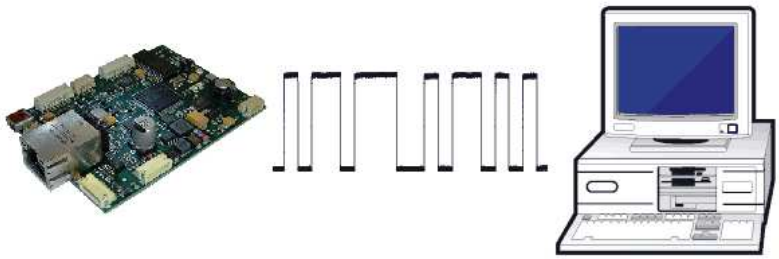
die zeichen lesen
die zeichen setzen

reading the signs
setting the signs



Protocol & CMD

Reference Manual
Rev. 1.07 (Aug. 2012)

ddm hopt+schuler
D-78628 Rottweil
Königsberger Straße 12
Tel. (++49) 741 / 2607-0
Fax (++49) 741 / 1 33 98
ddm@hopt-schuler.com
www.hopt-schuler.com

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing.

History

Date	Rev	Note
10.05.2010	1.00	Initial draft
22.11.2010	1.01	Added dispenser commands
		corrected Stats in position command
04.04.2011	1.02	Added external control commands
08.04.2011	1.03	Added boot loader commands and reader initialization sequence
23.01.2012	1.04	Added “Additional Commands” section
14.05.2012	1.05	Added a new command PCD_STANDBY
16.05.2012	1.06	Descriptions of status codes are added
04.05.2012	1.07	NFC command added

Table of Content

1 INTRODUCTION

This document describes the common protocol and command set for the 882 and derived models and applies to all interfaces like RS232, USB or Ethernet.

1.1 Transmission Format

The default parameter settings are:

Format: 1 start bit, 8 data bits, 1 stop bit, no parity
Baud rate: 38400 bps (baud)
Handshaking: No DTR/CTS control
Address: 01h

These parameters can be changed by changing the Communication Parameters data.

1.2 Possible Transmission Formats

Baud rate: Supported baud rates are; 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 and 230400.

Parity: No parity, even parity and odd parity.

Address: The default address is 01h. In a network configuration, for example if many readers connected to the host via the RS485 connection, each reader can be configured with a different address (between 2 and 127). So the host has to specify the address of the reader it wants to request. Address 99 is fixed and works for all readers and can be used to read the actual address.

1.3 Frame Format

All frames are structured as followed:

SOH	ADDR	LEN	DATA	BCC

With:

SOH = 01h

A maximum of 500ms is allowed between two consecutive characters.

ADDRESS: The device address field is one byte. This value is normally “1”. For extended configuration i.e. network configuration, diverse values should be used.

LEN: This field is the DATA length and is encoded in two bytes (MSB first).

DATA: This is the command or the response message. The next section defines it.

BCC: Is the “Block Check Character”. Its value is equal to the results of exclusive OR of all preceding bytes (SOH byte is included).

1.4 Data Format

The DATA format is structured as followed:

1.4.1 Host-to-Reader (Command)

Command (1 Byte)	Message (may be 0 length)

1.4.2 Reader-to-Host (Answer)

Status (1 Byte)	Message (may be 0 length)

2 MEMORY CONFIG

The reader stores the configuration data in a non-volatile memory. The standard memory is a 2-MBit NAND-Flash.

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	**MEMSTAT**	Memory status (Read only)							
1	**NETADDR**	Address for network configuration							
2	**SERIAL0**	Serial Number LSB byte0							
3	**SERIAL1**	Serial Number byte1							
4	**SERIAL2**	Serial Number byte2							
5	**SERIAL3**	Serial Number MSB byte3							
6	**RCOUNT0**	Number of read cycles LSB byte0							
7	**RCOUNT1**	Number of read cycles byte1							
8	**RCOUNT2**	Number of read cycles byte2							
9	**RCOUNT3**	Number of read cycles MSB byte3							
10	**MODCON**	Type B Modulation Conductance							
11	**ANT1PW**	Antenna 1 transmitter power level							
12	**ANT2PW**	Antenna 2 transmitter power level							
13	**CONFIG0**	DLED	CI	TYPEA	TYPEB	RM1	RM0	AC1	AC0
14	**CONFIG1**	DTR	CTS	P1	P0	BR3	BR2	BR1	BR0
15	**CONFIG2**	RFU	RFU	RFU	SPI	MDB	USB	ETH	RS
16	**CONFIG3**	RO	CTLS	HICO	SHUT	CHIP	TR3	TR2	TR1
17	**CONFIG4**	ECC	ERC	ICC	IRC	MC	AATR	ASR	ISO
18	**CONFIG5**	RFU	RFU	RFU	RFU	RFU	RFU	RFU	POL
19	**RXTHA**	Receiver threshold A							
20	**RXTHB**	Receiver threshold B							
21	**MAGSTART**	Magstripe write start parameter							
22	**PDATE**	Production Date DAY							
23	**PDATE**	Production Date MONTH							
24	**PDATE**	Production Date YEAR							

2.1 Main Configuration CONFIG0

Register: CONFIG0

Bit	7	6	5	4	3	2	1	0	
	DLED	**CI**	**TYPEA**	**TYPEB**	**RM1**	**RM0**	**AC1**	**AC0**	**CONFIG0**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Init Value	0	0	0	0	1	0	0	0	0x08

Bit 1...0 – AC1, AC0: Antenna Configuration

AC1	AC0	Antenna Configuration
0	0	Normal push-pull antenna
0	1	Antenna1 only
1	0	Antenna2 only
1	1	Antenna1+2

Bit 3-2 – RM1, RM0: Running Mode

RM1	RM0	Running Mode	Description
0	0	Normal	Depends on last state
0	1	Polling	Continues polling until command
1	0	Detection	Stops after tag in range
1	1	Soft Detection	Low power slow detection

To activate a running mode, use the SET_RUNNING_MODE command.

Normal: Do nothing.
Polling: In this mode, the reader polls for both Type A and Type B PICCs. Following events will be reported until the host sends a command to the card;

- Tag in range
- Second tag in range
- Tag removed (first or second)
- Collision error (if “Multiple Tags” is disabled)

Detection: The reader polls for tags and stops as soon as a tag enters the field. So only a tag detected event will be reported.
Soft detection: To detect a card in low power mode the carrier signal is switched on every 100ms and switched off again after the request a card signal.

TYPEB: If this bit is set, the reader polls for Type B cards
TYPEA: If this bit is set, the reader polls for Type A cards
CI Card Interrupt: If this bit is set, card detection interrupt will be active.
DLED: A one disables the red LED blinking.

2.2 Communication Parameters CONFIG1

Register: CONFIG1

Bit	7	6	5	4	3	2	1	0	
	DTR	**CTS**	**P1**	**P0**	**BR3**	**BR2**	**BR1**	**BR0**	**CONFIG1**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Init Value	0	0	0	0	0	1	1	1	0x07

Bit 3...0 – BR3, BR2, BR1, BR0: Baudrate

BR3	BR2	BR1	BR0	Baud rate
0	0	0	0	2400
0	0	0	1	4800
0	0	1	0	9600
0	0	1	1	19200
0	1	0	0	38400
0	1	0	1	57600
0	1	1	0	76800
0	1	1	1	115200
1	0	0	0	230400

Bit 5...4 – P1, P0: Parity

P1	P0	Parity
0	0	None
0	1	Even parity
1	0	Odd parity

Bit 6 – CTS: If this bit is set, the CTS control is active.
Bit 7 – DTR: If this bit is set, the DTR control is active.

2.3 Interface Configuration CONFIG2

Register: CONFIG2

Bit	7	6	5	4	3	2	1	0	
	RFU	RFU	RFU	SPI	MDB	USB	ETH	RS	CONFIG2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Init Value	0	0	0	0	0	0	0	1	0x01

Bit 0 – RS: RS232 interface selected
Bit 1 – Eth: Ethernet interface selected
Bit 2 – USB: USB interface selected
Bit 3 – MDB: MDB interface selected
Bit4 – SPI: SPI interface selected
Bit5 – 7 RFU: Reserved for future use

2.4 Function Select CONFIG3

Register: CONFIG3

Bit	7	6	5	4	3	2	1	0	
	RO	CTLS	HICO	SHUT	CHIP	TR3	TR2	TR1	CONFIG3
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Init Value	0	0	0	0	0	0	0	0	0x00

Bit 0 – TR1: Track1 is active
Bit 1 – TR2: Track2 is active
Bit 2 – TR3: Track3 is active
Bit 3 – CHIP: Chip module is active
Bit4 – SHUT: Shutter is available
Bit5 – HICO: Hico selected
Bit6 – CTLS: Contactless module is active
Bit7 – RO: Read only for magstripe

2.5 Behavior Select CONFIG4

Register: CONFIG4

Bit	7	6	5	4	3	2	1	0	
	ECC	**ERC**	**ICC**	**IRC**	**MC**	**AATR**	**ASR**	**ISO**	**CONFIG4**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Init Value	0	0	0	0	0	0	0	0	0x00

Bit 0 – ISO: Magstripe write position ISO compatible (7mm)
Bit 1 – ASR: Automatic switch report
Bit 2 – AATR: Auto ATR for chip cards
Bit 3 – MC: Multiple contactless cards are allowed
Bit4 – IRC: If card is present at power on, return it
Bit5 – ICC: If card is present at power on, capture it
Bit6 – ERC: If card is present at power fail, return it
Bit7 – ECC: If card is present at power fail, capture it

2.6 Behavior Select CONFIG5

Register: CONFIG5

Bit	7	6	5	4	3	2	1	0	
	RFU	**RFU**	**RFU**	**RFU**	**RFU**	**RFU**	**RFU**	**POL**	**CONFIG4**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Init Value	0	0	0	0	0	0	0	0	0x00

Bit 0 – POL: Start polling after reset
Bit1...7 – RFU: Reserved for future use

3 COMMAND SET

3.1 Reader Commands

READ_STATUS	Send	Len
Transmit	0x28	1
Receive	0x00 (stat_OK) or error code	1
	STATUS[0]-STATUS[3]	4
Description	Read status bytes <*STATUS1*>: Bit 0: 0: If no card is detected 1: if a card is detected Bit 1: 0: If no card is inserted 1: if a card is completely inserted Bit 2: 0: If card is not locked 1: if card is locked Bit 3: 0: ICC/SAM is inactive 1: ICC/SAM is activated (powered ON) Bit 4: 0: Non auto switch report 1: automatic switch report mode Bit 5: 0: No magnet stripe 1: magnet stripe has been detected Bit 6: 0: Idle mode (not armed) 1: armed mode Bit 7: 0: No-auto ATR mode 1: auto ATR sending mode <*STATUS2*>: Bits 1-0: Green led state: 00: led is off; 01: led is on; 10: led is flashing Bits 3-2: Red led state: 00: led is off; 01: led is on; 10: led is flashing Bit 4: Arm mode active in “debug” mode Bit 5: Arm mode active, with magnetic “Pre-head” detection Bit 6: Intake mode active Bit 7: 0: Intake to head 1: Intake to chip position <*STATUS3*>: Bits 0 to 2: Current ICC/SAM selection (0 = User card; 1 to 5: SAM id.) Bit 3: 0: If no SAM#1 is present 1: SAM#1 present in its socket Bit 4: 0: If no SAM#2 is present 1: SAM#2 present in its socket Bit 5: 0: If no SAM#3 is present 1: SAM#3 present in its socket Bit 6: 0: If no SAM#4 is present 1: SAM#4 present in its socket Bit 7: 0: If no SAM#5 is present 1: SAM#5 present in its socket <*STATUS4*>: It's the error code of the last ICC data exchange (with asynchronous card) Bit 0: 1: Electrical abnormality Bit 1: 1: Timeout in ATR reception Bit 2: 1: Character reception timeout Bit 3: 1: Parity error during reception Bit 4: 1: Collision detection during transmission Bit 5: 1: Parity error during transmission Bit 6: 1: Error in protocol T=0 Bit 7: 1: Error in protocol T=1	

READ_REGISTER	Send	Len
Transmit	0x30	1
	ADDR[0]-ADDR[1] (MSB first)	2
	LEN[0]-LEN[1] (MSB first)	2
Receive	0x00 (stat_OK) or error code	1
	DATA[0]-DATA[n]	n
Description	Read from module configuration registers	

WRITE_REGISTER	Send	Len
Transmit	0x31	1
	ADDR[0]-ADDR[1] (MSB first)	2
	LEN[0]-LEN[1] (MSB first)	2
	DATA[0]-DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
Description	Update module configuration registers	

MODULE_RESET	Send	Len
Transmit	0x33	1
Receive	0x00 (stat_OK) or error code	1
Description	Reader reset	

SET_RUNNING_MODE	Send	Len
Transmit	0x5D	1
	MODE 0: Normal 1: Polling 2: Detection 3: EMV Detection	1
Receive	0x00 (stat_OK) or error code	1
Description	Set polling mode **Normal:** Do nothing. **Polling:** In this mode, the reader polls for both Type A and Type B PICCs. Following events will be reported until the host sends a command to the card; • Tag in range • Second tag in range • Tag removed (first or second) • Collision error (if "Multiple Tags" is disabled) **Detection:** The reader polls for tags and stops as soon as a tag enters the field. So only a tag detected event will be reported. **EMV detection:** (See section 6) If the reader receives a contactless card command, it stops polling to avoid collisions.	

GET_INFO	Send	Len
Transmit	0x72	1
	INDEX 0: All Infos (92 Bytes) 1: Serial number (4 Bytes) 2: Firmware Version (6 Bytes) 3: Version & Copyright (58 Bytes) 4: Model (6 Bytes) 5: PCB (11 Bytes) 6: Cycle (4 Bytes) 7: Production date (3 Bytes DDMMYY)	1
Receive	0x00 (stat_OK) or error code	1
	Info string	n
Description	Read reader informations	

3.2 Transport Commands

POSITION	Send	Len
Transmit	0xC2	1
Receive	0x00 (stat_OK) or error code	1
	STAT 0x70: No card is detected 0x71: Card is detected at front position 0x72: Card is completely inserted 0x73: Long card!	1
Description	Read card position If the "Auto switch report" mode is active, the switch status word is automatically sent to the host, each time a switch position change is detected. <SWITCH_STAT1> 0x00: if the reader has got less than 8 switches Bit7…….Bit0 S15…S8 <SWITCH_STAT2> Bit7…….Bit0 S7…S1	

LOCK	Send	Len
Transmit	0xC3	1
Receive	0x00 (stat_OK) or error code	1
Description	Activate the locking motor to **lock** the card	

CAPTURE_CARD	Send	Len
Transmit	0xC4	1
Receive	0x00 (stat_OK) or error code	1
Description	Move the card to the rear direction, until the card is ejected through the rear side Note: This command is refused, while the chip card is activated (powered on). In this case, the code **stat_INVALID** will be returned.	

CHIP_POS	Send	Len
Transmit	0xC5	1
Receive	0x00 (stat_OK) or error code	1
Description	Move the card to chip position This command is available on the motorized reader 866. After the card is seated in chip position, only the Unlocking command will be allowed. Capture is in chip position mechanically not possible.	

UNLOCK	Send	Len
Transmit	0xC6	1
Receive	0x00 (stat_OK) or error code	1
Description	Deactivate the locking system. With motorized reader, the card is moved to front position Note: The Unlocking command is refused, while the chip card is activated (powered on). In this case, the code **stat_INVALID** will be returned.	

ABORT	Send	Len
Transmit	0xC7	1
Receive	0x00 (stat_OK) or error code	1
Description	Abort the ARM mode	

ARM	Send	Len
Transmit	0xC8	1
Receive	0x00 (stat_OK) or error code	1
Description	ARM to read and clear all previous data. The purpose of the ARM command is to prepare the reader waiting for a new magnetic card. This reader status is herein called "ARM mode". When a card is being moved through the reader, magnetic decoding is performed. Data are analyzed, and the reader is then able to return the valid (or invalid) magnetic stripe data. At this state, the reader is exiting the "ARM" mode. Depending on the reader's configuration, the option "***Arm-And-Ack***" makes the reader automatically inform the host when it leaves the ARM mode and data are available. This option is useful to avoid a reader polling. When the option "***Arm-And-Ack***" is not activated, the reader does not transmit, by itself, any message when the ARM mode is done. So this principle requires the host to poll the reader (using for example the **READ_STATUS** command) to know about a status change on the reader. When the option "***Arm-And-Ack***" is activated, the following sequence applies just after a card has been inserted into the reader. • Send an **event_ARM_ACK** and three track status bytes when the magnetic decoding process has been done • Send an **event_ARM_NACK** if no magnetic stripe has been detected	

INTAKE	Send	Len
Transmit	0xCA	1
	POS 0: Intake to head 1: Intake to chip position	1
Receive	0x00 (stat_OK) or error code	1
Description	Set the reader to intake mode This command makes the device ready to accept a new card. When a card is detected at front position, the reader will intake the card to the defined position.	

ABORT_INTAKE	Send	Len
Transmit	0xCB	1
Receive	0x00 (stat_OK) or error code	1
Description	Abort the Intake mode	

MOVE	Send	Len
Transmit	0xC1	1
	INDEX 0: Capture card 1: Unlock card 2: Transport to front 3: Lock card	1
Receive	0x00 (stat_OK) or error code	1
Description	This command could be used alternatively to move a card inside the reader.	

3.3 Magnetic Stripe Read / Write Commands

MAG_READ	Send	Len
Transmit	0xB0	1
	MODE 0: ISO mode 1: Transparent mode	1
Receive	0x00 (stat_OK) or error code	1
	TRACK1_STATUS	1
	TRACK2_STATUS	1
	TRACK3_STATUS	1
Description	This command is not necessary when the ARM mode is used. If a card is already in the reader, this command could be used to read the magnetic stripe data in both directions.	

MAG_GET_ISO_TRK	Send	Len
Transmit	0xB2	1
	TRACK (1,2 or 3)	1
Receive	0x00 (stat_OK) or error code	1
	DATA[0]-DATA[n]	n
Description	Get track data This command performs the ISO decoding from the bits read from the requested track, and then returns the response into an ASCII format. Standard ISO format specifications are: - Track 1: 210 bpi; 6 bits per char + odd parity bit. Length < 79 char. Start char is '%' (45h); End char is '?' (1Fh) - Track 2: 75 bpi; 4 bits per char + odd parity bit. Length < 40 char. Start char is ';' (0Bh); End char is '?' (1Fh) - Track 3: 210 bpi; 4 bits per char, odd parity bit. Length < 107 char. Start char is ';' (0Bh); End char is '?' (1Fh)	

DEFINE_MAG_DATA	Send	Len
Transmit	0xB7	1
	TRACK (1,2 or 3)	1
	LEN	1
	DATA[0]-DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
Description	Prepare data to write. If the magnetic data is not valid, a stat_MAG_DATA_NOT_VALID status will be returned.	

MAG_WRITE	Send	Len
Transmit	0xB6	1
	MODE 0: ISO mode 1: Transparent mode	1
Receive	0x00 (stat_OK) or error code	1
Description	Write predefined data on magstripe card If ISO mode is selected, the START, STOP and LRC bytes are inserted automatically. If no data is defined for a specific track, the track will not be modified.	

GET_HICO_FLAG	Send	Len
Transmit	0xB8	1
Receive	0x00 (stat_OK) or error code	1
	FLAG 0: LOCO 1: HICO	1
Description	0: HICO flag off, reader will write LOCO 1: HICO flag on, reader will write HICO	

SET_HICO_FLAG	Send	Len
Transmit	0xB9	1
	FLAG 0: LOCO 1: HICO	1
Receive	0x00 (stat_OK) or error code	1
Description	0: turn HICO flag off, reader will write LOCO 1: turn HICO flag on, reader will write HICO In order to write onto a HICO card the reader has also to be equipped with a special HICO electronic board. Turning on the HICO switch on a LOCO reader does not affect the process of writing. When writing HICO cards only one track should be written at a time. Otherwise the reader will require high current (>2A).	

3.4 Chip Card Commands

CHIP_ACTIVATE	Send	Len
Transmit	0xBF	1
	AS_EMV (Optional) 0: Memory card 1: ISO Asynch. card 2: EMV2000 Asynch. Card 3: ISO Asynch. 3.3V card	1
Receive	0x00 (stat_OK) or error code	1
	ATR[0]-ATR[n]	n
Description	Chip card activation When a card is present inside the reader, or a SAM is introduced into its socket, this command has to be used to power on, and to activate the ICC (chip card or SAM). If this activation is successful, the reader returns the ATR frame from the newly activated chip. The parameter is optional; it permits to set the specification to be applied for the ICC activation. • 0: to activate only synchronous cards (memory card or I²C card) • 1: to activate only asynchronous ISO card conformed to ISO 7816-3 • 2: to activate payment chip card, with respect of the standard EMV (Europay, Mastercard and Visa) specification. This parameter is mandatory to use the reader as an "EMV – Level 1" device. When this parameter is **not** given, the activation sequence is performed as following: • Power ON the chip and start the asynchronous clock signal • Decode the ATR send by the chip (if the chip is a micro-processor unit) • If the ATR is valid, its value is send as response to the host. This ATR always starts with the character 3Bh or 3Fh • If the ATR is not valid (transmission error, or data outside specifications range) or if the chip does not respond, the chip is deactivated (powered off) • If the ATR was not valid, the chip is activated once more. The clock line is activated in synchronous mode (50 kHz) • Decode the data send by the chip (if the chip is a synchronous memory card) • If this ATR data is valid, its value (which is always 4-bytes length) is returned as response to the host. • If no data is returned by the chip, the terminal deactivates it • If the chip returned no data, the chip is then activated for a third time. The reader uses the I2C protocol to get data from the card (if the chip is a I²C memory card) • If the chip is responding at this request, the terminal returns an 8-bytes length frame. (Normally data read from address 0000h to 0007h). • If there's still no valid response, the reader abort activation sequences, and the error code **stat_ERROR** is transmitted to the host. The activation is always first performed, by using the asynchronous protocol (ISO 7816). If this first activation fails, then a second activation is performed, using the synchronous protocol (for memory cards). If this second activation fails, then a third activation is performed, using the Philips I²C protocol. If the mode "Auto ATR" is activated (in the reader's configuration data), the chip activation procedure is automatically performed, as soon as a card is detected into the seated position. If the activation is successful, the ATR is automatically returned to the host, and the card is locked.	

	The user has to take care of the returned ATR; he must set the appropriate memory card protocol (see command **MEMCARD_SET**) for future synchronous data transfer.

CHIP_DEACTIVATE	**Send**	**Len**
Transmit	0xC0	1
Receive	0x00 (stat_OK) or error code	1
Description	This command is used to power down the chip card, as defined into ISO 7816-3 and EMV.	

CHIP_TRANSFER	**Send**	**Len**
Transmit	0xBB	1
	CAPDU[0]-CAPDU[n]	n
Receive	0x00 (stat_OK) or error code	1
	RAPDU[0]-RAPDU[n]	n
Description	This request is used to pass an APDU command to an ICC (Asynchronous chip: the user chip card, or a secure module) and get the APDU response returned by the ICC. APDU format are conformed to the structure defined in ISO 7816-4 The APDU command shall be included in the request: **CHIP_TRANSFER CLA INS P1 P2 Lc …Data… Le** Where: CLA, INS, P1, and P2 are mandatory "Lc", "Data" and "Le" are optional. Action: • Clear ICC data buffer • Check the given C-APDU syntax. If it is not correct, the error code **stat_INVALID** is returned. (No communication with the chip is performed) • Check that the chip card is still active. If the card has been prematurely removed, or an electrically shutdown has occurred, the error code **stat_INVALID** is returned. • Perform the data exchange with the chip card • If successful, the returned data (if present) and the status word SW1 / SW2 are returned to the host, as R-APDU: **…Response Data… SW1 SW2** • If unsuccessful, the card is automatically deactivated, and the error code **stat_ERROR** is returned. This command is independent of the card protocol: T=0 or T=1. **Complement information about ICC transfer** • The reader never performs any **PTS** (Protocol Type Selection) sequence. • If the inserted card is conformed to protocol T=1, the reader automatically performs the SBlock[IFSD] sequence just before the first data exchange. (This sequence sets the reader buffer size to 254 bytes, as defined into EMV2000)	

CHIP_SELECTION	**Send**	**Len**
Transmit	0xBC	1
	DEV 0: Card 1: SAM1 2: SAM2 n: SAMn	1
Receive	0x00 (stat_OK) or error code	1
Description	This command tells the reader which connector is to be used with	

	succeeding chip commands. This command does not alter the link with the previous one. Note: After a reader reset/restart, the default connector is always the user card one.

MEMCARD_CMD	Send	Len
Transmit	0xBD	1
	DATA[0]-DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
	DATA[0]-DATA[n]	n
Description	To be conformed to ISO format (like with microprocessor card), requests for memory cards also use the APDU format The APDU command shall be included in the request: **cMEM_TRANSFER CLA INS ADDR_H ADDR_L Lc … Data …** Or **cMEM_TRANSFER CLA INS ADDR_H ADDR_L Le** Or **cMEM_TRANSFER CLA INS ADDR_H ADDR_L Lc … Data … Le** With: **CLA** is mandatory, but its value may always be null. This parameter is ignored. **INS** defines the command to be performed. The defined values are • **INS_READ_BINARY,** coded B0h: to read, and get data bytes from the card. The number of data to be returned has to be set in **Le**. The reading will start from the address given by **ADDR.** • **INS_WRITE_BINARY,** coded D0h: to write data onto the card. The writing will start at the address given by **ADDR,** and **Lc** defines the length of the **Data** part. • **INS_PASS_VERIFY,** coded 20h: to unlock the card's write-protection. The length of the given password has to be set in **Lc**. The given password (or PIN code) will be set into the **Data** part. • **INS_PASS_CHANGE,** coded 88h: to modify the protection code. The length of the new password has to be set in **Lc**. The new password (or PIN code) will be set into the **Data** part. As response, the password, read back from the card, is returned. The user has to take care that the returned password is identical to the given one. • **INS_RESTORE_DATA,** coded C0h: to be used for pre-payment card (SLE4406) with units. This command is used to restore bits (units). It shall be used with one data byte. This byte will be written at the given address, to automatically erasing the 8 bits from the next counter (at the following address) • **INS_SET_PROTECTION,** coded C1h: to set a permanent write protection on some memory areas. To protect an area, it is necessary to input again the data. The protection flag will only be set if the input data and the card data are identical. • **INS_READ_PROTECTION,** coded C2h: to read the protection status from each memory area on the card. The byte **'s'** (73h) is returned to indicate a protected area. A null byte (00h) is returned for each address that is still able to be written **ADDR** defines the address (first High byte, then Low byte), where readings or writings operations have to start. **Lc:** Length of the DATA part **Data**: Data to be written onto the card **Le**: Length of the expected data the card has to return as response Action: • Check the given C-APDU syntax. If it is not correct, the error code

	stat_INVALID is returned. (no communication with the card is performed) • Check that the card is still active. If the card has been prematurely removed, or an electrically shutdown has occurred, the error code **stat_INVALID** is returned. • The reader interprets the request, then perform the data exchanges with the memory card, using the communication protocol set by the command **cMEM_CARDTYPE** • If successful, the returned data given by the card, followed by the status word SW are returned to the host, as an R-APDU. • If unsuccessful, the terminal only returns a status word; its value indicates the type of error. The returned Status Word **SW** takes one of the following values: • **SW_OK** coded 9000h means the read/write process has been successfully performed. • **SW_INS_ERROR** coded 6E00h : means that the given command cannot be performed, or is not compatible with the current selected card type. • **SW_ADDR_ERROR**, coded 6E01h : means that the given address, where writing has to begin, is not allowed. This case may appears with some I²C cards, where data block writings may start at the beginning of the memory page. • **SW_LC_ERROR**, coded 6E02h : means that the given parameter (DATA part) is not compatible with the associated command (INS). Example: using **INS_WRITE_BINARY** with no data, so **Lc** is null • **SW_LE_ERROR**, coded 6E03h : means that the value **Le** isn't compatible with the associated command. Example: using **INS_READ_BINARY** with **Le** being null • **SW_COMMAND_EXE**, coded 6E10h : means that a problem occurs during the command transmission to the card. Transmission is then aborted • **SW_PROCESS_EXE**, coded 6E11h : means that a problem occurs during the processing phase, by the card. In this case, the read/write process is aborted. • **SW_LOCKED**, coded 6E12h : means that the writing command is not allowed, because, the PIN code has not previously been successfully checked. • **SW_EXE_ERRORS**, coded 64*XX*h : This error code is generally returned when the requested action has not been completely performed. The 2nd SW2 code (*XX*h) contains the number of errors that has occurred during the process phase. • **SW_PASSWD_ERRORS**, coded 65*XX*h : This error code is returned when the password (PIN) verification has failed. The 2nd byte SW2 contains the remaining PIN-trial count. When the value 6500h is returned, it means that the card is definitely locked.

MEMCARD_SET	**Send**	**Len**
Transmit	0xBE	1
	TYPE 0: SLE4406, SLE4436, ST1305 1: SLE4418, SLE4428, Gemplus GPM8K 2: SLE4404 3: SLE4432, SLE4442 4: I2C Card 1 byte addressing 5: I2C Card 2 bytes addressing	1
	PAGE_SIZE (4, 8,16, 32 ,64 ,128)	1
Receive	0x00 (stat_OK) or error code	1
Description	It has to be used to set the type of protocol for the communication with the inserted memory card The optional parameter <Page size> defines the memory size of one	

	page from i²c cards. The following default values are set: • For type '4': Size = 0x08 (for 8 bytes/page) • For type '5': Size = 0x20 (for 32 bytes/page)

3.5 PCD Commands

PCD_KILL	Send	Len
Transmit	0x1F	1
Receive	0x00 (stat_OK) or error code	1
Description	Reset the PCD, set RF off	

PCD_TYPEA_INIT	Send	Len
Transmit	0x20	1
Receive	0x00 (stat_OK) or error code	1
Description	Initialize the PCD with Type A configuration, set RF on	

PCD_TYPEB_INIT	Send	Len
Transmit	0x50	1
Receive	0x00 (stat_OK) or error code	1
Description	Initialize the PCD with Type B configuration, set RF on	

PCD_WRITEE2	Send	Len
Transmit	0x21	1
	ADDR[0]-ADDR[1] (MSB First)	2
	LEN	1
	DATA[0]…DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
Description	Update PCD's internal EEPROM	

PCD_READE2	Send	Len
Transmit	0x22	1
	ADDR[0]-ADDR[1] (MSB First)	2
	LEN	1
Receive	0x00 (stat_OK) or error code	1
	DATA[0]…DATA[n]	n
Description	Read from PCD's internal EEPROM	

PCD_RESETPHASE	Send	Len
Transmit	0x23	1
Receive	0x00 (stat_OK) or error code	1
Description	Run the reset and initialization phase	

PCD_RF_RESET	Send	Len
Transmit	0x25	1
	ms (milliseconds)	1
Receive	0x00 (stat_OK) or error code	1
Description	Reset the RF field	

PCD_RF_OFF	Send	Len
Transmit	0x26	1
Receive	0x00 (stat_OK) or error code	1
Description	Switch RF field off	

WRITERC	Send	Len
Transmit	0x57	1
	ADDR	1
	DATA	1
Receive	0x00 (stat_OK) or error code	1
Description	Write a byte to PCD's given register.	

READRC	Send	Len
Transmit	0x58	1
	ADDR	1
Receive	0x00 (stat_OK) or error code	1
	DATA	1
Description	Read a byte from PCD's given register.	

PCD_SET_ATTRIB	Send	Len
Transmit	0x29	1
	DSI (Divisor Send Integer)	1
	DRI (Divisor Receive Integer)	1
Receive	0x00 (stat_OK) or error code	1
Description	Set RF communication baud rate	

PICC_EXCHANGE_BLOCK	Send	Len
Transmit	0x2A	1
	APPEND_CRC	1
	TIMEOUT[0]-TIMEOUT[3]	4
	DATA_LEN	1
	DATA[0]-DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
	DATA[0]-DATA[n]	n
Description	Transparent communication with the PICC	

PCD_ACTIVATE_ANT	Send	Len
Transmit	0x61	1
	ANT (0: Both, 1 or 2)	1
Receive	0x00 (stat_OK) or error code	1
Description	Activate selected antenna	

PICC_REMOVE	Send	Len
Transmit	0x62	1
Receive	0x00 (stat_OK) or error code	1
Description	Wait until the tag is out of the field	

3.6 ISO14443-A Commands

PICC_REQUEST	Send	Len
Transmit	0x10	1
	REQA: 0x26 (Request idle) WUPA: 0x52 (Request all)	1
Receive	0x00 (stat_OK) or error code	1
	ATQA[0]-ATQA[1] (Request code)	2
Description	Send request command	

PICC_ANTICOLL	Send	Len
Transmit	0x11	1
	SEL_CODE (Anti-collision level) Level1: 0x93 Level2: 0x95 Level3: 0x97	1
	nbits (known bits)	1
Receive	0x00 (stat_OK) or error code	1
	UID[0]-UID[3]	4
Description	Get UID from one of the PICCs	

PICC_SELECT	Send	Len
Transmit	0x12	1
	SEL_CODE (level) Level1: 0x93 Level2: 0x95 Level3: 0x97	1
	UID[0]-UID[1]	4
Receive	0x00 (stat_OK) or error code	1
	SAK (Select ACK)	1
Description	Activate a PICC by selecting the UID	

PICC_ANTICOLLSEL	Send	Len
Transmit	0x19	1
	BR (Baud rate) Default: 0	1
Receive	0x00 (stat_OK) or error code	1
	UID_LEN	1
	UID[0]-UID[n]	n
	SAK (Select ACK)	1
Description	Anti-collision and select performed together	

PICC_HALTA	Send	Len
Transmit	0x1C	1
Receive	0x00 (stat_OK) or error code	1
Description	Set PICC to Halt state	

PICC_ACTIVATE_IDLE	Send	Len
Transmit	0x59	1
	BR (Baud rate) Default: 0	1
Receive	0x00 (stat_OK) or error code	1
	ATQA[0]-ATQA[1] (Request code)	2
	SAK (Select ACK)	1
	UID_LEN	1
	UID[0]-UID[n]	n

Description	Activates the idle PICC with the given baud rate

PICC_ACTIVATE_WAKEUP	**Send**	**Len**
Transmit	0x5A	1
	BR (Baud rate) Default: 0	1
	UID_LEN	1
	UID[0]-UID[n]	n
Receive	0x00 (stat_OK) or error code	1
	ATQA[0]-ATQA[1] (Request code)	2
	SAK (Select ACK)	1
Description	Wakes the halted UID up with the given baud rate	

PICC_DO_PPS	**Send**	**Len**
Transmit	0x2B	1
	DSI (Data Send Integer) 0: 106 kbit/s 1: 212 kbit/s 2: 424 kbit/s 3: 848 kbit/s	1
	DRI (Data Receive Integer) 0: 106 kbit/s 1: 212 kbit/s 2: 424 kbit/s 3: 848 kbit/s	1
Receive	0x00 (stat_OK) or error code	1
Description	Perform protocol parameter selection	

PICC_REQUEST_ATS	**Send**	**Len**
Transmit	0x3A	1
Receive	0x00 (stat_OK) or error code	1
	ATS_LEN	1
	ATS[0]-ATS[n]	n
Description	Activate an ISO14443-4 compliant PICC	

3.7 ISO14443-B Commands

PICC_REQUESTB	Send	Len
Transmit	0x51	1
	iswup 0: Request 1: Wakeup	1
	afi	1
	num_slots	1
Receive	0x00 (stat_OK) or error code	1
	ATQB[0]-ATQB[11] (Request code)	12
Description	Send request command	

PICC_SLOTMARKER	Send	Len
Transmit	0x5F	1
	num_slots	1
Receive	0x00 (stat_OK) or error code	1
	ATQB[0]-ATQB[11] (Request code)	12
Description	Send slot marker command	

PICC_ATTRIB	Send	Len
Transmit	0x52	1
	UID[0]-UID[3]	4
Receive	0x00 (stat_OK) or error code	1
	ATA	1
Description	Activate a Type B PICC	

PICC_ATTRIB_HBR	Send	Len
Transmit	0x2C	1
	DSI (Data Send Integer) 0: 106 kbit/s 1: 212 kbit/s 2: 424 kbit/s 3: 848 kbit/s	
	DRI (Data Receive Integer) 0: 106 kbit/s 1: 212 kbit/s 2: 424 kbit/s 3: 848 kbit/s	1
	UID[0]-UID[3]	4
Receive	0x00 (stat_OK) or error code	1
	ATA	1
Description	Activate a Type B PICC with higher baud rate	

PICC_HALTB	Send	Len
Transmit	0x53	1
Receive	0x00 (stat_OK) or error code	1
Description	Set PICC to Halt state	

3.8 ISO14443-4 (T=CL) Commands

PICC_DETECT	Send	Len
Transmit	0x5B	1
Receive	0x00 (stat_OK) or error code	1
	PICC_TYPE 0x70: TypeA PICC 0x71: TypeB PICC	1
	Type A: ATQA[0]-ATQA[1] + SAK + UID[0]-UID[n] + ATS[0]-ATS[m] Type B: ATQB[0]-ATQB[11] + ATTRIBRESPONSE	n+m+5 13
Description	Detects and activates the PICC	

PICC_SEND_BLOCK	Send	Len
Transmit	0x54	1
	LEN[0]-LEN[1] (MSB first)	2
	DATA[0]-DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
	RCVDATA[0]-RCVDATA[n]	n
Description	Send and receive data	

PICC_SEND_ACK	Send	Len
Transmit	0x55	1
	ACK: 0x00 NACK: 0x10	1
Receive	0x00 (stat_OK) or error code	1
Description	Send and receive data	

PICC_SEND_REQ	Send	Len
Transmit	0x56	1
	DESELECT: 0x00 WTX: 0x30	1
	WTXM	1
Receive	0x00 (stat_OK) or error code	1
Description	Send T=CL request, the reader waits until the PICC is removed	

PICC_DESELECT	Send	Len
Transmit	0x86	1
Receive	0x00 (stat_OK) or error code	1
Description	T=CL deselect command	

3.9 ISO14443 mifare Commands

PICC_AUTHENT	Send	Len
Transmit	0x13	1
	MODE 0x60: Auth. with Key A 0x61: Auth. with Key B	1
	Key sector (0x00-0x0F	1
	Block number (0x00-0x3F)	1
Receive	0x00 (stat_OK) or error code	1
Description	Performs mifare authentication with stored keys	

PICC_AUTHENT_KEY	Send	Len
Transmit	0x14	1
	MODE 0x60: Auth. with Key A 0x61: Auth. with Key B	1
	KEYS[0]-KEYS[5]	6
	Block number (0x00-0x3F)	1
Receive	0x00 (stat_OK) or error code	1
Description	Performs mifare authentication with given keys	

PICC_READ	Send	Len
Transmit	0x15	1
	Block number (0x00-0x3F)	1
Receive	0x00 (stat_OK) or error code	1
	DATA[0]-DATA[15]	16
Description	Read 16 bytes from mifare block	

PICC_WRITE	Send	Len
Transmit	0x16	1
	Block number (0x00-0x3F)	1
	DATA[0]-DATA[15]	16
Receive	0x00 (stat_OK) or error code	1
Description	Write 16 bytes to mifare block	

PICC_WRITE4	Send	Len
Transmit	0x17	1
	Block number	1
	DATA[0]-DATA[3]	4
Receive	0x00 (stat_OK) or error code	1
Description	Write 4 bytes to mifare ultralight block	

PICC_VALUE	Send	Len
Transmit	0x18	1
	MODE 0xC0: Decrement 0xC1: Increment 0xC2: Restore	1
	ADDRESS (0x00-0x3F)	1
	VALUE[0]-VALUE[3]	4
	TRANSFER_ADDR (0x00-0x3F)	1
Receive	0x00 (stat_OK) or error code	1
Description	Perform a value operation	

PCD_LOADKEYE2	Send	Len
Transmit	0x1E	1
	KEY_TYPE 0x60: Key A 0x61: Key B	1
	SECTOR (0x00-0x0F)	1
	DATA[0]-DATA[5]	6
Receive	0x00 (stat_OK) or error code	1
Description	Load given keys in PCD's secure eeprom	

3.10 Card Dispenser Commands

DISPENSER_DO_MOVE	Send	Len
Transmit	0x90	1
	*restricted move commands 0x01: move to POS1 0x02: move to POS2 0x03: move to POS3 (Eject card) 0x04: move to chip position ** 0x05: intake card 0x06: mechanic to chip position 0x07: mechanic to transport position 0x08: mechanic to capture card 0x09: move to contactless antenna ***maintenance move commands 0x11: move to POS1 0x12: move to POS2 0x13: move to POS3 ***22x110 card dispenser commands 0x20: dispense 1 card 0x21: dispense 5 cards 0x22: dispense 10 cards 0x23: dispense 15 cards	1
Receive	0x00 (stat_OK) or error code	1
Description	Moves the card to the defined position	

* To avoid card jams, it is recommended that you use the restricted move commands. The following movements are allowed;
"Pos1 -> Pos2", "Pos2 -> Pos3" and *"Pos3 -> Pos1"*
The following movements are inhibited;
"Pos3->Pos2", "Pos2->Pos1" and *"Pos1->Pos2/3"*, only when a card is at front.

** If the card is moved to the chip position, the chip mechanic will be automatically moved up, to contact the chip module.

*** In case of card jam, the maintenance commands could be used without restrictions, to overcome the failure.

DISPENSER_SWITCH_STAT	Send	Len
Transmit	0x91	1
Receive	0x00 (stat_OK) or error code	1
	STAT[0]-STAT[1]	2
Description	Read switch states If the "Auto switch report" mode is active, the switch status word is automatically sent to the host, each time a switch position change is detected. <SWITCH_STAT1> 0x00: if the reader has got less than 8 switches Bit7…….Bit0 S15…S8 <SWITCH_STAT2> Bit7…….Bit0 S7…S1	

3.11 External Control Commands

SET_RELAY	Send	Len
Transmit	0x88	1
	P1 0: Both relays are OFF 1: Relay1 is ON, Relay2 is OFF 2: Relay1 is OFF, Relay2 is ON 3: Both relays are ON	1
Receive	0x00 (stat_OK) or error code	1
Description	This command activates the selected relay output.	

ACT_RELAY	Send	Len
Transmit	0x40	1
	MS[0]-MS[1] (MSB First)	2
Receive	0x00 (stat_OK) or error code	1
Description	This command triggers the relay1 “MS” milliseconds long.	

SET_LED	Send	Len
Transmit	0x41	1
	P1 0: Both LEDs are OFF 1: LED1 is ON, LED2 is OFF 2: LED1 is OFF, LED2 is ON 3: Both LEDs are ON	1
Receive	0x00 (stat_OK) or error code	1
Description	This command activates the selected LED output.	

GET_INPUT	Send	Len
Transmit	0x87	1
Receive	0x00 (stat_OK) or error code	1
	INPUT 0: Both pins are LOW 1: Pin1 is HIGH, Pin2 is LOW 2: Pin2 is HIGH, Pin1 is LOW 3: Both pins are HIGH	1
Description	This command sets the LED pins as input and returns the states.	

WRITE_LCD	Send	Len
Transmit	0x32	1
	LCD_LINE 0: Init and clear LCD 1: Line1 2: Line2 3: Line3 4: Line4	1
	DATA[0]-DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
Description	This command controls a dot-matrix LCD display. Up to 4x20 character LCDs are supported.	

SET_BUZZER	Send	Len
Transmit	0x42	1
	P1 0: Short beep 1: Long beep 2: Double beep	1
Receive	0x00 (stat_OK) or error code	1
Description	This command activates the internal buzzer.	

3.12 NFC Reader Commands

NFC_CMD	Send	Len
Transmit	0x4F	1
	CDATA[0]…CDATA[n]	n
Receive	0x00 (stat_OK) or error code RDATA[0]…RDATA[n]	1 n
Description	This command sends the given frame to the NFC controller and returns the answer.	

This command can be used to communicate transparently with the PN532 NFC controller.

Mifare Card Activation Example:

Command **InListPassiveTarget**
(7.3.5 InListPassiveTarget PN532 User Manual UM0701-02)

Send:
4F 4A0100

4F: NFC Command
4A: Command Code InListPassiveTarget
01: MaxTg
00:106kbps TypeA

Receive:
00 0101000408043C05AC67

00: Status OK
01: NbTg (number of initialized tags)
01: Tag number one
0004: SENS_RES
08: SEL_RES (mifare 1K)
04: UID length
3C05AC67: UID

As you see, the first byte “0xD4” of the NFC frame will not be added to the send buffer. The reader adds this byte internally. The receive frame also doesn’t include the “0xD5” and “0x4B” bytes. The reader checks these bytes internally.

4 READER RESPONSE MESSAGES

The reader sends in some cases automatic messages to inform the host. To distinguish the events from the response messages, the status bytes are divided into two areas. The status values from 0x00 to 0x2F are reserved to status codes for command responses. All other values greater than 0x30 are considered as event reports.

4.1 Status Codes <0x30

Codes which are returned as command status.

Status	Code	Description
stat_OK	0x00	Command successfully performed
stat_NO_TAG_ERR	0x01	No tag in the field or no response
stat_COLL_ERR	0x02	There is more than one tag in the field. According to the EMVCo specifications, an error during the activation is also considered as a collision
stat_AUTH_ERR	0x03	mifare sector authentication error
stat_PROTOCOL_ERR	0x04	Protocol error will be reported when the coding of the frame is not compliant to the EMVCo specifications
stat_TRANSMISSION_ERR	0x05	Transmission error will be reported when the received frame includes; crc error, parity error, coding error, framing error or bit count error. This error mostly happens when the tag enters the operating field. While polling about every 10ms, the reader can catch the tag outside the safe operating distance. In this case, an RF reset and a second activation should be performed
stat_TIMEOUT_ERR	0x06	Timeout error will be reported when the tag doesn't answer to the APDU. In this case the tag should be reactivated
stat_BUFFER_OVERFLOW_ERR	0x07	The received frame is too long
stat_ADR_OVERFLOW_ERR	0x08	The given address + length overflows
stat_UNKNOWN_CMD	0x09	This command is not supported
stat_ERROR	0x0A	Non categorized error
stat_COMM_TIMEOUT	0x0B	Communication timeout
stat_USB_DEVICE_ERROR	0x0C	USB connection or communication error
stat_BOOT_ERROR	0x0D	Non categorized boot command error
stat_BOOT_OVERFLOW	0x0E	Boot command length or address overflows
stat_BOOT_TIMEOUT	0x0F	Boot command timeout error
stat_NO_USER_CODE	0x10	User firmware doesn't exists
stat_INVALID	0x11	Invalid operation
stat_NO_DATA	0x12	No magnetic data decoded
stat_UNAVAILABLE	0x13	Cannot perform this command
stat_MAG_DATA_NOT_VALID	0x14	Magstripe data not valid
stat_CARD_JAM	0x15	Card jam within motorized reader
stat_BCC_ERROR	0x16	Received frame has a wrong BCC
stat_XC_TOUT	0x17	FPGA timeout
stat_NFC_ERR	0x18	NFC controller communication error

4.2 Event Codes >0x30

Event	Code	Description
event_REMOVED	0x30	Contactless tag is removed
event_PICC_ACK	0x31	Contactless tag is detected
event_USER_MODE_START	0x32	Application firmware has started
event_BOOT_MODE_START	0x33	Boot firmware has started
event_SWITCH_REPORT	0x34	Automatic switch report
event_MEM_CARD_ACTIVATED	0x35	Memory smart card is activated
event_CHIP_CARD_ACTIVATED	0x36	Smart card is activated
event_MDB_DATA	0x37	MDB data packet
event_MDB_STATE	0x38	MDB state change

event_KEY_STROKE	0x39	Key entered
event_ARM_ACK	0x3A	Arm to read ACK
event_ARM_NACK	0x3B	Arm to read NACK
event_NO_MAG_CARD	0x3C	No magnetic tracks detected
event_INTAKE_ACK	0x3D	Intake ACK
event_INTAKE_NACK	0x3E	Intake NACK
event_PICC_PPSE	0x3F	Contactless card detected and activated

4.3 Events Reported by the Reader

event_SWITCH_REPORT	**Send**	**Len**
Sent by the Reader	0x34	1
	<SWITCH_STAT1> 0x00: if the reader has got less than 8 switches Bit 0: S9 Bit 1: S10 Bit 2: S11 Bit 3: S12 Bit 4: S13 Bit 5: S14 Bit 6: S15 Bit 7: S16	1
	<SWITCH_STAT2> Bit 0: S1 Bit 1: S2 Bit 2: S3 Bit 3: S4 Bit 4: S5 Bit 5: S6 Bit 6: S7 Bit 7: S8	1
Description	Automatic switch report	

event_ARM_(N)ACK	**Send**	**Len**
Sent by the Reader	0x3A: ARM_ACK 0x3B: ARM_NACK	1
	TRACK1_STATUS	1
	TRACK2_STATUS	1
	TRACK3_STATUS	1
Description	ARM ack	

event_PICC_ACK	**Send**	**Len**
Sent by the Reader	0x31: PICC_ACK 0x30: PICC_REMOVED	1
	TAG_INFO SL3 | SL2 | SL1 | SL0 | COLL | ISO4 | TYPE | AA BIT 7 | BIT 6 | BIT 5 | BIT 4 | BIT 3 | BIT 2 | BIT 1 | BIT 0 AA: Activated antenna. 0 for Ant1 and 1 for Ant2 TYPE: Detected PICC type. 0 for Type A PICC and 1 for Type B PICC ISO4: ISO14443-4 compliant PICC COLL: Collision detected, at least one more tag in the field SL: UID length (default: 4, extended UID: 7 or 10)	1
	UID[0]-UID[n]	n
Description	PICC acknowledged or PICC removed	

5 BOOTLOADER

5.1 Boot882 Secondary Bootloader

Boot882 Bootloader is a piece of code which allows user's application code to be downloaded without the need of resetting the hardware.

Since Boot882 Bootloader is an add-on option, the blank 882 Module has to be programmed first with Boot882 firmware. This can be done via RS232 Interface.

Boot882 supports RS232, USB, SPI and TCP/IP interfaces and resides with 28KB in Flash memory. The Application Firmware for the 882 Module can utilize the remaining 100KB of flash memory.

Flash Memory

ROM Bootloader (On-Chip)	**Address** 0x7D000	Runs first and checks the dedicated hardware pin to determine if the entry is valid.
Application Firmware Flash Program Memory 100 KB	0x7000	Application Firmware can jump to the Boot882 Firmware.
Boot882 Bootloader Flash Program Memory 28 KB	0x0000	Boot882 can program the entire 100KB user space and jump to the start vector 0x7000.

- ***On-Chip ROM Bootloader:*** This code is executed every time the part is powered on or reset. A LOW level after reset at the dedicated hardware pin (RS232-CTS pin or a switch on the PCB) is considered as an external request to start the bootloader. Since the software entry & exit mechanism are not supported, it will only be used to program the Boot882 code.
- ***Boot882 Advanced Bootloader:*** This code is executed first after start-up. Since the communication parameters are stored in external flash memory, the application and the boot firmware have the same communication settings. Using the **Boot Command Set**, it is possible to erase, program and verify the user program memory. It is also possible to jump to the user code without the need of a hardware reset.
- ***Application Firmware (User Code):*** This code starts at address 0x7000 and can only be executed from the Boot882 code.

5.1.1 Start-Up Sequence

After Power-Up:

- First, the On-Chip ROM Bootloader will check the dedicated hardware pin (RS232-CTS pin or a switch on the PCB) for a valid entry. It this entry is not valid, it will jump to address 0x0000.
- At address 0x0000 resides the Boot882 Firmware. The Boot882 firmware first reads the communication settings from the external flash memory and initializes the peripheral modules.
- If an application firmware is detected, the Boot882 Firmware waits 3 seconds for a host request.
- During this period the red LED flashes fast.
- If the reader doesn't receive a request during this time, it will check the presence of an application firmware and jump to it.
- If no application firmware is present, the Boot882 will continue running.
- If the Boot882 jumps to the application firmware the red LED will flash normal.

5.2 Boot882 Commands

GET_MODE	**Send**	**Len**
Transmit	0xA9	1
Receive	0x00 (stat_OK) or error code	1
	Mode **0x32**: User Mode is running **0x33**: Boot Mode is running	1
Description	This command will be used for both user mode and boot mode. It returns the mode which is currently running.	

BOOT_EXE_BOOT_CODE	**Send**	**Len**
Transmit	0xA8	1
Receive	0x00 (stat_OK) or error code	1
Description	This command is a user mode command and will be used to jump to the boot code. This is the software entry mechanism to the bootloader.	

BOOT_ERASE	**Send**	**Len**
Transmit	0xA0	1
Receive	0x00 (stat_OK) or error code	1
Description	This command erases the complete user program memory.	

BOOT_BLANK_CHECK	**Send**	**Len**
Transmit	0xA1	1
Receive	0x00 (stat_OK) or error code	1
Description	This command verifies that the user memory is successful erased.	

BOOT_PROGRAM	**Send**	**Len**
Transmit	0xA2	1
	OFFSET[0]..OFFSET[3] (MSB First)	3
	LEN[0]-LEN[1] (MSB First)	2
	DATA[0]…DATA[255]	256
Receive	0x00 (stat_OK) or error code	1
Description	This command is used to program the user flash memory.	

BOOT_READ	Send	Len
Transmit	0xA3	1
	OFFSET[0]..OFFSET[3] (MSB First)	3
	LEN[0]-LEN[1] (MSB First)	2
Receive	0x00 (stat_OK) or error code	1
	DATA[0]…DATA[255]	256
Description	This command is used to read from user flash memory.	

BOOT_CHECK_USER_CODE	Send	Len
Transmit	0xA7	1
Receive	0x00 (stat_OK) or error code	1
Description	This command verifies that a user code exists.	

BOOT_EXE_USER_CODE	Send	Len
Transmit	0xA6	1
Receive	0x00 (stat_OK) or error code	1
Description	This command is a boot mode command and will be used to jump to the user code. This is the software entry mechanism to the firmware.	

BOOT_RESET	Send	Len
Transmit	0xA5	1
Receive	0x00 (stat_OK) or error code	1
Description	This command is used to reset the boot mode.	

After verifying the firmware, a special sign should be programmed into the last blocks of the user memory. This sign is mandatory for the boot code to validate the firmware.

Last block start address: 0x1FF00
Last block data:
Data [0] = 0xFF
Data [1] = 0xFF
Data [2] = 0xFF
…
Data [249] = 0xFF
Data [250] = 0x11
Data [251] = 0x22
Data [252] = 0x33
Data [253] = 0x44
Data [254] = 0x55
Data [255] = 0x66

For details please refer to the Boot882.cpp source file!

6 ADDITIONAL COMMANDS

These commands are only supported by specific readers.

SET_BAUDRATE	Send	Len
Transmit	0x2F	1
	BR[0]-BR[2] (MSB First) 115200 Baud: 0x01C200 230400 Baud: 0x038400 460800 Baud: 0x070800	3
Receive	0x00 (stat_OK) or error code	1
Description	This command sets the given baudrate for the serial communication. If the baudrate is supported, it will be set after the stat_OK answer. Otherwise an error message will be reported.	

In order to improve the transaction time, we can use the EMV Detection. We call a regular SET_RUNNING_MODE command with a parameter of 3. The other values of parameters are discussed in the previous sections.

SET_RUNNING_MODE	Send	Len
Transmit	0x5D	1
	MODE 3: EMV Detection	1
Receive	*** Type A Card*** 0x3F (event_PICC_PPSE) ATQA[0]-ATQA[1] (Answer to request) SAK (Select Ack) UID (4, 7 or 10 Bytes) ATS (Answer to select) PPSE	*** 1 2 1 n (check ATQA) n (first byte=length) n
	*** Type B Card*** 0x3F (event_PICC_PPSE) ATQB[0]-ATQB[11] ATTRIB	*** 1 12 1
	*** non ISO-4 Card*** 0x31 (event_PICC_ACK) ATQA[0]-ATQA[1] (Answer to request) SAK (Select Ack) UID (4, 7 or 10 Bytes)	*** 1 2 1 n (check ATQA)
Description		

TypeA Card Example:

Set_Running_Mode EMV Detection:
Snd: 01 01 00 02 5D 03 5C
Rcv: 01 01 00 01 00 01 01

Detection:
01 00 46 3F 04 00 28 F7 79 FD C4 0D 78 80 80 02 00 73 C8 40 00 00 90 00 6F 2D 84 0E 32 50 41 59 2E 53 59 53 2E 44 44 46 30 31 A5 1B BF 0C 18 61 16 4F 07 A0 00 00 00 03 10 10 50 0B 56 49 53 41 20 43 52 45 44 49 54 90 00 30

event_PICC_PPSE: 3F
ATQA: 04 00 (Single UID: 4 Bytes)
SAK: 28
UID: F7 79 FD C4
ATS: 0D 78 80 80 02 00 73 C8 40 00 00 90 00 (0D = 13 total bytes)
PPSE: 6F 2D 84 0E 32 50 41 59 2E 53 59 53 2E 44 44 46 30 31 A5 1B BF 0C 18 61 16 4F 07 A0 00 00 00 03 10 10 50 0B 56 49 53 41 20 43 52 45 44 49 54 90 00

TypeB Card Example:

Set_Running_Mode EMV Detection:
Snd: 01 01 00 02 5D 03 5C
Rcv: 01 01 00 01 00 01 01

Detection:
01 00 35 3F 50 90 08 0B 08 00 00 00 00 00 81 81 00 6F 23 84 0E 32 50 41 59 2E 53 59 53 2E 44 44 46 30 31 A5 11 BF 0C 0E 61 0C 4F 07 A0 00 00 00 04 10 10 87 01 01 90 00 FC

event_PICC_PPSE: 3F
ATQB: 50 90 08 0B 08 00 00 00 00 00 81 81
ATTRIB: 00
UID: 98 08 0B 08
PPSE: 6F 23 84 0E 32 50 41 59 2E 53 59 53 2E 44 44 46 30 31 A5 11 BF 0C 0E 61 0C 4F 07 A0 00 00 00 04 10 10 87 01 01 90 00

PICC_SEND_BLOCK_BEEP	**Send**	**Len**
Transmit	0x4E	1
	LEN[0]-LEN[1] (MSB first)	2
	DATA[0]-DATA[n]	n
Receive	0x00 (stat_OK) or error code	1
	RCVDATA[0]-RCVDATA[n]	n
Description	Send and receive data. If the status word of the answer is 90 00, the LEDs and the buzzer will indicate the end of the transaction.	

PCD_STANDBY	**Send**	**Len**
Transmit	0x0E	1
	RF_ON_TIME 10: %10 On, %90 Off 50: %50 On, %50 Off 80: %80 On, %20 Off ..	1
Receive	0x00 (stat_OK) or error code	1
Description	This command sets the reader into standby mode to save power. The output RF signal will be activated only during the given on time.	

7 APPENDIX

7.1 Reader Initialization Sequence