Advanced Card Systems Ltd.
Card & Reader Technologies

ACR122L Serial NFC Reader with LCD

Communication Protocol V1.03

Table of Contents

List of Figures

List of Tables

1.0. Introduction

The ACR122L serial protocol defines the interface between the PC and reader, as well as the communication channel between the PC and the supported contactless tags/cards – ISO 14443-4 Type A and B, Mifare, ISO 18092 (NFC), and FeliCa. The major applications supported are the following:

- **Access Control, Identification:** Reading the serial numbers of all cards in the field.
- **Data Storage:** Performing encrypted read-and-write operations.
- **Ticketing:** Performing read, write, increment and decrement operations in an encrypted environment.
- **Multi-applications:** Performing read, write, increment and decrement operations on various sectors of the card.

2.0. Features

- Serial RS-232 Interface: Baud Rate = 115200 bps, 8-N-1
- 7 V DC adaptor for power supply
- CCID-like frame format (Binary format)
- Smart Card Reader:
  - Read/Write speed of up to 424 kbps
  - Built-in antenna for contactless tag access, with card reading distance of up to 50 mm (depending on tag type)
  - Support for ISO 14443 Part 4 Type A and B cards, Mifare, FeliCa and all four types of NFC (ISO/IEC 18092) tags
  - Built-in anti-collision feature (only one tag is accessed at any time)
  - Three ISO 7816 compliant SAM slots
- Built-in Peripherals:
  - Two-line graphic LCD with interactive operability (i.e. scroll up and down, left and right, etc.) and multi-language support (i.e. Chinese, English, Japanese and several European languages)
  - Four user-controllable LEDs
  - User-controllable buzzer
- Compliant with the following standards:
  - ISO 14443
  - CE
  - FCC
  - VCCI
  - RoHS

2.1. Serial Interface

The ACR122L is connected to a Host through the RS232C Serial Interface at 115200 bps, 8-N-1

Pin	Signal	Function
1	V_{CC}	+7 V power supply for the reader (max. 350 mA; normal 200 mA)
2	TXD	The signal from the reader to the host.
3	RXD	The signal from the host to the reader.
4	GND	Reference voltage level for power supply

Table 1: PIN Configuration

2.2. LCD

A user-controllable LCD is provided.

- 2 line x 16 character, 5 x 8 dot matrix, STN yellow-green LCD type
- Yellow-green backlight
- 6 O'clock view angle

2.3. LEDs

Four user-controllable single color LEDs are provided.

- Control can be selected by firmware or by user.
- From left to right, the colors of the LEDs are green, blue, orange and red.

2.4. Buzzer

A user-controllable monotone buzzer with a default state of OFF is provided.

2.5. SAM Interface

Three SAM sockets, supporting ISO 7816-1/2/3 T=0 cards, are provided.

2.6. Built-in Antenna

A 3-turn symmetric loop antenna, center-tapped, is provided.

- Estimated size is 46 mm x 64 mm.
- Loop inductance is approximately 1.6 μH to 2.5 μH.
- Operating distance for different tags, is approximately up to 50 mm (depends on the tag).
- Only one tag can be accessed at any one time.

3.0. Communication between the host and contactless interface, SAM and peripherals

The contactless interface and peripherals are accessed through the use of pseudo-APDUs.

The SAM interface is accessed through the use of standard APDUs.

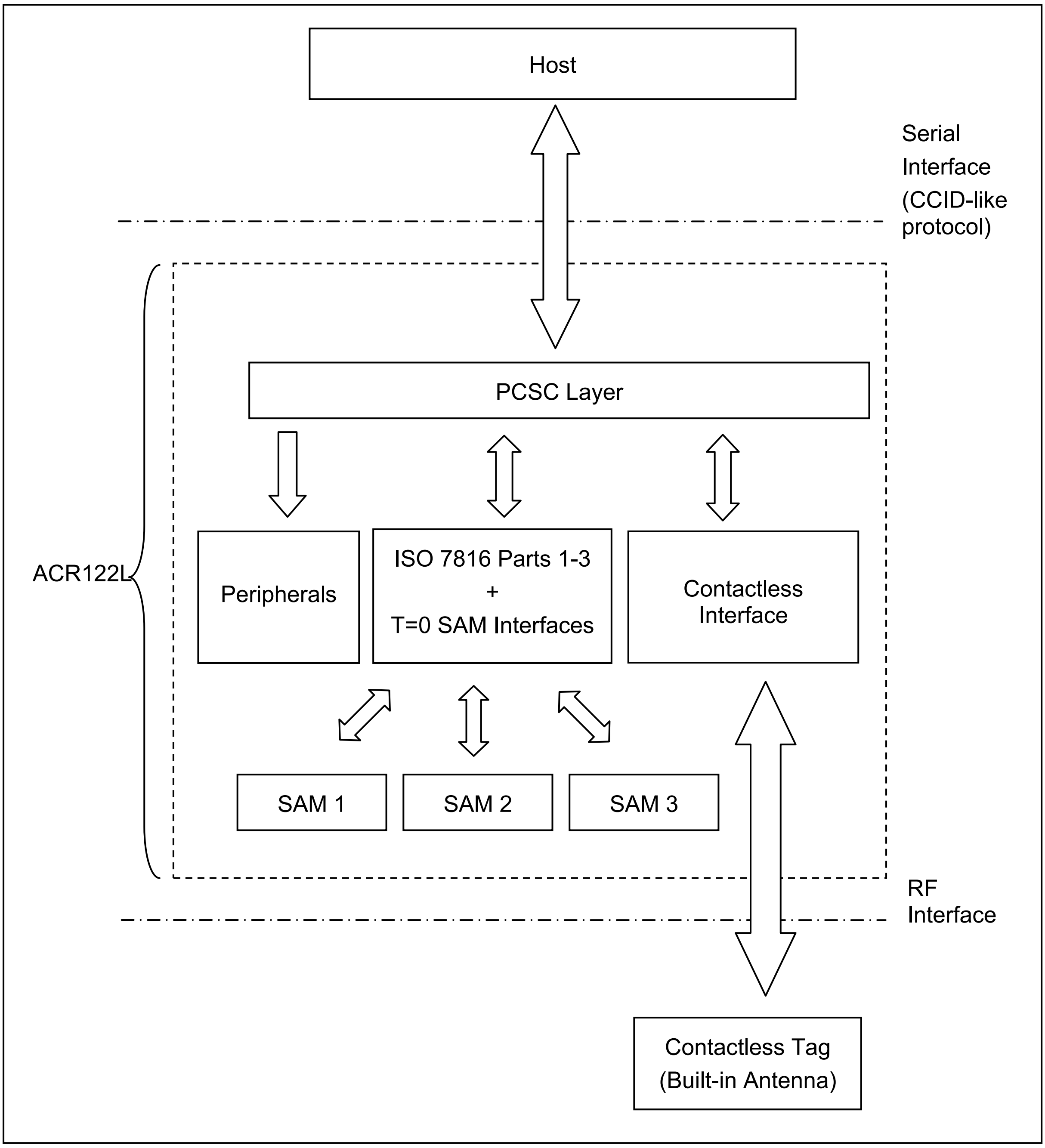


Figure 1: Communication Flowchart of ACR122L

4.0. Serial Interface (CCID-like Frame Format)

In normal operation, the ACR122L acts as a slave device with regard to the communication between a computer and the reader. The communication is carried out in the form of successive command-response exchanges. The computer transmits a command to the reader, and then receives a response from the reader after the command has been executed. A new command can be transmitted to the ACR122L only after the response to the previous command has been received. There are two cases where the reader transmits data without having received a command from the computer, namely, the Reset Message of the reader and the Card Status Message.

Note: *Communication setting: 115200 bps, 8-N-1.*

The communication protocol between the host and ACR122L is very similar to the CCID protocol.

ACR122L Command Frame Format

STX (02h)	Bulk-OUT Header	APDU Command or Parameters	Checksum	ETX (03h)
1 Byte	10 Bytes	M Bytes (If applicable)	1 Byte	1 Byte

ACR122L Status Frame Format

STX (02h)	Status	Checksum	ETX (03h)
1 Byte	1 Byte	1 Byte	1 Byte

ACR122L Response Frame Format

STX (02h)	Bulk-IN Header	APDU Response or abData	Checksum	ETX (03h)
1 Byte	10 Bytes	N Bytes (If applicable)	1 Byte	1 Byte

Checksum = XOR {Bulk-OUT Header, APDU Command or Parameters}

Checksum = XOR {Bulk-IN Header, APDU Response or abData}

For control SAM Socket 1, the STX must be equal to 02h and ETX must be equal to 03h.

For control SAM Socket 2, the STX must be equal to 12h and ETX must be equal to 13h.

For control SAM Socket 3, the STX must be equal to 22h and ETX must be equal to 23h.

For control access Contactless interface, Peripherals (i.e. LEDs, LCD and Buzzer), the STX must be equal to 02h and ETX must be equal to 03h, which is the same with control SAM Socket1.

In general, we would make use of three types of Bulk-OUT Header:

- **HOST_to_RDR_IccPowerOn:** To activate the SAM interface. The ATR of the SAM will be returned if available.
- **HOST_to_RDR_IccPowerOff:** To deactivate the SAM interface.
- **HOST_to_RDR_XfrBlock:** To exchange APDUs between the host and ACR122L.

The SAM1 interface must be activated in order to use the contactless interface and peripherals. In short, all the APDUs are exchanged through the SAM interface.

Similarly, two types of Bulk-IN Header are used:

- **RDR_to_HOST_DataBlock:** In response to the HOST_to_RDR_IccPowerOn and HOST_to_RDR_XfrBlock Frames.
- **RDR_to_HOST_SlotStatus:** In response to the HOST_to_RDR_IccPowerOff Frame.

RDR = ACR122L; HOST = Host Controller.

HOST_to_RDR = Host Controller -> ACR122L

RDR_to_HOST = ACR122L -> Host Controller

4.1. Protocol Flow Examples

(Use SAM Interface 1 as Example)

A. Activate a SAM.

	HOST			RDR
1.	HOST sends a frame.	→	02 62 00 00 00 00 00 01 01 00 00 [Checksum] 03	
2.	RDR sends back a positive status frame immediately.		02 00 00 03 (positive status frame)	←
			.. After some processing delay...	
3.	RDR sends back the response of the command.		02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [Checksum] 03	←

B. Activate a SAM (Incorrect Checksum, HOST)

	HOST			RDR
1.	HOST sends a corrupted frame.	→	02 62 00 00 00 00 00 01 01 00 00 [Incorrect Checksum] 03	
2.	RDR sends back a negative status frame immediately.		02 FF FF 03 (negative status frame)	←
3.	HOST sends the frame again.	→	02 62 00 00 00 00 00 01 01 00 00 [Checksum] 03	
4.	RDR sends back a positive status frame immediately.		02 00 00 03 (positive status frame)	←
			.. After some processing delay..	
5.	RDR sends back the response of the command.		02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [Checksum] 03	←

C. Activate a SAM (Incorrect Checksum, RDR).

HOST			RDR
1. HOST sends a frame.	→	02 62 00 00 00 00 00 01 01 00 00 [Checksum] 03	
2. RDR sends back a positive status frame immediately.		02 00 00 03 (positive status frame)	←
		.. After some processing delay...	
3. RDR sends back the response (corrupted) of the command.	→	02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [Incorrect Checksum] 03	←
4. HOST sends a NAK frame to get the response again.		02 00 00 00 00 00 00 00 00 00 00 00 03 (NAK)	
5. RDR sends back the response of the command.		02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [Checksum] 03	←

Note: *If the frame sent by the HOST is correctly received by the RDR, a positive status frame = {02 00 00 03} will be sent to the HOST immediately to inform the HOST the frame is correctly received. The HOST has to wait for the response of the command. The RDR will not receive any more frames while the command is being processed.*

In case of errors, a negative status frame will be sent to the HOST to indicate the frame is either corrupted or incorrectly formatted.

CheckSum Error Frame = {02 FF FF 03}

Length Error Frame = {02 FE FE 03}. The length "dDwLength" is greater than 0105h bytes.

ETX Error Frame = {02 FD FD 03}. The last byte is not equal to ETX "03h".

TimeOut Error Frame = {02 FC FC 03}. Not Complete Package Received.

The NAK Frame is only used by the HOST to get the last response.

{02 00 00 00 00 00 00 00 00 00 00 00 03} // 11 zeros

5.0. SAM Interface

The ACR122L comes with three SAM interfaces.

5.1. Activating the SAM Interface

ACR122L Command Frame Format

STX	Bulk-OUT Header (HOST_to_RDR_IccPowerOn)	Parameters	Checksum	ETX
1 Byte	10 Bytes	0 Byte	1 Byte	1 Byte

For SAM Interface 1, STX = 02h and ETX = 03h

For SAM Interface 2, STX = 12h and ETX = 13h

For SAM Interface 3, STX = 22h and ETX = 23h

HOST_to_RDR_IccPowerOn Format

Offset	Field	Size	Value	Description
0	*bMessageType*	1	62h	
1	*dDwLength* *<LSB .. MSB>*	4	00000000h	Message-specific data length.
5	*bSlot*	1	00-FFh	Identifies the slot number for this command. Default=00h.
6	*bSeq*	1	00-FFh	Sequence number for command.
7	*bPowerSelect*	1	00h 01h 02h 03h	Voltage that is applied to the ICC: 00h – Automatic Voltage Selection 01h – 5.0 V 02h – 3.0 V 03h – 1.8 V
8	*abRFU*	2		Reserved for Future Use

ACR122L Response Frame Format

STX	Bulk-IN Header (RDR_to_HOST_DataBlock)	abData	Checksum	ETX
1 Byte	10 Bytes	N Bytes (ATR)	1 Byte	1 Byte

For SAM Interface 1, STX = 02h and ETX = 03h

For SAM Interface 2, STX = 12h and ETX = 13h

For SAM Interface 3, STX = 22h and ETX = 23h

RDR_to_HOST_DataBlock Format

Offset	Field	Size	Value	Description
0	*bMessageType*	1	80h	Indicates that a data block is being sent from the ACR122L.
1	*dwLength* <*LSB .. MSB*>	4	N	Size of *abData* field (N Bytes).
5	*bSlot*	1	Same as Bulk-OUT	Identifies the slot number for this command.
6	*bSeq*	1	Same as Bulk-OUT	Sequence number for corresponding command.
7	*bStatus*	1		
8	*bError*	1		
9	*bChainParameter*	1		

Example 1: To activate the SAM Interface 1 slot 0 (default), sequence number = 1, 5 V card.

HOST -> **02** 62 00 00 00 00 00 01 01 00 00 [Checksum] **03**

RDR -> **02** 00 00 **03**

RDR -> **02** 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [Checksum] **03**

The ATR = 3B 2A 00 80 65 24 B0 00 02 00 82; SW1 SW2 = 90 00

Example 2: To activate the SAM Interface 2 slot 0 (default), sequence number = 1, 5 V card.

HOST -> **12** 62 00 00 00 00 00 01 01 00 00 [Checksum] **13**

RDR -> **12** 00 00 **13**

RDR -> **12** 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [Checksum] **13**

The ATR = 3B 2A 00 80 65 24 B0 00 02 00 82; SW1 SW2 = 90 00

Example 3: To activate the SAM Interface 3 slot 0 (default), sequence number = 1, 5 V card.

HOST -> **22** 62 00 00 00 00 00 01 01 00 00 [Checksum] **23**

RDR -> **22** 00 00 **23**

RDR -> **22** 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [Checksum] **23**

The ATR = 3B 2A 00 80 65 24 B0 00 02 00 82; SW1 SW2 = 90 00

5.2. Deactivating the SAM Interface

ACR122L Command Frame Format

STX	Bulk-OUT Header (HOST_to_RDR_IccPowerOff)	Parameters	Checksum	ETX
1 Byte	10 Bytes	0 Byte	1 Byte	1 Byte

For SAM Interface 1, STX = 02h and ETX = 03h

For SAM Interface 2, STX = 12h and ETX = 13h

For SAM Interface 3, STX = 22h and ETX = 23h

HOST_to_RDR_IccPowerOff Format

Offset	Field	Size	Value	Description
0	*bMessageType*	1	63h	
1	*dDwLength <LSB .. MSB>*	4	00000000h	Message-specific data length.
5	*bSlot*	1	00-FFh	Identifies the slot number for this command. Default=00h.
6	*bSeq*	1	00-FFh	Sequence number for command.
7	*abRFU*	3		Reserved for Future Use.

ACR122L Response Frame Format

STX	Bulk-IN Header (RDR_to_HOST_SlotStatus)	abData	Checksum	ETX
1 Byte	10 Bytes	0 Byte	1 Byte	1 Byte

For SAM Interface 1, STX = 02h and ETX = 03h

For SAM Interface 2, STX = 12h and ETX = 13h

For SAM Interface 3, STX = 22h and ETX = 23h

RDR_to_HOST_DataBlock Format

Offset	Field	Size	Value	Description
0	*bMessageType*	1	81h	Indicates that a data block is being sent from the ACR122L.
1	*dwLength <LSB .. MSB>*	4	0	Size of *abData* field (0 Bytes).
5	*bSlot*	1	Same as Bulk-OUT	Identifies the slot number for this command.
6	*bSeq*	1	Same as Bulk-OUT	Sequence number for corresponding command.
7	*bStatus*	1		
8	*bError*	1		
9	*bClockStatus*	1		

Example 1: To deactivate the SAM Interface 1 slot 0 (default), sequence number = 2.

HOST -> 02 63 00 00 00 00 00 02 00 00 00 [Checksum] 03

RDR -> 02 00 00 03

RDR -> 02 81 00 00 00 00 00 02 00 00 00 [Checksum] 03

Example 2: To deactivate the SAM Interface 2 slot 0 (default), sequence number = 2.

HOST -> 12 63 00 00 00 00 00 02 00 00 00 [Checksum] 13

RDR -> 12 00 00 13

RDR -> 12 81 00 00 00 00 00 02 00 00 00 [Checksum] 13

Example 3: To deactivate the SAM Interface 3 slot 0 (default), sequence number = 2.

HOST -> 22 63 00 00 00 00 00 02 00 00 00 [Checksum] 23

RDR -> 22 00 00 23

RDR -> 22 81 00 00 00 00 00 02 00 00 00 [Checksum] 23

5.3. Exchanging data through the SAM Interface

ACR122L Command Frame Format

STX	Bulk-OUT Header (HOST_to_RDR_XfrBlock)	Parameters	Checksum	ETX
1 Byte	10 Bytes	M Byte	1 Byte	1 Byte

For SAM Interface 1, STX = 02h and ETX = 03h

For SAM Interface 2, STX = 12h and ETX = 13h

For SAM Interface 3, STX = 22h and ETX = 23h

HOST_to_RDR_XfrBlock Format

Offset	Field	Size	Value	Description
0	*bMessageType*	1	6Fh	
1	*dDwLength <LSB .. MSB>*	4	M	Message-specific data length.
5	*bSlot*	1	00-FFh	Identifies the slot number for this command. Default=00h.
6	*bSeq*	1	00-FFh	Sequence number for command.
7	*bBWI*	1	00-FFh	Used to extend the Block Waiting Timeout.
8	*wLevelParameter*	2	0000h	

ACR122L Response Frame Format

STX	Bulk-IN Header (RDR_to_HOST_DataBlock)	abData	Checksum	ETX
1 Byte	10 Bytes	N Bytes (ATR)	1 Byte	1 Byte

For SAM Interface 1, STX = 02h and ETX = 03

For SAM Interface 2, STX = 12h and ETX = 13h

For SAM Interface 3, STX = 22h and ETX = 23h

RDR_to_HOST_DataBlock Format

Offset	Field	Size	Value	Description
0	*bMessageType*	1	80h	Indicates that a data block is being sent from the ACR122L.
1	*dwLength* *<LSB .. MSB>*	4	N	Size of *abData* field (N Bytes).
5	*bSlot*	1	Same as Bulk-OUT	Identifies the slot number for this command.
6	*bSeq*	1	Same as Bulk-OUT	Sequence number for corresponding command.
7	*bStatus*	1		
8	*bError*	1		
9	*bChainParameter*	1		

Example 1: To send an APDU “80 84 00 00 08” to the SAM Interface 1 slot 0 (default), sequence number = 3.

HOST -> 02 6F 05 00 00 00 00 03 00 00 00 80 84 00 00 08 [Checksum] 03

RDR -> 02 00 00 03

RDR -> 02 80 0A 00 00 00 00 03 00 00 00 E3 51 B0 FC 88 AA 2D 18 90 00 [Checksum] 03

Response = E3 51 B0 FC 88 AA 2D 18; SW1 SW2 = 90 00

Example 2: To send an APDU “80 84 00 00 08” to the SAM Interface 2 slot 0 (default), sequence number = 3.

HOST -> 12 6F 05 00 00 00 00 03 00 00 00 80 84 00 00 08 [Checksum] 13

RDR -> 12 00 00 13

RDR -> 12 80 0A 00 00 00 00 03 00 00 00 E3 51 B0 FC 88 AA 2D 18 90 00 [Checksum] 13

Response = E3 51 B0 FC 88 AA 2D 18; SW1 SW2 = 90 00

Example 3: To send an APDU "80 84 00 00 08" to the SAM Interface 3 slot 0 (default), sequence number = 3.

HOST -> 22 6F 05 00 00 00 00 03 00 00 00 80 84 00 00 08 [Checksum] 23

RDR -> 22 00 00 23

RDR -> 22 80 0A 00 00 00 00 03 00 00 00 E3 51 B0 FC 88 AA 2D 18 90 00 [Checksum] 23

Response = E3 51 B0 FC 88 AA 2D 18; SW1 SW2 = 90 00

6.0. Pseudo-APDUs for contactless interface and peripherals control

ACR122L comes with two primitive commands for this purpose <Class FFh>.

***Note:** For all the pseudo-APDUs below (except sections 5.2 – Changing the communication speed and 5.3 – Get firmware version), STX MUST BE EQUAL to 02h and ETX MUST BE EQUAL to 03h.*

6.1. Direct Transmit

This command is used to send a pseudo-APDU (Tag Commands), and returns the length of the Response Data.

Direct Transmit Command Format (Length of the Tag Command + 5 Bytes)

Command	Class	INS	P1	P2	Lc	Data In	
Direct Transmit	FFh	00h	00h	00h	Number of bytes to send	Tag Command	Data

Where:

Lc Number of bytes to send (1 Byte).

Maximum 255 bytes.

Data In Tag Command.

The data to be sent to the tag.

Direct Transmit Response Format (TAG Response + Data + 2 Bytes)

Item	Command	Data			Meaning
1	D4 40	Tg		[DataOut[]]	Tag Exchange Data
2	D4 4A	MaxTg	BrTy	[InitiatorData[]]	Tag Polling

Where:

Tg A byte containing the logical number of the relevant target. This byte also contains the *More Information* (MI) bit (bit 6). When the MI bit is set to 1, this indicates that the host controller wants to send more data which is all the data contained in the DataOUT[] array. This bit is only valid for a TPE target.

DataOut An array of raw data (from 0 up to 262 bytes) to be sent to the target by the contactless chip.

MaxTg Maximum number of targets to be initialized by the contactless chip. The chip is capable of handling 2 targets maximum at once, so this field should not exceed 02h.

Brty Baud rate and the modulation type to be used during the initialization.

00h: 106 kbps type A (ISO/IEC14443 Type A),

01h: 212 kbps (FeliCa polling),

02h: 424 kbps (FeliCa polling),

03h: 106 kbps type B (ISO/IEC 14443-3B),

04h: 106 kbps Innovision Jewel tag.

InitiatorData[] An array of data to be used during the initialization of the target(s). Depending on the Baud Rate specified, the content of this field is different.

106 kbps type A

The field is optional and is present only when the host controller wants to initialize a target with a known UID.

In that case, InitiatorData[] contains the UID of the card (or part of it). The UID must include the cascade tag CT if it is cascaded level 2 or 3.

Cascade Level 1

UID1	UID2	UID3	UID4

Cascade Level 2

UID1	UID2	UID3	UID4	UID5	UID6	UID7

Cascade Level 3

UID1	UID2	UID3	UID4	UID5	UID6	UID7	UID8	UID9	UID10

106 kbps type B

In this case, InitiatorData[] is formatted as following:

AFI (1byte)	[Polling Method]

AFI The AFI (Application Family Identifier) parameter represents the type of application targeted by the device IC and is used to preselect the PICCs before the ATQB.

This field is mandatory.

Polling Method This field is optional. It indicates the approach to be used in the ISO/IEC 14443-3B initialization:

If bit 0 = 1: Probabilistic approach (option 1) in the ISO/IEC 14443-3B initialization,

If bit 0 = 0: Timeslot approach (option 2) in the ISO/IEC 14443-3B initialization,

If this field is absent, the timeslot approach will be used.

212/424 kbps In that case, this field is mandatory and contains the complete pay load information that should be used in the polling request command (5bypes, length bytes is excluded)

106 kbps InnoVision Jewel tag. This field is not used.

Data Out Tag Response returned by the reader.

Direct Transmit Response Format

<table>
<tr><th>Response</th><th colspan="5">Data Out</th></tr>
<tr><td rowspan="2">Result</td><td>D5 41</td><td>Status</td><td colspan="2">[DataIn[]]</td><td rowspan="2">SW1 SW2</td></tr>
<tr><td>D5 4B</td><td>NbTg</td><td>[TargetData1[]]</td><td>[TargetData2[]]</td></tr>
</table>

Where:

Status A byte indicating if the process has been terminated successfully or not. When in either DEP or ISO/IEC 14443-4 PCD mode, this byte also indicates if *NAD (Node Address)* is used and if the transfer of data is not completed with bit *More Information*.

DataIn An array of raw data (from 0 up to 262 bytes) received by the contactless chip.

NbTg The number of initialized Targets (minimum 0, maximum 2 targets).

TargetDatai[] The "i" in TargetDatai[] refers to "1" or "2". This contains the information about the detected targets and depends on the baud rate selected. The following information is given for one target, it is repeated for each target initialized (NbTg times).

106 kbps Type A

Tg	SENS_RES10 (2 bytes)	SEL_RES (1 byte)	NFCIDLength (1 byte)	NFCID1[] (NFCIDLength bytes)	[ATS[]] (ATSLength bytes11))

106 kbps Type B

Tg	ATQB Response (12 bytes)	ATTRIB_RES Length (1 byte)	ATTTRIB_RES[] (ATTRIB_RES Length)

212/424 kbps

Tg	POL_RES length	01h (response code)	NFCID2t	Pad	SYST_CODE (optional)
1 byte	1 byte	1 byte	8 bytes	8 bytes	2 bytes
	POL_RES (18 or 20 bytes)				

106 kbps Innovision Jewel tag

Tg	SENS_RES (2 bytes)	JEWELID[] (4 bytes)

Data Out SW1 SW2. Status Code returned by the reader.

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.
Time Out Error	63	01h	The TAG does not response.
Checksum Error	63	27h	The checksum of the Response is wrong.

Results	SW1	SW2	Meaning
Parameter Error	63	7Fh	The TAG Command is wrong.

6.2. Change Communication Speed

This command is used to change the baud rate.

Note: *STX = 32h and ETX = 33h*

Baud Rate Control Command Format (9 Bytes)

Command	Class	INS	P1	P2	Lc
Baud Rate Control	FFh	00h	44h	New Baud Rate	00h

Where:

P2 New Baud Rate.

00h = Set the new baud rate to 9600 bps.

01h = Set the new baud rate to 115200 bps.

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	Current Baud Rate	The operation is completed successfully.
Error	63	00h	The operation is failed.

Where:

SW2 Current Baud Rate.

00h = The current baud rate is 9600 bps.

01h = The current baud rate is 115200 bps.

Note: *After the communication speed is changed successfully, the program has to adjust its communication speed to continue the rest of the data exchanges.*

The initial communication speed is determined by the existence of R12 (0 ohm).

- With R12 = 115200 bps
- Without R12 = 9600 bps (default)

Example 1: To initialize a FeliCa Tag (Tag Polling).

Step 1. Issue a “Direct Transmit” APDU.

The APDU Command should be “FF 00 00 00 09 D4 4A 01 01 00 FF FF 01 00”

In which,

Direct Transmit APDU = “FF 00 00 00”

Length of the Tag Command = “09”

Tag Command (InListPassiveTarget 212Kbps) = “D4 4A 01 01”

Tag Command (System Code Request) = “00 FF FF 01 00”

To send an APDU to the slot 0 (default), sequence number = 1.

HOST -> 02 6F 0E 00 00 00 00 01 00 00 00
FF 00 00 00 09 D4 4A 01 01 00 FF FF 01 00 [Checksum] 03

RDR -> 02 00 00 03

RDR -> 02 81 1A 00 00 00 00 01 00 00 00
D5 4B 01 01 14 01 01 01 05 01 86 04 02 02 03 00
4B 02 4F 49 8A 8A 80 08 90 00 [Checksum] 03

The APDU Response is “D5 4B 01 01 14 01 01 01 05 01 86 04 02 02 03 00 4B 02 4F 49 8A 8A 80 08 90 00”

In which,

Response returned by the contactless chip = “D5 4B 01 01 14 01 01 01 05 01 86 04 02 02 03 00 4B 02 4F 49 8A 8A 80 08”

NFCID2t of the FeliCa Tag = “01 01 05 01 86 04 02 02”

Status Code returned by the reader = “90 00”

Example 2: To write 16 bytes data to the FeliCa Tag (Tag Write).

Step 1. Issue a “Direct Transmit” APDU.

The APDU Command should be “FF 00 00 00 23 D4 40 01 20 08 01 01 05 01 86 04 02 02 01 09 01 01 80 00 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA”

In which,

Direct Transmit APDU = “FF 00 00 00”

Length of the Tag Command = “23”

Tag Command (InDataExchange) = “D4 40 01”

Tag Command (Write Data) = “20 08 01 01 05 01 86 04 02 02 01 09 01 01 80 00 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA”.

To send an APDU to the slot 0 (default), sequence number = 2.

HOST -> 02 6F 26 00 00 00 00 02 00 00 00
FF 00 00 00 21 D4 40 01 20 08 01 01 05 01 86
04 02 02 01 09 01 01 80 00 00 AA 55 AA 55 AA 55
AA 55 AA 55 AA 55 AA
[Checksum] 03

RDR -> 02 00 00 03

RDR -> 02 81 11 00 00 00 00 02 00 00 00

D5 41 00 0C 09 01 01 05 01 86 04 02 02 00 00 90 00
[Checksum] 03

The APDU Response would be “D5 41 00 0C 09 01 01 05 01 86 04 02 02 00 00 90 00”

In which,

Response returned by the contactless chip = “D5 41”

Response returned by the FeliCa Tag = “00 0C 09 01 01 05 01 86 04 02 02 00 00”

Status Code returned by the reader = “90 00”

Example 3: To read 16 bytes data from the FeliCa Tag (Tag Write).

Step 1. Issue a “Direct Transmit” APDU.

The APDU Command should be “FF 00 00 00 13 D4 40 01 10 06 01 01 05 01 86 04 02 02 01 09 01 01 80 00”

In which,

Direct Transmit APDU = “FF 00 00 00”

Length of the Tag Command = “13”

Tag Command (InDataExchange) = “D4 40 01”

Tag Command (Read Data) = “10 06 01 01 05 01 86 04 02 02 01 09 01 01 80 00”

To send an APDU to the slot 0 (default), sequence number = 3.

HOST -> 02 6F 18 00 00 00 00 03 00 00 00
FF 00 00 00 13 D4 40 01 10 06 01 01 05 01 86 04
02 02 01 09 01 01 80 00
[Checksum] 03

RDR -> 02 00 00 03

RDR -> 02 81 22 00 00 00 00 03 00 00 00
D5 41 00 1D 07 01 01 05 01 86 04 02 02 00 00 01 00
AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 90 00
[Checksum] 03

The APDU Response would be

“D5 41 00 1D 07 01 01 05 01 86 04 02 02 00 00 01 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 90 00”

In which,

Response returned by the contactless chip = “D5 41”

Response returned by the FeliCa Tag =

"00 1D 07 01 01 05 01 86 04 02 02 00 00 01 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA"

Status Code returned by the reader = "90 00"

Example 4: To initialize an ISO 14443-4 Type B Tag (Tag Polling).

Step 1. Issue a "Direct Transmit" APDU.

The APDU Command should be "FF 00 00 00 05 D4 4A 01 03 00"

In which,

Direct Transmit APDU = "FF 00 00 00"

Length of the Tag Command = "05"

Tag Command (InListPassiveTarget Type B 106Kbps) = "D4 4A 01 03 00"

To send an APDU to the slot 0 (default), sequence number = 4.

HOST -> 02 6F 0A 00 00 00 00 04 00 00 00
FF 00 00 00 05 D4 4A 01 03 00
[Checksum] 03

RDR -> 02 00 00 03

RDR -> 02 81 14 00 00 00 00 04 00 00 00
D5 41 01 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21
90 00 [Checksum] 03

The APDU Response is

"D5 4B 01 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21 90 00"

In which,

Response returned by the contactless chip = "D5 4B 01 01"

ATQB of the Type B Tag = "50 00 01 32 F4 00 00 00 00 33 81 81"

CRC-B = "01 21"

Status Code returned by the reader = "90 00"

Example 5: To send an APDU to an ISO 14443-4 Type B Tag (Data Exchange).

Step 1. Issue a "Direct Transmit" APDU.

The USER APDU Command should be "00 84 00 00 08"

The Composed APDU Command should be "FF 00 00 00 08 D4 40 01 00 84 00 00 08"

In which,

Direct Transmit APDU = "FF 00 00 00"

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

Length of the Tag Command = "08"

Tag Command (InDataExchange) = "D4 40 01"

Tag Command (Get Challenge) = "00 84 00 00 08"

To send an APDU to the slot 0 (default), sequence number = 5.

HOST -> 02 6F 0D 00 00 00 00 05 00 00 00
FF 00 00 00 08 D4 40 01 00 84 00 00 08
[Checksum] 03

RDR -> 02 00 00 03

RDR -> 02 81 0F 00 00 00 00 05 00 00 00
D5 41 00 01 02 03 04 05 06 07 08 90 00 90 00
[Checksum] 03

The APDU Response is "D5 41 00 0B 01 02 03 04 05 06 07 08 90 00"

In which,

Response returned by the contactless chip = "D5 41 00"

Response from the Type B Tag = "01 02 03 04 05 06 07 08 90 00"

Status Code returned by the reader = "90 00"

6.3. Get firmware version

This command is used to derive the firmware version of the reader.

For SAM Interface 1 controller, STX = 02h and ETX = 03h

For SAM Interface 2 controller, STX = 12h and ETX = 13h

For SAM Interface 3 controller, STX = 22h and ETX = 23h

Get Firmware Version Command Format (5 Bytes)

Command	Class	INS	P1	P2	Le
Get Response	FFh	00h	48h	00h	00h

Where:

Le Number of bytes to retrieve (1 Byte).
Maximum 255 bytes.

For SAM Interface 1 controller, the feedback's STX = 02h and ETX = 03h

For SAM Interface 2 controller, the feedback's STX = 12h and ETX = 13h

For SAM Interface 3 controller, the feedback's STX = 22h and ETX = 23h

Get Firmware Version Response Format (10 Bytes)

Response	Data Out
Result	Firmware Version

Example 1: Response for SAM Interface 1 controller.

= 41 43 52 31 32 32 4C 31 30 31 53 41 4D 31(Hex) = ACR122L101SAM1 (ASCII)

Example 2: Response for SAM Interface 2 controller.

= 41 43 52 31 32 32 4C 31 30 31 53 41 4D 32(Hex) = ACR122L101SAM2 (ASCII)

Example 3: Response for SAM Interface 3 controller.

= 41 43 52 31 32 32 4C 31 30 31 53 41 4D 33(Hex) = ACR122L101SAM3 (ASCII)

Note: *The device firmware version is the response from SAM Interface 1 controller.*

6.4. LCD Display (ASCII Mode)

This command is used to display the LCD Message in ASCII Mode.

LCD Display Command Format (5 Bytes + LCD Message Length)

Command	Class	INS	P1	P2	Lc	Data In (Max. 16 Bytes)
LCD Display	FFh	Option Byte	68h	LCD XY Position	LCD Message Length	LCD Message

INS Option Byte (1 Byte)

CMD	Item	Description
Bit 0	Character Bold Font	1 = Bold; 0 = Normal
Bit 1 - 3	Reserved	
Bit 4 - 5	Table Index	00 = Fonts Set A 01 = Fonts Set B 10 = Fonts Set C
Bits 6 – 7	Reserved	

P2 LCD XY Position. The character to be displayed on the LCD position specified by DDRAM Address.

Please follow the DDRAM table below for the LCD character position's representation.

Row\Col	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	DISPLAY POSITION
1st LINE	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	LCD XY POSITION
2nd LINE	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	

Table 2: DDRAM Address for Font Sets 1 and 2

Row\Col	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	DISPLAY POSITION
1st LINE	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	LCD XY POSITION
2nd LINE	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	
3rd LINE	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	
4th LINE	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F	

Table 3: DDRAM Address for Font Set 3

Lc LCD Message Length.

The length of the LCD message (max. 10h); If the message length is longer than the number of characters that the LCD screen can show, then the redundant character will not be shown on the LCD.

Data In LCD Message.

The data to be sent to LCD, maximum 16 Character for each line;

Please follow the Font tables (selected by INS Bit 4 - 5) below for the LCD Character Index.

Note: *Size of the characters in Font Set A and Font Set B is 8 x 16, but size of the characters in Font Set C is 8 x 8.*

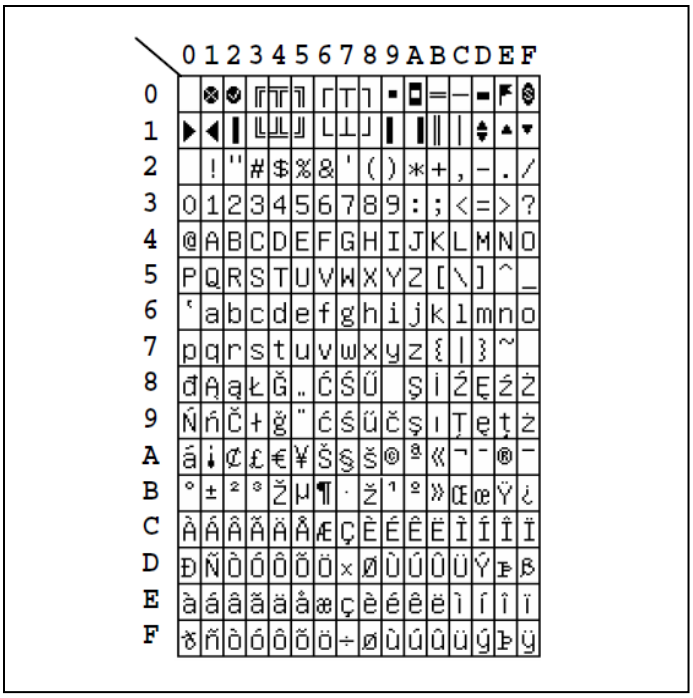


Figure 2: Character Set A

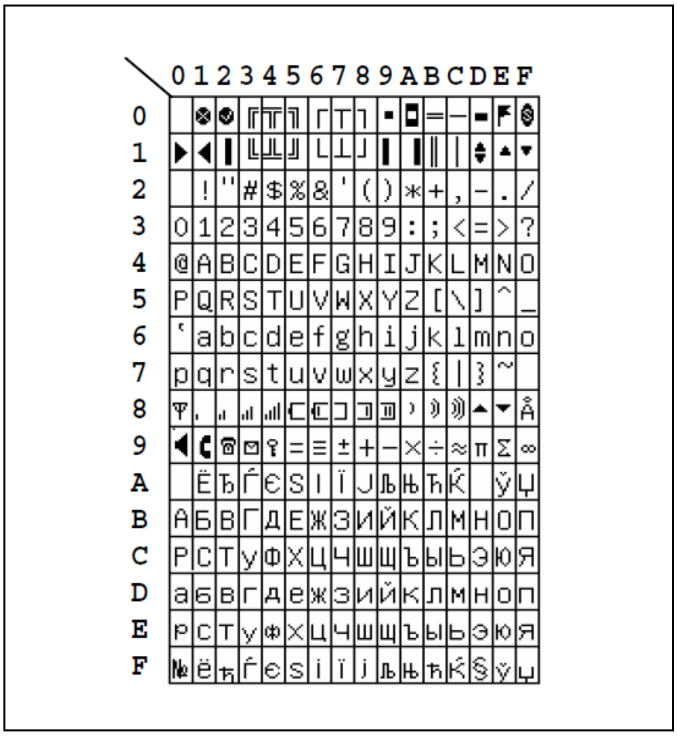


Figure 3: Character Set B

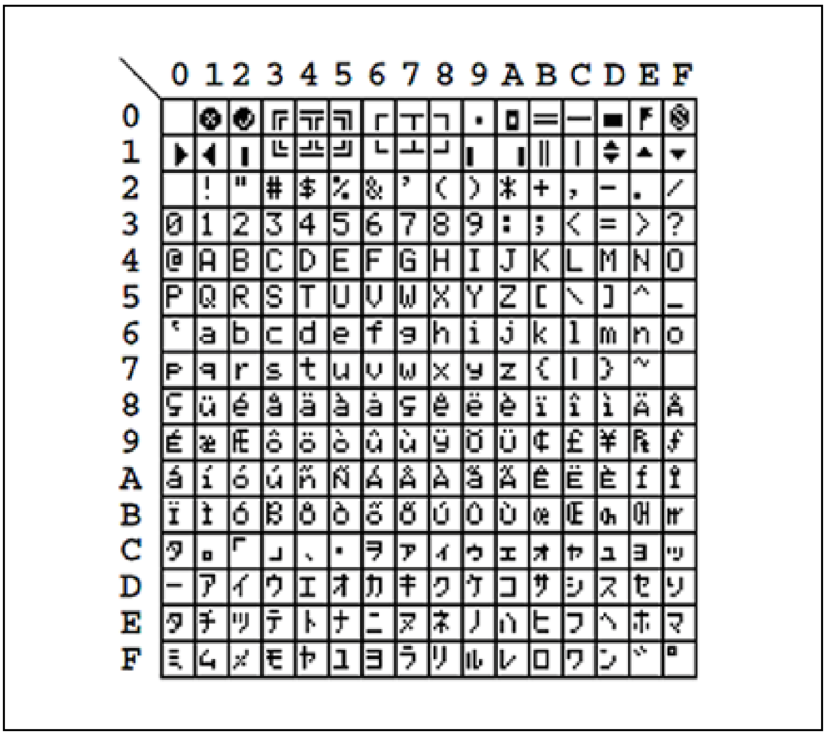


Figure 4: Character Set C

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.5. LCD Display (GB Mode)

This command is used to display the LCD Message in GB Mode.

LCD Display Command Format (5 Bytes + LCD Message Length)

Command	Class	INS	P1	P2	Lc	Data In (Max. 16 Bytes)
LCD Display	FFh	Option Byte	69h	LCD XY Position	LCD Message Length	LCD Message

INS Option Byte (1 Byte)

CMD	Item	Description
Bit 0	Character Bold Font	1 = Bold; 0 = Normal
Bit 1 - 7	Reserved	

P2 LCD XY Position.

The character to be displayed on the LCD position specified by DDRAM Address.

Please follow the DDRAM table below for the LCD character position's representation.

<table>
<tr><th></th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>DISPLAY POSITION</th></tr>
<tr><td>FIRST LINE</td><td colspan="2">00</td><td colspan="2">01</td><td colspan="2">02</td><td colspan="2">03</td><td colspan="2">04</td><td colspan="2">05</td><td colspan="2">06</td><td colspan="2">07</td><td rowspan="2">LCD XY POSITION</td></tr>
<tr><td>SECOND LINE</td><td colspan="2">40</td><td colspan="2">41</td><td colspan="2">42</td><td colspan="2">43</td><td colspan="2">44</td><td colspan="2">45</td><td colspan="2">46</td><td colspan="2">47</td></tr>
</table>

Table 4: LCD Character Position Representation

Lc LCD Message Length.

The length of the LCD message (max. 10h); If the message length is longer than the number of characters that the LCD screen can show, then the redundant character will not be shown on the LCD.

The length of the LCD message should be a multiple of 2 because each Chinese Character (GB code) should contain two bytes.

Data In LCD Message.

The data to be sent to the LCD, maximum of 8 (2 x 8bit each character) characters for each line. Please follow the Fonts table of GB Coding.

If ASCII code is to be sent at this mode, the number of characters should be a multiple of 2, otherwise, add 00h after the last character.

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.6. LCD Display (Graphic Mode)

This command is used to display the LCD Message in Graphic Mode.

LCD Display Command Format (5 Bytes + LCD Message Length)

Command	Class	INS	P1	P2	Lc	Data In (max. 128 Bytes)
LCD Display	FFh	00h	6Ah	Line Index	Pixel Data Length	Pixel Data

Where:

P2 Line Index. To set which line to start to update the LCD Display (Refer to below LCD Display Position).

Lc Pixel Data Length. The length of the pixel data (max. 80h).

Data In Pixel Data. The pixel data to be sent to LCD for display.

	Byte 00h (X = 00h)								Byte 01h (X = 01h)								…	Byte 0Fh (X = 0Fh)							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	…	7	6	5	4	3	2	1	0
00h																									
01h																									
02h																									
03h																									
04h																									
05h																									
06h																									
07h																									
08h																									
09h																									
…	…																								
1Fh																									

Figure 5: LCD Display Position

Total LCD Size: 128 x 32.

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.7. Scroll Current LCD Display

This command is used to set the scrolling feature of the current LCD Display.

Scrolling LCD Command Format (5 Bytes + LCD Message Length)

Command	Class	INS	P1	P2	Lc	Data In (6 Bytes)
Scroll LCD	FFh	00h	6Dh	00h	06h	Scroll Ctrl

Data In Scroll Ctrl.

Scrolling Control Format (6 Bytes)

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
00h	00h	0Fh	1Fh	Refresh Speed Ctrl	Scrolling Direction

Where:

Refresh Speed Ctrl Bit 0 ~ Bit 3 – Number of pixels to move per scroll.

Bit 4 ~ Bit 7 – Scrolling period.

Bit 7	Bit 6	Bit 5	Bit 4	Scrolling Period
0	0	0	0	1 Unit
0	0	0	1	3 Units
0	0	1	0	5 Units
0	0	1	1	7 Units
0	1	0	0	17 Units
0	1	0	1	19 Units
0	1	1	0	21 Units
0	1	1	1	23 Units
1	0	0	0	129 Units
1	0	0	1	131 Units
1	0	1	0	133 Units
1	0	1	1	135 Units
1	1	0	0	145 Units
1	1	0	1	147 Units

Bit 7	Bit 6	Bit 5	Bit 4	Scrolling Period
1	1	1	0	149 Units
1	1	1	1	151 Units

Table 5: Scrolling Period

Bit 1	Bit 0	Scrolling Direction
0	0	From Left to Right
0	1	From Right to Left
1	0	From Top to Bottom
1	1	From Bottom to Top

Table 6: Scrolling Direction

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.8. Pause LCD Scrolling

This command is used to pause the LCD scrolling that has been previously set.

To resume the scrolling, send the Scrolling LCD command (section 5.7 – Scroll Current LCD Display) again.

Pause Scrolling Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
Pause Scroll LCD	FFh	00h	6Eh	00h	00h

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.9. Stop LCD Scrolling

This command is used to stop the LCD scrolling that has been previously set. The LCD display will return to normal display position.

Stop Scrolling LCD Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
Stop Scroll LCD	FFh	00h	6Fh	00h	00h

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.10. Clear LCD

This command is used to clear all contents shown in the LCD.

Clear LCD Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
Clear LCD	FFh	00h	60h	00h	00h

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

***Note:** For ACR122L firmware versions 307 and above, using the Clear LCD function successively with other LCD functions requires the application to handle an additional 100 ms time delay.*

6.11. LCD Backlight Control

This command is used to control the LCD backlight.

LCD Backlight Control Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
LCD Backlight Control	FFh	00h	64h	Backlight Control	00h

P2 Backlight Control.

Backlight Control Format (1 Byte)

CMD	Description
00h	LCD Backlight Off
FFh	LCD Backlight On

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.12. LCD Contrast Control

This command is used to control the LCD contrast.

LCD Contrast Control Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
LCD Contrast Control	FFh	00h	6Ch	Contrast Control	00h

Where:

P2 Contrast Control.

The value range is between 00h (brightest) to 0Fh (darkest).

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.13. LED Enable/Disable

This command is used to enable/disable the LEDs by user.

Note: *Default "Disable." LED control performed by the firmware.*

LED Control Enable Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
LED Control Enable	FFh	00h	43h	bLEDCtrl	00h

P2 bCtrlEable (1 Byte).

CMD	Description
00h	Disable LED Control by user
FFh	Enable LED Control by user

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.14. LED Control

This command is used to control the four LEDs.

LED Control Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
LED Control	FFh	00h	41h	bLEDsState	00h

P2 bLEDsState.

LED_0, LED_1, LED_2 and LED_3 Control Format (1 Byte)

CMD	Item	Description
Bit 0	LED_0 State	1 = On; 0 = Off
Bit 1	LED_1 State	1 = On; 0 = Off
Bit 2	LED_2 State	1 = On; 0 = Off
Bit 3	LED_3 State	1 = On; 0 = Off
Bits 4 – 7	Reserved	

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.15. LED and Buzzer Control

This command is used to control the states of the LED_0, LED_1 and Buzzer.

LED_0, LED_1 and Buzzer Control Command Format (9 Bytes)

Command	Class	INS	P1	P2	Lc	Data In (4 Bytes)
LEDs and Buzzer Control	FFh	00h	40h	LED State Control	04h	Blinking Duration Control

P2 LED State Control.

LED_0, LED_1 and Buzzer Control Format (1 Byte)

CMD	Item	Description
Bit 0	Final LED_1 State	1 = On; 0 = Off
Bit 1	Final LED_0 State	1 = On; 0 = Off
Bit 2	LED_1 State Mask	1 = Update the State 0 = No change
Bit 3	LED_0 State Mask	1 = Update the State 0 = No change
Bit 4	Initial LED_1 Blinking State	1 = On; 0 = Off
Bit 5	Initial LED_0 Blinking State	1 = On; 0 = Off
Bit 6	LED_1 Blinking Mask	1 = Blink 0 = Not Blink
Bit 7	LED_0 Blinking Mask	1 = Blink 0 = Not Blink

Data In Blinking Duration Control.

LED_0, LED_1 Blinking Duration Control Format (4 Bytes)

Byte 0	Byte 1	Byte 2	Byte 3
T1 Duration Initial Blinking State (Unit = 100 ms)	T2 Duration Toggle Blinking State (Unit = 100 ms)	Number of repetition	Link to Buzzer

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

Where:

Byte 3 Link to Buzzer. Controls the buzzer state during the LED Blinking.

00h: The buzzer will not turn on.

01h: The buzzer will turn on during the T1 Duration.

02h: The buzzer will turn on during the T2 Duration.

03h: The buzzer will turn on during the T1 and T2 Duration.

Data Out SW1 SW2. Status Code returned by the reader.

Status Code

Results	SW1	SW2	Meaning
Success	90	Current LED State	The operation is completed successfully.
Error	63	00h	The operation is failed.

Current LED State (1 Byte)

Status	Item	Description
Bit 0	Current LED_1 LED	1 = On; 0 = Off
Bit 1	Current LED_0 LED	1 = On; 0 = Off
Bits 2 – 7	Reserved	

Notes:

1. *The LED State operation will be performed after the LED Blinking operation is completed.*
2. *The LED will not change if the corresponding LED Mask is not enabled.*
3. *The LED will not blink if the corresponding LED Blinking Mask is not enabled. Also, the number of repetition must be greater than zero.*
4. *T1 and T2 duration parameters are used for controlling the duty cycle of LED blinking and Buzzer Turn-On duration.*
5. *For example, if T1=1 and T2=1, the duty cycle = 50%. Duty Cycle = T1 / (T1 + T2).*
6. *To control the buzzer only, set the P2 "LED State Control" to zero.*
7. *The make the buzzer operate, the "number of repetition" must greater than zero.*
8. *To control the LED only, set the parameter "Link to Buzzer" to zero.*

Example 1: To read the existing LED State.

// Assume both LED_0 and LED_1 are OFF initially //

// Not linked to the buzzer //

APDU = "FF 00 40 00 04 00 00 00 00"

Response = "90 00". LED_0 and LED_1 LEDs are OFF.

Example 2: To turn on LED_0 and LED_1.

// Assume both LED_0 and LED_1 are OFF initially //

// Not linked to the buzzer //

APDU = "FF 00 40 0F 04 00 00 00 00"

Response = "90 03". LED_0 and LED_1 are ON,

To turn off both LED_0 and LED_1, APDU = "FF 00 40 0C 04 00 00 00 00"

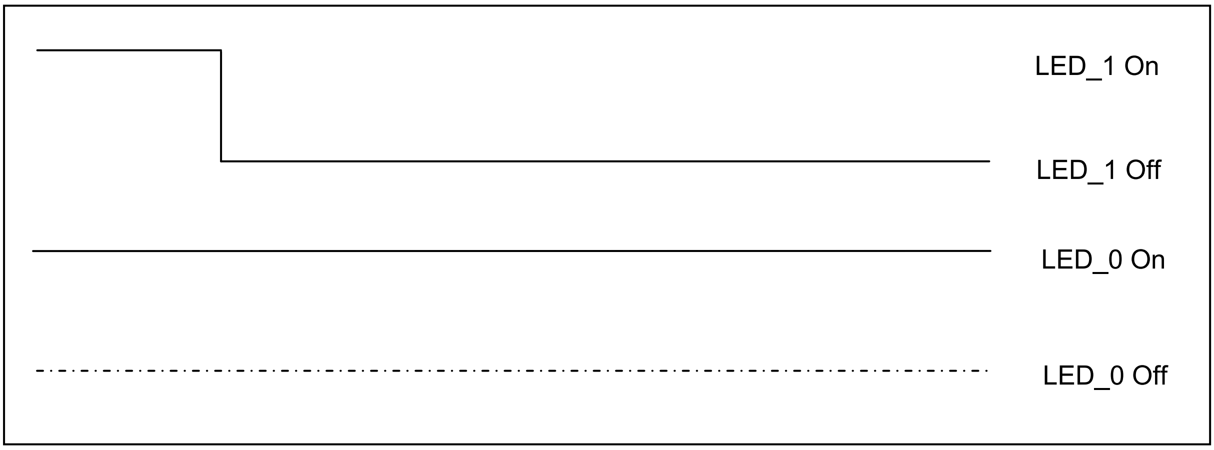
Example 3: To turn off the LED_1 only, and leave the LED_0 unchanged.

// Assume both LED_0 and LED_1 are ON initially //

// Not linked to the buzzer //

APDU = "FF 00 40 04 04 00 00 00 00"

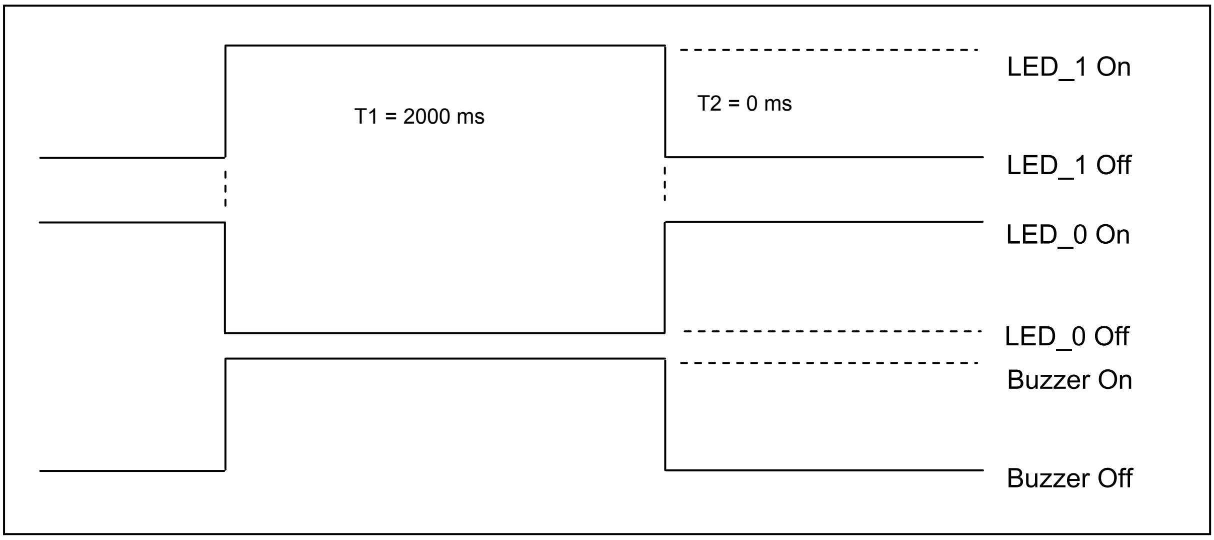
Response = "90 02". LED_0 is not changed (ON); LED_1 is OFF.



Example 4: To turn on the LED_1 for 2 sec. After that, resume to the initial state.

// Assume the LED_1 is initially OFF, while the LED_0 is initially ON. //

// The LED_1 and buzzer will turn on during the T1 duration, while the LED_0 will turn off during the T1 duration. //



1 Hz = 1000 ms Time Interval = 500 ms ON + 500 ms OFF

T1 Duration = 2000 ms = 14h

T2 Duration = 0 ms = 00h

Number of repetition = 01h

Link to Buzzer = 01h

APDU = “FF 00 40 50 04 14 00 01 01”

Response = “90 02”

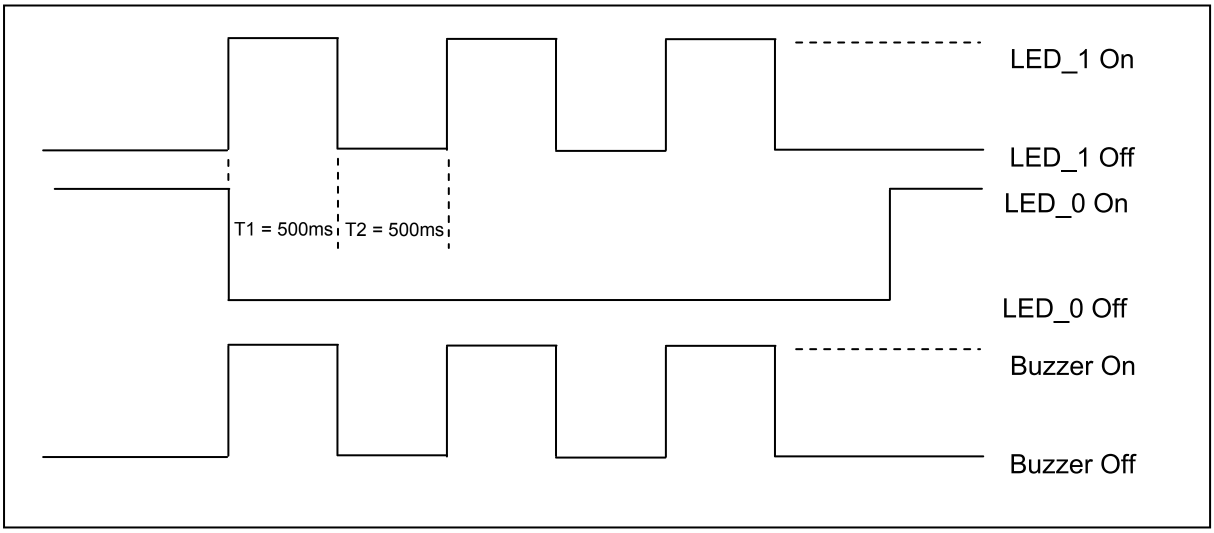
Example 5: To make LED_1 blink of 1 Hz for 3 times. After that, resume to initial state.

// Assume the LED_1 is initially OFF, while the LED_0 is initially ON. //

// The Initial LED_1 Blinking State is ON. Only the LED_1 will be blinking.

// The buzzer will turn on during the T1 duration, while the LED_0 will turn off during both the T1 and T2 duration.

// After the blinking, the LED_0 will turn ON. The LED_1 will resume to the initial state after the blinking //



1 Hz = 1000 ms Time Interval = 500 ms ON + 500 ms OFF

T1 Duration = 500 ms = 05h

T2 Duration = 500 ms = 05h

Number of repetition = 03h

Link to Buzzer = 01h

APDU = “FF 00 40 50 04 05 05 03 01”

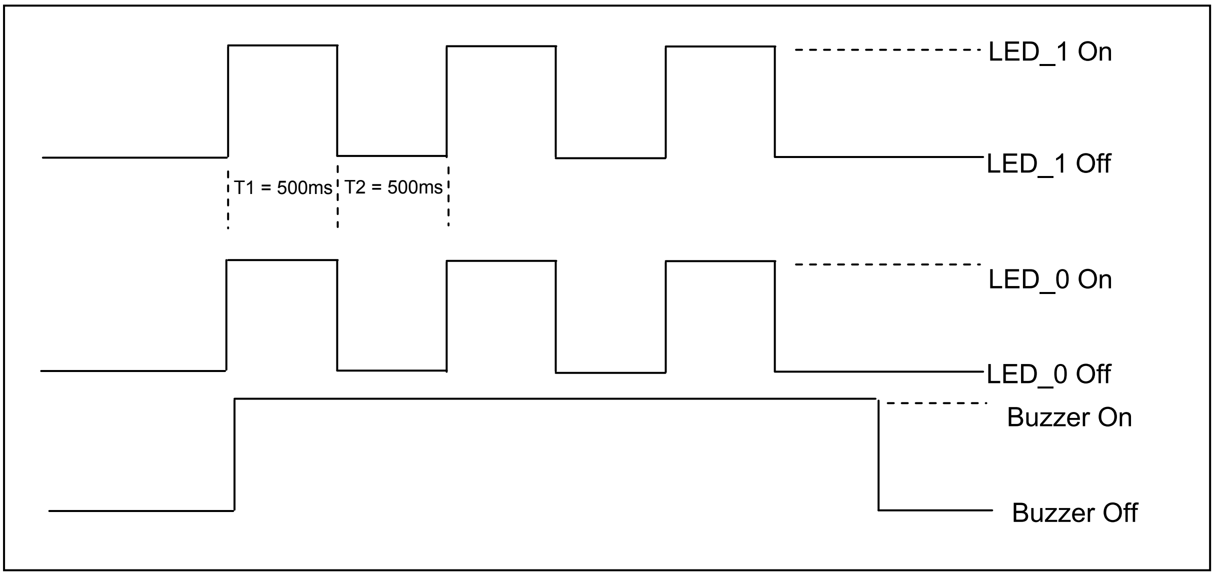
Response = “90 02”

Example 6: To make LED_1 and LED_0 blink of 1 Hz for 3 times.

// Assume both the LED_0 and LED_1 are initially OFF. //

// Both Initial LED_0 and LED_1 Blinking States are ON //

// The buzzer will turn on during both the T1 and T2 duration//



1 Hz = 1000 ms Time Interval = 500 ms ON + 500 ms OFF

T1 Duration = 500 ms = 05

T2 Duration = 500 ms = 05

Number of repetition = 03

Link to Buzzer = 03

APDU = "FF 00 40 F0 04 05 05 03 03"

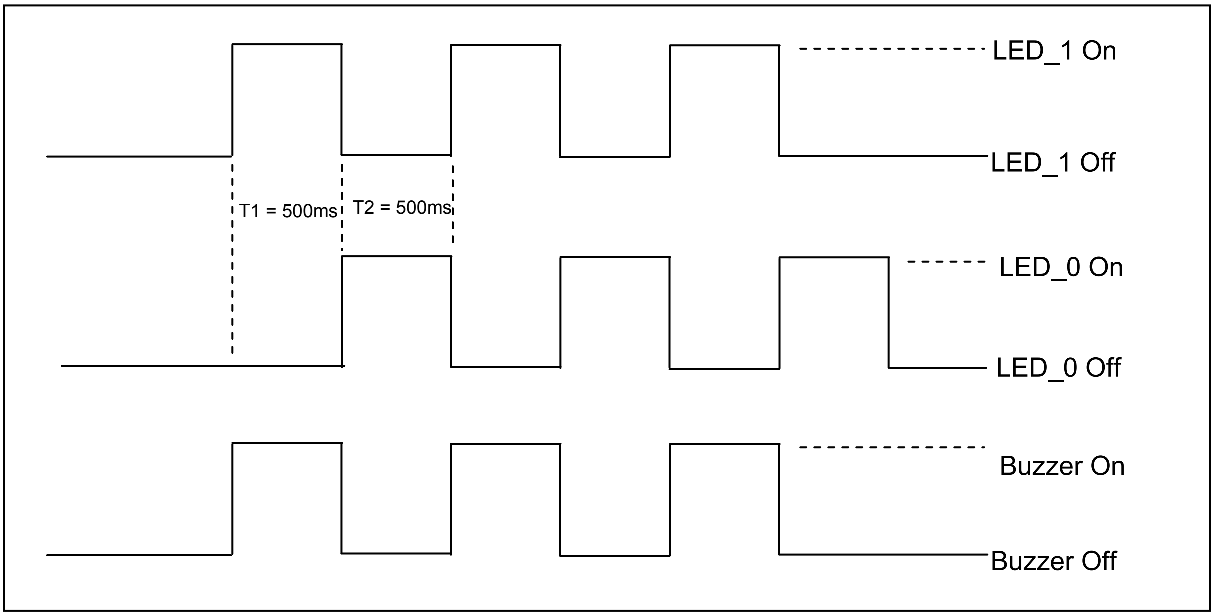
Response = "90 00"

Example 7: To make LED_1 and LED_0 blink in turn of 1 Hz for 3 times.

// Assume both LED_0 and LED_1 LEDs are initially OFF. //

// The Initial LED_1 Blinking State is ON; The Initial LED_0 Blinking States is OFF //

// The buzzer will turn on during the T1 duration//



1 Hz = 1000 ms Time Interval = 500 ms ON + 500 ms OFF

T1 Duration = 500 ms = 05h

T2 Duration = 500 ms = 05h

Number of repetition = 03h

Link to Buzzer = 01h

APDU = "FF 00 40 D0 04 05 05 03 01"

Response = "90 00"

6.16. Buzzer Control

This command is used to control the buzzer.

Buzzer Control Command Format (5 Bytes)

Command	Class	INS	P1	P2	Lc	Data In (3 Bytes)
Buzzer Control	FFh	00h	42h	00h	03h	Buzzer Control

Data In Buzzer Control.

Buzzer On/Off Duration Control Format (4 Bytes)

Byte 0	Byte 1	Byte 2
T1 Duration On State (Unit = 100 ms)	T2 Duration Off State (Unit = 100 ms)	Number of repetition

Data Out SW1 SW2.

Status Code

Results	SW1	SW2	Meaning
Success	90	00h	The operation is completed successfully.
Error	63	00h	The operation is failed.

6.17. Basic program flow for ISO 14443-4 Type A and B tags

Typical sequence may be:

1. Scan the tags in the field (Polling) with the correct parameter (Type A or B).
2. Change the Baud Rate (optional for Type A tags only).
3. Perform any T=CL command.
4. Deselect the tag.

Step 1. Polling for the ISO 14443-4 Type A Tag, 106 kbps

HOST -> 02 6F 09 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock Format)

HOST -> FF 00 00 00 04 D4 4A 01 00 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 00 08 28 04 85 82 2F A0 07 77 F7 80 02 47 65 90 00 [Checksum] 03

In which, Number of Tag found = [01]; Target number = 01

SENS_RES = 00 08; SEL_RES = 28,

Length of the UID = 4; UID = 85 82 2F A0

ATS = 07 77 F7 80 02 47 65

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

Operation Finished = 90 00

OR

Step 2. Polling for the ISO14443-4 Type B Tag, 106 kbps

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock Format)

HOST -> FF 00 00 00 05 D4 4A 01 03 00 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 14 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21 90 00 [Checksum] 03

In which, Number of Tag found = [01]; Target number = 01

ATQB = 50 00 01 32 F4 00 00 00 00 33 81 81

ATTRIB_RES Length = 01; ATTRIB_RES = 21

Operation Finished = 90 00

Step 3. Change the default Baud Rate to other Baud Rate (optional).

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock Format)

HOST -> FF 00 00 00 05 D4 4E 01 02 02 [Checksum] 03 // Change to Baud Rate 424 kbps

OR

HOST -> FF 00 00 00 05 D4 4E 01 01 01 [Checksum] 03 // Change to Baud Rate 212 kbps

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 4F [00] 90 00 [Checksum] 03

Note: *Please check the maximum baud rate supported by the tags. Only Type A tags are supported.*

Step 3. Perform T=CL command, Get Challenge APDU = 00 84 00 00 08.

HOST -> 02 6F 0D 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock Format)

HOST -> FF 00 00 00 08 D4 40 01 00 84 00 00 08 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 0F 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 62 89 99 ED C0 57 69 2B 90 00 90 00 [Checksum] 03

In which, Response Data = 62 89 99 ED C0 57 69 2B 90 00

Step 4. Deselect the Tag.

HOST -> 02 6F 08 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock Format)

HOST -> FF 00 00 00 03 D4 44 01 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [Checksum] 03

Step 5. Turn off the Antenna Power (optional).

HOST -> 02 6F 09 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock Format)

HOST -> FF 00 00 00 04 D4 32 01 00

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 04 00 00 00 00 01 01 00 00

RDR -> D5 33 90 00 [Checksum] 03

Note: *Please refer to the Tag specification for more detailed information.*

6.18. Basic program flow for Mifare applications

Typical sequence may be:

1. Scanning the tags in the field (Polling).
2. Authentication.
3. Read/Write the memory of the tag.
4. Halt the tag (optional).

Step 1. **Polling** for the MIFARE 1K/4K Tags, 106 kbps

<< 02 6F 09 00 00 00 00 01 00 00 00

FF 00 00 00 04 D4 4A 01 00 [Checksum] 03

>> 02 00 00 03

>> 02 80 0E 00 00 00 00 01 01 00 00

D5 4B 01 01 00 02 18 04 F6 8E 2A 99 90 00 [Checksum] 03

In which, Number of Tag found = [01]; Target number = 01

SENS_RES = 00 02; SEL_RES = 18,

Length of the UID = 4; UID = F6 8E 2A 99

Operation Finished = 90 00

Note: *The tag type can be determined by recognizing the SEL_RES.*

SEL_RES of some common tag types.

00 = Mifare Ultralight

08 = Mifare 1K

09 = Mifare Mini

18 = Mifare 4K

20 = Mifare DESFire

28 = JCOP30

98 = Gemplus MPCOS

Step 2. KEY A Authentication, Block **04**, KEY = FF FF FF FF FF FF, UID = F6 8E 2A 99.

<< 02 6F 14 00 00 00 00 00 01 00 00 00

FF 00 00 00 0F D4 40 01 **60** **04** FF FF FF FF FF FF F6 8E 2A 99 [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 01 00 00

D5 41 [00] 90 00 [Checksum] 03

Note: *If the authentication failed, the error code [XX] will be returned.*

[00] = Valid, other = Error. Please refer to Error Codes Table for more details.

For KEY B Authentication:

<< 02 6F 14 00 00 00 00 00 01 00 00 00

FF 00 00 00 0F D4 40 01 **61** **04** FF FF FF FF FF FF F6 8E 2A 99 [Checksum] 03

Step 3. Read the content of Block **04**.

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **30** **04** [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 01 00 00

D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [Checksum] 03

In which, Block Data = 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16

Step 4. Update the content of Block **04**.

<< 02 6F 1A 00 00 00 00 01 00 00 00

FF 00 00 00 15 D4 40 01 **A0** **04** 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 00 00 00

D5 41 [00] 90 00 [Checksum] 03

Step 5. Halt the tag (optional).

<< 02 6F 08 00 00 00 00 01 00 00 00

FF 00 00 00 03 D4 44 01 [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 01 00 00

D5 45 [00] 90 00 [Checksum] 03

6.18.1. Handling the value blocks of Mifare 1K/4K tag

The value blocks are used for performing electronic purse functions, e.g. Increment, Decrement, Restore, Transfer, etc. The value blocks have a fixed data format which permits error detection and correction and a backup management.

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

Where:

Value A signed 4-Byte value. The lowest significant byte off a value is stored in the lowest address byte. Negative values are stored in standard 2's complement format.

Adr 1-Byte address, which can be used to save the storage address of a block. (Optional)

Example:

Value 100 (decimal) = 64 (Hex), assume Block = 05h

The formatted value block = 64 00 00 00 9B FF FF FF 64 00 00 00 05 FA 05 FA

Step 1. Update the content of Block **05 with a value 100 (dec)**.

<< 02 6F 1A 00 00 00 00 01 00 00 00

FF 00 00 00 15 D4 40 01 **A0** **05** 64 00 00 00 9B FF FF FF 64 00 00 00 05

FA 05 FA [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 00 00 00

D5 41 [00] 90 00 [Checksum] 03

Step 2. Increment the value of Block **05 by 1 (dec)**.

<< 02 6F 0E 00 00 00 00 01 00 00 00

FF 00 00 00 09 D4 40 01 **C1** **05** 01 00 00 00 [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 00 00 00

D5 41 [00] 90 00 [Checksum] 03

Note: *Decrement the value of Block **05 by 1 (dec)**.*

<< 02 6F 0E 00 00 00 00 01 00 00 00

FF 00 00 00 09 D4 40 01 **C0** **05** 01 00 00 00 [Checksum] 03

Step 3. Transfer the prior calculated value of Block **05 (dec)**.

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **B0** **05** [Checksum] 03

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

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 00 00 00

D5 41 [00] 90 00 [Checksum] 03

Note: *Restore the value of Block **05** (cancel the prior Increment or Decrement operation).*

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **C2** **05** [Checksum] 03

Step 4. Read the content of Block **05**.

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **30** **05** [Checksum] 03

>> 02 00 00 03

>> 02 80 15 00 00 00 00 01 00 00 00

D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00 [Checksum] 03

In which, the value = 101 (dec)

Step 5. Copy the value of Block **05** to Block **06** **(dec)**.

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **C2** **05** [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 00 00 00

D5 41 [00] 90 00 [Checksum] 03

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **B0** **06** [Checksum] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 00 00 00

D5 41 [00] 90 00 [Checksum] 03

Step 6. Read the content of Block **06**.

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **30** **06** [Checksum] 03

>> 02 00 00 03

>> 02 80 15 00 00 00 00 01 00 00 00

D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00 [Checksum] 03

In which, the value = 101 (dec). The Adr "05 FA 05 FA" tells us the value is copied from Block 05.

Note: *Please refer to the MIFARE specification for more detailed information.*

6.18.2. Accessing Mifare Ultralight tags

Typical sequence may be:

1. Scanning the tags in the field (Polling).
2. Read/Write the memory of the tag.
3. Halt the tag (optional).

Step 1. Polling for the MIFARE Ultralight Tags, 106 kbps.

HOST -> 02 6F 09 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 04 D4 4A 01 00 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 11 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 00 44 00 07 04 6E 0C A1 BF 02 84 90 00 [Checksum] 03

In which, Number of Tag found = [01]; Target number = 01

SENS_RES = 00 44; SEL_RES = 00,

Length of the UID = 7; UID = 04 6E 0C A1 BF 02 84

Operation Finished = 90 00

Step 2. Read the content of Page **04**.

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 05 D4 40 01 **30** **04** [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [Checksum] 03

In which, Block Data = 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16

Note: *4 consecutive Pages will be retrieved. Pages 4, 5, 6 and 7 will be retrieved. Each data page consists of 4 bytes.*

Step 3. Update the content of Page **04 with** the data “AA BB CC DD”.

HOST -> 02 6F 0E 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 09 D4 40 01 **A2** **04** AA BB CC DD [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [Checksum] 03

OR

Step 3. Write (Mifare compatible Write) the content of Page **04 with** the data "AA BB CC DD"

HOST -> 02 6F 1A 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 15 D4 40 01 **A0** **04** AA BB CC DD 00 00 00 00 00 00 00 00 00 00 00 00 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [Checksum] 03

Note: *This command is implemented to accommodate the established Mifare 1K/4K infrastructure. We have to assemble the data into a 16 bytes frame. The first 4 bytes are for data, the rest of the bytes (12 ZEROS) are for padding. Only the page 4 (4 bytes) is updated even through 16 byte are sent to the reader.*

Step 4. Read the content of Page **04** again.

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 05 D4 40 01 **30** **04** [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [Checksum] 03

In which, Block Data = AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16

Note: *Only the page 4 is updated. Pages 5, 6 and 7 remain the same.*

Step 5. Halt the tag (optional).

HOST -> 02 6F 08 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 03 D4 44 01 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 45 [00] 90 00 [Checksum] 03

Note: *Please refer to the Mifare Ultralight specification for more detailed information.*

Byte Number	0	1	2	3	Page
Serial Number	SN0	SN1	SN2	BCC0	0
Serial Number	SN3	SN4	SN5	SN6	1
Internal / Lock	BCC1	Internal	Lock0	Lock1	2
OTP	OPT0	OPT1	OTP2	OTP3	3
Data read/write	Data0	Data1	Data2	Data3	4
Data read/write	Data4	Data5	Data6	Data7	5
Data read/write	Data8	Data9	Data10	Data11	6
Data read/write	Data12	Data13	Data14	Data15	7
Data read/write	Data16	Data17	Data18	Data19	8
Data read/write	Data20	Data21	Data22	Data23	9
Data read/write	Data24	Data25	Data26	Data27	10
Data read/write	Data28	Data29	Data30	Data31	11
Data read/write	Data32	Data33	Data34	Data35	12
Data read/write	Data36	Data37	Data38	Data39	13
Data read/write	Data40	Data41	Data42	Data43	14
Data read/write	Data44	Data45	Data46	Data47	15

512 bits or 64 Bytes

Table 7: Mifare Ultralight Memory Map

6.18.3. Accessing Mifare Ultralight C tags

Typical sequence may be:

1. Scanning the tags in the field (Polling).
2. Authentication.
3. Read/Write the memory of the tag.
4. Halt the tag (optional).

Step 1. Polling for the MIFARE Ultralight C Tags, 106 kbps

HOST -> 02 6F 09 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 04 D4 4A 01 00 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 11 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 00 44 00 07 04 6E 0C A1 BF 02 84 90 00 [Checksum] 03

In which, Number of Tag found = [01]; Target number = 01

SENS_RES = 00 44; SEL_RES = 00,

Length of the UID = 7; UID = 04 6E 0C A1 BF 02 84

Operation Finished = 90 00

Step 2. 3DES Authentication.

HOST -> 02 6F 09 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 04 D4 42 1A 00 10 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 0E 00 00 00 00 01 01 00 00

RDR -> D5 43 [00] 04 77 64 89 99 74 24 67 90 00 [Checksum] 03

In which, 3DES challenge from the card = [04 77 64 89 99 74 24 67];

Operation Finished = 90 00

HOST -> 02 6F 18 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 13 D4 42 AF 88 68 45 07 65 86 99 67 00 53 77 56 98 65 49 67 [Checksum] 03

In which, 3DES reply to the card = [88 68 45 07 65 86 99 67 00 53 77 56 98 65 49 67];

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 0E 00 00 00 00 01 01 00 00

RDR -> D5 43 [00] 00 06 78 53 80 68 89 61 24 90 00 [Checksum] 03

In which, 3DES reply from the card = [06 78 53 80 68 89 61 24];

Operation Finished = 90 00

Note: *The 3DES reply from the card should be checked to make sure the card is legitimate.*

Step 3. Read the content of Page **04**.

HOST -> 02 6F 09 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 05 D4 40 01 **30** **04** [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [Checksum] 03

In which, Block Data = 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16

Note: *4 consecutive Pages will be retrieved. Pages 4, 5, 6 and 7 will be retrieved. Each data page consists of 4 bytes.*

Step 4. Update the content of Page **04 with** the data "AA BB CC DD".

HOST -> 02 6F 0E 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 09 D4 40 01 **A2** **04** AA BB CC DD [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [Checksum] 03

OR

Step 4. Write (Mifare compatible Write) the content of Page **04 with** the data "AA BB CC DD".

HOST -> 02 6F 1A 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 15 D4 40 01 **A0** **04** AA BB CC DD 00 00 00 00 00 00 00 00 00 00 00 00 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [Checksum] 03

Note: *This command is implemented to accommodate the established Mifare 1K/4K infrastructure. We have to assemble the data into a 16-byte frame. The first 4 bytes are for data, the rest of the bytes (12 ZEROS) are for padding. Only the page 4 (4 bytes) is updated even through 16 byte are sent to the reader.*

Step 5. Read the content of Page **04 again**.

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 05 D4 40 01 **30** **04** [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [Checksum] 03

In which, Block Data = AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16

Note: *Only page 4 is updated. Pages 5, 6 and 7 remain the same.*

Step 6. Halt the tag (optional).

HOST -> 02 6F 08 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 03 D4 44 01 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 45 [00] 90 00 [Checksum] 03

Note: *Please refer to the Mifare Ultralight C specifications for more detailed information.*

Byte Number	0	1	2	3	Page
Serial Number	SN0	SN1	SN2	BCC0	0
Serial Number	SN3	SN4	SN5	SN6	1
Internal/Lock	BCC1	Internal	Lock	Lock	2
OTP	OTP0	OTP1	OTP2	OTP3	3
Data read/write	Data0	Data1	Data2	Data3	4
Data read/write	Data4	Data5	Data6	Data7	5

Byte Number	0	1	2	3	Page
Data read/write	Data8	Data9	Data10	Data11	6
Data read/write	Data12	Data13	Data14	Data15	7
Data read/write	Data16	Data17	Data18	Data19	8
Data read/write	Data20	Data21	Data22	Data23	9
Data read/write	Data24	Data25	Data26	Data27	10
Data read/write	Data28	Data29	Data30	Data31	11
Data read/write	Data32	Data33	Data34	Data35	12
Data read/write	Data36	Data37	Data38	Data39	13
Data read/write	Data40	Data41	Data42	Data43	14
Data read/write	Data44	Data45	Data46	Data47	15
Data read/write	Data48	Data49	Data50	Data51	16
Data read/write	Data52	Data53	Data54	Data55	17
Data read/write	Data56	Data57	Data58	Data59	18
Data read/write	Data60	Data61	Data62	Data63	19
Data read/write	Data64	Data65	Data66	Data67	20
Data read/write	Data68	Data69	Data70	Data71	21
Data read/write	Data72	Data73	Data74	Data75	22
Data read/write	Data76	Data77	Data78	Data79	23
Data read/write	Data80	Data81	Data82	Data83	24
Data read/write	Data84	Data85	Data86	Data87	25
Data read/write	Data88	Data89	Data90	Data91	26
Data read/write	Data92	Data93	Data94	Data95	27
Data read/write	Data96	Data97	Data98	Data99	28
Data read/write	Data100	Data101	Data102	Data103	29
Data read/write	Data104	Data105	Data106	Data107	30
Data read/write	Data108	Data109	Data110	Data111	31
Data read/write	Data112	Data113	Data114	Data115	32
Data read/write	Data116	Data117	Data118	Data119	33
Data read/write	Data120	Data121	Data122	Data123	34
Data read/write	Data124	Data125	Data126	Data127	35
Data read/write	Data128	Data129	Data130	Data131	36
Data read/write	Data132	Data133	Data134	Data135	37
Data read/write	Data136	Data137	Data138	Data139	38
Data read/write	Data140	Data141	Data142	Data143	39
Lock	Lock	Lock	-	-	40
16 bit counter	16 bit counter	16 bit counter	-	-	41

Byte Number	0	1	2	3	Page
Authentication configuration	Authentication configuration	Authentication configuration	Authentication configuration	Authentication configuration	42
Authentication configuration	Authentication configuration	Authentication configuration	Authentication configuration	Authentication configuration	43
Authentication key	Authentication key	Authentication key	Authentication key	Authentication key	44
Authentication key	Authentication key	Authentication key	Authentication key	Authentication key	45
Authentication key	Authentication key	Authentication key	Authentication key	Authentication key	46
Authentication key	Authentication key	Authentication key	Authentication key	Authentication key	47

Table 8: Mifare Ultralight C Memory Map

Total Page Size: 792 bits of 198 Bytes.

6.19. Basic program flow for FeliCa applications

Step 0. Start the application. The first thing is to activate the “SAM Interface”. The ATR of the SAM (if a SAM is inserted) or a Pseudo-ATR “3B 00” (if no SAM is inserted) will be returned. In other words, the SAM always exists from the view of the application.

Step 1. The second thing to do is to change the operating parameters of the PN531. Set the Retry Time to one.

Step 2. Poll a FeliCa Tag by sending “Direct Transmit” and “Get Response” APDUs (Tag Polling).

Step 3. If no tag is found, go back to Step 2 until a FeliCa Tag is found.

Step 4. Access the FeliCa Tag by sending APDUs (Tag Read or Write)

Step 5. If there is no any operation with the FeliCa Tag, then go back to Step 2 to poll the other FeliCa Tag.

..

Step N. Deactivate the “SAM Interface”. Shut down the application.

Notes:

1. *The default Retry Time of the Tag command “InListPassiveTarget” is infinity. Send the APDU “FF 00 00 00 06 D4 32 05 00 00 00” to change the Retry Time to one.*
2. *It is recommended to turn off the Antenna if there is no contactless access.*

   APDU for turning on the Antenna Power = APDU “FF 00 00 00 04 D4 32 01 03”

   APDU for turning off the Antenna Power = APDU “FF 00 00 00 04 D4 32 01 02”

6.20. Basic program flow for NFC Forum Type 1 tag applications

Example: Jewel and Topaz tags

Typical sequence may be:

1. Scanning the tags in the field (Polling)
2. Read/Update the memory of the tag
3. Deselect the tag

Step 1. Polling for the Jewel or Topaz Tag, 106 kbps.

HOST -> 02 6F 09 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock Format)

HOST -> FF 00 00 00 04 D4 4A 01 04 [Checksum] 03

RDR -> 02 00 00 03 (Waiting the Tag)

RDR -> 02 80 0C 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 0C 00 B5 3E 21 00 90 00 [Checksum] 03

In which, Number of Tag found = [01]; Target number = 01

ATQA_RES = 0C 00; UID = B5 3E 21 00

Operation Finished = 90 00

Step 2. Read the memory address 08 (Block 1: Byte-0).

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00 FF 00 00 00 05 D4 40 01 01 08 [Checksum] 03

RDR -> 02 00 00 03 02 80 06 00 00 00 00 01 01 00 00 D5 41 [00] 18 90 00 [Checksum] 03

In which, Response Data = 18

Note: *To read all the memory content of the tag starting from the memory address 00*.

HOST -> 02 6F 09 00 00 00 00 01 00 00 00 FF 00 00 00 04 D4 40 01 00 [Checksum] 03

RDR -> 02 00 00 03 02 80 7F 00 00 00 00 01 01 00 00 D5 41 00 11 48

RDR -> show all data … 90 00 [Checksum] 03

Step 3. Update the memory address 08 (Block 1: Byte-0) with the data FF.

HOST -> 2 6F 0B 00 00 00 00 01 00 00 00 FF 00 00 00 06 D4 40 01 53 08 FF [Checksum] 03

RDR -> 02 00 00 03 02 80 05 00 00 00 00 01 01 00 00 D5 41 [00] FF 90 00 [Checksum] 03

In which, Response Data = FF

Note: *To update more than one memory content of the tag starting form the memory address 08 (Block 1: Byte-0).*

HOST -> 02 6F 0D 00 00 00 00 01 00 00 00 FF 00 00 00 08 D4 40 01 58 08 02 AA BB [Checksum]

03

RDR -> 02 00 00 03 02 80 06 00 00 00 00 01 01 00 00 D5 41 [00] 90 00 [Checksum] 03

In which, Command = 58; Starting memory address = 08;
Number of write content = 02; Memory content = AA, BB;

Step 4. Deselect the tag.

HOST -> 02 6F 08 00 00 00 00 01 00 00 00 FF 00 00 00 03 D4 44 01 [Checksum] 03

RDR -> 02 00 00 03 02 80 05 00 00 00 00 01 01 00 00 D5 45 [00] 90 00 [Checksum] 03

Appendix A. ACR122 Error Codes

Error Code	Error
00h	No error.
01h	Time Out, the target has not answered.
02h	A CRC error has been detected by the contactless UART.
03h	A Parity error has been detected by the contactless UART.
04h	During a Mifare anti-collision/select operation, an erroneous Bit Count has been detected.
05h	Framing error during Mifare operation.
06h	An abnormal bit-collision has been detected during bit wise anti-collision at 106 kbps.
07h	Communication buffer size insufficient.
08h	RF Buffer overflow has been detected by the contactless UART (bit BufferOvfl of the register CL_ERROR).
0Ah	In active communication mode, the RF field has not been switched on in time by the counterpart (as defined in NFCIP-1 standard).
0Bh	RF Protocol error (cf. reference [4], description of the CL_ERROR register).
0Dh	Temperature error: the internal temperature sensor has detected overheating, and therefore has automatically switched off the antenna drivers.
0Eh	Internal buffer overflow
10h	Invalid parameter (range, format, etc.)
12h	DEP Protocol: The chip configured in target mode does not support the command received from the initiator (the command received is not one of the following: ATR_REQ, WUP_REQ, PSL_REQ, DEP_REQ, DSL_REQ, RLS_REQ, ref. [1]).
13h	DEP Protocol/Mifare/ISO/IEC 14443-4: The data format does not match to the specification. Depending on the RF protocol used, it can be: • Bad length of RF received frame, • Incorrect value of PCB or PFB, • Invalid or unexpected RF received frame, • NAD or DID incoherence.
14h	Mifare: Authentication error.
23h	ISO/IEC 14443-3: UID Check byte is wrong.
25h	DEP Protocol: Invalid device state, the system is in a state which does not allow the operation.
26h	Operation not allowed in this configuration (host controller interface).
27h	This command is not acceptable due to the current context of the chip (Initiator vs. Target, unknown target number, Target not in the good state, etc.).
29h	The chip configured as target has been released by its initiator.
2Ah	ISO/IEC 14443-3B only: the ID of the card does not match, meaning that the expected card has been exchanged with another one.
2Bh	ISO/IEC 14443-3B only: the card previously activated has disappeared.

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

Error Code	Error
2Ch	Mismatch between the NFCID3 initiator and the NFCID3 target in DEP 212/424 kbps passive.
2Dh	An over-current event has been detected.
2Eh	NAD missing in DEP frame.