



Advanced Card Systems Ltd.
Card & Reader Technologies

ACR38x

Smart Card Reader

CCID Reference Manual



Table of Contents

| | | |
|--------------------|---|-----------|
| 1.0. | Introduction | 3 |
| 2.0. | Features | 4 |
| 3.0. | Smart Card Support | 5 |
| 3.1. | MCU Cards | 5 |
| 3.2. | Memory-Based Smart Cards | 5 |
| 4.0. | Smart Card Interface | 6 |
| 4.1. | Smart Card Power Supply VCC (C1) | 6 |
| 4.2. | Programming Voltage VPP (C6)..... | 6 |
| 4.3. | Card Type Selection | 6 |
| 4.4. | Interface for Microcontroller-based Cards | 6 |
| 4.5. | Card Tearing Protection..... | 6 |
| 5.0. | Power Supply..... | 7 |
| 5.1. | Status LED..... | 7 |
| 6.0. | USB Interface..... | 8 |
| 6.1. | Communication Parameters | 8 |
| 6.2. | Endpoints..... | 8 |
| 7.0. | Communication Protocol | 9 |
| 8.0. | Commands..... | 11 |
| 8.1. | CCID Command Pipe Bulk-OUT Messages..... | 11 |
| 8.1.1. | PC_to_RDR_IccPowerOn..... | 11 |
| 8.1.2. | PC_to_RDR_IccPowerOff..... | 11 |
| 8.1.3. | PC_to_RDR_GetSlotStatus | 11 |
| 8.1.4. | PC_to_RDR_XfrBlock..... | 12 |
| 8.1.5. | PC_to_RDR_GetParameters..... | 12 |
| 8.1.6. | PC_to_RDR_ResetParameters | 12 |
| 8.1.7. | PC_to_RDR_SetParameters | 13 |
| 8.1.8. | CCID Bulk-IN Messages | 14 |
| 8.1.9. | RDR_to_PC_DataBlock..... | 14 |
| 8.1.10. | RDR_to_PC_SlotStatus | 15 |
| 8.1.11. | RDR_to_PC_Parameters..... | 15 |
| 8.2. | Commands Accessed via PC_to_RDR_XfrBlock..... | 16 |
| 8.2.1. | GET_READER_INFORMATION | 16 |
| Appendix A. | Supported Card Types..... | 17 |
| Appendix B. | Response Error Codes | 18 |

Tables

| | | |
|-----------------|-----------------------------------|----------|
| Table 1. | USB Interface Wiring | 8 |
|-----------------|-----------------------------------|----------|



1.0. Introduction

This document contains information regarding the ACR38x using the PC/SC platform. The ACR38x uses CCID interface to communicate with the USB port. CCID refers to the Device Class Specification for USB Chip/Smart Card Interface Devices that defines the communication protocol and commands for the USB chip-card interface devices.

The ACR38x acts as an interface for the communication between a computer (for example, a PC) and a smart card. Different types of smart cards have different commands and different communication protocols, which prevents, in most cases the direct communication between a smart card and a computer. The ACR38x establishes a uniform interface from the computer to the smart card for a wide variety of cards. By taking care of the card specific particulars, it releases the computer software programmer from getting involved with the technical details of the smart card operation, which are not relevant in many cases of the implementation of smart card system.

Note: Although the ACR38x is a true *card reader/writer* as it can read and write data from and to smart cards. The terms *reader* or *card reader* will be used indifferently to refer to the ACR38x. These designations are commonly used for this kind of devices.



2.0. Features

- USB 2.0 Full Speed Interface
- Plug and Play – CCID support brings utmost mobility
- Smart card reader:
 - Supports ISO 7816 Class A, B, and C (5 V, 3 V, 1.8 V) cards
 - Has read and write support to all microprocessor cards with T=0 or T=1 protocols
 - Supports memory cards:
 - I2C bus Protocol Cards (1K bits to 1024K bits)
 - Secure Memory Cards (Atmel AT88SC153 and AT88SC1608)
 - Memory Card with Secure Logic (AT88SC101/102/1003)
 - Supports SLE 4404/06/18/28/32/36/42, SLE 5518/28/32/36/42, SLE 6636
 - Supports PPS (Protocol and Parameters Selection)
 - Has Short Circuit Protection
- Compliant with the following standards:
 - EN60950/IEC60950
 - PC/SC
 - CCID
 - RoHS
 - CE
 - FCC
 - Microsoft WHQL



3.0. Smart Card Support

3.1. MCU Cards

ACR38x operates with any MCU card following either the T=0 or T=1 protocol.

3.2. Memory-Based Smart Cards

ACR38x works with several memory-based smart cards such as:

- Cards following the I2Cbus protocol (free memory cards) with maximum 128 bytes page with capability, including:
 - Atmel: AT24C01/02/04/08/16/32/64/128/256/512/1024
 - SGS-Thomson: ST14C02C, ST14C04C
 - Gemplus: GFM1K, GFM2K, GFM4K, GFM8K
- Cards with secure memory IC with password and authentication, including:
 - Atmel: AT88SC153 and AT88SC1608
- Cards with intelligent 1k bytes EEPROM with write-protect function, including:
 - Infineon: SLE4418, SLE4428, SLE5518 and SLE5528
- Cards with intelligent 256 bytes EEPROM with write-protect function, including:
 - Infineon: SLE4432, SLE4442, SLE5532 and SLE5542
- Cards with '104' type EEPROM non-reloadable token counter cards, including:
 - Infineon: SLE4406, SLE4436, SLE5536 and SLE6636
- Cards with Intelligent 416-Bit EEPROM with internal PIN check, including:
 - Infineon: SLE4404
- Cards with Security Logic with Application Zone(s), including:
 - Atmel: AT88SC101, AT88SC102 and AT88SC1003



4.0. Smart Card Interface

The interface between the ACR38x and the inserted smart card follows the specifications of ISO 7816-3 with certain restrictions or enhancements to increase the practical functionality of ACR38x.

4.1. Smart Card Power Supply VCC (C1)

The current consumption of the inserted card must not be higher than 50 mA.

4.2. Programming Voltage VPP (C6)

According to ISO 7816-3, the smart card contact C6 (VPP) supplies the programming voltage to the smart card. Since all common smart cards in the market are EEPROM-based and do not require the provision of an external programming voltage, the contact C6 (VPP) has been implemented as a normal control signal in the ACR38x. The electrical specifications of this contact are identical to those of the signal RST (at contact C2).

4.3. Card Type Selection

The controlling PC has to select the card type always, through the proper command sent to the ACR38x prior to activating the inserted card. This includes both the memory cards and MCU-based cards.

For MCU-based cards, the reader allows to select the preferred protocol, T=0 or T=1. However, this selection is only accepted and carried out by the reader through the PPS when the card inserted in the reader supports both protocol types. Whenever an MCU-based card supports only one protocol type, T=0 or T=1, the reader automatically uses that protocol type, regardless of the protocol type selected by the application.

4.4. Interface for Microcontroller-based Cards

For microcontroller-based smart cards, only the contacts C1 (VCC), C2 (RST), C3 (CLK), C5 (GND) and C7 (I/O) are used. A frequency of 4 MHz is applied to the CLK signal (C3).

4.5. Card Tearing Protection

The ACR38x provides a mechanism to protect the inserted card when it is suddenly withdrawn while it is powered up. The power supply to the card and the signal lines between the ACR38x and the card is immediately deactivated when the card is being removed. As a rule however, a card should only be removed from the reader while it is powered down, to avoid any electrical damage.

Note: The ACR38x never switch on the power supply to the inserted card by itself. The controlling computer through the proper command sent to the reader must explicitly do this.



5.0. Power Supply

The ACR38x requires a voltage of 5 V DC, 100 mA, regulated, power supply. The ACR38x Smart Card Reader gets power supply from a PC (through the cable supplied along with each type of reader).

5.1. Status LED

The Green LED on the front of the reader indicates the activation status of the smart card interface:

- **Flashing slowly (turns on 200ms for every 2 seconds)**
Indicates that the ACR38x is powered up and in the standby state; either the smart card has not been inserted or the smart card has not been powered up (if it is inserted).
- **Lighting up**
Indicates that the power supply to the smart card is switched on, i.e., the smart card is activated.
- **Flashing quickly**
Indicates there is communication between the ACR38x and a smart card.



6.0. USB Interface

The ACR38x is connected to a computer through a USB following the USB standard.

6.1. Communication Parameters

The ACR38x is connected to a computer through USB as specified in the USB Specification 2.0. The ACR38x is working in full speed mode, i.e. 12 Mbps.

| Pin | Signal | Function |
|-----|------------------|---|
| 1 | V _{BUS} | +5V power supply for the reader |
| 2 | D- | Differential signal transmits data between ACR38x and PC. |
| 3 | D+ | Differential signal transmits data between ACR38x and PC. |
| 4 | GND | Reference voltage level for power supply |

Table 1. USB Interface Wiring

Note: In order for the ACR38x (CCID) to function properly through USB interface, the ACS CCID driver or the Microsoft CCID Driver should be installed.

6.2. Endpoints

The ACR38x uses the following endpoints to communicate with the host computer:

- Control Endpoint** For setup and control purposes
- Bulk OUT** For command to be sent from host to ACR38x (data packet size is 64 bytes)
- Bulk IN** For response to be sent from ACR38x to host (data packet size is 64 bytes)
- Interrupt IN** For card status message to be sent from ACR38x to host (data packet size is 8 bytes)



7.0. Communication Protocol

The ACR38x (CCID) shall interface with the host thru USB connection. A specification, namely CCID, has been released within the industry defining such a protocol for the USB chip-card interface devices. CCID covers all the protocols required for operating smart cards and PIN.

The configurations and usage of USB endpoints on ACR38x (CCID) shall follow CCID Section 3. An overview is summarized below:

1. *Control Commands* are sent on control pipe (default pipe). These include class-specific requests and USB standard requests. Commands that are sent on the default pipe report information back to the host on the default pipe.
2. *CCID Events* are sent on the interrupt pipe.
3. *CCID Commands* are sent on BULK-OUT endpoint. Each command sent to ACR38x (CCID) has an associated ending response. Some commands can also have intermediate responses.
4. *CCID Responses* are sent on BULK-IN endpoint. All commands sent to ACR38x (CCID) have to be sent synchronously. (i.e. `bMaxCCIDBusySlots` is equal to 1 for ACR38x (CCID))

The supported CCID features by ACR38x (CCID) are indicated in its Class Descriptor:

| Offset | Field | Size | Value | Description |
|--------|-------------------------------------|------|-----------|---|
| 0 | <code>bLength</code> | 1 | 36h | Size of this descriptor, in bytes. |
| 1 | <code>bDescriptorType</code> | 1 | 21h | CCID Functional Descriptor type. |
| 2 | <code>bcdCCID</code> | 2 | 0100h | CCID Specification Release Number in Binary-Coded decimal. |
| 4 | <code>bMaxSlotIndex</code> | 1 | 00h | One slot is available on ACR38x (CCID) |
| 5 | <code>bVoltageSupport</code> | 1 | 07h | ACR38x (CCID) can supply 1.8V, 3.0V and 5.0V to its slot. |
| 6 | <code>dwProtocols</code> | 4 | 00000003h | ACR38x (CCID) supports T=0 and T=1 Protocol |
| 10 | <code>dwDefaultClock</code> | 4 | 00000FA0h | Default ICC clock frequency is 4MHz |
| 14 | <code>dwMaximumClock</code> | 4 | 00000FA0h | Maximum supported ICC clock frequency is 4MHz |
| 18 | <code>bNumClockSupported</code> | 1 | 00h | Does not support manual setting of clock frequency |
| 19 | <code>dwDataRate</code> | 4 | 00002A00h | Default ICC I/O data rate is 10752 bps |
| 23 | <code>dwMaxDataRate</code> | 4 | 0001F808h | Maximum supported ICC I/O data rate is 344 kbps |
| 27 | <code>bNumDataRatesSupported</code> | 1 | 00h | Does not support manual setting of data rates |
| 28 | <code>dwMaxIFSD</code> | 4 | 00000Feh | Maximum IFSD supported by ACR38x (CCID) for protocol T=1 is 254 |
| 32 | <code>dwSynchProtocols</code> | 4 | 00000000h | ACR38x (CCID) does not support synchronous card |
| 36 | <code>dwMechanical</code> | 4 | 00000000h | ACR38x (CCID) does not support special mechanical characteristics |



| Offset | Field | Size | Value | Description |
|--------|------------------------|------|-----------|--|
| 40 | dwFeatures | 4 | 00010030h | ACR38x (CCID) supports the following features: <ul style="list-style-type: none">• Automatic ICC clock frequency change according to parameters• Automatic baud rate change according to frequency and FI,DI parameters• TPDU level exchange with ACR38x |
| 44 | dwMaxCCIDMessageLength | 4 | 0000010Fh | Maximum message length accepted by ACR38x (CCID) is 271 bytes |
| 48 | bClassGetResponse | 1 | 00h | Insignificant for TPDU level exchanges |
| 49 | bClassEnvelope | 1 | 00h | Insignificant for TPDU level exchanges |
| 50 | wLCDLayout | 2 | 0000h | No LCD |
| 52 | bPINSupport | 1 | 00h | No PIN Verification |
| 53 | bMaxCCIDBusySlots | 1 | 01h | Only 1 slot can be simultaneously busy |



8.0. Commands

8.1. CCID Command Pipe Bulk-OUT Messages

The ACR38x (CCID) shall follow the CCID Bulk-OUT Messages as specified in CCID section 4. In addition, this specification defines some extended commands for operating additional features. This section lists the CCID Bulk-OUT Messages to be supported by ACR38x (CCID).

8.1.1. PC_to_RDR_IccPowerOn

Activate the card slot and return ATR from the card.

| Offset | Field | Size | Value | Description |
|--------|--------------|------|-----------|---|
| 0 | bMessageType | 1 | 62h | |
| 1 | dwLength | 4 | 00000000h | Size of extra bytes of this message |
| 2 | bSlot | 1 | | Identifies the slot number for this command |
| 5 | bSeq | 1 | | Sequence number for command |
| 6 | bPowerSelect | 1 | | Voltage that is applied to the ICC 00h – Automatic Voltage Selection 01h – 5 volts 02h – 3 volts |
| 7 | abRFU | 2 | | Reserved for future use |

The response to this message is the RDR_to_PC_DataBlock message and the data returned is the Answer to Reset (ATR) data.

8.1.2. PC_to_RDR_IccPowerOff

Deactivate the card slot.

| Offset | Field | Size | Value | Description |
|--------|--------------|------|-----------|---|
| 0 | bMessageType | 1 | 63h | |
| 1 | dwLength | 4 | 00000000h | Size of extra bytes of this message |
| 5 | bSlot | 1 | | Identifies the slot number for this command |
| 6 | bSeq | 1 | | Sequence number for command |
| 7 | abRFU | 3 | | Reserved for future use |

The response to this message is the RDR_to_PC_SlotStatus message.

8.1.3. PC_to_RDR_GetSlotStatus

Get the current status of the slot.

| Offset | Field | Size | Value | Description |
|--------|--------------|------|-----------|---|
| 0 | bMessageType | 1 | 65h | |
| 1 | dwLength | 4 | 00000000h | Size of extra bytes of this message |
| 5 | bSlot | 1 | | Identifies the slot number for this command |
| 6 | bSeq | 1 | | Sequence number for command |
| 7 | abRFU | 3 | | Reserved for future use |

The response to this message is the RDR_to_PC_SlotStatus message.



8.1.4. PC_to_RDR_XfrBlock

Transfer data block to the ICC.

| Offset | Field | Size | Value | Description |
|--------|-----------------|------------|-------|---|
| 0 | bMessageType | 1 | 6Fh | |
| 1 | dwLength | 4 | | Size of abData field of this message |
| 5 | bSlot | 1 | | Identifies the slot number for this command |
| 6 | bSeq | 1 | | Sequence number for command |
| 7 | bBWI | 1 | | Used to extend the CCIDs Block Waiting Timeout for this current transfer. The CCID will timeout the block after "this number multiplied by the Block Waiting Time" has expired. |
| 8 | wLevelParameter | 2 | 0000h | RFU (TPDU exchange level) |
| 10 | abData | Byte array | | Data block sent to the CCID. Data is sent "as is" to the ICC (TPDU exchange level) |

The response to this message is the RDR_to_PC_DataBlock message.

8.1.5. PC_to_RDR_GetParameters

Get slot parameters.

| Offset | Field | Size | Value | Description |
|--------|--------------|------|-----------|---|
| 0 | bMessageType | 1 | 6Ch | |
| 1 | DwLength | 4 | 00000000h | Size of extra bytes of this message |
| 5 | BSlot | 1 | | Identifies the slot number for this command |
| 6 | BSeq | 1 | | Sequence number for command |
| 7 | AbRFU | 3 | | Reserved for future use |

The response to this message is the RDR_to_PC_Parameters message.

8.1.6. PC_to_RDR_ResetParameters

Reset slot parameters to default value.

| Offset | Field | Size | Value | Description |
|--------|--------------|------|-----------|---|
| 0 | bMessageType | 1 | 6Dh | |
| 1 | DwLength | 4 | 00000000h | Size of extra bytes of this message |
| 5 | BSlot | 1 | | Identifies the slot number for this command |
| 6 | BSeq | 1 | | Sequence number for command |
| 7 | AbRFU | 3 | | Reserved for future use |

The response to this message is the RDR_to_PC_Parameters message.



8.1.7. PC_to_RDR_SetParameters

Set slot parameters.

| Offset | Field | Size | Value | Description |
|--------|--------------------------|------------|-------|--|
| 0 | bMessageType | 1 | 61h | |
| 1 | dwLength | 4 | | Size of extra bytes of this message |
| 5 | bSlot | 1 | | Identifies the slot number for this command |
| 6 | bSeq | 1 | | Sequence number for command |
| 7 | bProtocolNum | 1 | | Specifies what protocol data structure follows. 00h = Structure for protocol T=0 01h = Structure for protocol T=1 The following values are reserved for future use. 80h = Structure for 2-wire protocol 81h = Structure for 3-wire protocol 82h = Structure for I2C protocol |
| 8 | abRFU | 2 | | Reserved for future use |
| 10 | abProtocolData Structure | Byte array | | Protocol Data Structure |

Protocol Data Structure for Protocol T=0 (dwLength=00000005h)

| Offset | Field | Size | Value | Description |
|--------|--------------------|------|-------|---|
| 10 | bmFindexDindex | 1 | | B7-4 – FI – Index into the table 7 in ISO/IEC 7816-3:1997 selecting a clock rate conversion factor B3-0 – DI - Index into the table 8 in ISO/IEC 7816-3:1997 selecting a baud rate conversion factor |
| 11 | bmTCKKST0 | 1 | | B0 – 0b, B7-2 – 000000b B1 – Convention used (b1=0 for direct, b1=1 for inverse) Note: The CCID ignores this bit. |
| 12 | bGuardTimeT0 | 1 | | Extra Guardtime between two characters. Add 0 to 254 etu to the normal guardtime of 12 etu. FFh is the same as 00h. |
| 13 | bWaitingInteger T0 | 1 | | WI for T=0 used to define WWT |
| 14 | bClockStop | 1 | | ICC Clock Stop Support 00h = Stopping the Clock is not allowed 01h = Stop with Clock signal Low 02h = Stop with Clock signal High 03h = Stop with Clock either High or Low |



Protocol Data Structure for Protocol T=1 (dwLength=00000007h)

| Offset | Field | Size | Value | Description |
|--------|-------------------|------|-------|---|
| 10 | bmFindexDindex | 1 | | B7-4 – FI – Index into the table 7 in ISO/IEC 7816-3:1997 selecting a clock rate conversion factor B3-0 – DI - Index into the table 8 in ISO/IEC 7816-3:1997 selecting a baud rate conversion factor |
| 11 | BmTCKCKST1 | 1 | | B7-2 – 000100b B0 – Checksum type (b0=0 for LRC, b0=1 for CRC) B1 – Convention used (b1=0 for direct, b1=1 for inverse) Note: The CCID ignores this bit. |
| 12 | BGuardTimeT1 | 1 | | Extra Guardtime (0 to 254 etu between two characters). If value is FFh, then guardtime is reduced by 1 etu. |
| 13 | BwaitingIntegerT1 | 1 | | B7-4 = BWI values 0-9 valid B3-0 = CWI values 0-Fh valid |
| 14 | bClockStop | 1 | | ICC Clock Stop Support 00h = Stopping the Clock is not allowed 01h = Stop with Clock signal Low 02h = Stop with Clock signal High 03h = Stop with Clock either High or Low |
| 15 | bIFSC | 1 | | Size of negotiated IFSC |
| 16 | bNadValue | 1 | 00h | Only support NAD = 00h |

The response to this message is the RDR_to_PC_Parameters message.

8.1.8. CCID Bulk-IN Messages

The Bulk-IN messages are used in response to the Bulk-OUT messages. ACR38x (CCID) shall follow the CCID Bulk-IN Messages as specified in section 4. This section lists the CCID Bulk-IN Messages to be supported by ACR38x (CCID).

8.1.9. RDR_to_PC_DataBlock

This message is sent by ACR38x (CCID) in response to PC_to_RDR_IccPowerOn, PC_to_RDR_XfrBlock and PC_to_RDR_Secure messages.

| Offset | Field | Size | Value | Description |
|--------|-----------------|------------|-------|---|
| 0 | bMessageType | 1 | 80h | Indicates that a data block is being sent from the CCID |
| 1 | dwLength | 4 | | Size of extra bytes of this message |
| 5 | bSlot | 1 | | Same value as in Bulk-OUT message |
| 6 | bSeq | 1 | | Same value as in Bulk-OUT message |
| 7 | bStatus | 1 | | Slot status register as defined in CCID section 4.2.1 |
| 8 | bError | 1 | | Slot error register as defined in CCID section 4.2.1 and this specification section 5.2.8 |
| 9 | bChainParameter | 1 | 00h | RFU (TPDU exchange level) |
| 10 | abData | Byte array | | This field contains the data returned by the CCID |



8.1.10. RDR_to_PC_SlotStatus

This message is sent by ACR38x (CCID) in response to PC_to_RDR_IccPowerOff, PC_to_RDR_GetSlotStatus, PC_to_RDR_Abort messages and Class specific ABORT request.

| Offset | Field | Size | Value | Description |
|--------|--------------|------|-----------|--|
| 0 | bMessageType | 1 | 81h | |
| 1 | dwLength | 4 | 00000000h | Size of extra bytes of this message |
| 5 | bSlot | 1 | | Same value as in Bulk-OUT message |
| 6 | bSeq | 1 | | Same value as in Bulk-OUT message |
| 7 | bStatus | 1 | | Slot status register as defined in CCID section 4.2.1 |
| 8 | bError | 1 | | Slot error register as defined in CCID section 4.2.1 and this specification section 5.2.8 |
| 9 | bClockStatus | 1 | | value = 00h Clock running 01h Clock stopped in state L 02h Clock stopped in state H 03h Clock stopped in an unknown state All other values are RFU. |

8.1.11. RDR_to_PC_Parameters

This message is sent by ACR38x (CCID) in response to PC_to_RDR_GetParameters, PC_to_RDR_ResetParameters and PC_to_RDR_SetParameters messages.

| Offset | Field | Size | Value | Description |
|--------|--------------------------|------------|-------|--|
| 0 | bMessageType | 1 | 82h | |
| 1 | dwLength | 4 | | Size of extra bytes of this message |
| 5 | bSlot | 1 | | Same value as in Bulk-OUT message |
| 6 | bSeq | 1 | | Same value as in Bulk-OUT message |
| 7 | bStatus | 1 | | Slot status register as defined in CCID section 4.2.1 |
| 8 | bError | 1 | | Slot error register as defined in CCID section 4.2.1 and this specification section 5.2.8 |
| 9 | bProtocolNum | 1 | | Specifies what protocol data structure follows. 00h = Structure for protocol T=0 01h = Structure for protocol T=1 The following values are reserved for future use. 80h = Structure for 2-wire protocol 81h = Structure for 3-wire protocol 82h = Structure for I2C protocol |
| 10 | abProtocolData Structure | Byte array | | Protocol Data Structure as summarized in section 5.2.3. |



8.2. Commands Accessed via PC_to_RDR_XfrBlock

8.2.1. GET_READER_INFORMATION

This command returns relevant information about the particular ACR38x model and the current operating status such as the firmware revision number; the maximum data length of a command and response; the supported card types; and whether a card is inserted and powered up or not.

Note: This command can only be used after the logical smart card reader communication has been established using the SCardConnect() API. For details of SCardConnect() API, please refer to PC/SC specifications.

Command format (abData field in the PC_to_RDR_XfrBlock)

| Pseudo-APDU | | | | |
|-----------------|-----------------|-----------------|-----------------|-----------------|
| CLA | INS | P1 | P2 | Lc |
| FF _H | 09 _H | 00 _H | 00 _H | 10 _H |

Response data format (abData field in the RDR_to_PC_DataBlock)

| FIRMWARE | | | | | | | | | | MAX_C | MAX_R | C_TYPE | C_SEL | C_STAT |
|----------|--|--|--|--|--|--|--|--|--|-------|-------|--------|-------|--------|
| | | | | | | | | | | | | | | |

FIRMWARE 10 bytes data for firmware version

MAX_C The maximum number of command data bytes.

MAX_R The maximum number of data bytes that can be requested to be transmitted in a response.

C_TYPE The card types supported by the ACR38x (CCID). This data field is a bitmap with each bit representing a particular card type. A bit set to '1' means the corresponding card type is supported by the reader and can be selected with the *SELECT_CARD_TYPE* command. The bit assignment is as follows:

| Byte | 1 | | | | | | | | 2 | | | | | | | |
|-----------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | F | E | D | C | B | A | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| card type | | | | | | | | | | | | | | | | |

See Appendix A for the correspondence between these bits and the respective card types.

C_SEL The currently selected card type. A value of 00_H means that no card type has been selected.

C_STAT Indicates whether a card is inserted in the reader and whether the card is powered up:

00_H: no card inserted

01_H: card inserted, not powered up

03_H: card powered up



Appendix A. Supported Card Types

This table lists the card types returned by `GET_READER_INFORMATION` corresponding with the respective card type code:

| Card type code | Card Type |
|-----------------|---|
| 00 _H | Auto-select T=0 or T=1 communication protocol |
| 01 _H | I2C memory card (1k, 2k, 4k, 8k and 16k bits) |
| 02 _H | I2C memory card (32k, 64k, 128k, 256k, 512k and 1024k bits) |
| 03 _H | Atmel AT88SC153 secure memory card |
| 04 _H | Atmel AT88SC1608 secure memory card |
| 05 _H | Infineon SLE4418 and SLE4428 |
| 06 _H | Infineon SLE4432 and SLE4442 |
| 07 _H | Infineon SLE4406, SLE4436 and SLE5536 |
| 08 _H | Infineon SLE4404 |
| 09 _H | Atmel AT88SC101, AT88SC102 and AT88SC1003 |
| 0C _H | MCU-based cards with T=0 communication protocol |
| 0D _H | MCU-based cards with T=1 communication protocol |



Appendix B. Response Error Codes

This table lists the error codes that may be returned by the ACR38x:

| Error Code | Status |
|-----------------|--------------------------------------|
| FF _H | SLOTERROR_CMD_ABORTED |
| FE _H | SLOTERROR_ICC_MUTE |
| FD _H | SLOTERROR_XFR_PARITY_ERROR |
| FC _H | SLOTERROR_XFR_OVERRUN |
| FB _H | SLOTERROR_HW_ERROR |
| F8 _H | SLOTERROR_BAD_ATR_TS |
| F7 _H | SLOTERROR_BAD_ATR_TCK |
| F6 _H | SLOTERROR_ICC_PROTOCOL_NOT_SUPPORTED |
| F5 _H | SLOTERROR_ICC_CLASS_NOT_SUPPORTED |
| F4 _H | SLOTERROR_PROCEDURE_BYTE_CONFLICE |
| F3 _H | SLOTERROR_DEACTIVATED_PROTOCOL |
| F2 _H | SLOTERROR_BUSY_WITH_AUTO_SEQUENCE |
| E0 _H | SLOTERROR_CMD_SLOT_BUSY |