MIFARE Plus
Technical details

Renke Bienert

MIFARE Plus, technical details

CAS Training M2

2011

Contents

▸ Features and Functionality
  – Concept & security
  – Memory mapping

▸ Security Levels 0, 1, 2, and 3
  – Level 0: Personalisation
  – Level 1: MIFARE Classic compatible
  – Level 2: AES and more secure use of MIFARE Crypto
  – Level 3: Use of AES and T=CL protocol

▸ Additional features
  – Proximity Check
  – Virtual Card Architecture

▸ Migration concept from MIFARE Classic to MIFARE Plus

Features and Functionality

‣ MIFARE Plus
  – is a new main stream smart card IC of the MIFARE product family,
  – has been designed for use in public transport and access management,
  – uniquely features outstanding AES based security enhancements,
  – protects investments with existing MIFARE infrastructure,
  – is available via NXP's worldwide partner network.
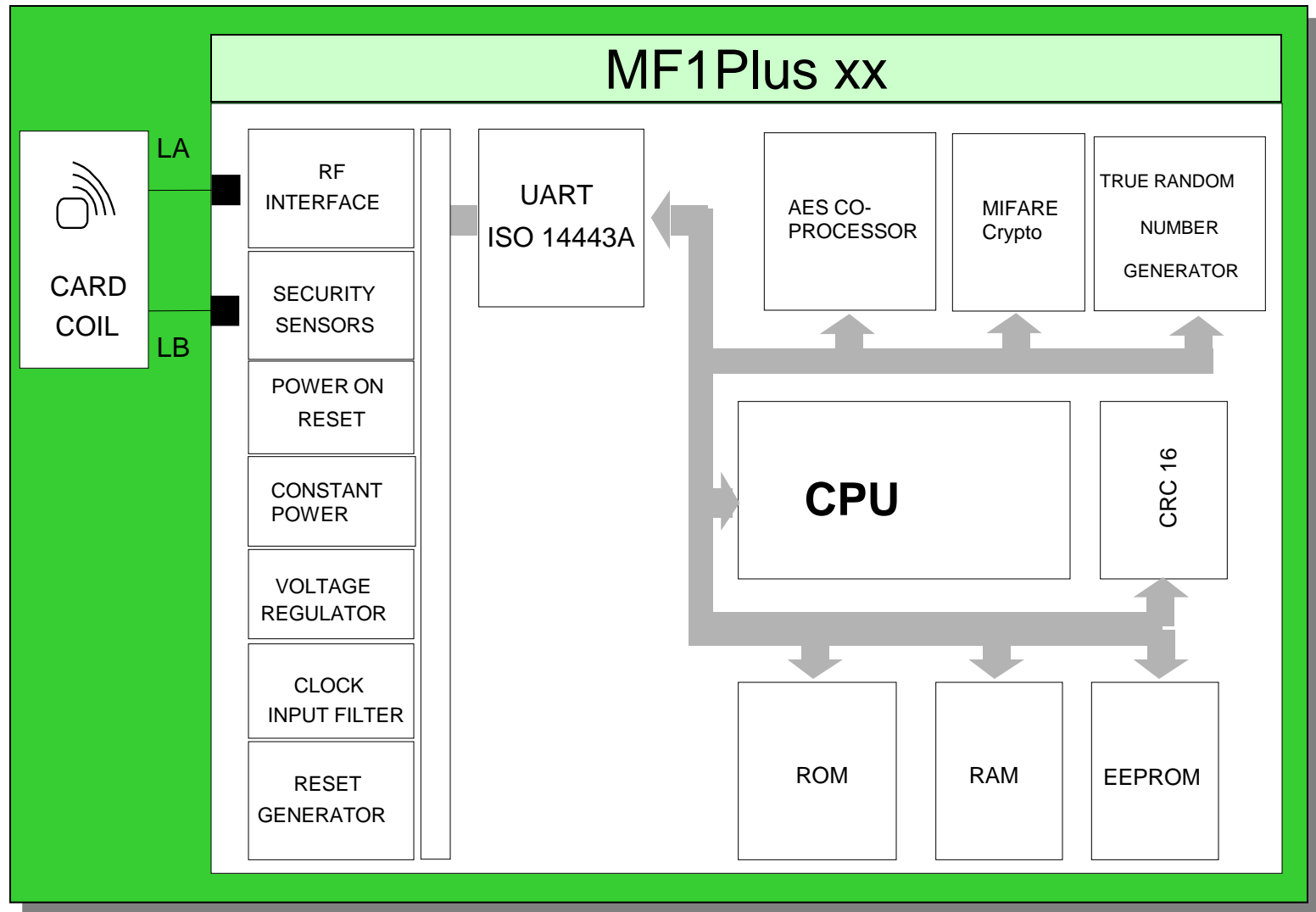
‣ Further information can be found here:
  http://www.mifare.net/products/mifare_plus.asp

Features and Functionality

▸ MIFARE Classic compatible
  – Security Level concept allows easy system upgrade

▸ 4 Byte ONUID or 7 Byte UID (double size UID acc. to ISO/IEC 14443)
  – Optional Random ID
  – UID always available e.g. to use key diversification

▸ 2K / 4K Byte EEPROM
  – Same memory structure as MIFARE Classic
  – No need to change the card data layout in the system

▸ AES Authentication/Encryption/MAC
  – Different levels of MAC and encryption
  – Configurable acc. to system requirements

▸ Anti-tearing for AES Keys (SL2 & SL3) and Sector Trailers (SL3)

▸ Originality Function
  – Guarantees correct NXP card IC

▸ Proximity Check
  – Offers option to prevent relay atacks

▸ Data rates up to 848 kbit/s
  – According to ISO/IEC 14443

▸ Common Criteria evaluation and certification level 4+ (HW & SW)

▸ Supports system migration from MIFARE Classic

MIFARE Plus Block diagram

MF1Plus xx

CARD COIL — LA / LB

RF INTERFACE

SECURITY SENSORS

POWER ON RESET

CONSTANT POWER

VOLTAGE REGULATOR

CLOCK INPUT FILTER

RESET GENERATOR

UART ISO 14443A

AES CO-PROCESSOR

MIFARE Crypto

TRUE RANDOM NUMBER GENERATOR

CPU

CRC 16

ROM

RAM

EEPROM

Abbreviations & Terms

▸ Auth = Authentication (i.e. 3-pass mutual authentication)

▸ SLx = one of the 4 Security Levels of MIFARE Plus

▸ MAC = Message Authentication Code

▸ POR = Power on Reset

▸ VC = Virtual Card

▸ LSB = Least Significant Byte

▸ MSB = Most Significant Byte

Be aware that the training slides do not replace any of the official documents.

MIFARE Plus derivates

- MIFARE Plus S (2 KByte)
  - 4 Byte ONUID
  - 7 Byte UID

- MIFARE Plus S (4 KByte)
  - 4 Byte ONUID
  - 7 Byte UID

- MIFARE Plus X (2 KByte)
  - 4 Byte ONUID
  - 7 Byte UID

- MIFARE Plus X (4 KByte)
  - 4 Byte ONUID
  - 7 Byte UID

MIFARE Plus Type Identification

MIFARE Plus ATS Coding

'OC'	'75'	'77'	'80'	'02'	'C1'	'05'	'2F'	'2F'	'01'	'BC'	'D6'	C0	C1
TL	T0	TA(1)	TB(1)	TC(1)				T1 „Historical Characters					CRC

T1: 'Historical characters': see next slide

'Interface byte TC(1)': CID supported, NAD not supported

'Interface byte TB(1)':
High Nibble: Frame Waiting Time (FWT) (77.33 ms)
Low Nibble: Start-up frame guard time (SFGT) (302 μs)

'Interface byte TA(1)': possible data rates supported by the PICC.
(The *MIFARE Plus* supports up to 848 kbaud in both directions.)
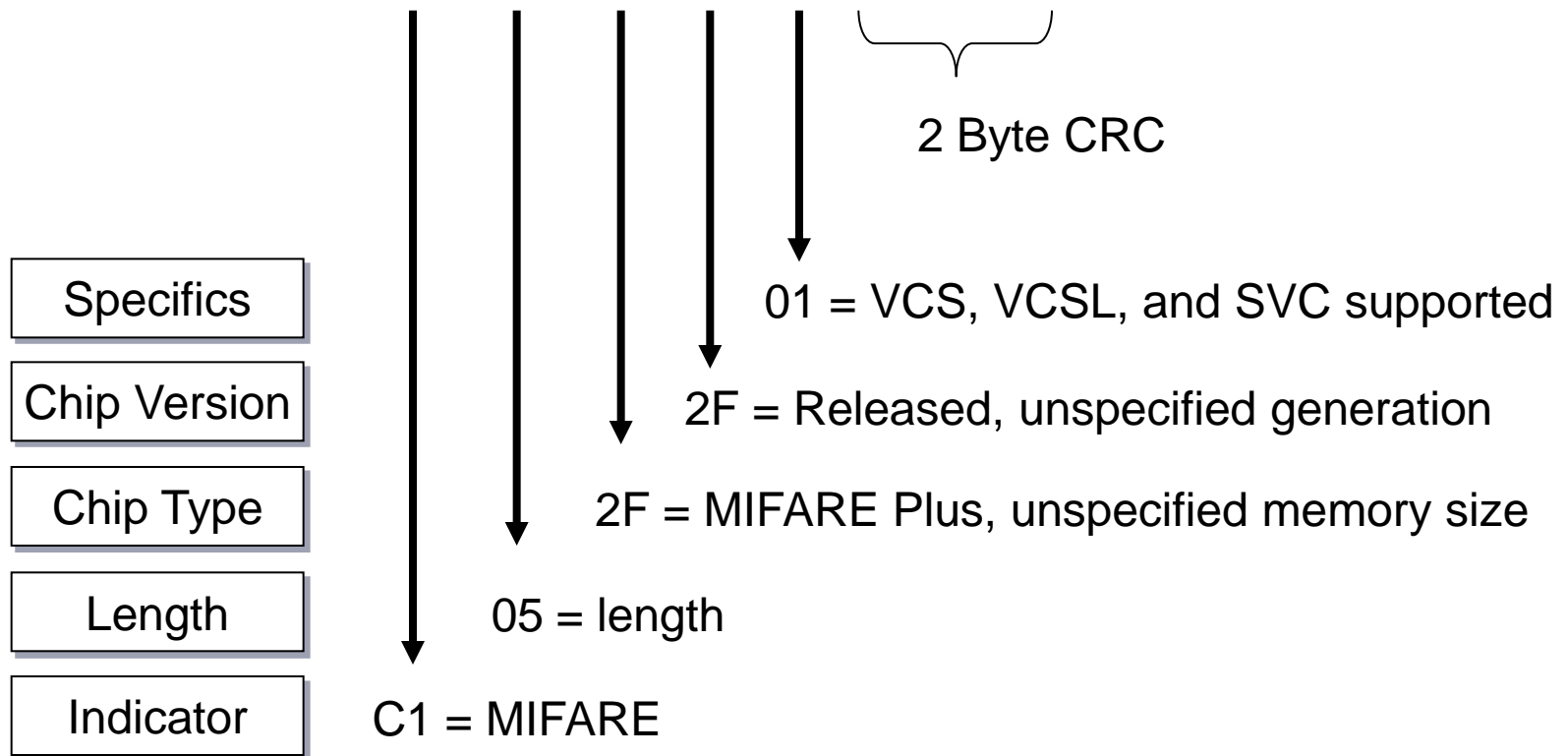
T0: 'Format Byte'
High Nibble: presence of TA(1), TB(1) and TC(1)
Low Nibble: 'FSCI' (maximum accepted size of a frame)

TL: 'Length Byte' of the transmitted ATS
(including itself, but excluding the two CRC bytes)

MIFARE Plus X ATS Coding of Historical Characters

'OC'	'75'	'77'	'80'	'02'	'C1'	'05'	'2F'	'2F'	'01'	'BC'	'D6'	C0	C1

TL   T0   TA(1)   TB(1)   TC(1)   T1 „Historical Characters   CRC

2 Byte CRC

Specifics

Chip Version

Chip Type

Length

Indicator

01 = VCS, VCSL, and SVC supported

2F = Released, unspecified generation

2F = MIFARE Plus, unspecified memory size

05 = length

C1 = MIFARE

MIFARE Plus S ATS Coding of Historical Characters

'0C'	'75'	'77'	'80'	'02'	'C1'	'05'	'2F'	'2F'	'00'	'35'	'C7'	C0	C1
TL	T0	TA(1)	TB(1)	TC(1)			T1 „Historical Characters					CRC	

2 Byte CRC

Specifics

Chip Version

Chip Type

Length

Indicator

00 = only VCSL supported

2F = Released, unspecified generation

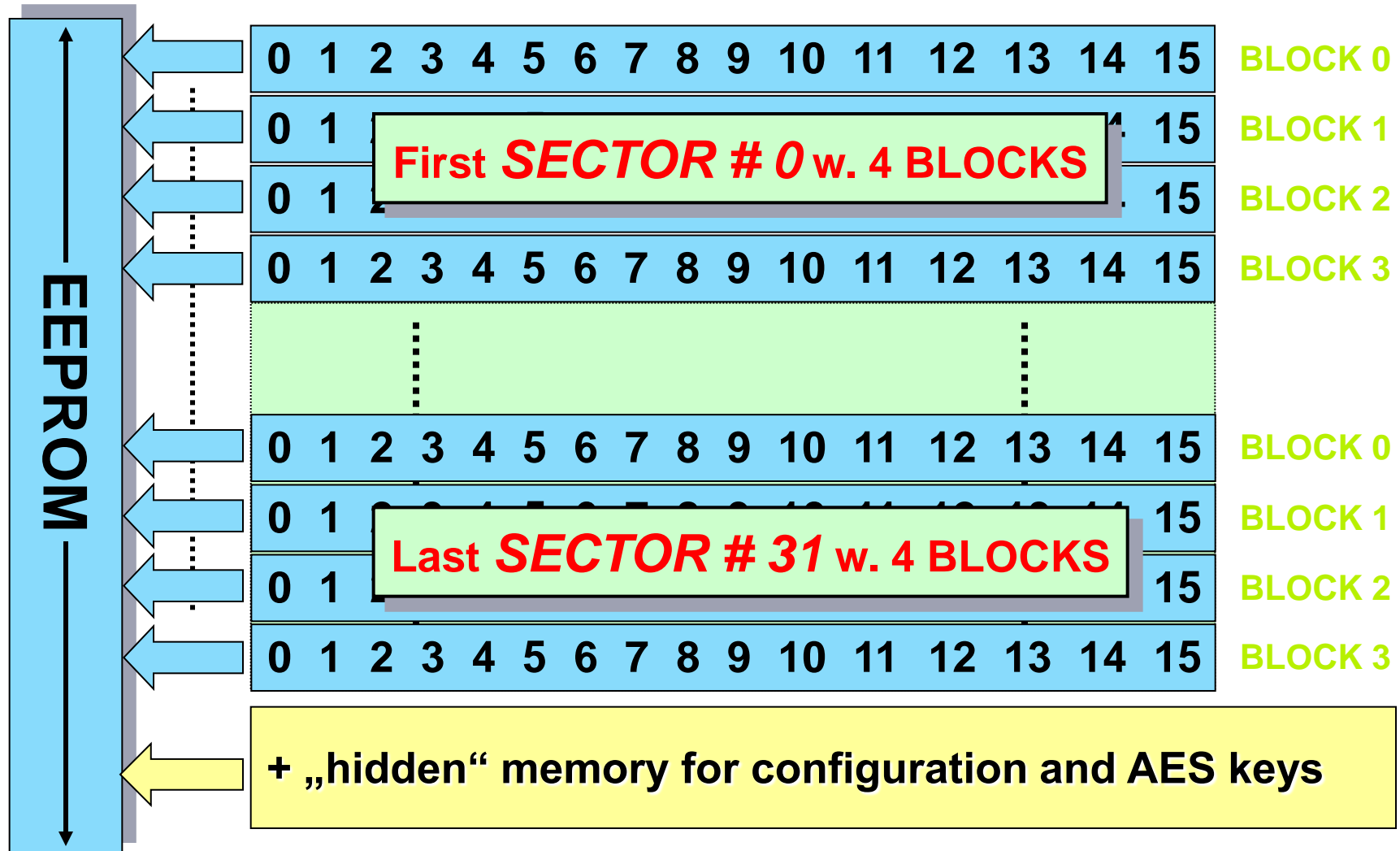2F = MIFARE Plus, unspecified memory size

05 = length
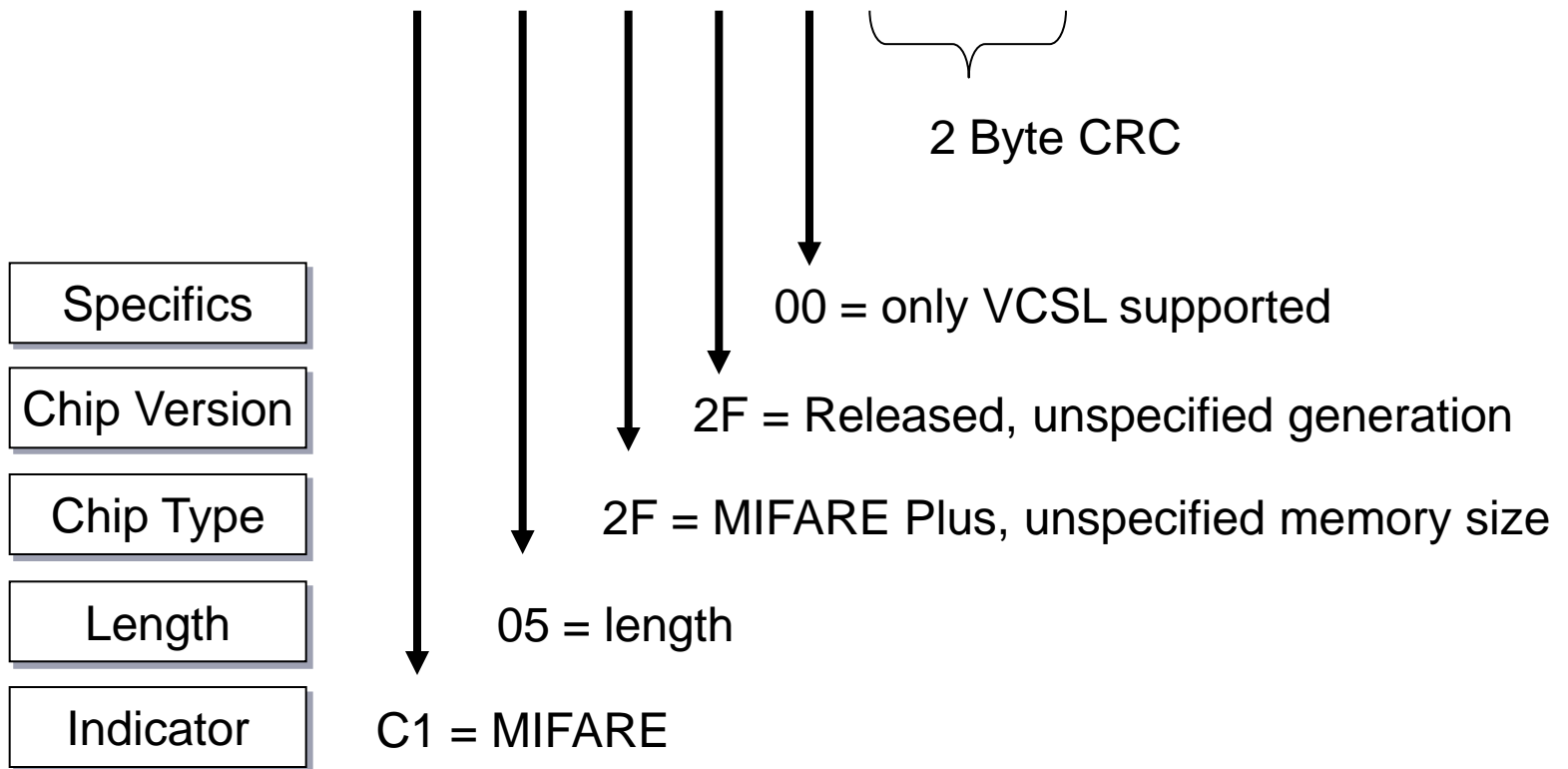
C1 = MIFARE

MIFARE Plus Memory Mapping

Memory Mapping of MF1 Plus 60 (2 kByte)

2048 Byte in 32 *SECTORS* with 128 addressable BLOCKS @ 16 BYTE each

EEPROM	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	**BLOCK 0**

First *SECTOR # 0* w. 4 BLOCKS

Last *SECTOR # 31* w. 4 BLOCKS

BLOCK 0
BLOCK 1
BLOCK 2
BLOCK 3

BLOCK 0
BLOCK 1
BLOCK 2
BLOCK 3

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

+ „hidden" memory for configuration and AES keys

Memory Mapping of MF1 Plus 80 (4 kByte)

4048 Byte in 40 *SECTORS* with 256 addressable BLOCKS @ 16 BYTE each

EEPROM

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	**BLOCK 0**

0	*SECTOR # 0-31* w. 4 BLOCKS	15	**BLOCK 1**
0		15	**BLOCK 2**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	**BLOCK 3**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	**BLOCK 0**
0															15	**BLOCK 1**

***SECTOR # 32-39* w. 16 BLOCKS**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	**BLOCK 15**

+ „hidden" memory for configuration and AES keys
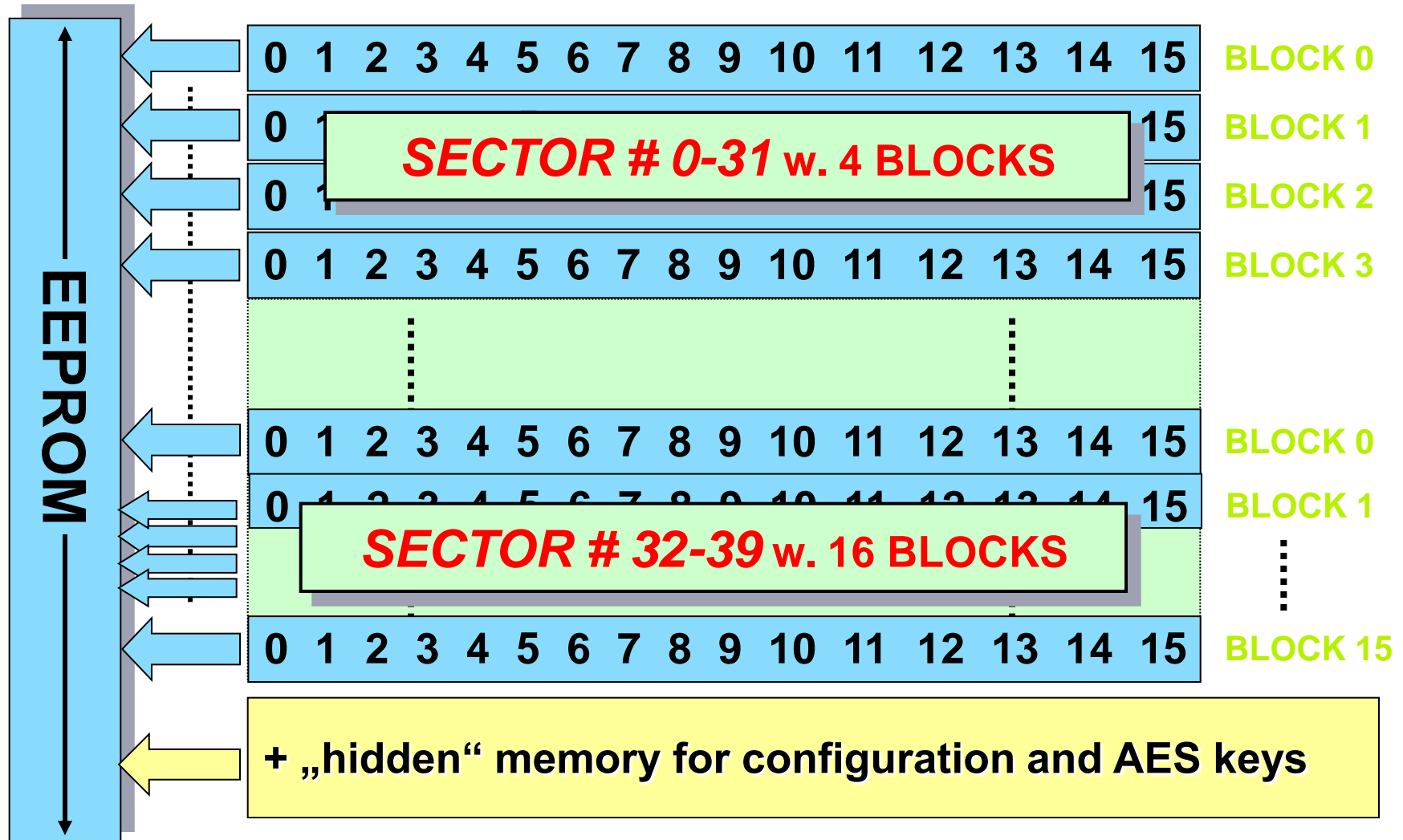
MIFARE Plus Security Levels

MIFARE Plus Security Levels
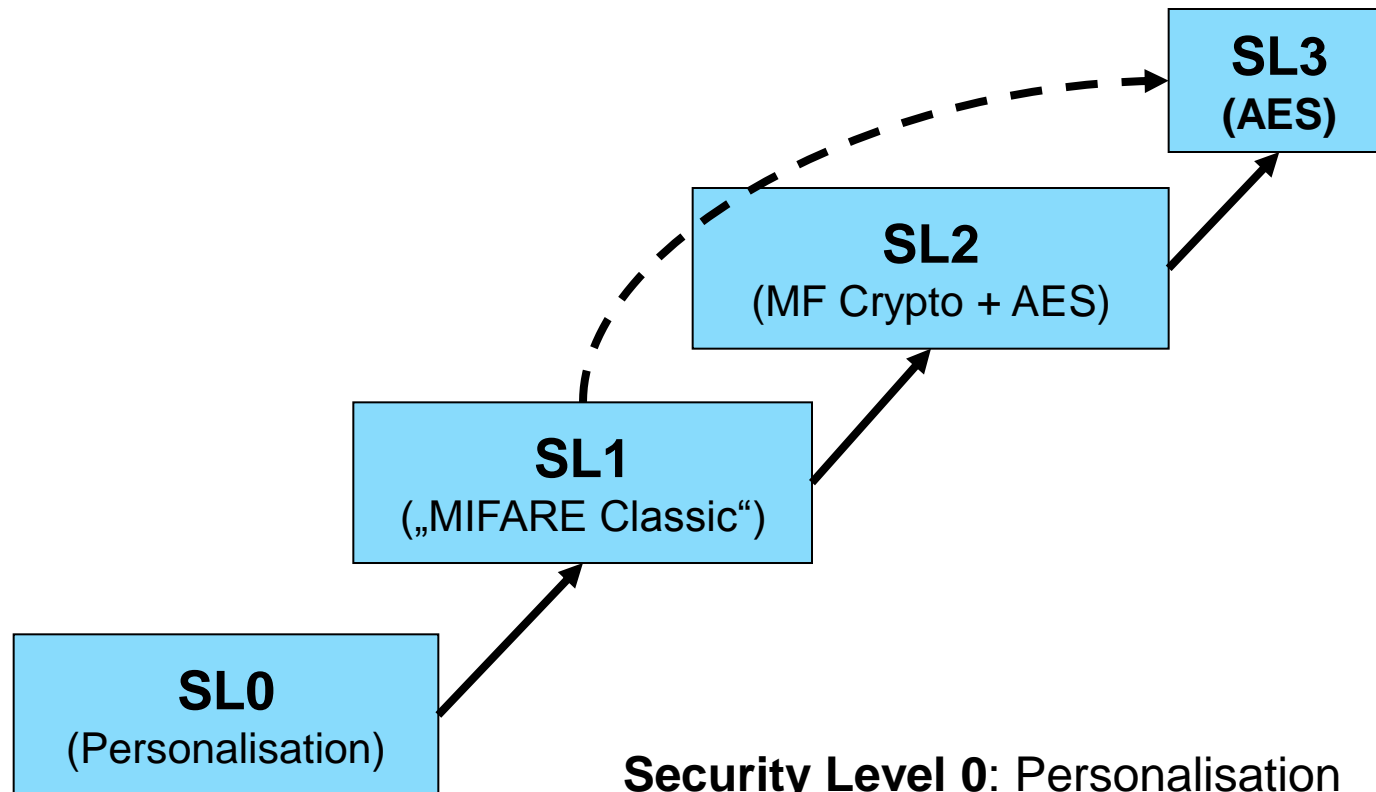
Security Level 0: Personalisation
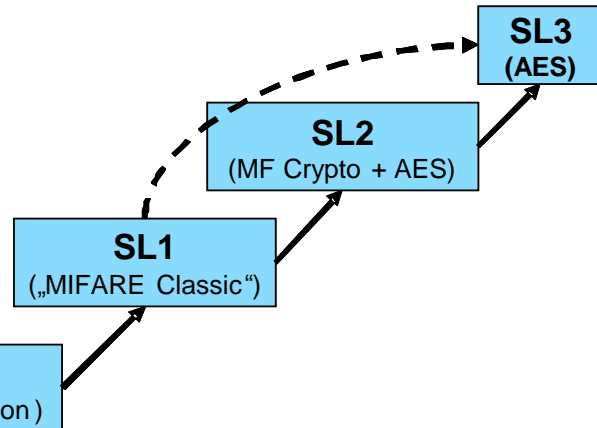**Security Level 1**: MIFARE Classic compatible
Security Level 2: AES + use of MIFARE Crypto
Security Level 3: Use of AES and T=CL protocol

MIFARE Plus Security Level 0

Personalisation

MIFARE Plus Security Level 0

```
        ┌──────────┐
        │   SL3    │
        │  (AES)   │
        └──────────┘
            ↑  ↑
   ┌──────────────────┐
   │       SL2        │
   │ (MF Crypto + AES)│
   └──────────────────┘
          ↑
 ┌──────────────────┐
 │      SL1         │
 │("MIFARE Classic")│
 └──────────────────┘
       ↑
┌──────────────────┐
│      SL0         │
│(Personalisation )│
└──────────────────┘
```
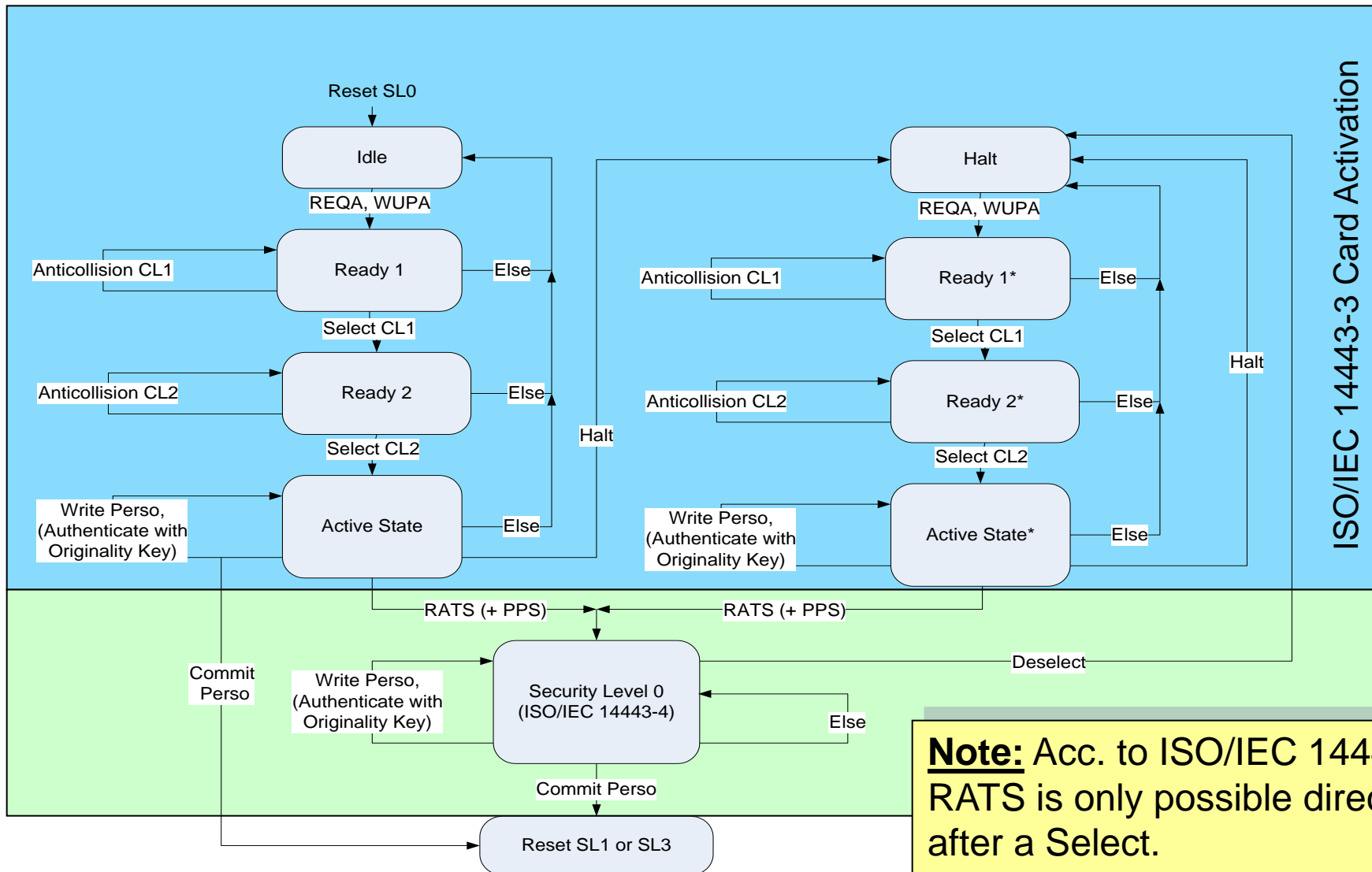
Personalisation

Security Level 0 just allows the personalisation of the MIFARE Plus:
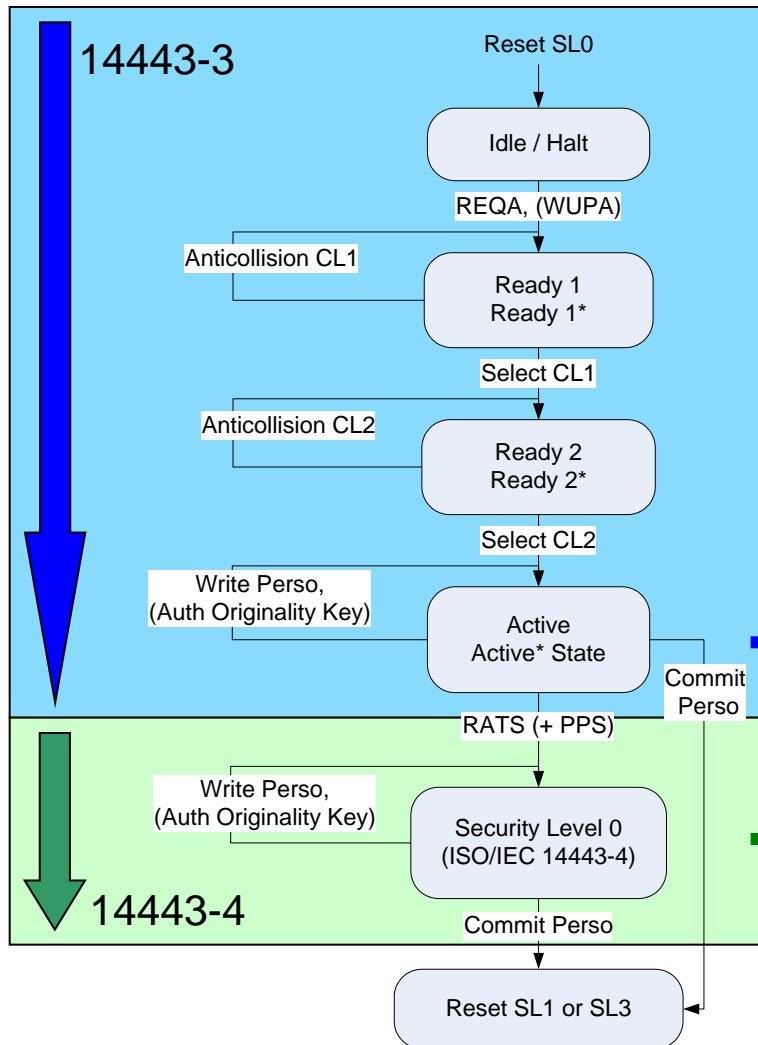
- **Write Perso**:
  - Write Data (optional)
  - Write Configuration (optional)
  - Write Keys (mandatory)

- **Commit Perso**:
  - finalises personalisation
  - switches to SL1

State diagram Security Level 0 (Details)

ISO/IEC 14443-3 Card Activation

Reset SL0 → Idle

Idle → (REQA, WUPA) → Ready 1

Anticollision CL1 → Ready 1 → Else

Ready 1 → (Select CL1) → Ready 2

Anticollision CL2 → Ready 2 → Else

Ready 2 → (Select CL2) → Active State

Write Perso, (Authenticate with Originality Key) → Active State → Else

Halt → (REQA, WUPA) → Ready 1*

Anticollision CL1 → Ready 1* → Else

Ready 1* → (Select CL1) → Ready 2*

Anticollision CL2 → Ready 2* → Else

Ready 2* → (Select CL2) → Active State*

Write Perso, (Authenticate with Originality Key) → Active State* → Else

Active State → RATS (+ PPS) → Security Level 0 (ISO/IEC 14443-4) ← RATS (+ PPS) → Active State*

Commit Perso

Write Perso, (Authenticate with Originality Key) → Security Level 0 (ISO/IEC 14443-4) → Else

Deselect

Security Level 0 (ISO/IEC 14443-4) → Commit Perso → Reset SL1 or SL3

Note: Acc. to ISO/IEC 14443 RATS is only possible directly after a Select.
Ready 2 is only applicable for 7-byte UID

State diagram Security Level 0 (simplified)

Notes:
- Card Activation acc. to ISO/IEC 14443
- PPS is optional

- Error behaviour as in ISO/IEC 14443

Personalisation (Write Perso + Commit Perso) can be done

Using protocol like MIFARE Classic

or

Using 14443-4 protocol („T=CL").

Diagram labels:
- 14443-3
- Reset SL0
- Idle / Halt
- REQA, (WUPA)
- Anticollision CL1
- Ready 1 / Ready 1*
- Select CL1
- Anticollision CL2
- Ready 2 / Ready 2*
- Select CL2
- Write Perso, (Auth Originality Key)
- Active / Active* State
- Commit Perso
- RATS (+ PPS)
- Write Perso, (Auth Originality Key)
- Security Level 0 (ISO/IEC 14443-4)
- Commit Perso
- 14443-4
- Reset SL1 or SL3

Note: Acc. to ISO/IEC 14443 RATS is only possible directly after a Select. Ready 2 is only applicable for 7-byte UID

Personalisation of MIFARE Plus

Write Perso

- ▸ Mandatory:
    - – **Write Card Master Key (9000$_{hex}$)**
    - – **Write Card Configuration Key (9001$_{hex}$)**
    - – **Write Level 2 Switch Key* (9002$_{hex}$)**
    - – **Write Level 3 Switch Key (9003$_{hex}$)**

MUST!

- ▸ Optional (recommended):
    - – Write all other Keys
    - – Write configuration blocks

- ▸ Optional
    - – Write Initial data

> * <u>MIFARE Plus S</u> does not support SL2:
> -> no Level 2 Switch Key is required.

Commit Perso

- ▸ Mandatory.

The importance of writing ALL keys at Security Level 0

▸ AES keys cannot be written in Security Level 1

▸ When switching to Security Level 2 or 3 and the AES keys are not written, sectors will be protected only by
  - **default keys = no protection**.

▸ So if AES keys are not written during Security Level 0, the switching to a higher security level cannot take place in the field:
  – Cards need to be taken from the user to a secure environment
  – Switch must be made to the higher security level
  – Keys must be replaced
  – Card can be handed back to the user

MIFARE Plus Security Level 1

MIFARE Classic compatible

MIFARE Plus Security Level 1

SL3
(AES)

SL2
(MF Crypto + AES)

SL1
(„MIFARE Classic")

SL0
(Personalisation)
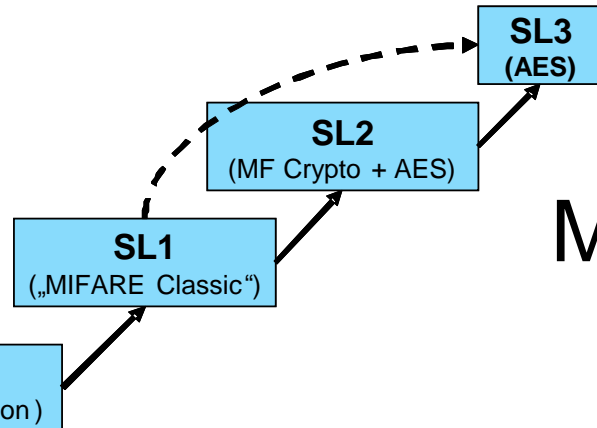
MIFARE Classic compatible

Security Level 1 offers the features of the well known MIFARE Classic:

- **MIFARE Authentication / Encryption**
  - Same as MF1 ICS 50 or MF 1ICS70
  - **Consider security risks!**
- **MIFARE Read / Write**
  - Same as MF1 ICS 50 or MF 1ICS70
- **MIFARE Value operations**
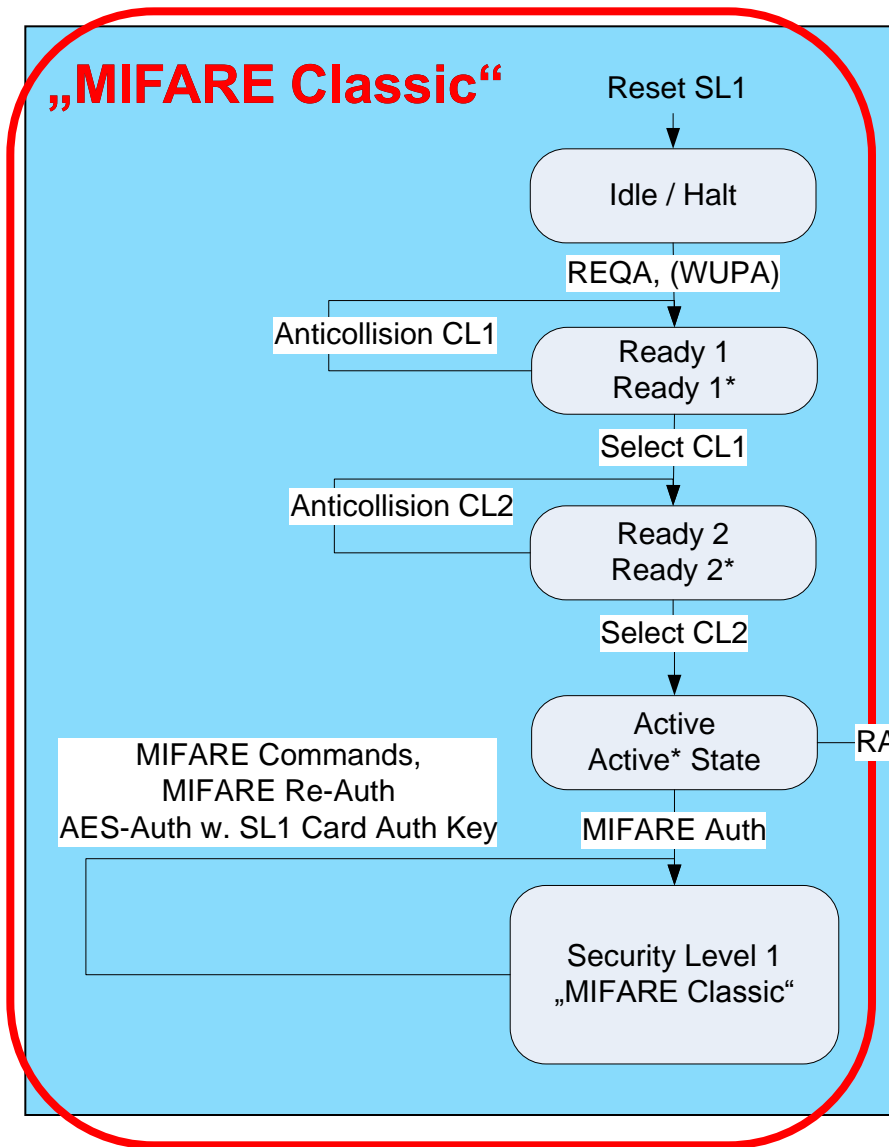 - Increment / Decrement / Restore + Transfer
  - Same as MF1 ICS 50 or MF 1ICS70

Be aware of the
UID length!

Vulnerability: Differences between IC types

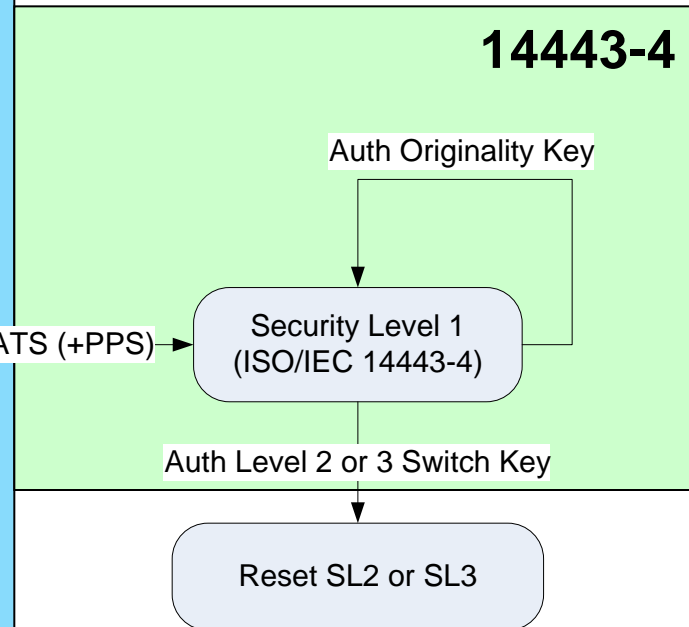|	Vulnerability	MIFARE Classic Card	MIFARE Classic Emulation on ProX or SmartMX	MIFARE Classic Emulation on MIFARE Plus Security Level 1		Non-NXP MIFARE Classic implem.
				No AES card auth.	With AES card auth. (note 1)	
1	Eavesdropping Tx + Rx data during one valid transaction	Yes	Yes	Yes	Depends	Yes
2	Eavesdropping Tx data during two valid transactions	Yes	Yes	Yes	Depends	Yes
3	Eavesdropping the result of two failed authentications	Yes	Yes	Yes	No	Yes
4	Attack without a legitimate transaction	Yes	Yes	No	No	Depends
5	With one key all other keys of the card can be retrieved	Yes	No	No	No	Depends

Note 1: Other attacks (not described here) will remain possible (with other/ less impact)

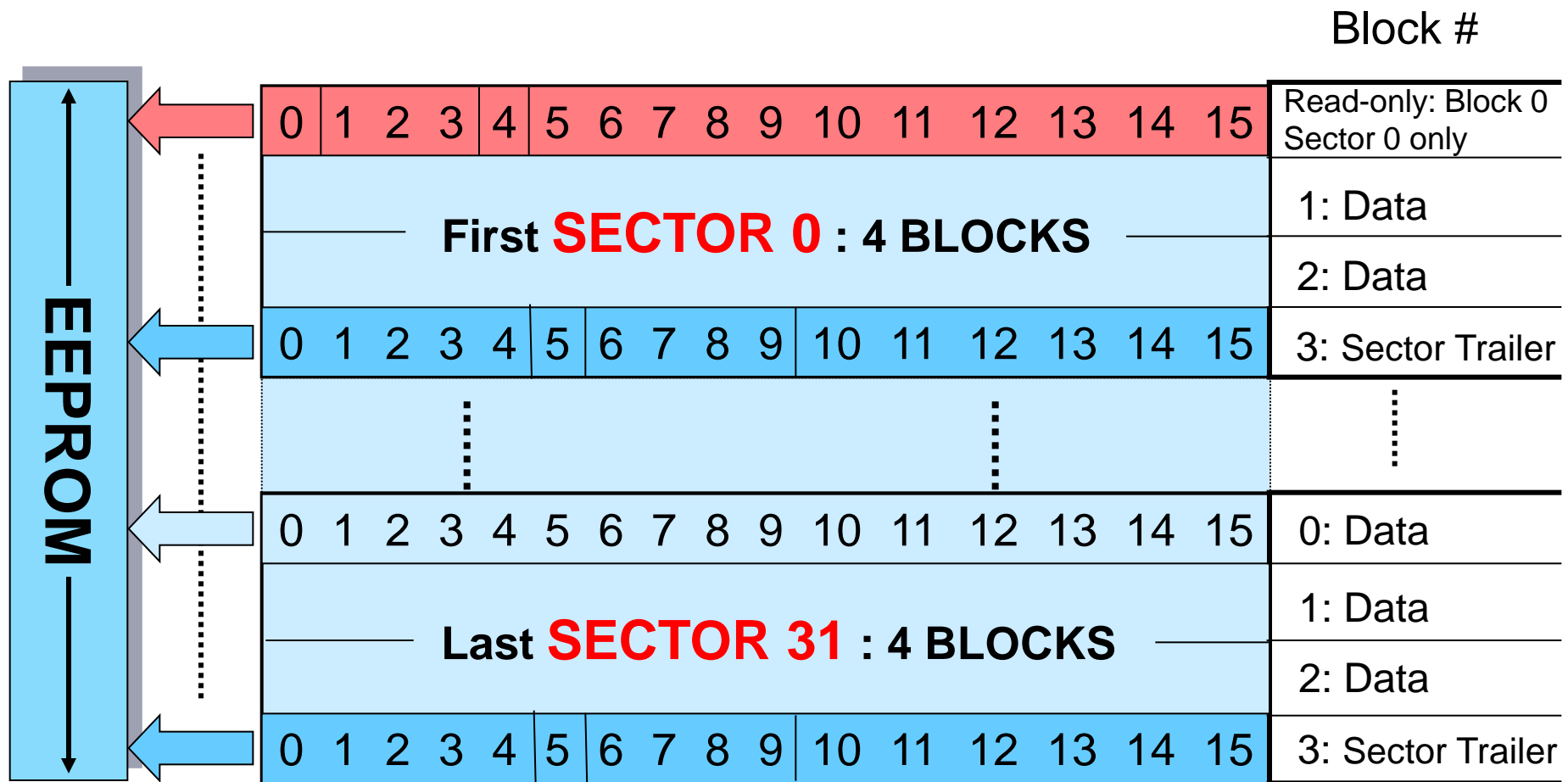# State diagram Security Level 1 (simplified)

"MIFARE Classic"

Reset SL1

Idle / Halt

REQA, (WUPA)

Anticollision CL1 → Ready 1 / Ready 1*

Select CL1

Anticollision CL2 → Ready 2 / Ready 2*

Select CL2

Active / Active* State

MIFARE Commands, MIFARE Re-Auth AES-Auth w. SL1 Card Auth Key

MIFARE Auth

Security Level 1 "MIFARE Classic"

Notes:
- Card Activation acc. to ISO/IEC 14443
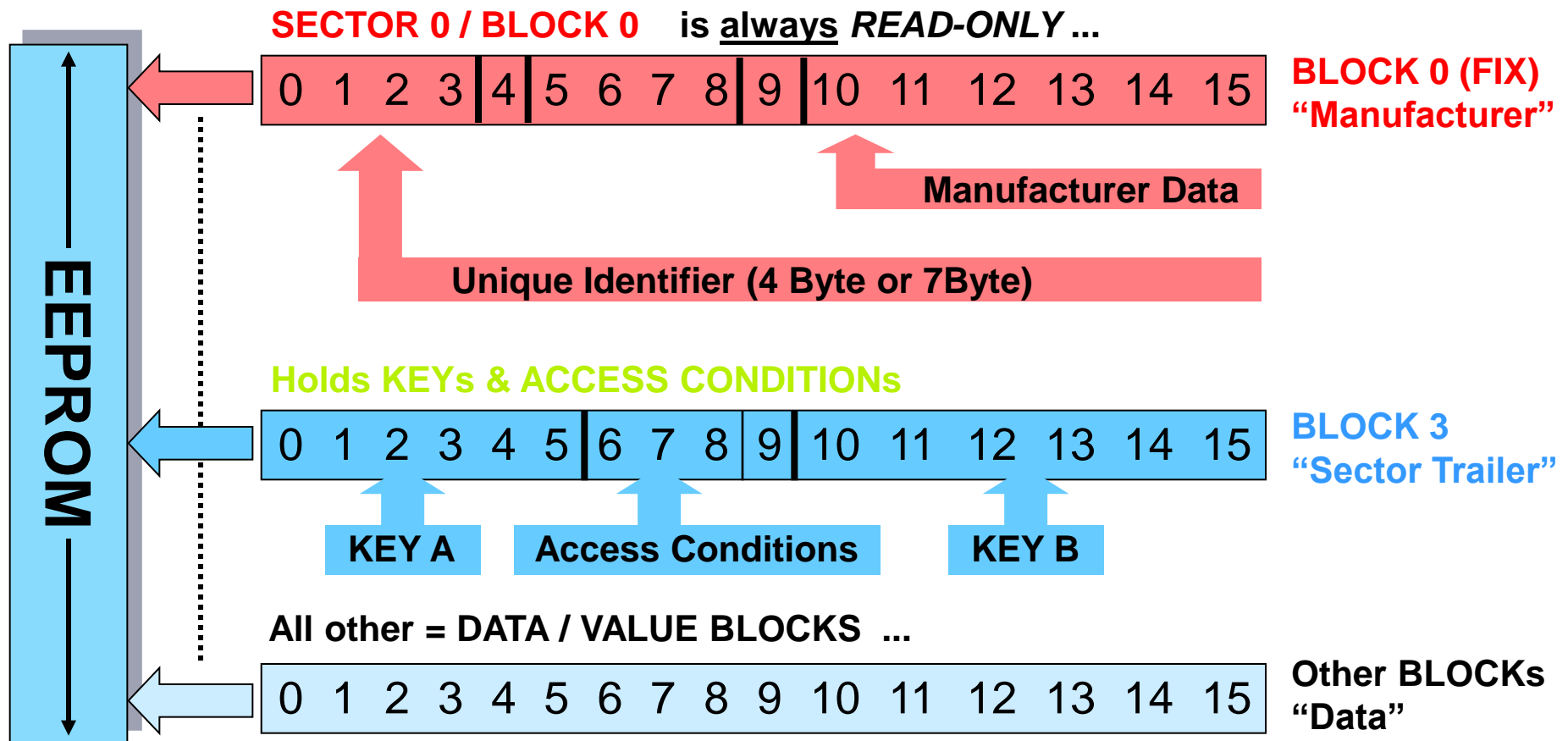- PPS is optional
- Error behaviour as in ISO/IEC 14443

14443-4

RATS (+PPS) →

Auth Originality Key

Security Level 1 (ISO/IEC 14443-4)

Auth Level 2 or 3 Switch Key

Reset SL2 or SL3

Blocks and Sectors of MF1 Plus 60 (2 kByte)

Block #

EEPROM	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Read-only: Block 0 Sector 0 only

First SECTOR 0 : 4 BLOCKS

1: Data

2: Data

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 — 3: Sector Trailer

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 — 0: Data

1: Data

Last SECTOR 31 : 4 BLOCKS

2: Data

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 — 3: Sector Trailer

Same as in MIFARE Classic!

Block function SL1 („MIFARE Classic")

SECTOR 0 / BLOCK 0 is <u>always</u> *READ-ONLY ...*

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

BLOCK 0 (FIX) "Manufacturer"

Manufacturer Data

Unique Identifier (4 Byte or 7Byte)

Holds KEYs & ACCESS CONDITIONs

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

BLOCK 3 "Sector Trailer"

KEY A   **Access Conditions**   **KEY B**

All other = DATA / VALUE BLOCKS ...

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

Other BLOCKs "Data"

EEPROM

Sector structure in SL1 and SL2 of MF1 Plus 60 (2 kByte)

SECTOR 0 … 31 *(except SECTOR 0, Block 0!)*

DATA BLOCK OPERATIONS (enable or disable)

- READ
- WRITE

- INCREMENT

- DECREMENT
- TRANSFER
- RESTORE

BLOCK 0, 1, 2 (Data or Value)

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

BLOCK 3 "Sector Trailer"

KEY A — *KEY FUNCTIONALITY* — **KEY B**

Sector Access Conditions (24 Bit)

Byte 9 = GPB = xx (not defined)
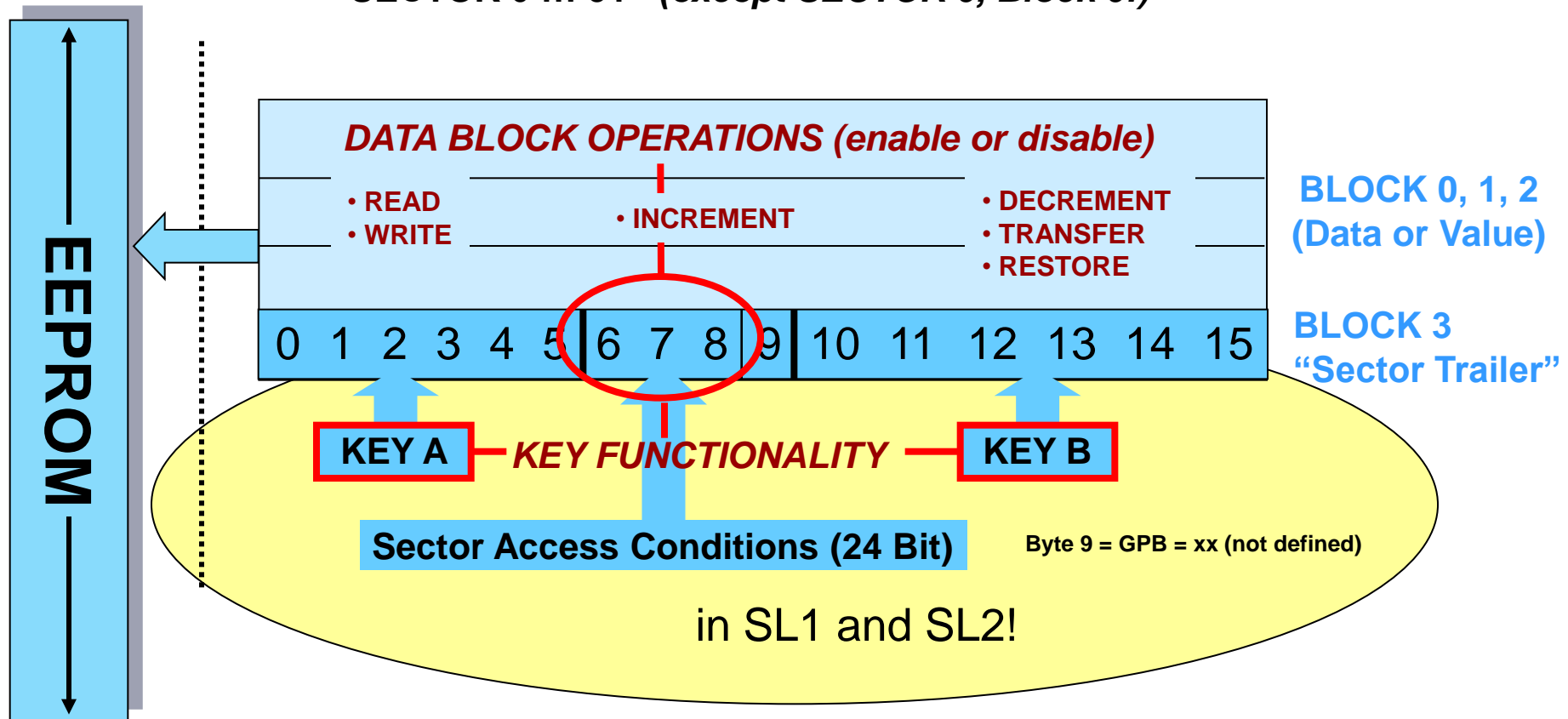
in SL1 and SL2!

Sector structure in SL1 and SL2 of MF1 Plus 80 (4 kByte)

SECTOR 0 … 31   same as MF1 Plus 60
SECTOR 32…39   with 16 Block per Sector

EEPROM

DATA BLOCK OPERATIONS (enable or disable)

- **READ**
- **WRITE**

- **INCREMENT**

- **DECREMENT**
- **TRANSFER**
- **RESTORE**

**BLOCK 0 …14
(Data or Value)**

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

**BLOCK 15
"Sector Trailer"**

KEY A — ***KEY FUNCTIONALITY*** — **KEY B**

Sector Access Conditions (24 Bit)

Byte 9 = GPB = xx (not defined)

in SL1 and SL2!
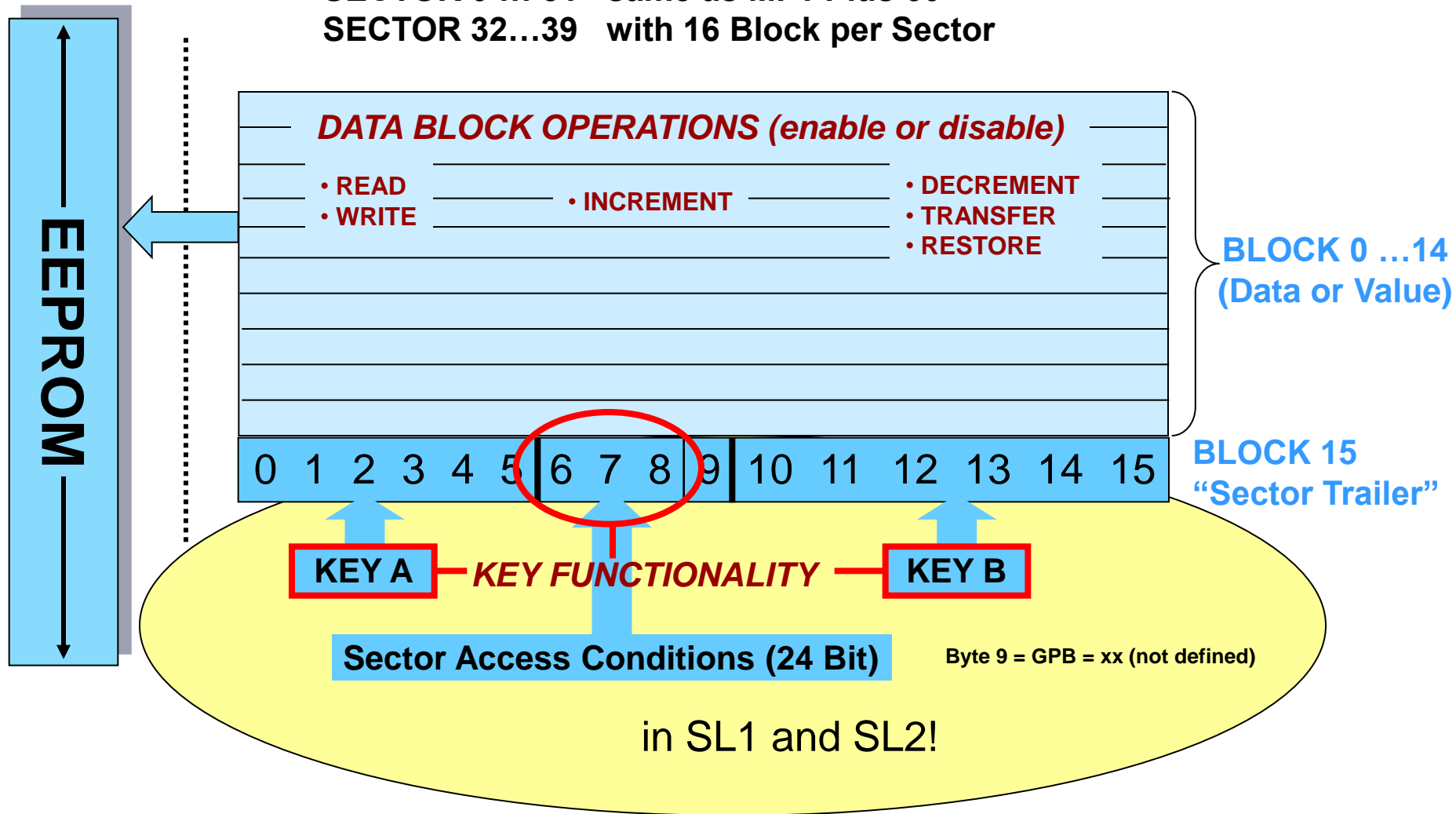
Access condition coding Sector 0...31

3 Control bits ($C1_n$, $C2_n$, $C3_n$) for **each** block

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	BLOCK 0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	BLOCK 1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	BLOCK 2
0	1	KEY A		4	5	6	7	8	9	10	11	KEY B	13	14	15	BLOCK 3

$\overline{Cx}$ = INV Cx

BLOCK 3 = Sector Trailer

BIT 0	$\overline{C1}$	$\overline{C3}$	C2	Control Bits C1 / C2 / C3 - **BLOCK 0**
BIT 1	$\overline{C1}$	$\overline{C3}$	C2	Control Bits C1 / C2 / C3 - **BLOCK 1**
BIT 2	$\overline{C1}$	$\overline{C3}$	C2	Control Bits C1 / C2 / C3 - **BLOCK 2**
BIT 3	$\overline{C1}$	$\overline{C3}$	C2	Control Bits C1 / C2 / C3 - **BLOCK 3**
BIT 4	$\overline{C2}$	C1	C3	Control Bits C1 / C2 / C3 - **BLOCK 0**
BIT 5	$\overline{C2}$	C1	C3	Control Bits C1 / C2 / C3 - **BLOCK 1**
BIT 6	$\overline{C2}$	C1	C3	Control Bits C1 / C2 / C3 - **BLOCK 2**
BIT 7	$\overline{C2}$	C1	C3	Control Bits C1 / C2 / C3 - **BLOCK 3**

EEPROM

3 Control bits (C1$_n$, C2$_n$, C3$_n$) for each 5 blocks
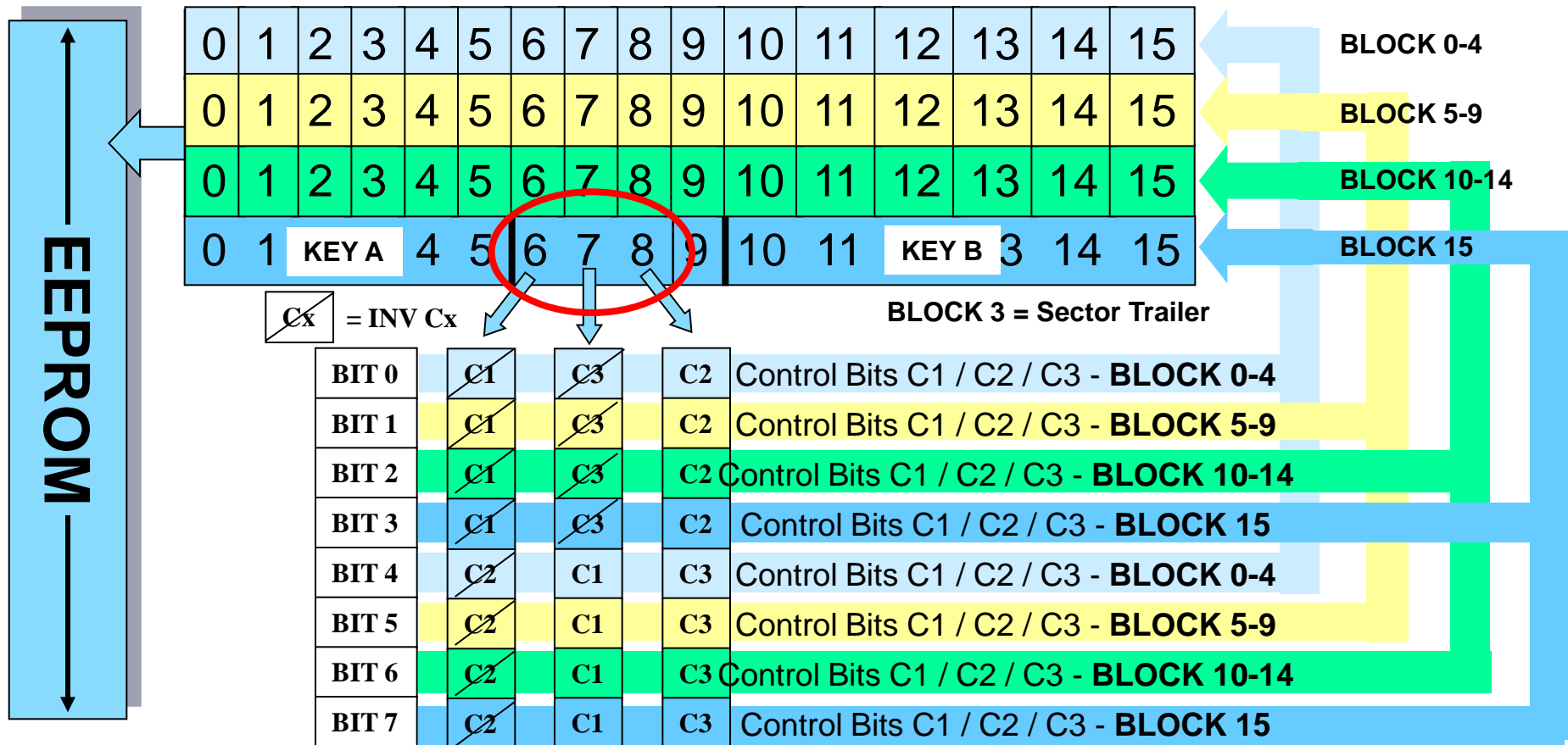
3 Control bits for each Sector Trailer

Sector Trailer (Block 3 or 15)

(WRITE: Access Condition Code for next Authentication)

`0 1 2 3 4 5 | 6 7 8 | 9 | 10 11 12 13 14 15`

---- = n.a.

KEY A READ	KEY A WRITE	C1	C2	C3	KEY B READ	KEY B WRITE	AC READ	AC WRITE	
-----	KEY A	0	0	0	KEY A	KEY A	KEY A	-----	
-----	-----	0	1	0	KEY A	-----	KEY A	-----	
-----	KEY B	1	0	0	-----	KEY B	KEY A or B	-----	
-----	-----	1	1	0	-----	-----	KEY A or B	-----	
-----	KEY A	0	0	1	KEY A	KEY A	KEY A	KEY A	*
-----	KEY B	0	1	1	-----	KEY B	KEY A or B	KEY B	
-----	-----	1	0	1	-----	-----	KEY A or B	KEY B	
-----	-----	1	1	1	-----	-----	KEY A or B	-----	

Condition Code Pattern

* Transport Access Condition

EEPROM

3 Control bits for each Data Block (each 5 Data Blocks)

Data Block - *Condition Code Pattern fixed in Block 3*

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

BLOCK ADDR

EEPROM

READ	WRITE	C1	C2	C3	INCR	DECR / TRANSFER / RESTORE	
KEY A or B	KEY A or B	0	0	0	KEY A or B	KEY A or B	**Transport Access Condition**
KEY A or B	-----	0	1	0	-----	-----	
KEY A or B	KEY B	1	0	0	-----	-----	**---- = n.a.**
KEY A or B	KEY B	1	1	0	KEY B	KEY A or B	
KEY A or B	-----	0	0	1	-----	KEY A or B	
KEY B	KEY B	0	1	1	-----	-----	
KEY B	-----	1	0	1	-----	-----	
-----	-----	1	1	1	-----	-----	

Condition Code Pattern
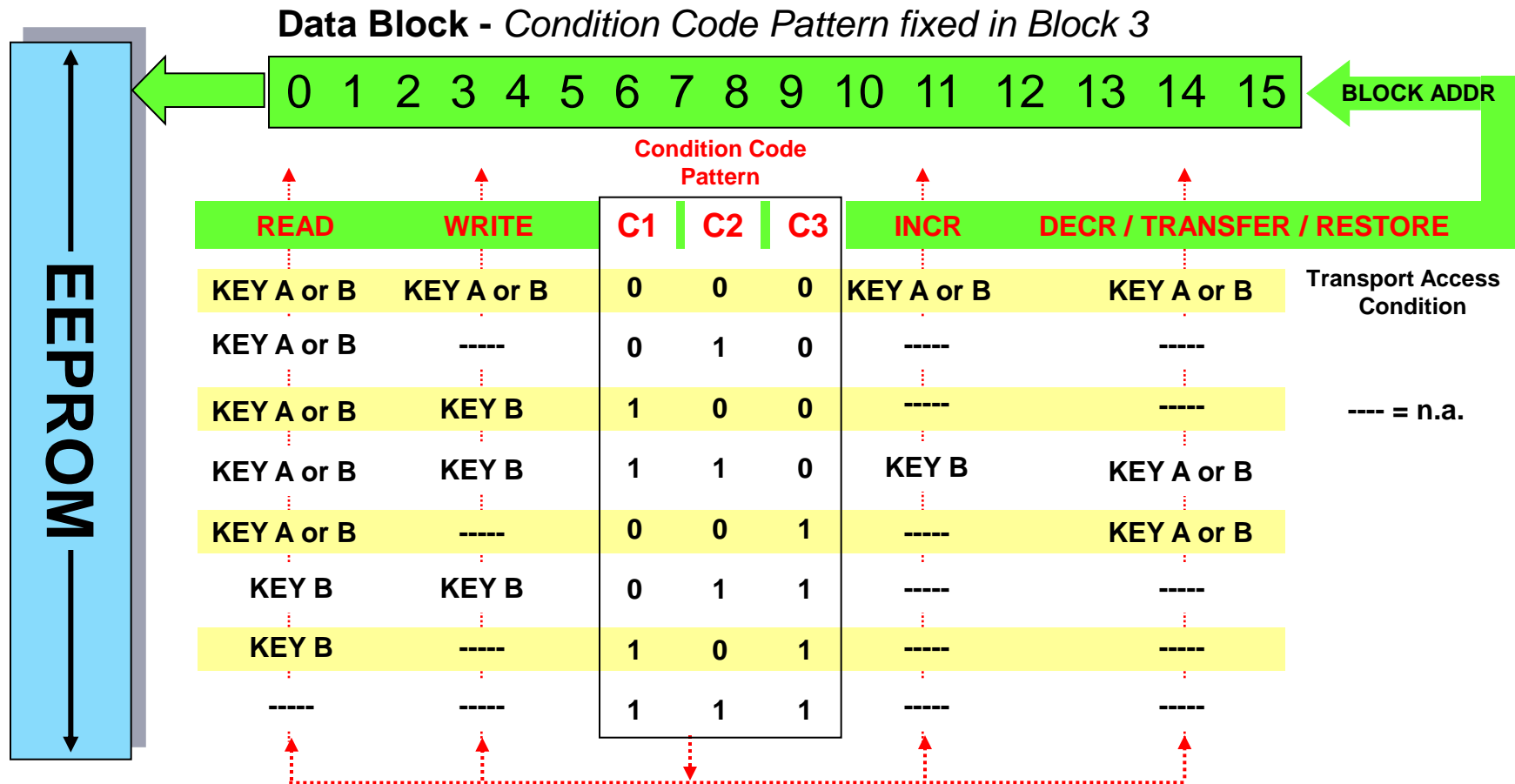
Value Block Format

Format of "Value Block" for electronic purse and / or anti-tearing function:

0 1 2 3	4 5 6 7	8 9 10 11	12	13	14	15
Value	$\overline{\text{Value}}$	Value	Adr	$\overline{\text{Adr}}$	Adr	$\overline{\text{Adr}}$

Commands:

- **READ**
- **WRITE**
- **INCREMENT**
- **DECREMENT**
- **RESTORE**
- **TRANSFER**

- **Fixed block- data format (generation via WRITE)**
- **Automatic value error detection & correction**

"Value": stored 3 times in 32-bit signed 2's complement (LSB first)

"Adr": stored 4 times in 8-bit numbers - altered only via WRITE

Value Operations

Operand

0 1 2 3	4 5 6 7	8 9 10 11	12	13	14	15	SOURCE
Value	Value	Value	Adr	Adr	Adr	Adr	ADDRESS

SOURCE Value (3xsigned 4-byte)

ALU

← DECREMENT **Operand**

INCREMENT **Operand**

Byte 0 … 15 of Block (within same Sector)

RESULT BUFFER

0 1 2 3	4 5 6 7	8 9 10 11	12	13	14	15	DESTINATION
Value	Value	Value	Adr	Adr	Adr	Adr	ADDRESS

RESULT Value (3xsigned 4-byte)

TRANSFER

0 1 2 3	4 5 6 7	8 9 10 11	12	13	14	15	DESTINATION
Value	Value	Value	Adr	Adr	Adr	Adr	ADDRESS

"COPY" Value (3xsigned 4-byte)

RESTORE

State diagram Security Level 1 (simplified)

„MIFARE Classic"

Reset SL1 → Idle / Halt

Idle / Halt → (REQA, (WUPA)) → Ready 1 / Ready 1*

Anticollision CL1 (loop on Ready 1 / Ready 1*)

Ready 1 / Ready 1* → (Select CL1) → Ready 2 / Ready 2*

Anticollision CL2 (loop on Ready 2 / Ready 2*)

Ready 2 / Ready 2* → (Select CL2) → Active / Active* State

MIFARE Commands, MIFARE Re-Auth, AES-Auth w. SL1 Card Auth Key

Active / Active* State → (MIFARE Auth) → Security Level 1 „MIFARE Classic"

14443-4

Active / Active* State → (RATS (+PPS)) → Security Level 1 (ISO/IEC 14443-4)

Auth Originality Key (loop on Security Level 1 (ISO/IEC 14443-4))

Security Level 1 (ISO/IEC 14443-4) → (Auth Level 2 or 3 Switch Key) → Reset SL2 or SL3

Notes:
- Card Activation acc. to ISO/IEC 14443
- PPS is optional
- Error behaviour as in ISO/IEC 14443

Commands additional to „MIFARE Classic" in SL1

▸ **Authenticate** with SL1 Authentication Key ⟶ **Proofs valid card!**

▸ **RATS** ⟶ Activates T=CL protocol.

▸ **PPS** ⟶ Switches to higher bit rates.

▸ **Authenticate** with Originality Key ⟶ Proofs NXP!

▸ **Authenticate** with Level 2 Switch Key ⟶ Switches to SL2.

▸ **Authenticate** with Level 3 Switch Key ⟶ Switches to SL3.

RF-Reset!

Authenticate with SL1 Authentication Key: Why?

Existing System, using the **same** MIFARE Classic Card

Application A

Reader A
Reader A
Reader A

Application B

Reader B
Reader B
Reader B

Application C

Reader C
Reader C
Reader C
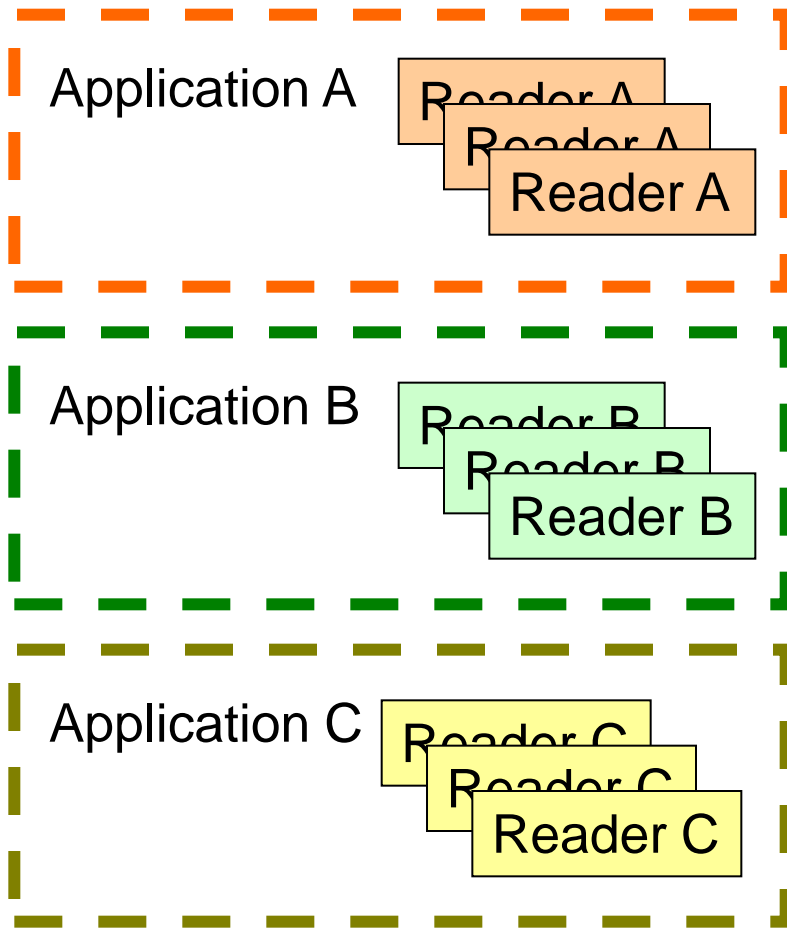
Application A and B:
- No need for Security.
- (Readers cannot be upgraded.)

Application C:
- Readers can be upgraded.
- Need for Security.

Authenticate with SL1 Authentication Key: Why?

Option 1:

▸ **MIFARE Plus SL2 or 3**

▸ Advantages:
  – Better Security.

▸ Disadvantage:
  – **All readers** must be upgraded.
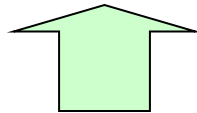
Recommendation!

Option 2:

▸ **MIFARE Plus SL1 + Authenticate with SL1 Authentication Key**

▸ Advantages:
  – Only **few readers** must be upgraded. (all those, where the application needs more security)

▸ Disadvantage:
  – Limited Security Improvement.
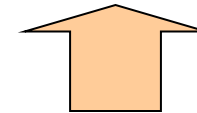
Cheaper solution…

Authenticate with SL1 Authentication Key: Option 2

Existing System, using the **same** MIFARE Plus Card in SL1

Application A

Reader A

Reader A

Reader A

Application B

Reader B

Reader B

Reader B

Application C

Reader C

Reader C

Reader C

No change in the system, but no more security.

Application A and B:
• No need for Security.
• (Readers cannot be upgraded.)

Application C:
Readers using Authenticate w. **SL1 Authentication Key**.
➡ Better Security
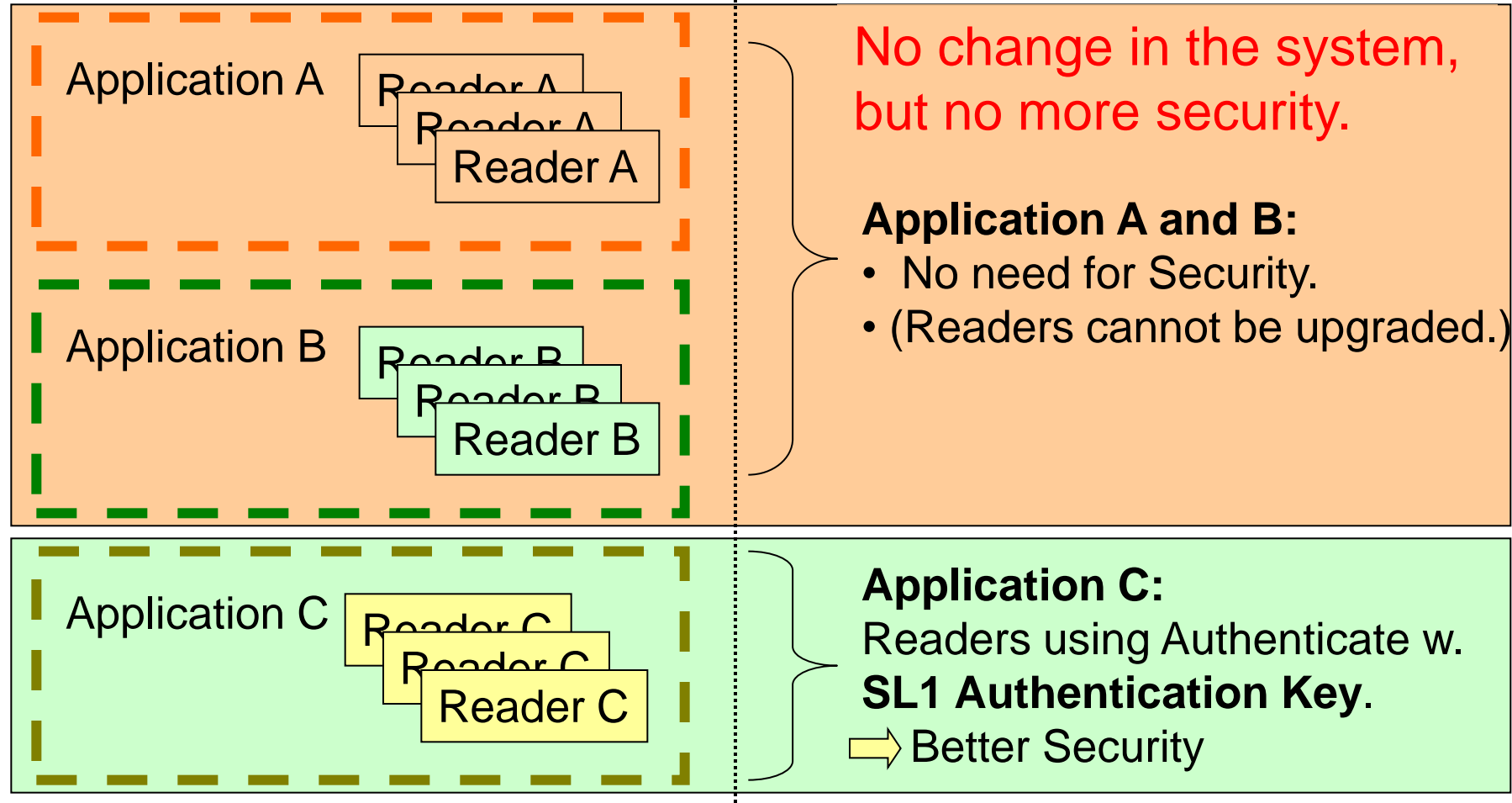
MIFARE Plus Security Level 2

 AES + use of MIFARE Crypto

MIFARE Plus Security Level 2

SL3
(AES)

SL2
(MF Crypto + AES)

SL1
(„MIFARE Classic")

SL0
(Personalisation)
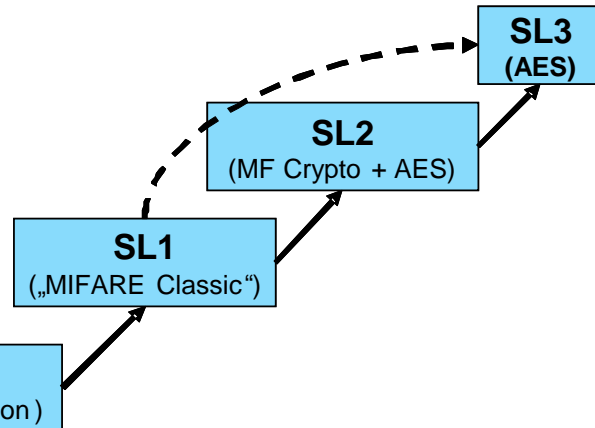
AES + use of MIFARE Crypto

Security Level 2 offers the features of the well known MIFARE Classic:

- **MIFARE Authentication / Encryption**
  - Same as MF1 ICS 50 or MF 1ICS70
  - **Consider security risks!**
- **MIFARE Read / Write**
  - Same as MF1 ICS 50 or MF 1ICS70
- **MIFARE Value operations**
  - Increment / Decrement / Restore + Transfer
  - Same as MF1 ICS 50 or MF 1ICS70

Previous AES authentication required!

State diagram Security Level 2 (simplified)
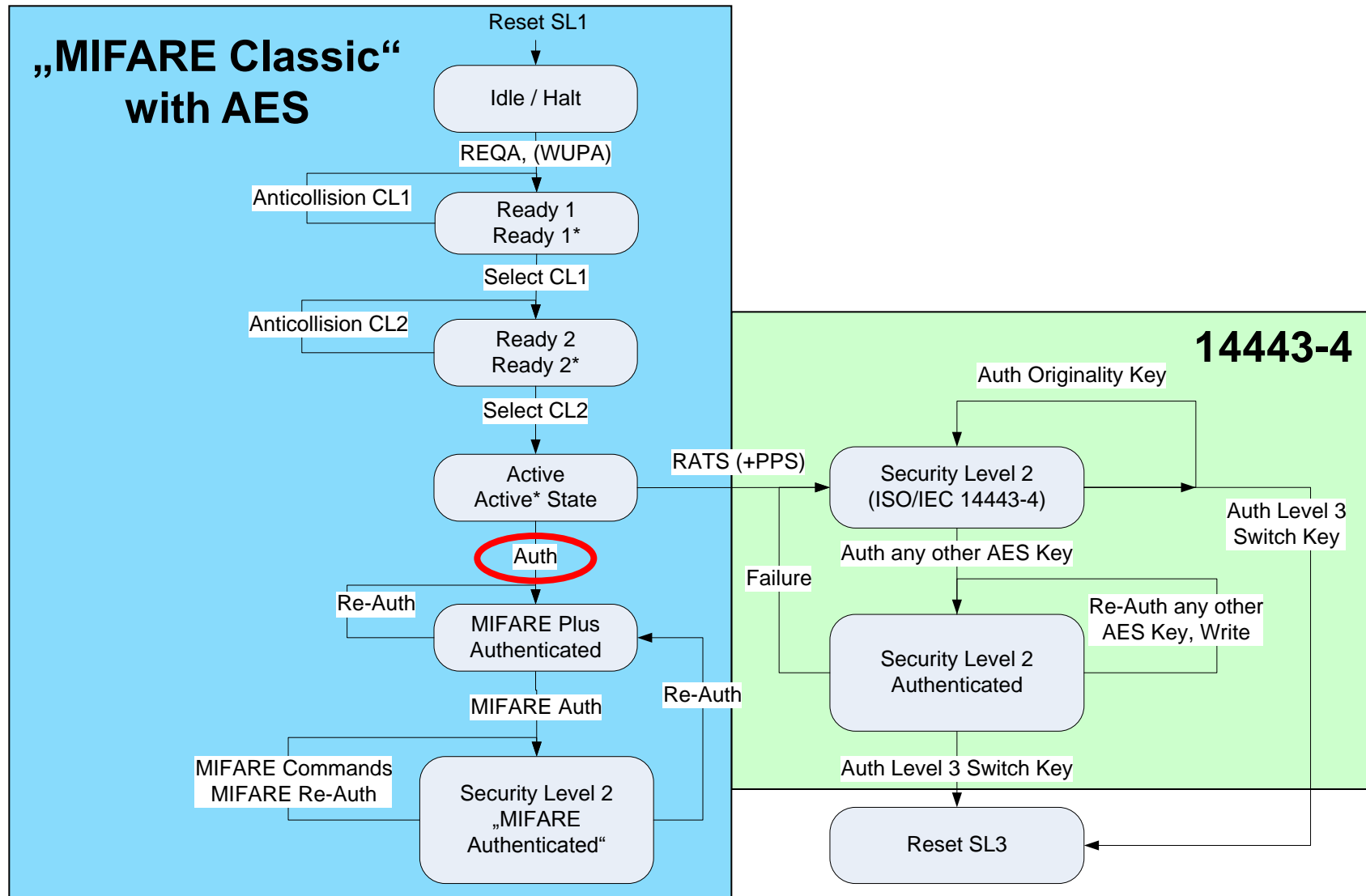
MIFARE Plus SL2 Authentication

PCD   PICC

1	2
CMD 0x76	KeyBNr

PCD to PICC →   Generate 16 byte RndB

Generate 16 byte RndA

1	16
0x90	E(Kx, RndB)

← PICC to PCD

RndB' = (RndB rotate left 1 byte)

1	32
CMD 0x72	E(Kx, RndA || RndB')

PCD to PICC →

Verify RndB'
RndA' = (RndA rotate left 1 byte)

1	16
0x90	E(Kx, RndA')

Verify RndA'   ← PICC to PCD

Result ↓   Result ↓

16	16		16	16
----	----		----	----
RndA	RndB		RndA	RndB

Generate the Session MIFARE Key, part 1
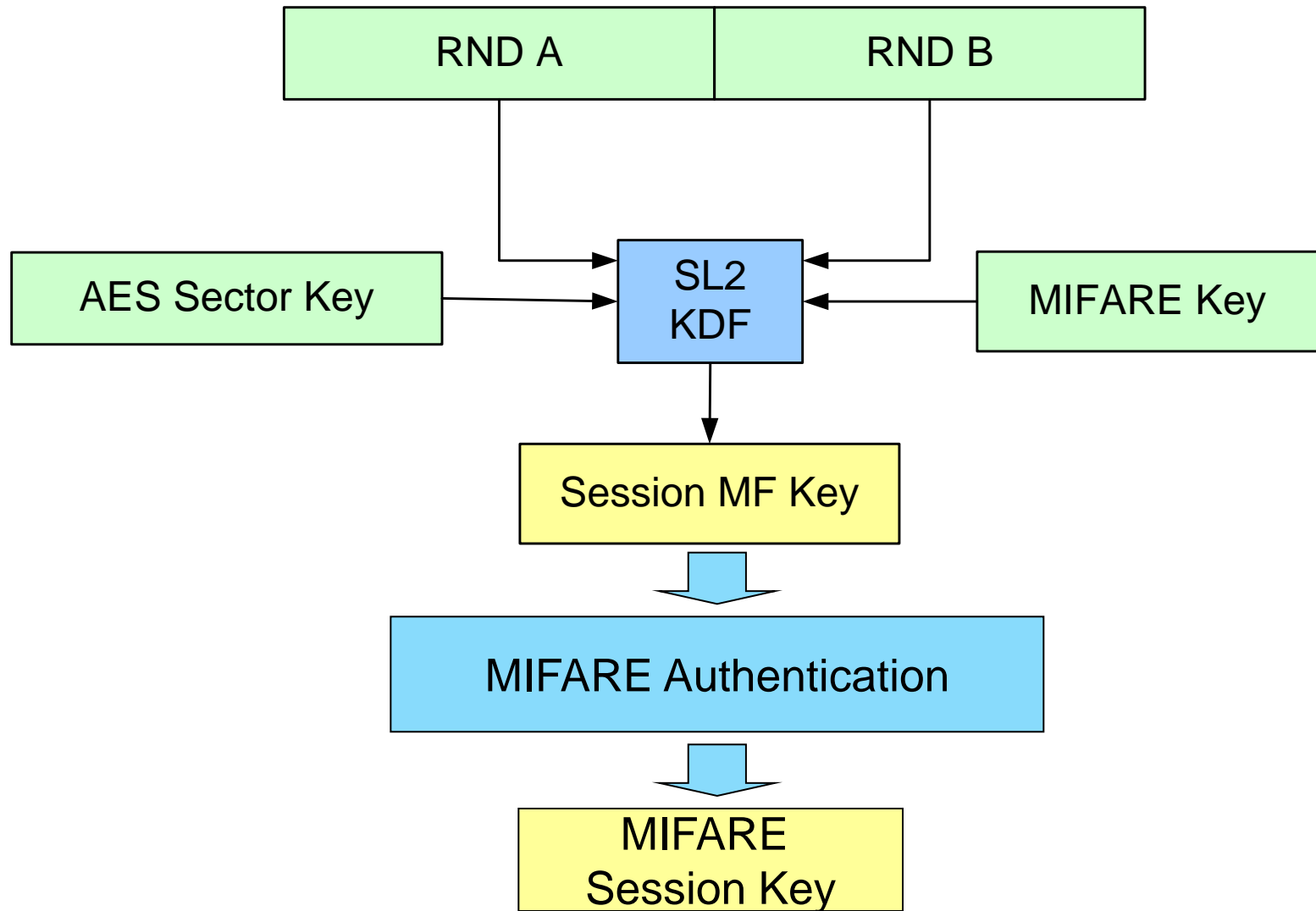
Generate the Session MIFARE Key, part 2
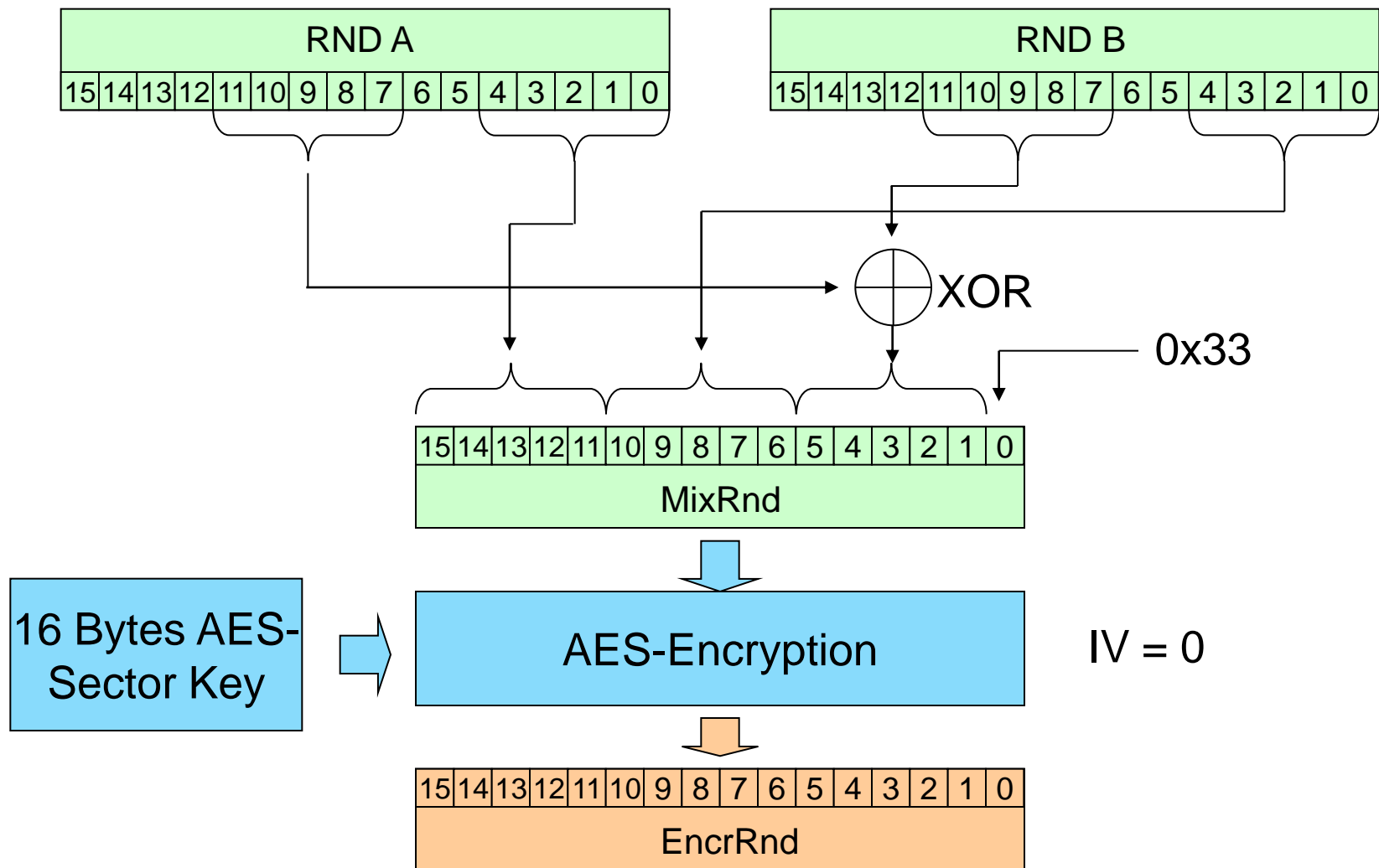
Generate the Session MIFARE Key, part 3
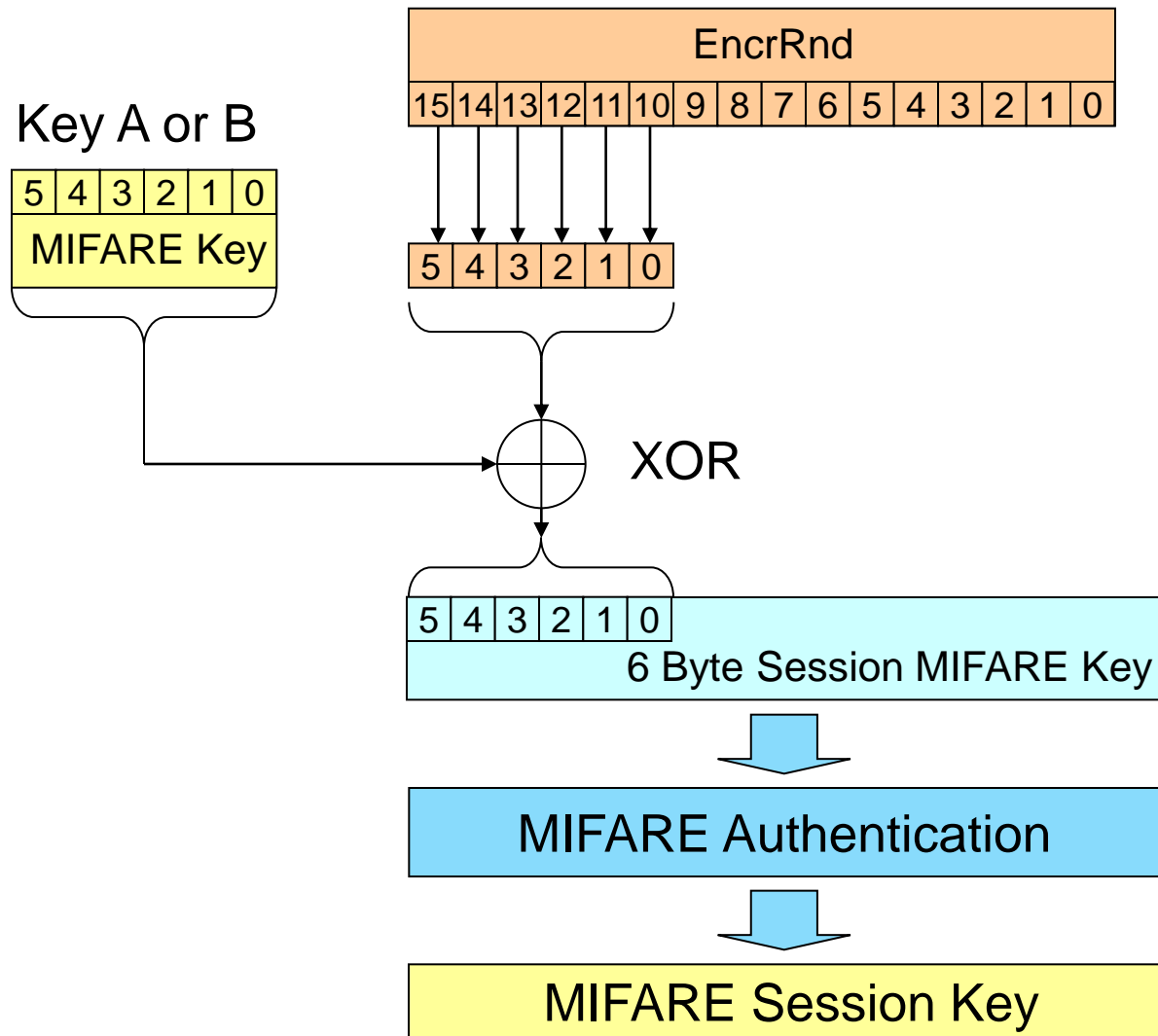
Multi Sector Authentication

▸ If the AES Key #X = AES Key #Y, no new AES authentication is required.

▸ If the *also* MIFARE Classic Key #X = MIFARE Classic Key #Y no new MIFARE Classic authentication is needed.
  – If AES Key #X <> AES Key #Y, both new AES and MIFARE Classic authentication is required.

▸ Sector X and Sector Y can but do not have to be consecutive.

▸ Key type must be the same (A or B).

Remarks:
• Authentication is done with Key #X, so Key #Y can be changed without loosing the authentication.
• If all keys are the same, the card can e.g. be read with one authentication only.

Commands additional to „MIFARE Classic" in SL2

Multi Block Read ⟶ Allows to read 2 or 3 blocks at once.

Multi Block Write ⟶ Allows to write 2 or 3 blocks at once.

RATS ⟶ Activates T=CL protocol.

PPS ⟶ Switches to higher bit rates.

Authenticate with Originality Key ⟶ Proofs NXP!

Authenticate with Level 3 Switch Key ⟶ Switches to SL3.

Additional commands: MultiBlockRead

MIFARE Read

```
  1     1
┌─────┬─────┐
│CMD  │     │
│0x30 │ BNr │        PCD to PICC ──▶          Reading 1 Block
└─────┴─────┘
                              16
             ◀── PICC to PCD        ┌────────┐
                                    │  DATA  │
                                    └────────┘
```

Multi Block Read

```
  1     1     1
┌─────┬─────┬─────┐
│CMD  │     │     │
│0x38 │ BNr │  n  │     PCD to PICC ──▶        Reading n Blocks
└─────┴─────┴─────┘
                              32
         ◀── PICC to PCD   ┌──────────────┐
                           │     DATA     │
                           └──────────────┘
```
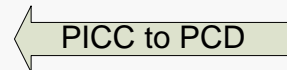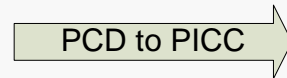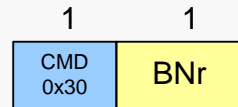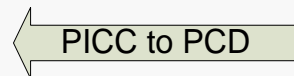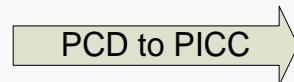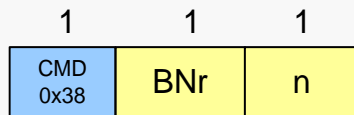
Notes: n = 1…3
        No Multi Sector Read!
        Multi Block Read does not include the Sector Trailer.
        For n>1: Framesize must be at least 32 resp. 48 Bytes.

Additional commands: MultiBlockWrite

MIFARE Write

1	1
CMD 0xA0	BNr

PCD to PICC →

Writing 1 Block

← PICC to PCD   ACK

16
Data

PCD to PICC →

← PICC to PCD   ACK

Multi Block Write

1	1	1
CMD 0x38	BNr	n

PCD to PICC →

Writing n Blocks

← PICC to PCD   ACK

32 / 48
Data

PCD to PICC →

← PICC to PCD   ACK
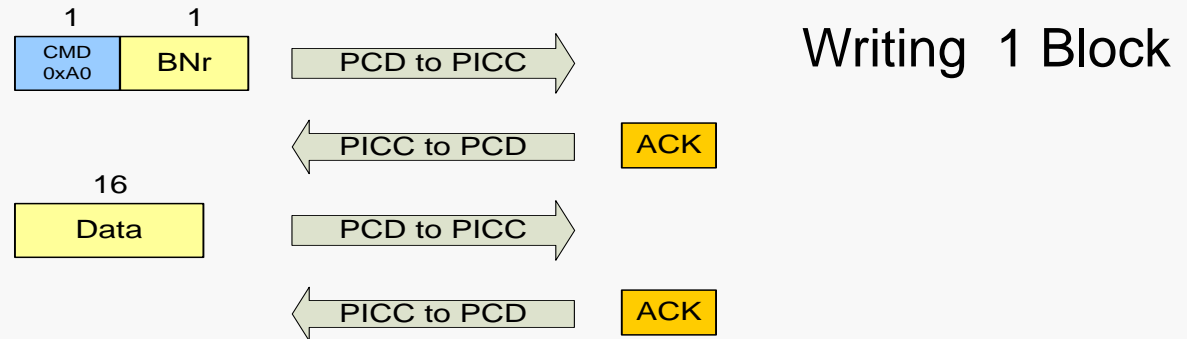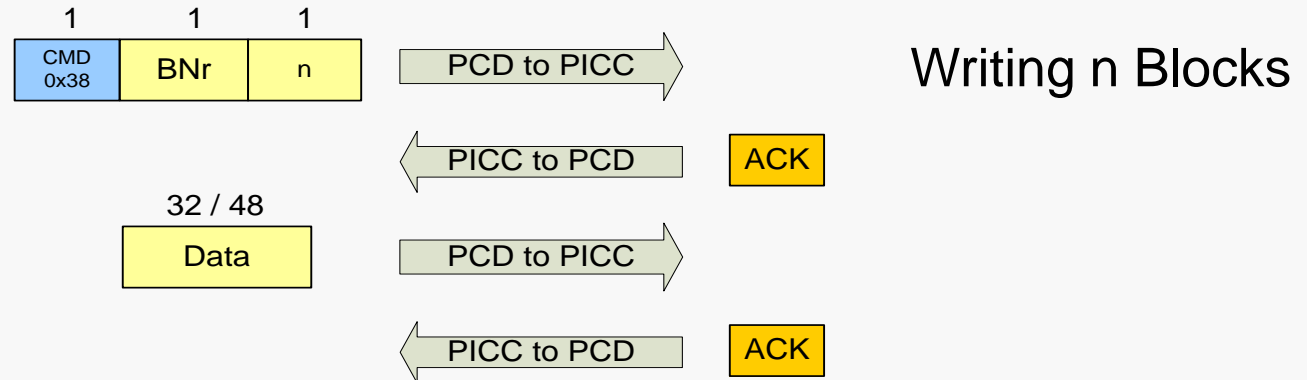
Notes: n = 1…3
No Multi Sector Write!
Multi Block Write does not include the Sector Trailer.
For n>1: Framesize must be at least 32 resp. 48 Bytes.

MIFARE Plus Security Level 3

Use of AES and T=CL protocol

MIFARE Plus Security Level 3

SL3
(AES)

SL2
(MF Crypto + AES)

SL1
(„MIFARE Classic")

SL0
(Personalisation)
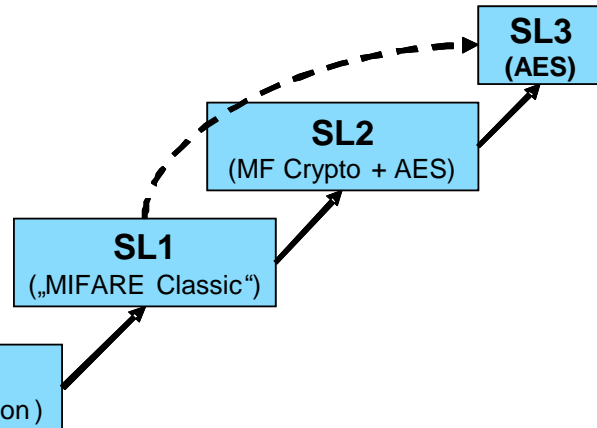
AES and T=CL protocol

Security Level 3 offers the following features :

- **AES Authentication / Encryption**
    - First Auth., Following Auth., Reset Auth.
- **Read / Write**
    - Encrypted or Plain, with or without MAC
- **Value operations**
    - Increment / Decrement / Restore + Transfer, with or without MAC
- Proximity Check
- Select Virtual Card

State diagram Security Level 3 (simplified)

ISO/IEC 14443 Card Activation

Reset SL1

Idle / Halt

REQA, (WUPA)

Anticollision CL1

Ready 1
Ready 1*

Select CL1

Anticollision CL2

Ready 2
Ready 2*

Select CL2

Active
Active* State

RATS (+PPS)

14443-4

Auth Originality Key

Security Level 3
(ISO/IEC 14443-4)

Auth any other AES Key

Failure

Re-Auth any other AES Key,
MIFARE Plus Commands

Security Level 3
Authenticated

Note: This state diagram is slightly different when RID is used. Then first the Virtual Card selection must be done after the protocol has been established.

Note: The auth with originality key will not work if proximity check is mandatory.
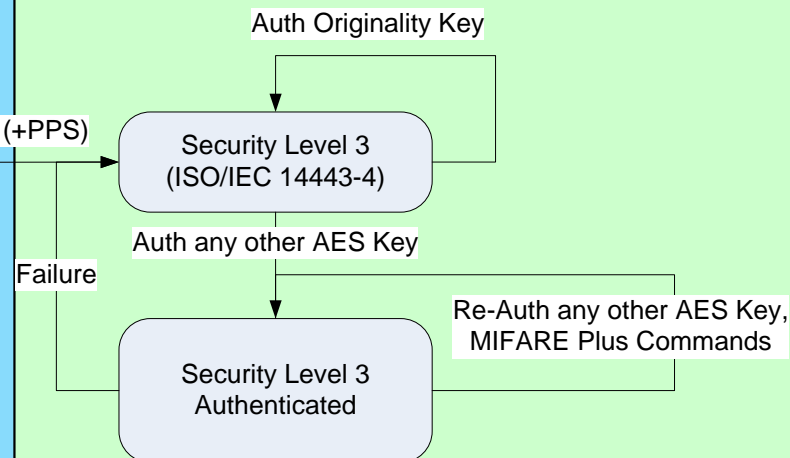
Anti tearing of AES keys in Security Level 3

▸ The MIFARE Plus provides an anti tearing mechanism for
  – the AES keys
  – the Sector Trailer

▸ When the update of an AES key or Sector Trailer is interrupted,
  – Either the old key is valid or the new one.
  – No check possible which key is written, so if updating is interrupted: try again.
  – The MIFARE Plus card needs up to appr. 25 ms after next POR (before REQA).

The PCD needs to „know", wether a roll-back of an interrupted key update takes place or not.

Sector structure in SL3

The same structure as in the previous SLs except the Sector Trailer:

BLOCK 3, resp. Block 15
"Sector Trailer"

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
Key A						AC			x	Key B						SL1 & SL2
unused						AC			x	unused						SL3

EEPROM
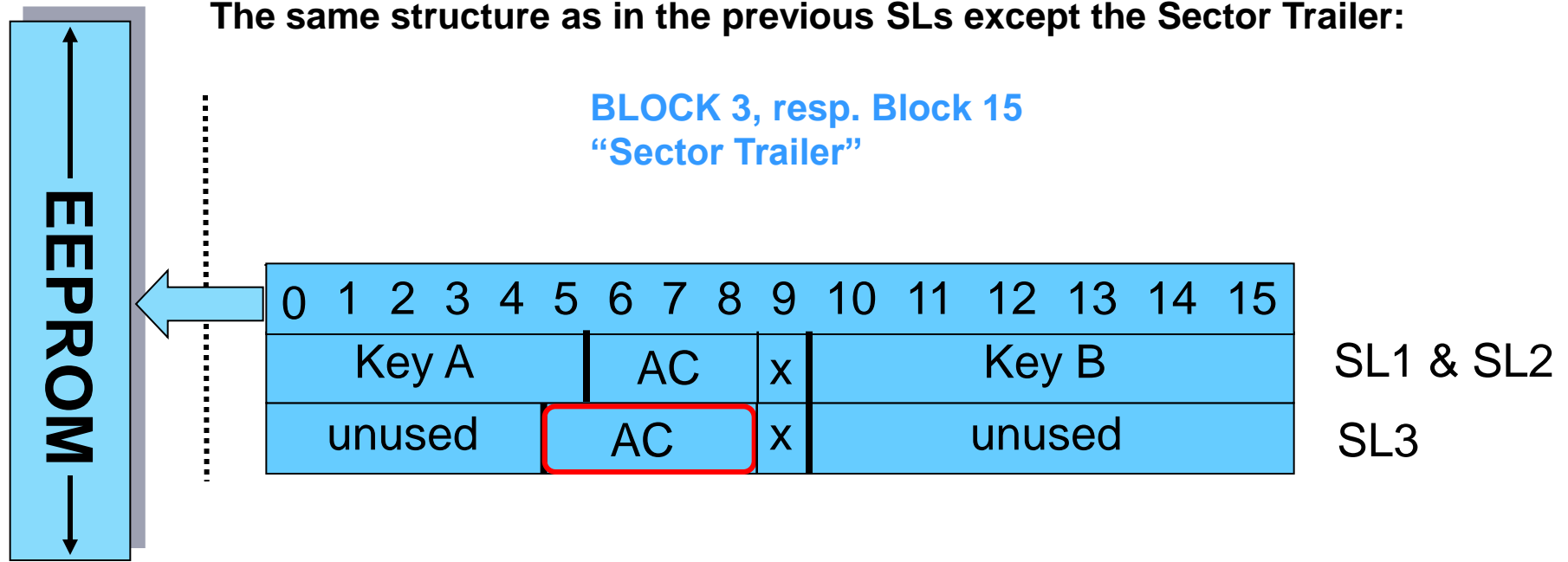
Bytes 6,7,8: Same as in SL1
Byte 5 defines whether plain communication is allowed or not.

0x0F: Plain communication allowed for all blocks.

AC coding for plain communication (Sector 0..31)

Byte 5 of Block 3

7	6	5	4	3	2	1	0		
0				1				Plain allowed	Block 3
1				0				No Plain allowed	Block 3
	0				1			Plain allowed	Block 2
	1				0			No Plain allowed	Block 2
		0				1		Plain allowed	Block 1
		1				0		No Plain allowed	Block 1
			0				1	Plain allowed	Block 0
			1				0	No Plain allowed	Block 0

AC coding for plain communication (Sector 32..39)

Byte 5 of Block 15

7	6	5	4	3	2	1	0		
0				1				Plain allowed	Block 15
1				0				No Plain allowed	Block 15
	0				1			Plain allowed	Block 10..14
	1				0			No Plain allowed	Block 10..14
		0				1		Plain allowed	Block 5..9
		1				0		No Plain allowed	Block 5..9
			0				1	Plain allowed	Block 0..4
			1				0	No Plain allowed	Block 0..4

Switch to SL3 -> AC coding for plain communication

▸ The Byte 5 is used as MIFARE Key Byte in SL1 and SL2.

▸ The MFP Configuration Block contains the **Default AC coding of Byte 5**.

▸ The Default AC of Byte 5 is copied from the MFP Configuration Block into each Sector Trailer during Level Switch.

▸ Make sure that the right AC for plain communication is set during Personalisation! (Only possible in SL0!)

▸ Default is 0x0F (plain allowed in every block).

Transaction & Session
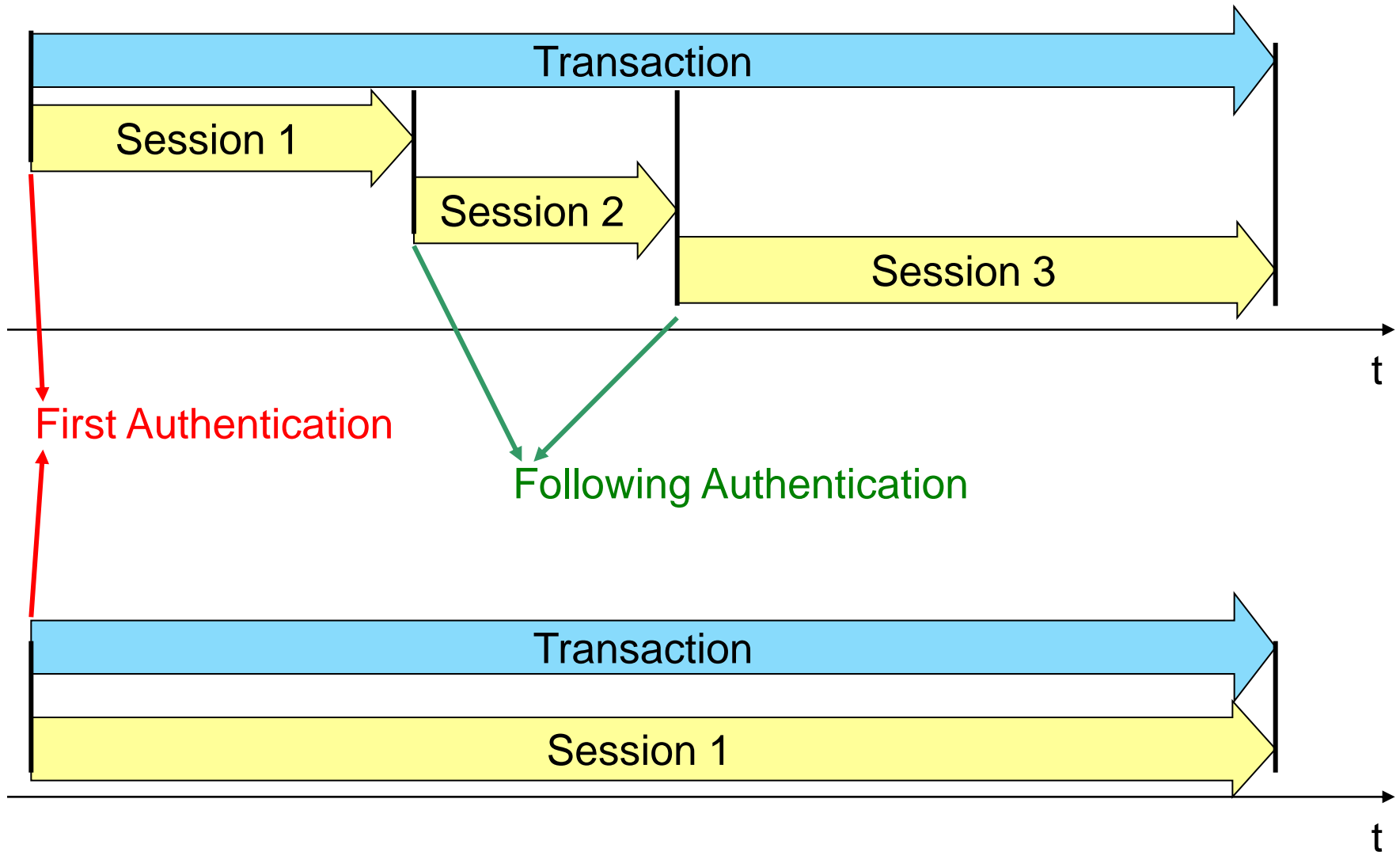
AES Authentication for MIFARe Plus SL3

▸ **Authentication** (general)
  – Is always required, and guarantees authenticity.
  – Is based on AES.
  – Starts a Session, and ends the previous Session (if available).
  – Generates 2(!) Session keys
  – Releases the Transfer buffer.

▸ **First Authentication**
  – Starts a transaction.
  – Generates a Transaction Identifier (TI).
  – Exchanges the PICC capabilities and the PCD capabilities.
  – Resets Read & Write Counter (R_Ctr & C_Ctr).

▸ **Following Authentication**
  – Does not reset the counters

A **Transaction** may contain several **Sessions**

First Authentication
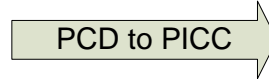
1	2	1	0 ... 6
70$_{hex}$	KeyBNr	Len Cap	PCDcap2

PCD to PICC →

Generate 16 byte RndB

1	16
90$_{hex}$	E(Kx, RndB)

Generate 16 byte RndA

RndB' = (RndB <<< 1)

← PICC to PCD

1	32
72$_{hex}$	E(Kx, RndA || RndB')

PCD to PICC →

Verify RndB'
RndA' = (RndA <<< 1)

1	32
90$_{hex}$	E(Kx, TI || RndA' || PICCcap2 || PCDcap2)

Verify RndA'

← PICC to PCD

Result

Result
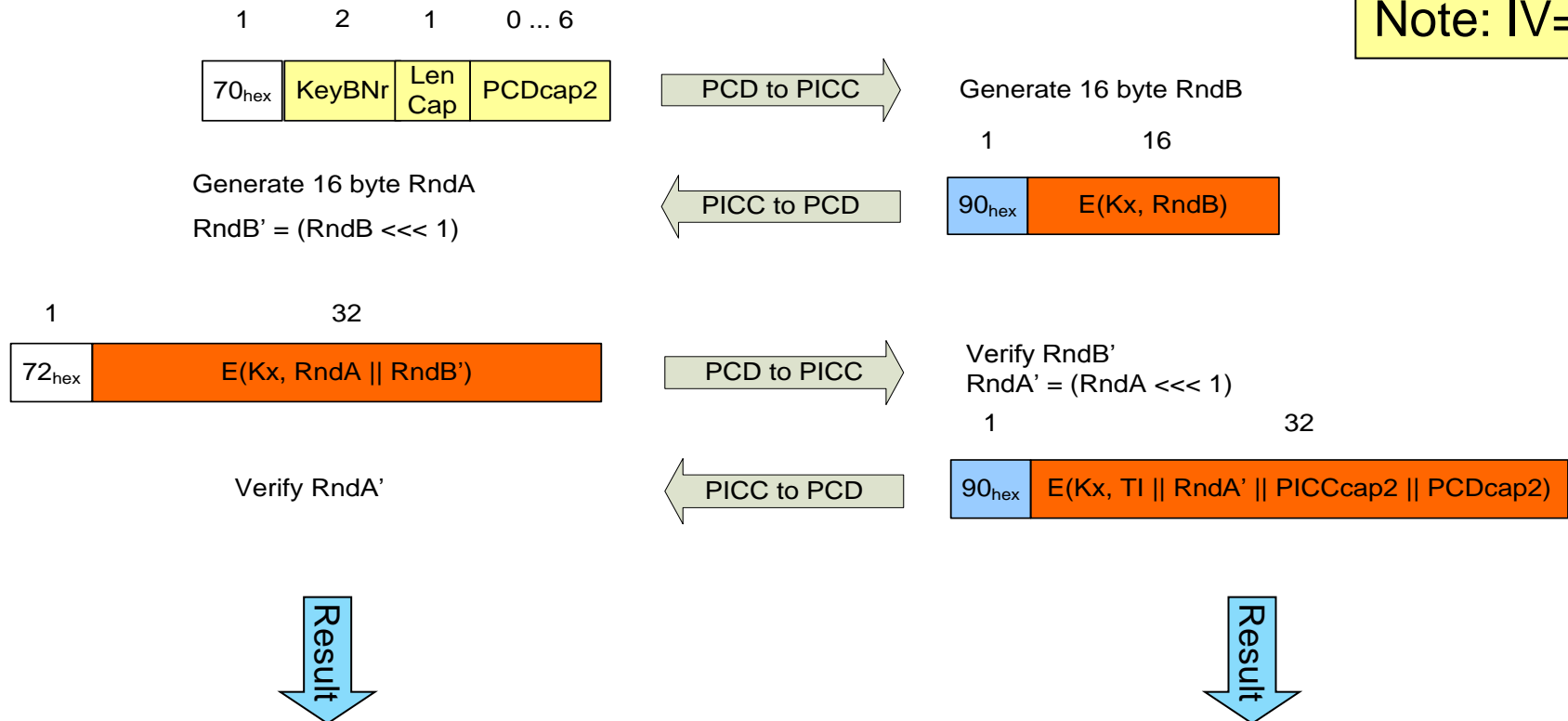
TI	= Transaction Identifier	4 Bytes	RND from PICC
RND A	= Random Number A	16 Bytes	RND from PCD
RND B	= Random Number B	16 Bytes	RND from PICC
PICCcap2	= PICC capabilities	6 Bytes	see next slides
PCDcap2	= PCD capabilities	0..6 Bytes	see next slides

PICC and PCD capabilities

▸ **PICCcap1 and PCDcap1 (3Bytes each)**

  – 6 Bytes: Details refer to the Virtual Card Selection

▸ **PCDcap2**

  – PCD capabilities: 6 Bytes
  – Defined by the system / reader

There is no use case for the PCDCapabilities now:
• Either use no PCDCap (LenCap = 0) or fill all bytes with 00.
• All „missing" bytes in the second response of authentication are padded with 00.
• The PCD **must** check the PCDCap in the second response!

▸ **PICCcap2**

  – PICC capabilities: 6 Bytes
  – Bytes 0..3 are defined by NXP (all bytes 00)
  – Bytes 4 and 5 are definable by the user (Configuration Block)
  – PCD must not check the PICCcap now. -> Will change later.

Following Authentication

```
      1        2
   ┌──────┬──────────┐
   │ 76hex│  KeyBNr  │
   └──────┴──────────┘
```

PCD to PICC →   Generate 16 byte RndB

```
                         1            16
                      ┌──────┬─────────────────┐
   PICC to PCD  ←      │ 90hex│  E(Kx, RndB)    │
                      └──────┴─────────────────┘
```

Generate 16 byte RndA

RndB' = (RndB <<< 1)

```
   1                    32
┌──────┬─────────────────────────────────┐
│ 72hex│     E(Kx, RndA || RndB')         │
└──────┴─────────────────────────────────┘
```

PCD to PICC →   Verify RndB'
RndA' = (RndA <<< 1)

```
                         1            16
                      ┌──────┬─────────────────┐
   PICC to PCD  ←      │ 90hex│  E(Kx, RndA')   │
                      └──────┴─────────────────┘
```

Verify RndA'

Result   Result
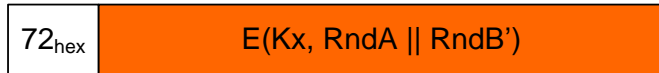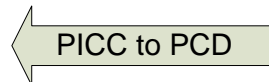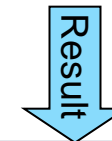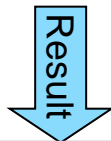
TI	= Transaction Identifier	4 Bytes		known
RND A	**= Random Number A**	**16 Bytes**		**RND from PCD**
RND B	**= Random Number B**	**16 Bytes**		**RND from PICC**
PICCcap2	= PICC capabilities	6 Bytes		known
PCDcap2	= PCD capabilities	6 Bytes		known

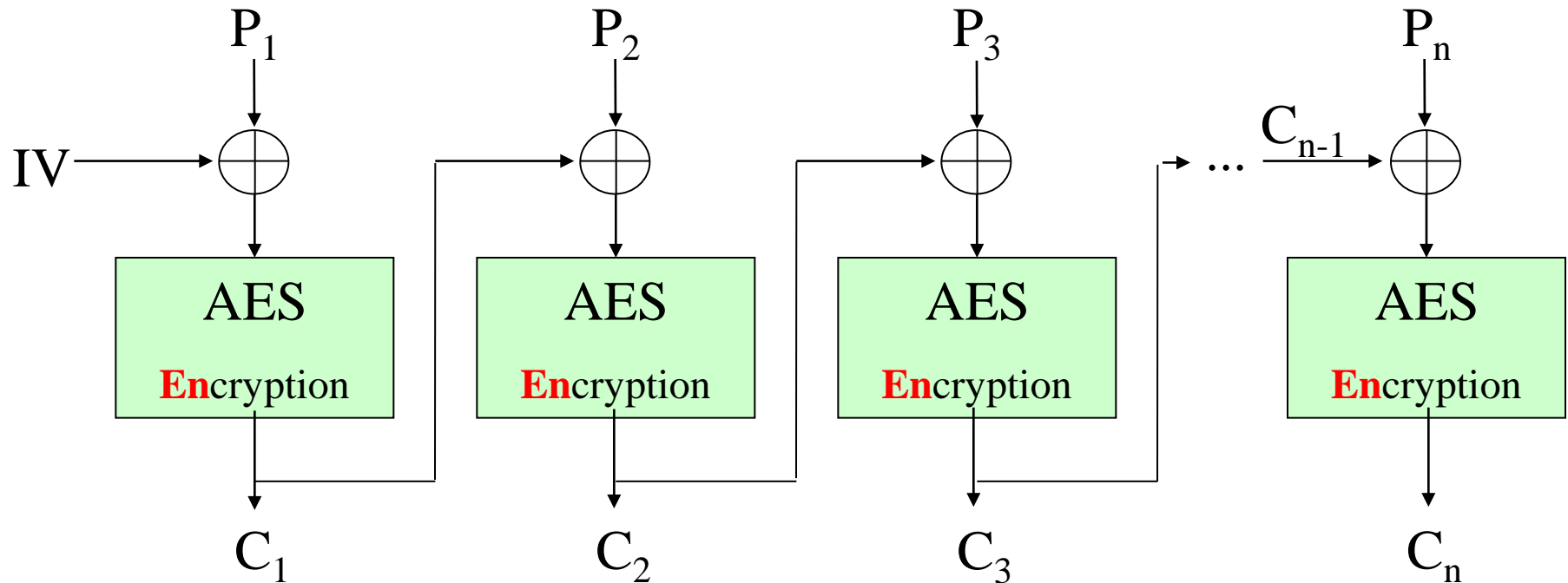# Generation of Session Key for Encryption
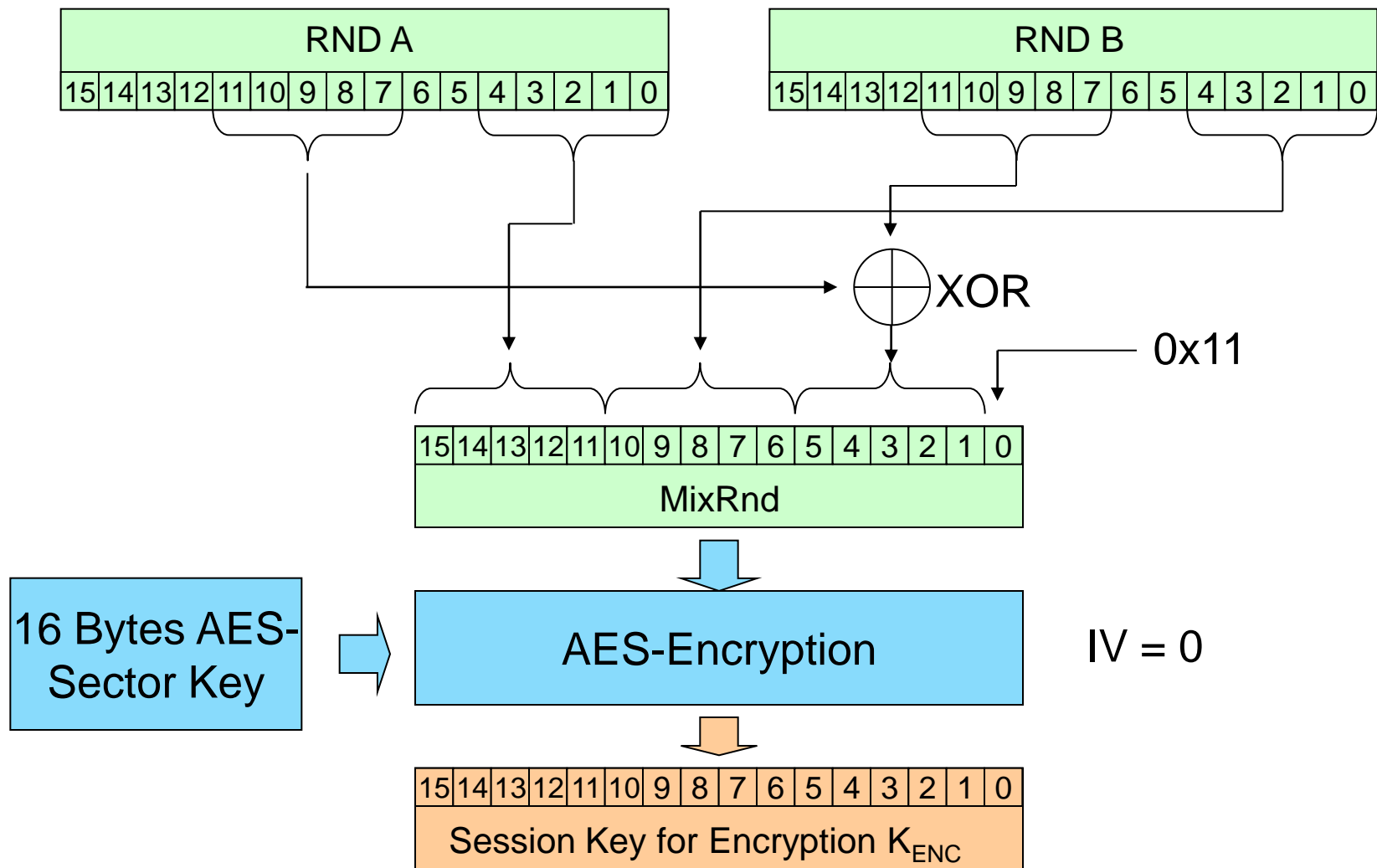
Generation of Session Key for calculating Message Authentication Code
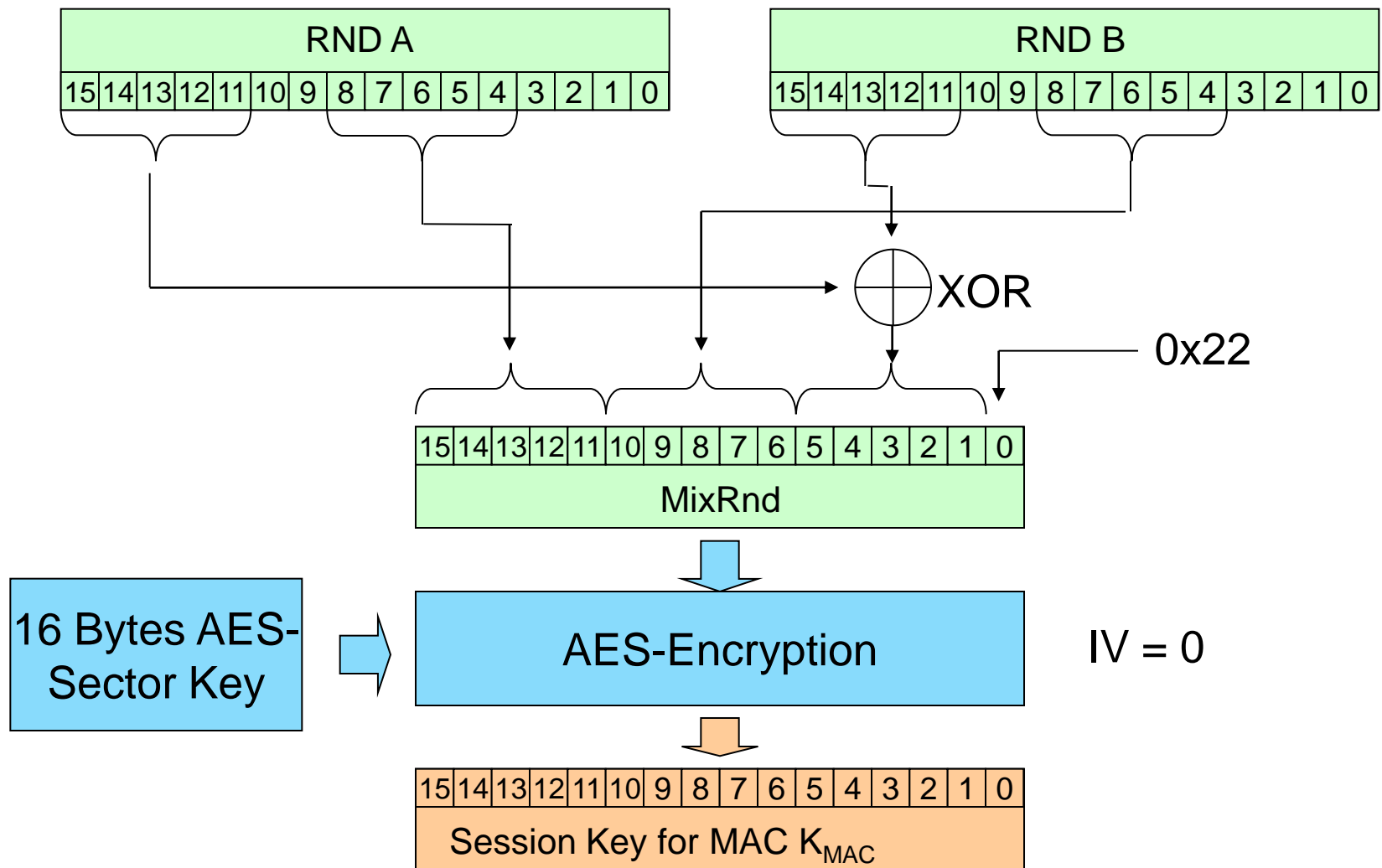
Confidentiality: MIFARE Plus AES Encryption

If plain data is more than 16 bytes long, chaining according to standard CBC mode

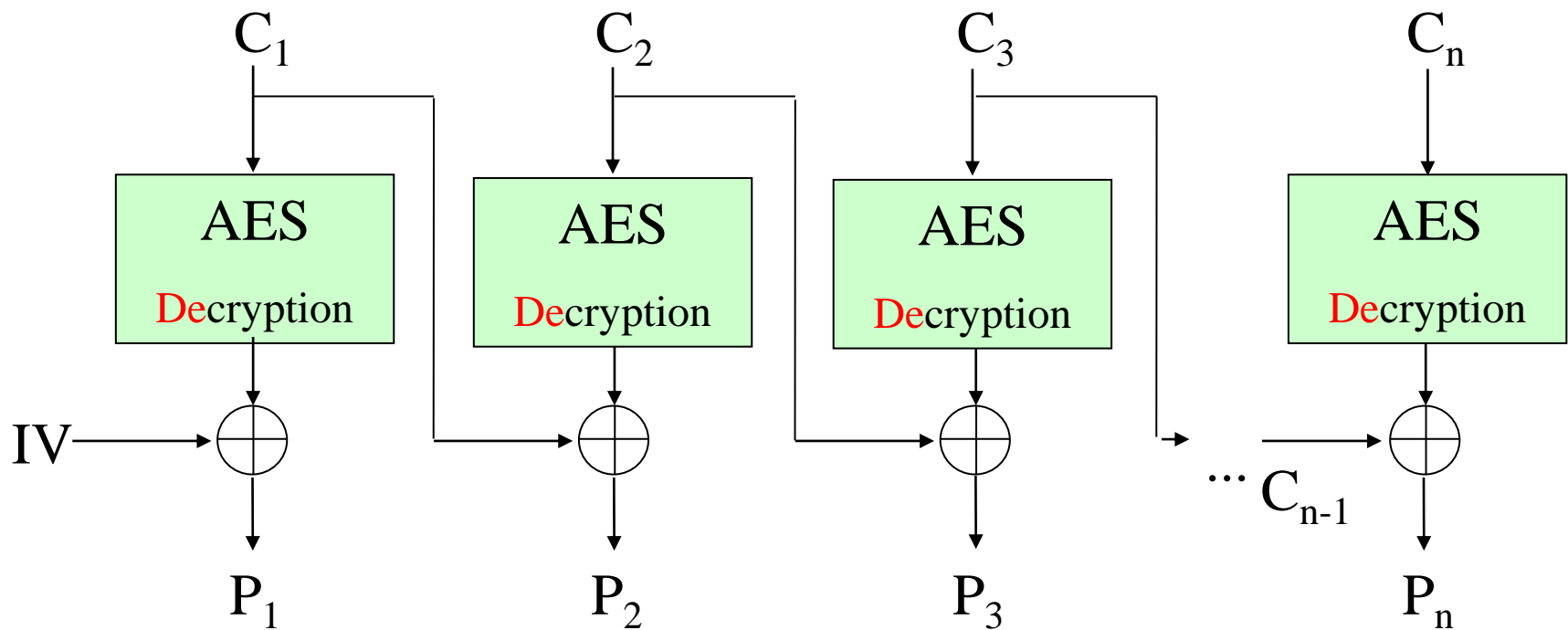| P$_n$: Plain Block (16 bytes) | C$_n$: Ciphered Block (16 bytes) |
| $\oplus$: addition modulo 2 ("XOR") | IV : Init vector 16 bytes |

IV:

4	2	2	2	2	2	2
Transaction Identifier	R_Ctr	W_Ctr	R_Ctr	W_Ctr	R_Ctr	W_Ctr

Note: R_Ctr and W_Ctr -> LSB first
TI is a bytestring, so LSB/MSB does not apply.

MIFARE Plus, CONFIDENTIAL 68
Renke Bienert, BUID/CAS   March 11, 2011

Confidentiality: MIFARE Plus AES Decryption

C_1 $\quad$ C_2 $\quad$ C_3 $\quad$ C_n

| AES Decryption | AES Decryption | AES Decryption | AES Decryption |

IV → ⊕ → P_1 $\quad$ ⊕ → P_2 $\quad$ ⊕ → P_3 $\quad$ $\cdots$ C_{n-1} $\quad$ ⊕ → P_n

P_n : Plain Block (16 bytes)	C_n : Ciphered Block (16 bytes)
⊕ : addition modulo 2 ("XOR")	IV : Init vector 16 bytes

IV:

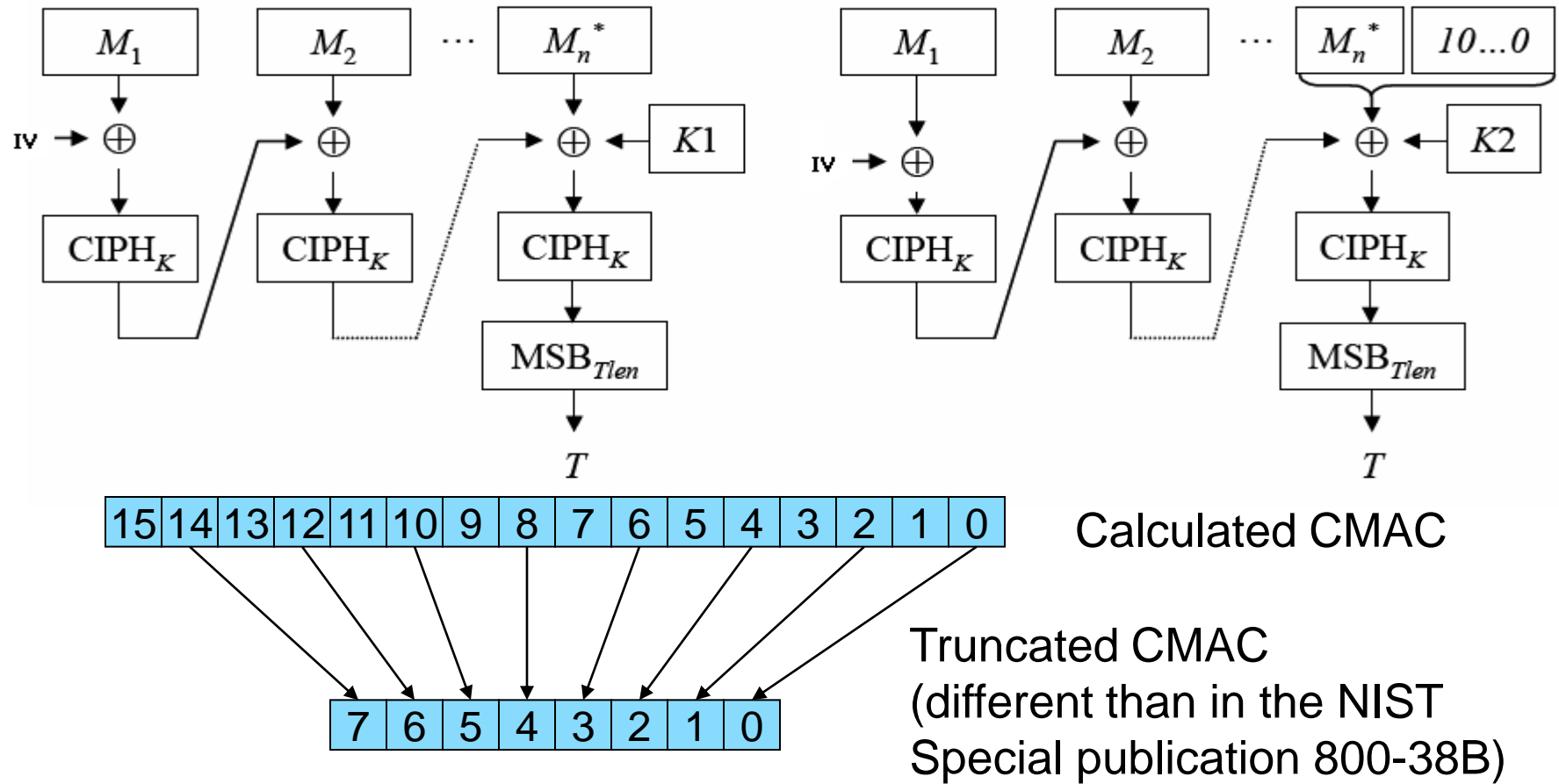| 4	2	2	2	2	2	2
Transaction Identifier	R_Ctr	W_Ctr	R_Ctr	W_Ctr	R_Ctr	W_Ctr

**Note:** R_Ctr and W_Ctr -> LSB first
TI is a bytestring, so LSB/MSB does not apply.

Integrity: MAC

CMAC: According to NIST Special Publication 800-38B

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Calculated CMAC

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Truncated CMAC
(different than in the NIST
Special publication 800-38B)

Read Counter R_Ctr

- Format:
  - 2 Byte counter (integer),
  - LSByte first ("Little Endian")

- The counter values are never transferred.

- Read Counter counts the Read commands.
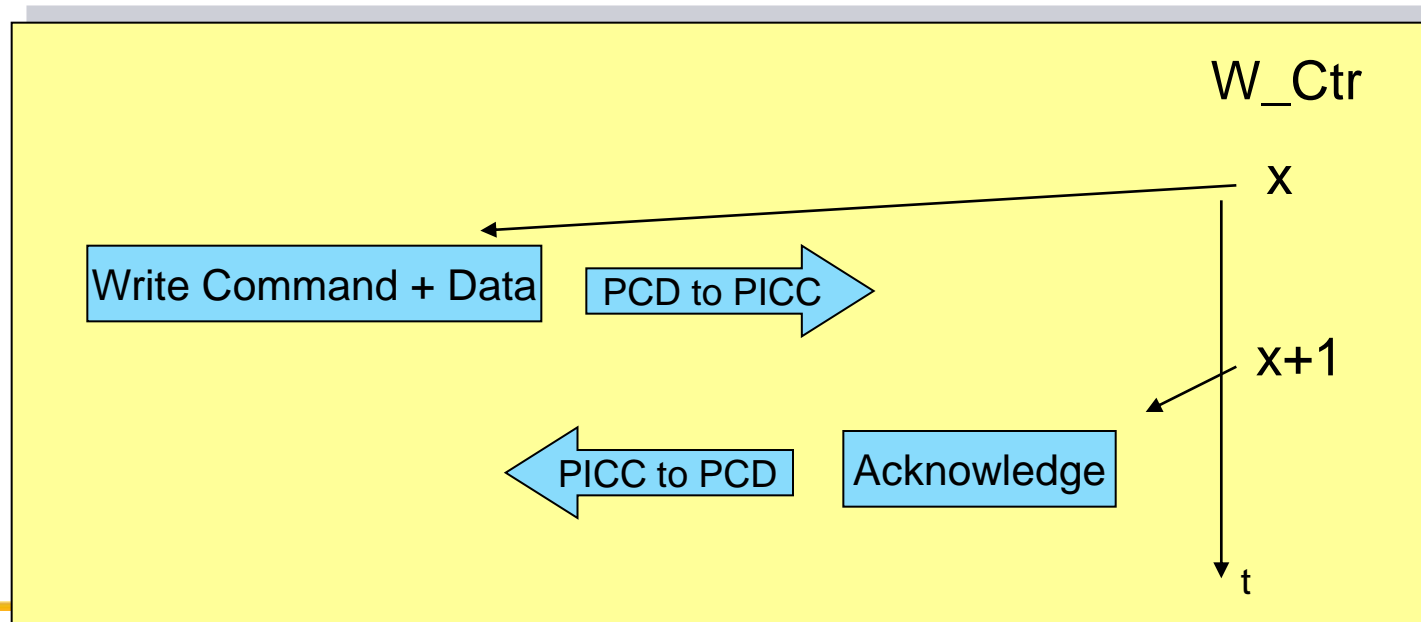
- First Authentication resets the R_Ctr.
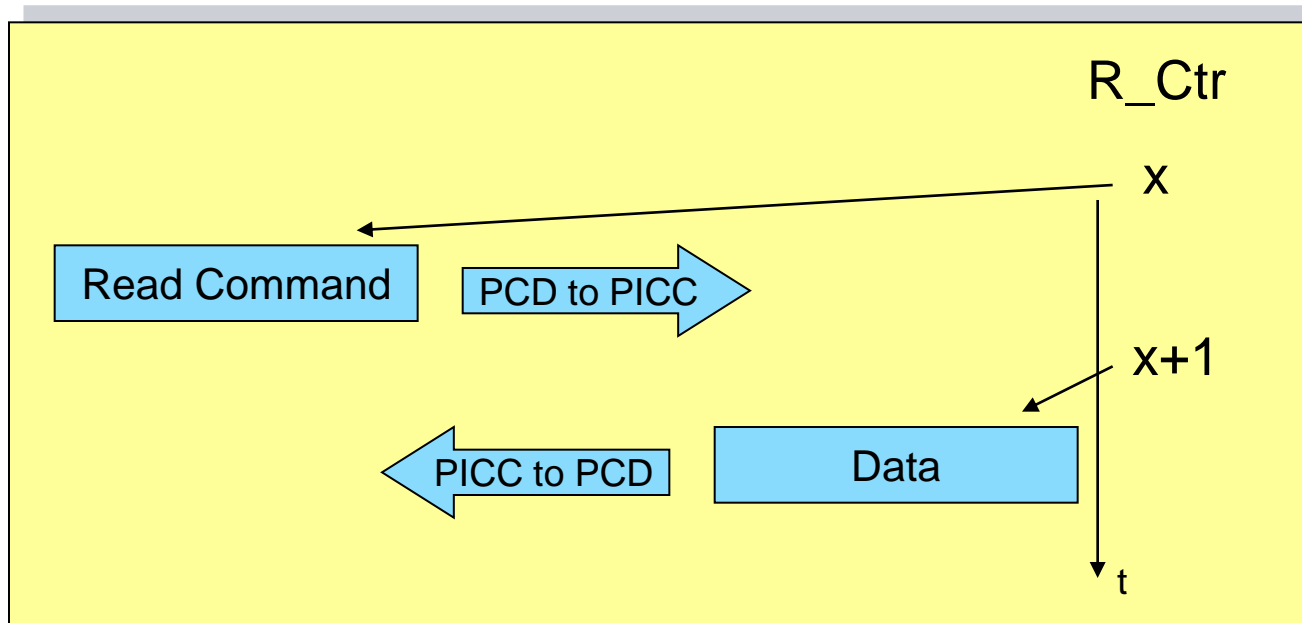
Write Counter W_Ctr

- Format:
  - 2 Byte counter (integer),
  - LSByte first („Little Endian")

- The counter values are never transferred.

- Write Counter counts the Write, Increment, Decrement, Restore and Transfer commands.

- First Authentication resets the W_Ctr

Reset Authentication

1	2
78h	KeyBNr

PCD to PICC →

← PICC to PCD

1
SC

Status code 90_{hex}
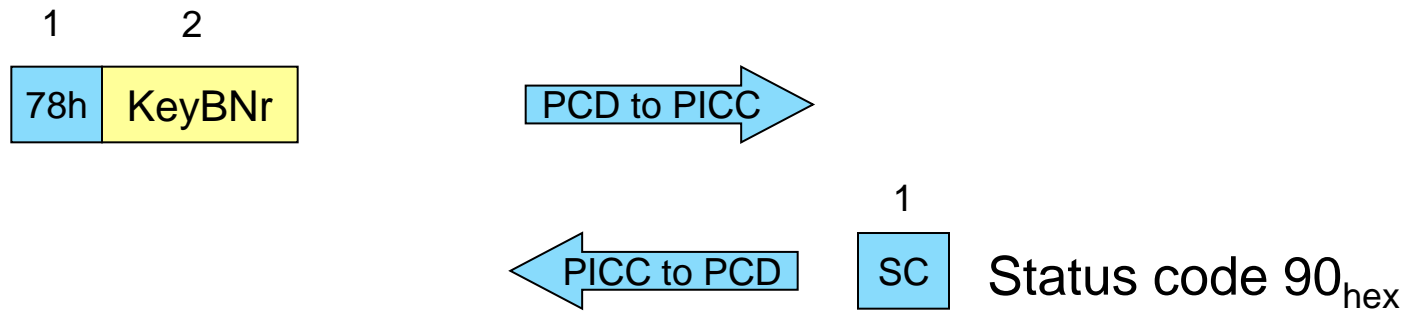
This command resets the authentication.

MIFARE Plus SL3 Read and Write commands

With and without MAC, Encrypted or Plain

MIFARE Plus Read

MIFARE Plus S

Data in Plain	Data encrypted
MAC on response (data)	no MAC on response (data)
MAC on command	no MAC on command

All combinations possible with MIFARE Plus X

MIFARE Plus Read commands

Command Code (hex)	Data	MAC on Command	MAC on Response
30	Encrypted	Yes	No
31	Encrypted	Yes	Yes
32	plain	Yes	No
33	plain	Yes	Yes
34	Encrypted	No	No
35	Encrypted	No	Yes
36	plain	No	No
37	plain	No	Yes

S

MIFARE Plus X

Example of Read 30h

Read encrypted, MAC on commnand, no MAC on Response

1	2	1	8
30h	BNr	Ext	MAC

PCD to PICC →

← PICC to PCD

1	16/32/48…3456
SC	Encrypted Data
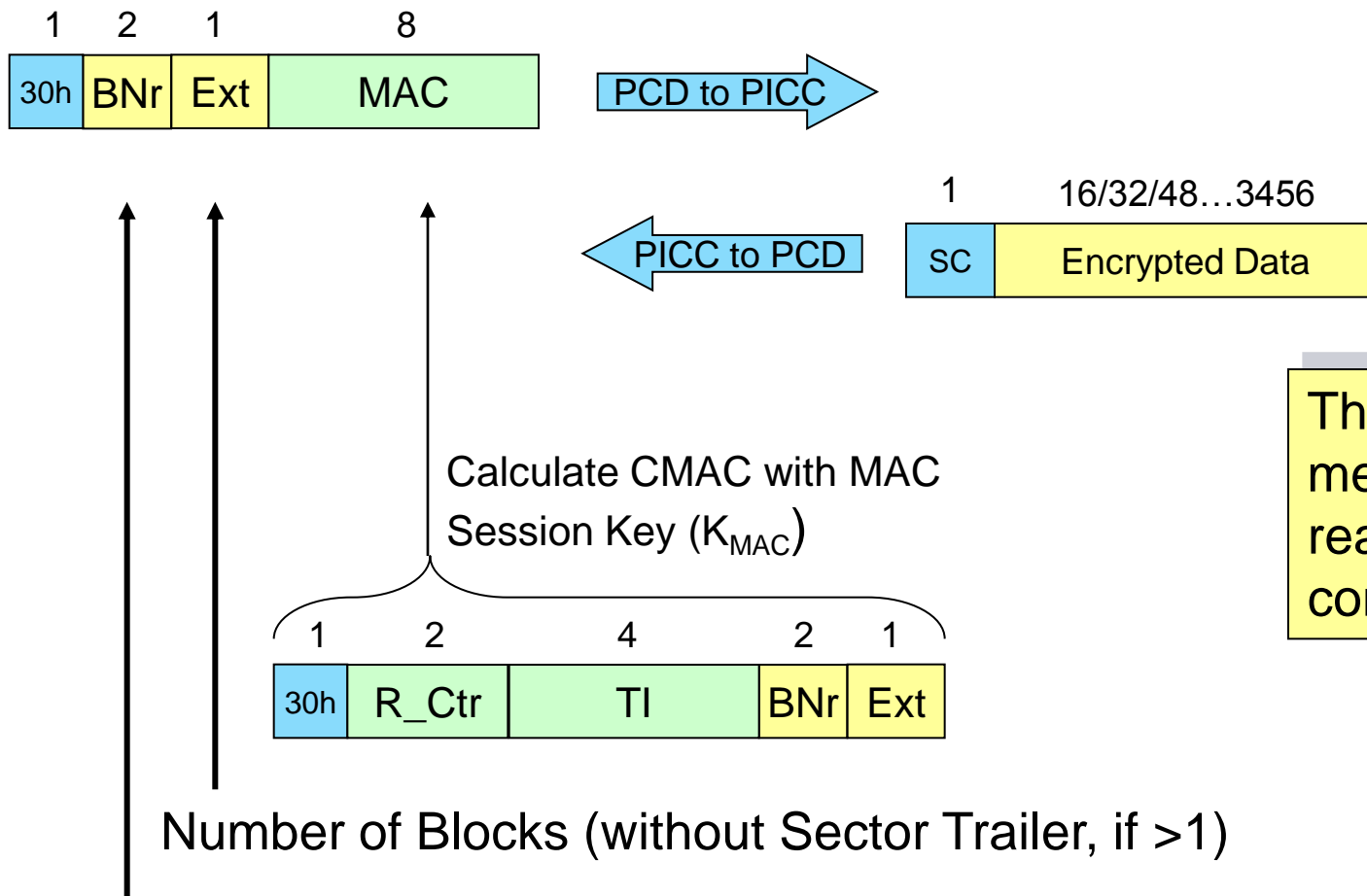
The complete memory can be read with one command.

Calculate CMAC with MAC Session Key (K_{MAC})

1	2	4	2	1
30h	R_Ctr	TI	BNr	Ext

Number of Blocks (without Sector Trailer, if >1)

Blocknumber of the first block

LSByte first!

Example of Read 31h

Read encrypted, MAC on commnand, MAC on Response

1	2	1	8
31h	BNr	Ext	MAC

PCD to PICC →

← PICC to PCD

1	16/32/48…3456	8
SC	Encrypted Data	MAC

Be aware: R_Ctr is incremented between command and response

Calculate CMAC with MAC Session Key (K_{MAC})

1	2	4	2	1	16/32/…3456
90h	R_Ctr	TI	BNr	Ext	Encrypted Data

LSByte first!
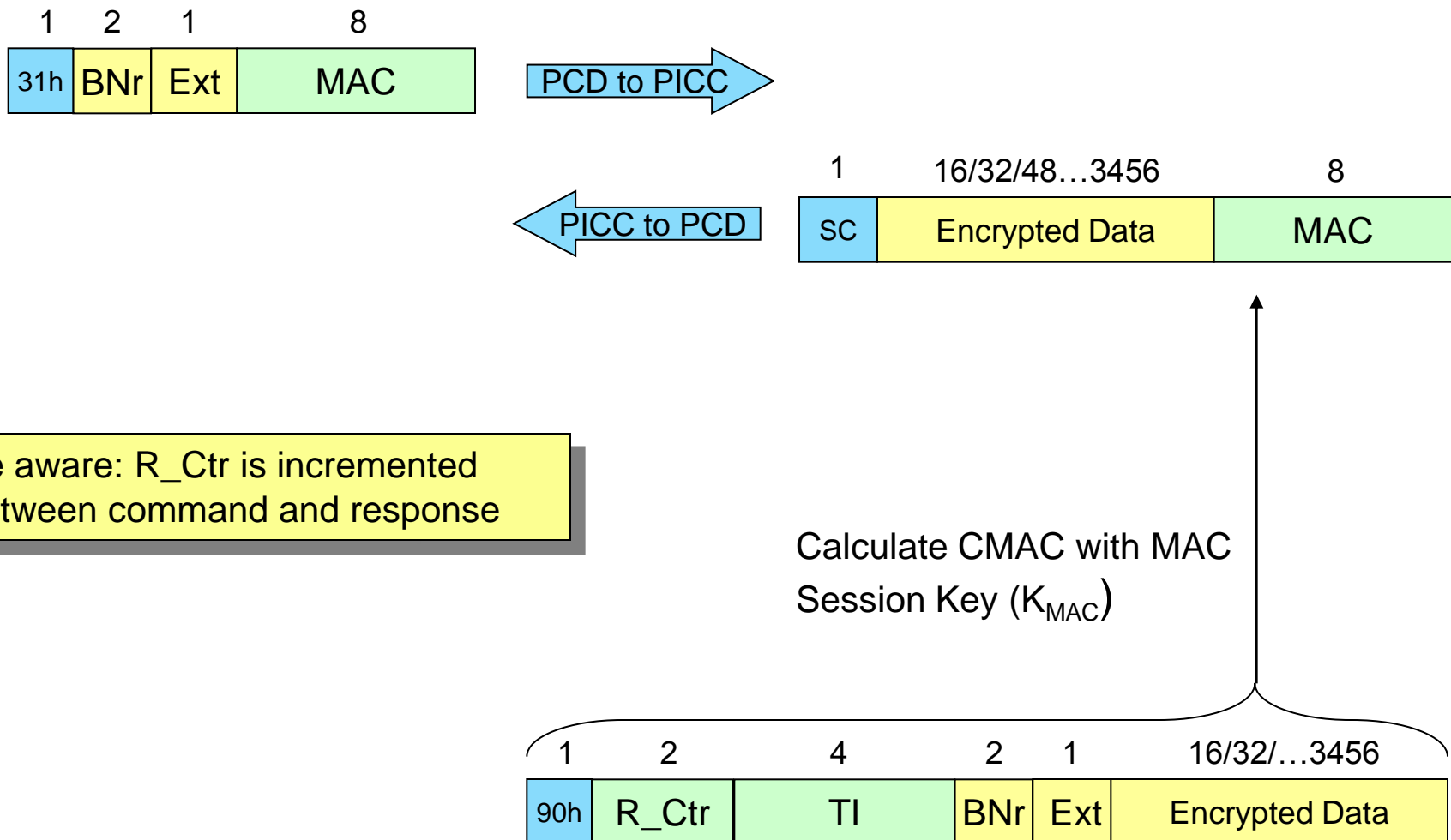
MAC on Response over more than one Read

2 Reads

1	2	1	8
30h	BNr1	2	MAC

PCD to PICC →

← PICC to PCD

1	32
SC	Encrypted Data 1

1	2	1	8
31h	BNr2	3	MAC

PCD to PICC →

← PICC to PCD

1	48	8
SC	Encrypted Data 2	MAC

Calculate CMAC with MAC
Session Key (K_{MAC})

1	2	4	2	1	32	2	1	48
SC	R_Ctr	TI	BNr1	2	Encrypted Data 1	BNr2	3	Encrypted Data 2

LSByte first!
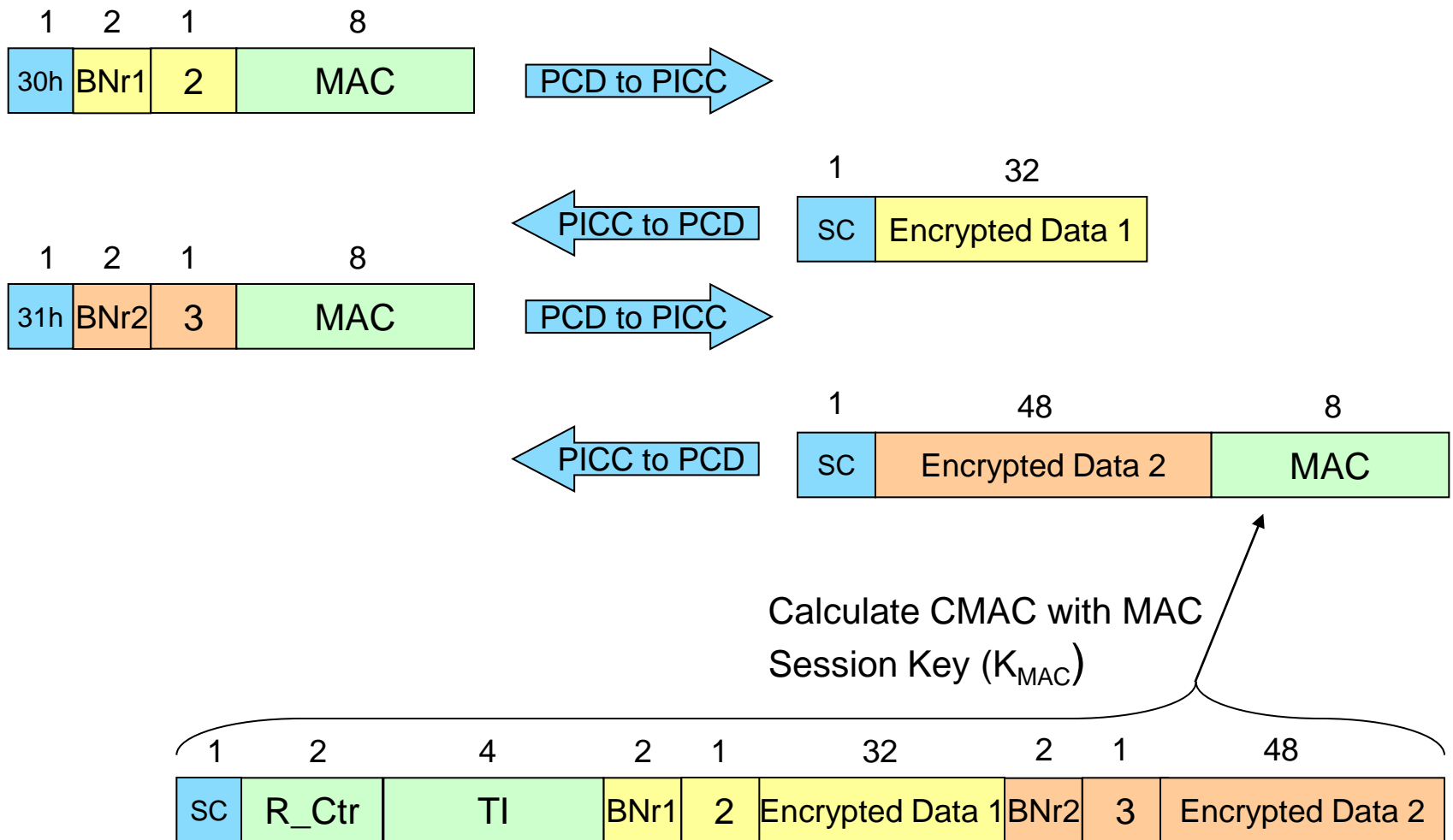
MIFARE Plus Write

MIFARE Plus S

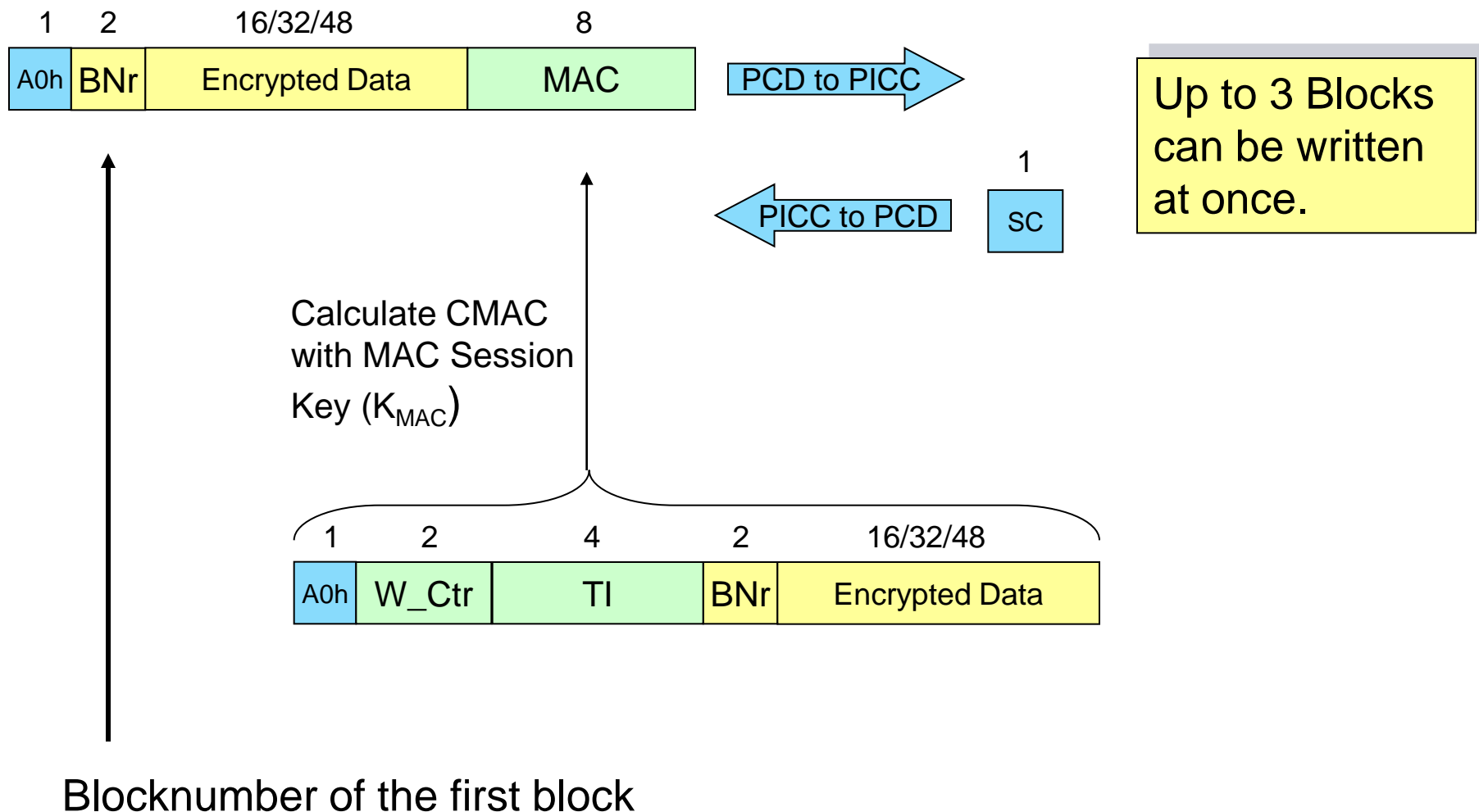| Data in Plain	Data encrypted
MAC on response	**no MAC on response**

Always MAC on command + data

All combinations possible with MIFARE Plus X

MIFARE Plus Write commands

Command Code (hex)	Data	MAC on Command	MAC on Response
A0	Encrypted	Yes	No
A1	Encrypted	Yes	Yes
A2	plain	Yes	No
A3	plain	Yes	Yes

MIFARE Plus S

MIFARE Plus X

Example of Write A0h

Write encrypted, MAC on commnand, no MAC on Response

1	2	16/32/48	8
A0h	BNr	Encrypted Data	MAC

PCD to PICC →

← PICC to PCD

1
SC

Up to 3 Blocks can be written at once.

Calculate CMAC with MAC Session Key (K_{MAC})

1	2	4	2	16/32/48
A0h	W_Ctr	TI	BNr	Encrypted Data

Blocknumber of the first block

Example of Write A1h

Write encrypted, MAC on commnand, MAC on Response

1	2	16/32/48	8
A1h	BNr	Encrypted Data	MAC

PCD to PICC →

← PICC to PCD

1	8
SC	MAC

Calculate CMAC with MAC Session Key (K_{MAC})

Be aware: W_Ctr is incremented between command and response

1	2	4
90h	W_Ctr	TI
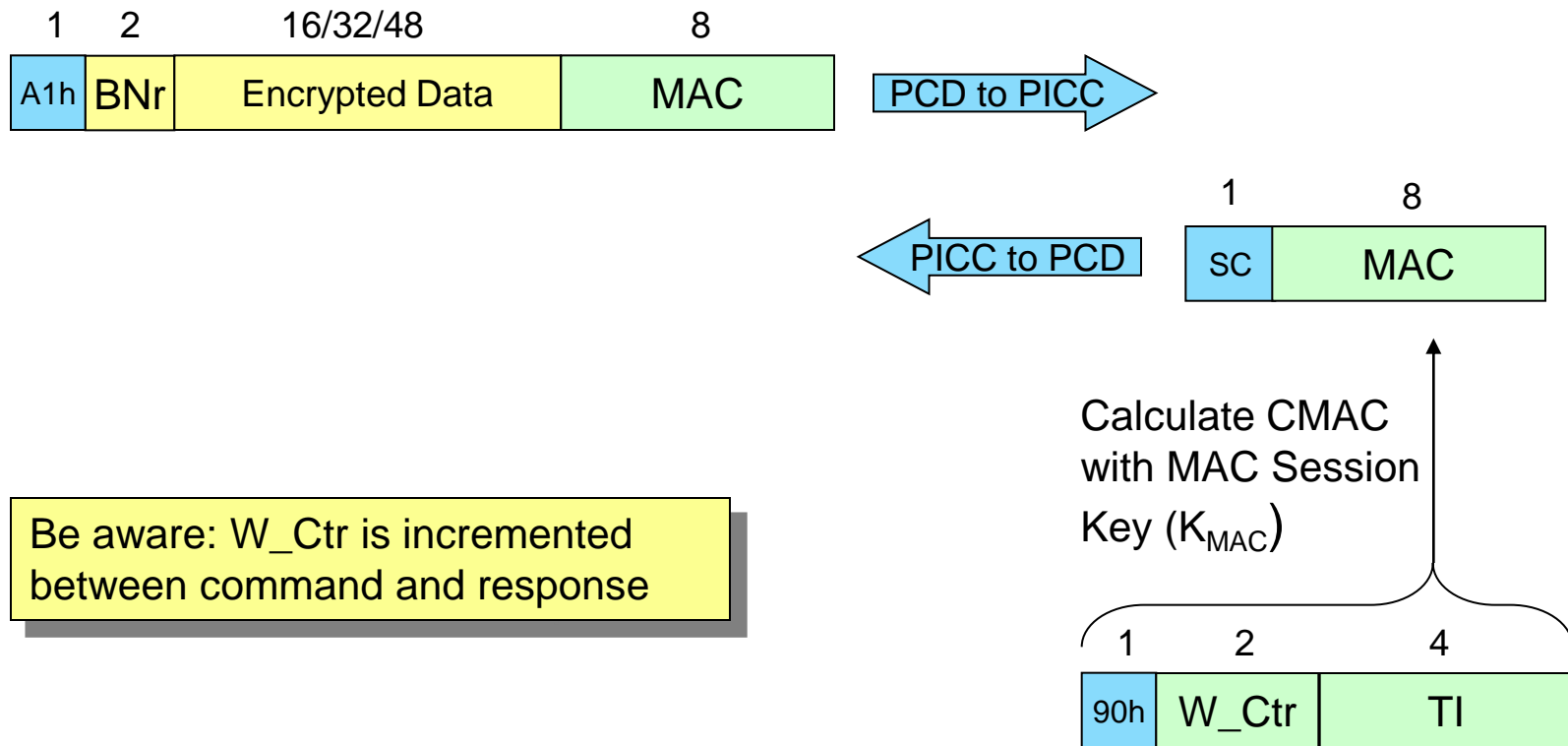
MIFARE Plus Virtual Card Architecture (VCA)

VCA Part 1: Principle

One VC available for the installation

Anti-collision, preferably with Random ID

VCSL command

PCD → "I support installation XYZ" → **PICC**

VCSL response

"My UID is 1234567"

only that PCD that really supports XYZ can decrypt

XYZ

ABC

DEF

No VC available for the installation

Anti-collision, preferably with Random ID

PCD

VCSL command

"I support installation GHI"

VCSL response

GARBAGE

PICC

XYZ

ABC

DEF

Attacker cannot tell difference between garbage and UID
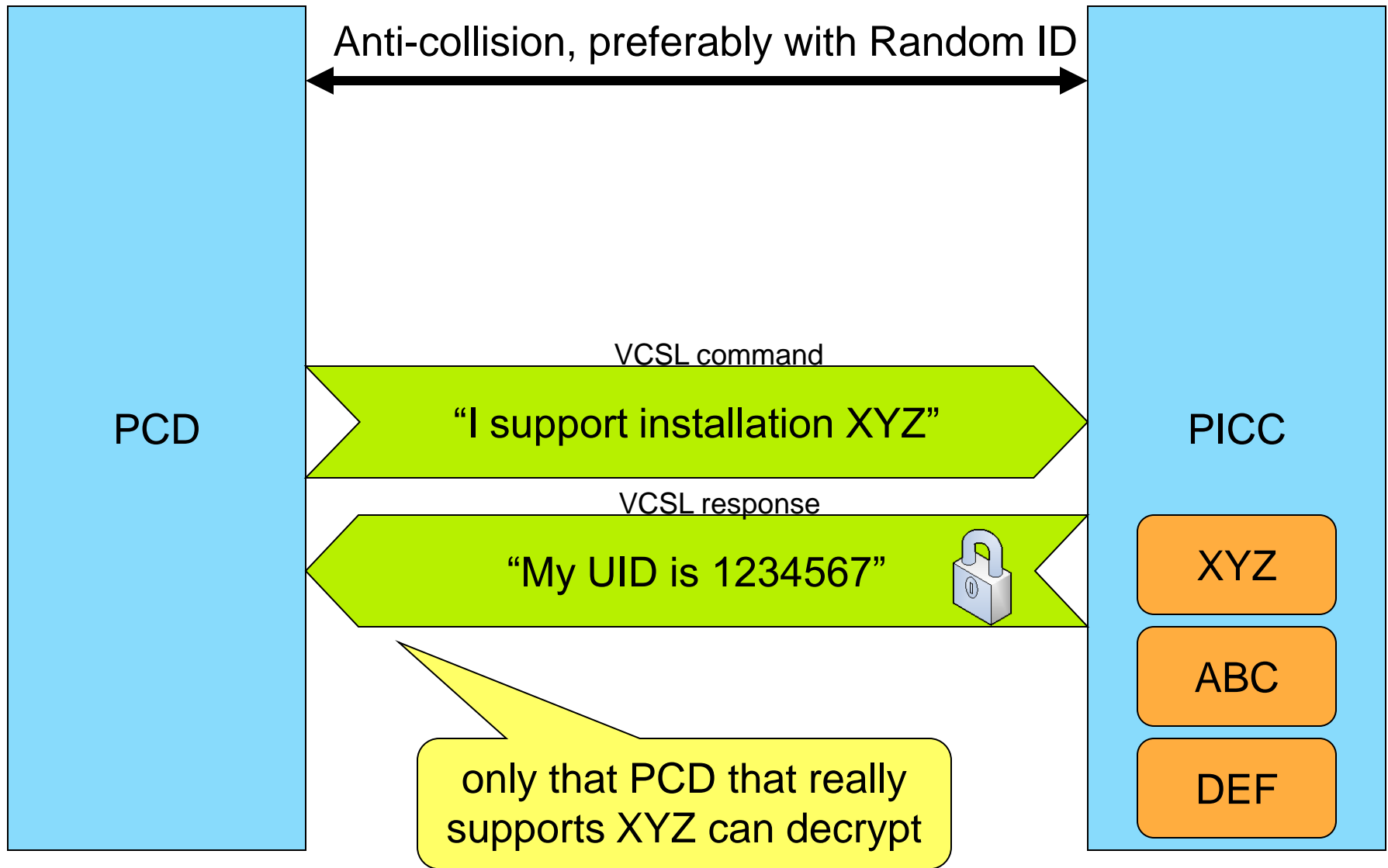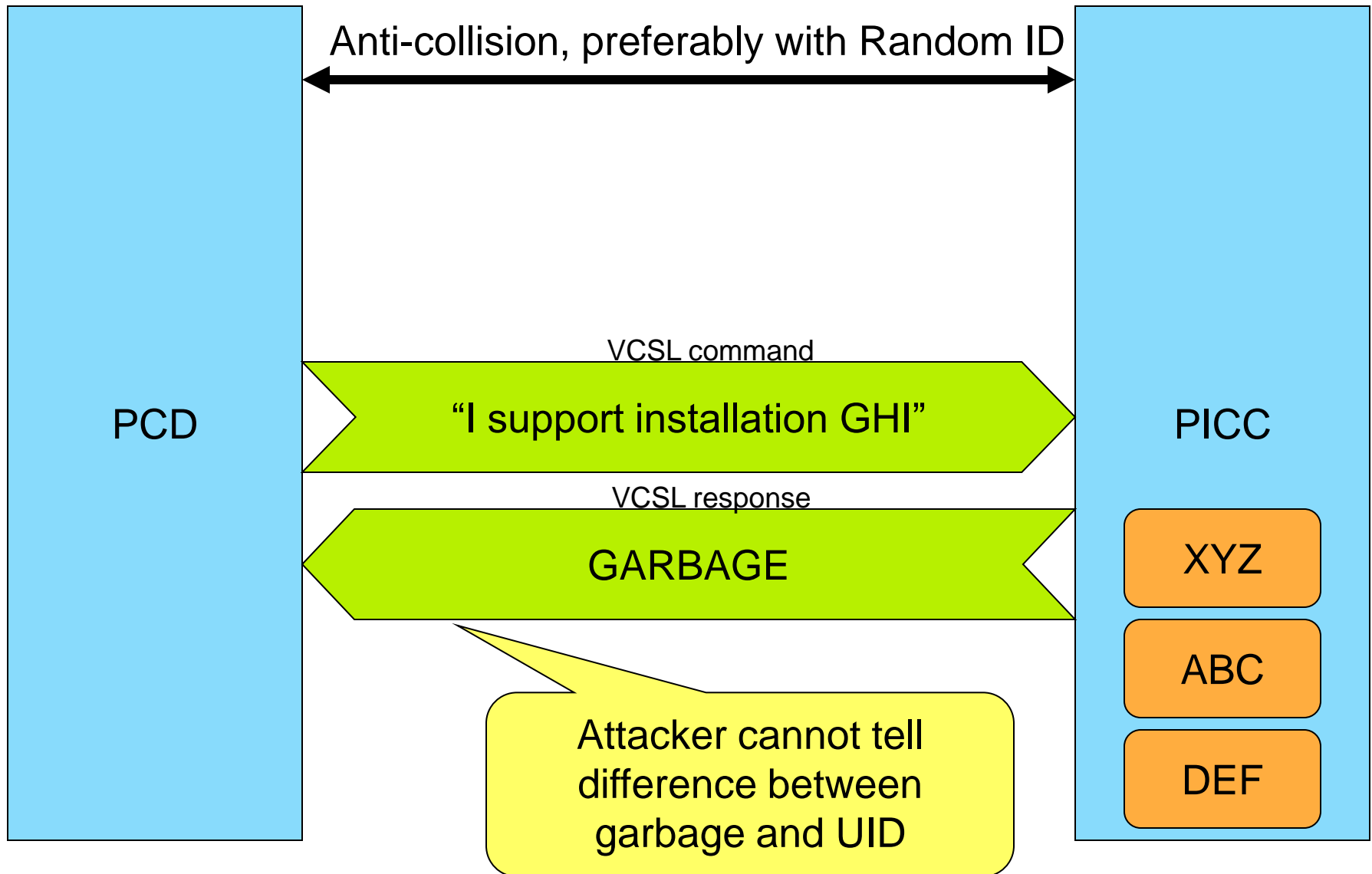
PCD supports multiple installations, PICC makes the choice

PCD

PICC

Anti-collision, preferably with Random ID

VCS command
"I support installation GHI"

VCS response
OK

VCS command
"I support installation ABC"

VCS response
OK

VCSL command
"I support installation XYZ"

VCSL response
"My UID is 1234567"

XYZ

ABC

DEF

Can be UID of ABC or XYZ.
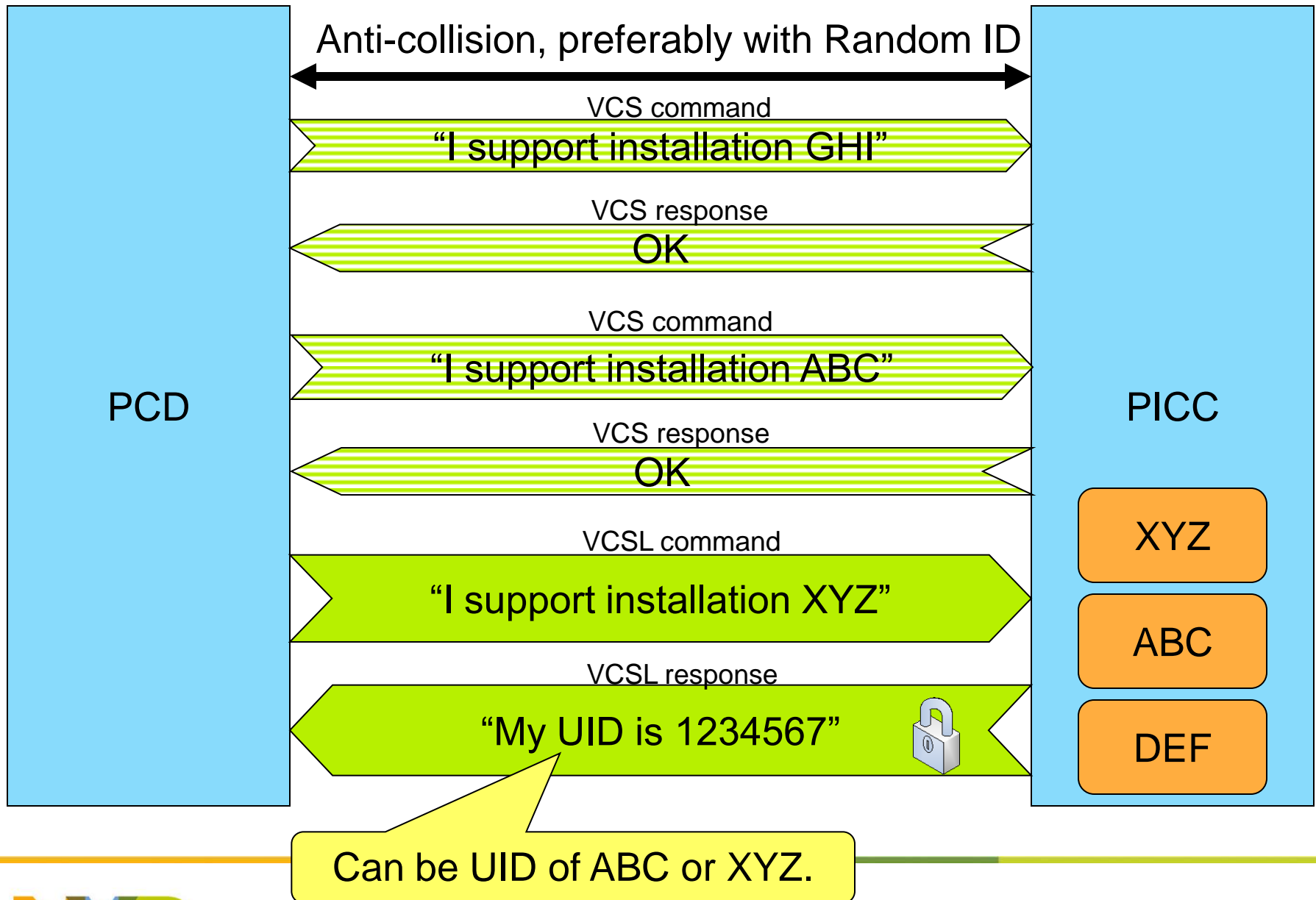
PCD supports multiple installations, PCD makes the choice

PCD

PICC

Anti-collision, preferably with Random ID

VCSL command
"I support installation ABC"

VCSL response
"My UID is 1234567"

VCSL command
"I support installation XYZ"

VCSL response
"My UID is 7654321"

SVC command
Select the VC for ABC

SVC response
OK

XYZ

ABC

DEF
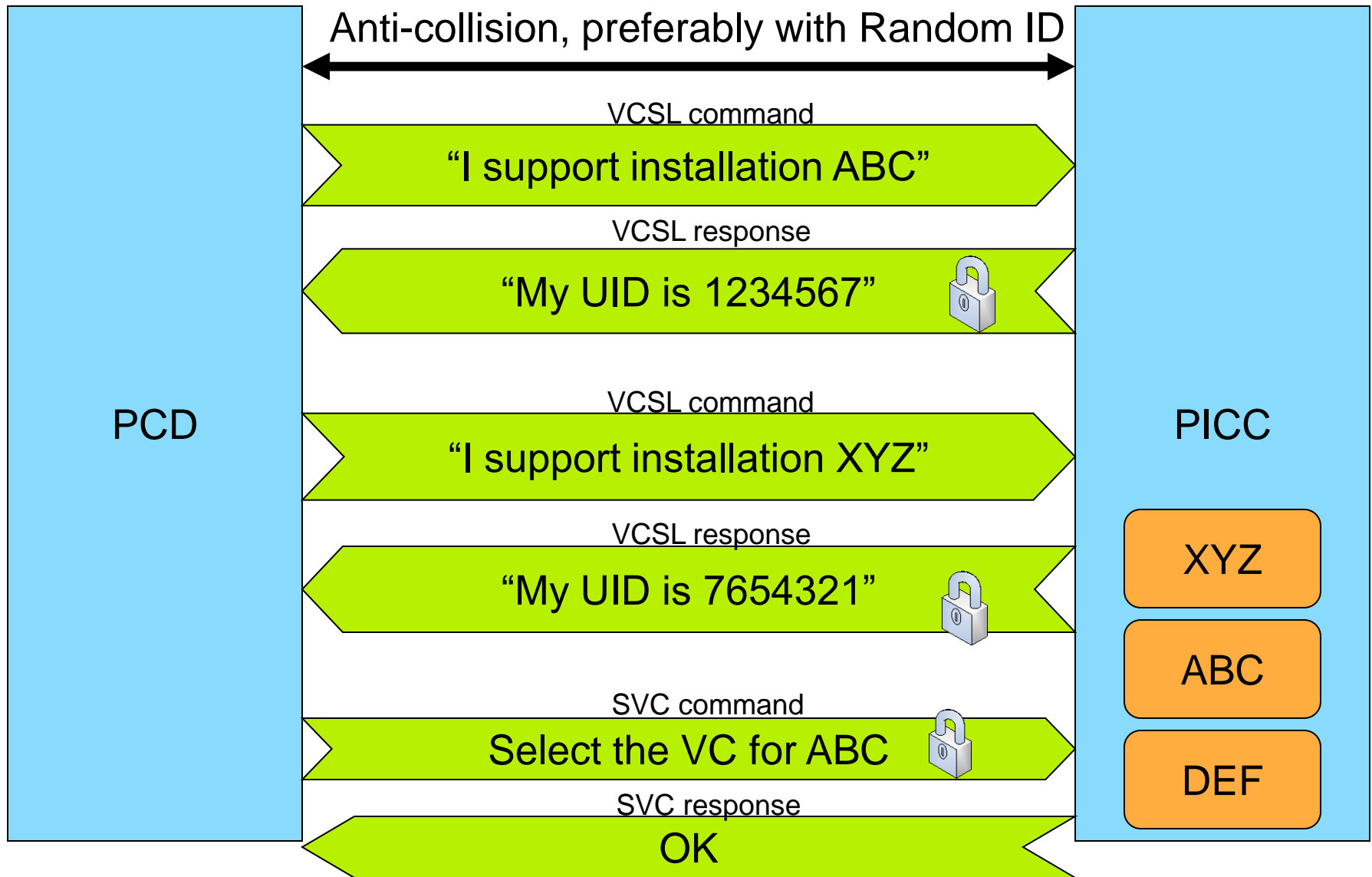
MIFARE Plus Virtual Card Architecture (VCA)

VCA Part 2: Commands

VCA commands

▸ **VCS: Virtual Card Select**
  - To inform the PICC about the IID (Installation supported by PCD)
  - Always returns an „OK".
  - Can be cascaded.

▸ **VCSL: Virtual Card Select Last**
  - To inform the PICC about the IID (Installation supported by PCD)
  - Always returns the encrypted UID or garbage.
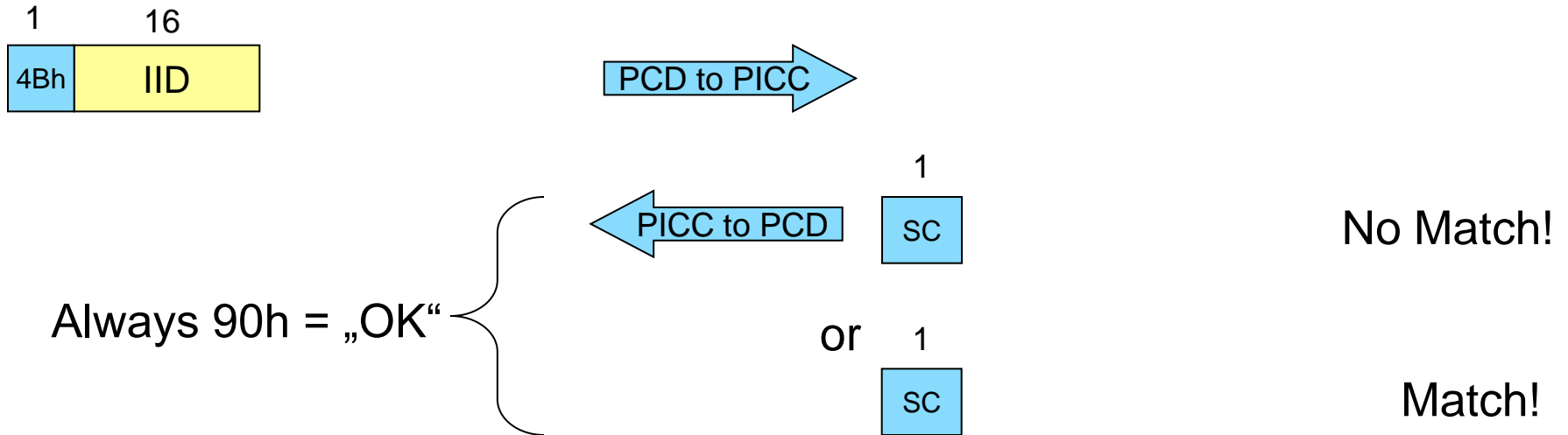 - Can be cascaded.

▸ **SVC: Select Virtual Card**
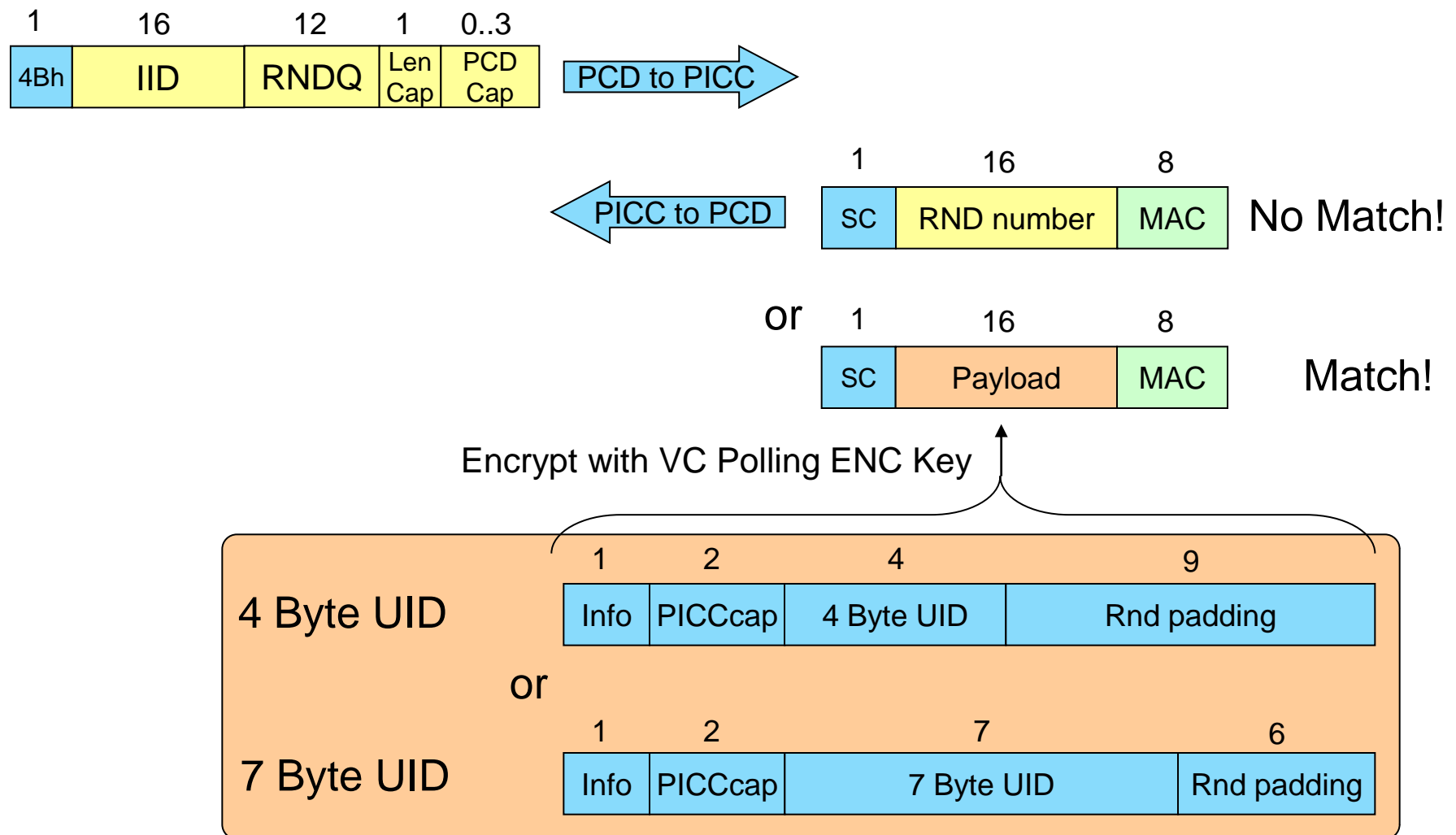 - Selects a VC with its UID.

▸ **DVC**
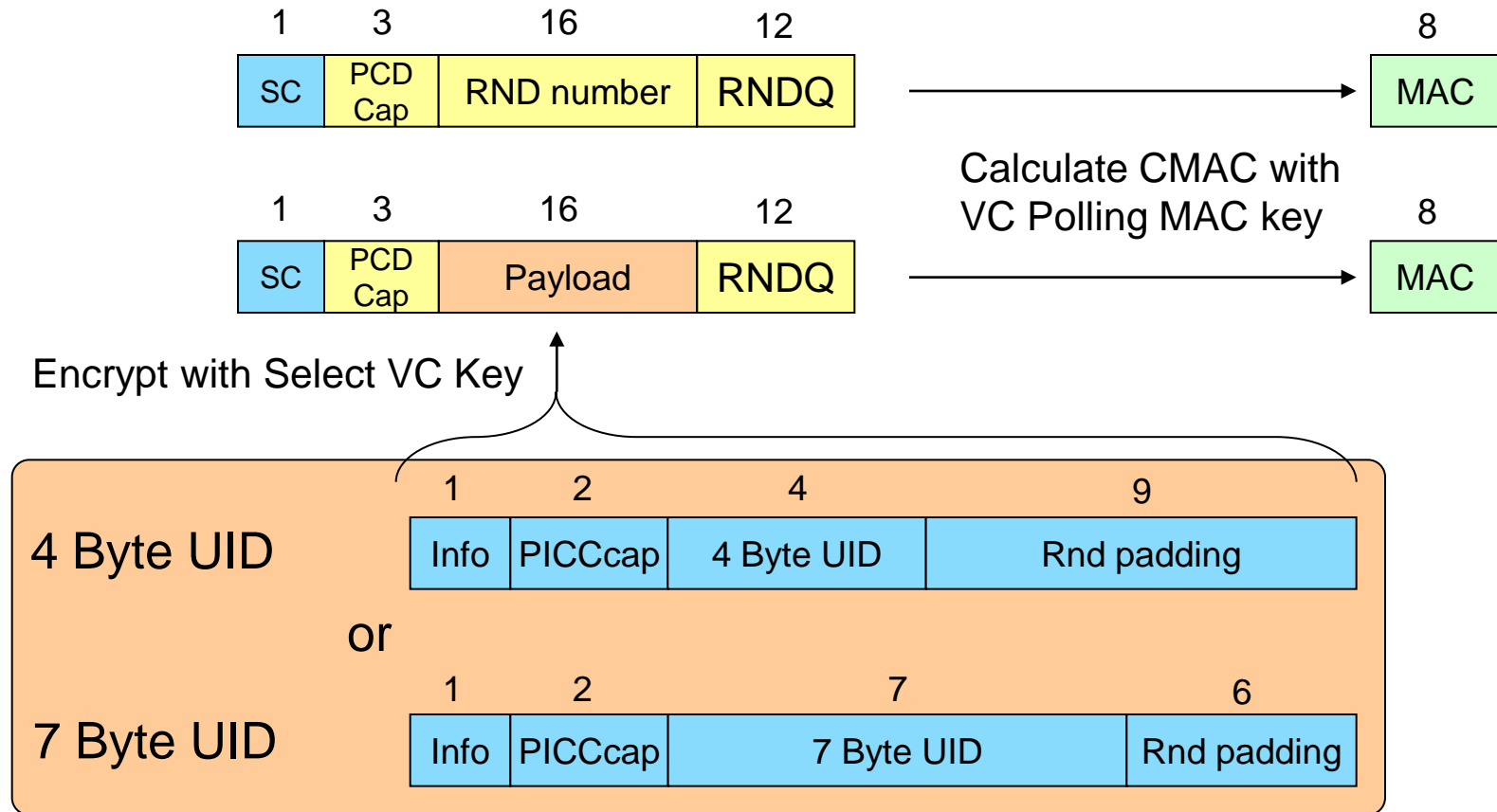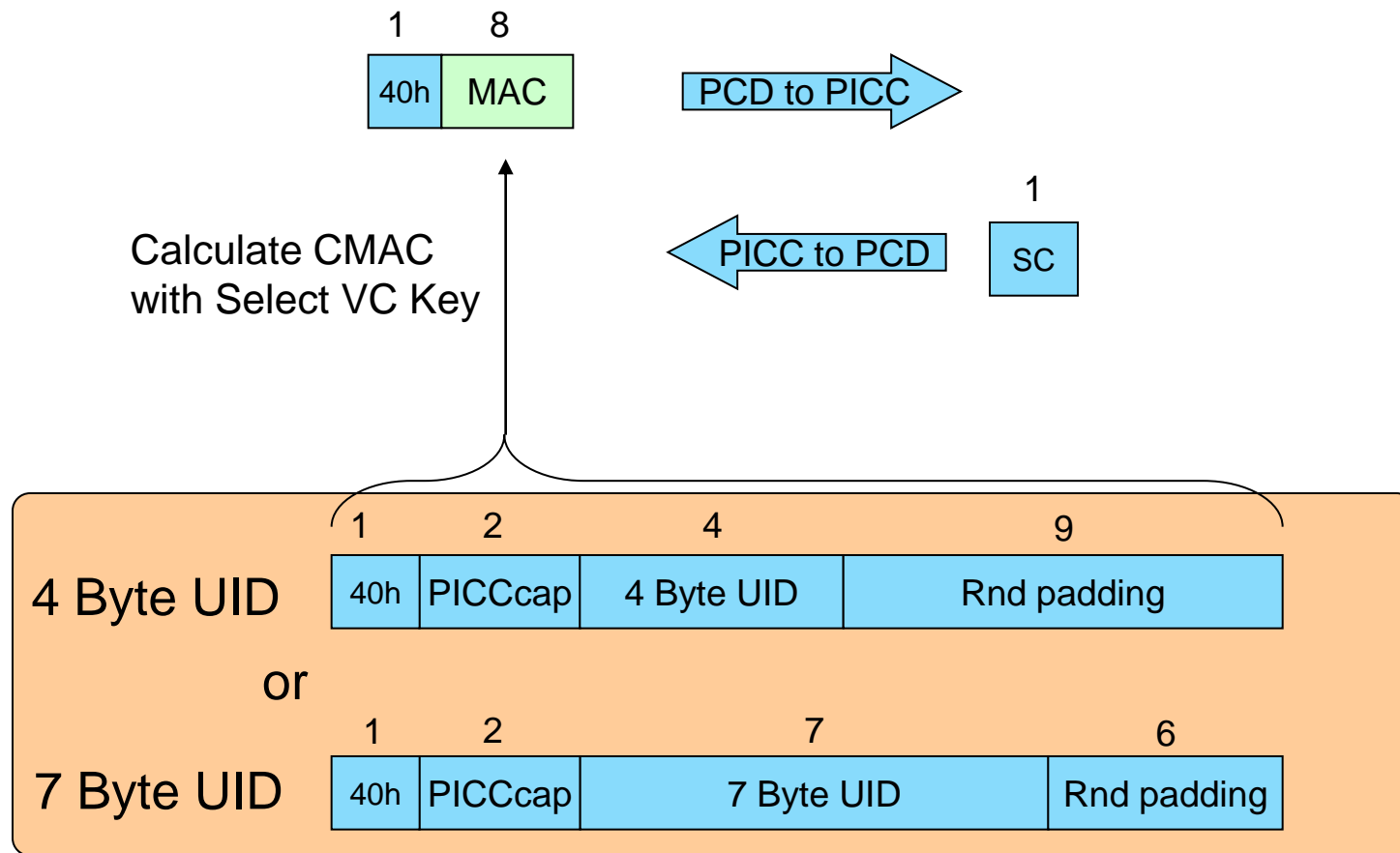 - Deselect a VC with its UID.

Virtual Card Support

1	16
4Bh	IID

PCD to PICC →

← PICC to PCD | 1 | SC |

No Match!

Always 90h = „OK"

or | 1 | SC |

Match!

Virtual Card Support Last

1	16	12	1	0..3
4Bh	IID	RNDQ	Len Cap	PCD Cap

PCD to PICC →

PICC to PCD ←

1	16	8
SC	RND number	MAC

No Match!

or

1	16	8
SC	Payload	MAC

Match!

Encrypt with VC Polling ENC Key

4 Byte UID

1	2	4	9
Info	PICCcap	4 Byte UID	Rnd padding

or

7 Byte UID

1	2	7	6
Info	PICCcap	7 Byte UID	Rnd padding

MAC on Virtual Card Support Last response

1	3	16	12
SC	PCD Cap	RND number	RNDQ

→ | 8 |
|---|
| MAC |

Calculate CMAC with
VC Polling MAC key

1	3	16	12
SC	PCD Cap	Payload	RNDQ

→ | 8 |
|---|
| MAC |

Encrypt with Select VC Key

4 Byte UID

1	2	4	9
Info	PICCcap	4 Byte UID	Rnd padding

or

7 Byte UID

1	2	7	6
Info	PICCcap	7 Byte UID	Rnd padding

Select Virtual Card

1   8

| 40h | MAC |

Calculate CMAC
with Select VC Key

PCD to PICC

PICC to PCD

1

| SC |

4 Byte UID

1   2   4   9

| 40h | PICCcap | 4 Byte UID | Rnd padding |

or

7 Byte UID

1   2   7   6

| 40h | PICCcap | 7 Byte UID | Rnd padding |

Deselect Virtual Card

1

48h

PCD to PICC

1

PICC to PCD

SC
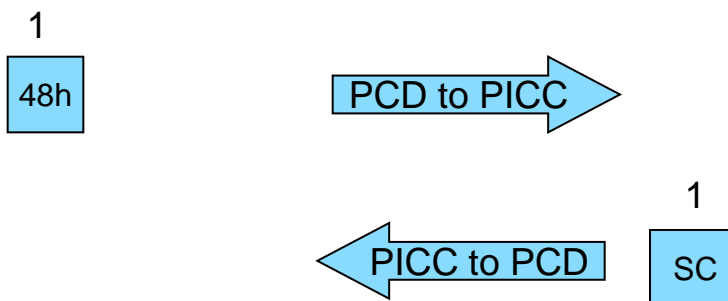
MIFARE Plus Virtual Card Architecture (VCA)

VCA Part 3: How to build the VC IID

Installation Identifier: Proposal of NXP

> **If no MAD is used:**   „Golden Device UID" see next slides

Variant	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
7 byte UID	0xA0	0x00	0x00	0x03	0x96	0x56	0x43	0x41	0x01	UID0	UID1	UID2	UID3	UID4	UID5	UID6
4 byte UID	0xA0	0x00	0x00	0x03	0x96	0x56	0x43	0x41	0x02	UID0	UID1	UID2	UID3	0x00	0x00	0x00
AID	0xA0	0x00	0x00	0x03	0x96	0x56	0x43	0x41	0x03	M0	M1	M2	0x00	0x00	0x00	0x00
Factory default	0xA0	0x00	0x00	0x03	0x96	0x56	0x43	0x41	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF
Explanation	International RID for PIX (assigned to NXP) according ISO 7816-5					Virtual Card Architecture (VCA)			Variant ID	UID, MAD ID or default bytes, and filler bytes						

> **If MAD is used:**   MAD based IID see next slides

MAD based IID

If MAD is used:

AID	0xA0	0x00	0x00	0x03	0x96	0x56	0x43	0x41	0x03	M0	M1	M2	0x00	0x00	0x00	0x00

MAD
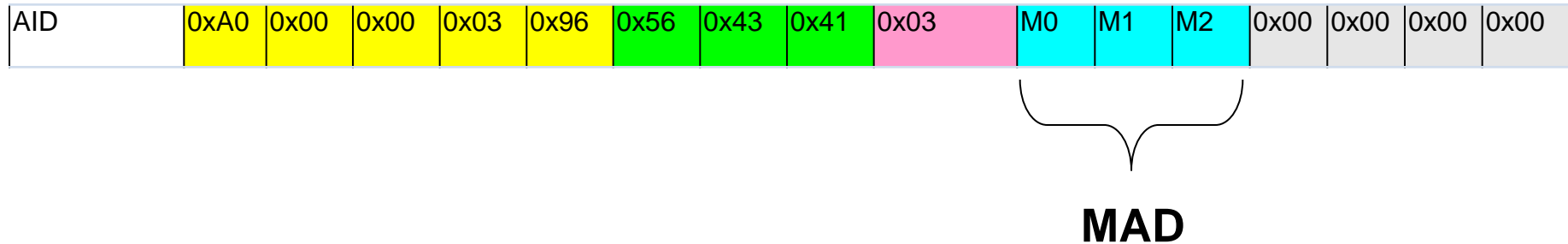
3 bytes from the MAD (like with MIFARE DESFire)
M0 = MIFARE DESFire AID byte 0
M1 = MIFARE DESFire AID byte 1
M2 = MIFARE DESFire AID byte 2

MIFARE DESFire AID Byte 0		MIFARE DESFire AID Byte 1		MIFARE DESFire AID Byte 2	
Nibble 0	Nibble 1	Nibble2	Nibble3	Nibble4	Nibble5
0xF	MIFARE classic AID				0x0 .. 0xF

Details refer to AN „MIFARE Application Directory".

„Golden Reference UID" based IID

If no MAD is used :

7 byte UID	0xA0	0x00	0x00	0x03	0x96	0x56	0x43	0x41	0x01	UID0	UID1	UID2	UID3	UID4	UID5	UID6
4 byte UID	0xA0	0x00	0x00	0x03	0x96	0x56	0x43	0x41	0x02	UID0	UID1	UID2	UID3	0x00	0x00	0x00

UID

▸ Take a MIFARE card (MIFARE Plus).

▸ Read out the UID.

▸ Clearly mark this card that this is **the "Golden Device"**.

▸ Make sure that the UID stays readable,
   – Do not configure **this card** into Random ID.
   – After reading out the UID do not further interact with it.

▸ Lock the card away ("Golden Reference").

▸ Compose the IID using the table in the previous slide using the row 7 byte UID or 4 byte UID depending on the length of the UID that was read from the card.

MIFARE Plus Virtual Card Architecture (VCA)

VCA Part 4: VCA Quick and Easy!!

VCA Quick and Easy!!

▸ What is the target?

▸ How to prepare the MIFARE Plus

▸ How to use VCSL

▸ Additional Remarks

What is the target?

- We want to use the MIFARE Plus with RID -> Privacy protection!

- We want to use the UID to diversify keys -> Security!

- We want to use the fastest (& secure) way to retrieve the UID!

- **VCSL = Virtual Card Select Last command**

How to prepare the MIFARE Plus

1. Create an IID for **YOUR** installation.
   - IID = Installation Identifier = „Unique" Installation ID
   - Proposal from NXP available: use a „Golden Device UID" and store ayway.
   - Details see slides in VCA part 3 above.

2. Create a VC Polling ENC Key for **YOUR** installation.

3. Create a VC Polling MAC Key for **YOUR** installation.

4. Personalize the MIFARE Plus with
   - IID   (Block $B001_{hex}$)
   - VC Polling ENC Key   (Block $A080_{hex}$)
   - VC Polling MAC Key   (Block $A081_{hex}$)

5. Switch MIFARE Plus to use RID in SL3
   - Write Field Configuration Block (e.g. during Personalization)
   - Default   = 00 **55** 55 00 00 00000000 00 00 0000000000   –> no RID
   - change   = 00 **AA** 55 00 00 00000000 00 00 0000000000   –> RID

How to use VCSL

- Activate Card: REQA – Anticollision – Select - RATS (-optionally PPS)
  - This Activation sequence uses the Random ID now.
  - Privacy is protected.

- **Run VCSL to get UID.**
  - You need to know **YOUR IID**.
  - You need to know **YOUR VC Polling ENC Key**.
  - You need to know **YOUR VC Polling MAC Key**.
  - UID is transferred encrypted and MACed -> Secure!
  - Only YOU can retrieve the UID to diversify keys.

- Simple and fast!

Remarks

- VCSL is the fastest way to retrieve the UID.
    - Read Block 0 might be possible, but takes longer.
    - Read Block 0 might not be encrypted (MIFAR Plus S): not secure!

- VC Polling ENC Key must NOT be diversified!

- VC Polling MAC Key must NOT be diversified!