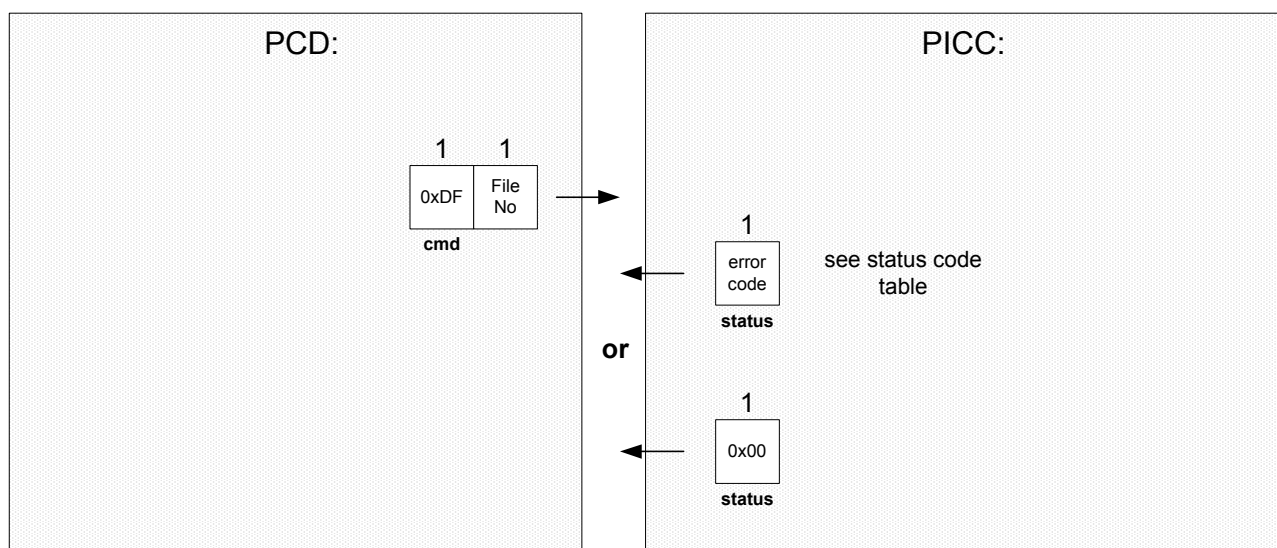
Functional Specification

mifare DESFire MF3 IC D40

4.5.9 DeleteFile

The DeleteFile command permanently deactivates a file within the file directory of the currently selected application.

DeleteFile(FileNo) [2 bytes]



This command takes one byte as parameter coding the file number which is to be in the range from 0x00 to 0x0F.

The operation of this command invalidates the file directory entry of the specified file which means that the file can't be accessed anymore.

Depending on the application master key settings, see chapter 4.3.2, a preceding authentication with the application master key is required.

Allocated memory blocks associated with the deleted file are not set free. The FileNo of the deleted file can be re-used to create a new file within that application.

To release memory blocks for re-use, the whole PICC user NV-memory needs to be erased using the FormatPICC command, see chapter 4.4.5.

Functional Specification

mifare DESFire MF3 IC D40

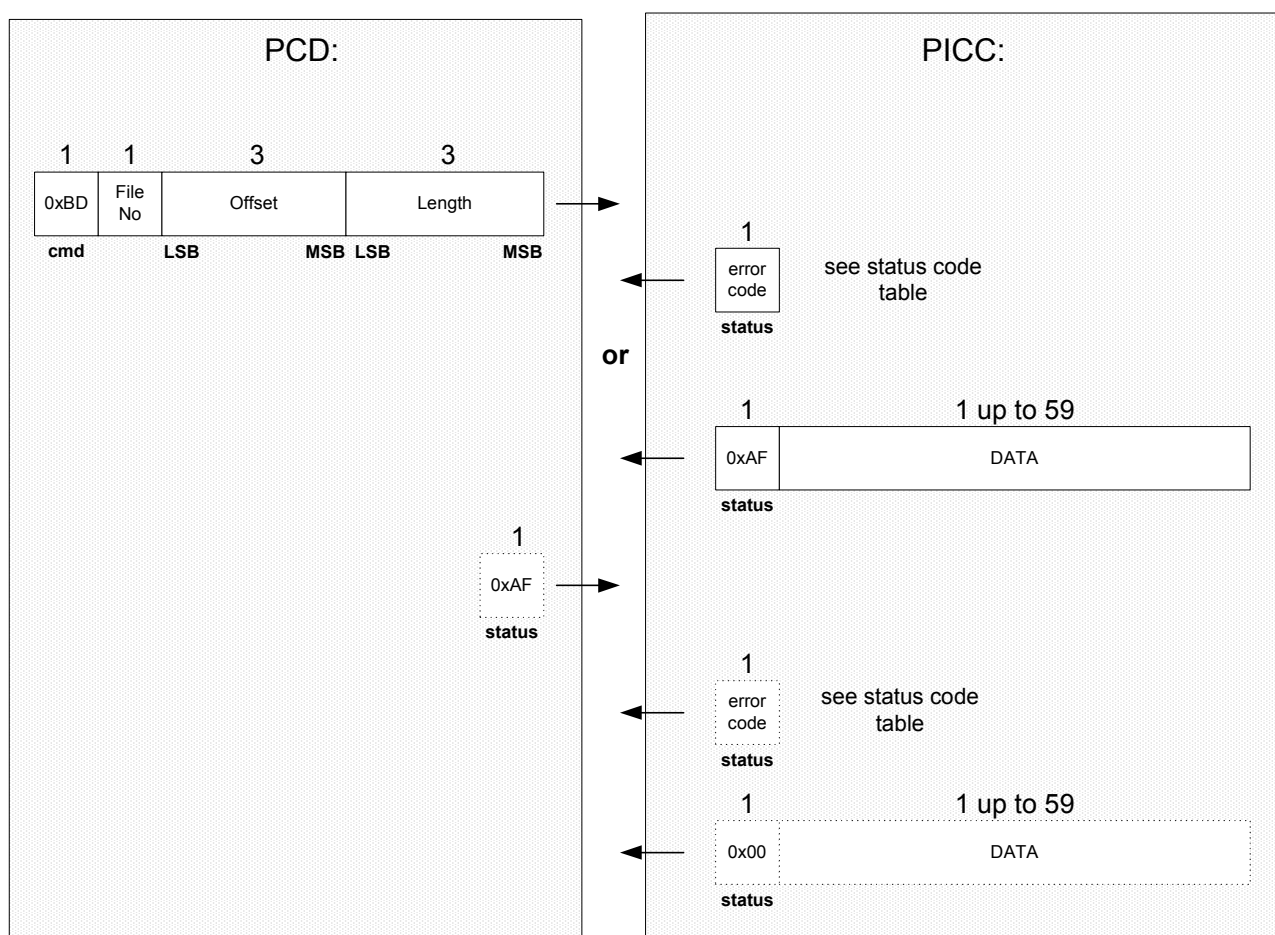
4.6 MF3 IC D40 Command Set – Data Manipulation Commands

The MF3 IC D40 provides the following command set for Data Manipulation:

4.6.1 ReadData

The ReadData command allows to read data from Standard Data Files or Backup Data Files.

ReadData(FileNo,Offset,Length) [8bytes]



The first parameter is of 1 byte length and defines the file number to be read from. This parameter has to be in the range from 0x00 to 0x0F for Standard Data Files and 0x00 to 0x07 for Backup Data Files, respectively.

The next parameter is of three byte length and codes the starting position for the read operation within the file (= offset value). This parameter has to be in the range from 0x00 00 00 to file size -1.

The third parameter is also three byte long and specifies the number of data bytes to be read. This parameter can be in the range from 0x00 00 00 to 0xFF FF FF.

If the third parameter is coded as 0x00 00 00, the entire data file, starting from the position specified in the offset value, is read.

Functional Specification**mifare DESFire MF3 IC D40**

If the number of bytes to send to the PCD does not fit into one single frame, the PICC waits for a status frame with status byte 0xAF, before the next frame is sent to the PCD.

If Backup Data Files are read after writing to them, but before issuing the CommitTransaction command, see chapter 4.6.10, the ReadData command will always retrieve the old, unchanged data stored in the PICC. All data written to a Backup Data File is validated and externally “visible” for a ReadData command only after a CommitTransaction command.

The Read command requires a preceding authentication either with the key specified for “Read” or “Read&Write” access, see chapter 3.3.

Depending on the communication settings, see chapter 3.2, linked to the file, data will be sent by the PICC either plain, MACed or enciphered. All cryptographic operations are done in CBC mode.

For MACed- and enciphered communication padding on the data is necessary to reach an overall data-length of multiples of eight. In case of a specified data length (command parameter Length \neq 0x00 00 00) padding is done with all 0x00. In case that the file is read until it's limit (command parameter Length = 0x00 00 00), the first byte appended for padding is 0x80, all other bytes appended are 0x00 (in accordance with ISO 9797-1).

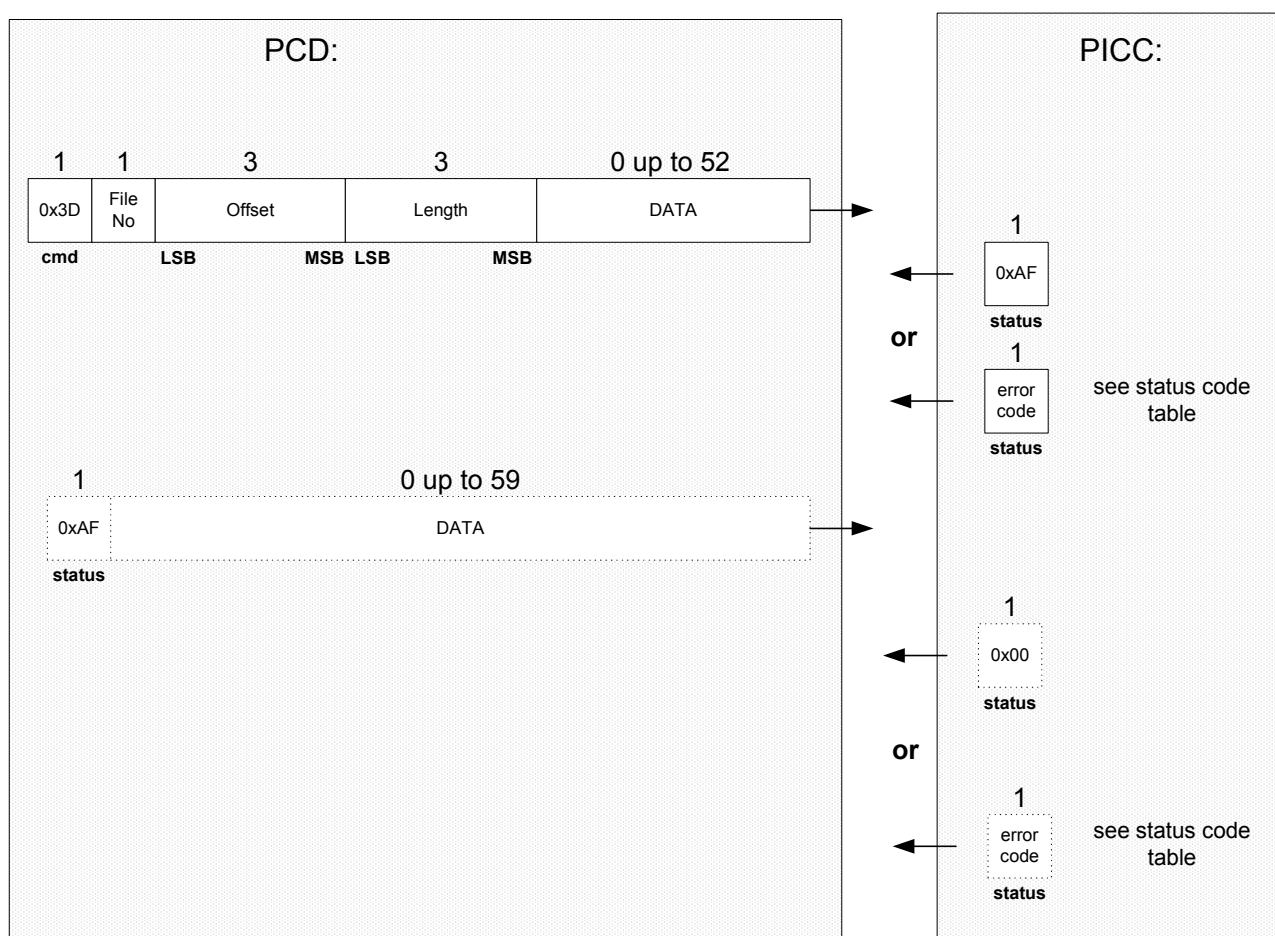
Functional Specification

mifare DESFire MF3 IC D40

4.6.2 WriteData

The WriteData command allows to write data to Standard Data Files and Backup Data Files.

WriteData(FileNo,Offset,Length,Data) [8+bytes]



The first parameter is of 1 byte length and defines the file number to be written to; valid range is 0x00 to 0x0F for Standard Data Files and 0x00 to 0x07 for Backup Data Files, respectively.

The next parameter is of three bytes length and codes the starting position for the write operation within the file (= offset value). This parameter has to be in the range from 0x00 00 00 to file size -1.

The third parameter is also three bytes long and specifies the number of data bytes to be written. This parameter can be in the range from 0x00 00 01 to 0xFF FF FF.

If the number of bytes to send does not fit into one single frame, the PCD has to wait for a status frame with status byte 0xAF before sending the next frame to the PICC.

The Write command requires a preceding authentication either with the key specified for "Write" or "Read&Write" access, see chapter 3.3.

Functional Specification

mifare DESFire MF3 IC D40

Depending on the communication settings, see chapter 3.2, linked to the file, data needs to be sent by the PCD either plain, MACed or enciphered. All cryptographic operations are done in CBC mode.

For MACed and enciphered communication padding of the data string is necessary to reach an overall data-length of multiples of eight. This padding is done with all 0x00.

If the WriteData operation is performed on a Backup Data File, it is necessary to validate the written data with a CommitTransaction command, see chapter 4.6.10. An AbortTransaction command will invalidate all changes, see chapter 4.6.11.

If data is written to Standard Data Files (without integrated backup mechanism), data is directly programmed into the visible NV-memory of the file. The new data is immediately available to any following ReadData commands performed on that file.

In case of MACed or enciphered communication the validity of data is verified by the PICC by checking the MAC or the CRC (including necessary padding bytes) which is transmitted at the end of the data frame. If the verification fails (MAC / CRC does not fit data, padding bytes invalid), the PICC stops further NV-programming and returns an Integrity Error to the PCD, see chapter 3.4. As a consequence of the Integrity Error any transaction, which might have begun, is automatically aborted.

Note: Getting an Integrity Error when writing on a Standard Data File can corrupt the content of the file.

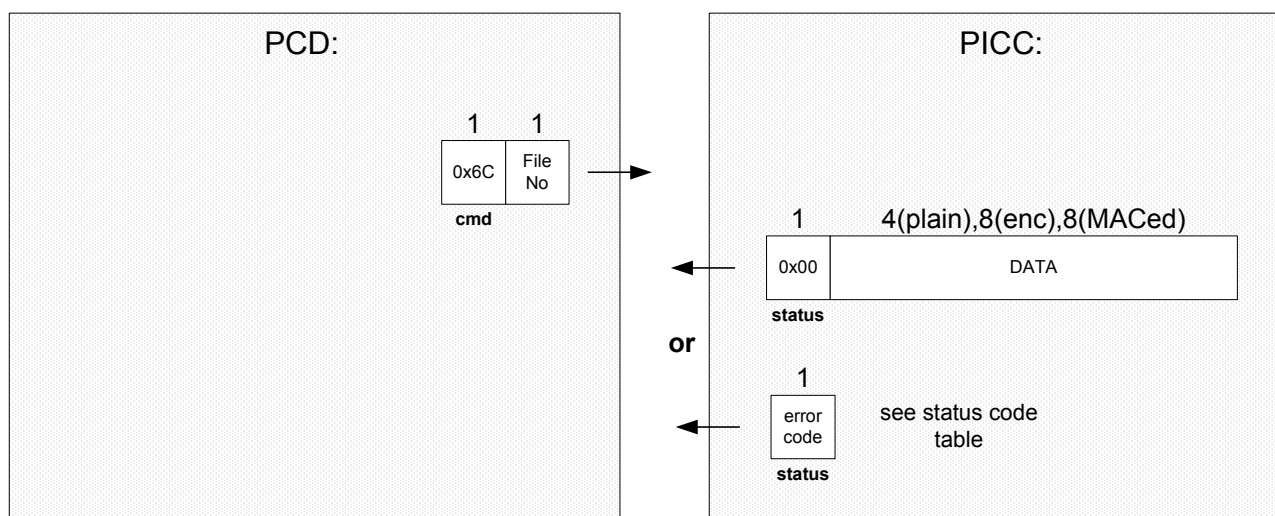
Functional Specification

mifare DESFire MF3 IC D40

4.6.3 GetValue

The GetValue command allows to read the currently stored value from Value Files.

GetValue(FileNo) [2 bytes]



The only parameter sent with this command is of one byte length and codes the file number. This parameter has to be in the range from 0x00 to 0x07.

The PICC response holds the current value of the Value file. Depending on the communication mode data is either transferred plain (4 bytes), enciphered (8 bytes enciphered data: 4 bytes value, 2 bytes CRC, 2 bytes padding with 0x00) or MACed (4 bytes value + 4 bytes MAC).

The value is always represented LSB first.

The GetValue command requires a preceding authentication with the key specified for Read, Write or Read&Write access, see chapter 3.3.

After updating a value file's value but before issuing the CommitTransaction command, the GetValue command will always retrieve the old, unchanged value which is still the valid one.

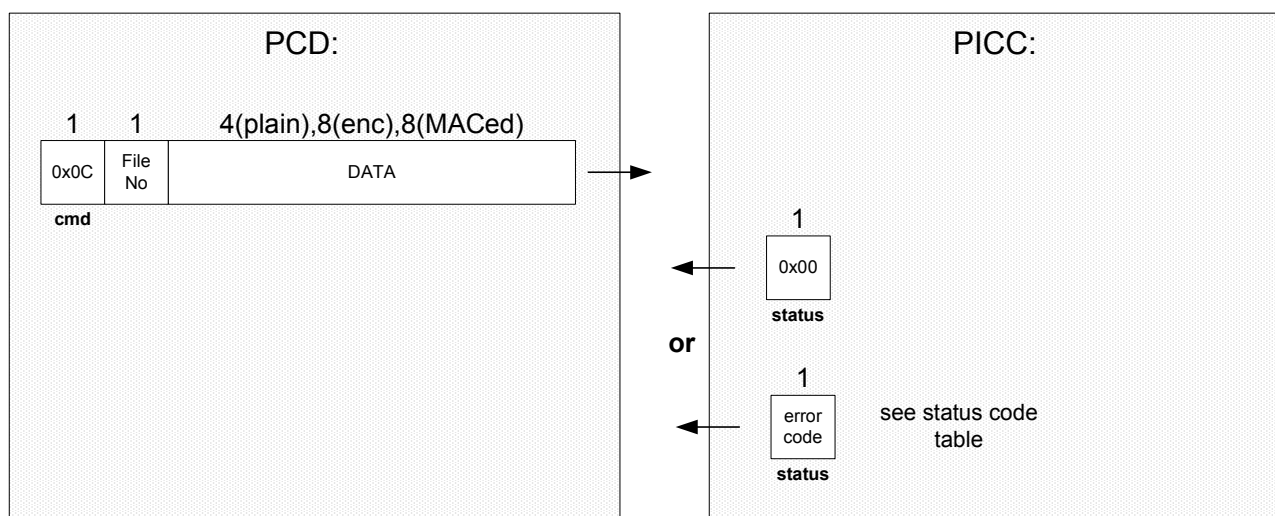
Functional Specification

mifare DESFire MF3 IC D40

4.6.4 Credit

The Credit command allows to increase a value stored in a Value File.

Credit(FileNo,Data) [6/10 bytes]



The first parameter sent with this command is of one byte length and codes the file number. This parameter has to be in the range from 0x00 to 0x07.

This command increases the current value stored in the file by a certain amount (4 byte signed integer) which is transmitted in the data field. Only positive values are allowed for the Credit command.

Depending on the communication mode the value is either transferred plain (4 bytes), enciphered (8 bytes enciphered data: 4 bytes value, 2 bytes CRC, 2 bytes padding with 0x00) or MACed (4 bytes value + 4 bytes MAC).

The value is always represented LSB first.

It is necessary to validate the updated value with a CommitTransaction command, see chapter 4.6.10. An AbortTransaction command will invalidate all changes, see chapter 4.6.11.

The value modifications of Credit, Debit and LimitedCredit commands are cumulated until a CommitTransaction command is issued.

Credit commands do NEVER modify the Limited Credit Value of a Value file. However, if the Limited Credit Value needs to be set to 0, a LimitedCredit with value 0 can be used.

The Credit command requires a preceding authentication with the key specified for "Read&Write" access, see chapter 3.3.

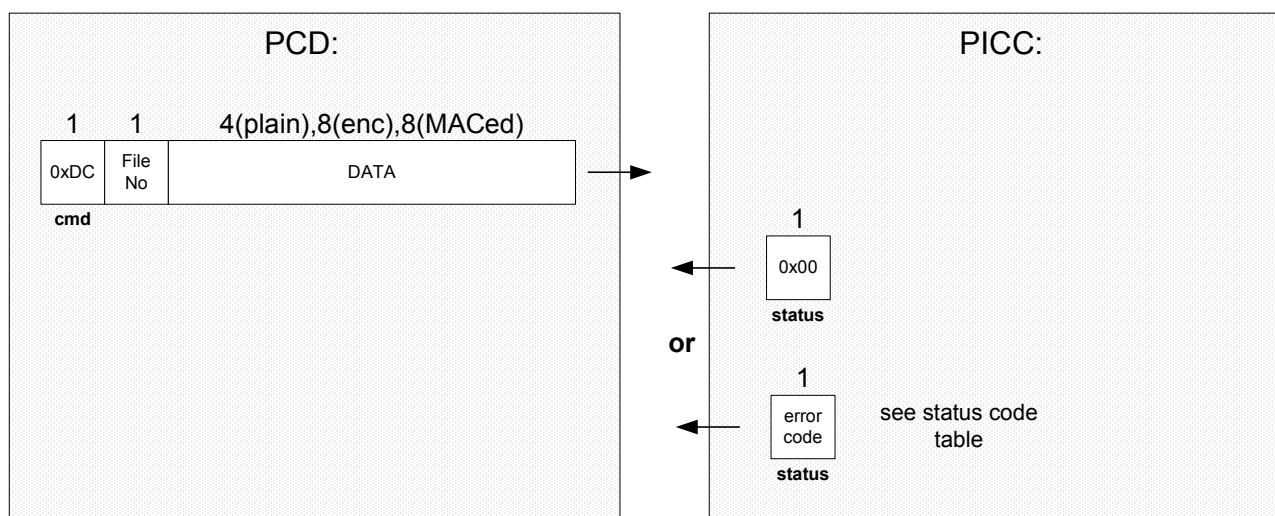
Functional Specification

mifare DESFire MF3 IC D40

4.6.5 Debit

The Debit command allows to decrease a value stored in a Value File.

Debit(FileNo,Data) [6/10 bytes]



The first parameter is of one byte length and codes the file number. This parameter has to be in the range from 0x00 to 0x07.

The next parameter (4 byte signed integer) holds the value which will be subtracted from the current value stored in the file. Only positive values are allowed for the Debit command.

Depending on the communication mode the value is either transferred plain (4 bytes), enciphered (8 bytes enciphered data: 4 bytes value, 2 bytes CRC, 2 bytes padding with 0x00) or MACed (4 bytes value + 4 bytes MAC).

The value is always represented LSB first.

It is necessary to validate the updated value with a CommitTransaction command, see chapter 4.6.10. An AbortTransaction command will invalidate all changes, see chapter 4.6.11.

The value modifications of Credit, Debit and LimitedCredit commands are cumulated until a CommitTransaction command is issued.

The Debit command requires a preceding authentication with one of the keys specified for Read, Write or Read&Write access, see chapter 3.3.

If the usage of the LimitedCredit feature is enabled, the new limit for a subsequent LimitedCredit command is set to the sum of Debit commands within one transaction before issuing a CommitTransaction command. This assures that a LimitedCredit command can not re-book more values than a debiting transaction deducted before.

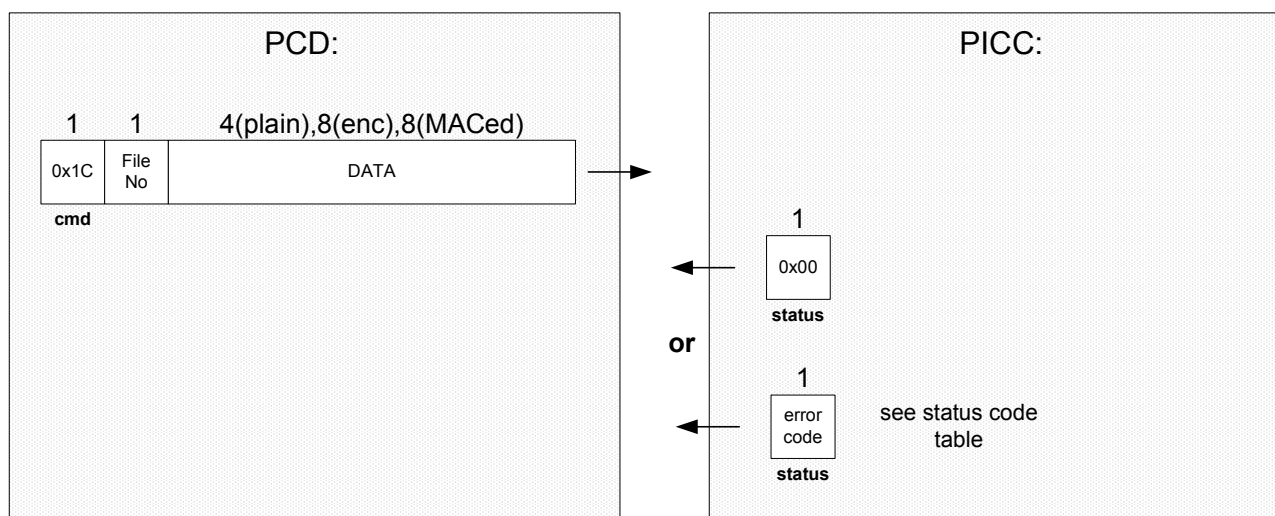
Functional Specification

mifare DESFire MF3 IC D40

4.6.6 LimitedCredit

The LimitedCredit command allows a limited increase of a value stored in a Value File without having full Read&Write permissions to the file. This feature can be enabled or disabled during value file creation.

LimitedCredit(FileNo,Data) [6/10 bytes]



The first parameter sent with this command is of one byte length and codes the file number. This parameter has to be in the range from 0x00 to 0x07.

This command increases the current value stored in the file by a certain amount (4 byte signed integer) which is transmitted in the data field. Only positive values are allowed for the LimitedCredit command.

Depending on the communication mode the value is either transferred plain (4 bytes), enciphered (8 bytes enciphered data: 4 bytes value, 2 bytes CRC, 2 bytes padding with 0x00) or MACed (4 bytes value + 4 bytes MAC).

The value is always represented LSB first.

It is necessary to validate the updated value with a CommitTransaction command, see chapter 4.6.10. An AbortTransaction command will invalidate all changes, see chapter 4.6.11.

The value modifications of Credit, Debit and LimitedCredit commands are cumulated until a CommitTransaction command is issued.

The LimitedCredit command requires a preceding authentication with the key specified for "Write" or "Read & Write" access, see chapter 3.3.

The value for LimitedCredit is limited to the sum of the Debit commands on this value file within the most recent transaction containing at least one Debit. After executing the LimitedCredit command the new limit is set to 0 regardless of the amount which has been re-booked. Therefore the LimitedCredit command can only be used **once** after a Debit transaction.

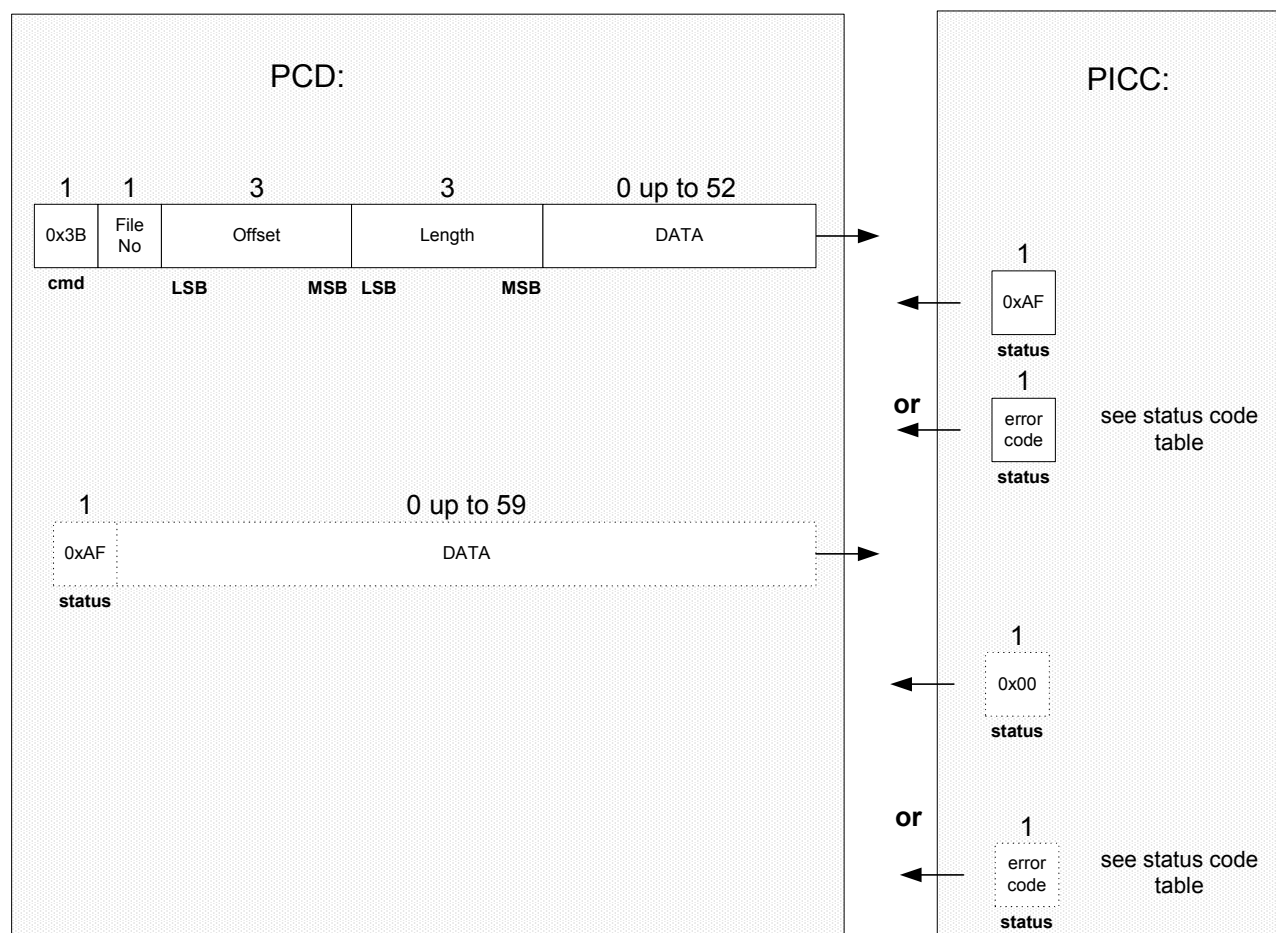
Functional Specification

mifare DESFire MF3 IC D40

4.6.7 WriteRecord

The WriteRecord command allows to write data to a record in a Cyclic or Linear Record File.

WriteRecord(FileNo,Offset,Length,Data) [8+bytes]



The first parameter is of one byte length and codes the file number. This parameter has to be in the range from 0x00 to 0x07.

The next three bytes code the offset within one single record (in bytes). This parameter has to be in the range from 0x00 00 00 to record size - 1.

This parameter is followed by another three bytes coding the length of data which is to be written to the record file. This parameter has to be in the range from 0x00 00 01 to record size - offset.

The WriteRecord command appends one record at the end of the linear record file, it erases and overwrites the oldest record in case of a cyclic record file if it is already full. The entire new record is cleared before data is written to it.

If no CommitTransaction command (see chapter 4.6.10) is sent after a WriteRecord command, the next WriteRecord command to the same file writes to the already created record. After sending a CommitTransaction command, a new WriteRecord command will create a new record in the record file. An AbortTransaction command will invalidate all changes, see chapter 4.6.11.

Functional Specification

mifare DESFire MF3 IC D40

After issuing a ClearRecordFile command, but before a CommitTransaction / AbortTransaction command, a WriteRecord command to the same record file will fail.

Depending on the communication settings, see chapter 3.2, linked to the file, data needs to be sent by the PCD either plain, MACed or enciphered. All cryptographic operations are done in CBC mode.

For MACed- and enciphered communication padding on the data is necessary to reach an overall data-length of multiples of eight. This padding is done with all 0x00.

The WriteRecord command requires a preceding authentication either with the key specified for "Write" or "Read&Write" access, see chapter 3.3.

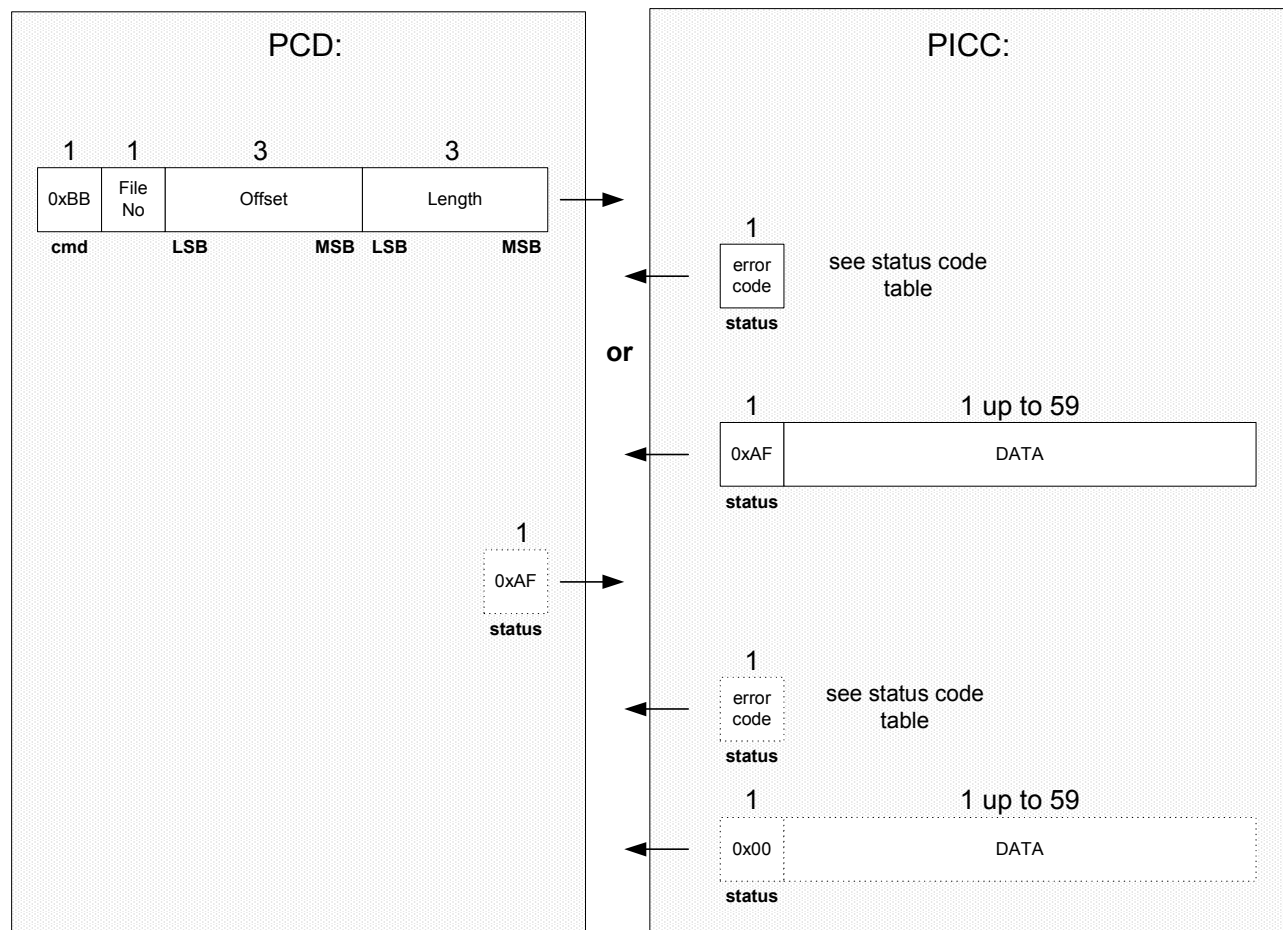
Functional Specification

mifare DESFire MF3 IC D40

4.6.8 ReadRecords

The ReadRecords command allows to read out a set of complete records from a Cyclic or Linear Record File.

ReadRecords(FileNo,Offset,Length) [8bytes]



The first parameter is of one byte length and codes the file number in the range from 0x00 to 0x07.

The next parameter is three bytes long and codes the offset of the newest record which is read out. In case of 0x00 00 00 the latest record is read out. The offset value must be in the range from 0x00 to number of existing records – 1.

The third parameter is another three bytes which code the number of records to be read from the PICC. Records are always transmitted by the PICC in chronological order (= starting with the oldest, which is number of records – 1 before the one addressed by the given offset). If this parameter is set to 0x00 00 00 then all records, from the oldest record up to and including the newest record (given by the offset parameter) are read.

The allowed range for the number of records parameter is from 0x00 00 00 to number of existing records – offset. Note that in cyclic record files the maximum number of stored valid records is one less than the number of records specified in the CreateCyclicRecordFile command.

Functional Specification

mifare DESFire MF3 IC D40

A ReadRecords command on an empty record file (directly after creation or after a committed clearance, see chapter 4.6.9) will result in an error.

Depending on the communication settings, see chapter 3.2, linked to the file, data will be sent by the PICC either plain, MACed or enciphered. All cryptographic operations are done in CBC mode.

For MACed- and enciphered communication padding on the data is necessary to reach an overall data-length of multiples of eight. In case of a specified number of records (command parameter Length \neq 0x00 00 00) padding is done with all 0x00. In case the file is read until it's limit (command parameter Length = 0x00 00 00), the first byte appended for padding is 0x80, all other bytes appended are 0x00 (in accordance with ISO 9797-1).

The ReadRecords command requires a preceding authentication either with the key specified for "Read" or "Read&Write" access, see chapter 3.3.

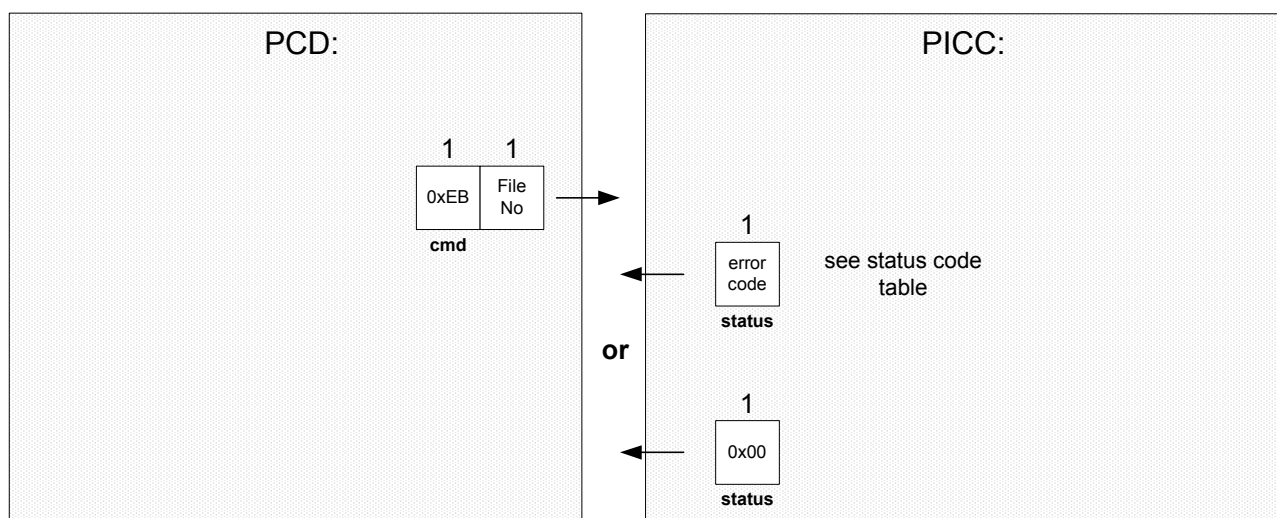
Functional Specification

mifare DESFire MF3 IC D40

4.6.9 ClearRecordFile

The ClearRecordFile command allows to reset a Cyclic or Linear Record File to the empty state.

ClearRecordFile(FileNo) [2 bytes]



The only parameter sent with this command is of one byte length and codes the file number in the range from 0x00 to 0x07.

After executing the ClearRecordFile command but before CommitTransaction, all subsequent WriteRecord commands, see chapter 4.6.7, will fail. The ReadRecords command, see chapter 4.6.8, will return the old still valid records.

After the CommitTransaction command is issued, a ReadRecords command will fail, WriteRecord commands will be successful.

An AbortTransaction command (instead of CommitTransaction) will invalidate the clearance, see chapter 4.6.11.

Full "Read&Write" permission on the file is necessary for executing this command, see chapter 3.3.

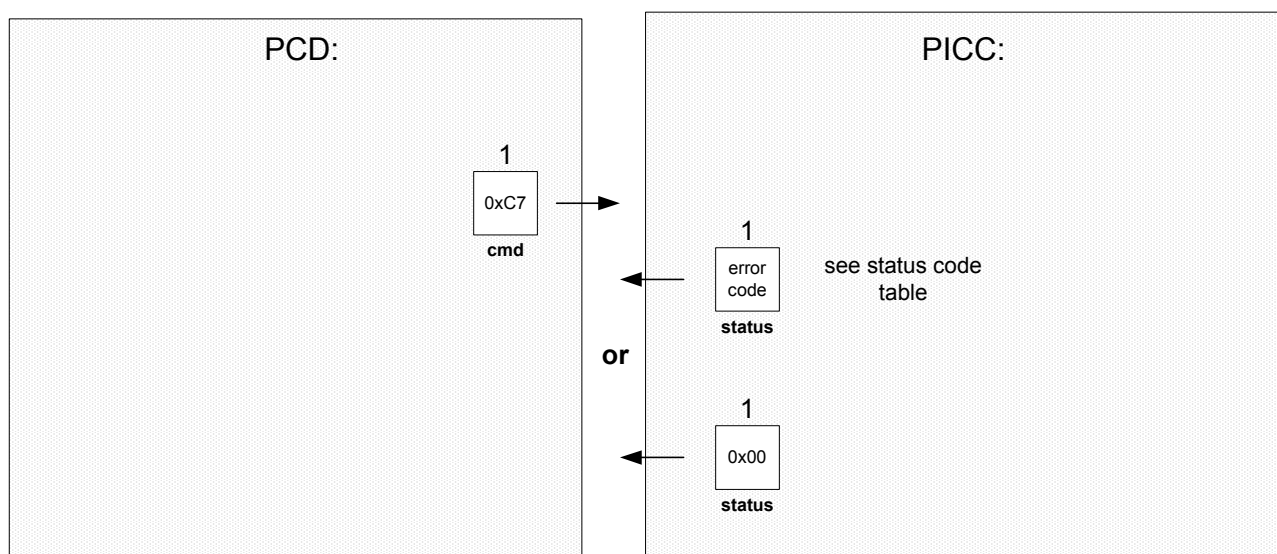
Functional Specification

mifare DESFire MF3 IC D40

4.6.10 CommitTransaction

The CommitTransaction command allows to validate all previous write access on Backup Data Files, Value Files and Record Files within one application.

CommitTransaction(AID) [1 bytes]



This command takes no parameter.

The CommitTransaction command validates all write access to files with integrated backup mechanisms:

- Backup Data Files
- Value Files
- Linear Record Files
- Cyclic Record Files

The CommitTransaction is typically the last command of a transaction before the ISO 14443-4 Deselect command or before proceeding with another application (SelectApplication command).

As logical counter-part of the CommitTransaction command the AbortTransaction command allows to invalidate changes on files with integrated backup management, see chapter 4.6.11.

Functional Specification

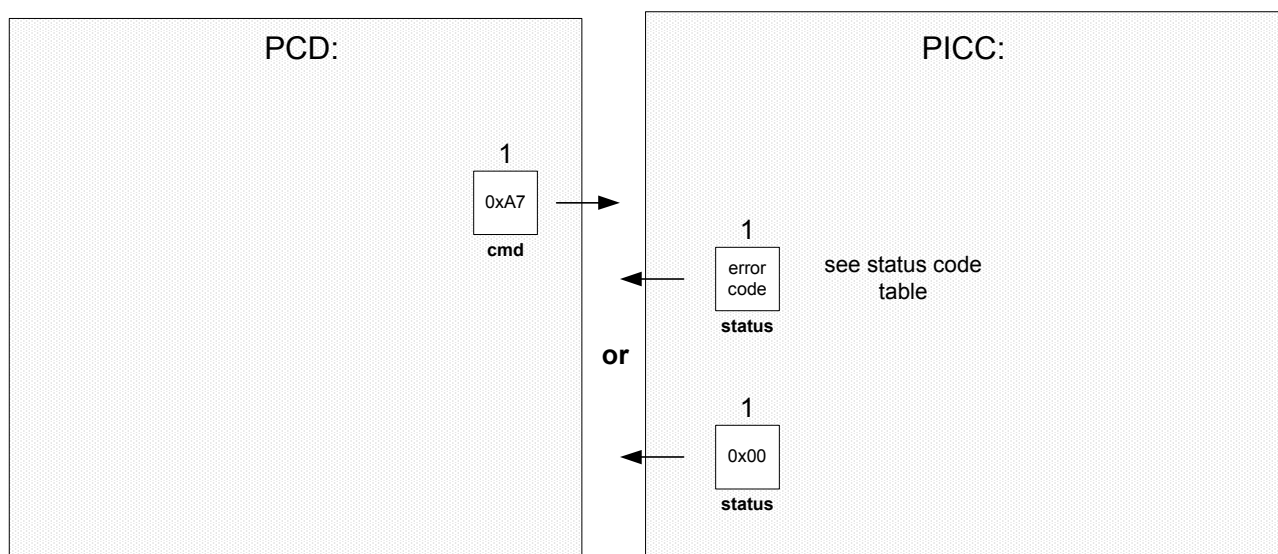
mifare DESFire MF3 IC D40

4.6.11 AbortTransaction

The AbortTransaction command allows to **invalidate** all previous write access on Backup Data Files, Value Files and Record Files within one application.

This is useful to cancel a transaction without the need for re-authentication to the PICC, which would lead to the same functionality.

AbortTransaction(AID) [1 bytes]



This command takes no parameter.

The AbortTransaction command **invalidates** all write access to files with integrated backup mechanisms without changing the authentication status:

- Backup Data Files
- Value Files
- Linear Record Files
- Cyclic Record Files

Functional Specification**mifare DESFire MF3 IC D40****5 DEFINITIONS**

Data sheet status	
Objective specification	This data sheet contains target or goal specifications for product development.
Preliminary specification	This data sheet contains preliminary data; supplementary data may be published later.
Product specification	This data sheet contains final product specifications.
Limiting values	
Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics section of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.	
Application information	
Where application information is given, it is advisory and does not form part of the specification.	

6 LIFE SUPPORT APPLICATIONS

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips customers using or selling these products for use in such applications do so on their own risk and agree to fully indemnify Philips for any damages resulting from such improper use or sale.

Functional Specification**mifare DESFire MF3 IC D40**

7 REVISION HISTORY**Table 1** Objective Specification MF3 IC D40 Revision History

REVISION	DATE	CPCN	PAGE	DESCRIPTION
1.04	January 2003			General Update and Re-wording
1.0	July 2002			First official version.

Functional Specification

mifare DESFire MF3 IC D40

NOTE

Philips Semiconductors - a worldwide company

Contact Information

For additional information please visit <http://www.semiconductors.philips.com>. Fax: +31 40 27 24825

For sales offices addresses send e-mail to: sales.addresses@www.semiconductors.philips.com.

© Koninklijke Philips Electronics N.V. 2002

SCA74

All rights are reserved. Reproduction in whole or in part is prohibited without the prior written consent of the copyright owner.

The information presented in this document does not form part of any quotation or contract, is believed to be accurate and reliable and may be changed without any notice. No liability will be accepted by the publisher for any consequence of its use. Publication thereof does not convey nor imply any license under patent- or other industrial or intellectual property rights.

Let's make things better.

**Philips
Semiconductors**



PHILIPS