Advanced Card Systems Ltd.
Card & Reader Technologies

ACR122U-SAM
USB NFC Reader

Application Programming Interface V2.01

Table of Contents

List of Figures

List of Tables

1.0. Introduction

The ACR122U-SAM is a PC-linked Contactless Smart Card Reader/Writer used for accessing ISO 14443-4 Type A and Type B, Mifare, ISO 18092 or NFC, and FeliCa tags. Since different contactless cards have different communication protocols and commands, the ACR122U-SAM serves as the intermediary device between the personal computer and the contactless tag.

The reader is connected to the computer through a USB interface and accepts commands from the computer. When the computer issues a command, the reader will identify whether the command will be used to communicate with a contactless tag or with the device peripherals (LED or buzzer). Then, the requested data or status information will be returned.

1.1. Features

- USB 2.0 Full Speed Interface
- CCID Compliance
- 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)
    - ISO 7816 compliant SAM slot
- Application Programming Interface:
    - Supports PC/SC for SAM interface only
    - Supports CT-API (through wrapper on top of PC/SC) for SAM interface only
- Built-in Peripheral
    - User-controllable bi-color LED
- Supports Android™ OS 3.1 and above
- Compliant with the following standards:
    - ISO 14443
    - CE
    - FCC
    - VCCI
    - PC/SC
    - CCID
    - Microsoft WHQL
    - RoHS

1.2. USB Interface

The ACR122U-SAM is connected to a computer through USB as specified in the USB Specification 1.1. The ACR122U-SAM is working in full-speed mode, i.e. 12 Mbps.

Pin	Signal	Function
1	V_{BUS}	+5 V power supply for the reader (max. 200 mA, normal 100 mA)
2	D-	Differential signal transmits data between ACR122U-SAM and PC
3	D+	Differential signal transmits data between ACR122U-SAM and PC
4	GND	Reference voltage level for power supply

Table 1: USB Interface

2.0. Implementation

2.1. Communication Flowchart of ACR122U-SAM

The Standard Microsoft CCID and PC/SC drivers are used for ACR122U-SAM. The CCID USB drivers are already built inside the Windows operating system or can be automatically downloaded from the internet via Windows Update.

The Contactless Cards, Buzzer and Bi-color LED interfaces are accessed through the use of Pseudo-APDUs. The Pseudo-APDUs will be discussed in more detail in Sections 4, 5 and 6.

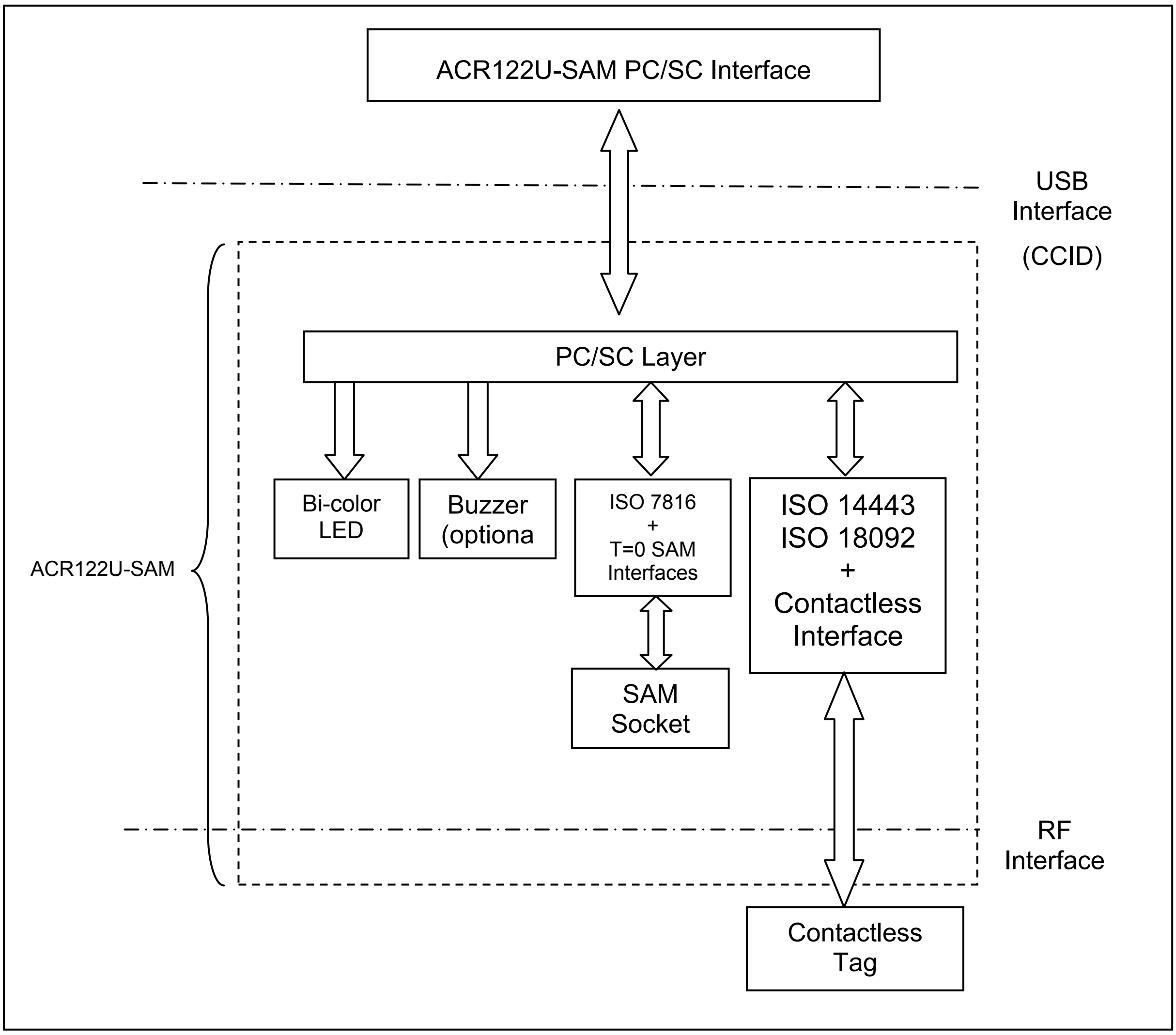


Figure 1: Communication Flowchart of ACR122U-SAM

3.0. Pseudo-APDU Commands

The PC/SC interface is used for exchanging APDUs and responses between the PC and tag. The ACR122U-SAM will handle the required protocol internally and comes with two primitive commands for this purpose, the Direct Transmit and Get Response commands.

If the reader finds that the APDU is in the form of “FF 00 00 00 Lc XX XXh ..” or “FF C0 00 00h Le,” the APDU will be routed to the contactless interface. Similarly, if the reader finds that the APDU is in the form of “FF 00 40 XX 04 XX XX XX XXh,” the APDU will be used for setting the LED and Buzzer State. The contact interface must be activated in order to send commands to the contactless or LED interface.

3.1. Direct Transmit

This is used to send an APDU (Contactless Interface and Contactless Commands) and the length of the Response Data that will be returned.

3.1.1. Direct Transmit Command

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

Command	Class	INS	P1	P2	Lc	Data In
Direct Transmit	0xFFh	0x00h	0x00h	0x00h	Number of Bytes to send	Contactless Command

Where:

Lc: 1 Byte. Number of Bytes to Send.

Maximum 255 bytes

Data In: Contactless Command

The data to be sent to the Contactless Interface and Contactless Tag.

3.1.2. Direct Transmit Response

Direct Transmit Response Format (2 Bytes)

Response	Data Out	
Result	SW1	SW2

Where:

Data Out: SW1 SW2

Status Code returned by the reader.

Status Code

Results	SW1 SW2	Meaning
Success	61 LEN	The operation is completed successfully. The response data has a length of LEN bytes. The APDU “Get Response” should be used to retrieve the response data.
Error	63 00h	The operation is failed.

Results	SW1 SW2	Meaning
Time Out Error	63 01h	The Contactless Interface does not respond.
Checksum Error	63 27h	The checksum of the Contactless Response is wrong.
Parameter Error	63 7Fh	The Contactless Command is wrong.

3.2. Get Response

This command is used to retrieve the response data after the "**Direct Transmit**" command is issued.

3.2.1. Get Response Command

Get Response Command Format (5 Bytes)

Command	Class	INS	P1	P2	Le
Get Response	0xFFh	0xC0h	0x00h	0x00h	Number of Bytes to retrieve

Where:

Le: 1 Byte. Number of Bytes to Retrieve.
Maximum 255 bytes

3.2.2. Get Response Result

Get Response Result Format (Le bytes, Length of the Response Data)

Response	Data Out
Result	Response Data

Where:

Data Out: Response Data
Error Code "63 00h" will be given if no response data is available

***Note**: In general, the Pseudo APDUs "**Direct Transmit**" and "**Get Response**" are used in pairs. Once the APDU "Direct Transmit" is sent, the reader will return the length of the response data. Then, the APDU "Get Response" is immediately used to retrieve the actual response data.*

3.3. Bi-color LED and Buzzer Control

This APDU is used to control the states of the Bi-color LED and Buzzer.

Bi-color LED and Buzzer Control Command Format (9 Bytes)

Command	Class	INS	P1	P2	Lc	Data In (4 Bytes)
Bi-color LED and Buzzer Control	0xFFh	0x00h	0x40h	LED State Control	0x04h	Blinking Duration Control

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

3.3.1. LED State Control (P2)

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

Table 2: Bi-color LED and Buzzer Control Format (1 Byte)

3.3.2. Blinking Duration Control (Data In)

Bi-color LED 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

Where:

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

0x00h: The buzzer will not turn on

0x01h: The buzzer will turn on during the T1 Duration

0x02h: The buzzer will turn on during the T2 Duration

0x03h: The buzzer will turn on during the T1 and T2 Duration.

3.3.3. Status Code Returned by the Reader (Data Out)

The data out, SW1 SW2 is the 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.

Status	Item	Description
Bit 0	Current Red LED	1 = On; 0 = Off
Bit 1	Current Green LED	1 = On; 0 = Off
Bits 2 – 7	Reserved	-

Table 3: Current LED State (1 Byte)

Notes:

A. *The LED State operation will be performed after the LED Blinking operation is completed.*

B. *The LED will not change if the corresponding LED Mask is not enabled.*

C. *The LED will not blink if the corresponding LED Blinking Mask is not enabled. Furthermore, the number of repetition should be greater than zero.*

D. *T1 and T2 duration parameters are used for controlling the duty cycle of LED blinking and Buzzer Turn-On duration. For example, if T1=1 and T2=1, the duty cycle = 50%.*

   Note: *Duty Cycle = T1/(T1 + T2).*

E. *To control the buzzer only, just set the P2 "LED State Control" to zero.*

F. *The make the buzzer operating, the "number of repetition" must greater than zero.*

G. *To control the LED only, set the parameter "Link to Buzzer" to zero.*

Example 1: To read the existing LED State.

Assume both Red and Green LEDs are OFF initially and not linked to the buzzer.

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

Response = "90 00h." RED and Green LEDs are OFF.

Example 2: To turn on Red and Green Color LEDs.

Assume both Red and Green LEDs are OFF initially and not linked to the buzzer.

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

Response = "90 03h." RED and Green LEDs are ON.

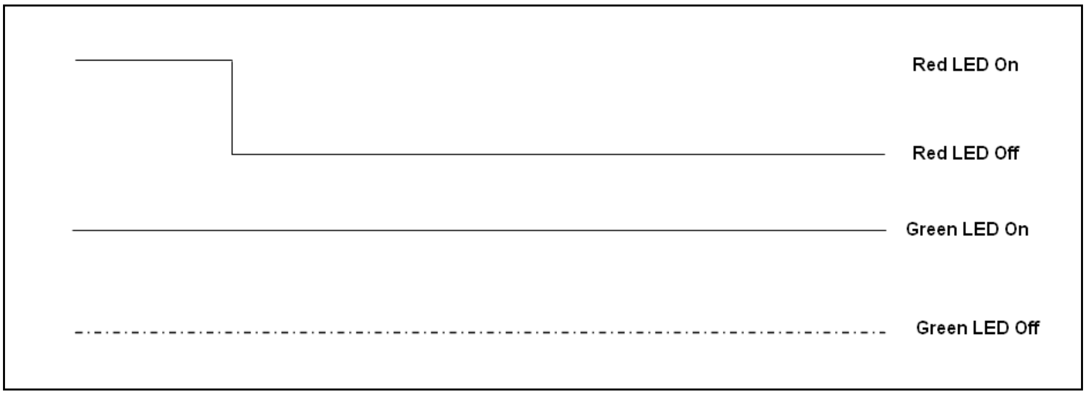
Note: *To turn off both RED and Green LEDs, APDU = "FF 00 40 0C 04 00 00 00 00h"*

Example 3: To turn off the Red LED and leave the Green LED unchanged.

Assume both Red and Green LEDs are ON initially and not linked to the buzzer.

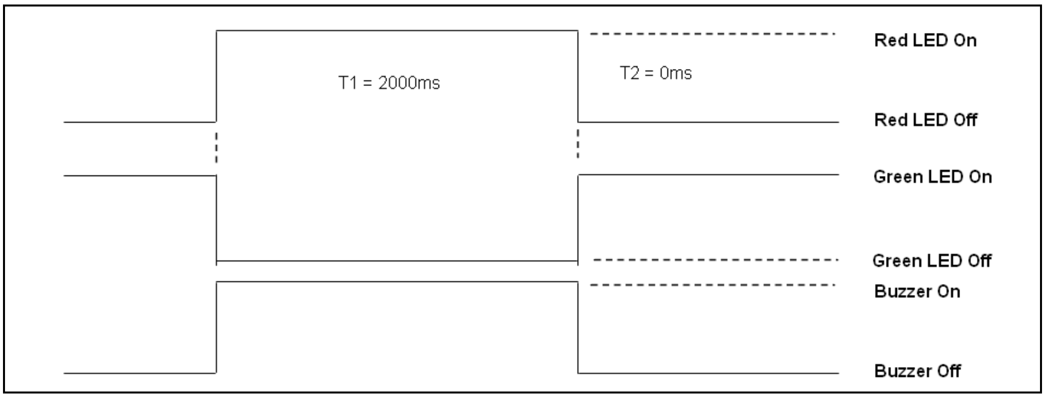
APDU = "FF 00 40 04 04 00 00 00 00h"

Response = "90 02h." Green LED is not changed (ON); Red LED is OFF.



Example 4: To turn on the Red LED for 2 seconds then resume to the initial state.

Assume the Red LED is initially OFF, while the Green LED is initially ON. The Red LED and buzzer will turn on during the T1 duration, while the Green LED will turn off during the T1 duration.



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

T1 Duration = 2000 ms = 0x14h

T2 Duration = 0 ms = 0x00h

Number of repetition = 0x01h

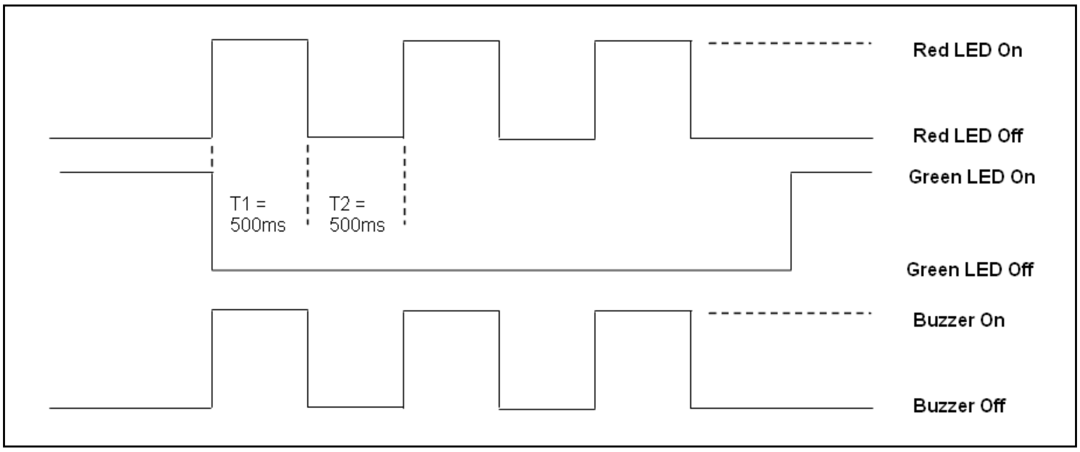
Link to Buzzer = 0x01h

APDU = “FF 00 40 50 04 14 00 01 01h”

Response = “90 02h”

Example 5: To blink the Red LED of 1 Hz for 3 times then resume to initial state.

Assumptions: Assume the Red LED is initially OFF, while the Green LED is initially ON. The Initial Red LED Blinking State is ON. Only the Red LED will be blinking. The buzzer will turn on during the T1 duration, while the Green LED will turn off during both the T1 and T2 duration. After the blinking, the Green LED will turn ON. The Red LED will resume its initial state after the blinking.



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

T1 Duration = 500ms = 0x05h

T2 Duration = 500ms = 0x05h

Number of repetition = 0x03h

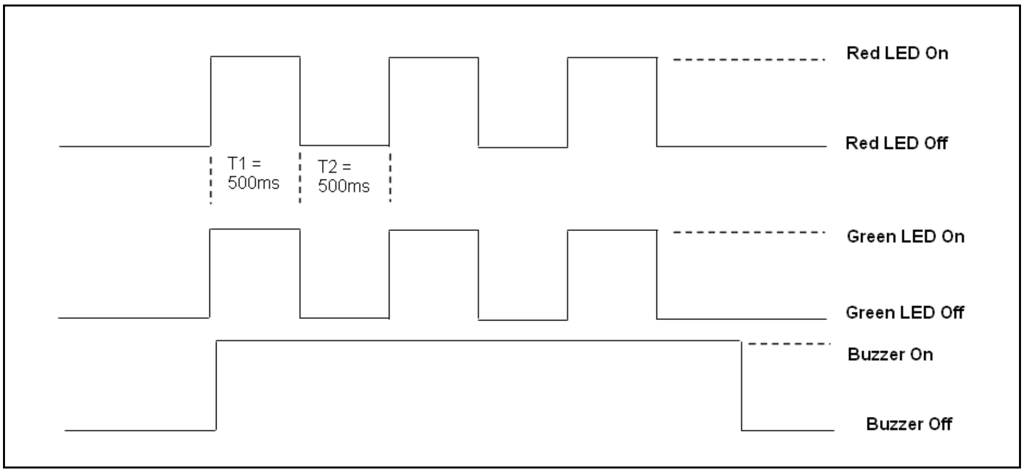
Link to Buzzer = 0x01h

APDU = "FF 00 40 50 04 05 05 03 01h"

Response = "90 02h"

Example 6: To blink the Red and Green LEDs of 1 Hz for 3 times.

Assumptions: Assume both the Red and Green LEDs are initially OFF. Both Initial Red and Green Blinking States are ON. The buzzer will turn on during both the T1 and T2 duration.



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

T1 Duration = 500 ms = 0x05h

T2 Duration = 500 ms = 0x05h

Number of repetition = 0x03h

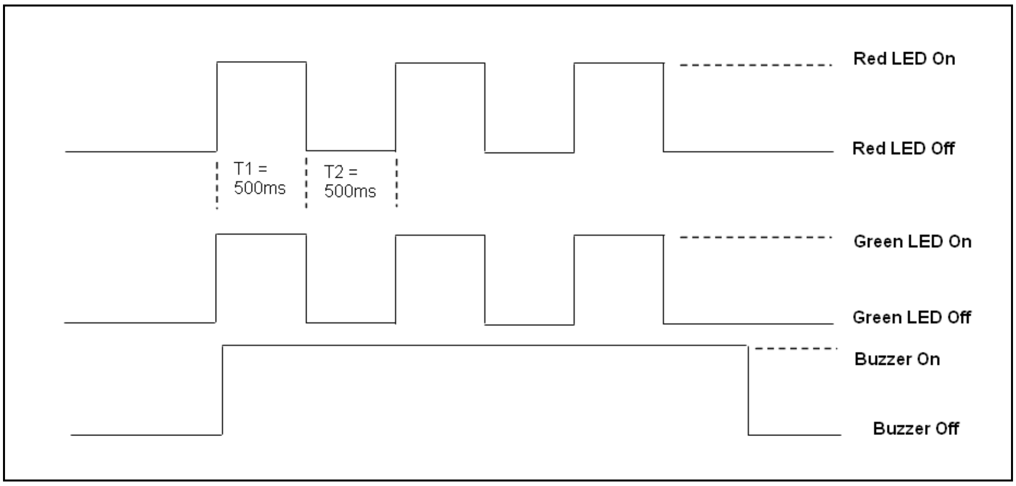
Link to Buzzer = 0x03h

APDU = “FF 00 40 F0 04 05 05 03 03h”

Response = “90 00h”

Example 7: To blink the Red and Green LED in turn with a rate of 1 Hz for 3 times.

Assumptions: Assume both Red and Green LEDs are initially OFF. The Initial Red Blinking State is ON, the Initial Green Blinking States is OFF. The buzzer will turn on during the T1 duration.



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

T1 Duration = 500 ms = 0x05h

T2 Duration = 500 ms = 0x05h

Number of repetition = 0x03h

Link to Buzzer = 0x01h

APDU = “FF 00 40 D0 04 05 05 03 01h”

Response = “90 00h”

3.4. Firmware Version of the Reader

This pseudo-APDU command is used to retrieve the firmware version of the reader.

Get Firmware Version Command Format (5 Bytes)

Command	Class	INS	P1	P2	Le
Get Response	0xFFh	0x00h	0x48h	0x00h	0x0Ah

Get Firmware Version Response Format (10 bytes)

Response	Data Out
Result	Firmware Version

E.g. Response = 41 43 52 31 32 32 55 31 30 31h (Hex) = ACR122U-SAM101 (ASCII)

4.0. Polling for More Than One Tag Type/Automatic Polling

Typical polling parameters are:

- Number of Polling **(PollNr)**
  0x01h to 0xFEh : 1 up to 254 polling
  0xFFh : Endless polling
- Polling Period **(Period)**
  0x01h to 0x0Fh : Polling period in units of 150 ms
- The mandatory Tag Type to be detected **(Type 1)**
- The [optional tag types] to be detected **(Type 2 .. Type N)**

Possible Tag Type Value

0x04h : Innovision Topaz or Jewel tag,
0x10h : Mifare card,
0x11h : FeliCa 212 kbps card,
0x12h : FeliCa 424 kbps card,
0x20h : Passive 106 kbps ISO/IEC14443-4A,
0x23h : Passive 106 kbps ISO/IEC14443-4B,

APDU Input Format

Pseudo APDU "Direct Transmit"					Polling Command (5 <= LEN)							
FFh	00h	00h	00h	LEN (The length of the Polling Command)	D4h	60h	PollNr	Period	Type 1	[Type 2]	…	[Type N]

APDU Output Format

D5	**61**	**NbTg (Number of Tags found)**	**[Tag 1]**	**[Ln 1]**	**[Tag 1 Info]**
			[Tag 2]	**[Ln 2]**	**[Tag 2 Info]**
			…		
			[Tag N]	**[Ln N]**	**[Tag N Info]**
			Status Code		

[Tag 1 … N]: The tag type detected by the reader. If two tag types are detected, [Tag 1] is the first tag type detected by the reader, while [Tag 2] is the second tag type detected by the reader.

[Ln 1 … N]: The length of the [Tag Info] field of the corresponding Tag.

[Tag 1 … N Info]: The Tag Information. E.g. Tag Serial Number, etc.

Status Code: The result of the operation.

4.1. Detecting Contactless Tags

4.1.1. Issue Polling Command

<< FF 00 00 00 09 D4 60 **01** **01** **20 23 11 04 10h**

Where:

01h = Number of polling

01h = Polling Period

20 23 11 04 10h = Tag types to be detected in ascending order.

***Note**: ISO 14443-4 Type A will be the first tag type to be detected while Mifare will be the last tag type to be detected.*

***Case 1:** Assume a Mifare 4K tag is found.*

>> 61 10h (at least one tag is found)

<< FF C0 00 00 10h

>> D5 61 **01** **10** **09** **01** 00 02 18 04 F6 8E 2A 99 90 00h

Where:

01h = One Tag is found

10h = Tag Type (Mifare)

09h = The Tag Info has 9 bytes in length

01h = Target number

00 02h = SENS_RES

18h = SEL_RES (Mifare 4K)

04h = Length of the UID

F6 8E 2A 99h = UID

90 00h = Operation Finished

***Case 2:** Assume a Mifare Ultralight tag is found.*

>> 61 13h (at least one tag is found)

<< FF C0 00 00 13h

>> D5 61 **01** **10** **0C** 01 00 44 00 07 04 6E 0C A1 BF 02 84 90 00h

Where:

01h = One Tag is found

10h = Tag Type (Mifare)

0Ch = The Tag Info has C bytes in length

01h = Target number

00 44h = SENS_RES

00h = SEL_RES (Ultralight)

07h = Length of the UID;

04 6E 0C A1 BF 02 84h = UID

90 00h = Operation Finished

Case 3: *Assume an ISO 14443-4 Type A tag is found.*

>> 61 1Ah (at least one tag is found)

<< FF C0 00 00 1Ah

>> D5 61 **01** **20** **13** 01 03 04 28 04 E0 7B 9E A9 0A 38 77 B1 4A 43 4F 50 33 30 90 00h

Where:

01h = One Tag is found

20h = Tag Type (ISO14443-4 Type A)

13h = The Tag Info has 13 bytes (hex) in length

01h = Target number

03 04h = SENS_RES

28h = SEL_RES

04h = Length of the UID

E0 7B 9E A9h = UID

0A 38 77 B1 4A 43 4F 50 33 30h = ATS

90 00h = Operation Finished

Case 4: *Assume an ISO 14443-4 Type B tag is found.*

>> 61 16h (at least one tag is found)

<< FF C0 00 00 16h

>> D5 61 **01** **23** **0F** 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21 90 00h

Where:

01h = One Tag is found

23h = Tag Type = ISO14443-4 Type B

0Fh = The Tag Info has 0F bytes (hex) in length

01h = Target number

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

01h = ATTRIB_RES Length

21h = ATTRIB_RES

90 00h = Operation Finished

Case 5: *Assume a FeliCa 212K tag is found.*

>> 61 1Ah (at least one tag is found)

<< FF C0 00 00 1Ah

>> D5 61 **01** 11 13 01 12 01 01 01 05 01 86 04 02 02 03 01 4B 02 4F 49 93 FF 90 00h

Where:

01h = One Tag is found

11h = Tag Type (FeliCa 212K)

13h = The Tag Info has 13 bytes (hex) in length

01h = Target number

12h = POL_RES Length

01h = Response Code

01 01 05 01 86 04 02 02h = NFCID2

03 01 4B 02 4F 49 93 FFh = PAD

90 00h = Operation Finished

Case 6: *Assume a Topaz tag is found.*

>> 61 0Eh (at least one tag is found)

<< FF C0 00 00 0Eh

>> D5 61 **01** 04 07 01 0C 00 18 26 21 00 90 00h

Where:

01h = One Tag is found

04h = Tag Type (Topaz)

07h = The Tag Info has 7 bytes (hex) in length

0C 00h = ATQA_RES

18 26 21 00h = UID

90 00h = Operation Finished

Case 7: *Assume two Mifare tags are found.*

>> 61 1Bh (at least one tag is found)

<< FF C0 00 00 1Bh

>> D5 61 **02** **10** **09** 01 00 04 08 04 76 85 3F E1

10 **09** 02 00 04 08 04 9A FE 10 3C

90 00h

Where:

02h = Two Tags are found and,

10h = Tag 1 Type (Mifare)

09h = The Tag 1 Info has 9 bytes in length

01h = Target number

00 04h = SENS_RES

08h = SEL_RES (Mifare 1K)

04h = Length of the UID

76 85 3F E1h = UID

and

10h = Tag 1 Type = Mifare

09h = The Tag 1 Info has 9 bytes in length

02h = Target number

00 04h = SENS_RES

08h (Mifare 1K) = SEL_RES

04h = Length of the UID

9A FE 10 3Ch =UID

90 00h = Operation Finished

Case 8: *No Tag Found.*

>> 61 05h

<< FF C0 00 00 05h

>> D5 61 **00** 90 00h

Where:

00h = No Tag found

90 00h = Operation Finished

4.1.2. Performing Operations with the Detected Tag Using the Target Number

Example: One Mifare 1K and one Mifare 4K tags are detected and assigned Target Numbers 01h and 02h.

Select the Mifare 1K tag **(Target Number = 01h)** and Read its memory block **04h** *(Assume the tag is already authenticated).*

Read the content of Block **04h**.

<< FF 00 00 00 05 D4 40 **01 30 04h**

>> 61 15h

<< FF C0 00 00 15h

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

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

Select the Mifare 4K tag (Target Number = **02h**) and Read its memory block 05 (Assume the tag is already authenticated).

Read the content of Block **05h**.

<< FF 00 00 00 05 D4 40 **02 30 05h**

>> 61 15h

<< FF C0 00 00 15h

>> D5 41 [00] 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 90 00h

In which, Block Data = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00h

5.0. Basic Program Flow for Contactless Applications

The contactless interface is operating on top of contact interface. If the device finds that the APDU is for the contactless interface, the command will be routed to this interface, otherwise the APDU will be routed to the contact interface. Furthermore, the contact and contactless interface can be operating at the same time.

The basic program flow for an ACR122U-SAM application is:

Step 0: Start the application by connecting to the “ACR122U-SAM PC/SC Interface.” The ATR of the SAM (if a SAM is inserted) or a Pseudo-ATR “3B 00h” (if no SAM is inserted) will be returned. This means that the SAM always exists from the view of the application.

Step 1: Afterwards, the user needs to change the operating parameters of the contactless interface (PN531). Set the Retry Time to one.

Step 2: Poll a contactless tag by using “**Direct Transmit”** and “**Get Response”** APDUs (Tag Polling).

Step 3: If no tag is found, go back to Step 2 until a contactless tag is found. For ISO 14443-4 tags polling, there might be a need to turn off the Antenna Field first then turn it on again before starting another polling process.

Step 4: Access the contactless tag by sending Pseudo APDUs – “**Direct Transmit”** and “**Get Response.”** These commands will be discussed in detail in the later sections.

Step 5: If the operation is finished or there is no operation anymore with the contactless tag, go back to Step 2 to poll for another contactless tag.

….

Step N: Disconnect the “ACR122U-SAM PC/SC Interface.” Shut down the application.

Notes:

1. *Some Type A tags may support both ISO 14443-3 Type A and ISO 14443-4 Type A operating modes. For example, JCOP30 supports Mifare 1K emulation (ISO 14443-3) and ISO 14443-4. If the reader sends a RATS command to the tag, the ISO 14443-4 mode will be activated, or the tag can remain in Mifare 1K emulation mode (ISO 14443-3). It is up to the application to decide which operating mode must be activated. By default, the reader will perform ISO 14443-4 activation automatically if the tag supports ISO 14443-4.*
    - *To disable automatic ISO 14443-4 activation: FF 00 00 00 03 D4 12 24h*
    - *To enable automatic ISO 14443-4 activation: FF 00 00 00 03 D4 12 34h*

2. *The default Retry Time of the contactless interface command “InListPassiveTarget” is infinity. This means that after the polling command is sent out, the reader will wait until a valid tag is found. If the application wants to get an immediate result of the polling command, please set the Retry Time to one.*
    - *Set the Retry Time to one: FF 00 00 00 06 D4 32 05 00 00 00h*

3. *The antenna can be switched off in order to save the power.*
    - *Turn off the antenna power: FF 00 00 00 04 D4 32 01 00h*
    - *Turn on the antenna power: FF 00 00 00 04 D4 32 01 01h*

4. *No Automatic Contactless Tag Insertion or Removal Event will be generated. The Contactless Polling is done by the application.*

5. *The contactless tag is accessed through the use of Pseudo-APDUs "**Direct Transmit**" and "**Get Response**." You can refer to the succeeding sections for more details.*

   The reader will check the content of the APDU to determine which interface will be used which is illustrated below:

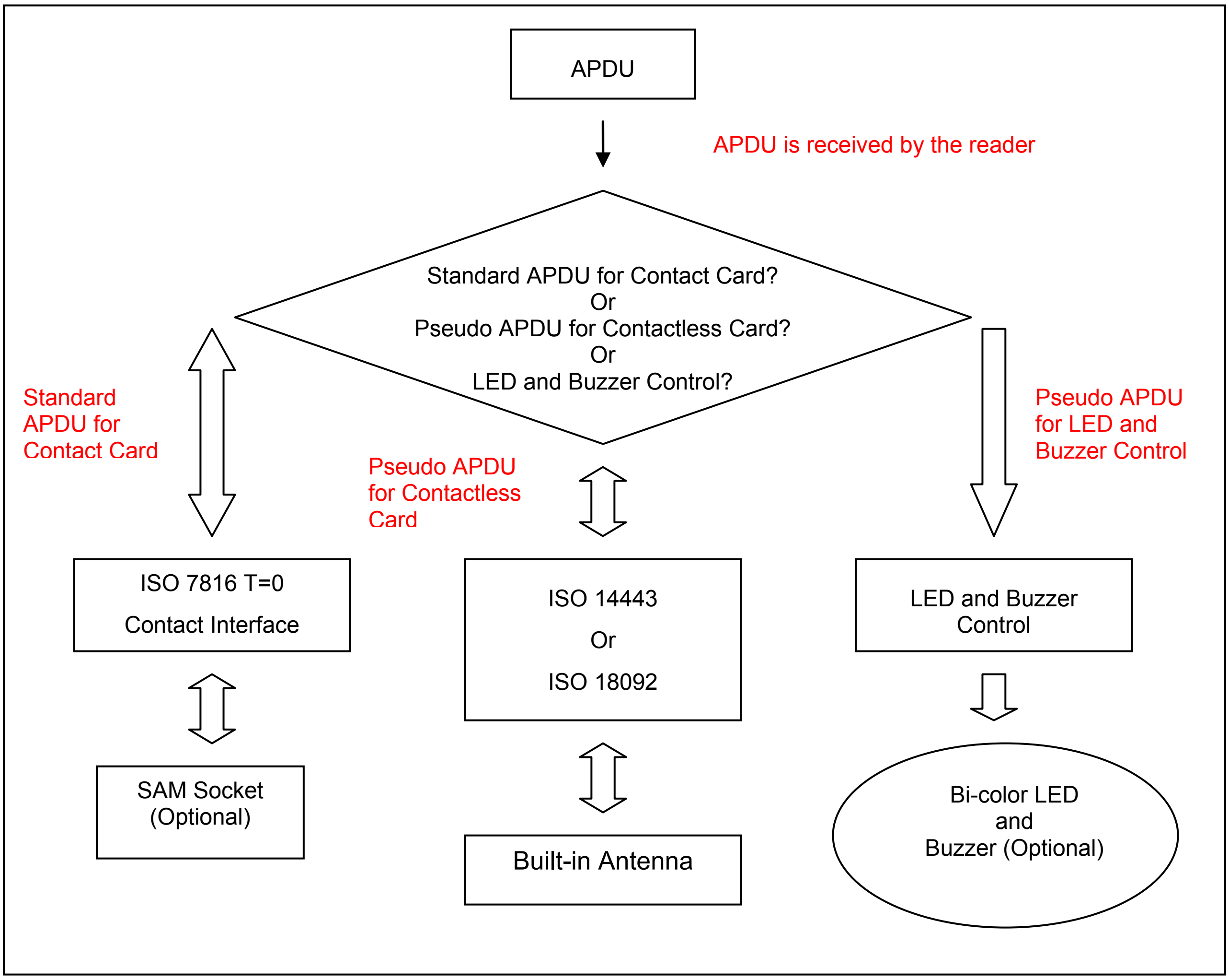


Figure 2: APDU Commands and the ACR122U-SAM Interface and Peripherals

6. *For the contactless interface, because of the limitation of ISO 7816 T=0 protocol (Standard Microsoft CCID drivers), it is not possible to send both "Lc" and "Le" in a single APDU. Therefore, the APDU is split into two separate APDUs. First, the APDU "**Direct Transmit**" is sent to get the length of the response data, then, the APDU "**Get Response**" is sent to retrieve the response data.*
   - *PC → Reader: Issue a Pseudo APDU "Direct Transmit" to the reader.*
   - *Reader → PC: The length of the response data is returned.*
   - *PC → Reader: Issue a Pseudo APDU "Get Response" to get the response data.*
   - *Reader → PC: The response data is returned.*

5.1. Accessing Mifare Classic Tags

The typical sequence is:

- Scanning the tags in the field (Polling)

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

- Authentication
- Read/Write the memory of the tag
- Halt the tag (optional)

The steps shown below provide an example for Mifare Classic Tags:

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

<< FF 00 00 00 04 D4 4A 01 00h

>> 61 0Eh (a tag is found) (Status Code)

<< FF C0 00 00 0Eh

>> D5 4B 01 01 00 02 18 04 F6 8E 2A 99 90 00h

Where:

Number of Tag found = [01h]

Target number = 01h

SENS_RES = 00 02h

SEL_RES = 18h

Length of the UID = 04h

UID = F6 8E 2A 99h

Operation Finished = 90 00h

***Tip:** If no tag is found, the following response will be returned.*

>> 61 05h (no tag found)

<< FF C0 00 00 05h

>> D5 4B 00 90 00h

Where:

00h is the Error Code. See **Appendix A** for more details.

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

SEL_RES of some common tag types.

00h = Mifare Ultralight

08h = Mifare 1K

09h = Mifare MINI

18h = Mifare 4K

20h = Mifare DESFire

28h = JCOP30

98h = Gemplus MPCOS

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

<< FF 00 00 00 0F D4 40 01 60 **04** FF FF FF FF FF FF F6 8E 2A 99h

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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

[00h] = Valid, other = Error. Please refer to Error Codes Table for more details. (Appendix A)

Tip: For KEY B Authentication

<< FF 00 00 00 0F D4 40 01 61 **04** FF FF FF FF FF FF F6 8E 2A 99h

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

<< FF 00 00 00 05 D4 40 01 30 **04h**

>> 61 15h

<< FF C0 00 00 15h

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

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

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

<< 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 10h

>> 61 05h (Status Code)

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

Step 5) **Halt the tag** (optional).

<< FF 00 00 00 03 D4 44 01h

>> 61 05h (Status Code)

<< FF C0 00 00 05h

>> D5 45 [00] 90 00h

Sectors (Total 16 sectors. Each sector consists of 4 consecutive blocks)	**Data Blocks (3 blocks, 16 bytes per block)**	**Trailer Block (1 block, 16 bytes)**	
Sector 0	0x00 ~ 0x02h	0x03h	1K Bytes
Sector 1	0x04 ~ 0x06h	0x07h	
..			
..			
Sector 14	0x38 ~ 0x0Ah	0x3Bh	
Sector 15	0x3C ~ 0x3Eh	0x3Fh	

Table 4: Mifare 1K Memory Map

Sectors (Total 32 sectors. Each sector consists of 4 consecutive blocks)	Data Blocks (3 blocks, 16 bytes per block)	Trailer Block (1 block, 16 bytes)
Sector 0	0x00 ~ 0x02h	0x03h
Sector 1	0x04 ~ 0x06h	0x07h
..		
..		
Sector 30	0x78 ~ 0x7Ah	0x7Bh
Sector 31	0x7C ~ 0x7Eh	0x7Fh

2K Bytes

Sectors (Total 8 sectors. Each sector consists of 16 consecutive blocks)	Data Blocks (15 blocks, 16 bytes per block)	Trailer Block (1 block, 16 bytes)
Sector 32	0x80 ~ 0x8Eh	0x8Fh
Sector 33	0x90 ~ 0x9Eh	0x9Fh
..		
..		
Sector 38	0xE0 ~ 0xEEh	0xEFh
Sector 39	0xF0 ~ 0xFEh	0xFFh

2K Bytes

Table 5: Mifare 4K Memory Map

Tip: *Once the authentication is done, all data blocks of the same sector are free to access. For example, once the data block 0x04h is successfully authenticated (Sector 1), the data blocks 0x04h ~ 0x07h are free to access.*

5.1.1. Accessing Mifare 7-byte UID Classic Tags

The typical sequence may be:

- Scanning the tags in the field (Polling)
- Authentication
- Read/Write the memory of the tag
- Halt the tag (optional)

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

<< FF 00 00 00 04 D4 4A 01 00h

>> 61 11h (a tag is found)

<< FF C0 00 00 11h

>> D5 4B 01 01 00 44 08 07 04 01 02 03 04 05 06 90 00h

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

SENS_RES = 00 44h; SEL_RES = 08h,

Length of the UID = 07h; UID = 04 01 02 03 04 05 06h

Operation Finished = 90 00h

Tip: *If no tag is found, the following response will be returned.*

>> 61 05h (no tag found)

<< FF C0 00 00 05h

>> D5 4B 00 90 00h

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

SEL_RES of some common tag types.

00h = Mifare Ultralight

08h = Mifare 1K

09h = Mifare MINI

18h = Mifare 4K

20h = Mifare DESFire

28h = JCOP30

98h = Gemplus MPCOS

Step 2) **KEY A Authentication,** Block **04h**, KEY = FF FF FF FF FF FFh, UID = 03 04 05 06h (just extract the last 4 bytes of the UID)

<< FF 00 00 00 0F D4 40 01 **60** **04** FF FF FF FF FF FF 03 04 05 06h

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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

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

Tip: *For **KEY B Authentication**.*

<< FF 00 00 00 0F D4 40 01 **61** **04** FF FF FF FF FF FF 03 04 05 06h

5.1.2. Handling Value Blocks of Mifare 1K/4K Tags

The value blocks are used for performing electronic purse functions. E.g. Increment, Decrement, Restore and 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	$\overline{\text{Adr}}$

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).

e.g. Value 100 (decimal) = 64 (Hex), assume Block = **0x05h**

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

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

<< 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 FAh

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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

<< FF 00 00 00 09 D4 40 01 **C1** **05** 01 00 00 00h

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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

<< FF 00 00 00 09 D4 40 01 **C0** **05** 01 00 00 00h

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

<< FF 00 00 00 05 D4 40 01 **B0** **05h**

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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

<< FF 00 00 00 05 D4 40 01 **C2** **05h**

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

<< FF 00 00 00 05 D4 40 01 **30** **05h**

>> 61 15h

<< FF C0 00 00 15h

>> D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00h

In which, the value = 101 (dec)

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

<< FF 00 00 00 05 D4 40 01 **C2** **05h**

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

<< FF 00 00 00 05 D4 40 01 **B0 06h**

>> 61 05h (Status Code)

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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

<< FF 00 00 00 05 D4 40 01 **30 06h**

>> 61 15h

<< FF C0 00 00 15h

>> D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00h

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

***Note**: You can refer to the Mifare specification for more detailed information.*

5.2. Accessing Mifare Ultralight Tags

Typical sequence may be:

- Scanning the tags in the field (Polling)
- Read/Write the memory of the tag
- Halt the tag (optional)

Step 1) **Polling** for the Mifare Ultralight Tags, 106 kbps

<< FF 00 00 00 04 D4 4A 01 00h

>> 61 11h (a tag is found)

<< FF C0 00 00 11h

>> D5 4B 01 01 00 44 00 07 04 6E 0C A1 BF 02 84 90 00h

Where:

Number of Tag found = [01h]

Target number = 01h

SENS_RES = 00 44h

SEL_RES = 00h,

Length of the UID = 7h

UID = 04 6E 0C A1 BF 02 84h

Operation Finished = 90 00h

***Tip:** If no tag is found, the following response will be returned.*

>> 61 05h (no tag found)

<< FF C0 00 00 05h

>> D5 4B 00 90 00h

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

<< FF 00 00 00 05 D4 40 01 **30 04h**

>> 61 15h

<< FF C0 00 00 15h

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

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

Tip: *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 **04h** with the data "AA BB CC DDh".

<< FF 00 00 00 09 D4 40 01 **A2** **04** AA BB CC DDh

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

Or

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

<< 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 00h

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

Tip: *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 the data to be written in the page while the rest of the bytes (12 ZEROS) are for padding. Only the block 4 (4 bytes) is updated even though 16 bytes are sent to the reader.*

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

<< FF 00 00 00 05 D4 40 01 **30** **04h**

>> 61 15h

<< FF C0 00 00 15h

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

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

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

Step 5) **Halt the tag** (optional)

<< FF 00 00 00 03 D4 44 01h

>> 61 05h

<< FF C0 00 00 05h

>> D5 45 [00] 90 00h

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 6: Mifare Ultralight Memory Map

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

5.3. Accessing ISO 14443-4 Type A and B Tags

Typical sequence may be:

- Scanning the tags in the field (Polling) with the correct parameter (Type A or B)
- Change the Baud Rate (optional for Type A tags only)
- Perform any T=CL command
- Deselect the tag

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

<< FF 00 00 00 04 D4 4A 01 00h

>> 61 15h (a tag is found)

<< FF C0 00 00 15h

>> D5 4B 01 01 00 08 28 04 85 82 2F A0 07 77 F7 80 02 47 65 90 00h

Where:

Number of Tag found = [01h]

Target number = 01h

SENS_RES = 00 08h

SEL_RES = 28h,

Length of the UID = 4h

UID = 85 82 2F A0h

ATS = 07 77 F7 80 02 47 65h

Operation Finished = 90 00h

Or

Step 1) **Polling** for the ISO14443-4 Type B Tag, 106 kbps.

<< FF 00 00 00 05 D4 4A 01 03 00h

>> 61 14h (a tag is found)

<< FF C0 00 00 14h

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

Where:

Number of Tag found = [01h]

Target number = 01h

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

ATTRIB_RES Length = 01h

ATTRIB_RES = 21h

Operation Finished = 90 00h

Step 2) **Change the default Baud Rate to other Baud Rate (optional)**.

<< FF 00 00 00 05 D4 4E 01 02 02h // Change to Baud Rate 424 kbps

Or

<< FF 00 00 00 05 D4 4E 01 01 01h // Change to Baud Rate 212 kbps

>> 61 05h

<< FF C0 00 00 05h

>> D5 4F [00] 90 00h

Please check the maximum baud rate supported by the tags. Only Type A tags is supported.

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

<< FF 00 00 00 08 D4 40 01 00 84 00 00 08h

>> 61 0Fh

<< FF C0 00 00 0Fh

>> D5 41 [00] 62 89 99 ED C0 57 69 2B 90 00 90 00h

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

Step 4) **Deselect the Tag**.

<< FF 00 00 00 03 D4 44 01h

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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

<< FF 00 00 00 04 D4 32 01 00h

>> 61 04h

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

5.4. Accessing FeliCa Tags

Typical sequence may be:

- Scanning the tags in the field (Polling)
- Read/Update the memory of the tag
- Deselect the tag

Step 1) **Polling** for the FeliCa Tag, 212 kbps, Payload = 00 FF FF 00 00h

<< FF 00 00 00 09 D4 4A 01 01 00 FF FF 00 00h

>> 61 1Ah (a tag is found)

<< FF C0 00 00 0Ch

>> 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 00h

Where:

Number of Tag found = [01h]

Target number = 01h

POL_RES Length = 14h

Response Code = 01h

NFCID2 = 01 01 05 01 86 04 02 02h

PAD = 03 00 4B 02 4F 49 8A 8A 80 08h

Operation Finished = 90 00h

Tip: *For FeliCa Tag, 424 kbps*

<< FF 00 00 00 09 D4 4A 01 ***02*** *00 FF FF 00 00h.*

Step 2) **Read the memory block**.

<< FF 00 00 00 13 D4 40 01 10 06 01 01 05 01 86 04 02 02 01 09 01 01 80 00h

>> 61 22h

<< FF C0 00 00 22h

>> 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 00h

Step 3) **Deselect the Tag**.

<< FF 00 00 00 03 D4 44 01h

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

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 00h"

In which,

Direct Transmit APDU = "FF 00 00 00h"

Length of the Contactless Command = "09h"

Command (InListPassiveTarget 212Kbps) = "D4 4A 01 01h"

Contactless Command (System Code Request) = "00 FF FF 01 00h"

The APDU Response would be "61 1Ah" for *a Tag is found*, or "61 05h" for *no Tag is found*.

*Step 2: Issue a "**Get Response**" APDU.*

The APDU Command would be "FF C0 00 00 1Ah"

The APDU Response may be

"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 00h"

In which,

Response returned by the Contactless Interface =

"D5 4B 01 01 14 01 01 01 05 01 86 04 02 02 03 00 4B 02 4F 49 8A 8A 80 08h"

NFCID2t of the Contactless Tag = "01 01 05 01 86 04 02 02h"

Status Code returned by the reader = "90 00h"

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 AAh"

In which,

Direct Transmit APDU = "FF 00 00 00h"

Length of the Contactless Command = "23h"

Command (InDataExchange) = "D4 40 01h"

Contactless 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 AAh"

The APDU Response would be "61 11h."

Step 2: Issue a "Get Response" APDU.

The APDU Command would be "FF C0 00 00 11h"

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

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

In which,

Response returned by the Contactless Interface = “D5 41h”

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

Status Code returned by the reader = “90 00h”

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 00h”

In which,

Direct Transmit APDU = “FF 00 00 00h”

Length of the Contactless Command = “13h”

Command (InDataExchange) = “D4 40 01h”

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

The APDU Response would be “61 22h.”

*Step 2: Issue a “**Get Response**” APDU.*

The APDU Command would be “FF C0 00 00 22h”

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 00h”

In which,

Response returned by the Contactless Interface = “D5 41h”

Response returned by the Contactless 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 AAh”

Status Code returned by the reader = “90 00h.”

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

5.5. Accessing NFC Forum Type 1 Tags (Jewel and Topaz Tags)

- Typical sequence may be:
- Scanning the tags in the field (Polling)
- Read/Update the memory of the tag
- Deselect the tag

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

<< FF 00 00 00 04 D4 4A 01 04h

>> 61 0Ch (a tag is found)

<< FF C0 00 00 0Ch

>> D5 4B 01 01 0C 00 18 26 21 00 90 00h

Where:

Number of Tag found = [01h]

Target number = 01h

ATQA_RES = 0C 00h

UID = 18 26 21 00h

Operation Finished = 90 00h

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

<< FF 00 00 00 05 D4 40 01 **01 08h**

>> 61 06h

<< FF C0 00 00 06h

>> D5 41 [00] 18 90 00h

In which, Response Data = 18h

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

<< FF 00 00 00 06 D4 40 01 **53 08 FFh**

>> 61 06h

<< FF C0 00 00 06h

>> D5 41 [00] FF 90 00h

In which, Response Data = FFh

Step 4) **Deselect the Tag.**

<< FF 00 00 00 03 D4 44 01h

>> 61 05h

<< FF C0 00 00 05h

>> D5 41 [00] 90 00h

5.5.1. Topaz Memory Map

Memory Address = Block No * 8 + Byte No

e.g. Memory Address 08 (hex) = 1 x 8 + 0 = Block 1: Byte-0 = Data0

e.g. Memory Address 10 (hex) = 2 x 8 + 0 = Block 2: Byte-0 = Data8

HR0	HR1
11_h	xx_h

EEPROM Memory Map										
Type	Block No.	Byte-0 **(LSB)**	Byte-1	Byte-2	Byte-3	Byte-4	Byte-5	Byte-6	Byte-7 **(MSB)**	Lockable
UID	0	UID-0	UID-1	UID-2	UID-3	UID-4	UID-5	UID-6		Locked
Data	1	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7	Yes
Data	2	Data8	Data9	Data10	Data11	Data12	Data13	Data14	Data15	Yes
Data	3	Data16	Data17	Data18	Data19	Data20	Data21	Data22	Data23	Yes
Data	4	Data24	Data25	Data26	Data27	Data28	Data29	Data30	Data31	Yes
Data	5	Data32	Data33	Data34	Data35	Data36	Data37	Data38	Data39	Yes
Data	6	Data40	Data41	Data42	Data43	Data44	Data45	Data46	Data47	Yes
Data	7	Data48	Data49	Data50	Data51	Data52	Data53	Data54	Data55	Yes
Data	8	Data56	Data57	Data58	Data59	Data60	Data61	Data62	Data63	Yes
Data	9	Data64	Data65	Data66	Data67	Data68	Data69	Data70	Data71	Yes
Data	A	Data72	Data73	Data74	Data75	Data76	Data77	Data78	Data79	Yes
Data	B	Data80	Data81	Data82	Data83	Data84	Data85	Data86	Data87	Yes
Data	C	Data88	Data89	Data90	Data91	Data92	Data93	Data94	Data95	Yes
Reserved	D									
Lock/Reserved	E	LOCK-0	LOCK-1	OTP-0	OTP-1	OTP-2	OTP-3	OTP-4	OTP-5	

Reserved for internal use
User Block Lock & Status
OTP bits

Figure 3: Topaz Memory Map

***Note**: Please refer to the Jewel and Topaz specification for more detailed information.*

5.6. Obtaining the Current Setting of the Contactless Interface

Step 1) Get Status Command.

<< FF 00 00 00 02 D4 04h

>> 61 0Ch

<< FF C0 00 00 0Ch

>> D5 05 [Err] [Field] [NbTg] [Tg] [BrRx] [BrTx] [Type] 80 90 00h

Or if no tag is in the field,

>> 61 08h

<< FF C0 00 00 08h

>> D5 05 00 00 00 80 90 00h

[Err]: an error code corresponding to the latest error detected by the Contactless Interface.

[Field]: indicates if an external RF field is present and detected by the contactless interface or not, Field = 0x01h or Field = 0x00h, respectively.

[NbTg]: the number of targets currently controlled by the contactless interface The default value is 1.

[Tg]: logical number

[BrRx]: bit rate in reception

- 0x00h : 106 kbps
- 0x01h : 212 kbps
- 0x02h : 424 kbps

[BrTx]: bit rate in transmission

- 0x00h : 106 kbps
- 0x01h : 212 kbps
- 0x02h : 424 kbps

[Type]: modulation type

- 0x00h : ISO14443 or Mifare®
- 0x10h : FeliCa™
- 0x01h : Active mode
- 0x02h : Innovision Jewel® tag

***Note**: See Appendix A and Appendix B for more details on the Error Codes and Contactless Tags Command and Response.*

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

Appendix A. Error Codes

Error Code	Error
0x00h	No Error
0x01h	Time Out, the target has not answered
0x02h	A CRC error has been detected by the contactless UART
0x03h	A Parity error has been detected by the contactless UART
0x04h	During a Mifare anti-collision/select operation, an erroneous Bit Count has been detected
0x05h	Framing error during Mifare operation
0x06h	An abnormal bit-collision has been detected during bit wise anti-collision at 106 kbps
0x07h	Communication buffer size insufficient
0x08h	RF Buffer overflow has been detected by the contactless UART (bit BufferOvfl of the register CL_ERROR)
0x0Ah	In active communication mode, the RF field has not been switched on in time by the counterpart (as defined in NFCIP-1 standard)
0x0Bh	RF Protocol error (cf. reference [4], description of the CL_ERROR register)
0x0Dh	Temperature error: the internal temperature sensor has detected overheating, and therefore has automatically switched off the antenna drivers
0x0Eh	Internal buffer overflow
0x10h	Invalid parameter (range, format, …)
0x12h	DEP Protocol: The Contactless Interface 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]).
0x13h	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.
0x14h	Mifare Authentication error
0x23h	ISO/IEC 14443-3: UID Check byte is wrong
0x25h	DEP Protocol: Invalid device state, the system is in a state which does not allow the operation
0x26h	Operation not allowed in this configuration (host controller interface)
0x27h	This command is not acceptable due to the current context of the Contactless Interface (Initiator vs. Target, unknown target number, Target not in the good state, …)
0x29h	The Contactless Interface configured as target has been released by its initiator

Error Code	Error
0x2Ah	The Contactless Interface and ISO/IEC 14443-3B only: the ID of the card does not match, meaning that the expected card has been exchanged with another one.
0x2Bh	The Contactless Interface and ISO/IEC 14443-3B only: the card previously activated has disappeared.
0x2Ch	Mismatch between the NFCID3 initiator and the NFCID3 target in DEP 212/424 kbps passive.
0x2Dh	An over-current event has been detected
0x2Eh	NAD missing in DEP frame

Appendix B. Contactless Related Commands and Responses Summary

Item	Command	Response	Meaning	Reference (Page Number)
1	**D4 04**	**D5 05**	Get the Interface Status	34
2	**D4 12**	**D5 13**	Automatic ATR Generation	17
3	**D4 32**	**D5 33**	Contactless Interface Setting	17
4	**D4 40**	**D5 41**	Tag Exchange Data	19 - 34
5	**D4 44**	**D5 45**	Tag Deselect	28 - 34
6	**D4 4A**	**D5 4B**	Tag Polling	19 - 34
7	**D4 60**	**D5 61**	Auto Tag Polling	14

<< Typical Commands and Responses Flow >>

PC	**Reader**	**Tag**
Sequences	USB Interface (12 Mbps)	RF Interface (13.56 MHz)
1. The PC issues a command.	Contactless Related Command → e.g. D4 40 [Payload]	Tag-specific Command Frame → e.g. [Payload] embedded in ISO 14443 Frame or ISO 18092 Frame
2. The Reader returns a response.	Contactless Related Response ← e.g. D5 41 [Result]	Tag-specific Response Frame ← e.g. [Result] embedded in ISO 14443 Frame or ISO 18092 Frame