YHY638F

13.56MHz RFID Reader/Writer User Manual

Ordering Information:

Item No.	Description
YHY638FU	13.56MHz Multi Protocol Reader/Writer USB Interface
YHY638FR	13.56MHz Multi Protocol Reader/Writer RS232 Interface

1 Supported cards

ISO 14443 TypeA: Mifare_One(S50&S70)/UltraLight/Mifare_ProX
ISO 14443 TypeB: AT88RF020/SR176/SRIX4K
ISO 15693: Tag_it（2k）, HF-1/I.CODE SLI and more

2 Special Features

- Read and write contactless smart cards
- Frequency: 13.56 MHz.
- Typical time to read and write cards：<100ms
- Communications Interface: USB or RS232, baud rate 9600 ～ 115200 bps
- Power supply ：DC 5V
- Two LED indicators (software controlled)
- Buzzer alarm (software controlled)
- Mechanic and environmental characteristics:
  - Size: 110 ×81× 26 (mm)

- Cable length: 1.5m
- Color: Black
- Operating temperature: 0 ~60 ℃
- Storage temperature: －25 ~80℃
- Relative humidity: up to 90%
- Weight:**90g (YHY638FU)**

  160g (YHY638FR)

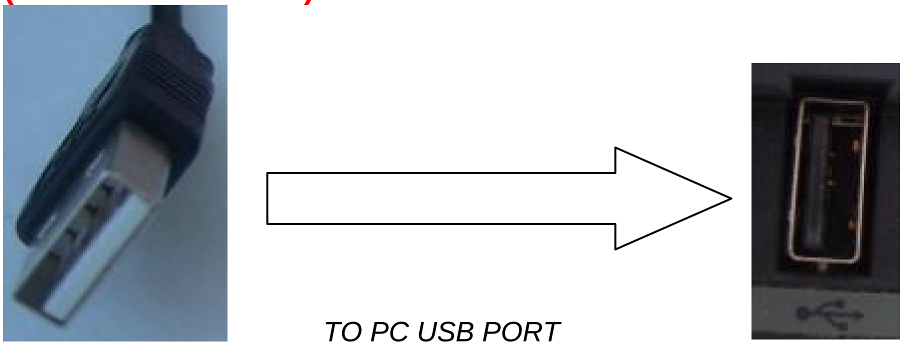
3 Connecting

Connect the YHY638F to the USB or RS232 port of PC, after power on the RED led and GREEN led will flash one time, then the GREEN led will light on again, next the buzzer will beep twice, it means that the reader is ready now.

1) For YHY638FU item

Connect one of the A type USB connector to the rear side of the YHY638FU, another USB connector connect to the host USB port.

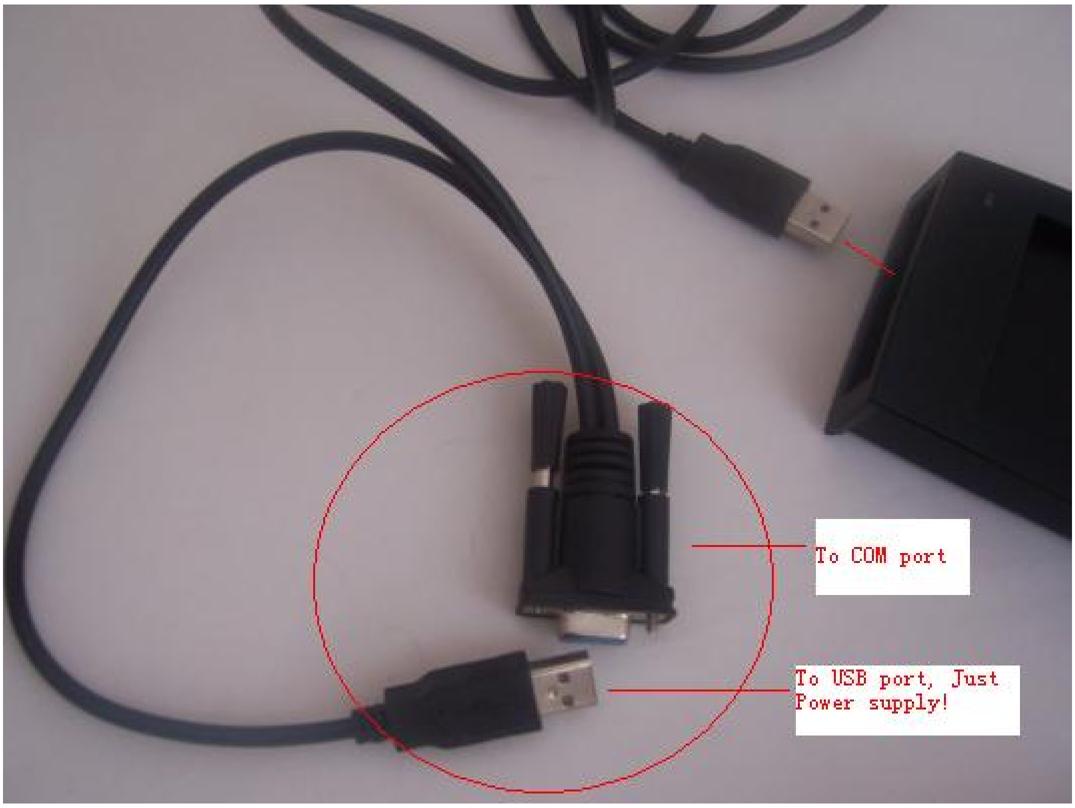
(Item No:YHY638FU)



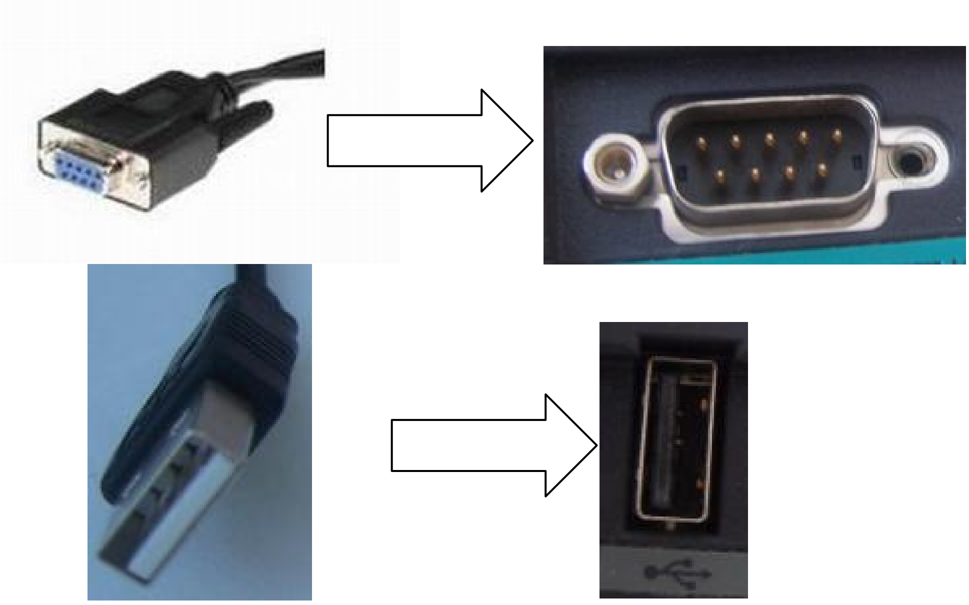
TO PC USB PORT

2) For YHY638FR item

Connect one of the A type USB connector to the rear side of the YHY638FR, another USB connector which near the serial connector connect to the host USB port for power the device, and connect the 9pin COM port to HOST COM port.



(Item No:YHY638FR)



4 Electrical Characteristics

4.1 Operating Condition Range

Relative humidity:up to 90%

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Tamb	Ambient Temperature		0	25	60	℃
VDD	DC Supply Voltage	DVSS = 0V	4.5	5	5.5	V

Table 1 - Operating Condition Range

4.2 Current Consumption

Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
IDVDD	Supply Current	Reading started but no cards in the reader range		90		mA
IDVDD		Reading started, 1 card in the reader range		95		mA

Table 2 - Current Consumption

4.3 Operating Distance

Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
OD	14443 Type A S50 tag Operating Distance,	Measured from the reader bottom		0-70		mm
OD	14443 Type B tag Operating Distance	Measured from the reader bottom		0-45		mm
OD	15693 tag Operating Distance,	Measured from the reader bottom		0-90		mm

Table 3 - Operating Distance

4.4 COM Interface Characteristics

Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**

USB	baud rate			**115200**		
RS232	baud rate		9600	**115200**	115200	baud

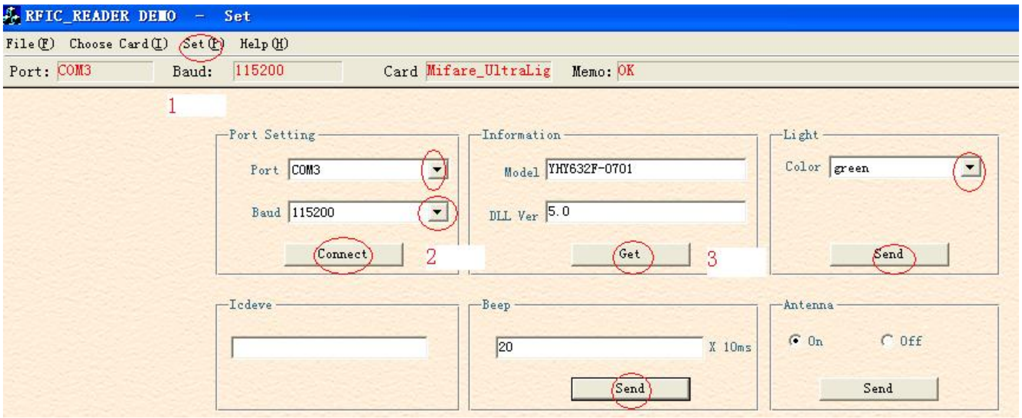
Table 4 - Serial Interface

5. DEMO

This software rfidxray.exe run on Win32 system.

5.1 COM setup

First run RFIDXray.exe, then connect the reader to PC COM port.Choose the correct COM number, click [Connect] button to connect the Reader to PC.Baud set to 115200. Click [Read] the product information button, you can check the specific type of the Reader and the supported cards.



Port: COM3 Baud: 115200 Card Mifare_UltraLig Memo: OK

Port COM3

Baud 115200

Model YHY632F-0701

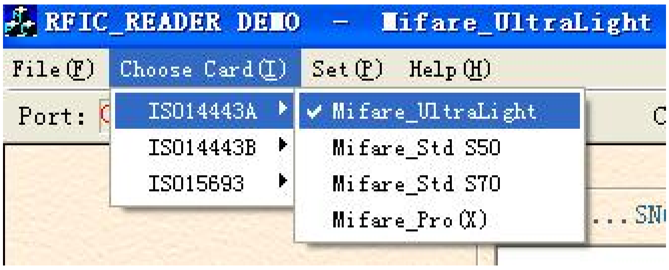
DLL Ver 5.0

Color green

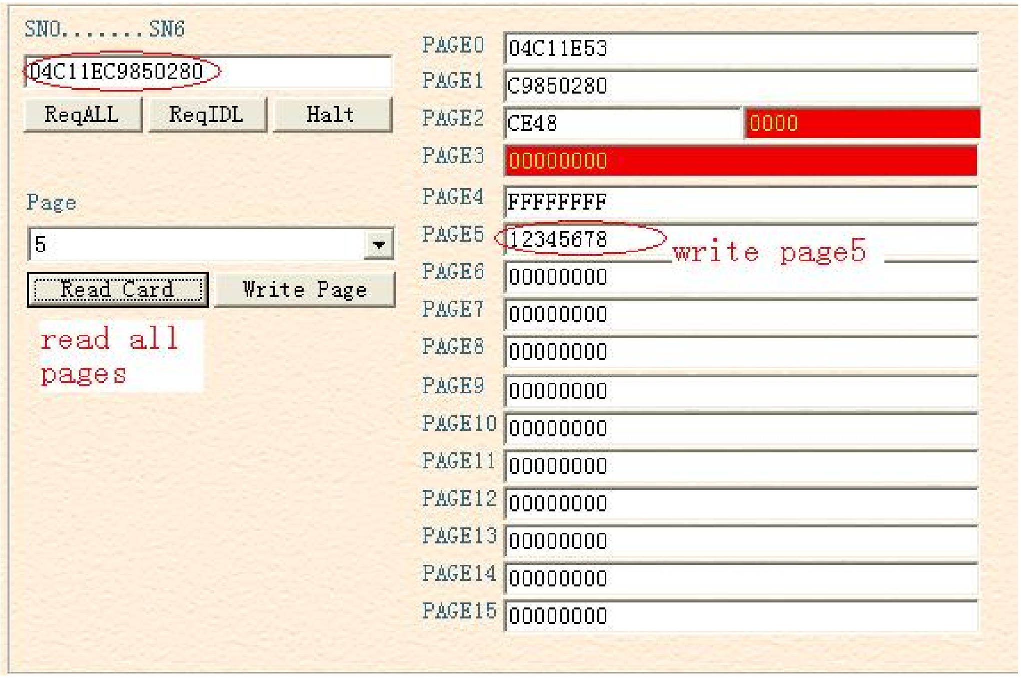
20 X 10ms

● On ○ Off

5.2 UltraLight



Click the [ReqALL] button to obtain the card Serial Number. Choose the corresponding address to read/write the card.



SN0.......SN6: 04C11EC9850280

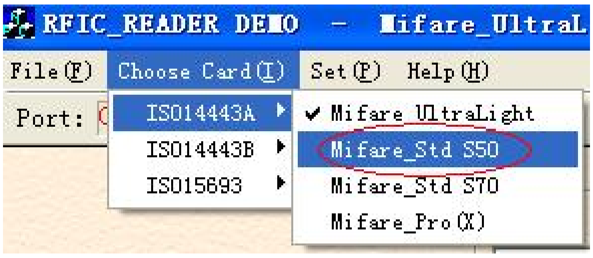
Page: 5

PAGE0: 04C11E53
PAGE1: C9850280
PAGE2: CE48 0000
PAGE3: 00000000
PAGE4: FFFFFFFF
PAGE5: 12345678
PAGE6: 00000000
PAGE7: 00000000
PAGE8: 00000000
PAGE9: 00000000
PAGE10: 00000000
PAGE11: 00000000
PAGE12: 00000000
PAGE13: 00000000
PAGE14: 00000000
PAGE15: 00000000

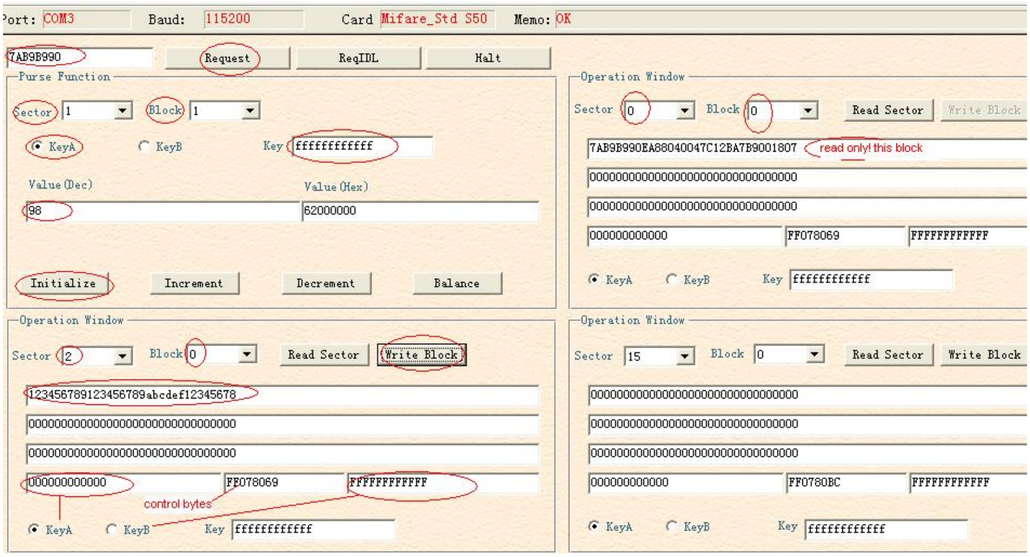
read all pages

write page5

5.3 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.



7AB9B990

Purse Function: Sector 1, Block 1, ● KeyA, Key ffffffffffff, Value (Dec) 98, Value (Hex) 62000000

Operation Window: Sector 0, Block 0

7AB9B990EA88040047C12BA7B9001807 read only! this block
00000000000000000000000000000000
00000000000000000000000000000000
000000000000 FF078069 FFFFFFFFFFFF
● KeyA ○ KeyB Key ffffffffffff

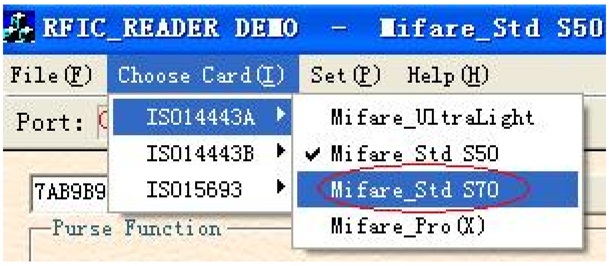
Operation Window: Sector 2, Block 0

123456789123456789abcdef12345678
00000000000000000000000000000000
00000000000000000000000000000000
000000000000 FF078069 FFFFFFFFFFFF
control bytes
● KeyA ○ KeyB Key ffffffffffff

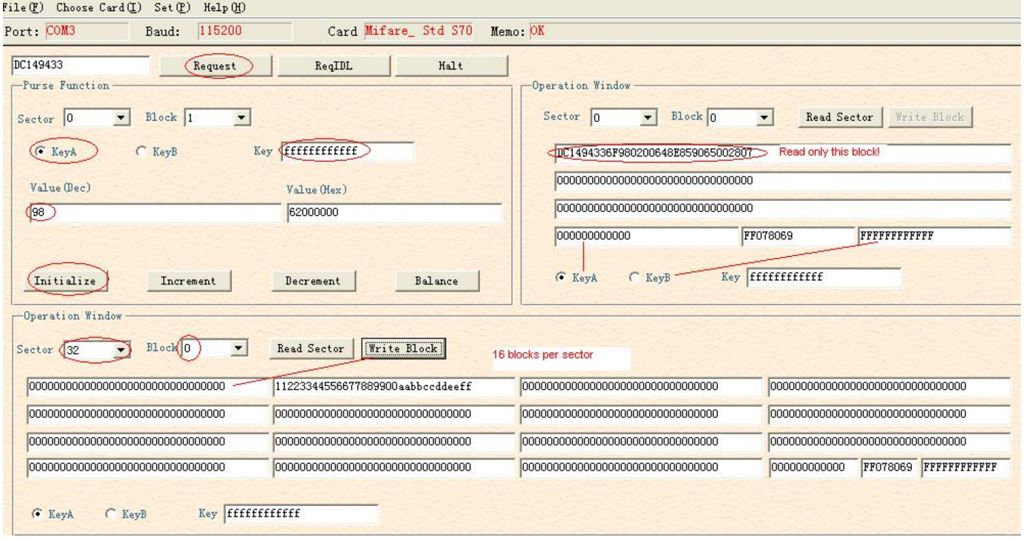
Operation Window: Sector 15, Block 0

00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
000000000000 FF0780BC FFFFFFFFFFFF
● KeyA ○ KeyB Key ffffffffffff

5.4 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.



DC149433

Purse Function: Sector 0, Block 1, ● KeyA, Key ffffffffffff, Value (Dec) 98, Value (Hex) 62000000

Operation Window: Sector 0, Block 0

DC1494336F980200648E859065002807 Read only this block!
00000000000000000000000000000000
00000000000000000000000000000000
000000000000 FF078069 FFFFFFFFFFFF
● KeyA ○ KeyB Key ffffffffffff

Operation Window: Sector 32, Block 0

16 blocks per sector

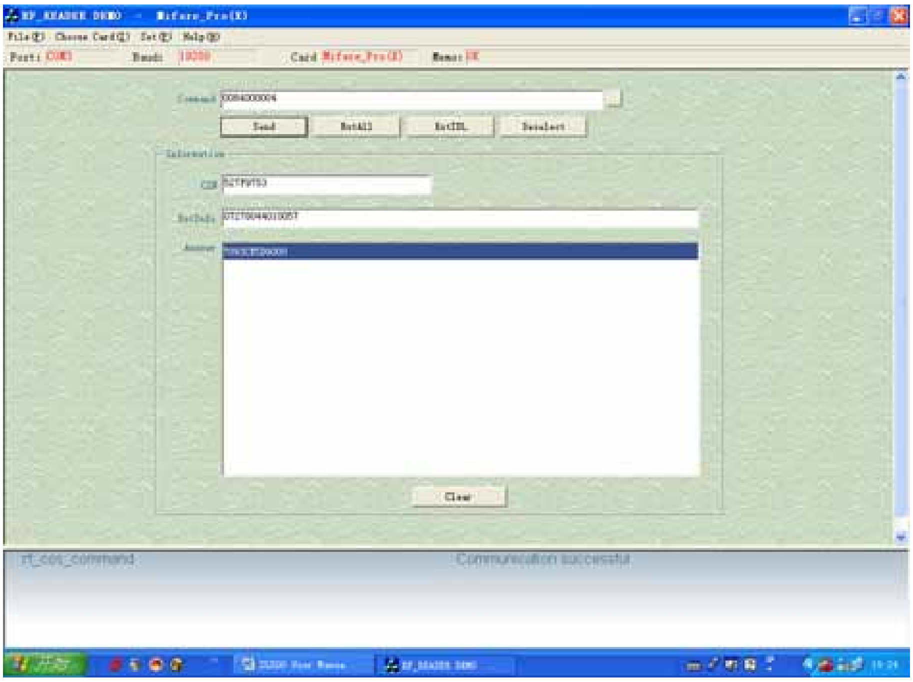
00000000000000000000000000000000 112233445566778899 00aabbccddeeff 00000000000000000000000000000000 00000000000000000000000000000000
00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000
00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000
00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 000000000000 FF078069 FFFFFFFFFFFF

● KeyA ○ KeyB Key ffffffffffff

5.5 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.


RF_READER DEMO - Mifare_Pro(X)
File(F) Choose Card(C) Set(S) Help(H)
Send
RstAll
RstIDL
Deselect
Information
Clear

Port: COM1 Baud: 19200 Card Mifare_Pro(X) Memo: OK

Command 0084000004

CSN 52TF9T53

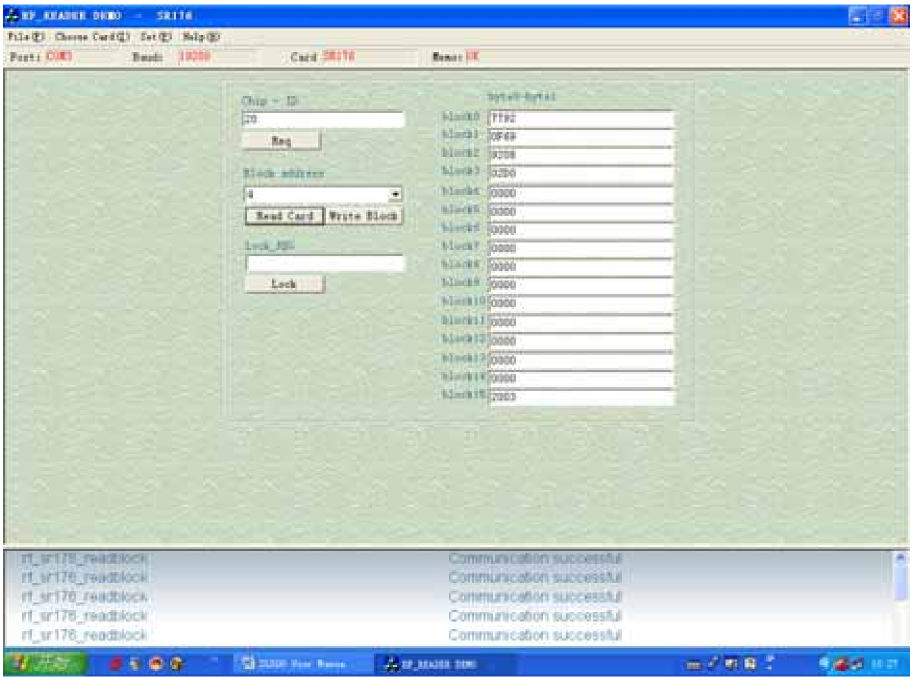
ResData 0?2T0044010057

Answer 7?6?C?T?0000?

rf_cos_command Communication successful

5.6 SR176

Click [Req] button to obtain the ID number of the card.
Then you can read, write and lock blocks of the card.



Port: COM1 Baud: 19200 Card SR176 Memo: OK

Chip - ID 20

Block address 4

Lock_REG

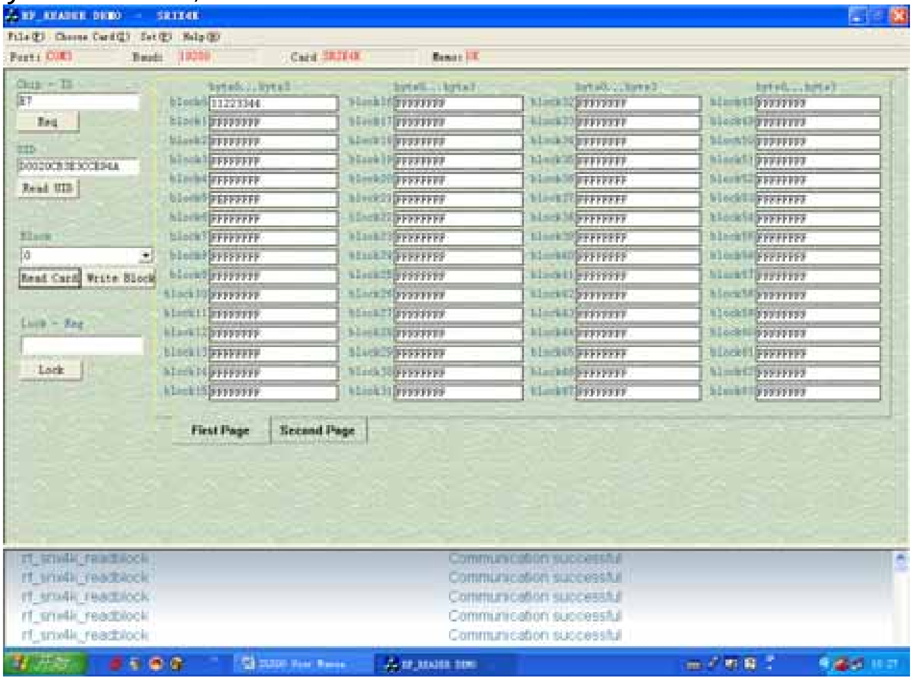
	byte0-byte1
block0	FF82
block1	0F69
block2	9C08
block3	3250
block4	0000
block5	0000
block6	0000
block7	0000
block8	0000
block9	0000
block10	0000
block11	0000
block12	0000
block13	0000
block14	0000
block15	2003

```
rf_sr176_readblock          Communication successful
rf_sr176_readblock          Communication successful
rf_sr176_readblock          Communication successful
rf_sr176_readblock          Communication successful
rf_sr176_readblock          Communication successful
```

5.7 SRIX4K

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.



Port: COM1 Baud: 19200 Card SRIX4K Memo: OK

Chip - ID: E7

UID: D00200CB3E3CCE94A

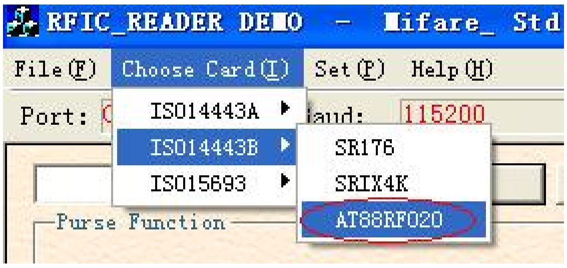
Block: 0

Lock - Reg:

	byte0...byte3		byte0...byte3		byte0...byte3		byte0...byte3
block0	11223344	block16	FFFFFFFF	block32	FFFFFFFF	block48	FFFFFFFF
block1	FFFFFFFF	block17	FFFFFFFF	block33	FFFFFFFF	block49	FFFFFFFF
block2	FFFFFFFF	block18	FFFFFFFF	block34	FFFFFFFF	block50	FFFFFFFF
block3	FFFFFFFF	block19	FFFFFFFF	block35	FFFFFFFF	block51	FFFFFFFF
block4	FFFFFFFF	block20	FFFFFFFF	block36	FFFFFFFF	block52	FFFFFFFF
block5	FFFFFFFF	block21	FFFFFFFF	block37	FFFFFFFF	block53	FFFFFFFF
block6	FFFFFFFF	block22	FFFFFFFF	block38	FFFFFFFF	block54	FFFFFFFF
block7	FFFFFFFF	block23	FFFFFFFF	block39	FFFFFFFF	block55	FFFFFFFF
block8	FFFFFFFF	block24	FFFFFFFF	block40	FFFFFFFF	block56	FFFFFFFF
block9	FFFFFFFF	block25	FFFFFFFF	block41	FFFFFFFF	block57	FFFFFFFF
block10	FFFFFFFF	block26	FFFFFFFF	block42	FFFFFFFF	block58	FFFFFFFF
block11	FFFFFFFF	block27	FFFFFFFF	block43	FFFFFFFF	block59	FFFFFFFF
block12	FFFFFFFF	block28	FFFFFFFF	block44	FFFFFFFF	block60	FFFFFFFF
block13	FFFFFFFF	block29	FFFFFFFF	block45	FFFFFFFF	block61	FFFFFFFF
block14	FFFFFFFF	block30	FFFFFFFF	block46	FFFFFFFF	block62	FFFFFFFF
block15	FFFFFFFF	block31	FFFFFFFF	block47	FFFFFFFF	block63	FFFFFFFF

```
rf_srix4k_readblock                    Communication successful
rf_srix4k_readblock                    Communication successful
rf_srix4k_readblock                    Communication successful
rf_srix4k_readblock                    Communication successful
rf_srix4k_readblock                    Communication successful
```

5.8 AT88RF020



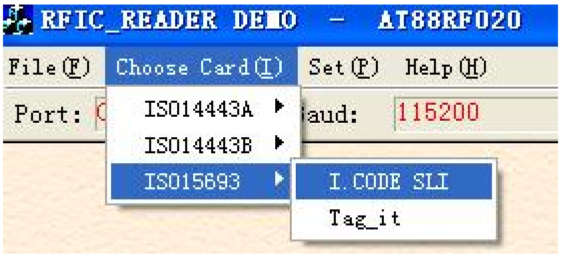
Port: C

Baud: 115200

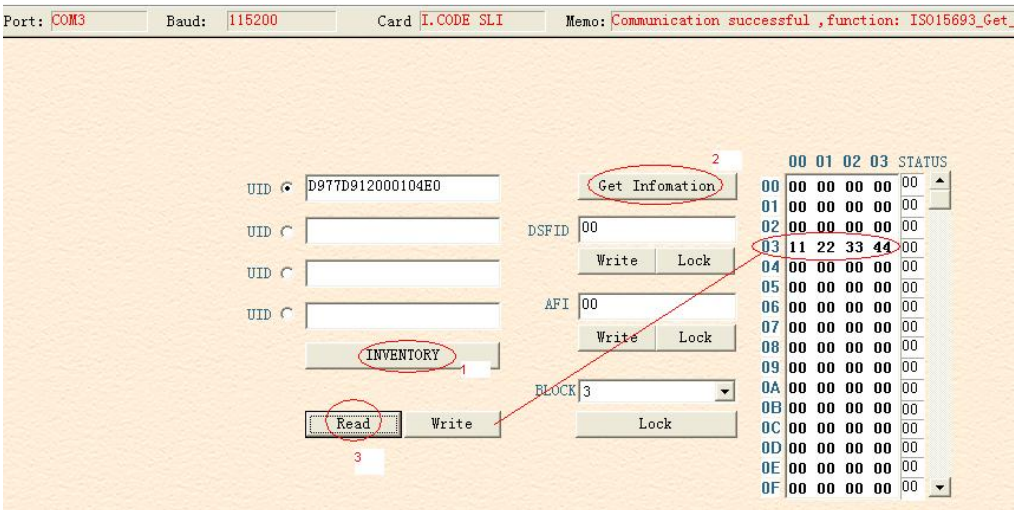
Click [ReqB] button to obtain the serial number of the card.

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

5.9 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.



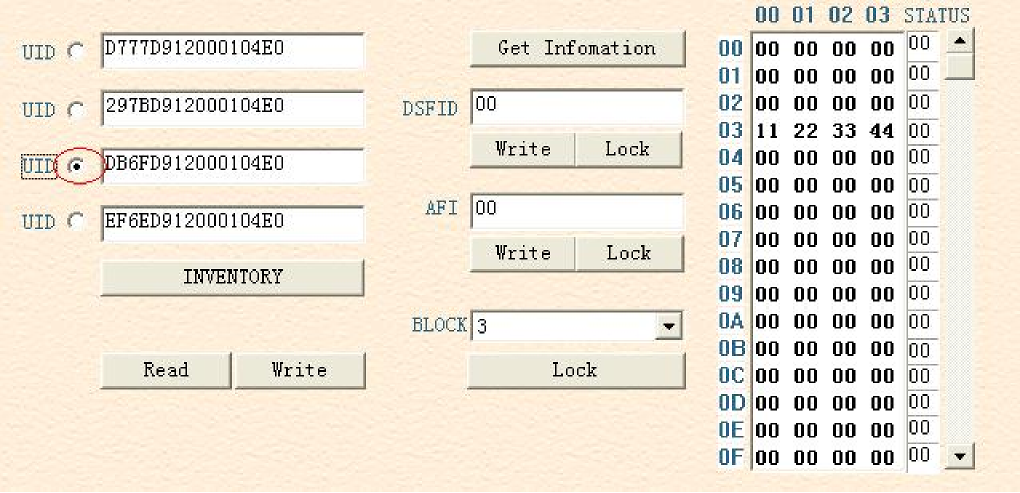
Port: COM3 Baud: 115200 Card I.CODE SLI Memo: Communication successful ,function: ISO15693_Get_

UID ● D977D912000104E0
UID ○
UID ○
UID ○

DSFID 00
AFI 00
BLOCK 3

	00	01	02	03	STATUS
00	00	00	00	00	00
01	00	00	00	00	00
02	00	00	00	00	00
03	11	22	33	44	00
04	00	00	00	00	00
05	00	00	00	00	00
06	00	00	00	00	00
07	00	00	00	00	00
08	00	00	00	00	00
09	00	00	00	00	00
0A	00	00	00	00	00
0B	00	00	00	00	00
0C	00	00	00	00	00
0D	00	00	00	00	00
0E	00	00	00	00	00
0F	00	00	00	00	00

This demo can display 4 label’s id only.

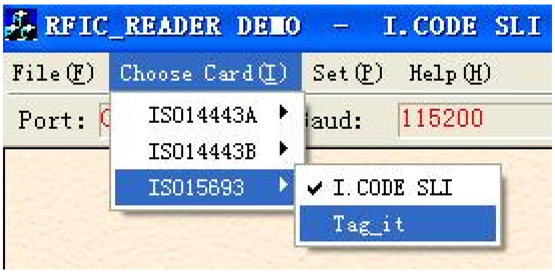


UID ○ D777D912000104E0
UID ○ 297BD912000104E0
UID ● DB6FD912000104E0
UID ○ EF6ED912000104E0

DSFID 00
AFI 00
BLOCK 3

	00	01	02	03	STATUS
00	00	00	00	00	00
01	00	00	00	00	00
02	00	00	00	00	00
03	11	22	33	44	00
04	00	00	00	00	00
05	00	00	00	00	00
06	00	00	00	00	00
07	00	00	00	00	00
08	00	00	00	00	00
09	00	00	00	00	00
0A	00	00	00	00	00
0B	00	00	00	00	00
0C	00	00	00	00	00
0D	00	00	00	00	00
0E	00	00	00	00	00
0F	00	00	00	00	00

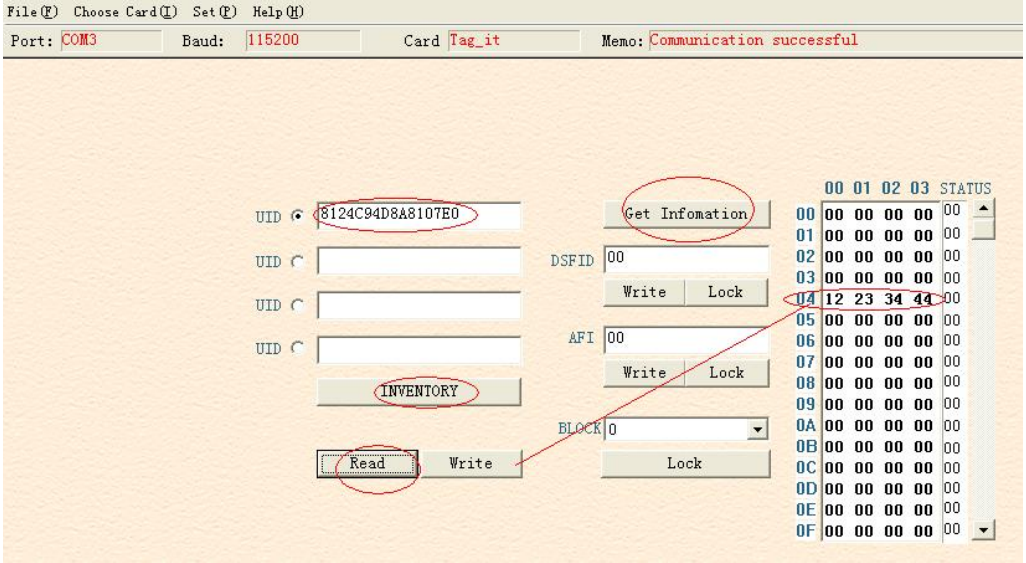
5.10 Tag_IT



Port: C

Baud: 115200

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.



Port: COM3

Baud: 115200

Card Tag_it

Memo: Communication successful

UID ● 8124C94D8A8107E0

UID ○

UID ○

UID ○

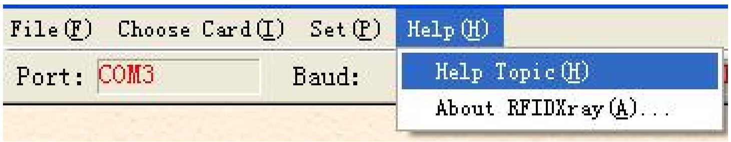
DSFID 00

AFI 00

BLOCK 0

	00	01	02	03	STATUS
00	00	00	00	00	00
01	00	00	00	00	00
02	00	00	00	00	00
03	00	00	00	00	00
04	12	23	34	44	00
05	00	00	00	00	00
06	00	00	00	00	00
07	00	00	00	00	00
08	00	00	00	00	00
09	00	00	00	00	00
0A	00	00	00	00	00
0B	00	00	00	00	00
0C	00	00	00	00	00
0D	00	00	00	00	00
0E	00	00	00	00	00
0F	00	00	00	00	00

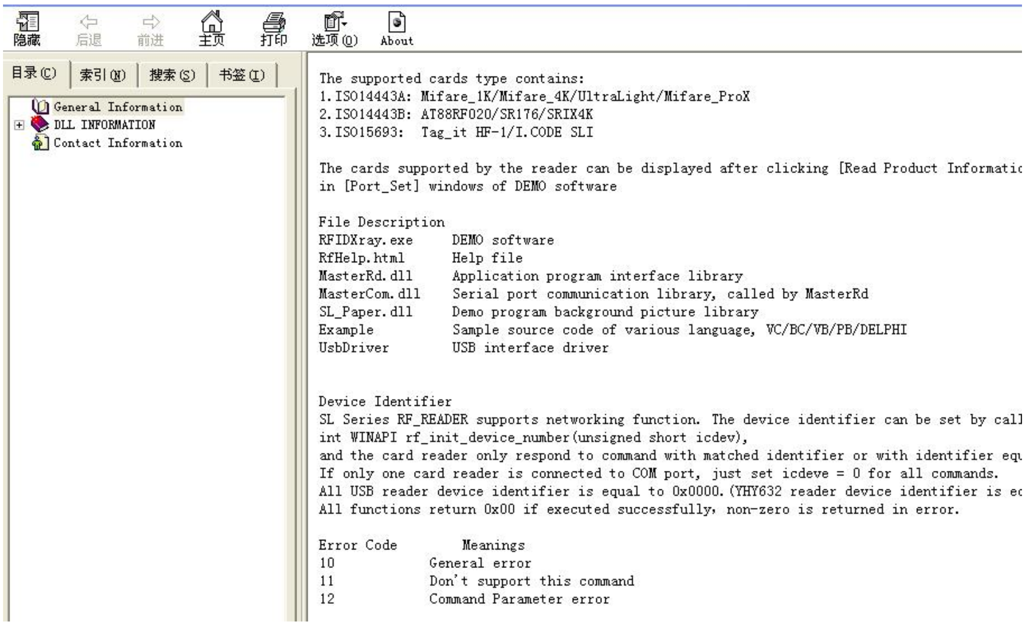
5.11 HELP



Port: COM3

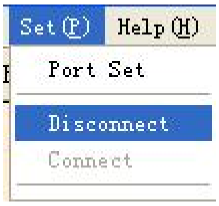
Baud:

You can search DLL function defined in the help file.

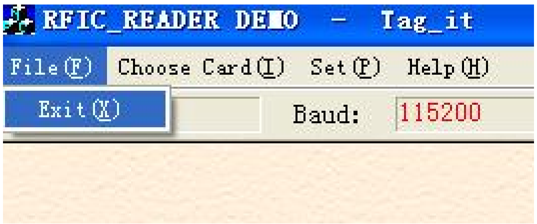


```
The supported cards type contains:
1.ISO14443A: Mifare_1K/Mifare_4K/UltraLight/Mifare_ProX
2.ISO14443B: AT88RF020/SR176/SRIX4K
3.ISO15693:  Tag_it HF-1/I.CODE SLI

The cards supported by the reader can be displayed after clicking [Read Product Informatic
in [Port_Set] windows of DEMO software

File Description
RFIDXray.exe     DEMO software
RfHelp.html      Help file
MasterRd.dll     Application program interface library
MasterCom.dll    Serial port communication library, called by MasterRd
SL_Paper.dll     Demo program background picture library
Example          Sample source code of various language, VC/BC/VB/PB/DELPHI
UsbDriver        USB interface driver


Device Identifier
SL Series RF_READER supports networking function. The device identifier can be set by call
int WINAPI rf_init_device_number(unsigned short icdev),
and the card reader only respond to command with matched identifier or with identifier equ
If only one card reader is connected to COM port, just set icdeve = 0 for all commands.
All USB reader device identifier is equal to 0x0000.(YHY632 reader device identifier is ec
All functions return 0x00 if executed successfully, non-zero is returned in error.

Error Code        Meanings
10              General error
11              Don't support this command
12              Command Parameter error
```

5.12 DISCONNECT READER



5.13 EXIT



Baud: 115200

6. DLL INFORMATION (icdev=0)

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, 19200 ~ 115200 bps
Return: return 0 if successful

6.1.3 INT WINAPI RF_CLOSEPORT

Function: Disconnect
Prototyp: int WINAPI rf_ClosePort(void)
Return: return 0 if successful

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: return 0 if successful

6.1.5 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: return 0 if successful
Explanation: this function is not effective to the readers only support single protocol.
type = 'A': set YHY638 into ISO14443A mode
type = 'B': set ISO14443B mode
type = 'r': set AT88RF020 card mode
type = '1': set ISO15693 mode

6.1.6 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: return 0 if successful
Explanation:model = 0: turn off RF transmittal
model = 1: turn on RF transmittal

6.1.7 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 (RFU)
Return: return 0 if successful

6.1.8 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 Msec
Return: return 0 if successful

6.2 ISO14443A FUNCTION

6.2.1 UltraLight

6.2.1.1 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: return 0 if successful
Explanation:mode = 0x26: REQ_STD
mode = 0x52: REQ_ALL

6.2.1.2 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: return 0 if successful

6.2.1.3 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: return 0 if successful
Explanation: this function is also applicable for UltraLight card. Every page of UltraLight card has 4 bytes. Transfering this function every time, return data of 4 consecutive pages.

6.2.1.4 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: return 0 if successful

6.2.1.5 INT WINAPI RF_HALT

Function: TYPE_A card HALT
Prototype: int WINAPI rf_halt (unsigned short icdev)
Parameter: icdev: [IN] Device ID
Return: return 0 if successful

6.2.2 Mifare_Std

6.2.2.1 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: return 0 if successful
Explanation: mode = 0x26: REQ_STD
mode = 0x52: REQ_ALL

6.2.2.2 INT WINAPI RF_ANTICOLL

Function: Mifare card Anticollision
Prototype: int WINAPI rf_anticoll (unsigned short icdev,
unsigned char bcnt,
unsigned char *pSnr,
unsigned char *pLen)

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

6.2.2.3 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: return 0 if successful
Explanation: 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.

6.2.2.4 INT WINAPI RF_M1_AUTHENTICATION2

Function: Mifare_Std Authentify
Prototype: int WINAPI rf_M1_authentication2 (unsigned short icdev,
unsigned char model,
unsigned char block,
unsigned char *pKey)
Parameter: icdev: [IN] Device ID
model: [IN] key validate mode
block: [IN] block absolute address
pKey: [IN] 6 bytes password
Return: return 0 if successful
Explanation:model = 0x60: via KeyA
model = 0x61: via KeyB

6.2.2.5 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: return 0 if successful

6.2.2.6 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: return 0 if successful

6.2.2.7 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: return 0 if successful

6.2.2.8 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: return 0 if successful

6.2.2.9 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: return 0 if successful

6.2.2.10 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: return 0 if successful

6.2.2.11 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: return 0 if successful

6.2.2.12 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
Return: return 0 if successful
Explanation: this function only be transferred after increment, decrement and restore command

6.2.2.13 INT WINAPI RF_HALT

Function: Mifare Halt
Prototype: int WINAPI rf_halt (unsigned short icdev)
Parameter: icdev: [IN] Device ID
Return: return 0 if successful
Explanation: card exit active estate after received this command

6.2.3 Mifare_ProX

6.2.3.1 INT WINAPI RF_TYPE_RST

Function: Request ISO14443A-4 card and reset

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

Parameter: icdev: [IN] Device ID
model: [IN] request mode
pData: [OUT] response data from card
pMsgLg: [OUT] length of response data
Return: return 0 if successful
Explanation:mode = 0x26: REQ_STD
mode = 0x52: REQ_ALL
pData: 4bytes CSN + RATS according to ISO14443A

6.2.3.2 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: return 0 if successful

6.2.3.3 INT WINAPI RF_CL_DESELECT

Prototype: int WINAPI rf_cl_deselect (unsigned short icdev)
Parameter: icdev: [IN] Device ID
Return: return 0 if successful

6.3 ISO14443 Type B FUNCTION

6.3.2 AT88RF020

6.3.2.1 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: return 0 if successful

6.3.2.2 INT WINAPI RF_ AT020_CHECK

Function: AT88RF020 card Authentify
Prototype: int WINAPI rf_at020_check (unsigned short icdev, unsigned char *pKey)
Parameter: icdev: [IN] Device ID
pKey: [IN] 8 bytes pass word
Return: return 0 if successful

6.3.2.3 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: return 0 if successful

6.3.2.4 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: return 0 if successful

6.3.2.5 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: return 0 if successful

6.3.2.6 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: return 0 if successful

6.3.2.7 INT WINAPI RF_ AT020_DESELECT

Function: AT88RF020 card Deselect
Prototype: int WINAPI rf_at020_deselect (unsigned short icdev)
Parameter: icdev: [IN] Device ID
Return: return 0 if successful

6.3.3 SR176/SRIX4K

6.3.3.1 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: return 0 if successful

6.3.3.2 INT WINAPI INT_RF_SR176_READBLOCK

Function: SR176 Read
Prototype: int WINAPI int rf_sr176_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: return 0 if successful

6.3.3.3 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: return 0 if successful

6.3.3.4 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: return 0 if successful
Explanation: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

6.3.3.5 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: return 0 if successful

6.3.3.6 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: return 0 if successful

6.3.3.7 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: return 0 if successful

6.3.3.8 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: return 0 if successful
Explanation: 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 ISO15693 FUNCTION

6.4.1 INT WINAPI ISO15693_INVENTORY

Function: ISO15693_Inventory (single card)
Prototype: int WINAPI ISO15693_Inventory (unsigned short icdev,
unsigned char *pData,
unsigned char *pLen)
Parameter: icdev: [IN] Device ID
pData: [OUT] response data from tag, 1 byte DSFID + 8 bytes UID
pLen: [OUT] length of response data
Return: return 0 if successful

6.4.2 INT WINAPI ISO15693_INVENTORYS

Function: ISO15693_Inventory (several cards)
Prototype: int WINAPI ISO15693_Inventorys (unsigned short icdev,

unsigned char *pData,
unsigned char *pLen)

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: return 0 if successful

6.4.3 INT WINAPI ISO15693_GET_SYSTEM_INFORMATION

Function: ISO15693_Get_System_Information
Prototype: int WINAPI ISO15693_Get_System_Information(unsigned short icdev,
unsigned char model,
unsigned char *pUID,
unsigned char *pData,
unsigned char *pLen)
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: return 0 if successful
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
Clear Option_flag = 0

6.4.4 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: return 0 if successful

6.4.5 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=Addres_flag, bit2=Option_flag
pUID: [IN] 8 bytes UID
Return: return 0 if successful
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
Clear Option_flag = 0

6.4.6 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: return 0 if successful

6.4.7 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=Addres_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: return 0 if successful
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
Clear Option_flag = 0

6.4.8 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=Addres_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: return 0 if successful
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
Clear Option_flag = 0

6.4.9 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=Addres_flag, bit2=Option_flag
pUID: [IN] 8 bytes UID
block: [IN] block address
pData: [IN] written data, 4 bytes
Return: return 0 if successful
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

6.4.10 INT WINAPI ISO15693_LOCK_BLOCK

Function: ISO15693_Lock_Block
Prototype: int WINAPI ISO15693_Lock_Block (unsigned short icdev,
unsigned char model,
unsigned char *pUID,
unsigned char block)
Parameter: icdev: [IN] Device ID
model: [IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag
pUID: [IN] 8 bytes UID
block: [IN] block address
Return: return 0 if successful
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

6.4.11 INT WINAPI ISO15693_WRITE_AFI

Function: ISO15693_Write_AFI
Prototype: int WINAPI ISO15693_Write_AFI (unsigned short icdev,
unsigned char model,
unsigned char *pUID,
unsigned char AFI)
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: return 0 if successful
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

6.4.12 INT WINAPI ISO15693_LOCK_AFI

Function: ISO15693_Lock_AFI
Prototype: int WINAPI ISO15693_Lock_AFI (unsigned short icdev,
unsigned char model,
unsigned char *pUID)
Parameter: icdev: [IN] Device ID
model: [IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag
pUID: [IN] 8 bytes UID
Return: return 0 if successful
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

6.4.13 INT WINAPI ISO15693_WRITE_DSFID

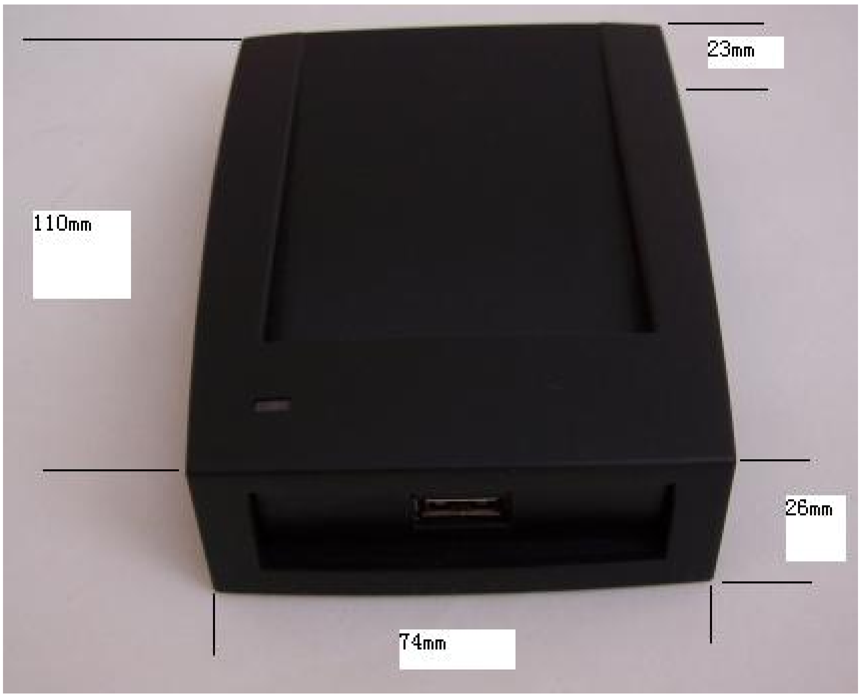
Function: ISO15693_Write_DSFID
Prototype: int WINAPI ISO15693_Write_DSFID (unsigned short icdev,
unsigned char model,
unsigned char *UID,
unsigned char DSFID)
Parameter: icdev: [IN] Device ID
model: [IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag
pUID: [IN] 8 bytes UID
DSFID: [IN] DSFID to be written
Return: return 0 if successful
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

6.4.14 INT WINAPI ISO15693_LOCK_DSFID

Function: ISO15693_Lock_DSFID
Prototype: int WINAPI ISO15693_Lock_DSFID (unsigned short icdev,
unsigned char model,
unsigned char *pUID)
Parameter: icdev: [IN] Device ID
model: [IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag
pUID: [IN] 8 bytes UID
Return: return 0 if successful
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

7 Pictures



YHY638 size picture

YHY638 Top View

YHY638 Side View

YHY638 Back Side View

YHY638 Rear Side View

Contact Information:
EHUOYAN Technology Co., Ltd.
Tel: +86 -010-59870151
Fax: +86 -010-59754725
email: info@ehuoyan.com
WebSite: www.ehuoyan.com

FILE END