

13.56MHz Reader / Writer

E1356D

User Manual

Version 2.6

Nov 2011



CONTENT

| | |
|--|-----------|
| 1. GENERAL INFORMATION | 5 |
| 2. TYPES AND EXPLANATION..... | 6 |
| 3. CONNECTING TO PC | 6 |
| 3.1 E1356D-RS232..... | 6 |
| 3.2 E1356D-USB | 7 |
| 4. SDK..... | 8 |
| 5. DEMO..... | 8 |
| 5.1 ONLINE | 8 |
| 5.2 SHC1102 | 9 |
| 5.3 ULTRALIGHT..... | 9 |
| 5.4 MIFARE_1K | 10 |
| 5.5 MIFARE_4K | 10 |
| 5.6 MIFARE_PROX | 11 |
| 5.7 TRH1064..... | 11 |
| 5.9 SRIX4K..... | 12 |
| 5.10 AT88RF020..... | 13 |
| 5.11 ISO14443B-4 PROTOCOL SMART CARD..... | 13 |
| 5.12 I.CODE SLI..... | 14 |
| 5.13 TAG_IT | 14 |
| 5.14 SRF55V02P..... | 15 |
| 5.15 SRF55V10P..... | 15 |
| 5.16 PASS_THROUGH..... | 16 |
| 6. DLL INFORMATION..... | 17 |
| 6.1 SYSTEM FUNCTION | 17 |
| 6.1.1 INT WINAPI LIB_VER | 17 |
| 6.1.2 INT WINAPI RF_INIT_COM | 17 |
| 6.1.3 INT WINAPI RF_CLOSEPORT | 17 |
| 6.1.4 INT WINAPI RF_GET_MODEL | 17 |
| 6.1.5 INT WINAPI RF_INIT_DEVICE_NUMBER | 17 |
| 6.1.6 INT WINAPI RF_GET_DEVICE_NUMBER..... | 18 |
| 6.1.7 INT WINAPI RF_INIT_TYPE | 18 |
| 6.1.8 INT WINAPI RF_ANTENNA_STA | 18 |
| 6.1.9 INT WINAPI RF_LIGHT | 18 |
| 6.1.10 INT WINAPI RF_BEEP | 18 |
| 6.2 DES FUNCTION..... | 19 |

| | | |
|--------------|---------------------------------------|-----------|
| 6.2.2 | INT WINAPI DES_DECRYPT | 19 |
| 6.3 | ISO14443A FUNCTION | 19 |
| 6.3.1 | UltraLight | 19 |
| 6.3.1.1 | INT WINAPI RF_REQUEST | 19 |
| 6.3.1.2 | INT WINAPI INT RF_UL_SELECT | 20 |
| 6.3.1.3 | INT WINAPI RF_M1_READ..... | 20 |
| 6.3.1.4 | INT WINAPI INT RF_UL_WRITE | 20 |
| 6.3.1.5 | INT WINAPI RF_HALT..... | 20 |
| 6.3.2 | Mifare_Class | 21 |
| 6.3.2.1 | INT WINAPI RF_REQUEST | 21 |
| 6.3.2.2 | INT WINAPI RF_ANTICOLL | 21 |
| 6.3.2.3 | INT WINAPI RF_SELECT | 21 |
| 6.3.2.4 | INT WINAPI RF_M1_AUTHENTICATION2..... | 22 |
| 6.3.2.5 | INT WINAPI RF_M1_READ..... | 22 |
| 6.3.2.6 | INT WINAPI RF_M1_WRITE..... | 22 |
| 6.3.2.7 | INT WINAPI RF_M1_INITVAL..... | 22 |
| 6.3.2.8 | INT WINAPI RF_M1_READVAL..... | 23 |
| 6.3.2.9 | INT WINAPI RF_M1_INCREMENT | 23 |
| 6.3.2.10 | INT WINAPI RF_M1_DECREMENT | 23 |
| 6.3.2.11 | INT WINAPI RF_M1_RESTORE | 23 |
| 6.3.2.12 | INT WINAPI RF_M1_TRANSFER | 23 |
| 6.3.2.13 | INT WINAPI RF_HALT..... | 24 |
| 6.3.3 | Mifare_DESFire | 24 |
| 6.3.3.1 | INT WINAPI RF_DESFIRE_RST | 24 |
| 6.3.3.2 | INT WINAPI RF_COS_COMMAND..... | 24 |
| 6.3.4 | Mifare_ProX..... | 25 |
| 6.3.4.1 | INT WINAPI RF_TYPE_RST..... | 25 |
| 6.3.4.2 | INT WINAPI RF_COS_COMMAND..... | 25 |
| 6.3.4.3 | INT WINAPI RF_CL_DESELECT..... | 25 |
| 6.3.5 | SHC1102..... | 25 |
| 6.3.5.1 | INT WINAPI RF_REQUEST | 25 |
| 6.3.5.2 | INT WINAPI RF_SHC1102_AUTH | 26 |
| 6.3.5.3 | INT WINAPI RF_SHC1102_READ | 26 |
| 6.3.5.4 | INT WINAPI RF_SHC1102_WRITE..... | 26 |
| 6.4 | ISO14443B FUNCTION | 27 |
| 6.4.1 | THR1064 | 27 |
| 6.4.1.1 | INT WINAPI RF_TYPEB_RST | 27 |
| 6.4.1.2 | INT WINAPI RF_THR1064_READ | 27 |
| 6.4.1.3 | INT WINAPI RF_THR1064_WRITE | 27 |
| 6.4.1.4 | INT WINAPI RF_THR1064_CHECK..... | 27 |
| 6.4.2 | AT88RF020 | 28 |
| 6.4.2.1 | INT WINAPI RF_TYPEB_RST | 28 |
| 6.4.2.2 | INT WINAPI RF_AT020_CHECK..... | 28 |
| 6.4.2.3 | INT WINAPI RF_AT020_COUNT..... | 28 |

| | | |
|--------------|--|-----------|
| 6.4.2.4 | INT WINAPI RF_AT020_READ | 28 |
| 6.4.2.5 | INT WINAPI RF_AT020_WRITE..... | 29 |
| 6.4.2.6 | INT WINAPI RF_AT020_LOCK | 29 |
| 6.4.2.7 | INT WINAPI RF_AT020_DESELECT | 29 |
| 6.4.3 | SR176SRIX4K..... | 29 |
| 6.4.3.1 | INT WINAPI RF_ST_SELECT..... | 29 |
| 6.4.3.2 | INT WINAPI INT_RF_SR176_READBLOCK | 29 |
| 6.4.3.3 | INT WINAPI INT_RF_SR176_WRITEBLOCK | 30 |
| 6.4.3.4 | INT WINAPI INT_RF_SR176_PROTECTBLOCK | 30 |
| 6.4.3.5 | INT WINAPI INT_RF_SRIX4K_GETUID | 30 |
| 6.4.3.6 | INT WINAPI INT_RF_SRIX4K_READBLOCK | 30 |
| 6.4.3.7 | INT WINAPI INT_RF_SRIX4K_WRITEBLOCK | 31 |
| 6.4.3.8 | INT WINAPI INT_RF_SRIX4K_PROTECTBLOCK | 31 |
| 6.4.3.9 | INT WINAPI RF_ST_COMPLETION..... | 31 |
| 6.4.4 | TYPE_B SmartCard..... | 32 |
| 6.4.4.1 | INT WINAPI RF_TYPEB_RST | 32 |
| 6.4.4.2 | INT WINAPI RF_COS_COMMAND..... | 32 |
| 6.4.4.3 | INT WINAPI RF_CL_DESELECT | 32 |
| 6.5 | ISO15693 FUNCTION..... | 33 |
| 6.5.1 | INT WINAPI ISO15693_INVENTORY | 33 |
| 6.5.2 | INT WINAPI ISO15693_INVENTORYS | 33 |
| 6.5.3 | INT WINAPI ISO15693_GET_SYSTEM_INFORMATION | 33 |
| 6.5.4 | INT WINAPI ISO15693_SELECT..... | 34 |
| 6.5.5 | INT WINAPI ISO15693_RESET_TO_READY | 34 |
| 6.5.6 | INT WINAPI ISO15693_STAY_QUIET..... | 34 |
| 6.5.7 | INT WINAPI ISO15693_GET_BLOCK_SECURITY..... | 34 |
| 6.5.8 | INT WINAPI ISO15693_READ..... | 35 |
| 6.5.9 | INT WINAPI ISO15693_WRITE..... | 35 |
| 6.5.10 | INT WINAPI ISO15693_LOCK_BLOCK..... | 36 |
| 6.5.11 | INT WINAPI ISO15693_WRITE_AFI | 36 |
| 6.5.12 | INT WINAPI ISO15693_LOCK_AFI..... | 36 |
| 6.5.13 | INT WINAPI ISO15693_WRITE_DSFD | 37 |
| 6.5.14 | INT WINAPI ISO15693_LOCK_DSFD | 37 |
| 6.6 | FUNCTION OF INFINEON ELECTRIC TAG | 38 |
| 6.6.1 | INT WINAPI SRF55VP_READ..... | 38 |
| 6.6.2 | INT WINAPI SRF55VP_WRITEBYTE..... | 38 |
| 6.6.3 | INT WINAPI SRF55VP_WRITE..... | 38 |
| 6.6.4 | INT WINAPI SRF55VP_WRITE_REREAD | 39 |
| 6.7 | PASS THROUGH FUNCTION | 39 |
| 6.7.1 | INT WIN API RF_TRANSCEIVE1 | 39 |

1. GENERAL INFORMATION



- RS232 or USB Interface
- 4.5 ~ 5.5VDC Operating
- Windows 32 Operating Systems Compatibility
- 13.56MHz RF Operating Frequency
- ISO14443A ISO1443B ISO15693 Protocols
- 150MA Working Current
- Operating Temperature Range: -20°C ~ +50°C
- Storage Temperature Range: -25°C ~ +60°C
- Dimension: 110 × 81 × 26 mm
- Weight: 100g

1. TYPES AND EXPLANATION

E1356D series readers are in accord with ISO14443A, ISO14443B and ISO15693 protocols, and are classified as following sheet

| | E1356D-NFC | E1356D-DES | E1356D-I2 | E1356D-DP |
|-----------|------------|------------|-----------|-----------|
| ISO14443A | √ | √ | | √ |
| ISO14443B | | | | √ |
| ISO15693 | | | √ | √ |

NOTICE: The difference between E1356D-NFC and E1356D-DES

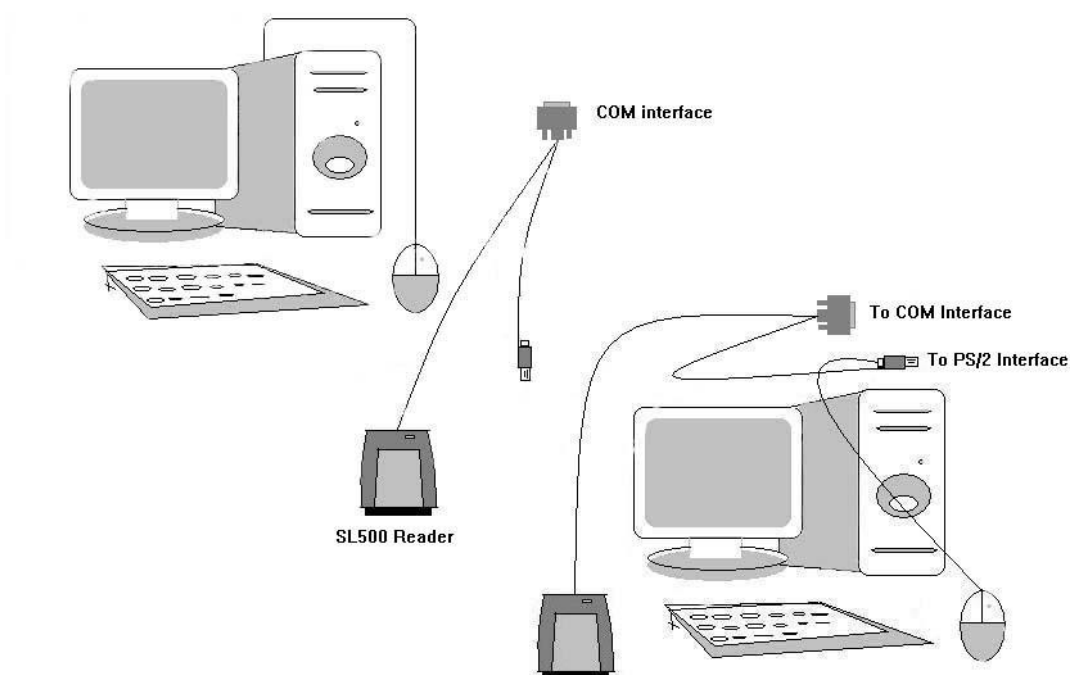
E1356DL supported cards: Mifare_1k, Mifare_4k, UltraLight, NTAG203

E1356DA supported cards: Mifare_1k, Mifare_4k, UltraLight, NTAG203, Mifare_ProX

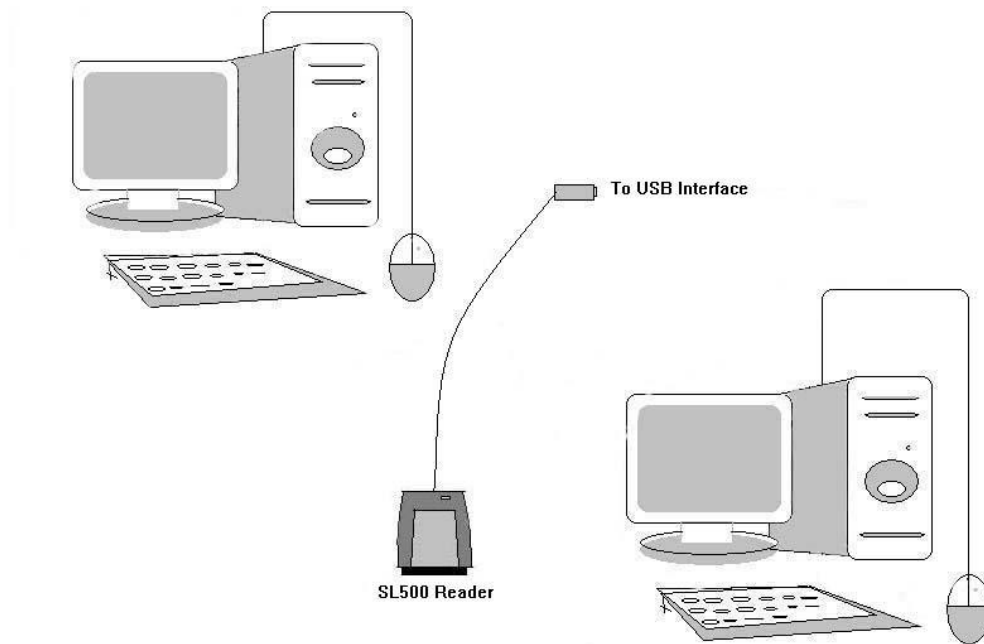
2. CONNECTING TO PC

3.1 E1356D-RS232

The PS/2 port power to Reader

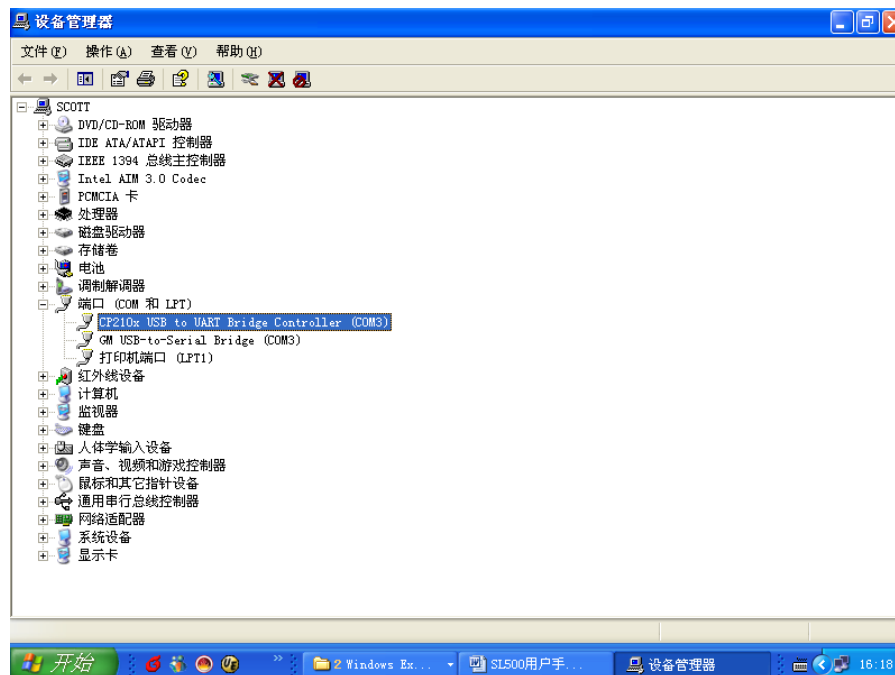


3.2 E1356D-USB



E1356D-USB Reader is USB bridge to COM. Connect E1356D to the USB port of PC, after installing the driver will come out a virtual COM, the operations hereafter are as same as E1356D-RS232.

You can find the virtual COM number on the “Device Manager” as follows:



4. SDK

Responding InstDemo.exe to install the DEMO software and the DLL of the reader to PC, and create corresponding logo on the desk.

The default installation directory is C:\RFREADER, including the following content:

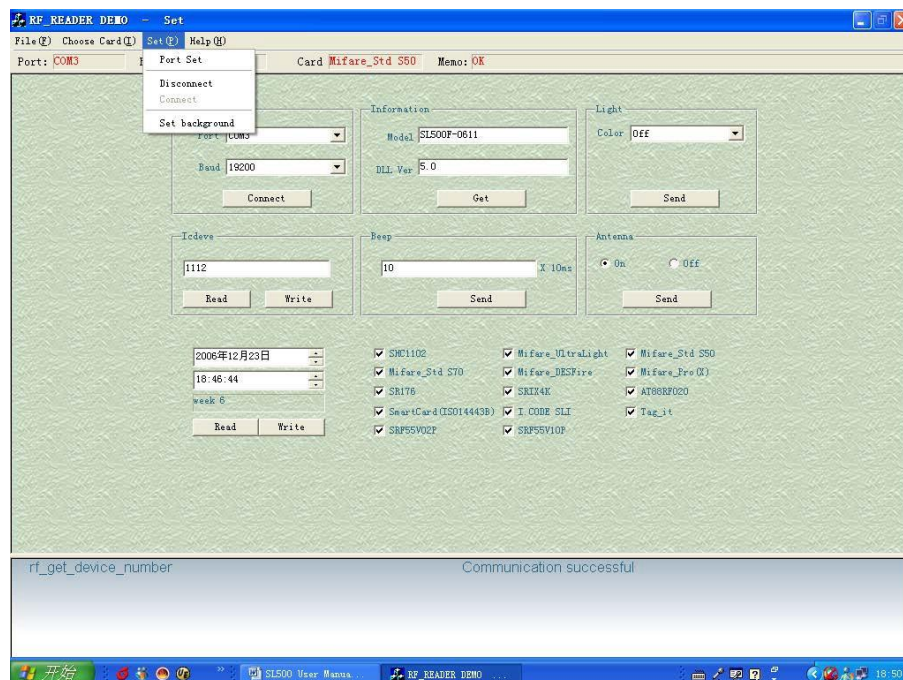
| | |
|----------------------------|--|
| C:\RFREADER\Examples | Sample source code |
| C:\RFREADER\UsbDriver | USB interface driver |
| C:\RFREADER\ICTransfer.exe | DEMO software |
| C:\RFREADER\MasterRD.dll | Reader interface library with application |
| C:\RFREADER\MasterCOM.dll | Connect and transfer data with COM device. |
| C:\RFREADER\SL_Paper.dll | DEMO software background library |
| C:\RFREADER\RFHELP.chm | DLL explanations at chm format |
| C:\RFREADER\AppConfig.ini | DEMO software configuration files |

5. DEMO

This software run on Win32 system, and need 1024 x 768 dpi at least

5.1 Online

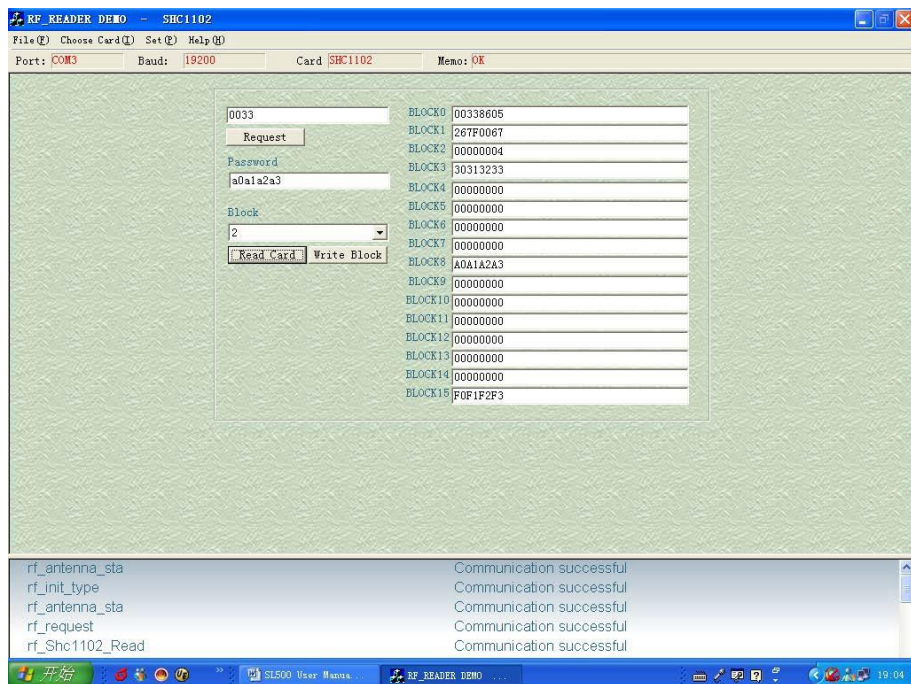
Choose the correct COM number, click [Connect] button to connect the Reader to PC. Click [Read] the product information button, you can check the specific type of the Reader and the supported cards.



5.2 SHC1102

Click [Request] button to obtain the card serial number.

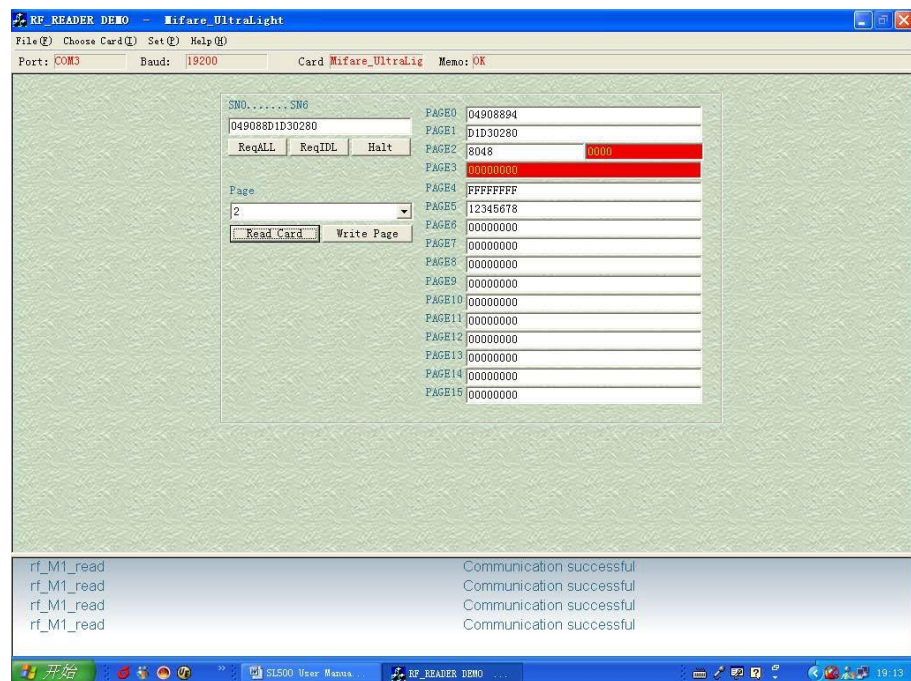
Input the correct key to read/write the card



5.3 UltraLight

Click the [Request] button to obtain the card Serial Number.

Choose the corresponding address to read/write the card.



5.4 Mifare_1k (STD S50)

Click the [Request] button to obtain the card serial number.

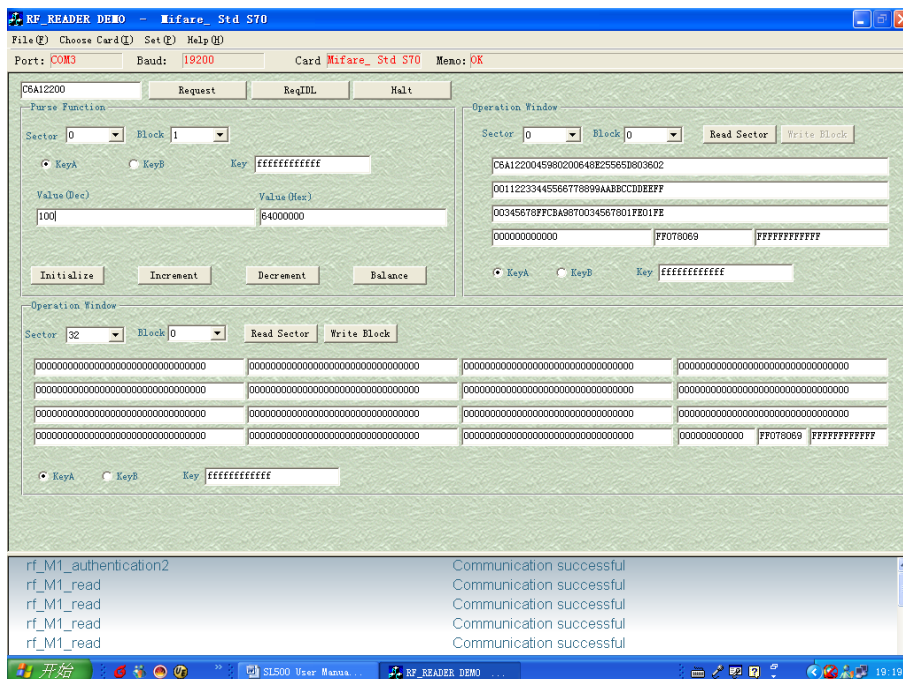
Input the correct password to read, write, increase or decrease the card.



5.5 Mifare_4k (STD S70)

Click the [Request] button to obtain the card serial number.

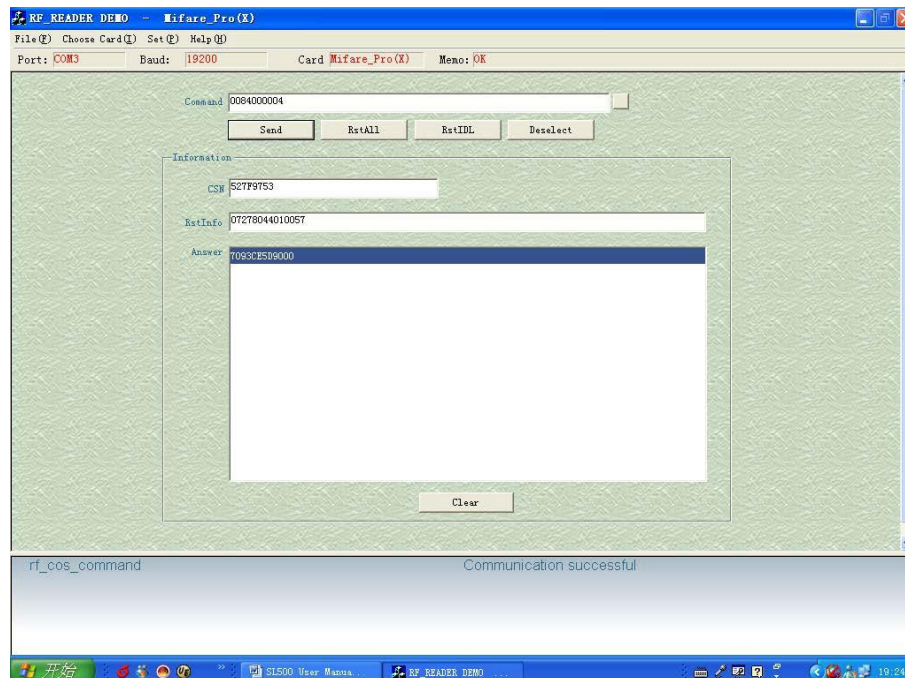
Input the correct password to read, write, increase or decrease the card.



5.6 Mifare_ProX

Click [Reset] button to obtain the serial number and the reset information of the card according to ISO14443-4 protocol.

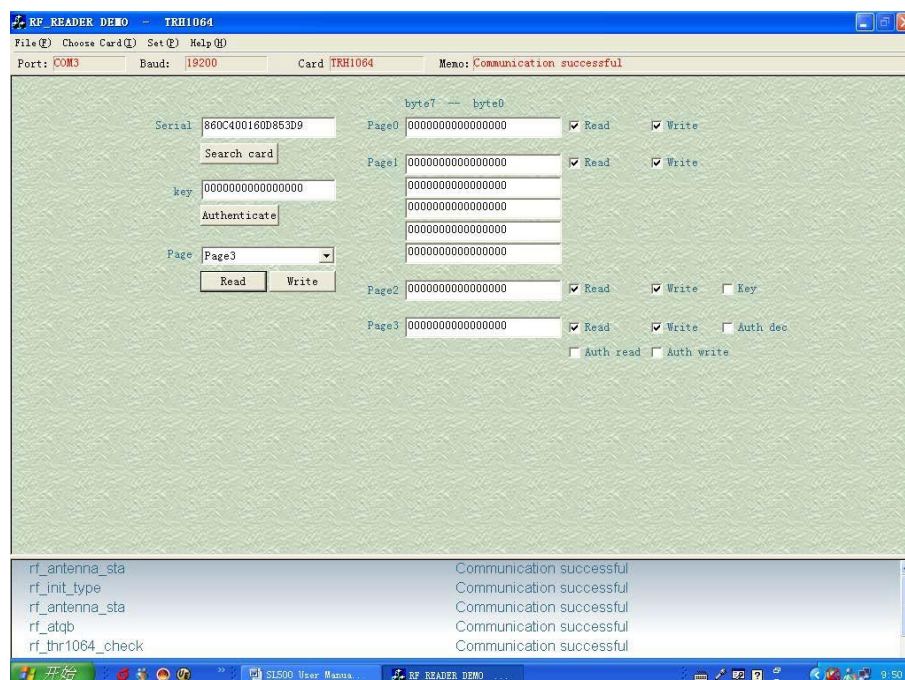
Input the COS command, click [Send] button to commute data to card.



5.7 TRH1064

Click [Request] button to obtain the card serial number.

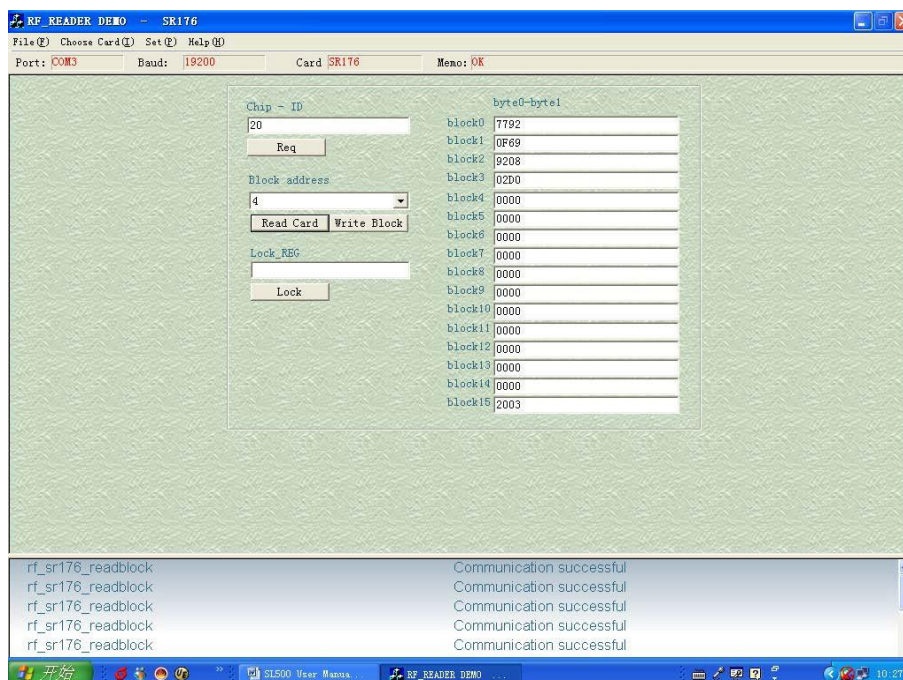
Hereafter can read, write and validate.



5.8 SR176

Click [Req] button to obtain the ID number of the card.

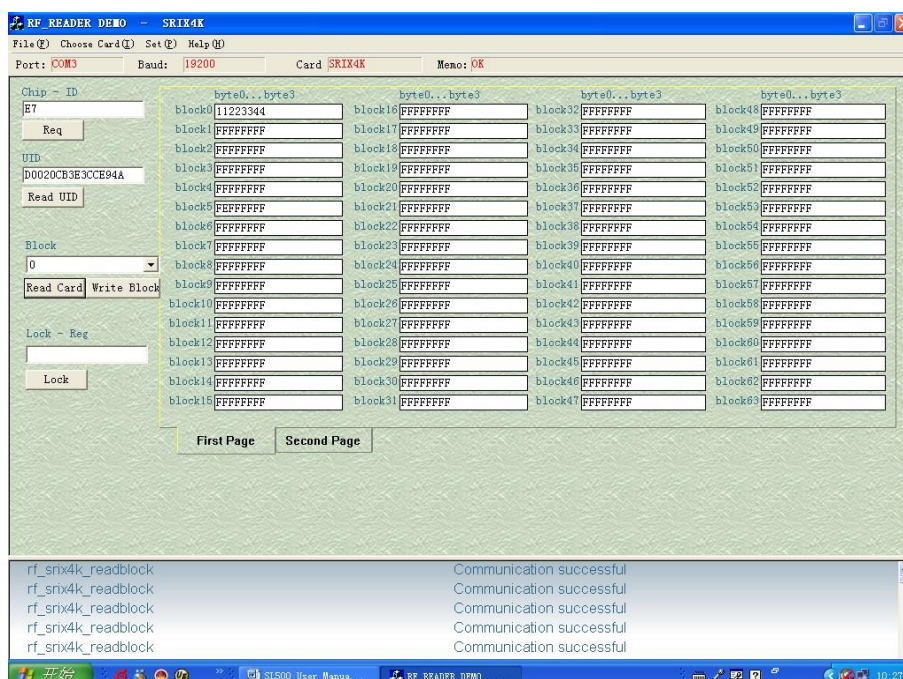
Then you can read, write and lock blocks of the card.



5.9 SR1X4K

Click [Req] button to obtain the ID number of the card and click [Read UID] to obtain the UID of the card.

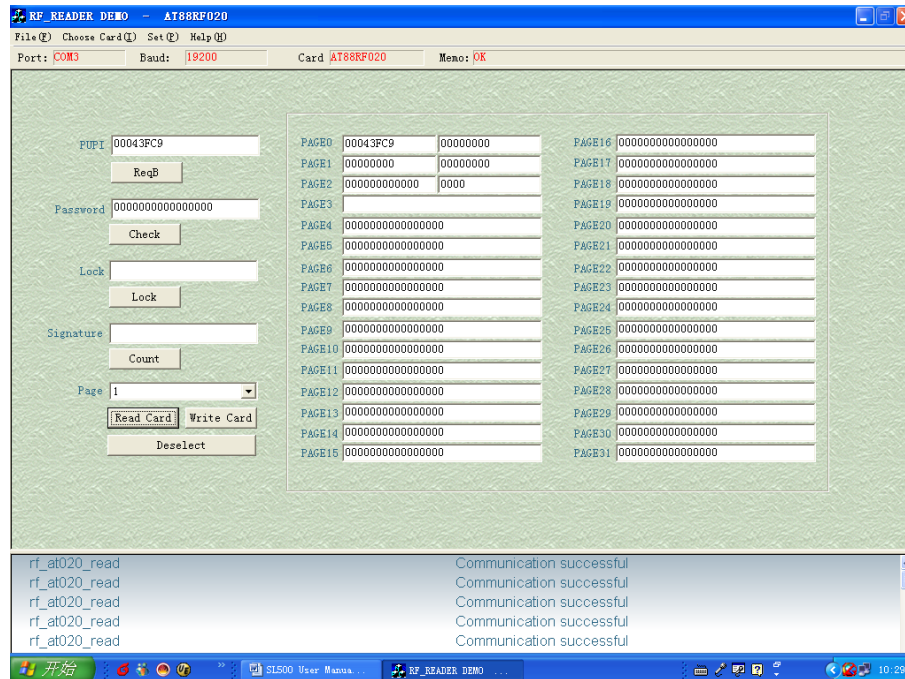
Then you can read, write and lock blocks of the card.



5.10 AT88RF020

Click [ReqB] button to obtain the serial number of the card.

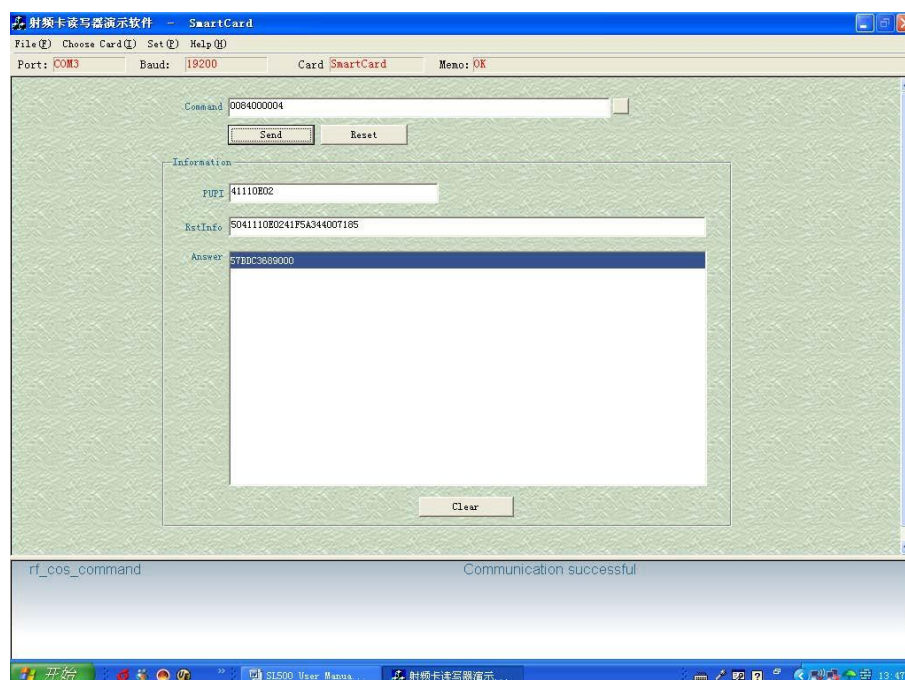
After check password, you can read, write, signature and lock blocks of the card.



5.11 ISO14443B-4 Protocol Smart Card

Click [Reset] button to obtain the serial number and the reset information of the card according to ISO14443-4 protocol.

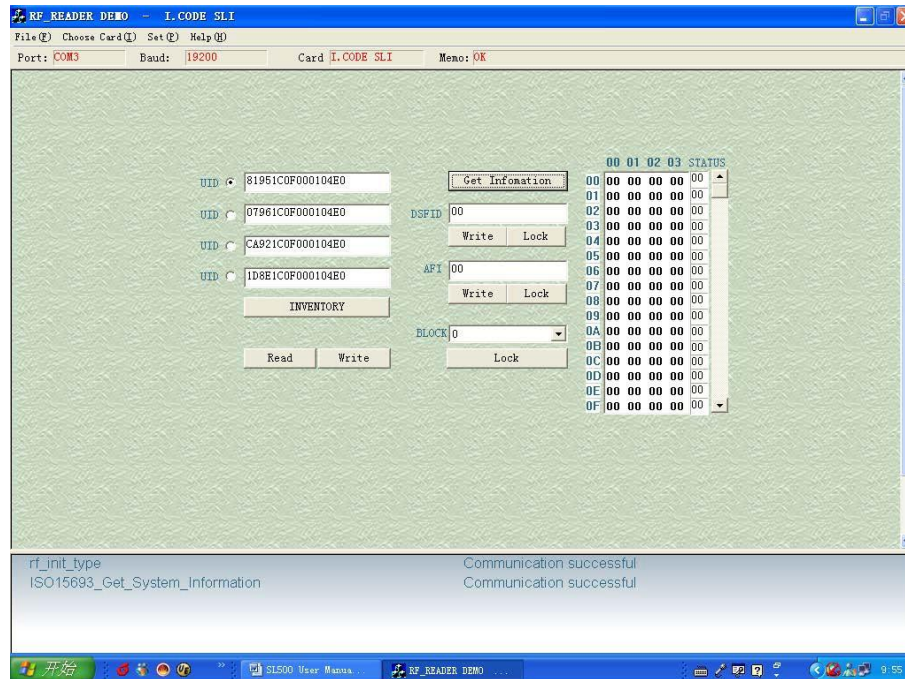
Input the COS command, click [Send] button to commute data to card.



5.12 I.CODE SLI

Click [INVENTORY] button to obtain the serial number of the card. You can operate 4 cards at most.

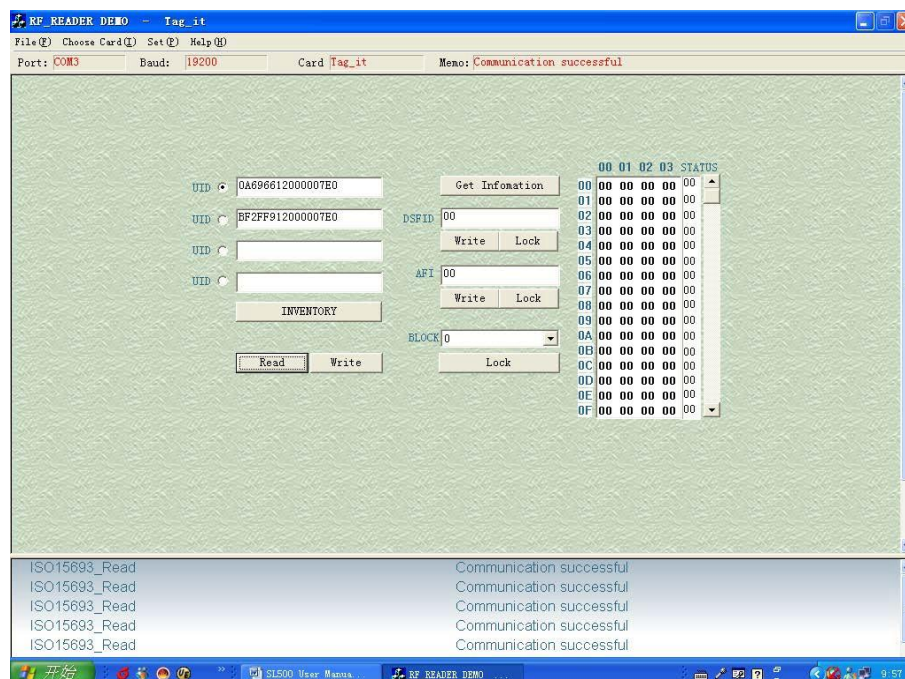
Choose certain card according to the UID to read or write.



5.13 Tag_IT

Click [INVENTORY] button to obtain the serial number of the card. You can operate 4 cards at most.

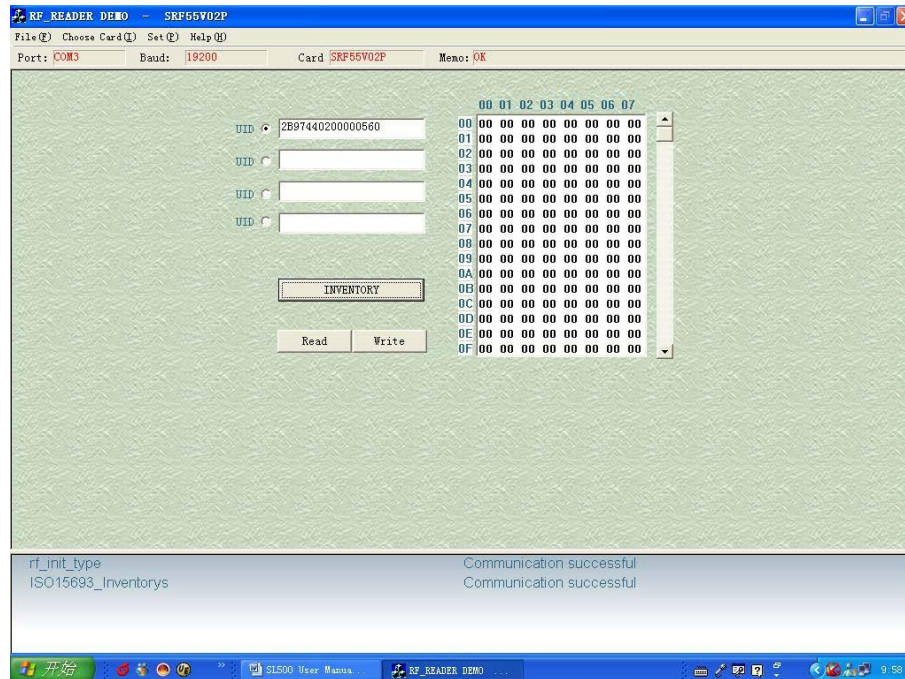
Choose certain card according to the UID to read/write.



5.14 SRF55V02P

Click [INVENTORY] button to obtain the serial number of the card. You can operate 4 cards at most.

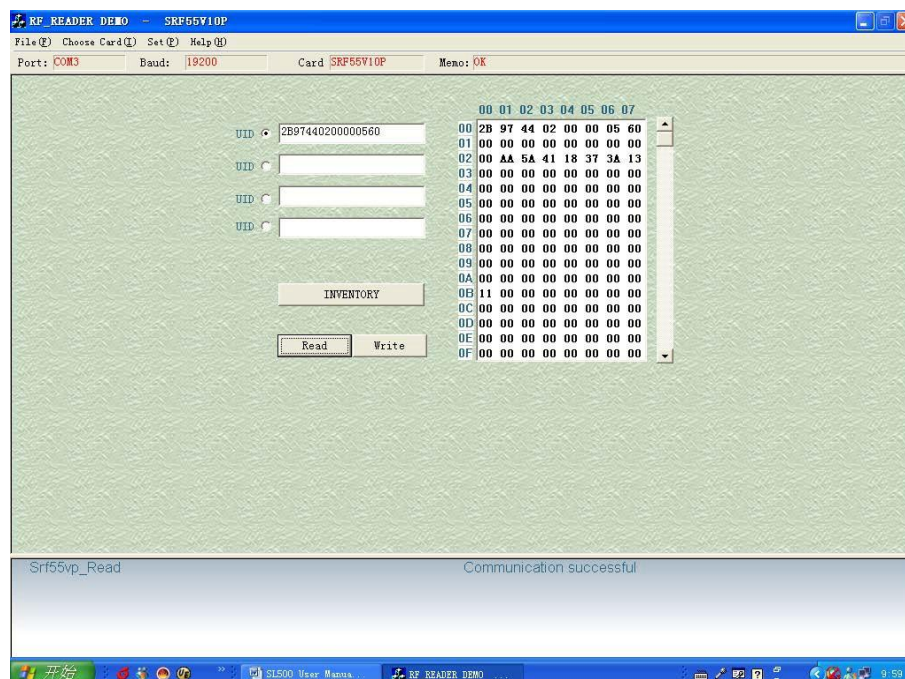
Choose certain card according to the UID to read/write.



5.15 SRF55V10P

Click [INVENTORY] button to obtain the serial number of the card. You can operate 4 cards at most.

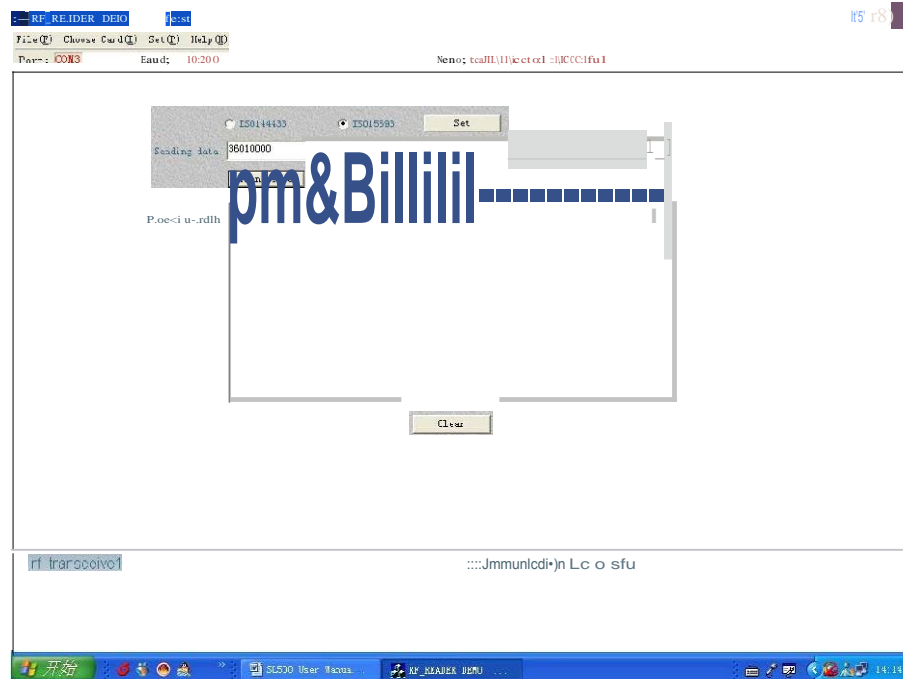
Choose certain card according to the UID to read/write.



5.16 Pass_Through

In this windows, input parameters according to IS014443B and IS015693 protocol, click [Transceive] button to get response data from tag

CRC bytes is auto managed by reader, it will not be contained in the stream



6. DLL INFORMATION

All types of readers have system function and encrypt function.

Whether readers support other functions depends on their specific types.

6.1 SYSTEM FUNCTION

6.1.1 INT WINAPI LIB_VER

Function: Get DLL Version

Prototype: int WINAPI lib_ver (unsigned int *pVer)

Parameter: pVer: [OUT] DLL version

Return: return 0 if successful

6.1.2 INT WINAPI RF_INIT_COM

Function: Connect

Prototype: int WINAPI rf_init_com (int port, long baud)

Parameter: port: [IN] serial port number

baud: [IN] communication baud rate, 4800 ~ 115200 bps

Return 0 on success

6.1.3 INT WINAPI RF_CLOSEPORT

Function: Disconnect

Prototyp: int WINAPI rf_ClosePort(void)

Return 0 on success

6.1.4 INT WINAPI RF_GET_MODEL

Function: Get Device Type

Prototype: int WINAPI rf_get_model (unsigned short icdev,
unsigned char *pVersion,
unsigned char *pLen)

Parameter: icdev: [IN] Device ID

pVersion: [OUT] response information

pLen: [OUT] length of response information

Return 0 on success

6.1.5 INT WINAPI RF_INIT_DEVICE_NUMBER

Function: Designate Device ID

Prototype: int WINAPI rf_init_device_number (unsigned short icdev)

Parameter: icdev: [IN] Device ID

Return 0 on success

6.1.6 INT WINAPI RF_GET_DEVICE_NUMBER

Function: Read Device ID

Prototype: int WINAPI rf_get_device_number (unsigned short *pIcdev)

Parameter: pIcdev: [OUT] response Device ID

Return 0 on success

6.1.7 INT WINAPI RF_INIT_TYPE

Function: Set Reader contactless working mode

Prototype: int WINAPI rf_init_type(unsigned short icdev, unsigned char type)

Parameter: icdev: [IN] Device ID

type: [IN] reader working mode

Return 0 on success

Explanation: this function is not effective to the readers only support single protocol.

type = 'A': set E1356D into ISO14443A mode

type = 'B': set ISO14443B mode

type = 'r': set AT88RF020 card mode

type = 'l': set ISO15693 mode

6.1.8 INT WINAPI RF_ANTENNA_STA

Function: Manage RF Transmittal

Prototype: int WINAPI rf_antenna_sta (unsigned short icdev, unsigned char model)

Parameter: icdev: [IN] Device ID

model: [IN] transmittal state

Return 0 on success

Explanation: model = 0: turn off RF transmittal

model = 1: turn on RF transmittal

6.1.9 INT WINAPI RF_LIGHT

Function: Manage LED

Prototype: int WINAPI rf_light (unsigned short icdev, unsigned char color)

Parameter: icdev: [IN] Device ID

color: [IN] 0 = off

1 = red

2 = green

3 = yellow

Return 0 on success

6.1.10 INT WINAPI RF_BEEP

Function: beep

Prototype: int WINAPI rf_beep (unsigned short icdev, unsigned char msec)

Parameter: icdev: [IN] Device ID

msec: [IN] beep time, unit 10 MSEL

Return 0 on success

6.2 DES FUNCTION

6.2.1 INT WINAPI DES_ENCRYPT

Function: DES_Encrypt

```

Prototype: int WINAPI des_encrypt ( unsigned char *pSzOut,  
                                   unsigned char *pSzIn,  
                                   unsigned int   inlen,  
                                   unsigned char *pKey,  
                                   unsigned int   keylen)

```

| | | | |
|------------|---------|-------|--|
| Parameter: | pSzOut: | [OUT] | ciphertext, bytes length equal to plaintext |
| | pSzIn: | [IN] | plaintext |
| | inlen: | [IN] | length of plaintext, integer times of 8 bytes |
| | pKey: | [IN] | encrypt key |
| | keylen: | [IN] | length of key, 8 bytes for single DES, 16 bytes for triple DES |

Return 0 on success

6.2.2 INT WINAPI DES_DECRYPT

Function: DES_Decrypt

```
Prototype: int WINAPI des_decrypt ( unsigned char *pSzOut,  
                                unsigned char *pSzIn,  
                                unsigned int   inlen,  
                                unsigned char *pKey,  
                                unsigned int   keylen)
```

| | | | |
|------------|---------|-------|--|
| Parameter: | pSzOut: | [OUT] | plaintext, bytes length equal to ciphertext |
| | pSzIn: | [IN] | ciphertext |
| | inlen: | [IN] | length of ciphertext, integer times of 8 bytes |
| | pKey: | [IN] | encrypt key |
| | keylen: | [IN] | length of key, 8 bytes for single DES, 16 bytes for triple DES |

Return 0 on success

6.3 ISO14443A FUNCTION

6.3.1 UltraLight

6.3.1.1 INT WINAPI RF_REQUEST

Function: ReqA

```
Prototype: int WINAPI rf_request ( unsigned short icdev,  
                                unsigned char model,  
                                unsigned short *pTagType)
```

| | | | |
|------------|-----------|-------|-------------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | model: | [IN] | REQ MODE |
| | pTagType: | [OUT] | response data, chip type code |

Return 0 on success

```
Annotation: mode = 0x26: REQ_STD
            mode = 0x52: REQ_ALL
```

6.3.1.2 INT WINAPI INT RF_UL_SELECT

Function: Select UltraLight

Prototype: int WINAPI int rf_ul_select (unsigned short icdev,
unsigned char *pSnr,
unsigned char *pLen)

Parameter: icdev: [IN] Device ID
pSnr: [OUT] response data, card unique serial number
pLen: [OUT] length of response data

Return 0 on success

6.3.1.3 INT WINAPI RF_M1_READ

Function: MifareOne read

Prototype: int WINAPI rf_M1_read (unsigned short icdev,
unsigned char block,
unsigned char *pData,
unsigned char *pLen)

Parameter: icdev: [IN] Device ID
block: [IN] block absolute address
pData: [OUT] response data from card
pLen: [OUT] length of response data

Return 0 on success

Annotation: this function is also applicable for UltraLight card. Every page of UltraLight card has 4 bytes. After calling this function, return data of 4 consecutive pages.

6.3.1.4 INT WINAPI INT RF_UL_WRITE

Function: UltraLight Write

Prototype: int WINAPI int rf_ul_write (unsigned short icdev,
unsigned char page,
unsigned char *pData)

Parameter: icdev: [IN] Device ID
page: [IN] UltraLight card page address , 0 ~ 0x0F
pData: [IN] written data, 4 bytes

Return 0 on success

6.3.1.5 INT WINAPI RF_HALT

Function: TYPE_A card HALT

Prototype: int WINAPI rf_halt (unsigned short icdev)

Parameter: icdev: [IN] Device ID

Return 0 on success

6.3.2 Mifare Class

6.3.2.1 INT WINAPI RF_REQUEST

Function: ReqA

```
Prototype: int WINAPI rf_request ( unsigned short icdev,  
                                unsigned char   model,  
                                unsigned short *pTagType)
```

| | | | |
|------------|-----------|-------|-------------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | model: | [IN] | REQ MODE |
| | pTagType: | [OUT] | response data, chip type code |

Return 0 on success

```
Annotation: mode = 0x26: REQ_STD
            mode = 0x52: REQ_ALL
```

6.3.2.2 INT WINAPI RF_ANTICOLL

Function: Mifare card Anticollision

[illegible]

| | | | |
|------------|--------|-------|---|
| Parameter: | icdev: | [IN] | Device ID |
| | bcnt: | [IN] | must be 4 |
| | pSnr: | [OUT] | response data from card, unique serial number |
| | pLen: | [OUT] | length of response data |

Return: return 0 if successful

6.3.2.3 INT WINAPI RF_SELECT

Function: Mifare card Selectting

```

Prototype:  int WINAPI rf_select (unsigned short  icdev,
                                unsigned char    *pSnr,
                                unsigned char    snrLen,
                                unsigned char    *pSize)

```

| | | | |
|------------|---------|-------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | pSnr: | [IN] | card unique serial number |
| | snrLen: | [IN] | length of pSnr |
| | pSize: | [OUT] | response data from card, capacity code |

Return 0 on success

Annotation: card will be on active estate after received this command, only one TYPE_A card on active estate at the same influence range at same time.

6.3.2.4 INT WINAPI RF_M1_AUTHENTICATION2

Function: Mifare_Std Authenticate

[illegible]

| | | | |
|------------|--------|------|------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | model: | [IN] | key validate mode |
| | block: | [IN] | block absolute address |
| | pKey: | [IN] | 6 bytes password |

Return 0 on success

Annotation: model = 0x60: use KeyA
model = 0x61: use KeyB

6.3.2.5 INT WINAPI RF M1 READ

Function: MifareOne Read

```

Prototype:  int WINAPI rf_M1_read ( unsigned short  icdev,
                                   unsigned char    block,
                                   unsigned char    *pData,
                                   unsigned char    *pLen)

```

| | | | |
|------------|--------|-------|-------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |
| | pData: | [OUT] | response data from card |
| | pLen: | [OUT] | length of response data |

Return 0 on success

6.3.2.6 INT WINAPI RF M1 WRITE

Function: Mifare_Std Write

```

Prototype:  int WINAPI rf_M1_write (unsigned short icdev,
                                     unsigned char  block,
                                     unsigned char *pData)

```

| | | | |
|------------|--------|------|------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |
| | pData: | [IN] | written data, 16 bytes |

Return 0 on success

6.3.2.7 INT WINAPI RF M1 INITVAL

Function: Mifare Std card Initialize Value

```

Prototype:  int WINAPI rf_M1_initval ( unsigned short  icdev,
                                     unsigned char   block,
                                     long    value)

```

| | | | |
|------------|---------|------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |
| | pValue: | [IN] | initialize purse value at HEX format, low byte in former |

Return 0 on success

6.3.2.8 INT WINAPI RF_M1_READVAL

Function: Mifare_Std Read Value

```

Prototype:  int WINAPI rf_M1_readval ( unsigned short  icdev,
                                     unsigned char    block,
                                     long *pValue)

```

| | | | |
|------------|---------|-------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |
| | pValue: | [OUT] | response value at HEX format, low byte in former |

Return 0 on success

6.3.2.9 INT WINAPI RF_M1_INCREMENT

Function: Mifare purse increment

```

Prototype:  int WINAPI rf_M1_increment (unsigned short  icdev,
                                         unsigned char  block,
                                         long  value)

```

| | | | |
|------------|--------|------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |
| | value: | [IN] | increase value at HEX format, low byte in former |

Return 0 on success

6.3.2.10 INT WINAPI RF M1 DECREMENT

Function: Mifare purse decrement

```

Prototype:  int WINAPI rf_M1_decrement (unsigned short  icdev,
                                         unsigned char  block,
                                         long  value)

```

| | | | |
|------------|--------|------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |
| | value: | [IN] | decrease value at HEX format, low byte in former |

Return 0 on success

6.3.2.11 INT WINAPI RF M1 RESTORE

Function: Mifare Std Restore

Prototype: int WINAPI rf_M1_restore (unsigned short icdev, unsigned char block)

| | | | |
|------------|--------|------|------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |

Return 0 on success

6.3.2.12 INT WINAPI RF_M1_TRANSFER

Function: Mifare_Std Transfer

Prototype: `int WINAPI rf_M1_transfer (unsigned short icdev, unsigned char block)`

| | | | |
|------------|--------|------|------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block absolute address |

Annotation: mode = 0x26: REQ_STD
mode = 0x52: REQ_ALL

6.3.5.2 INT WINAPI RF_SHC1102_AUTH

Function: SHC1102 card Authenticate

Prototype: int WINAPI rf_Shc1102_Auth (unsigned short icdev, unsigned char *pPassword)

Parameter: icdev: [IN] Device ID
pPassword: [IN] 4 bytes password

Return 0 on success

6.3.5.3 INT WINAPI RF_SHC1102_READ

Function: SHC1102 card read

Prototype: int WINAPI rf_Shc1102_Read (unsigned short icdev,
unsigned char block,
unsigned char *pData,
unsigned char *pLen)

Parameter: icdev: [IN] Device ID
block: [IN] SHC1102 card block address, 0x00 ~ 0x0F
pData: [OUT] response data from card
pLen: [OUT] length of response data

Return 0 on success

6.3.5.4 INT WINAPI RF_SHC1102_WRITE

Function: SHC1102 card write

Prototype: int WINAPI rf_Shc1102_Write (unsigned short icdev,
unsigned char block,
unsigned char *pData)

Parameter: icdev: [IN] Device ID
block: [IN] SHC1102 card block address, 0x00 ~ 0x0F
pData: [IN] written data, 16 bytes

Return 0 on success

pKey: [IN] 8 bytes pass word

Return 0 on success

6.4.2 AT88RF020

6.4.2.1 INT WINAPI RF_ TYPEB_RST

Function: REQ ISO14443B protocol card and set SLOT

Prototype: int WINAPI rf_atqb(unsigned short icdev,
 unsigned char model,
 unsigned char *pData,
 unsigned char *pMsgLg)

Parameter: icdev: [IN] Device ID
 model: [IN] REQ MODE 0 = REQB, 1 = WUPB
 pData: [OUT] response data from card
 pMsgLg: [OUT] length of response data

Return 0 on success

6.4.2.2 INT WINAPI RF_ AT020_CHECK

Function: AT88RF020 card Authenticate

Prototype: int WINAPI rf_at020_check (unsigned short icdev, unsigned char *pKey)

Parameter: icdev: [IN] Device ID
 pKey: [IN] 8 bytes pass word

Return 0 on success

6.4.2.3 INT WINAPI RF_ AT020_COUNT

Function: AT88RF020 card count

Prototype: int WINAPI rf_at020_count(unsigned short icdev, unsigned char *pData)

Parameter: icdev: [IN] Device ID
 pData: [IN] signature, 6 bytes

Return 0 on success

6.4.2.4 INT WINAPI RF_ AT020_READ

Function: AT88RF020 read

Prototype: int WINAPI rf_at020_read (unsigned short icdev, unsigned
 char page,
 unsigned char *pData,
 unsigned char *pMsgLen)

Parameter: icdev: [IN] Device ID
 page: [IN] page address, 0 ~ 31
 pData: [OUT] response data from card
 pMsgLen: [OUT] length of response data

Return 0 on success

6.4.2.5 INT WINAPI RF_AT020_WRITE

Function: AT88RF020 write

```
Prototype: int WINAPI rf_at020_write ( unsigned short icdev,  
                                     unsigned char   page,  
                                     unsigned char   *pData)
```

| | | | |
|------------|--------|------|-----------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | page: | [IN] | page address, 0 ~ 31 |
| | pData: | [IN] | written data, 8 bytes |

Return 0 on success

6.4.2.6 INT WINAPI RF_AT020_LOCK

Function: AT88RF020 LOCK

Prototype: `int WINAPI rf_at020_lock (unsigned short icdev, unsigned char *pData)`

| | | | |
|------------|--------|------|--------------|
| Parameter: | icdev: | [IN] | Device ID |
| | pData: | [IN] | 4 bytes data |

Return 0 on success

6.4.2.7 INT WINAPI RF_AT020_DESELECT

Function: AT88RF020 card Deselect

Prototype: `int WINAPI rf_at020_deselect (unsigned short icdev)`

Parameter: icdev: [IN] Device ID

Return 0 on success

6.4.3 SR176SRIX4K

6.4.3.1 INT WINAPI RF_ST_SELECT

Function: ST card (SR176/SRIX4K) Lock

Prototype: `int WINAPI rf_st_select (unsigned short icdev, unsigned char *pChip_ID)`

| | | | |
|------------|-----------|------|---|
| Parameter: | icdev: | [IN] | Device ID |
| | pChip_ID: | [IN] | response data from card, 1 byte ID code |

Return 0 on success

6.4.3.2 INT WINAPI INT_RF_SR176_READBLOCK

Function: SR176 Read

[illegible]

| | | | |
|------------|--------|-------|-------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | block: | [IN] | block address |
| | pData: | [OUT] | response data from card |
| | pLen: | [OUT] | length of response data |

Return 0 on success

6.4.3.3 INT WINAPI INT_RF_SR176_WRITEBLOCK

Function: SR176 Write

Prototype: int WINAPI int rf_sr176_writeblock (unsigned short icdev,
 unsigned char block,
 unsigned char *pData)

Parameter: icdev: [IN] Device ID
 block: [IN] block address
 pData: [IN] written data, 2 bytes

Return 0 on success

6.4.3.4 INT WINAPI INT_RF_SR176_PROTECTBLOCK

Function: SR176 Lock

Prototype: int WINAPI int rf_sr176_protectblock (unsigned short icdev, unsigned char lockreg)

Parameter: icdev: [IN] Device ID
 lockreg: [IN] LOCKREG

Return 0 on success

Annotation: SR17 6has 16 blocks, every lockreg controls 2 blocks

| lockreg | BLOCK | bit_setting | |
|---------|---------|----------------|--------------------|
| b7 | 14 & 15 | 0:Write Enable | 1:Block set as ROM |
| b6 | 12 & 13 | 0:Write Enable | 1:Block set as ROM |
| b5 | 10 & 11 | 0:Write Enable | 1:Block set as ROM |
| b4 | 8 & 9 | 0:Write Enable | 1:Block set as ROM |
| b3 | 6 & 7 | 0:Write Enable | 1:Block set as ROM |
| b2 | 4 & 5 | 0:Write Enable | 1:Block set as ROM |
| b1 | 2 & 3 | 0:Write Enable | 1:Block set as ROM |
| b0 | 0 & 1 | 0:Write Enable | 1:Block set as ROM |

6.4.3.5 INT WINAPI INT_RF_SRIX4K_GETUID

Function: SRIX4K Get UID

Prototype: int WINAPI int rf_srix4k_getuid (unsigned short icdev,
 unsigned char *pUid,
 unsigned char *pLen)

Parameter: icdev: [IN] Device ID
 pUid: [OUT] response data from card, UID
 pLen: [OUT] length of response data

Return 0 on success

6.4.3.6 INT WINAPI INT_RF_SRIX4K_READBLOCK

Function: SRIX4K Read

Prototype: int WINAPI int rf_srix4k_readblock (unsigned short icdev,
 unsigned char block,
 unsigned char *pData,
 unsigned char *pLen)

Parameter: icdev: [IN] Device ID
 block: [IN] block address
 pData: [OUT] response data from card
 pLen: [OUT] length of response data

Return 0 on success

6.4.3.7 INT WINAPI INT_RF_SRIX4K_WRITEBLOCK

Function: SRIX4K Write

Prototype: int WINAPI int_rf_srix4k_writeblock(unsigned short icdev,
 unsigned char block,
 unsigned char *pData)

Parameter: icdev: [IN] Device ID
 block: [IN] block address
 pData: [IN] written data, 4bytes

Return 0 on success

6.4.3.8 INT WINAPI INT_RF_SRIX4K_PROTECTBLOCK

Function: SRIX4K Lock

Prototype: int WINAPI int_rf_srix4k_protectblock(unsigned short icdev, unsigned char lockreg)

Parameter: icdev: [IN] Device ID
 Lockreg: [IN] LOCKREG

Return 0 on success

Annotation: 7~15 blocks of SRIX4K card can be written protect

| lockreg | BLOCK | bit_setting | |
|---------|-------|----------------|--------------------|
| b7 | 15 | 1:Write Enable | 0:Block set as ROM |
| b6 | 14 | 1:Write Enable | 0:Block set as ROM |
| b5 | 13 | 1:Write Enable | 0:Block set as ROM |
| b4 | 12 | 1:Write Enable | 0:Block set as ROM |
| b3 | 11 | 1:Write Enable | 0:Block set as ROM |
| b2 | 10 | 1:Write Enable | 0:Block set as ROM |
| b1 | 9 | 1:Write Enable | 0:Block set as ROM |
| b0 | 7 & 8 | 1:Write Enable | 0:Block set as ROM |

6.4.3.9 INT WINAPI RF_ST_COMPLETION

Function: ST Desactivated

Prototype: int WINAPI rf_st_completion (unsigned short icdev)

Parameter: icdev: [IN] Device ID

Return 0 on success

6.4.4 TYPE_B SmartCard

6.4.4.1 INT WINAPI RF_TYPEB_RST

Function: Req ISO14443B-4 protocol Smart card and Reset

Prototype: int WINAPI rf_atqb (unsigned short icdev,
 unsigned char model,
 unsigned char *pData,
 unsigned char *pMsgLg)

Parameter: icdev: [IN] Device ID
 model: [IN] REQ MODE 0 = REQB, 1 = WUPB
 pData: [OUT] response data from card
 pMsgLg: [OUT] length of response data

Return 0 on success

6.4.4.2 INT WINAPI RF_COS_COMMAND

Prototype: int WINAPI rf_cos_command (unsigned short icdev,
 unsigned char *pCommand,
 unsigned char cmdLen,
 unsigned char *pData,
 unsigned char *pMsgLg)

Parameter: icdev: [IN] Device ID
 pCommand: [IN] cos command
 cmdLen: [IN] length of cos command
 pData: [OUT] response data from card, including SW1, SW2
 pMsgLg: [OUT] length of response data

Return 0 on success

6.4.4.3 INT WINAPI RF_CL_DESELECT

Function: ISO14443B card Deselect

Prototype: int WINAPI rf_cl_deselect (unsigned short icdev)

Parameter: icdev: [IN] Device ID

Return 0 on success

6.5 ISO15693 FUNCTION

6.5.1 INT WINAPI ISO15693_INVENTORY

Function: ISO15693_Inventory (single card)

[illegible]

| | | | |
|------------|--------|-------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | pData: | [OUT] | response data from tag, 1 byte DSFID + 8 bytes UID |
| | pLen: | [OUT] | length of response data |

Return 0 on success

6.5.2 INT WINAPI ISO15693 INVENTORYS

Function: ISO15693_Inventory (several cards)

[illegible]

| | | | |
|------------|--------|-------|---|
| Parameter: | icdev: | [IN] | Device ID |
| | pData: | [OUT] | response data from tag, every 9 bytes is a team, the structure of every team is: 1byte DSFID + 8 bytes UID |
| | pLen: | [OUT] | length of response data |

Return 0 on success

6.5.3 INT WINAPI ISO15693 GET SYSTEM INFORMATION

Function: ISO15693_Get_System_Information

[illegible]

| | | | |
|------------|--------|-------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | model: | [IN] | bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag |
| | pUID: | [IN] | 8 bytes UID |
| | pData: | [OUT] | response data from tag |
| | pLen: | [OUT] | length of response data |

Return 0 on success

Annotation: If set `Select_flag`, only the cards on Selected state respond this command
If set `Address_flag`, only the cards that the UID are congruous will respond this command
Clear `Option_flag = 0`

6.5.4 INT WINAPI ISO15693_SELECT

Function: ISO15693_Select

Prototype: int WINAPI ISO15693_Select (unsigned short icdev, unsigned char *pUID)

Parameter: icdev: [IN] Device ID
 pUID: [IN] 8 bytes UID

Return 0 on success

6.5.5 INT WINAPI ISO15693_RESET_TO_READY

Function: ISO15693_Reset_To_Ready

Prototype: int WINAPI ISO15693_Reset_To_Ready (unsigned short icdev,
 unsigned char model,
 unsigned char *pUID)

Parameter: icdev: [IN] Device ID
 model: [IN] bit0=Select_flag, bit1=Address_flag, bit2=Option_flag
 pUID: [IN] 8 bytes UID

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
 If set Address_flag, only the cards that the UID are congruous will respond this command
 Clear Option_flag = 0

6.5.6 INT WINAPI ISO15693_STAY_QUIET

Function: ISO15693_Stay_Quiet

Prototype: int WINAPI ISO15693_Stay_Quiet (unsigned short icdev, unsigned char *pUID)

Parameter: icdev: [IN] Device ID
 pUID: [IN] 8 bytes UID

Return 0 on success

6.5.7 INT WINAPI ISO15693_GET_BLOCK_SECURITY

Function: ISO15693_Get_Block_Security

Prototype: int WINAPI ISO15693_Get_Block_Security (unsigned short icdev,
 unsigned char model,
 unsigned char *pUID,
 unsigned char block,
 unsigned char number,
 unsigned char *pData,
 unsigned char *pLen)

Parameter: icdev: [IN] Device ID
 model: [IN] bit0=Select_flag, bit1=Address_flag, bit2=Option_flag
 pUID: [IN] 8 bytes UID
 block: [IN] block address
 number: [IN] the number of block to be read, < 0x40
 pData: [OUT] response data from tag
 pLen: [OUT] length of response data

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
 If set Address_flag, only the cards that the UID are congruous will respond this command
 Clear Option_flag = 0

6.5.8 INT WINAPI ISO15693_READ

Function: ISO15693_Read

Prototype: int WINAPI ISO15693_Read (unsigned short icdev,
 unsigned char model,
 unsigned char *pUID,
 unsigned char block,
 unsigned char number,
 unsigned char *pData,
 unsigned char *pLen);

Parameter: icdev: [IN] Device ID
 model: [IN] bit0=Select_flag, bit1=Address_flag, bit2=Option_flag
 pUID: [IN] 8 bytes UID
 block: [IN] block address
 number: [IN] the number of block to be read, < 0x40
 pData: [OUT] response data from tag
 pLen: [OUT] length of response data

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
 If set Address_flag, only the cards that the UID are congruous will respond this command
 Clear Option_flag = 0

6.5.9 INT WINAPI ISO15693_WRITE

Function: ISO15693_Write

Prototype: int WINAPI ISO15693_Write (unsigned short icdev,
 unsigned char model,
 unsigned char *pUID,
 unsigned char block,
 unsigned char *pData)

Parameter: icdev: [IN] Device ID
 model: [IN] bit0=Select_flag, bit1=Address_flag, bit2=Option_flag
 pUID: [IN] 8 bytes UID
 block: [IN] block address
 pData: [IN] written data, 4 bytes

Return 0 on success

Explanation: If set Select_flag, only the cards on Selected state respond this command
 If set Address_flag, only the cards that the UID are congruous will respond this command

If write TI card, set Option_flag,
If write I.CODE SLI card, clear Option_flag

6.5.10 INT WINAPI ISO15693_LOCK_BLOCK

Function: ISO15693_Lock_Block

[illegible]

| | | | |
|------------|--------|------|---|
| Parameter: | icdev: | [IN] | Device ID |
| | model: | [IN] | bit0=Select_flag, bit1=Address_flag, bit2=Option_flag |
| | pUID: | [IN] | 8 bytes UID |
| | block: | [IN] | block address |

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
If set Address_flag, only the cards that the UID are congruous will respond this command
If write TI card, set Option_flag,
If write I.CODE SLI card, clear Option_flag

6.5.11 INT WINAPI ISO15693_WRITE_AFI

Function: ISO15693_Write_AFI

[illegible]

| | | | |
|------------|--------|------|--|
| Parameter: | icdev: | [IN] | Device ID |
| | model: | [IN] | bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag |
| | pUID: | [IN] | 8 bytes UID |
| | AFI: | [IN] | AFI to be written |

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
If set Address_flag, only the cards that the UID are congruous will respond this command
If write TI card, set Option_flag,
If write I.CODE SLI card, clear Option_flag

6.5.12 INT WINAPI ISO15693_LOCK_AFI

Function: ISO15693 Lock AFI

[illegible]

| | | | |
|------------|--------|------|---|
| Parameter: | icdev: | [IN] | Device ID |
| | model: | [IN] | bit0=Select flag, bit1=Address flag, bit2=Option flag |

pUID: [IN] 8 bytes UID

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
 If set Address_flag, only the cards that the UID are congruous will respond this command
 If write TI card, set Option_flag,
 If write I.CODE SLI card, clear Option_flag

6.5.13 INT WINAPI ISO15693_WRITE_DSFD

Function: ISO15693_Write_DSFD

Prototype: int WINAPI ISO15693_Write_DSFD (unsigned short icdev,
 unsigned char model,
 unsigned char *UID,
 unsigned char DSFD)

Parameter: icdev: [IN] Device ID
 model: [IN] bit0=Select_flag, bit1=Address_flag, bit2=Option_flag
 pUID: [IN] 8 bytes UID
 DSFD: [IN] DSFD to be written

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
 If set Address_flag, only the cards that the UID are congruous will respond this command
 If write TI card, set Option_flag,
 If write I.CODE SLI card, clear Option_flag

6.5.14 INT WINAPI ISO15693_LOCK_DSFD

Function: ISO15693_Lock_DSFD

Prototype: int WINAPI ISO15693_Lock_DSFD (unsigned short icdev,
 unsigned char model,
 unsigned char *pUID)

Parameter: icdev: [IN] Device ID
 model: [IN] bit0=Select_flag, bit1=Address_flag, bit2=Option_flag
 pUID: [IN] 8 bytes UID

Return 0 on success

Annotation: If set Select_flag, only the cards on Selected state respond this command
 If set Address_flag, only the cards that the UID are congruous will respond this command
 If write TI card, set Option_flag,
 If write I.CODE SLI card, clear Option_flag

6.6 Function of Infineon Electric Tag

6.6.1 INT WINAPI SRF55VP_READ

Function: SRF55XP Read a PAGE

```
Prototype: int WINAPI Srf55vp_Read ( unsigned short   icdev,  
                                   unsigned char    *pUID,  
                                   unsigned char     page,  
                                   unsigned char    *pData,  
                                   unsigned char    *pLen)
```

| | | | |
|------------|--------|-------|-------------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | pUID: | [IN] | 8 bytes UID |
| | page: | [IN] | address |
| | pData: | [OUT] | response data from tag |
| | pLen: | [OUT] | length of response data |

Return 0 on success

6.6.2 INT WINAPI SRF55VP WRITEBYTE

Function: SRF55XP Write 1BYTE

```
Prototype: int WINAPI Srf55vp_WriteByte ( unsigned short icdev,  
                                         unsigned char *pUID,  
                                         unsigned char page,  
                                         unsigned char byteaddr,  
                                         unsigned char data)
```

| | | | |
|------------|-----------|------|---|
| Parameter: | icdev: | [IN] | Device ID |
| | pUID: | [IN] | 8 bytes UID |
| | page: | [IN] | address |
| | byteaddr: | [IN] | write the bytes to excursion address of the PAGE, 0 ~ 8 |
| | data: | [IN] | written data |

Return 0 on success

6.6.3 INT WINAPI SRF55VP WRITE

Function: SRF55XP Write a page

[illegible]

| | | | |
|------------|--------|------|-----------------------|
| Parameter: | icdev: | [IN] | Device ID |
| | pUID: | [IN] | 8 bytes UID |
| | page: | [IN] | address |
| | pData: | [IN] | written data, 8 bytes |

Return 0 on success

6.6.4 INT WINAPI SRF55VP_WRITE_REREAD

Function: SRF55XXP write PAGE and Return to the real data of this PAGE

Prototype: int WINAPI Srf55vp_Write_Reread(unsigned short icdev,
 unsigned char *pUID,
 unsigned char page,
 unsigned char *pWdata,
 unsigned char *pRdata,
 unsigned char *pLen)

Parameter: icdev: [IN] Device ID
 pUID: [IN] 8 bytes UID
 page: [IN] address
 pWdata: [IN] written data, 8bytes
 pWdata: [OUT] response data from tag
 pLen: [OUT] length of response data

Return 0 on success

6.7 PASS THROUGH FUNCTION

6.7.1 INT WIN API RF_TRANSCEIVE1

Function: Send parameters to Tag and receive response data

Prototype: int WINAPI rf_transceive1(unsigned short icdev,
 unsigned char *pTxData,
 unsigned char sendLen,
 unsigned char *pRxData,
 unsigned char *pMsgLg)

Parameter: icdev: [IN] Communication device identifier
 pTxData: [IN] parameter sent to tag, without CRC bytes
 CRC bytes is auto managed by reader
 sendLen: [IN] length of parameter
 pRxData: [OUT] response data from tag
 pMsgLg: [OUT] length of response data

Return 0 on success