AN11340

MIFARE Ultralight EV1 features and hints

Rev. 3.2 — 5 May 2021
073132

Application note
COMPANY PUBLIC

Document information

Information	Content
Keywords	Multiple ticketing, secured data storage, implementation hints
Abstract	This document presents features and hints for a secured and optimized application development using MIFARE Ultralight EV1 cards.

Revision history

Revision history

Rev	Date	Description
3.2	20210505	• General update, remove of MIFARE Ultralight • The format of this application note has been redesigned to comply with the new identity guidelines of NXP Semiconductors.
3.1	20180709	Editorial update
3.0	20130227	Updated examples from DES to AES and added MIFARE Ultralight EV1 updated section 2.1.2 Transaction Speed, added section 2.1.2.1 FAST_READ Time Saving, added section 4 MIFARE Ultralight EV1 Counters, added section 5 MIFARE Ultralight EV1 Password and PACK, added section 6 MIFARE Ultralight EV1 Anti-cloning based on Originality Check, added section 7 MIFARE Ultralight EV1 Tearing Application Implementation
2.1	20070826	Example has been corrected
2.0	20061019	Security features (section 2.2) and example (annex 6.3) added
1.0	20020515	First release

1 Introduction

1.1 Purpose and scope

This application note is intended to describe the features of the MIFARE Ultralight EV1 MF0ULx1 (see [MF0ULx1]), and the comparison to the MIFARE Classic product family.

This application note addresses some security mechanisms which may be used to protect the data stored in the card and demonstrates the implementation of MIFARE Ultralight EV1 into an existing MIFARE Classic application and shows the necessary modifications over the existing environment for NFC readers for MIFARE ICs. These modifications apply to all relevant NXP Semiconductors NFC readers for MIFARE ICs.

Although MIFARE Ultralight EV1 can in some low security applications be used as an alternative for MIFARE Classic, please consider other products like MIFARE Plus EV2 or MIFARE DESFire EV3 for high secure applications.

1.2 Disclaimer

Note that whenever terms are used like locking, read-only, fraud protection, security feature and the like, this does not imply that there would never be any attack possible to circumvent the feature.

MIFARE Ultralight EV1 is not a security certified product. Depending on the value of the assets that need to be protected, one may consider using Common Criteria certified products with security features that have been demonstrated to resist certain attack potential during certification. (E.g. MIFARE Plus and MIFARE DESFire have CC high attack potential profile and MIFARE DESFire Light has CC enhanced-basic attack potential profile).

1.3 How to use this document

This document contains a collection of hints and features that could be of interest for users, who plan to use the MIFARE Ultralight EV1.

None of this information is intended to replace any of the relevant data sheets or design guides.

All the numerical examples are just examples, describing the usage of commands and providing reference values to verify any implementation.

Any data, value, cryptogram are expressed here as hex string format if not other mentioned.

In this document the term „MIFARE Classic card“ refers to a MIFARE Classic IC-based contactless card, the term „MIFARE Ultralight card“ refers to a MIFARE Ultralight IC-based contactless card.

1.4 Reference documents

[MF0ULx1]	MIFARE Ultralight EV1, Contactless Single-trip Ticket IC MF0ULx1, Product data sheet, Doc. No: 2345**[1]
[AN10922]	AN10922 Symmetric Key Diversifications, Doc. No: 1653**
[AN11093]	AN11093 Card Coil Design Guide, Doc. No.: 0117**

[SI070010]	Temperature Management, Inlet Design
[MFRC530]	MFRC530; Reader solution for MIFARE products, Doc. No. 0574**
[MFRC531]	MFRC531; Standard ISO/IEC 14443 A/B reader solution
[CLRC632]	CLRC632; Standard multiprotocol reader solution.
[AN11341]	AN11341 MIFARE Ultralight EV1 Originality Signature Validation, Doc. No: 2591**
[ISO/IEC 14443-3]	ISO/IEC 14443-3 Identification cards — Contactless integrated circuit cards — Proximity cards - Part 3: Initialization and anticollision
[NIST SP800-38A]	National Institute of Standards and Technology (NIST). NIST Special Publication 800-38A: Recommendation for Block Cipher Modes of Operation. December 2001
[NIST SP800-38B]	National Institute of Standards and Technology (NIST). NIST Special Publication 800-38B: Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. May 2005
[ISO/IEC 9797-1]	Information technology Security techniques Message Authentication Codes.

[1] ** ... document version number

2 MIFARE Ultralight application hints

2.1 Memory features

In addition to the user memory area the MIFARE Ultralight EV1 offers the features of an OTP[1] area and lock bytes to lock the OTP and user area. The usages of LOCK bits are described in [MF0ULx1].

A MIFARE Ultralight EV1 dedicated 4-byte WRITE-command provides a high transaction speed. All the add-on features are dedicated to support special application functionality, e.g., ticketing.

2.1.1 Memory organization

The EEPROM memory is organized in pages with 4 bytes per page. The memory organization can be seen in Figure 1 and Figure 2 below, the functionality of the different memory sections is described in the following sections.

Page 03h is the OTP page and the default value of the OTP bytes is 00 00 00 00h. These bytes can be bit-wise modified using the WRITE or COMPATIBILITY_WRITE command.

Page 02h contains the Lock bytes 0 and 1 which represent the field programmable read-only locking mechanism.

Page Adr		Byte number within a page				
Dec	Hex	0	1	2	3	Description
0	0h	serial number				Manufacturer data and lock bytes
1	1h	serial number				
2	2h	serial number	internal	lock bytes		
3	3h	OTP	OTP	OTP	OTP	One Time Programmable
4	4h	user memory				User memory pages
5	5h					
...	...					
14	Eh					
15	Fh					
16	10h	CFG0				Configuration pages
17	11h	CFG1				
18	12h	PWD				
19	13h	PACK		RFUI		
		One-Way counters 1)				Counter pages

aaa-006275

Figure 1. Memory organization MF0UL1x

1 One Time Programming

Page Adr		Byte number within a page				
Dec	Hex	0	1	2	3	Description
0	0h	serial number				Manufacturer data and lock bytes
1	1h	serial number				
2	2h	serial number	internal	lock bytes		
3	3h	OTP	OTP	OTP	OTP	One Time Programmable
4	4h	user memory				User memory pages
5	5h					
...	...					
34	22h					
35	23h					
36	24h	lock bytes			RFUI	Lock bytes
37	25h	CFG0				Configuration pages
38	26h	CFG1				
39	27h	PWD				
40	28h	PACK		RFUI		
		one-way counters 1)				Counter pages

aaa-006276

Figure 2. Memory organization MF0UL2x

2.1.2 Using OTP memory for multiple ticketing

The MIFARE Ultralight EV1 offers the possibility to use the OTP bytes in page 03h as counter. All bits of the OTP bytes are pre-set to “0” at the delivery. These bits can be set one time to “1”. This gives the possibility to interpret them as a counter e.g. in a public transport use case to count the number of trips. Therefore, the number of “1” in OTP area of page 03h can be considered as counter value, beside the 3x independent 24-bit counters on MIFARE Ultralight EV1 (see Section 4). Meaning, the OTP bytes of page 03h offer a number of 32 states that could be used to allow a certain number of passings through a turnstile.

2.1.2.1 Recommended implementation of One-Time Programmable bits as counter

There are different ways the One-Time Programmable bits can be used as counter, but there is one recommended way to implement the counter to identify unintended OTP values. The defined start value of the counter in the OTP area, page 03h is recommended to define at a high value during personalization. That can be applied by setting the counter value to the highest possible value with the remaining trips allowed trips allowed in the system. e.g. set the page 03h of the OTP area to 0xFF FF FF F0. Now, the OTP is pre-set for 4 trips by increase of the value from 0xFF FF FF F0 to 0xFF FF FF FF (max). To disable the counter at the max value of 0xFF FF FF FF, the WRITE or COMPATIBILITY_WRITE command must be applied twice to set the value in both the valid page and the internal backup page to the maximum. (see also Section 7.1)

2.1.2.2 Example of the OTP bytes as counter

An example of a 4 trips ticket is shown in Figure 3. For this application, a ticket issuing the OTP memory of the MIFARE Ultralight has to be pre-set to 0xFFFFFFF0.

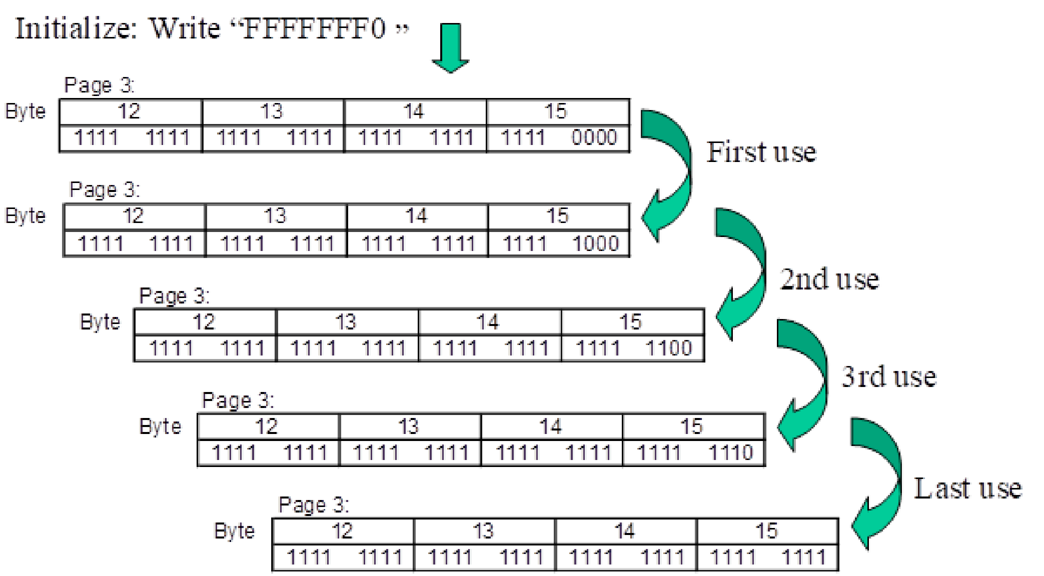


Figure 3. Example of a four rides ticket

For every access, the 4-byte OTP (page 3) has to be checked and updated (as shown in Figure 4 to ensure the validity of the tickets. The structure of programmed “1” shall be either from left to right or vice versa, to enable the possibility to identify non-compliant counter values. A ticket with OTP value not following this structure, must be rejected.

Additionally, to reduce the chance of fraud on the OTP bytes, it is recommended to support a data integrity protection of the OTP bytes by a MAC calculated outside the ticket that is stored in the password protected user memory on the ticket to allow the possibility to identify at a read, if there is an unintended change in the OTP bytes. A MAC calculation of data including the UID introduces an additional level of data integrity checks against unintended manipulations. (See Section 2.2)

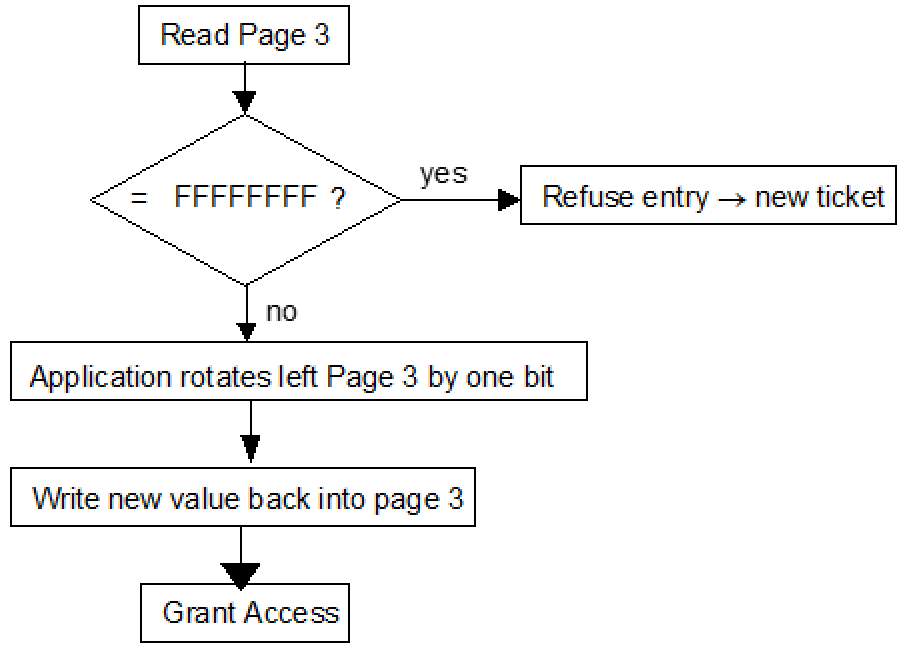


Figure 4. Ticket Counter Command flow using left rotation of the value

The example in Figure 4 implements a counter that will be filled "from the right side". The current value in page 3 is read and checked, if the counter is not full (not all bits set, e.g. value is not equal to 0xFFFFFFFF). If there are still unset bits, the value is rotated left by one bit and written back to page 3. This will set the next bit left to the current set bits, regardless of the current value[2]. All other bits are kept untouched (existing ones stay ones, as they cannot be reprogrammed, zeros stay zeros). This corresponds to a logic OR of the previous and the rotated value.

Table 1. Left rotation example explained

Step	Comment	Value hex	Value bin
1	Value read from MIFARE Ultralight EV1 page 3	0x0000007F	00000000000000000000000001111111
2	Rotate left by one bit	0x000000FE	00000000000000000000000011111110
3	After writing value of step 2 to page 3 of the MIFARE Ultralight EV1, one additional bit will be set	0x000000FF	00000000000000000000000011111111

When a maximum of 16 rides need to be possible on one ticket, then some extra protection is possible by introducing some redundancy as is shown in the next example. For a ten times counter, the OTP would be to set to FC00FC00. For each count, two bits should be set (e.g. after first count, the counter value should be FD00FD00, corresponding to the first examples, but with right-rotation, i.e. filling from the left), ideally in one write command (this ensures that a card does not become invalid if a genuine tear-off event happens). Both, the upper and lower half of the 4 bytes always need to be equal. By rejecting OTP values that do not adhere to the expected structure, an additional layer of integrity protection is offered.

2 With the initial value “00000000” for 32 times counter, the rotate left at the very first access check after selling the ticket has to be exchanged into another initial WRITE command in order to get the first "1" programmed

2.1.3 Lock bytes

To lock the memory area pages of MIFARE Ultralight EV1 starting at page address 10h onwards is supported by the lock bytes 2 and 3. The granularity of the number of pages locked by the bits depends on the memory area size.

Each page from 03h (OTP) to 0Fh can be individually locked by setting the corresponding locking bit Lx to logic 1 to prevent further write access. After locking, the corresponding page becomes read-only memory. Additionally, the block-lock bits in page 2 byte 2 (lock byte 0) lock the configuration of the lock bits.

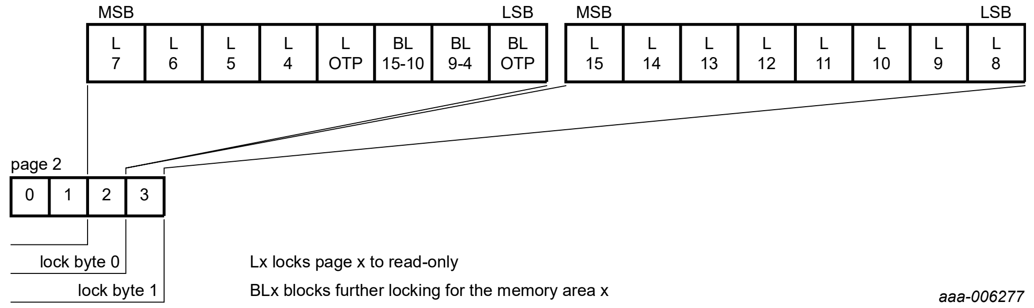


Figure 5. Lock bytes page 2

For MF0UL2x types, lock bytes 2 to 4 are located in page 36d. The granularity here is 2 pages each, covering a total of 20 pages (16d - 35d). The block-locking bits are the 5 LSBs in lock byte 4.

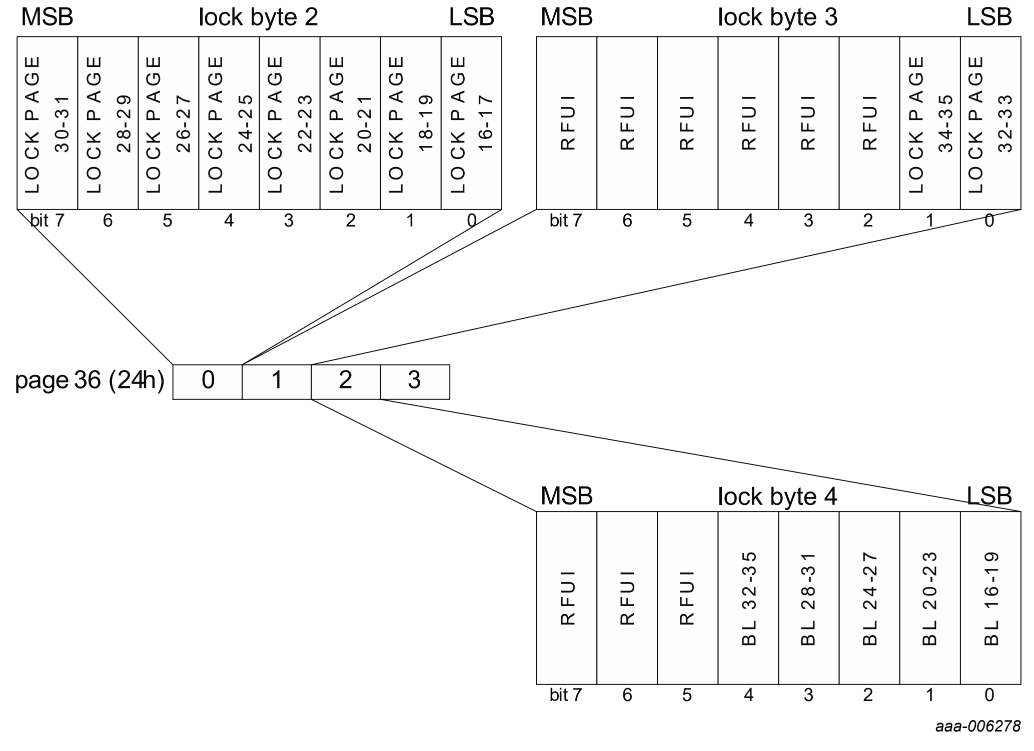


Figure 6. Lock bytes 2-4 (only MF0UL2x)

At personalization, after configuration of the memory area, it is recommended to set all block locking bits.

Note: It is recommended that all block-locking bits are set with a double WRITE or COMPATIBILITY_WRITE command to freeze the configuration of page 03h (OTP) and the memory area. (see Section 7.1)

In case the use case does not allow for all block-locking bits to be set, it is recommended to implement an integrity protection as an additional defense to allow detection of manipulation, e.g. based on a MAC on the data stored on locked pages (for example, as suggested in Section 2.2.1)

2.1.4 Transaction speed

Although the MIFARE Ultralight offers the 16-byte Compatibility Write command to be compatible with the MIFARE Classic environment, the use of the 4 byte Write command is recommended, if the application requires a fast transaction. The Write saves approximately 20% of the transaction time[3] compared to the Compatibility Write, as can be seen in Table 2 and Figure 7.

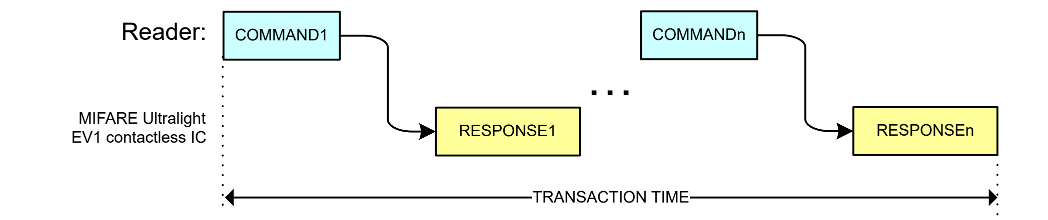


Figure 7. Transaction time

Table 2. Transaction time

Command	4 pages		12 pages	
REQA	0.4 ms		0.4 ms	
Anticollision level 1 + 2 & Select	3.7 ms		3.7 ms	
Read	2.1 ms		6.3 ms	
Write	19.7 ms		59.2 ms	
Compatibility Write		25.4 ms		76.4 ms
Halt	0.5 ms		0.5 ms	
Transaction Time (approximately)	26.4 ms	32.1 ms	70.1 ms	87.3 ms

The transaction time[2] as shown in Table 2 and Figure 7 counts the time needed for

- the communication from the reader to the MIFARE Ultralight,
- the response time of the MIFARE Ultralight and
- the communication from the MIFARE Ultralight back to the reader.

2.1.4.1 FAST_READ time saving

MIFARE Ultralight EV1 (See [MF0ULx1]) introduces the FAST_READ command. The FAST_READ command has a variable frame length depending on the start and end

3 This does not include the time requires the host computer for the application itself (e.g. calculating and checking ticket values, displaying results, opening gates, etc.).

address parameters. The maximum frame length supported by the PCD needs to be taken into account when issuing this command.

The table below shows the comparison in terms of timing between READ and FAST_READ. The FAST_READ command is able to speed up the reading compared with the READ command in case of amount of data that is smaller than 4 pages but also bigger than 4 pages. Only in case of 4 pages reading the READ command is faster than the FAST_READ.

Table 3. READ and FAST_READ timing comparison

Command	1 page	4 pages	12 pages	32 pages
READ	2.1 ms	2.1 ms	6.3 ms	16.8 ms
FAST_READ	1.2 ms	3.7 ms	3.7 ms	11.8 ms
FAST_READ Time Saving	+43%	-6%	+21%	+30%

2.2 Proposed security mechanism

MIFARE Ultralight EV1 has been designed to support the faster application with the cheapest solution. Therefore, it does not address any security feature except:

- the Unique IDentity (UID),
- the Password (see [MF0ULx1])
- the Originality Signature Validation based on ECDSA (see [AN11341])

From the application point of view this means, no cryptographic challenge-response based authentication has to be performed and no key is needed, therefore only limited security is offered.

If required, MIFARE Ultralight EV1 can be integrated in an application with additional security by using additional cryptographic protection at the system level. The following two subsections demonstrate how additional integrity protection (see Section 2.2.1) and if needed also confidentiality protection (see Section 2.2.2) of the stored data can be achieved.

MIFARE SAM AV3 (secure access module) can be used to store the required key(s) and execute the cryptographic calculations. This SAM module facilitates the system in the following ways:

- The key is stored securely, without being able to be read out
- The module provides functions for calculation of MAC and encryption (including key diversification if required).
- The cryptographic operations are fast enough for real-time operations.

2.2.1 Integrity of stored data

The content of the MIFARE Ultralight EV1 memory lacks guaranteed integrity. To avoid this inconvenience, we propose a MAC to be calculated and appended to the protected data. For this purpose, a CMAC (Cipher-based Message Authentication Code) according to the [NIST SP800-38B] may be a good choice. The complete scheme is shown in Figure 8.

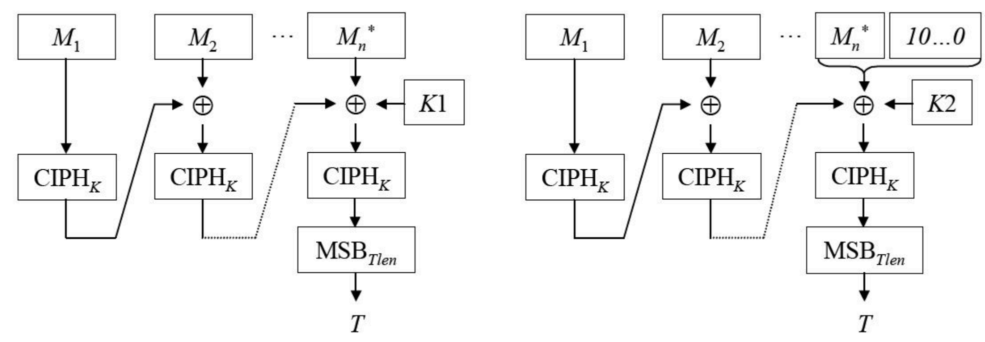


Figure 8. CMAC calculation according to NIST SP800-38. Left side shows input data as a multiple of 16 bytes, right shows padded input data

- The recommended cipher (CIPH) is AES-128.
- Use a secret key (K), which is only known by the reader infrastructure and/or backend.
- The input (M_1 … M_n) is the data to be protected concatenated with the UID, e.g. UID || Data.
- Input data blocks (M_n) need to have a size of 16 bytes. If the number of input data bytes is not a multiple of 16, padding is added acc. NIST SP800-38B (80h following by 00s). This will typically be the case with the UID being 7 bytes and a data page being 4 pages.
- The result of the CMAC calculation is a block of 16 bytes length, which can be truncated to a shorter size as well. Refer to [NIST SP800-38B] for recommendation on the truncation size, but in general, a size below 8 bytes is not recommended for general applications.

By storing the (truncated) CMAC together with the protected user data in the MIFARE Ultralight EV1 user memory, the data is protected against manipulation, as long as the key is kept secret. Including the UID in the MAC calculation, ensures a different MAC is required for each card, even if the protected data is the same. This supports detecting that data has been copied from one MIFARE Ultralight EV1 to another. Alternatively, the UID can also be included via a key diversification step, as outlined in Section 2.2.2.

Note that this method does not protect against recording the combination of old content and a valid MAC, and writing it back to the card at a later point of time (i.e. a so-called replay attack). Additional measures can be taken by e.g. including a monotonically increasing counter in the user data and maintaining and checking this in the reader and/ or back-end infrastructure. There may also remain residual risks of integrity protected data being copied to a clone or emulator. If more high-level security features, like card-integrated cryptographic support are required, other members of the MIFARE card IC family can be used in the application, e.g. the MIFARE Plus EV2 or the MIFARE DESFire EV3.

2.2.2 Confidentiality of stored data

If the MIFARE Ultralight EV1 pages can be read without any authentication, anyone can read the pages using any standard reader. But if the stored data is encrypted with a secured key then these are just some bytes to one who does not have the secret key and information regarding the encryption method. Therefore, by storing the encrypted data in MIFARE Ultralight EV1 memory, one can add confidentiality to the data itself.

Note that the password verification method available in the MIFARE Ultralight EV1, does not offer a high security protection. It is an easy and convenient way to prevent unauthorized memory access. However, be aware the even if applied, the data is still exchanged in plain. If a higher level of protection is required, cryptographic methods on application layer can be used to increase the overall system security.

In general, encryption does not provide integrity protection. Therefore, it is recommended to combine encryption still with a MAC, to also avoid manipulation of the data as also discussed in Section 2.2.1. The recommended cipher is AES-128. This leads to the following function, composed of the steps described below.

$$Data_{stored} = f\,(Mk, data_{origin}, UID)$$

A 16-byte Master Key (Mk) has to be defined by the application provider. For each card, two card keys are derived from the Master Key (Mk):

- Ck_{MAC} for MACing
- CK_{ENC} for encryption

This can be done using the key diversification of [AN10922] including the UID. Including the UID in the key diversification is another method of ensuring unique MACs for each different card (even if the protected data is the same), compared to appending the UID to the input data as described in Section 2.2.1. For the encryption, this also ensures different ciphertext is generated for different cards, i.e. not disclosing potentially the same plaintext data is stored. Note that including the UID in the encryption, in a similar way as done for the integrity protection method of the previous section, would result in a bigger ciphertext, consuming more storage space on the card. Therefore, key diversification is proposed here.

The steps to be followed for the key diversification are indicated in [AN10922] section 2.2 “AES-128 key” where the inputs to the 128-bit AES key diversification are:

- M: the concatenation of a constant with the 7 bytes UID, i.e. respectively:
  – "$CONST_{MAC}$ || UID" for Ck_{MAC}
  – "$CONST_{ENC}$ || UID" for Ck_{ENC}
- K, the 16 bytes AES-128 Master Key (Mk)

And the output is:

- Diversification Key, respectively the 16 bytes AES-128 bits Ck_{MAC} or Ck_{ENC}

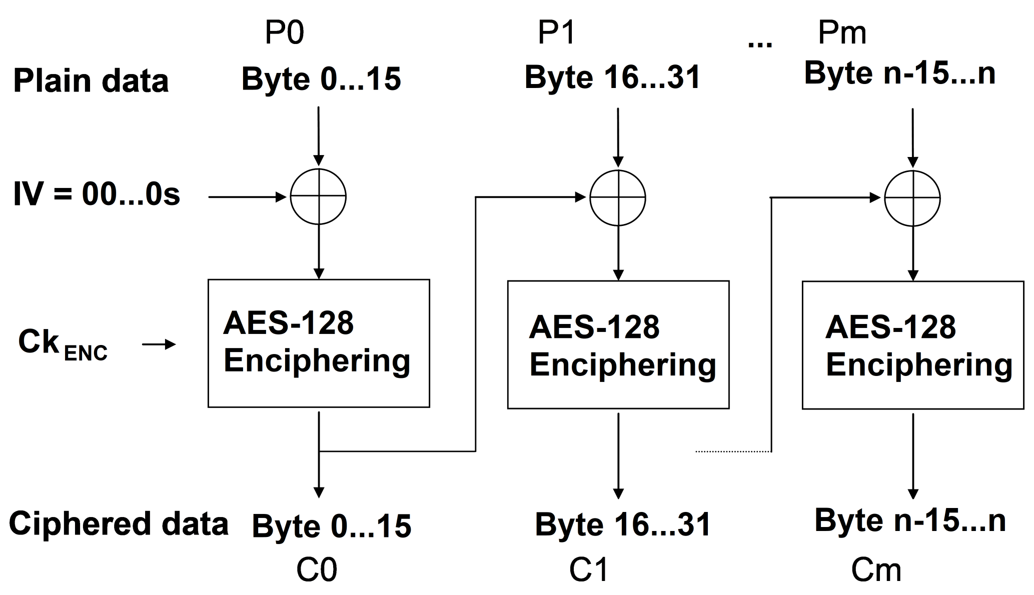
After this step, the plain data is encrypted using the encryption Card Key (Ck_{ENC}) in CBC mode according [NIST SP800-38A].

- Use 16 bytes initial vector (IV) of all ‘00’s, IV= “00...00” (also a random IV can be used. This has the advantage that identical plaintexts would result in a different ciphertext. The drawback is, that the IV needs to be then stored on the ticket as well)
- As AES 128 works with 16-byte block wise, organize the data in multiple of 16 by adding one of the padding schemes of [ISO/IEC 9797-1], e.g. Padding Method 1 which results in padding with all zeros ‘00’. As example (‘xx’ is the data bytes):

```
10 padding bytes: xxxxxxxx xxxx0000 00000000 00000000

15 padding bytes: xx000000 00000000 00000000 00000000
```

The complete CBC encryption scheme is shown in the following figures (Figure 9 for encryption and Figure 10 for decryption):



m = (n+1)/16 -1; where m and n are integer
P = plain 16-byte block
C = ciphered 16-byte block

Figure 9. Data encryption scheme

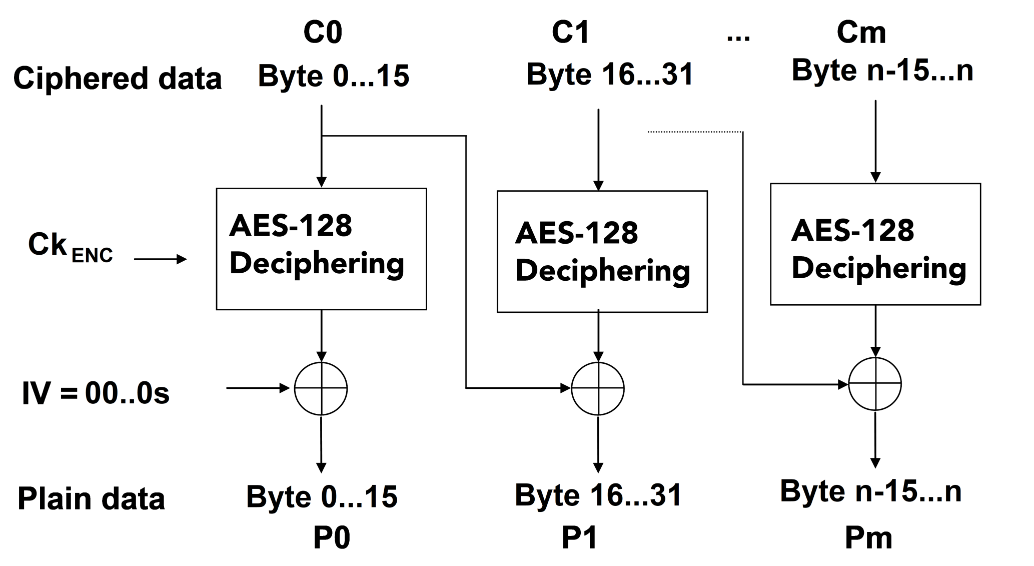


Figure 10. Data decryption scheme

As a final step, a MAC is calculated over the ciphertext. This can be done by applying the CMAC algorithm according [NIST SP800-38B], similar as also used in the previous section, see Figure 8. In this case, Ck_{MAC} is to be applied as the key K, and the ciphertext "C0 … Cn" is the input message M.

Both the ciphertext and the calculated (and eventually truncated) MAC are to be stored on the card.

3 Using MIFARE Ultralight EV1 in an existing MIFARE Classic application

The MIFARE Ultralight EV1 offers the feature to be used in an existing MIFARE Classic application. Therefore, the MIFARE Ultralight EV1 command structure is compatible to the MIFARE Classic one.

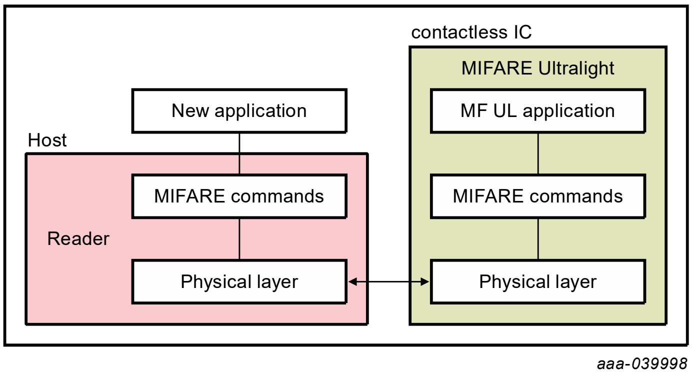


Figure 11. MIFARE Ultralight in an existing MIFARE Classic application

Because the MIFARE Ultralight EV1 addresses a different application category (single trip ticketing, fast and cheap transactions), some application changes have to be done anyway, but the existing MIFARE transaction command structure and the NFC reader for MIFARE ICs may be used with the MIFARE Ultralight (as shown in Figure 11).

3.1 Differences: MIFARE Classic - MIFARE Ultralight EV1

The basic differences between MIFARE Classic and MIFARE Ultralight EV1 are:

1. The MIFARE Ultralight EV1 uses only **7-byte UIDs** instead of possible 4-byte non-unique ID for MIFARE Classic EV1. There are 2 possibilities to select a MIFARE Ultralight EV1 card. It suggests using the anti-collision cascade level 2 (as specified in the ISO14443A - 3) to get the complete UID and select one MIFARE Ultralight EV1 (see Section 3.2.1).
   If the reader does not support the ISO cascade level 2 anti-collision and there is no chance to update the reader to do so, a combination of anti-collision cascade level 1 (using the first 3 bytes of the UID) and a read of block 0 after the Select can be used instead. Please note that this workaround has the limitation that there is no chance to fully RESOLVE a collision between two cards in case of the unlikely event, that the first part of the UID is equal. The collision can only be DETECTED, allowing the reader to inform the user to present only one card to the reader (see Section 3.2.2).
2. The MIFARE Ultralight does not need the authentication and no keys, as it uses **no encryption**.
   Note that MIFARE Ultralight EV1 has the password authentication feature anyhow different from MIFARE Classic authentication. Performing a MIFARE Classic authentication generates an error, and the MIFARE Ultralight EV1 goes back to the Idle (or Halt) state.

3. The MIFARE Ultralight only uses **"Read", "Write" and "C.Write"** (MIFARE Classic compatible write command with 16 Bytes).
   Note that MIFARE Ultralight EV1 is backward compatible with MIFARE Ultralight supporting an additional command set.
   No Value-commands are used.

3.2 Transaction command flows

3.2.1 Transaction flow using Cascade Level 2

The anti-collision cascade level 1 and 2 (as described in the ISO14443-A part 3) should be used to select a MIFARE Ultralight. This command sequence gives back the complete 7-byte UID of the MIFARE Ultralight EV1, and allows selecting only one card (as shown in Figure 12).

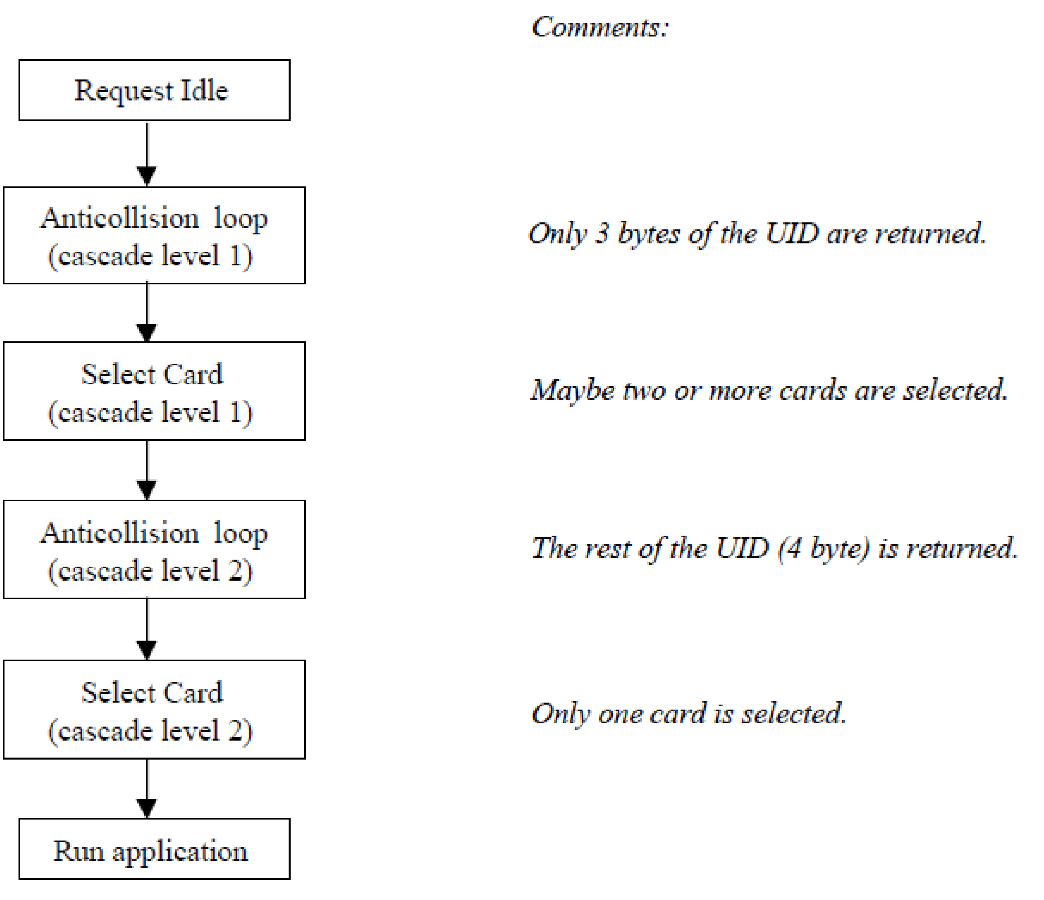


Figure 12. Transaction flow with Cascade Level 2

3.2.2 Transaction flow using Cascade Level 1 and Read Block 0

If the reader does not support the anti-collision cascade level 2, only the anti-collision cascade level 1 (ISO14443A-3) can be used to select a MIFARE Ultralight EV1. This is the "Classic Anti-collision and it returns 3 significant bytes of the UID. In this case, the complete UID shall be checked after selection with a read of block 0 to make sure, that only one card is selected. **If a collision is detected** during that read of block 0, the user has to be informed, that **only one card** has to be presented to the reader (see Figure 13).

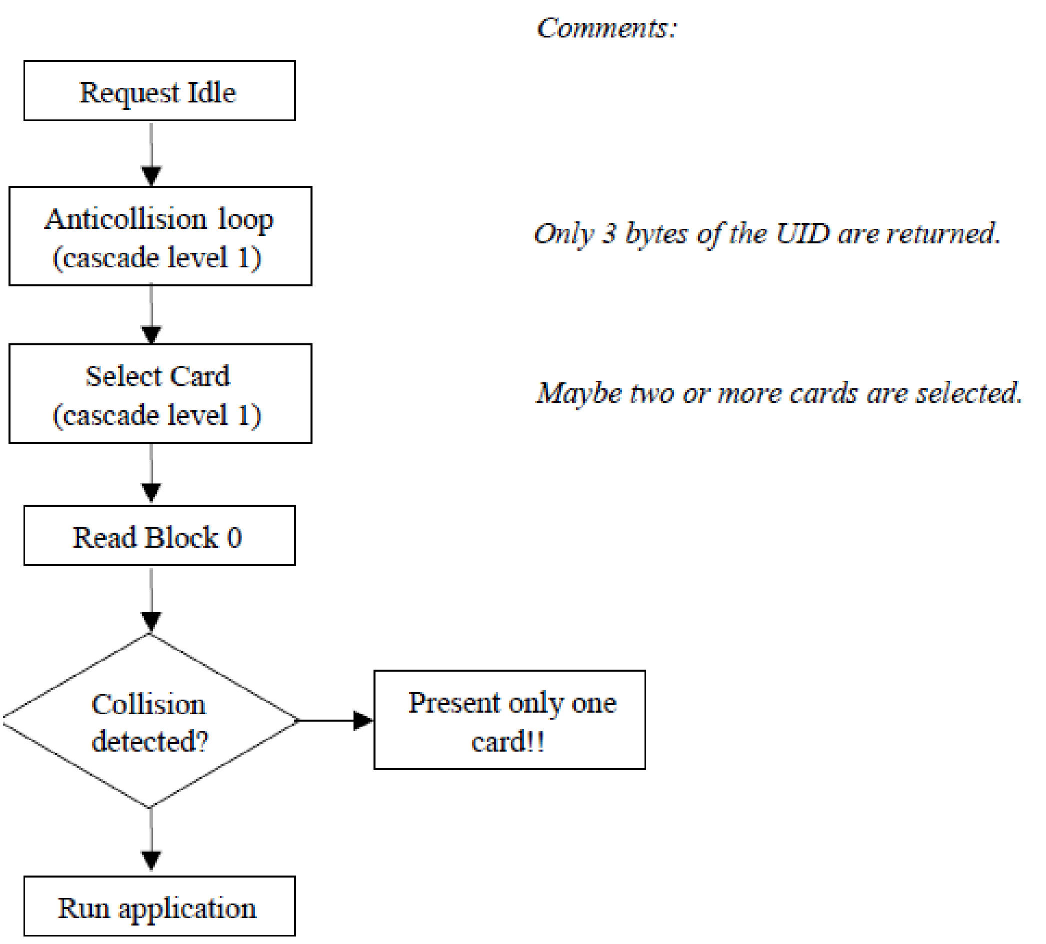


Figure 13. Transaction flow with Cascade Level 1 and Read Block 0

Remark:

This command flow as given in Section 3.2.2 does not follow the ISO14443, and only offers a compatible command flow to work with some old reader environment. If possible, the use of the complete anti-collision cascade level 1 and 2 is recommended.

3.3 MIFARE Ultralight and NFC readers for MIFARE ICs

The MIFARE Ultralight can be selected, read, and written by every NFC reader for MIFARE Classic.

The MIFARE Classic authentication has to be skipped, and the selection of the MIFARE Ultralight has to be done as shown either in Figure 8 or Figure 9.

A MIFARE Classic Read command can be used. In this case only the first 4 Bytes contain valid data according to the addressed page; the other 12 bytes refer to the next 3 pages (see the related data sheet of the MIFARE Ultralight).

To write data into the memory of the MIFARE Ultralight, either the (4-byte) WRITE or the COMPATIBILITY WRITE can be used (see the related data sheet of the MIFARE Ultralight).

Reader Modules:

Table 4.

Reader	Anti-collision	WRITE	Comment
MF CM200	cascade level 2 possible, but LLL[1] has to be adapted[2]	possible, but LLL has to be adapted	supports MIFARE Ultralight
MF CM500	cascade level 2 possible, but LLL has to be adapted[2]	possible, but LLL has to be adapted	supports MIFARE Ultralight

[1] **L**ow **L**evel **L**ibrary
[2] example see 9.2

Reader Devices:

Table 5.

Reader	Anti-collision	WRITE	Comment
MF RD260	only cascade level 1, no firmware update or extension possible	only COMPATIBILITY WRITE, no firmware update or extension possible	supports MIFARE Ultralight only in compatibility mode
MF RD560	only cascade level 1, no firmware update or extension possible	only COMPATIBILITY WRITE, no firmware update or extension possible	supports MIFARE Ultralight only in compatibility mode

Reader ICs:

Table 6.

Reader	Anti-collision	WRITE	Comment
MFRC171	full cascade level 2 possible, but LLL has to be adapted	possible, but LLL has to be adapted	supports MIFARE Ultralight[1]
MFRC500	BFL[2] contains the full cascade level 2 support	BFL contains the full 4 byte WRITE support	supports MIFARE Ultralight
MFRC530	BFL contains the full cascade level 2 support	BFL contains the full 4 byte WRITE support	supports MIFARE Ultralight
MFRC531	BFL contains the full cascade level 2 support	BFL contains the full 4 byte WRITE support	supports MIFARE Ultralight
CLRC632	BFL contains the full cascade level 2 support	BFL contains the full 4 byte WRITE support	supports MIFARE Ultralight
MFRC522	BFL contains the full cascade level 2 support	BFL contains the full 4 byte WRITE support	supports MIFARE Ultralight
MFRC523	BFL contains the full cascade level 2 support	BFL contains the full 4 byte WRITE support	supports MIFARE Ultralight
CLRC663	NFC Reader Library contains the full cascade level 2 support	NFC Reader Library contains the full 4 byte WRITE support	supports MIFARE Ultralight

[1] example see 9.1
[2] BFL means Basic Function Library

4 MIFARE Ultralight EV1 counters

The MIFARE Ultralight EV1 features three independent 24-bit one-way counters. The counters are initialized to 000000h at delivery. Each counter can be read using the READ_CNT command and increased using the INCR_CNT command. The INCR_CNT can work with values between zero and FFFFFFh, boundaries included. Note that the value of zero can be used as well, although it will not increment the counter in practice.

An example is indicated in Figure 14.

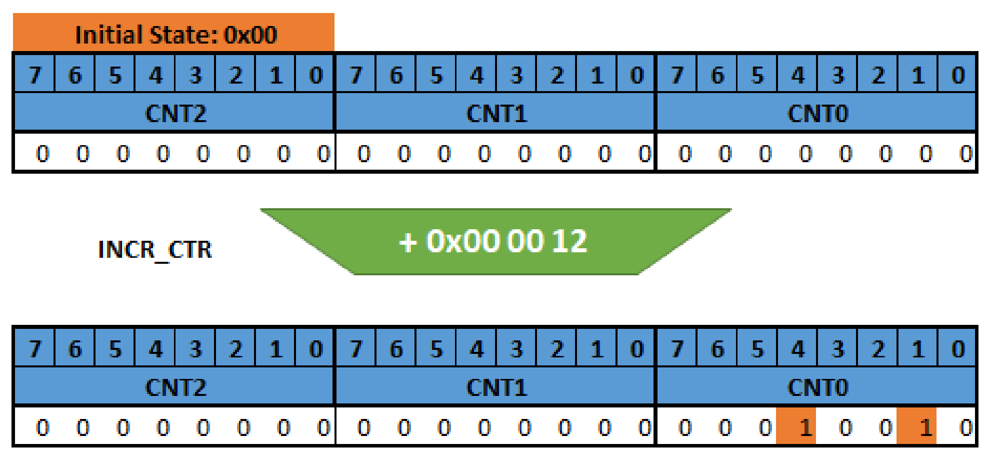


Figure 14. Counter increased of 18 (12h)

MIFARE Ultralight EV1 counters come with anti-tearing support to avoid unintended values caused by a tear-off from the reader during a transaction. Following steps are recommended on the counter for e.g. ticketing purposes:

1. Read the counter using READ_CNT in order to check the current counter value
2. Increase the counter using INCR_CNT:
   a. If the INCR_CNT response is equal to NAK5/7, this is a tearing event, repeat steps 1) to 3),
   b. If the INCR_CNT response is equal to NAK6, the counter is corrupted or unusable, invalidate the ticket, or
   c. In case of timeout, repeat steps 1) to 3),
   d. If the INCR_CNT response is equal to ACK, execute step 3)
3. Read again the counter using READ_CNT check that the new expected counter value has been correctly stored.
   a. If the value is not correctly stored, repeat steps 1) to 3) up to N times
   b. If after N times the value is still not correct, invalidate the ticket

How a ticket is invalidated depends on the underlying system, e.g. it could be done by writing to the OTP area, setting counters to the maximum value or writing something into the user memory.

4.1 Recommended 24-bit one-way counter implementation including countermeasures

To decrease the possibility for data manipulation in applications and to allow for the detection of manipulated counter values on the three independent anti-tearing supported 24-bit one-way counters following implementations and countermeasures shall be considered:

- Define the start value of the three independent 24-bit one-way counters at the highest possible counter value during personalization, by considering the required counts in application. That can be applied, by setting the counter value to the maximum 0xFF FF FF minus the number of counts allowed in the system, e.g. set the counter value to 0xFF FF F5. The counter value of 0xFF FF F5 allows 10 trips by increase of the value to 0xFF FF FF (max). (see Figure 15)

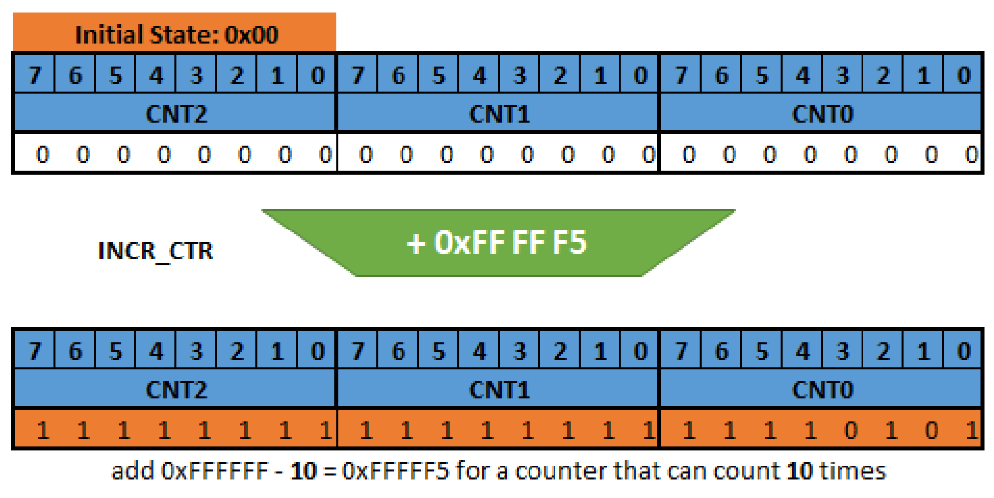


Figure 15. Example of a max. 1000 times counter

- Once the counter is at the maximum value (0xFFFFFF), it is recommended to disable it by issuing an additional INCR_CNT by zero to ensure the maximum value is programmed in both pages. Also if the maximum value in an application is not 0xFFFFFF, it is recommended to still disable a counter by incrementing to 0xFFFFFF followed by an additional INCR_CNT by zero.
- Implementation of a data integrity protection on the counter value. The MAC which has been calculated outside the ticket is stored in the user memory on the ticket to give the possibility to detect if there is a malicious change on the counter value.
- Tickets with a low start counter value e.g. 0x 00 00 00 shall be, if possible, migrated to a high start value e.g. 0x FF FF F5 as recommended in the first bullet point. It must be made sure that the system can recognize both variants for the transition phase.
- It is recommended to implement countermeasures outside the ticket to support a backend fraud detection mechanism in the infrastructure. It shall give the possibility to detect, based on the UID of MIFARE Ultralight EV1, if the counter value linked to the ticket (UID) is as expected. If not, the card with this UID shall be immediately rejected in case of online detection. If only offline detection is possible, the card shall be deny-listed for further operations once an inconsistency has been observed.

See also Section 7.1

4.2 Counter for zone/mileage based scheme

The MIFARE Ultralight EV1 counter can also be used in a zone- or mileage-based tariff scheme in a check-in – check-out system. In this case, a single counter is split into 2 parts: one part containing the *Accumulated Total Fare* of the ticket and a second part containing the *Temporary Fare* . In the following example, a ticket with a stored value of 1000 credits is bought as indicated in Figure 16. The value of 0xFFFF -1000 (maximum value of counter acc. to recommendations in Section 7.1) is added to the accumulated fare. The temporary fare is kept at 0x00, all in a single INCR_CTR command.

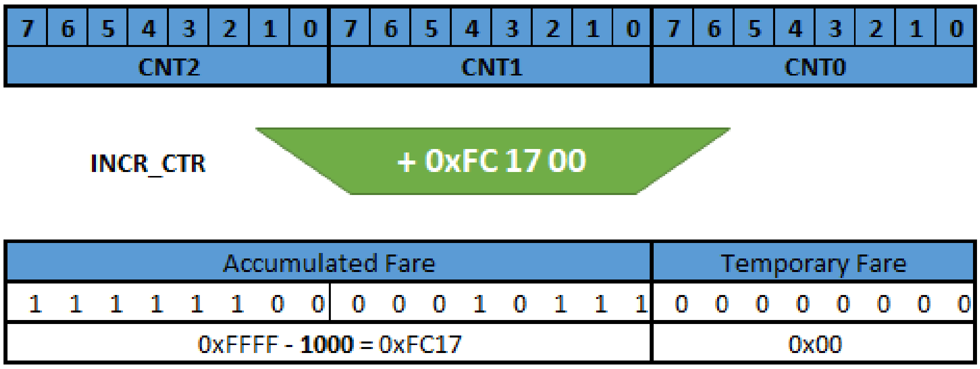


Figure 16. Counter used in a zone- or mileage-based tariff scheme after the ticket is bought.

At the check-in, the full-fare (here: 20 credits) is stored in the *Temporary Fare* in the bitwise inverted form (0xFF - 20) (see Section 7.1). This could for instance be the amount a train ride costs for the full distance. The accumulated fare stays untouched, as seen in Figure 17.

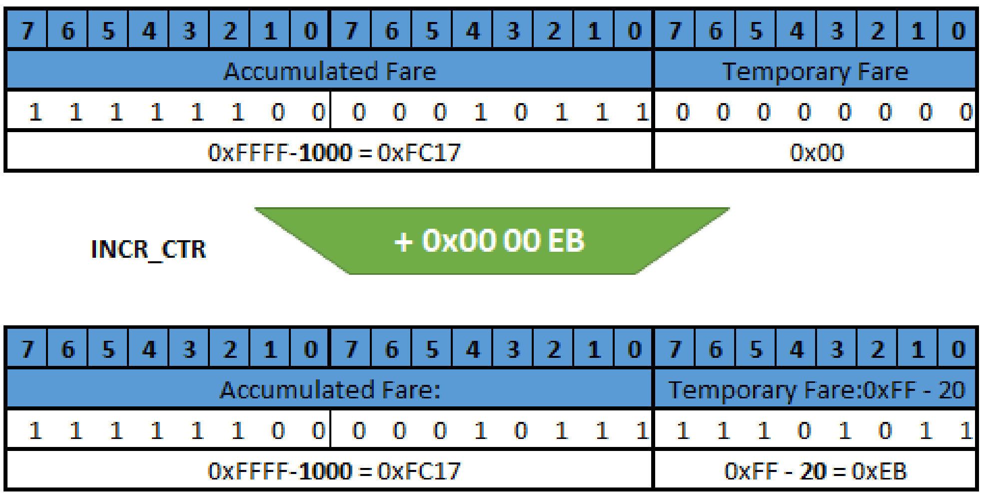


Figure 17. Full fare is added to the temporary fare

At the check-out, the real traveled credits (in case the train has been left before the final station) are accumulated in the *Accumulated Total Fare* and the *Temporary Fare* is reset, see Figure 18. Again, all this happens with a single INCR_CTR command, containing the real traveled credits minus 1 (as the reset of the temporary fare byte also will increment the accumulated counter value by one) in the 2 most significant bytes, and the full fare value plus one in the least significant byte.

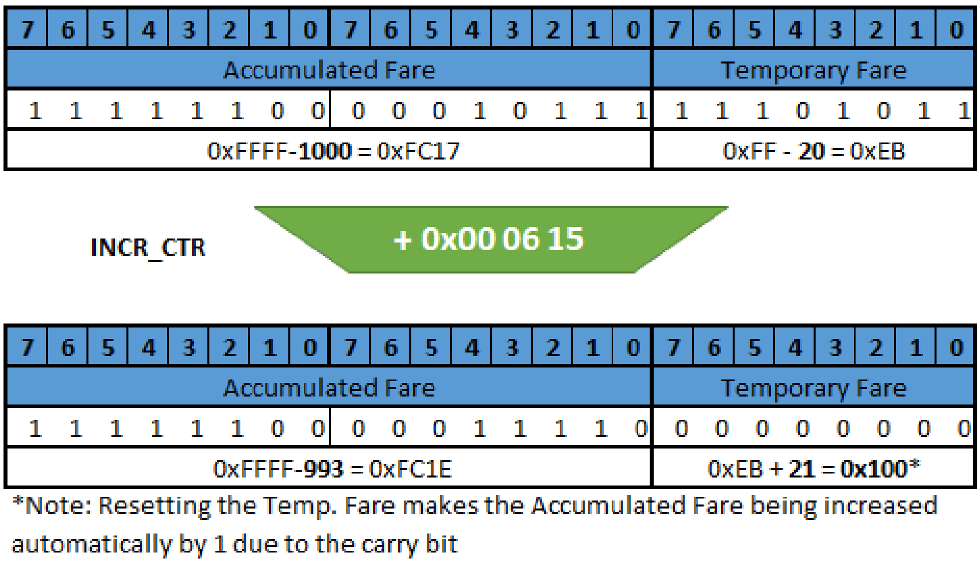


Figure 18. Counter used in a zone- or mileage-based tariff scheme at check-out where only 7 credits have been traveled

If the passenger does not check out, at the next check-in the full fare (20 credits) is charged in the *Accumulated Total Fare* and the *Temporary Fare* is set to the new full fare value (here: again 20 credits) (see: Figure 19)

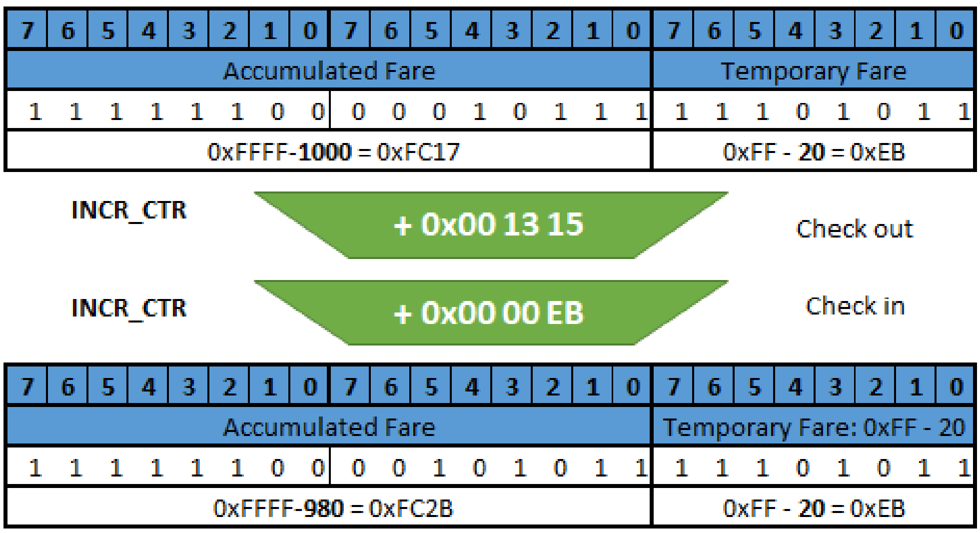


Figure 19. Counter used in a zone- or mileage-based tariff scheme at check-in in case the passenger did not check out, and checks in again

Once the accumulated fare has reached the limit (e.g. 0xFFFF), the ticked shall be invalidated (e.g. by writing to the OTP bits, or writing to the user memory) and the counter shall be set to its maximum value (0xFFFFFF).

Note: Be aware that a check-out could potentially be bypassed at the cost of incrementing the Accumulated Fare with one credit. To allow detection, one may consider to complement the counter value with a MAC stored on the card, as outlined in section Section 2.2.1. Alternatively, fraud detection via back-end checks can be considered.

5 MIFARE Ultralight EV1 password and PACK

The MIFARE Ultralight EV1 MF0ULx1 provides a password authentication to limit a part of the memory area for being accessed either in writing or reading and writing (see [MF0ULx1]).

Although the password verification method available in MIFARE Ultralight EV1 MF0ULx1 does not offer a high security protection, it can be as well used (beside the originality signature check described in [AN11341]) to verify the originality of the ticket/card. Please note that the password and the PACK are sent in plain and this needs to be considered when assessing the system security whether this is sufficient for the assets to be protected in the targeted application.

5.1 Password and PACK diversification

In case the password authentication is used, it is recommended to diversify the Password and the PACK to reduce the impact of compromised password/PACK. The diversification is done similarly to the key diversification described in [AN10922] section 2.2 “AES-128 Key”. In this case, the following items are defined:

- K: a 16 bytes AES 128 bits Master Key
- M: the 7 bytes UID of the MIFARE Ultralight EV1, also called diversification inputs
- CMAC: the output from the 128-bits AES key Diversification called “diversified key” as indicated in Section 2.2 and Figure 2 of [AN10922]
- Dp: diversified Password
- Dpack: diversified PACK

The figure below describes the diversification scheme and how to obtain the diversified Password and PACK.

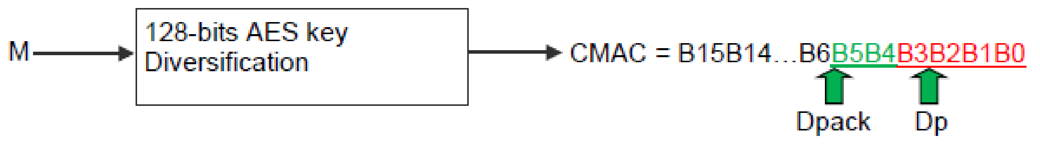


Figure 20. Password diversification scheme

From the figure, the Dp is obtained from the 4 LSB of the CMAC indicated as B3…B0, and the Dpack is derived from the next 2 bytes indicated as B5B4.

6 MIFARE Ultralight EV1 anti-cloning based on originality check

The MIFARE Ultralight EV1 MF0ULx1 supports the originality function based on a 32-byte ECC signature (see [MF0ULx1]). The application note [MFULEV1SIGNVA] describes how to validate the signature (retrieved from the MIFARE Ultralight EV1 using the READ_SIG command) using the MIFARE Ultralight EV1 UID (Unique IDentifier) and the ECC public key provided by NXP Semiconductors.

The purpose of originality check during (pre-)personalization is to protect customer investments by identifying mass penetration of non-NXP originated MIFARE Ultralight EV1 ICs into an infrastructure. As individual signatures can still be copied, it does not completely prevent hardware copy or emulation of individual MIFARE Ultralight EV1 ICs. As such, a valid signature is not a full guarantee. Therefore, this signature validation should be complemented with a check to detect if multiple ICs with the same UID are being introduced in the system.

7 MIFARE Ultralight EV1 anti-tearing implementation

The MIFARE Ultralight EV1 implements anti-tearing for OTP, lock bits and counters (see [MF0ULx1]). This means that in case of a tear-off event either the old value or the new (just written) value is present. This section describes what measures a MIFARE Ultralight EV1 application needs to implement in order to ensure the best tear-off protection for the user data pages.

For the tearing application implementation, 2 memory areas having the same size are needed see Figure 21.

Figure 21. Tearing application implementation

The application data is stored in 2 memory locations. The application data also contains a timestamp indicated in white and a CMAC (that can be calculated as indicated in Section 2.2). Every time a new update is needed i.e. new data has to be written, only the set of data with the older timestamp is updated. The CMAC is added to guarantee the integrity of the written application data.

In particular, the Figure 21 shows a typical update of the Application Data done on the older Application Data set (timestamp = t-1). As soon as the new application data is written, the timestamp is updated (timestamp = t+1) and the CMAC is also written.

If the update operation fails due to a tearing event and the application data becomes corrupted, this can be recognized based on the failure of the CMAC validation. In any case, the MIFARE Ultralight either contains the latest updated application data (timestamp = t+1) or the previous one (timestamp = t).

7.1 Recommended system implementations of tearing supported features

MIFARE Ultralight EV1 supports anti-tearing mechanisms for counter, OTP bits and Lock bits for tearing events that may occur during normal operation in the field. Security researches continuously advise the industry by publishing new attack vectors to advocate for higher secure products and implementation of system level countermeasures. It has been demonstrated that applying tearing events in specific sequences can intentionally alter the data of the counter, OTP bits and Lock bits. Therefore, it is important that additional measures are considered depending on the configuration and use case.

Table 7. System level countermeasures

Tearing supported features of MIFARE Ultralight EV1	Product and system level recommendation
Counter *(3 independent 24-bit one-way counter)*	1. Use Backend fraud detection e.g. deny listing of suspicious tickets based on UID. Once an unexpected counter decrease is seen on a specific UID, the card can be deny listed in the backend and will not be accepted anymore. 2. Disable the tickets when the maximum value of counter is reached. (See Section 2.1.2.1) 3. Start counting from high value, e.g. 0xFFFFF5 = 10 trips. Also make sure to apply recommendation 2) in this case. If migration from an old to a new system is to be done, then make sure that the system can reliable differentiate between cards in the old and the new situation in a way that cannot be misused by adversaries.
OTP bits *(One-Time-Programmable bits)*	1. In case OTP is used as counter, see above. Use a structure in programming of the bits (e.g. from left to right), and reject tickets not following this scheme. When counter is depleted then set all OTP bits (0xFFFFFF) **twice** [1] . If a depleted counter should be different from 0xFFFFFF, then set the the OTP lock bit **twice** once the counter has reached its target. 2. In case OTP does not need to be changed anymore then lock it. To do so, set the OTP lock bit and all block-locking bits and write this **twice** 3. Protect OTP by password protection
Lock bits - Potential reset of read-only pages back to writeable pages *(represent the field programmable read-only locking mechanism)*	1. Set all block-locking bits **twice** 2. Protect lock bits by password protection

[1] Writing those bits twice makes sure that the value is also written in the internal backup page

The proposed countermeasures can be applied on the IC by setting all block-locking bits and use of password. Of importance is also, that the programming of the block-locking bits and the OTP area is done twice to ensure a permanent lock. Similar can be achieved by applying an increment by 0 “INC0” for the counter when the maximum of the counter is achieved to lock it permanently. The reason for this is, that it makes sure the internal backup page is also updated correctly. (see Section 4.1 and Section 2.1.3)

System level countermeasures in general have an impact on the infrastructure (reader and backend system) and can require the storage of some extra information in the contactless card to increase the system security overall. In general these countermeasures, e.g. calculation of a CMAC over protected data can be implemented on any contactless card type, unless the storage capacity of the card is too limited to store all extra data. (see Section 7)

8 MIFARE Ultralight EV1 coil design hints

The MIFARE Ultralight EV1 chip is available in two versions: either the **“standard” version MF0ULx1** with an input capacitance of approximately 17 pF or a **high capacitance version MF0ULHx1** with approximately 50 pF. For a complete coil design, refer to the “MIFARE (Card) Coil Design Guide” [M011731].

Using the standard version of the MIFARE Ultralight EV1 chip it is recommended to use **the same coil design for the MIFARE Ultralight EV1 as for the MIFARE Classic**. Although the MIFARE Ultralight has a slightly higher capacitance than the MIFARE Classic (by 0.5 pF), the same coil design should be used to result in a slightly lower resonance frequency. This lower resonance frequency increases the overall performance of cheap antennas and ensures a similar performance compared to MIFARE Classic – but has its limitation, if multiple cards operate simultaneously in the field.

For coil design issues, it is recommended to use the Application notes “MIFARE (Card) Coil Design Guide” [M011731] and “Temperature Management, Inlet Design” [SI070010].

9 Appendix

9.1 MFRC171 low-level library extension: Cascade anti-collision

LLL adaptation to execute cascade level 2, see section 3.2

```
/************************************************/
int CALL_CONV MfPiccCascAnticoll (unsigned char select_code,
 unsigned char bcnt,
 unsigned char *snr)
/********************************************************/
{
 int status;
 unsigned char snr_chk = 0;
 int i;
 if (MfAssertMode(select_code,0x93|0x95|0x97))
 return (MI_WRONG_PARAMETER_VALUE);
MfOutp(ENABLE, _PEN | _PRE); // CRC-disable, Parity enable
MfOutp(MODE , __mode); // __mode preset
MfOutp(BCNTS ,(unsigned char)(bcnt + 16)); // 16 + number of bits
MfOutp(STACON, (unsigned char)(__stacon|_AC)); // anticollision-mode
 MfDelay50us(4); // BUS-access not allowed
 // for 35us
MfOutp(DATA, select_code); // "SELTYPE" of MIFARE1
MfOutp(DATA, (unsigned char)(((2 + (bcnt >> 3)) << 4) | (bcnt &
 0x07)));
 // bytecount higher nibble
 // bitcount lower nibble
 // incl. first 2 bytes!!
 for (i = 0; i < (bcnt + 7)/8; i++)
 {
MfOutp(DATA, snr[i] );
 }
MfOutp(TOC, TIMEOUT_14443_3); // set timeout
 while (!((status = MfInp(STACON)) & _DV));
MfOutp(TOC, 0); // reset timer
 if ((status = MfInp(STACON)) & (_TE | _BE)) // any error
 {
 if (status & _TE)
 return (MI_NOTAGERR);
 if (status & _BE)
 {
 MfDelay50us(10); // delay 500us
return (MI_BITCOUNTERR);
 }
 }
 for (i = 0; i < 4; i++)
 {
 snr[i] = MfInp(DATA);
 snr_chk ^= snr[i];
}
snr_chk ^= MfInp(DATA);
 // serialnumber check
 if (snr_chk)
return (MI_SERNRERR);
return (MI_OK);
}
```


9.2 MF CM200 / CM500 low-level library extension: Cascade anti-collision

LLL adaptation to execute cascade level 2, see section 3.2

```
/*************************************************/
int CALL_CONV MfPiccCascAnticoll (unsigned char select_code,
 unsigned char bcnt,
 unsigned char *snr)
/***********************************************************/
{
 int status;
 unsigned char snr_chk = 0;
 int i;
 if (MfAssertMode(select_code,0x93|0x95|0x97))
 return (MI_WRONG_PARAMETER_VALUE);
MfOutp(ENABLE, _PEN | _PRE); // CRC-disable, Parity enable
MfOutp(MODE , __mode); // __mode preset
MfOutp(BCNTS ,(unsigned char)(bcnt + 16)); // 16 + number of bits
MfOutp(STACON, (unsigned char)(__stacon|_AC)); // anticollision-mode
 MfDelay50us(4); // BUS-access not allowed
 // for 35us
MfOutp(DATA, select_code); // "SELTYPE" of MIFARE1
MfOutp(DATA, (unsigned char)(((2 + (bcnt >> 3)) << 4) | (bcnt &
 0x07)));
 // bytecount higher nibble
 // bitcount lower nibble
 // incl. first 2 bytes!!
 for (i = 0; i < (bcnt + 7)/8; i++)
 {
MfOutp(DATA, snr[i] );
 }
MfOutp(TOC, TIMEOUT_14443_3); // set timeout
 while (!((status = MfInp(STACON)) & _DV));
MfOutp(TOC, 0); // reset timer
 if ((status = MfInp(STACON)) & (_TE | _BE)) // any error
 {
 if (status & _TE)
 return (MI_NOTAGERR);
 if (status & _BE)
 {
 MfDelay50us(10); // delay 500us
return (MI_BITCOUNTERR);
 }
 }
 for (i = 0; i < 4; i++)
 {
 snr[i] = MfInp(DATA);
 snr_chk ^= snr[i];
}
snr_chk ^= MfInp(DATA);
 // serialnumber check
 if (snr_chk)
return (MI_SERNRERR);
return (MI_OK);
}
```

9.3 Worked out example of proposed security mechanism

An example application flow diagram is shown in the following:

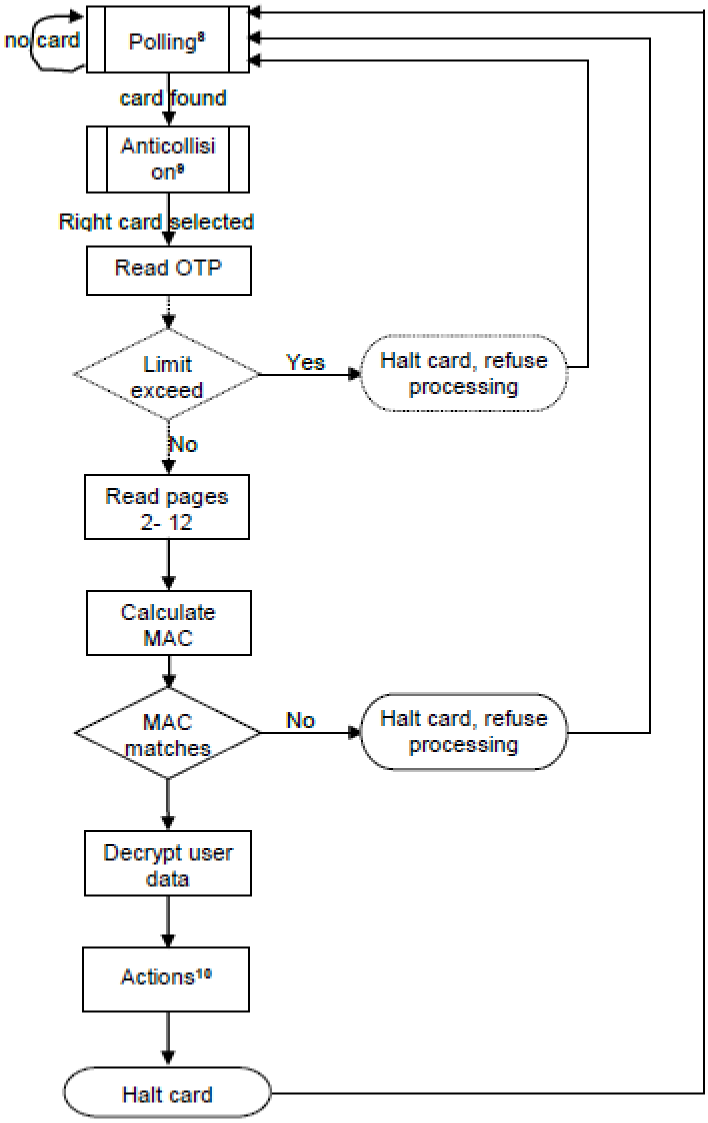


Figure 22. Example application flow diagram

Dotted blocks may be avoided, if the OTP bytes are not used.

[8] Pre-defined process for card detection, reader always sends REQA and checks if there is any answer.

[9] Standard anti-collision [ISO/IEC 14443-3], which includes the selection of the right card (also from the multiple cards).

[10] If OTP or any memory content is updated, MAC has to be recalculated and rewritten.

10 Legal information

10.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

10.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

10.3 Licenses

Purchase of NXP ICs with NFC technology
Purchase of an NXP Semiconductors IC that complies with one of the Near Field Communication (NFC) standards ISO/IEC 18092 and ISO/IEC 21481 does not convey an implied license under any patent right infringed by implementation of any of those standards. Purchase of NXP Semiconductors IC does not include a license to any NXP patent (or other IP right) covering combinations of those products with other products, whether hardware or software.

10.4 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

MIFARE — is a trademark of NXP B.V.

DESFire — is a trademark of NXP B.V.

MIFARE Plus — is a trademark of NXP B.V.

MIFARE Ultralight — is a trademark of NXP B.V.

MIFARE Classic — is a trademark of NXP B.V.

NXP — wordmark and logo are trademarks of NXP B.V.

Tables

Figures

Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 5 May 2021
Document identifier: AN11340
Document number: 073132