Advanced Card Systems Limited

Card and Reader Technologies

Application Programming Interface

ACR122 NFC Contactless Smart Card Reader

CONTENTS

Introduction

The ACR122U is a PC-linked Contactless Cards reader and writer for accessing MIFARE, ISO14443-4 Type A&B, FeliCa and NFC Tags.

Features

- USB PnP, 12Mbps
- PCSC Interface (API level only)
- CCID Standard
- Built-in Antenna for contactless tags access.
- Bi-Color LED (Rectangular Light Guide).
- Buzzer (optional)
- SAM Socket (optional)
- Compact size: 98 X 65 X 12.8mm
- Reading distance is 40~50mm (depended on the tag type)
- RoHS Compliant
- The ACR122U supports the following Tag Types:
    - MIFARE Classic. E.g. MIFARE 1K, 4K and Ultralight
    - ISO14443-4 Type A and B.
    - Sony FeliCa. 212 kbps and 424 kbps
    - NFC Forum Type 1. E.g. Topaz, Jewel.

USB Interface

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

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

Contact and Contactless Interfaces Handling

The contactless interface is operating on the top of contact interface. Some Pseudo APDUs are defined for contactless interface. If the reader finds that the APDUs are for contactless interface, the APDUs will be routed to the contactless interface, otherwise, the APDUs will be routed to contact interface. The Contact and Contactless Interfaces are able to be operating at the same time.

1. The Pseudo APDU "Direct Transmit" is used for sending commands to the contactless interface

Command	Class	INS	P1	P2	Lc	Data In
Direct Transmit	0xFF	0x00	0x00	0x00	Number of Bytes to send	PN532_Contactless Command

2. The Pseudo APDU "Get Response" is used for retrieving the responses from the contactless interface.

Command	Class	INS	P1	P2	Le
Get Response	0xFF	0xC0	0x00	0x00	Number of Bytes to retrieve

If the reader finds that the APDU is in the form of "FF 00 00 00 Lc XX XX .." or "FF C0 00 00 Le", the APDU will be routed to the contactless interface.

Also, one Pseudo APDU "Bi-Color LED and Buzzer Control" is defined for controlling the LED and Buzzer.

Command	Class	INS	P1	P2	Lc	Data In (4 Bytes)
Bi-Color and Buzzer LED Control	0xFF	0x00	0x40	LED State Control	0x04	Blinking Duration Control

Similarly, if the reader finds that the APDU is in the form of "FF 00 40 XX 04 XX XX XX XX", the APDU will be used for setting the LED and Buzzer State.

The control interface must be activated in order to send commands to the contactless or LED interface

Pseudo APDUs

PSCS interface is used for exchanging APDUs and Responses between the PC and Tag. The ACR122U will handle the required protocol internally. ACR122U comes with two primitive commands for this purpose.

Direct Transmit
To send an APDU (PN532 and Contactless Commands), and the length of the Response Data will be returned.

Table 1.0A: Direct Transmit Command Format (Length of the PN532_Contactless Command + 5 Bytes)

Command	Class	INS	P1	P2	Lc	Data In
Direct Transmit	0xFF	0x00	0x00	0x00	Number of Bytes to send	PN532_Contactless Command

Lc: Number of Bytes to Send (1 Byte)
Maximum 255 bytes

Data In: PN532_Contactless Command
The data to be sent to the PN532 and Contactless Tag.

Table 1.0B: Direct Transmit Response Format (2 Bytes)

Response	Data Out	
Result	SW1	SW2

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

Table 1.0C: 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	00	The operation is failed.
Time Out Error	63	01	The PN532 does not response.

Checksum Error	63	27	The checksum of the Contactless Response is wrong.
Parameter Error	63	7F	The PN532_Contactless Command is wrong.

Get Response
To retrieve the response data after the "Direct Command" is issued.

Table 2.0A: Get Response Command Format (5 Bytes)

Command	Class	INS	P1	P2	Le
Get Response	0xFF	0xC0	0x00	0x00	Number of Bytes to retrieve

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

Table 2.0B: Get Response Format (Le bytes, Length of the Response Data)

Response	Data Out
Result	Response Data

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

Remark:

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.

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

Table 3.0A: Bi-Color LED and Buzzer Control Command Format (9 Bytes)

Command	Class	INS	P1	P2	Lc	Data In (4 Bytes)
Bi-Color and Buzzer LED Control	0xFF	0x00	0x40	LED State Control	0x04	Blinking Duration Control

P2: LED State Control

Table 3.0B: Bi-Color LED and Buzzer Control Format (1 Byte)

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

Data In: Blinking Duration Control

Table 3.0C: Bi-Color LED Blinking Duration Control Format (4 Bytes)

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

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

0x00: The buzzer will not turn on
0x01: The buzzer will turn on during the T1 Duration
0x02: The buzzer will turn on during the T2 Duration
0x03: The buzzer will turn on during the T1 and T2 Duration.

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

Table 3.0D: Status Code

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

Table 3.0E: Current LED State (1 Byte)

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	

Remark:

A. The LED State operation will be performed after the LED Blinking operation is completed.
B. The LED will not be changed if the corresponding LED Mask is not enabled.
C. The LED will not be blinking if the corresponding LED Blinking Mask is not enabled. Also, the number of repetition must 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%. #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, just set the parameter "Link to Buzzer" to zero.

Get the Firmware Version of the reader
To retrieve the firmware version of the reader.

Table 4.0A: Get Firmware Version Command Format (5 Bytes)

Command	Class	INS	P1	P2	Le
Get Response	0xFF	0x00	0x48	0x00	0x00

Table 4.0B: 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 31 (Hex) = ACR122U101 (ASCII)

Basic Program Flow for Contactless Applications

Step 0. Start the application. The first thing is to connect the "ACR122U PCSC 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 word, the SAM is always existed 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 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 ISO14443-4 tags polling, it may have to turn off the Antenna Field before starting another polling process.

Step 4. Access the Contactless Tag by sending Pseudo APDUs. (Direct Transmit and Get Response)

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

..

Step N. Disconnect the "ACR122U PCSC Interface". Shut down the application.

Remarks:

1) Some Type A Tags may support both ISO14443-3 Type A and ISO14443-4 Type A operating modes. For example, JCOP30 supports MIFARE 1K emulation (ISO14443-3) and ISO14443-4. If the reader sends a RATS command to the tag, the ISO14443-4 mode will be activated, or the tag remains in MIFARE 1K emulation mode (ISO14443-3). It is up to the application to decide which operating mode to be activated. By default, the reader will perform automatic ISO14443-4 activation if the tag supports ISO14443-4.

   - To disable automatic ISO14443-4 activation: FF 00 00 00 03 D4 12 24
   - To Enable automatic ISO14443-4 activation: FF 00 00 00 03 D4 12 34

2) The default Retry Time of the PN532 command "InListPassiveTarget" is infinity. That means, after the polling command is sent out, the reader will wait until a valid tag is found. If the application wants to get the 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 00

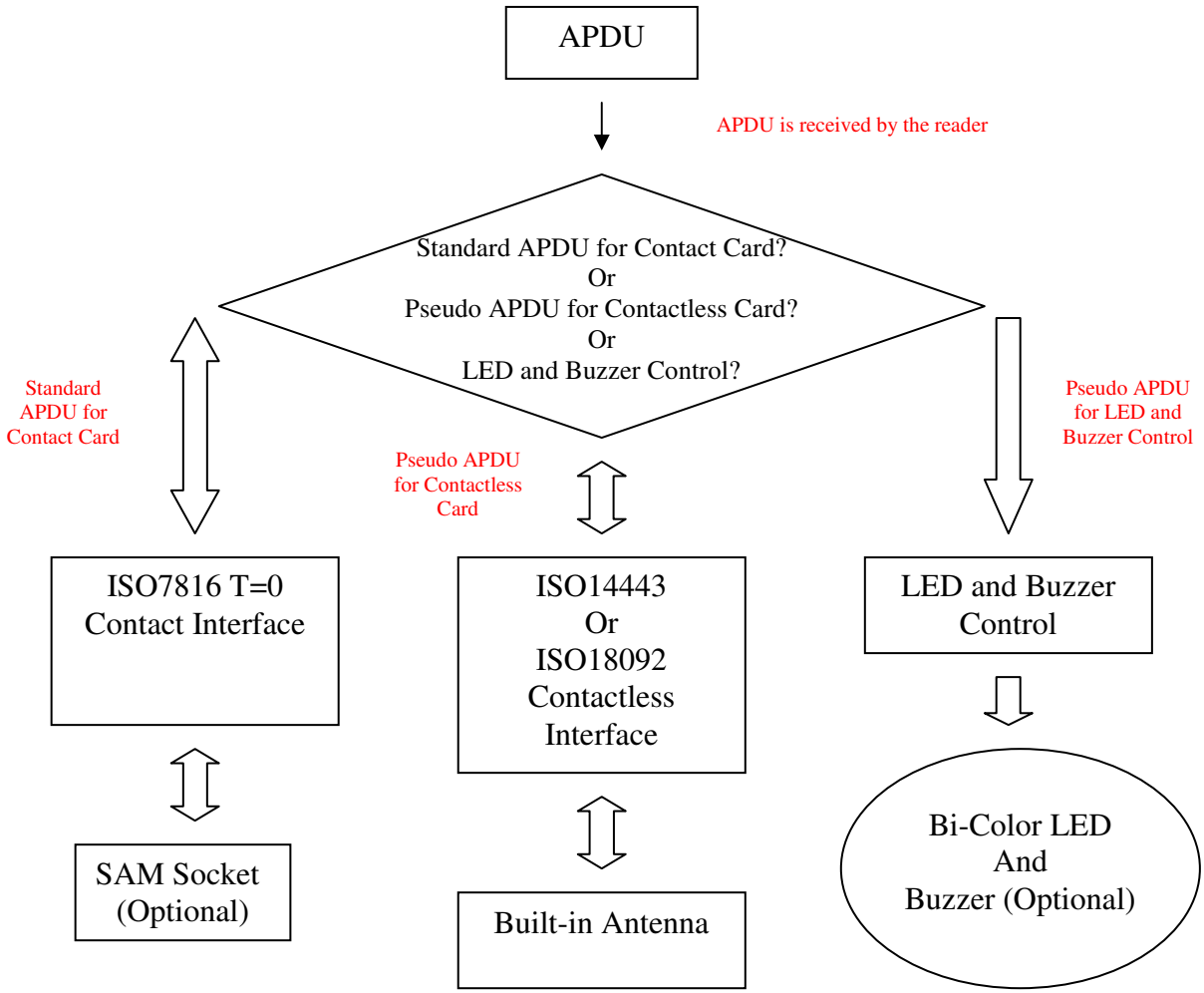
3) The antenna can be switched off in order to save the power.

   - Turn off the antenna power: FF 00 00 00 04 D4 32 01 00
   - Turn on the antenna power: FF 00 00 00 04 D4 32 01 01

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".

   The reader will check the content of the APDU to determine which interface will be used



6) For the contactless interface: Because of the limitation of ISO7816 T=0 protocol (Standard Microsoft CCID drivers), it is not possible to send both "Lc" and "Le" in a single APDU. Therefore, we have to split the APDU into two separate APDUs. Firstly, we send the APDU "Direct Transmit" to get the length of the response data, and then send the APDU "Get Response" 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.

Card Access

How to access MIFARE Classic Tags?

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 00
>> 61 0E (a tag is found)
<< FF C0 00 00 0E
>> D5 4B 01 01 00 02 18 04 F6 8E 2A 99 90 00
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

Tip: If no tag is found, the following response will be returned.
>> 61 05 (no tag found)
<< FF C0 00 00 05
>> D5 4B 00 90 00
```

Tip: 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
<< FF 00 00 00 0F D4 40 01 60 04 FF FF FF FF FF FF F6 8E 2A 99
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00

Tip: If the authentication failed, the error code [XX] will be returned.
      [00] = 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 F6 8E 2A 99


Step 3) Read the content of Block 04
<< FF 00 00 00 05 D4 40 01 30 04
>> 61 15
<< FF C0 00 00 15
>> D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00
```

```
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
<< 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
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00

Step 5) Halt the tag (optional)
<< FF 00 00 00 03 D4 44 01
>> 61 05
<< FF C0 00 00 05
>> D5 45 [00] 90 00
```

MIFARE 1K Memory Map.

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 ~ 0x02	0x03
Sector 1	0x04 ~ 0x06	0x07
..		
Sector 14	0x38 ~ 0x0A	0x3B
Sector 15	0x3C ~ 0x3E	0x3F

1K Bytes

MIFARE 4K 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 ~ 0x02	0x03
Sector 1	0x04 ~ 0x06	0x07
..		
Sector 30	0x78 ~ 0x7A	0x7B
Sector 31	0x7C ~ 0x7E	0x7F

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 ~ 0x8E	0x8F
Sector 33	0x90 ~ 0x9E	0x9F
..		
Sector 38	0xE0 ~ 0xEE	0xEF
Sector 39	0xF0 ~ 0xFE	0xFF

2K Bytes

Tip: Once the authentication is done, all the data blocks of the same sector are free to access. For example, once the data block 0x04 is successfully authenticated (Sector 1), the data blocks 0x04 ~ 0x07 are free to access.

How to handle Value Blocks of MIFARE 1K/4K Tag?

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				Value				Value			Adr	Adr	Adr	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 = 0x05
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)
<< 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
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00

Step 2) Increment the value of Block 05 by 1 (dec)
<< FF 00 00 00 09 D4 40 01 C1 05 01 00 00 00
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00

Tip: Decrement the value of Block 05 by 1 (dec)
<< FF 00 00 00 09 D4 40 01 C0 05 01 00 00 00

Step 3) Transfer the prior calculated value of Block 05 (dec)
<< FF 00 00 00 05 D4 40 01 B0 05
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00

Tip: Restore the value of Block 05 (cancel the prior Increment or Decrement
operation)
<< FF 00 00 00 05 D4 40 01 C2 05

Step 4) Read the content of Block 05
<< FF 00 00 00 05 D4 40 01 30 05
>> 61 15
<< FF C0 00 00 15
>> D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00
In which, the value = 101 (dec)
```

```
Step 5) Copy the value of Block 05 to Block 06 (dec)
<< FF 00 00 00 05 D4 40 01 C2 05
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00
<< FF 00 00 00 05 D4 40 01 B0 06
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00

Step 6) Read the content of Block 06
<< FF 00 00 00 05 D4 40 01 30 06
>> 61 15
<< FF C0 00 00 15
>> D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00
In which, the value = 101 (dec). The Adr “05 FA 05 FA” tells us the value is
copied from Block 05.
```

#please refer to the MIFARE specification for more detailed information.

How to access 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 00
>> 61 11 (a tag is found)
<< FF C0 00 00 11
>> D5 4B 01 01 00 44 00 07 04 6E 0C A1 BF 02 84 90 00
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

Tip: If no tag is found, the following response will be returned.
>> 61 05 (no tag found)
<< FF C0 00 00 05
>> D5 4B 00 90 00

Step 2) Read the content of Block 04
<< FF 00 00 00 05 D4 40 01 30 04
>> 61 15
<< FF C0 00 00 15
>> D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00
In which, Block Data = 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16

Tip: 4 consecutive blocks will be retrieved. Blocks 4, 5, 6 and 7 will be
retrieved. Each data block consists of 4 bytes.

Step 3) Update the content of Block 04 with the data "AA BB CC DD"
<< 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
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00

Tip: 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 block 4 (4
bytes) is updated even through 16 byte are sent to the reader.

Step 4) Read the content of Block 04 again
<< FF 00 00 00 05 D4 40 01 30 04
>> 61 15
<< FF C0 00 00 15
>> D5 41 [00] AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16 90 00
In which, Block Data = AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16

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

Step 5) Halt the tag (optional)
<< FF 00 00 00 03 D4 44 01
>> 61 05
<< FF C0 00 00 05
>> D5 45 [00] 90 00
```

#please refer to the MIFARE Ultralight specification for more detailed information.

How to access ISO14443-4 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 ISO14443-4 Type A Tag, 106 kbps
<< FF 00 00 00 04 D4 4A 01 00
>> 61 15 (a tag is found)
<< FF C0 00 00 15
>> D5 4B 01 01 00 08 28 04 85 82 2F A0 07 77 F7 80 02 47 65 90 00
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
             Operation Finished = 90 00
```

Or

```
Step 1) Polling for the ISO14443-4 Type B Tag, 106 kbps
<< FF 00 00 00 05 D4 4A 01 03 00
>> 61 14 (a tag is found)
<< FF C0 00 00 14
>> D5 4B 01 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21 90 00
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 2) Change the default Baud Rate to other Baud Rate (optional)
<< FF 00 00 00 05 D4 4E 01 02 02 // Change to Baud Rate 424 kbps
Or
<< FF 00 00 00 05 D4 4E 01 01 01 // Change to Baud Rate 212 kbps
>> 61 05
<< FF C0 00 00 05
>> D5 4F [00] 90 00
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
<< FF 00 00 00 08 D4 40 01 00 84 00 00 08
>> 61 0F
<< FF C0 00 00 0F
>> D5 41 [00] 62 89 99 ED C0 57 69 2B 90 00 90 00
In which, Response Data = 62 89 99 ED C0 57 69 2B 90 00
```

```
Step 4) Deselect the Tag
<< FF 00 00 00 03 D4 44 01
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00
```

```
Step 5) Turn off the Antenna Power (optional)
<< FF 00 00 00 04 D4 32 01 00
>> 61 04
```

#please refer to the Tag specification for more detailed information.

How to access 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 00
<< FF 00 00 00 09 D4 4A 01 01 00 FF FF 00 00
>> 61 1A (a tag is found)
<< FF C0 00 00 0C
>> 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,    Number of Tag found = [01]; Target number = 01
             POL_RES Length = 14; Response Code = 01
             NFCID2 = 01 01 05 01 86 04 02 02
             PAD = 03 00 4B 02 4F 49 8A 8A 80 08
             Operation Finished = 90 00
```

Tip: For FeliCa Tag, 424 kbps
<< FF 00 00 00 09 D4 4A 01 ***02*** *00 FF FF 00 00.*

```
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 00
>> 61 22
<< FF C0 00 00 22
>> 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
```

```
Step 3) Deselect the Tag
<< FF 00 00 00 03 D4 44 01
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00
```

#please refer to the FeliCa specification for more detailed information.

How to access NFC Forum Type 1 Tags? E.g. 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 04
>> 61 0C (a tag is found)
<< FF C0 00 00 0C
>> D5 4B 01 01 0C 00 18 26 21 00 90 00
In which,    Number of Tag found = [01]; Target number = 01
             ATQA_RES = 0C 00; UID = 18 26 21 00
             Operation Finished = 90 00

Step 2) Read the memory block 08
<< FF 00 00 00 05 D4 40 01 01 08
>> 61 06
<< FF C0 00 00 06
>> D5 41 [00] 18 90 00
In which, Response Data = 18

Step 3) Update the memory block 08
<< FF 00 00 00 06 D4 40 01 53 08 FF
>> 61 06
<< FF C0 00 00 06
>> D5 41 [00] FF 90 00
In which, Response Data = FF

Step 4) Deselect the Tag
<< FF 00 00 00 03 D4 44 01
>> 61 05
<< FF C0 00 00 05
>> D5 41 [00] 90 00
```

#please refer to the Jewel and Topaz specification for more detailed information.

Get the current setting of the contactless interface

```
Step 1) Get Status Command
<< FF 00 00 00 02 D4 04
>> 61 0C
<< FF C0 00 00 0C
>> D5 05 [Err] [Field] [NbTg] [Tg] [BrRx] [BrTx] [Type] 80 90 00

Or if no tag is in the field
>> 61 08
<< FF C0 00 00 08
>> D5 05 00 00 00 80 90 00
```

[Err] is an error code corresponding to the latest error detected by the PN532.
Field indicates if an external RF field is present and detected by the PN532 (Field = 0x01) or not (Field = 0x00).

[NbTg] is the number of targets currently controlled by the PN532 acting as initiator. The default value is 1.

[Tg]: logical number

[BrRx] : bit rate in reception
- 0x00 : 106 kbps
- 0x01 : 212 kbps
- 0x02 : 424 kbps

[BrTx] : bit rate in transmission
- 0x00 : 106 kbps
- 0x01 : 212 kbps
- 0x02 : 424 kbps

[Type]: modulation type
- 0x00 : ISO14443 or Mifare®
- 0x10 : FeliCa™
- 0x01 : Active mode
- 0x02 : Innovision Jewel® tag

Appendix 1: Error Codes Table

Error Cause	Error Code
No Error	**0x00**
Time Out, the target has not answered	0x01
A CRC error has been detected by the contactless UART	0x02
A Parity error has been detected by the contactless UART	0x03
During a MIFARE anticollision/select operation, an erroneous Bit Count has been detected	0x04
Framing error during MIFARE operation	0x05
An abnormal bit-collision has been detected during bit wise anticollision at 106 kbps	0x06
Communication buffer size insufficient	0x07
RF Buffer overflow has been detected by the contactless UART (bit BufferOvfl of the register CL_ERROR)	0x08
In active communication mode, the RF field has not been switched on in time by the counterpart (as defined in NFCIP-1 standard)	0x0A
RF Protocol error (cf. reference [4], description of the CL_ERROR register)	0x0B
Temperature error: the internal temperature sensor has detected overheating, and therefore has automatically switched off the antenna drivers	0x0D
Internal buffer overflow	0x0E
Invalid parameter (range, format, …)	0x10
DEP Protocol: The the PN532 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]).	0x12
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.	0x13
Mifare: Authentication error	0x14
ISO/IEC 14443-3: UID Check byte is wrong	0x23
DEP Protocol: Invalid device state, the system is in a state which does not allow the operation	0x25
Operation not allowed in this configuration (host controller interface)	0x26
This command is not acceptable due to the current context of the PN532 (Initiator vs. Target, unknown target number, Target not in the good state, …)	0x27
The the PN532 configured as target has been released by its initiator	0x29
The PN5321 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.	0x2A
The PN5321 and ISO/IEC 14443-3B only: the card previously activated has disappeared.	0x2B
Mismatch between the NFCID3 initiator and the NFCID3 target in DEP 212/424 kbps passive.	0x2C
An over-current event has been detected	0x2D
NAD missing in DEP frame	0x2E

Appendix 2: Sample Codes for accessing FeliCa Tags

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 PN532_Contactless Command = "09"
PN532 Command (InListPassiveTarget 212Kbps) = "D4 4A 01 01"
Contactless Command (System Code Request) = "00 FF FF 01 00"

The APDU Response would be "61 1A" is a Tag is found, or "61 05" is no Tag is found!

Step 2: Issue a "Get Response" APDU
The APDU Command would be "FF C0 00 00 1A"

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 00"

#In which,
Response returned by the PN532 =
"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 Contactless 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"

#In which,
Direct Transmit APDU = "FF 00 00 00"
Length of the PN532_Contactless Command = "23"
PN532 Command (InDataExchange) = "D4 40 01"
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".

The APDU Response would be "61 11"

Step 2: Issue a "Get Response" APDU
The APDU Command would be "FF C0 00 00 11"

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 PN532 = "D5 41"
Response returned by the Contactless 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 PN532_Contactless Command = "13"
PN532 Command (InDataExchange) = "D4 40 01"
Contactless Command (Read Data) = "10 06 01 01 05 01 86 04 02 02 01 09 01 01 80 00"

The APDU Response would be "61 22"

Step 2: Issue a "Get Response" APDU
The APDU Command would be "FF C0 00 00 22"

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 PN532 = "D5 41"
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 AA"
Status Code returned by the reader = "90 00"

Appendix 3: Sample Codes for Setting the LED

Example 1: To read the existing LED State.
// Assume both Red and Green LEDs are OFF initially //
// Not link to the buzzer //

APDU = “FF 00 40 00 04 00 00 00 00”
Response = “90 00”. RED and Green LEDs are OFF.

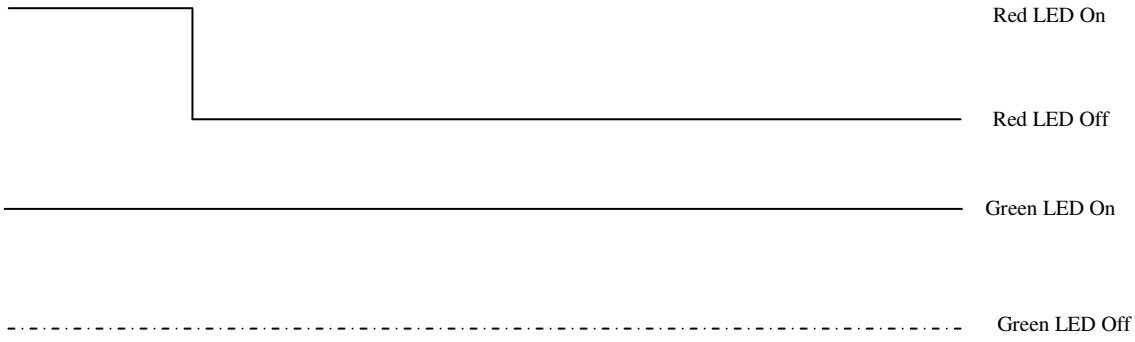
Example 2: To turn on RED and Green Color LEDs
// Assume both Red and Green LEDs are OFF initially //
// Not link to the buzzer //

APDU = “FF 00 40 0F 04 00 00 00 00”
Response = “90 03”. RED and Green LEDs are ON,

#To turn off both RED and Green LEDs, APDU = “FF 00 40 0C 04 00 00 00 00”

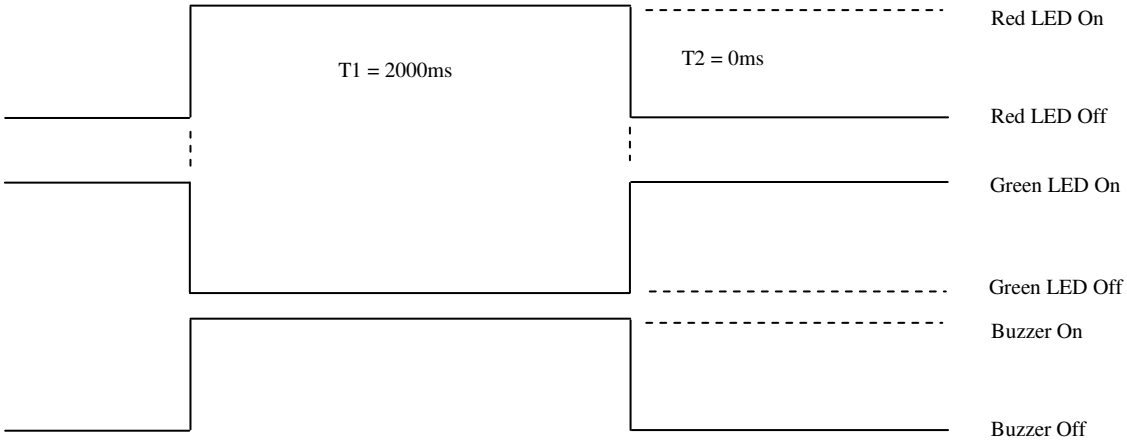
Example 3: To turn off the RED Color LED only, and left the Green Color LED unchanged.
// Assume both Red and Green LEDs are ON initially //
// Not link to the buzzer //

APDU = “FF 00 40 04 04 00 00 00 00”
Response = “90 02”. Green LED is not changed (ON); Red LED is OFF,



Example 4: To turn on the Red LED for 2 sec. After that, 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. //

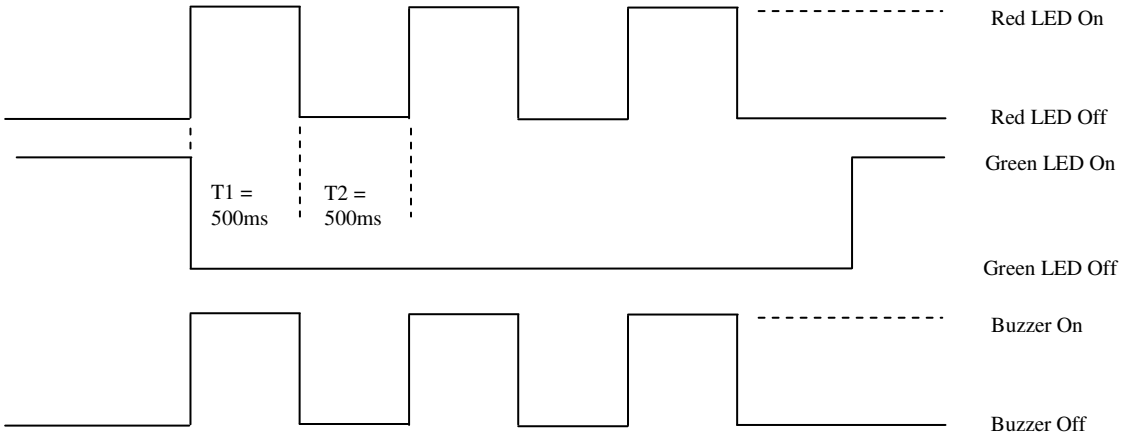


1Hz = 1000ms Time Interval = 500ms ON + 500 ms OFF
T1 Duration = 2000ms = 0x14
T2 Duration = 0ms = 0x00
Number of repetition = 0x01
Link to Buzzer = 0x01

APDU = "FF 00 40 50 04 14 00 01 01"
Response = "90 02"

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

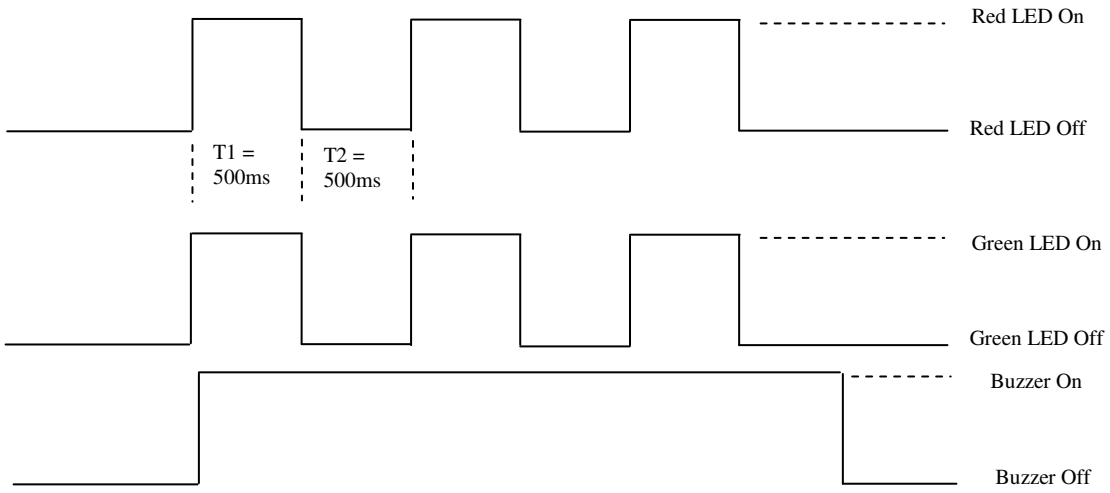
// Assume the Red LED is initially OFF, while the Green LED is initially ON. //
// The Initial Red LED Blinking State is ON. Only the Red LED will be blinking.
// The buzzer will turn on during the T1 duration, while the Green LED will turn off during both the T1 and T2 duration.
// After the blinking,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



1Hz = 1000ms Time Interval = 500ms ON + 500 ms OFF
T1 Duration = 500ms = 0x05
T2 Duration = 500ms = 0x05
Number of repetition = 0x03
Link to Buzzer = 0x01

APDU = “FF 00 40 50 04 05 05 03 01”
Response = “90 02”

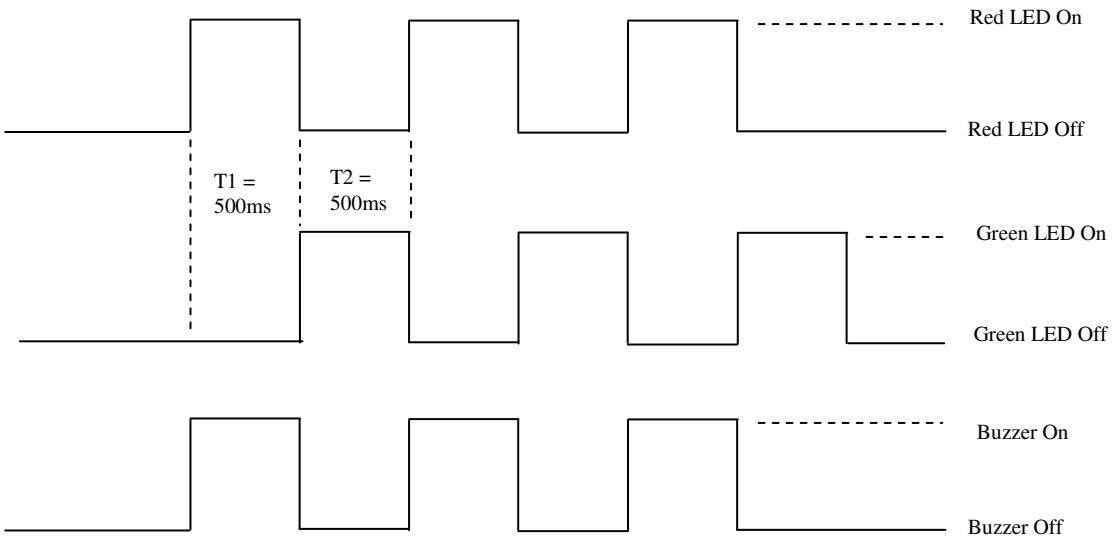
Example 6: To blink the Red and Green LEDs of 1Hz for 3 times
// Assume both the Red and Green LEDs are initially OFF. //
// Both Initial Red and Green Blinking States are ON //
// The buzzer will turn on during both the T1 and T2 duration//



1Hz = 1000ms Time Interval = 500ms ON + 500 ms OFF
T1 Duration = 500ms = 0x05
T2 Duration = 500ms = 0x05
Number of repetition = 0x03
Link to Buzzer = 0x03

APDU = “FF 00 40 F0 04 05 05 03 03”
Response = “90 00”

Example 7: To blink the Red and Green LED in turn of 1Hz for 3 times
// 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 = 500ms ON + 500 ms OFF
T1 Duration = 500ms = 0x05
T2 Duration = 500ms = 0x05
Number of repetition = 0x03
Link to Buzzer = 0x01
APDU = “FF 00 40 D0 04 05 05 03 01”; Response = “90 00”