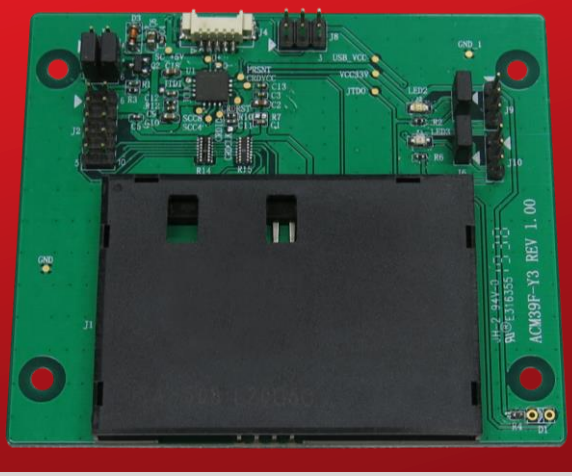
Advanced Card Systems Ltd.
Card & Reader Technologies

ACM39U-Y (CCID) Reader Module



Reference Manual V1.00

info@acs.com.hk
www.acs.com.hk

Revision History

Release Date	Revision Description	Version Number
2020-03-09	• Initial Release	1.00

Table of Contents

List of Figures

List of Tables

1.0. Introduction

The ACM39U-Y Module Reader acts as an interface for the communication between a computer and a smart card. Different types of smart cards have different commands and communication protocols, which, in most cases, prevent direct communication between a smart card and a computer. The ACM39U-Y module reader, which shares the same core as the ACR39 Smart Card Reader, establishes a uniform interface from the computer to the smart card for a wide variety of cards. By taking care of the card's particulars, it releases the computer software programmer from being responsible with smart card operations' technical details, which in many cases are not relevant to the implementation of a smart card system.

1.1. Reference Documents

The following related documents are available from www.usb.org

- Universal Serial Bus Specification 2.0 (also referred to as the USB specification), April 27, 2000
- Universal Serial Bus Common Class Specification 1.0, December 16, 1997
- Universal Serial Bus Device Class: Smart Card CCID Specification for Integrated Circuit(s) Cards Interface Devices, Revision 1.1, April 22, 2005

The following related documents can be ordered through www.ansi.org

- ISO/IEC 7816-1; Identification Cards – Integrated circuit(s) cards with contacts - Part 1: Physical Characteristics
- ISO/IEC 7816-2; Identification Cards – Integrated circuit(s) cards with contacts - Part 2: Dimensions and Locations of the contacts
- ISO/IEC 7816-3; Identification Cards – Integrated circuit(s) cards with contacts - Part 3: Electronic signals and transmission protocols

1.2. Symbols and Abbreviations

Abbreviation	Description
ATR	Answer-To-Reset
CCID	Chip/Smart Card Interface Device
ICC	Integrated Circuit Cards
IFSC	Information Field Sized for ICC for protocol T=1
IFSD	Information Field Sized for CCID for protocol T=1
NAD	Node Address
PPS	Protocol and Parameters Selection
RFU	Reserved for future use[1]
TPDU	Transport Protocol Data Unit
USB	Universal Serial Bus

Table 1: Symbols and Abbreviations

[1] *Must be set to zero unless stated differently.*

2.0. Features

- USB Full Speed Interface
- Plug and Play – CCID support brings utmost mobility
- Smart Card Reader:
  - Contact Interface:
    - Supports ISO 7816 Class A, B and C (5 V, 3 V, 1.8 V) cards
    - Supports CAC (Common Access Card)
    - Supports SIPRNET Card
    - Supports J-LIS Card
    - Supports microprocessor cards with T=0 or T=1 protocol
    - Supports memory cards
    - Supports PPS (Protocol and Parameters Selection)
    - Features Short Circuit Protection
- Application Programming Interface:
  - Supports PC/SC
  - Supports CT-API (through wrapper on top of PC/SC)
- Supports Android™ 3.1 and above[2]
- Compliant with the following standards:
  - EN 60950/IEC 60950
  - ISO 7816
  - EMV™ Level 1 (Contact)
  - PC/SC
  - CCID
  - CE
  - FCC
  - RoHS
  - REACH
  - Microsoft® WHQL

[2] *PC/SC and CCID support are not applicable*

3.0. Smart Card Support

3.1. MCU Cards

The ACM39U-Y is a PC/SC compliant smart card reader that supports ISO 7816 Class A, B and C (5 V, 3 V, and 1.8 V) smart cards. It also works with MCU cards following either the T=0 and T=1 protocol.

The card ATR indicates the specific operation mode (TA2 present; bit b5 of TA2 must be 0) and when that particular mode is not supported by the ACM38U-Y (CCID), the reader will reset the card to a negotiable mode. If the card cannot be set to negotiable mode, the reader will then reject the card.

When the card ATR indicates the negotiable mode (TA2 not present) and communication parameters other than the default parameters, the ACM38U-Y (CCID) will execute the PPS and try to use the communication parameters that the card suggested in its ATR. If the card does not accept the PPS, the reader will use the default parameters (F=372, D=1).

Note: *For the meaning of the aforementioned parameters, please refer to* ***ISO 7816-3****.*

3.2. Memory-based Smart Cards

The ACM39U-Y works with several memory-based smart cards, such as:

- Cards following the I2C bus 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 intelligent 1 KB EEPROM with write-protect function, including:
  - Infineon®: SLE4418, SLE4428, SLE5518 and SLE5528
- Cards with intelligent 256-byte EEPROM with write-protect function, including:
  - Infineon®: SLE4432, SLE4442, SLE5532 and SLE5542

4.0. Smart Card Interface

The interface between the ACM39U-Y and the inserted smart card follows the specification of ISO 7816-3 with certain restrictions or enhancements to increase the practical functionality of the ACM39U-Y (CCID).

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 ACM39U-Y. The electrical specifications of this contact are identical to those of the signal RST (at contact C2).

4.3. Card Type Selection

The controlling computer must always select the card type through the proper command sent to the ACM39U-Y prior to activating the inserted card. This includes both memory cards and MCU-based cards.

For MCU-based cards, the reader allows for the selection of 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. If 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 ACM39U-Y 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 ACM38U-Y (CCID) and the card is immediately deactivated when the card is being removed. However, as a rule, to avoid any electrical damage, a card should be removed from the reader only while the former is powered down.

Note: *The ACM39U-Y never switches 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 ACM39U-Y requires a voltage of 5 V DC, 100 mA, regulated, power supply. The ACM39U-Y gets its power supply from the computer (through the cable supplied along with each type of reader). Please refer to the ACM39U-Y user manual for LED usage.

5.1. LED

5.1.1. Status

There are two LEDs on the ACM39U-Y3 that display its operation status:

- **Red** is for power
- **Green** is for smart card operation

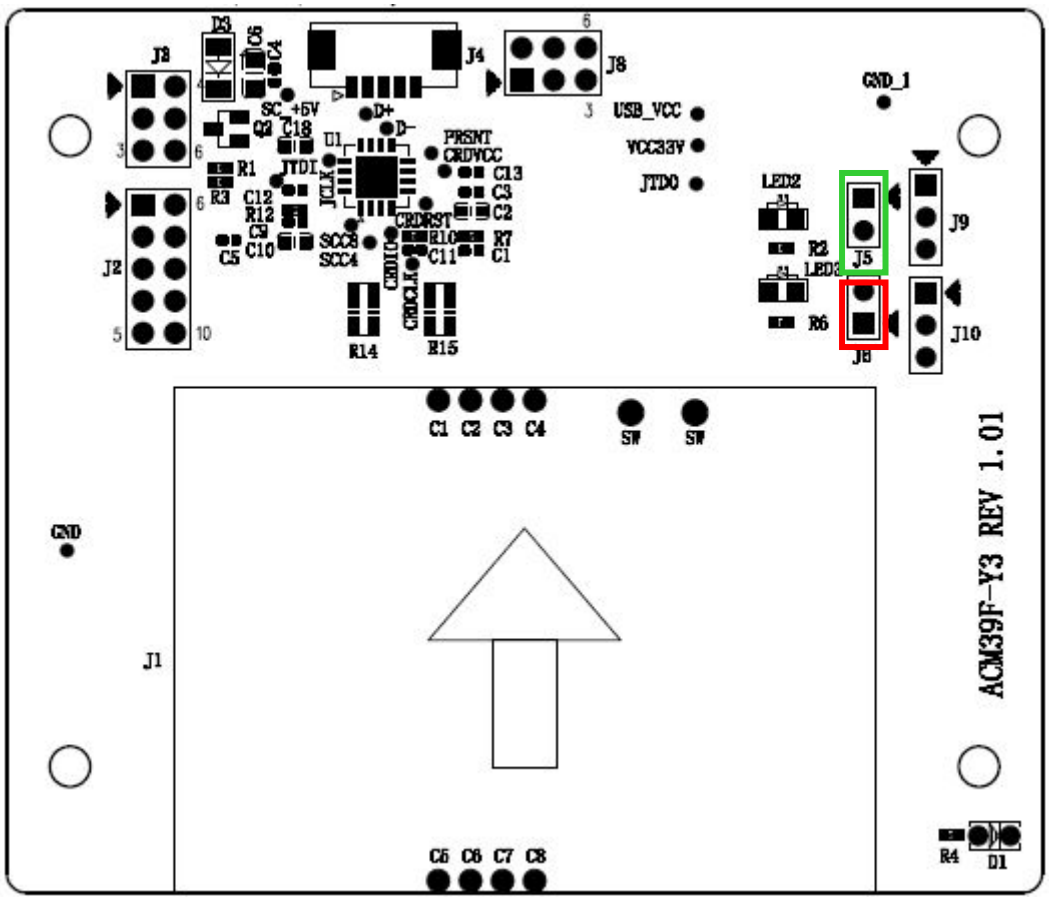


Figure 1: ACM39U-Y3 LED Configuration

5.1.2. LED Behaviors

The table below shows the behaviors of the LED during card polling.

LED Color	Behavior	Status
Red	On	Power

LED Color	Behavior	Status
Green	Slow flash[3]	No card present/ Card power off
	On	Card power on
	Fast flash	Card operation

[3] *The LED turns on for 200 ms every 2 seconds.*

6.0. Hardware Connections

Please refer to the ACM39U-Y user manual for the purpose of the connectors.

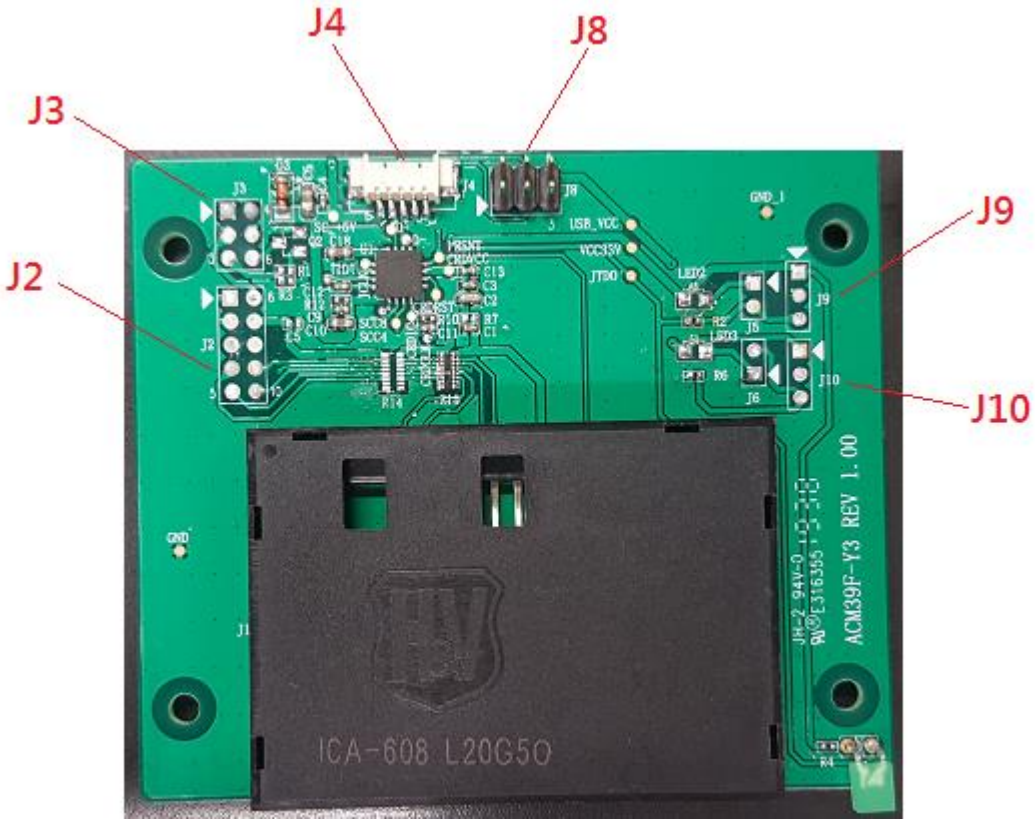


Figure 2: ACM38U-Y Hardware Connections

Jumper	Connector
J2	External contact card connector
J3	Normal-Short/Normal-Open card detection selection
J4	USB cable connector
J8	USB pinout connector
J9	Card connectivity signal connector
J10	USB power signal connector

6.1. J2: External Contact Card Connector

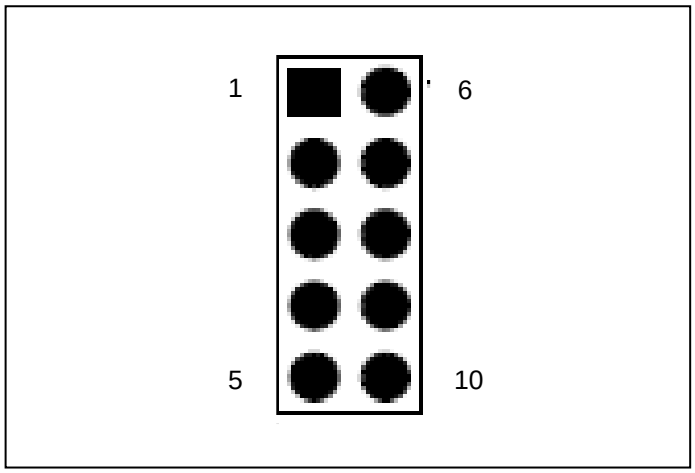


Figure 3: External Contact Card Connector

Connector	Description
1, 2	C5: GND
3	C6: NC
4	C7: Card I/O (Data input/output)
5	C8: Card contact number
6	Card detection pin
7	C1: Card VCC
8	C2: Card reset pin
9	C3: Card clock signal
10	C4: Card contact number

When using the external card connector, the selection resistor jumper should be changed. The resistor array should be soldered on (**R14**) instead of R15. See figure below.

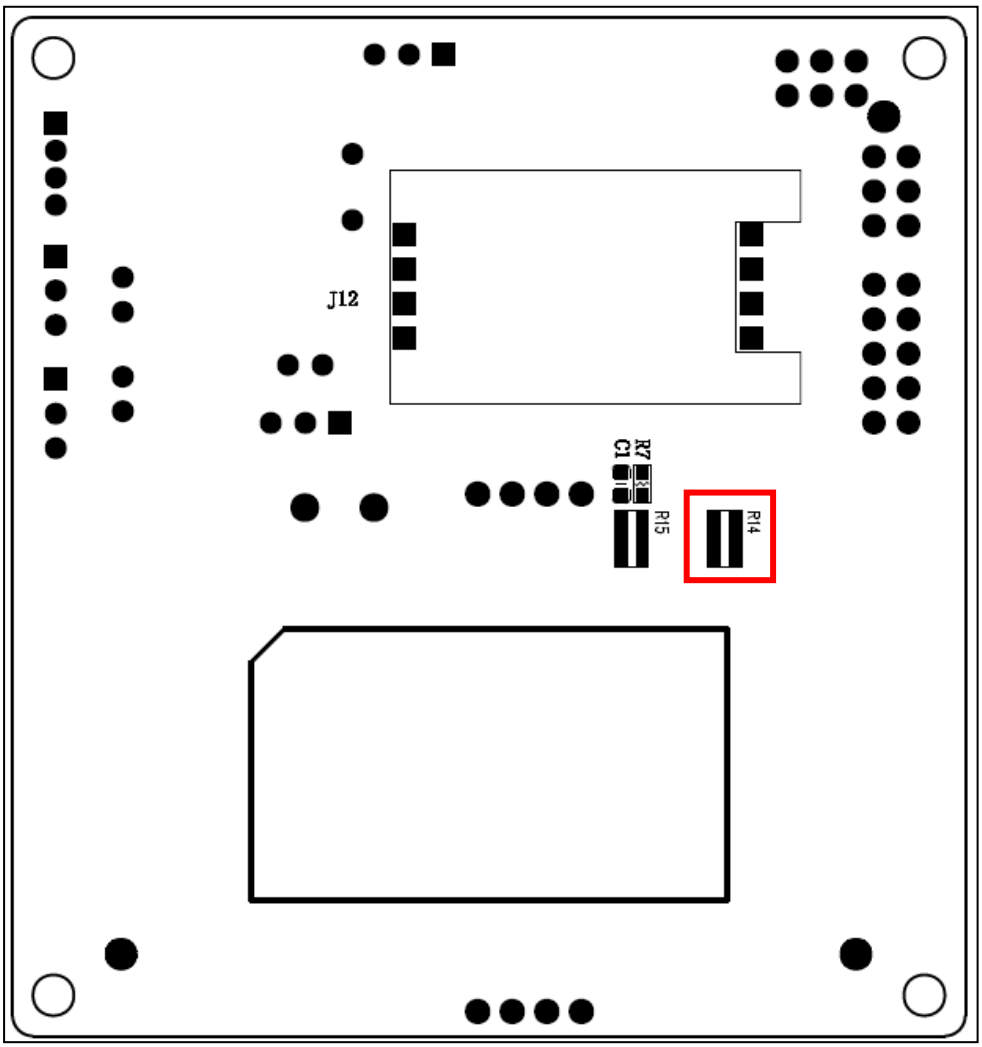
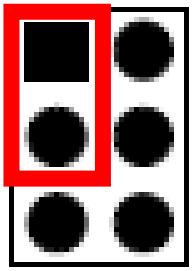
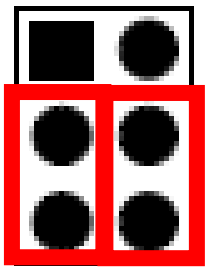


Figure 4: Selection Resistor Jumpers at the back of ACM39U-Y3

6.2. J3: Normal-Short/Normal-Open Card Detection Selection

Configuration	Connecting diagram	Descriptions
Normal open card detection circuit		Connect the two pins (as shown in red) using the jumper head provided if a normal open card detection card slot is used
Normal closed card detection circuit		Connect the two pair of pins (as shown in red) using the jumper heads provided if a normal closed card detection card slot is used

6.3. J4: USB Cable Connector

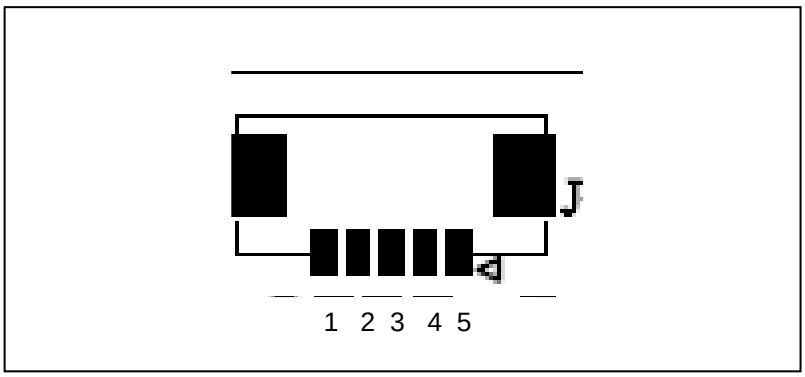


Figure 5: USB Cable Connector

Connector	Description
1	USB VCC
2	USB D-
3	USB D+
4	USB GND
5	USB Shielding

6.4. J8: USB Pinout Connector

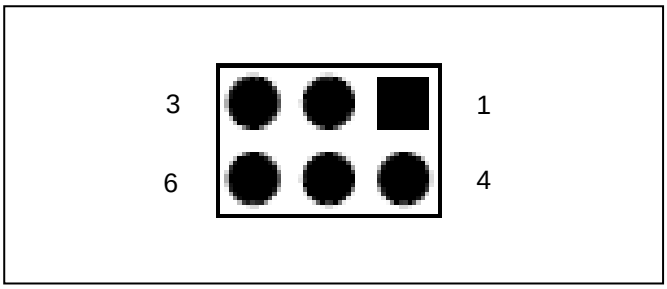


Figure 6: USB Pinout Connector

Connector	Description
1, 3, 4	USB GND
2	USB D+
5	USB D-
6	USB VCC

6.5. J9: Card Connectivity Signal Connector

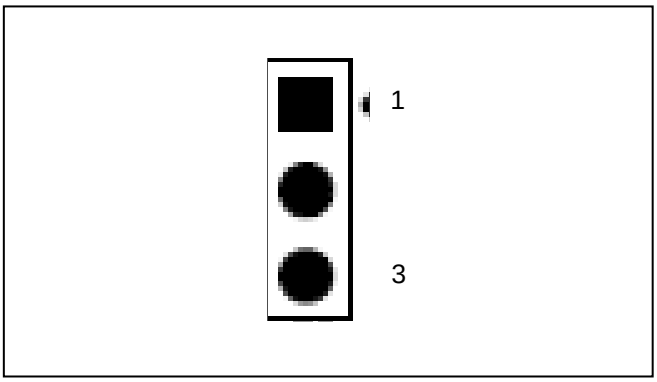


Figure 7: Card Connectivity Signal Connector

Connector	Description
1	USB VCC
2	Not connected
3	Connected signal from the MCU

6.6. J10: USB Power Signal Connector

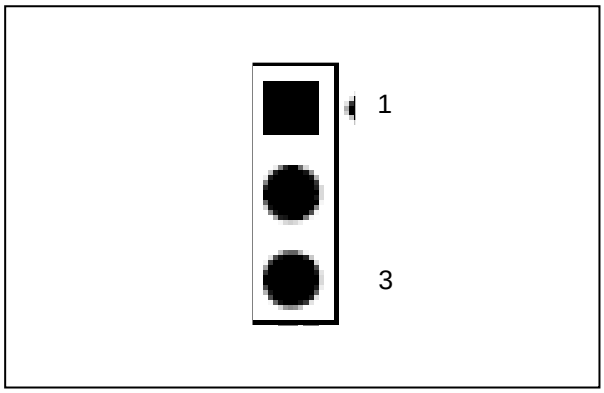


Figure 8: USB Power Signal Connector

Connector	Description
1	USB VCC
2	Not connected
3	GND

7.0. USB Interface

7.1. Communication Parameters

The ACM39U-Y is connected to a computer through USB as specified in the USB Specification 2.0. The ACM39U-Y works in full speed mode (i.e. 12 Mbps). Please refer to chapter 6 for USB connectors.

Pin	Signal	Function
1	V_{BUS}	+5 V power supply for the reader
2	D-	Differential signal transmits data between ACM39U-Y and computer
3	D+	Differential signal transmits data between ACM39U-Y and computer
4	GND	Reference voltage level for power supply

Table 2: USB Interface Wiring

Note: *In order for the ACM39U-Y to function properly through USB interface, either the ACS CCID driver or the Microsoft CCID driver has to be installed.*

7.2. Endpoints

The ACM39U-Y uses the following endpoints to communicate with the host computer:

Control Endpoint	For setup and control purpose
Bulk OUT	For command to be sent from host to ACM39U-Y (data packet size is 64 bytes)
Bulk IN	For response to be sent from ACM39U-Y to host (data packet size is 64 bytes)
Interrupt IN	For card status message to be sent from ACM39U-Y to host (data packet size is 8 bytes)

8.0. Communication Protocol

The ACM39U-Y shall interface with the host through the 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.

The configurations and usage of USB endpoints on the ACM39U-Y shall follow CCID Rev 1.0 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 the ACM39U-Y 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 the ACM39U-Y have to be sent synchronously [e.g., *bMaxCCIDBusySlots* is equal to 01h for ACM39U-Y].

The ACM39U-Y supported CCID features are indicated in its Class Descriptor:

Offset	Field	Size	Value	Description
0	*bLength*	1		Size of this descriptor, in bytes.
1	*bDescriptorType*	1		CCID Functional Descriptor type.
2	*bcdCCID*	2		CCID Specification Release Number in Binary-coded decimal.
4	*bMaxSlotIndex*	1		One slot is available on ACM39U-Y.
5	*bVoltageSupport*	1		ACM39U-Y can supply 1.8 V, 3 V, and 5 V to its slot.
6	*dwProtocols*	4		ACM39U-Y supports T=0 and T=1 protocol.
10	*dwDefaultClock*	4		Default ICC clock frequency is 4.8 MHz.
14	*dwMaximumClock*	4		Maximum supported ICC clock frequency is 4.8 MHz.
18	*bNumClockSupported*	1		Does not support manual setting of clock frequency.
19	*dwDataRate*	4		Default ICC I/O data rate is 12918 bps.
23	*dwMaxDataRate*	4		Maximum supported ICC I/O data rate is 826 Kbps.
27	*bNumDataRatesSupported*	1		Does not support manual setting of data rates.
28	*dwMaxIFSD*	4		Maximum IFSD supported by ACM39U-Y for protocol T=1 is 247.
32	*dwSynchProtocols*	4		ACM39U-Y does not support synchronous card.
36	*dwMechanical*	4		ACM39U-Y does not support special mechanical characteristics.

Offset	Field	Size	Value	Description
40	*dwFeatures*	4		ACM39U-Y supports the following features: • Automatic ICC clock frequency change according to parameters • Automatic baud rate change according to frequency and FI,DI parameters • TPDU level change with ACM39U-Y
44	*dwMaxCCIDMessageLength*	4		Maximum message length accepted by ACM39U-Y is 271 bytes.
48	*bClassGetResponse*	1		Insignificant for TPDU level exchanges.
49	*bClassEnvelope*	1		Insignificant for TPDU level exchanges.
50	*wLCDLayout*	2		No LCD.
52	*bPINSupport*	1		With PIN Verification.
53	*bMaxCCIDBusySlots*	1		Only 1 slot can be simultaneously busy.

9.0. Memory Card Type Selection

The `SELECT_CARD_TYPE` command must be executed first before other memory card commands. This command powers up and down the selected card inserted in the card reader and performs a card reset. This command can only be used after the logical smart card reader communication has been established using the *SCardConnect()* API.

Note: *For details of SCardConnect() API, please refer to PC/SC specifications. For the Memory Card Command Set, please refer to* ***Section 10.3*** *of this reference manual.*

A code snippet of the program flow is given below to demonstrate how to select the memory card type in the ACM39U-Y:

```
SCARDCONTEXT hContext;
SCARDHANDLE hCard;
unsigned long dwActProtocol;
SCARD_IO_REQUEST ioRequest;
DWORD size = 64, SendLen = 6, RecvLen = 255, retCode;
byte cardType;
//Establish PC/SC Connection
retCode   =   SCardEstablishContext   (SCARD_SCOPE_USER,   NULL,   NULL,
&hContext);
//List all readers in the system
retCode = SCardListReaders (hContext, NULL, readerName, &size);
//Connect to the reader
retCode   =   SCardConnect(hContext,   readerName,   SCARD_SHARE_SHARED,
SCARD_PROTOCOL_T0, &hCard, &dwActProtocol);
//Select Card Type
unsigned char SendBuff[] = {FF,A4,00,00,01,cardType};
retCode  =  SCardTransmit(  hCard,  &ioRequest,  SendBuff,  SendLen,  NULL,
RecvBuff, &RecvLen);
//Disconnect from the reader
retCode = SCardDisconnect(hCard, SCARD_UNPOWER_CARD);
//End the established context
retCode = SCardReleaseContext(hContext);
```

10.0.Commands

10.1. CCID Command Pipe Bulk-OUT Messages

The ACM39U-Y (CCID) shall follow the CCID Bulk-OUT Messages as specified in CCID Rev 1.0 Section 4.1. In addition, this specification defines some extended commands for operating additional features.

This section lists the CCID Bulk-OUT Messages to be supported by ACM39U-Y (CCID).

10.1.1. PC_to_RDR_IccPowerOn

This command activates the card slot and returns ATR data 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 V 02h = 3 V
7	*abRFU*	2		Reserved for future use.

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

10.1.2. PC_to_RDR_IccPowerOff

This command deactivates 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.

10.1.3. PC_to_RDR_GetSlotStatus

This command gets 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.

10.1.4. PC_to_RDR_XfrBlock

This command transfers a 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.

10.1.5. PC_to_RDR_GetParameters

This command gets the 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.

10.1.6. PC_to_RDR_ResetParameters

This command resets slot parameters to the 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.

10.1.7. PC_to_RDR_SetParameters

This command sets the 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	*abProtocolDataStructure*	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

Offset	Field	Size	Value	Description
11	*bmTCCKST0*	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	*bWaitingIntegerT0*	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	*bmTCCKST1*	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

Offset	Field	Size	Value	Description
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.

10.2. CCID Bulk-IN Messages

The Bulk-IN messages are used in response to the Bulk-OUT messages. The ACM39U-Y shall follow the CCID Bulk-IN Messages as specified in CCID Rev 1.0 Section 4.2.

This section lists the CCID Bulk-IN Messages to be supported by the ACM39U-Y.

10.2.1. RDR_to_PC_DataBlock

This command is sent by the ACM39U-Y in response to *PC_to_RDR_IccPowerOn* and *PC_to_RDR_XfrBlock*.

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 Rev 1.0 Section 4.2.1.
8	*bError*	1		Slot error register as defined in CCID Rev 1.0 Section 4.2.1.
9	*bChainParameter*	1	00h	RFU (TPDU exchange level).
10	*abData*	Byte array		This field contains the data returned by the CCID.

10.2.2. RDR_to_PC_SlotStatus

This command is sent by the ACM39U-Y in response to *PC_to_RDR_IccPowerOff* and *PC_to_RDR_GetSlotStatus.*

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 Rev 1.0 Section 4.2.1.
8	*bError*	1		Slot error register as defined in CCID Rev 1.0 Section 4.2.1.

Offset	Field	Size	Value	Description
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.

10.2.3. RDR_to_PC_Parameters

This message is sent by the ACM39U-Y 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 Rev 1.0 Section 4.2.1.
8	*bError*	1		Slot error register as defined in CCID Rev 1.0 Section 4.2.1.
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	*abProtocolDataStructure*	Byte array		Protocol Data Structure.

10.3. Memory Card Command Set

This section contains the Memory Card Command Set for the ACM39U-Y.

10.3.1. Memory Card – 1, 2, 4, 8 and 16 kilobit I2C Card

10.3.1.1. SELECT_CARD_TYPE

This command powers up and down the selected card that is inserted in the card reader and performs a card reset.

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**	**Card Type**
FFh	A4h	00h	00h	01h	01h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.1.2. SELECT_PAGE_SIZE

This command selects the page size to read the smart card. The default value is 8-byte page write. It resets to the default value whenever the card is removed or the reader is powered off.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Page Size**
FFh	01h	00h	00h	01h	

Where:

Page size = 03h for 8-byte page write
= 04h for 16-byte page write
= 05h for 32-byte page write
= 06h for 64-byte page write
= 07h for 128-byte page write

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.1.3. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	INS	Byte Address		MEM_L
		MSB	LSB	
FFh	B0h			

Where:

Byte Address Memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (abData field in the RDR_to_PC_DataBlock)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1 SW2 = 90 00h if no error

10.3.1.4. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte n
		MSB	LSB					
FFh	D0h							

Where:

Byte Address Memory address location of the memory card.

MEM_L Length of data to be read from the memory card.

BYTE X Data to be written to the memory card.

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.2. Memory Card – 32, 64, 128, 256, 512, and 1024 kilobit I2C Card

10.3.2.1. SELECT_CARD_TYPE

This command powers up and down the selected card that is inserted in the card reader and performs a card reset.

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**	**Card Type**
FFh	A4h	00h	00h	01h	02h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.2.2. SELECT_PAGE_SIZE

This command selects the page size to read the smart card. The default value is 8-byte page write. It resets to default value whenever the card is removed or the reader is powered off.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Page size**
FFh	01h	00h	00h	01h	

Where:

Data TPDU to be sent to the card

Page size = 03h for 8-byte page write
= 04h for 16-byte page write
= 05h for 32-byte page write
= 06h for 64-byte page write
= 07h for 128-byte page write

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.2.3. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	INS	Byte Address		MEM_L
		MSB	LSB	
FFh				

Where:

INS = B0h for 32, 64, 128, 256, 512 kilobit iic card
= 1011 000*b for 1024 kilobit iic card,
where * is the MSB of the 17 bit addressing

Byte Address Memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1 SW2 = 90 00h if no error

10.3.2.4. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte n
		MSB	LSB					
FFh								

Where:

INS = D0h for 32, 64, 128, 256, 512 kilobit iic card
= 1101 000*b for 1024 kilobit iic card,
where * is the MSB of the 17 bit addressing

Byte Address Memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.3. Memory Card – SLE 4418/SLE 4428/SLE 5518/SLE 5528

10.3.3.1. SELECT_CARD_TYPE

This command powers up and down the selected card that is inserted in the card reader and performs a card reset.

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**	**Card Type**
FFh	A4h	00h	00h	01h	05h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.3.2. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**Byte Address**		**MEM_L**
		MSB	**LSB**	
FFh	B0h			

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1, SW2 = 90 00h if no error

10.3.3.3. READ_PRESENTATION_ERROR_COUNTER_MEMORY_CARD (SLE 4428 and SLE 5528)

This command is used to read the presentation error counter for the secret code.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B1h	00h	00h	03h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

ERRCNT	DUMMY 1	DUMMY 2	SW1	SW2

Where:

ERRCNT Error Counter. FFh indicates that the last verification is correct. 00h indicates that the password is locked (exceeded the maximum number of retries). Other values indicate that the last verification has failed.

DUMMY Two bytes dummy data read from the card

SW1 SW2 = 90 00h if no error

10.3.3.4. READ_PROTECTION_BIT

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**Byte Address**		**MEM_L**
		MSB	**LSB**	
FFh	B2h			

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be read from the memory card (in multiple of 8 bits; maximum of 32).

MEM_L = 1 + INT [(number of bits – 1)/8]

For example: To read eight protection bits starting from memory 0010h, the following pseudo-APDU should be issued as:

FF B1 00 10 01h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

PROT 1	…	…	PROT L	SW1	SW2

Where:

PROT y Bytes containing the protection bits

SW1, SW2 = 90 00h if no error

The arrangement of the protection bits in the PROT bytes is as follows:

PROT 1								PROT 2								…							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

Where:

Px is the protection bit of BYTE x in the response data

'0' byte is write protected

'1' byte can be written

10.3.3.5. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU							
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte N
		MSB	LSB					
FFh	D0h							

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.3.6. WRITE_PROTECTION_MEMORY_CARD

Each byte specified in the command is used in the card to compare the byte stored in a specified address location. If the data match, the corresponding protection bit is irreversibly programmed to '0'.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte N
		MSB	LSB					
FFh	D1h							

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Byte values to be compared with the data in the card starting at *Byte Address*. BYTE 1 is compared with the data at Byte Address; BYTE N is compared with the data at (Byte Address+N-1).

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.3.7. PRESENT_CODE_MEMORY_CARD (SLE 4428 and SLE 5528)

This command is used to submit the secret code to the memory card to enable the write operation with the SLE 4428 and SLE 5528 card, the following actions are executed:

1. Search a '1' bit in the presentation error counter and write the bit to '0'.
2. Present the specified code to the card.
3. Try to erase the presentation error counter.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU						
CLA	INS	P1	P2	MEM_L	CODE	
					Byte 1	Byte 2
FFh	20h	00h	00h	02h		

Where:

CODE Two bytes secret code (PIN)

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2 ErrorCnt
90h	

Where:

SW1	= 90h
SW2 (ErrorCnt)	= Error Counter. FFh indicates successful verification. 00h indicates that the password is locked (or exceeded the maximum number of retries). Other values indicate that current verification has failed.

10.3.4. Memory Card – SLE 4432/SLE 4442/SLE 5532/SLE 5542

10.3.4.1. SELECT_CARD_TYPE

This command powers down and up the selected card that is inserted in the card reader and performs a card reset.

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**	**Card Type**
FFh	A4h	00h	00h	01h	06h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.4.2. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh	B0h	00h		

Where:

Byte Address = $A_7A_6A_5A_4\ A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	BYTE N	PROT 1	PROT 2	PROT 3	PROT 4	SW1	SW2

Where:

BYTE x Data read from memory card.

PROT y Bytes containing the protection bits from protection memory.

SW1 SW2 = 90 00h if no error.

The arrangement of the protection bits in the PROT bytes is as follows:

PROT 1								PROT 2								…							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

Where:

Px is the protection bit of BYTE x in the response data.

'0' byte is write protected.

'1' byte can be written.

10.3.4.3. READ_PRESENTATION_ERROR_COUNTER_MEMORY_CARD (SLE 4442 and SLE 5542)

This command is used to read the presentation error counter for the secret code.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B1h	00h	00h	04h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

ERRCNT	DUMMY 1	DUMMY 2	DUMMY 3	SW1	SW2

Where:

ERRCNT	Error counter. 07h indicates that the last verification is correct. 00h indicates that the password is locked (exceeded the maximum number of retries). Other values indicate that the last verification has failed.
DUMMY	Three bytes dummy data read from the card
SW1 SW2	= 90 00h if no error

10.3.4.4. READ_PROTECTION_BITS

This command is used to read the protection bits for the first 32 bytes.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B2h	00h	00h	04h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

PROT 1	PROT 2	PROT 3	PROT 4	SW1	SW2

Where:

PROT y Bytes containing the protection bits from protection memory

SW1, SW2 = 90 00h if no error

The arrangement of the protection bits in the PROT bytes is as follows:

PROT 1								PROT 2								...							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

Where:

Px is the protection bit of BYTE x in the response data

'0' byte is write protected

'1' byte can be written

10.3.4.5. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	P1	Byte Address	MEM_L	Byte 1	....	....	Byte N
FFh	D0h	00h						

Where:

Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.4.6. WRITE_PROTECTION_MEMORY_CARD

Each byte specified in the command is internally in the card compared with the byte stored at the specified address and if the data match, the corresponding protection bit is irreversibly programmed to '0'.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**	**Byte 1**	**....**	**....**	**Byte N**
FFh	D1h	00h						

Where:

Byte Address = $000A_4$ $A_3A_2A_1A_0$b (00h to 1Fh) is the protection memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Byte values to be compared with the data in the card starting at Byte Address. BYTE 1 is compared with the data at Byte Address; BYTE N is compared with the data at (Byte Address+N-1).

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.3.4.7. PRESENT_CODE_MEMORY_CARD (SLE 4442 and SLE 5542)

To submit the secret code to the memory card to enable the write operation with the SLE 4442 and SLE 5542 card, the following actions are executed:

1. Search a '1' bit in the presentation error counter and write the bit to '0'.
2. Present the specified code to the card.
3. Try to erase the presentation error counter.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU							
CLA	**INS**	**P1**	**P2**	**MEM_L**	**CODE**		
					Byte 1	**Byte 2**	**Byte 3**
FFh	20h	00h	00h	03h			

Where:

CODE Three bytes secret code (PIN)

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2 ErrorCnt
90h	

Where:

SW1 = 90h

SW2 (ErrorCnt) = Error Counter. 07h indicates that the verification is correct. 00h indicates the password is locked (exceeded the maximum number of retries). Other values indicate that the current verification has failed.

10.3.4.8. CHANGE_CODE_MEMORY_CARD (SLE 4442 and SLE 5542)

This command is used to write the specified data as new secret code in the card.

The current secret code must have been presented to the card with the *PRESENT_CODE* command prior to the execution of this command.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU							
CLA	INS	P1	P2	MEM_L	CODE		
					Byte 1	Byte 2	Byte 3
FFh	D2h	00h	01h	03h			

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

10.4. Other Commands Access via PC_to_RDR_XfrBlock

10.4.1. GET_READER_INFORMATION

This command is used to return the firmware revision number of the ACR39 reader.

Note: *This command can only be used after the logical smart card reader communication with T=0 protocol 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**
FFh	09h	00h	00h	11h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

FIRMWARE										

Where:

FIRMWARE 11 bytes data for firmware version.

10.5. Other Commands Access via PC-to_RDR_Escape

10.5.1. GET_READER_INFORMATION

This command is used to return the firmware revision number of the ACR39 reader.

Note: *This command can only be used on ACM39U with firmware 003R and above.*

Get Reader Information Format (5 bytes)

Command	CLA	INS	P1	P2	Lc
Get Firmware Version	E0h	00h	00h	19h	00h

Get Reader Information Response Format (5 bytes + Firmware Message Length)

Response	CLA	INS	P1	P2	Le	Data Out
Result	E1h	00h	00h	00h	Number of bytes to receive	Firmware Version

Example:

Response = E1 00 00 00 0C 41 43 52 33 39 55 2D 30 2E 30 33 52

Firmware Version (HEX) = 41 43 52 33 39 55 2D 30 2E 30 33 52

Firmware Version (ASCII) = “ACR39U-0.03R”

Appendix A. Response Error Codes

The following table summarizes the possible error code returned by the ACM39U-Y (CCID):

Error Code	Status
FFh	SLOTERROR_CMD_ABORTED
FEh	SLOTERROR_ICC_MUTE
FDh	SLOTERROR_XFR_PARITY_ERROR
FCh	SLOTERROR_XFR_OVERRUN
FBh	SLOTERROR_HW_ERROR
F8h	SLOTERROR_BAD_ATR_TS
F7h	SLOTERROR_BAD_ATR_TCK
F6h	SLOTERROR_ICC_PROTOCOL_NOT_SUPPORTED
F5h	SLOTERROR_ICC_CLASS_NOT_SUPPORTED
F4h	SLOTERROR_PROCEDURE_BYTE_CONFLICE
F3h	SLOTERROR_DEACTIVATED_PROTOCOL
F2h	SLOTERROR_BUSY_WITH_AUTO_SEQUENCE
E0h	SLOTERROR_CMD_SLOT_BUSY

Table 3: Response Error Codes