# ComPro Reader instruction

# [Function interface]

**V1.14**

# API List Of Functions

## Built In Function API

| Function name | Function description |
|---|---|
| lc_init<br>lc_init_ex | Connect reader / writer |
| lc_exit | Disconnect reader / writer |
| lc_getver | Read firmware version |
| asc_hex | ASC code to hexadecimal |
| hex_asc | Hexadecimal code to ASC |
| lc_getFlashUserMemSize | Get device memory size |
| lc_srd_flash | Read device memory |
| lc_swr_flash | Writer device memory |
| lc_devReboot | Device restart |
| lc_setAddress | Set device address |
| lc_getAddress | Get device address |
| lc_setBaudRate | Set baud rate |
| lc_SetDevPassword | Set device password |
| lc_crypt | Encrypt or Decrypt |
| lc_setReadOnlyPara | Set read-only parameters |
| lc_getReadOnlyPara | Get read-only parameters |
| lc_setAppendData | Set additional parameters |
| lc_getAppendData | Get additional parameters |
| lc_setReaderMode | Set operation mode |
| lc_getReaderMode | Get operation mode |
| lc_set_identifyResponsePara | Set response parameters for object recognition |
| lc_get_identifyResponsePara | Get response parameters for object recognition |
| lc_set_AutoReportDataType | Set automatic reporting parameters |
| lc_get_AutoReportDataType | Get automatic reporting parameters |
| lc_getAutoReturnedData | Get automatically reported data |

## Simple Hardware API

| Function name | Function description |
|---|---|

| | |
|---|---|
| lc_beep | The card reader sounds |
| lc_led | Control LED |

# RFID Function API

## ISO14443-A

| Function name | Function description |
|---|---|
| lc_setANT | Set RF antenna status |
| lc_selectANT | Select antenna |
| lc_rfReset | RF reset |
| lc_card | Find typeA card |
| lc_requestAndIdentifyTypeA | Find and identify chip of card |
| lc_authentication | Verify password |
| lc_read | General reading card |
| lc_read_convenient | One-click card reading |
| lc_write | General writing card |
| lc_write_convenient | One-click card writing |
| lc_initval | Create wallet |
| lc_increment | Charging |
| lc_decrement | Deductions |
| lc_restore | Dump value block |
| lc_readval | Read value |
| lc_updatekey | Change card password |
| lc_readTag | Read NTAG tags |
| lc_writeTag | Write NTAG tags |
| lc_tag_authenPwd | Verify NTAG password |
| lc_tag_setPwd | Set NTAG password |
| lc_halt | Stop card |
| lc_rawExchange | Transceve with NFC card |
| lc_pro_reset | CPU card reset |
| lc_pro_commandlink | CPU card APUD Instruction exchange |
| lc_getPBOC_PAN | Get UnionPay account |
| lc_readSSC | Get base information of Social-Security-Card of China |
| lc_ultralt_C_authen | Verify the Utralight-C password |
| lc_ultralt_C_setSafePage | Set Utralight-C Protection Page |
| lc_ultralt_C_changePwd | Change the Utralight-C password |
| lc_ultralt_C_lockPage | Lock Utralight-C page |
| lc_authen_desfire | Verify the defire card (DES mode) |
| lc_authen_desfire_aes | Verify the defire card (AES mode) |

| | |
|---|---|
| lc_getver_desfire | Get the defire card (version) parameters |
| lc_getAIDs_desfire | Get the desfire app ID |
| lc_selectApp_desfire | Select the application of desfire |
| lc_getKeySetting_desfire | Get the desfire key settings |
| lc_getKeyver_desfire | Get the version information of the desfire key |
| lc_createApp_desfire | Create a desfire app |
| lc_delAID_desfire | Delete the desfire app |
| lc_changeKeySetting_desfire | Change the desfire key configuration |
| lc_changeKey_desfire | Update the desfire key |
| lc_changeKey_desfire_aes | Update AES key for desfire |
| lc_getFileIDs_desfire | Get the file ID of desfire |
| lc_getFileProper | Get the file properties of desfire |
| lc_changeFileSetting | Update the file properties of desfire |
| lc_createDataFile_desfire | Create a desfire data file |
| lc_createBackupDataFile_desfire | Create a desfire backup file |
| lc_createValueFile_desfire | Create a desfire value file |
| lc_createCsyRecord_desfire | Create a desfire loop log file |
| lc_delFile_desfire | Delete the desfire file |
| lc_write_desfire | Write a desfire file |
| lc_read_desfire | Read the desfire file |
| lc_getvalue_desfire | Read value desfire |
| lc_credit_desfire | Bonus desfire |
| lc_debit_desfire | Impairment desfire |
| lc_writeRecord_desfire | Write a desfire log file |
| lc_readRecord_desfire | Read the desfire log file |
| lc_clearRecord_desfire | Clear the desfire log file |
| lc_commitTransfer_desfire | Submit desfire transfer |
| lc_abortTransfer_desfire | Terminate the desfire transmission |
| lc_formatPICC_desfire | Format desfire |
| lc_MFPlusL0_WritePerso | Plus card level 0 personalization |
| lc_MFPlusL0_CommitPerso | Plus card submission level 0 personalization |
| lc_MFPlusL1_AuthenKeyL1 | Plus card verification level 1 key |
| lc_MFPlusL1_SwitchToL2 | Plus card upgrade from level 1 to level 2 |
| lc_MFPlusL1_SwitchToL3 | Plus card upgrade from level 1 to level 3 |
| lc_MFPlusL2_SwitchToL3 | Plus card upgrade from level 2 to level 3 |
| lc_MFPlusL3_AuthenL3Key | Verification of level 3 master key with plus card |
| lc_MFPlusL3_AuthenSectorKey | Verify the level 3 sector key with the plus card |
| lc_MFPlusL3_UpdateKey | Plus card update level 3 key |
| lc_MFPlusL3_ReadWithPlain | Level 3 plaintext reading of plus card |

| | |
|---|---|
| lc_MFPlusL3_WriteWithPlain | Level 3 plaintext writing of plus card |
| lc_MFPlusL3_ReadWithEncrypt | Level 3 ciphertext reading of plus card |
| lc_MFPlusL3_WriteWithEncrypt | Level 3 ciphertext writing of plus card |
| lc_MFPlusL3_InitVal | Level 3 initialization value of plus card |
| lc_MFPlusL3_ReadVal | Level 3 read value of plus card |
| lc_MFPlusL3_Increment | Plus card level 3 bonus |
| lc_MFPlusL3_Decrement | Plus card level 3 impairment |

## ISO14443-B

| Function name | Function description |
|---|---|
| lc_findTypeB | Find TypeB card |
| lc_typeB_command | TypeB card instruction exchange |

## ISO15693

| Function name | Function description |
|---|---|
| lc_find15693 | Find ISO15693 card |
| lc_15693_select_uid | Select ISO15693 card |
| lc_15693_readBlock | Read ISO15693 card |
| lc_15693_writeBlock | Write ISO15693 card |
| lc_15693_lock_block | Lock ISO15693 card designation block |
| lc_15693_write_afi | ISO15693 card write AFI |
| lc_15693_lock_afi | ISO15693 card lock AFI |
| lc_15693_write_dsfid | ISO15693 card write DSFID |
| lc_15693_lock_dsfid | ISO15693 card lock DSFID |
| lc_15693_get_systemInfo | ISO15693 card to obtain system parameters |
| lc_15693_get_securityInfo | Obtaining security data with ISO15693 card |
| lc_15693set_password | ISO15693 card setting (verification) password |
| lc_15693_write_password | ISO15693 card setting (updating) password |
| lc_15693_icode_64bitProtect | ISO15693 card set 64 bit protection |
| lc_15693_protect_page | ISO15693 card setting protection page |
| lc_15693_command_custom | ISO15693 user instruction exchange |

## ISO18092(TypeF)

| Function name | Function description |
|---|---|
| lc_findTypeF | Find TypeF card |
| lc_typeF_command | TypeF card instruction exchange |

## UHF

| Function name | Function description |
|---|---|
| lc_uhf_inventory | Inventory (Find) UHF tag |
| lc_uhf_readTag | Read UHF tag |
| lc_uhf_writeTag | Write UHF tag |
| lc_uhf_lockTag | Lock UHF tag |
| lc_uhf_killTag | Kill UHF tag |

# Contact IC Card API

| Function name | Function description |
|---|---|
| lc_iccGetCardState | Check if a card in the slot |
| lc_iccSelCardType | Select the card type |
| lc_iccGetATR | Get ATR of ISO7816 card |
| lc_icc_APDU | APDU operate with ISO7816 card |
| lc_icc_ReadMem | Read data in the memory of card |
| lc_icc_WriteMem | Write data to the memory of card |
| lc_icc_VerifyUserPass | Verify user zone password of logic card |
| lc_icc_ReadErrorCounter | Read the counter of verify wrong password |
| lc_icc_UpdateUserPass | Update the user zone password of logic card |
| lc_iccSetBaud | Set baud when communicate with card |
| lc_iccPPS | Protocol and parameter exchange |
| lc_readSIM | Read SIM card base information |

# NET Function API

| Function name | Function description |
|---|---|
| lc_setNet_serverIP | Set the IP address of the server to which |

| | |
|---|---|
| | Net will connect |
| lc_getNet_serverIP | Get the server IP of Net connection |
| lc_setNet_serverPort | Set the server port of Net connection |
| lc_getNet_serverPort | Get the server port of Net connection |
| lc_setNet_URL | Set the URL of Net connection |
| lc_getNet_URL | Get the URL of Net connection |
| lc_setNet_heartBeatInterval | Set the heartbeat of Net terminal |
| lc_getNet_heartBeatInterval | Get the heartbeat of Net terminal |
| lc_setNet_paraTitle_Data | Set the para-data title for uploading package |
| lc_getNet_paraTitle_Data | Get the para-data title for uploading package |
| lc_setNet_paraTitle_Addr | Set the para-addr title for uploading package |
| lc_getNet_paraTitle_Addr | Get the para-addr title for uploading package |
| lc_setNet_paraTitle_Time | Set the para-time title for uploading package |
| lc_getNet_paraTitle_Time | Get the para-time title for uploading package |
| lc_setNet_paraTitle_DataType | Set the para-dataType title for uploading package |
| lc_getNet_paraTitle_DataType | Get the para-dataType title for uploading package |
| lc_setNet_paraTitle_PackType | Set the para-packType title for uploading package |
| lc_getNet_paraTitle_PackType | Get the para-packType title for uploading package |
| lc_setNet_MQTT_clientID | Set the client ID when using MQTT |
| lc_getNet_MQTT_clientID | Get the client ID when using MQTT |
| lc_setNet_MQTT_userName | Set the user name when using MQTT |
| lc_getNet_MQTT_userName | Get the user name when using MQTT |
| lc_setNet_MQTT_userPass | Set the user name when using MQTT |
| lc_getNet_MQTT_userPass | Get the user name when using MQTT |
| lc_setNet_MQTT_publish_topic | Set the publish topic when using MQTT |
| lc_getNet_MQTT_publish_topic | Set the publish topic when using MQTT |
| lc_setContent_type | Set the content type for uploading |
| lc_getContent_type | Get the content type for uploading |

# WIFI Function API

| Function name | Function description |
|---|---|
| lc_setWiFi_STA_Name | Set the station name ready for linking |
| lc_getWiFi_STA_Name | Get the station name ready for linking |
| lc_setWiFi_STA_Password | Set the station password for linking |
| lc_getWiFi_STA_Password | Get the station password for linking |

# Ethernet Function API

| Function name | Function description |
|---|---|
| lc_setETH_localIP | Set local IP address |
| lc_getETH_localIP | Get local IP address |
| lc_setETH_localPort | Set local listening port |
| lc_getETH_localPort | Get local listening port |
| lc_setETH_gatewayIP | Set the gateway IP address |
| lc_getETH_gatewayIP | Get the gateway IP address |

# Bluetooth Function API

| Function name | Function description |
|---|---|
| lc_getBTMAC | Get Bluetooth device's MAC |
| lc_setBTName | Set Bluetooth device's name |
| lc_getBTName | Get Bluetooth device's name |

# API Interface details

## Built In API Details

**int lc_init(int port, long baud);**

Description: Connect the reader / writer

Parameter: port: Enter the port number to be connected. The value of 0 ~ 99 corresponds to serial port com1 ~ 100 respectively, and the value of 100 corresponds to USB

baud: Input, port communication baud rate

Return: = - 1 failed

< > - 1 valid device handle

Example: int hdev;

hDev = lc_init (0, 115200);//link com1

if( hDev != -1) printf("Link reader ok, handle:%d", hDev);

else printf("Link reader error");

**int lc_init_ex(int portType, char* szPathName, long baud);**

Description: Connect the reader / writer

Parameter: porttype: Input, connection port type, serial port type value 1, USB port value 2, Bluetooth type value 3, TCPIP-Client type value 4, TCPIP-Server type value 5

szpathname: Input, the connection string of the reader / writer. When connecting Bluetooth, it is the string of its MAC code in the format of XX: XX: XX: XX: XX: XX; Format IP:port if link TCPIP .

baud: Input, port communication baud rate, invalid parameter in Bluetooth mode, set to 0

Return: = - 1 failed

< > - 1 valid device handle

Example: int hdev;

hdev = lc_init_ex (1, "/dev/ttyS3", 115200);

if( hdev != -1) printf("Link reader ok, handle:%d", hdev);

else printf("Link reader error");

**int lc_exit(int icdev);**

Description: Exit the connected reader / writer

Parameters: icdev: Input, connection handle of reader / writer

Return: < > 0 error code

=0 succeeded

Example: extern int icdev;

lc_exit(icdev);

**int asc_hex (unsigned char *strAsc_in,unsigned char *strHex_out,int len_hex);**

Description: Convert the string into hexadecimal byte array

Parameter: strasc_ in: Enter the string to convert

strHex_ out: Output, converted hexadecimal data (result)

len_ hex: Input, corresponding to hexadecimal data length (number of bytes)

Return: < > 0 error code

=0 succeeded

Example: unsigned char* strAsc= "616263";

unsigned char bufHex[4]={0};

asc_hex(strAsc, bufHex, 3);//bufHex->{'a','b','c'}

**int hex_asc(unsigned char *bufHex_in,unsigned char *strAsc_out,int len_hex);**

Description: Convert hexadecimal byte array into string

Parameter: bufhex_ in: Input, 16-digit array to convert

strAsc_ out: Output, converted String (result)

len_ hex: Input, corresponding to hexadecimal data length (number of bytes)

Return: < > 0 error code

=0 succeeded

Example: unsigned char bufHex[4]= {'a', 'b', 'c'};

unsigned char szAsc[7]={0};

hex_asc(bufHex, szAsc, 3);// szAsc ->”616263”

**int lc_getFlashUserMemSize (int icdev, unsigned int* pSize);**

Description: Get the maximum memory space of the device (nonvolatile)

Parameters: icdev: Input, connection handle of reader / writer

pSize: Output, obtained memory size (in bytes)

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned int flashMemSize = 0;

st = lc_getFlashUserMemSize(hdev, &flashMemSize );

**int lc_srd_flash(int icdev, int offset, int length, unsigned char* buf_out);**

Description: Read device memory data

Parameters: icdev: Input, connection handle of reader / writer

offset: Input, address of memory to read

length: Input, length of memory to read

buf_out: Output, pointer to receive memory data

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

int st;

char buf[128];

st = lc_srd_flash(hdev, 0, 10, buf);// Read 10 bytes from 0 memory location

**int lc_swr_flash(int icdev, int offset, int length, unsigned char* buf_in);**

Description: Write data to device memory

Parameters: icdev: Input, connection handle of reader / writer

offset: Input, address of data to be written in memory

length: Input, length of data to be written

buf_in: Input, data to be written

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

int st;

char

buf[128]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a};

st = lc_swr_flash(hdev, 0, 10, buf);// Write 10 bytes of data to address 0 in memory

**int lc_getver(int icdev, unsigned char* version);**

Description: Reader / writer gets device version number

Parameter: icdev: Input, connection handle of reader / writer

version: Output, get the device version number

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

char szVer[128];

lc_gerver(hdev, szVer);

printf("Device version: %s", szVer);

**int lc_devReboot(int icdev);**

Description: Perform device restart

Parameter: icdev: Input, connection handle of reader / writer

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

lc_devReboot(hdev);

**int lc_setAddress(int icdev, unsigned int addr);**

Description: Set the address of the device

Parameter: icdev: Input, connection handle of reader / writer

addr: Input, set the address of the device, mainly for 485 or network interfaces

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setAddress (hdev, 1 );// Set device address 1

**int lc_getAddress(int icdev, unsigned int* pAddr);**

Description: Get the address of the device

Parameter: icdev: Input, connection handle of reader / writer

pAddr : Output, current address number of device

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned int devAddr = 0;

st = lc_getAddress (hdev, &devAddr );

**int lc_setBaudRate(int icdev, int baud);**

Description: Set the communication baud rate of the device

Parameter: icdev: Input, connection handle of reader / writer

baud: Input, set the baud rate of the device

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setBaudRate (hdev, 9600 );// Set baud rate 9600

**int lc_SetDevPassword (int icdev, unsigned char pwdAddr,unsigned char pwdLen, unsigned char* pPwd);**

Description: Set the password for encryption of the device

Parameter: icdev: Input, connection handle of reader / writer

pwdAddr: Input, the start address in the password area (Valid value 0~31)

pwdLen: Input, password length (up to 32 bytes)

pPwd: Input, password content

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char pwd[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,
0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st = lc_setDevPassword (hdev, 0,16, pwd );

**int lc_crypt(int icdev,unsigned char fEn_De, unsigned char algorithm,unsigned int srcLen, unsigned char *pSrc_in, unsigned char* pIv_in,unsigned char* pDest_out, unsigned int* pDestLen_out);**

Description: Do a encrypt or decrypt calculate, with key from address 16 in the password area which should been set before.

NOTE: This function support only by some special products.

Parameter:

icdev: Input, connection handle of reader / writer

fEn_De: Input, the flag which indicate encrypt(set to 1) or decrypt(set to

0)

algorithm: Input, which algorithm crypt with, value can be set:

0 – DES with ECB mode

1 – DES with CBC mode

2 – 3DES with ECB mode

3 – 3DES with CBC mode

4 – AES with ECB mode

5 – AES with CBC mode

srcLen: Input, souce data length (should be multiple of 8)

pSrc_in:Input, souce data buffer

pIv_in: Input, initial vector value (should be 8 bytes)

pDest_out: Output, the buffer to receive result after calculate

pDestLen_out: Output, the result buffer length (always equal to length of source)

Return: < > 0 error code

=0 success

Example:

```
int   st;
extern int hdev;
unsigned                                                    char
source[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,
                                0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};
unsigned char bufIv[9]={0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00};
unsigned char bufRel[16];
unsigned char relSize;
st   = lc_crypt (hdev, 1, 3, 16, source, bufIv, bufRel, &relSize);
```

**int   lc_setReadOnlyPara (int icdev, unsigned char* pParaIn, unsigned char paraLen);**

Description: Set read-only parameters

Parameter: icdev: Input, connection handle of reader / writer

pParaIn: Input, input parameters, the meaning of each data segment is as follows:

RawDataSize: 1 byte, original data size, 0-all (by identified object), other values fixed (number of bytes)

Dir: 1 byte, data direction, 0 (reverse) high bit first, 1 (forward) high bit after

Data_Sys: 1 byte, data format, 0-Decimal, 1-Hexadecimal, 2-Hexadecimal String, 3-Wiegand format

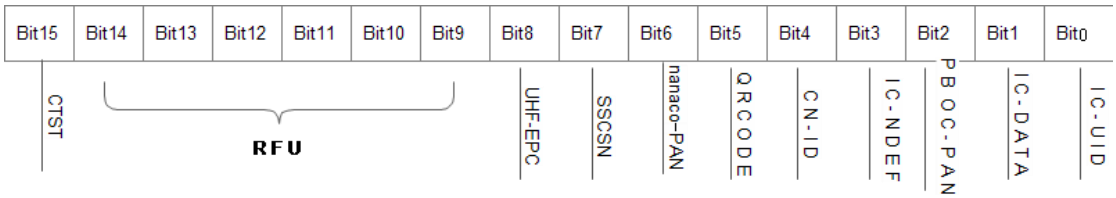
Show_Len: 1 byte, byte length of actual output, 0-default (total output), other values fixed (number of bytes)

DataAddr: 1 byte, address of data block/page, 0~255

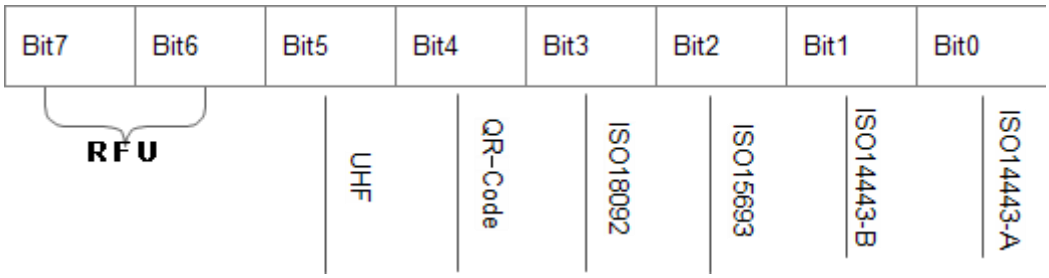
KeyOfData: 6 bytes, password to access data (M1/S70 is 6 bytes, NTag, ICode takes the first 4 bytes valid)

AuthenMode: 1 byte, 0x60 is KeyA (corresponding to KeyA of M1, password to read pages for NTag, password to read for ICode), 0x61 is KeyB (corresponding to KeyB of M1, password to write for NTag) and 0x61 is KeyB (corresponding to KeyB of M1)

Obj_Type: 2 bytes, identifies the object, defines whether or not the target is supported by the corresponding Bit bit is empty. Currently the supported types are IC-UID (IC card physical card number), IC-DATA (IC card data block or page), PBOC-PAN (UnionPay card number), IC-NDEF (IC tag card NDEF data), CN-ID (second generation ID code), QRCode (two-dimensional code/bar code), nanaco-PAN (Felica’s nanaco Account), SSCSN (Chinese Social Security Card Number), CTST (Contact Slot State), UHF-EPC (UHF Tag’s EPC), see the following figure:




Obj_Pro: 1 byte, device supports read-only protocol, defines whether to support the corresponding protocol according to whether the corresponding Bit is empty or not. The current supported protocols are ISO14443-A, ISO14443-B, ISO18092, QR-Code(two-dimensional code), UHF. See the following figure:




paraLen: Input, parameter length (15 bytes)

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
unsigned char para[16]={
0x00,// All raw data lengths
0x00,// Reverse, high first
0x02,// Hexadecimal String Output
0x00,// Output by original data length
0x00,// Address of the read data block (valid when the target is IC-DATA)
0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,// Validation password for data block (valid when IC-DATA)
0x60,// Verify password A (valid when IC-DATA)
0x00,0x01,// Identify Target Set to Read UID
0x01,// Identify protocol set to ISO14443-A
```

```
};
st = lc_setReadOnlyPara (hdev, para, 16 );
```

**int lc_getReadOnlyPara(int icdev, unsigned char* pPara_out, int* paraLen_out);**

Description: Get read-only settings

Parameter: icdev: Input, connection handle of reader / writer

pPara_out: Output, output parameters, meaning of each data segment see lc_ SetReadOnlyPara

paraLen_out: Output, parameter length

Return: < > 0 error code

=0 succeeded

Example:
```
int st;
extern int hdev;
unsigned char para[16];
int rLen;
st = lc_getReadOnlyPara (hdev, para, &rLen );
```

**int lc_setAppendData (int icdev, unsigned char* para_in, unsigned char paraLen);**

Description: Set read-only additional data

Parameter: icdev: Input, connection handle of reader / writer

para_in: Input, input parameters, the meaning of each data segment is as follows:

fAppendDevAddr: 1 byte, whether to attach device address, 0-not additional, 1- additional

fAppendTime: 1 byte, whether to append local time, 0-not additional, 1- additional

DataLenForword: 1 byte, the number of additional bytes before the original data (maximum 4)

DataForword: 4 bytes, preset 4 bytes before the original data

DataLenBehind: 1 byte, the number of additional bytes after the original data (maximum 4)

DataBehind: 4 bytes, preset the additional data after the original data

fAppendDataLen: 1 byte, whether to append the data length (before the original data), 0 - not additional, 1 - additional

bAppendCrc: 1 byte, additional verification method, 0-not additional, 1-additional XOR verification, 2-additional sum verification, 3-additional sum inversion verification

paraLen: input, parameter length (15 bytes)

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
unsigned char para[16]={
0x00,// No additional device address
0x00,// No additional local time
0x01,// Number of data attached before data
0x02,0x00,0x00,0x00,// Preset previously attached data
0x01,// Number of data attached after data
0x03,0x00,0x00,0x00,// Preset additional data after
0x00,// No additional data length
0x00// No additional validation of the original data
};
st = lc_setAppendData (hdev, para, 14 );
```

**int lc_getAppendData(int icdev, unsigned char* pPara_out, unsigned char* pLen_out);**

Description: Get read-only additional settings

Parameter: icdev: Input, connection handle of reader / writer

pPara_out : Output, output parameters, meaning of each data segment see lc_ setAppendData

paraLen_out : Output, parameter length

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char para[16];

int rLen;

st = lc_getAppendData (hdev, para, &rLen );

**int lc_set_AutoReportDataType(int icdev, unsigned char type,unsigned short outputPort);**

Description: Set parameters for automatic reporting

Parameter: icdev: Input, connection handle of reader / writer

type: Input, input parameters, output packet format, meaning as follows:

0x00: raw data

0x01: HTTP mode

0x02: Simulate keyboard style

0x04: MQTT mode

outputPort: Input, Output port, the value means as follows:

0x01: Serial/TTL

0x02: USB

0x04: Ethernet

0x08: Bluetooth

0x10: Wifi

0x20: NB

0x40: Wigan

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char repMode = 0x00;// Raw data

unsigned char outPort = 0x01;// Output to Serial Port

st = lc_set_AutoReportDataType (hdev, repMode, outPort );

**int lc_get_AutoReportDataType(int icdev, unsigned char* type_out, unsigned short* pOutputPort);**

Description: Get read-only additional settings

Parameter: icdev: Input, connection handle of reader / writer

type_out: Output, the form of data package currently set, meaning lc_ set_ AutoReportDataType

pOutputPort: Output, output port currently set, meaning lc_ Set_ AutoReportDataType

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char packMode;

unsigned short outPort;

st = lc_get_AutoReportDataType (hdev, &packMode, &outPort );

**int lc_set_identifyResponsePara (int icdev, unsigned char* pParaIn, unsigned char paraLen);**

Description: Set identification response parameters

Parameter: icdev: Input, connection handle of reader / writer

pParaIn: Input, input parameters, the meaning of each data segment is as follows:

fNeedHostResponse_ID_Come: 1 byte, need to wait for host to return reply, 0-no need, 1-need

bGreenLightPowerState_ID_Come: 1 byte, 0-extinct, 1-bright when the target is recognized as in the green light state

bGreenLightBlinkDelay_ID_Come: 2 bytes, the green light flashes once when the target is recognized, the duration of the green light lasts, the top position is

bBeepDelay_ID_Come: 2 bytes, the time it takes to beep when the target is recognized, the top position is

bGreenLightPowerState_ID_Leave: 1 byte, when the target is identified as leaving the green light

bGreenLightBlinkDelay_ID_Leave: 2 bytes, when the target is identified to leave the green light once, the duration of the green light, the top position is

bBeepDelay_ID_Leave: 2 bytes, top when recognizing how long the target left the buzzer

bNeedReportData_ID_Leave: 1 byte, whether the device should report data when recognizing whether the target is leaving (Note: Additional data needs to be set if reporting is started, see lc_setAppendData)

paraLen: Input, parameter length (12 bytes)

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
unsigned char para[16]={
0x00,// When the target enters, the device does not need to wait for the host to respond after reporting data
0x01,// When the target enters, the green light is on
0x00,0x00,// When the target enters，the green light does not flash
0x00,0x0A,// When the target enters, the beep lasts for 10ms
```

0x00,// The target leaves and the green light goes out

0x00,0x00,// The target leaves and the green light doesn't flash

0x00,0x00,// The target leaves and the buzzer doesn't sound

0x00,// The target leaves and the equipment doesn't report

};

st = lc_set_identifyResponsePara (hdev, para,12 );

**int lc_get_identifyResponsePara(int icdev, unsigned char* pParaOut, unsigned char* pParaLen);**

Description: Get target recognition response parameters

Parameter: icdev: Input, connection handle of reader / writer

pParaOut: Output, output parameters, meaning of each data segment see lc_ set_ identifyResponsePara

pParaLen: Output, parameter length

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char para[16];

unsigned char rLen;

st = lc_get_identifyResponsePara(hdev, para, &rLen );

**int lc_setReaderMode(int icdev, unsigned char mode);**

Description: Set the operating mode of the card reader

Parameter: icdev: Input, connection handle of reader / writer

mode: Input, run mode, read-write mode and read-only mode, the bit-7 indicate whether save to device, 0-save, 1-not save,values can set with:

0: Read and write mode(0x80 if temporary)

1: Read-only mode (device detects cards automatically, detects instant output. 0x81 if temporary)

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setReaderMode(hdev, 1 );// Set Read-Only Mode

Note: When set to read-only mode, no other APIs should be used to perform read-write card operations. The device changes to read-only mode and automatically senses after the card is monitored.

**int lc_getReaderMode(int icdev, unsigned char* mode_out);**

Description: Get the operating mode of the card reader

Parameter: icdev: Input, connection handle of reader / writer

mode_out: Output, run mode, read-write mode and read-only mode with the following parameters:

0: Read and write mode

1: Read-only mode (device detects cards automatically, detects instant output)

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char runMode;

st = lc_getReaderMode(hdev, &runMode );

**int lc_getAutoReturnedData(int icdev, unsigned char* pRData, unsigned int* pRLen);**

Description: Get data sent in device auto-run mode

Parameter: icdev: Input, connection handle of reader / writer

pRData: Output, data sent by the device

pRLen: Output, length of data sent by the device

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char rData[512];

unsigned int rLen;

st = lc_getAutoReturnedData(hdev, rData, &rLen );

# Detailed Simple Hardware API

**int lc_beep(int icdev, unsigned int _Msec);**

Description: Reader / writer beeps

Parameter: icdev: Input, connection handle of reader / writer

_Msec: Input, delay of beep in ms

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

lc_beep(hdev, 100);

**int lc_led(int icdev, int iLED, int on_off);**

Description: LED lamp control on reader / writer

Parameter: icdev: Input, connection handle of reader / writer

iLED: Input: LED serial number, 1-red light, 2-green light

on_off: Input, control light on(1) or light out(0)

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

lc_led(hdev, 2, 1);// Control green light on

# RFID Function API Details

## ISO14443-A Protocol API Details

**int lc_setANT(int icdev, unsigned char status);**

Description: set RF antenna status

Parameter: icdev: Input, connection handle of reader / writer

status: Input, turn off antenna (0), turn on antenna (1)

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

lc_setANT(hdev, 0);// Turn off antenna

**int lc_selectANT(int icdev, unsigned int iANT);**

Description: select and active special antenna, this function only match some module witch have multi-antenna.

Parameter: icdev: Input, connection handle of reader / writer

iANT: Input, the index of antenna to select, minimum value is 1

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

lc_selectANT(hdev, 1);// Select antenna 1

**int lc_rfReset(int icdev, unsigned int time);**

Description: Control radio frequency reset

Parameter: icdev: Input, connection handle of reader / writer

time: Input, Reset duration, MS

Return: < > 0 error code

=0 succeeded

Example: extern int hdev;

lc_rfReset(hdev, 10);

## Mifare Card

**int lc_card(int icdev, unsigned char _Mode, unsigned char* _Snr_out, unsigned char* outSnrSize,unsigned int* pTag, unsigned char* pSak);**

Description: Find a TypeA IC card

Parameter: icdev: Input, connection handle of reader / writer

_Mode: Input: Look for card mode, 0-single card mode, 1-multiple card mode

_Snr_out: Output, if successful, returns the card number byte stream

outSnrSize: Output, return card number length in bytes

pTag: Output, Tag identification of card

pSak: Output, sak value of card

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char snr[9];

unsigned char snSize;

unsigned long tag;

unsigned char sak;

extern int hdev;

st = lc_card(hdev, 1, snr, &snSize, &tag,&sak );// Multi-card mode card seeking

**int lc_requestAndIdentifyTypeA(int icdev,unsigned char _Mode,unsigned char *_Snr, unsigned char* _outSnrSize, unsigned char* pfTagType, unsigned char* pfCompliant14443_4);**

Description: Find a TypeA IC card and identify the chip type.

Parameter: icdev: Input, connection handle of reader / writer

_Mode: Input: Look for card mode, 0-single card mode, 1-multiple card mode

_Snr_out: Output, if successful, returns the card number byte stream

outSnrSize: Output, return card number length in bytes

pfTagType: Output, chip type of card

pfCompliant14443_4: Output, a flag indicate support ISO14443-4,1-yes,0-no

Return: < > 0 error code

=0 success

Example: int st;

unsigned char snr[9];

unsigned char snSize;

unsigned char chip;

unsigned char fSupport14443_4;

extern int hdev;

st = lc_requestAndIdentifyTypeA(hdev, 1, snr, &snSize, &chip,&fSupport14443_4 );// Multi-card mode card seeking

**int lc_authentication(int icdev, unsigned char authMode, unsigned char sector,unsigned char* pKey);**

Description: Verify the password of the Mifare class card

Parameter: icdev: Input, connection handle of reader / writer

authMode: Input: Password mode, 60h-Password A, 61h-Password B

sector: Input, sector address 0~15

pKey: Enter, zone password

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char defKey[] = {0xff, 0xff,0xff,0xff,0xff,0xff};

extern int hdev;

st = lc_authentication(hdev, 0x60, 1, defKey);// Verify Sector 1 password with default A password

**int lc_read(int icdev, unsigned char _Adr, unsigned char* _Data);**

Description: Read non-connected Mifare Class cards

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, block addresses 0~63

_Data: Output, read data

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char bufData[20];

extern int hdev;

st = lc_read(hdev, 1, bufData );// Data of reader 1

**int lc_read_convenient(int icdev, unsigned char blkAddr, unsigned char keyMode,unsigned char* pKey, unsigned char* _Data);**

Description: Easy to read non-connected Mifare class cards, this function integrates card seeking and verification

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, block addresses 0~63

keyMode: Input: Password mode, 60h-Password A, 61h-Password B

pKey: Enter, zone password

_Data: Output, read data

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char bufData[20];

unsigned char defKey[] = {0xff, 0xff,0xff,0xff,0xff,0xff};

extern int hdev;

st = lc_read_convenient(hdev, 1, 0x60, defKey, bufData );

**int lc_write(int icdev, unsigned char _Adr, unsigned char* _Data);**

Description: Write a non-connected Mifare Class card

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, block addresses 1~63

_Data: Input, data to be written

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char bufData[20]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b, 0xc,0x0d,0x0e,0xf,0x10};

extern int hdev;

st = lc_write(hdev, 1, bufData );// Data of reader 1

**int lc_write_convenient(int icdev, unsigned char blkAddr, unsigned char keyMode,unsigned char* pKey, unsigned char* _Data);**

Description: Quickly write non-connected Mifare class card, which integrates card search and verification

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, block addresses 1~63

keyMode: Input, password mode, 60h-Password A, 61h-Password B

pKey: Input, area password

_Data: Input, data to be written

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char

bufData[20]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,
0xc,0x0d,0x0e,0xf,0x10};

unsigned char defKey[] = {0xff, 0xff,0xff,0xff,0xff,0xff};

extern int hdev;

st = lc_write_convenient(hdev, 1, 0x60, defKey, bufData );

**int lc_initval(int icdev, unsigned char _Adr,unsigned long value);**

Description: Format wallet block

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, wallet block addresses 1~63

value: Input, initial value to format

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_initval(hdev, 1, 0);// Format block 1 into a wallet block with an initial value of 0

**int lc_increment(int icdev, unsigned char _Adr,unsigned long value);**

Description: Refill your wallet

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, wallet block addresses 1~63

value: Input, amount to recharge

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_increment(hdev, 1, 100);// Refill 100 to wallet block 1

**int lc_decrement(int icdev, unsigned char _Adr,unsigned long value);**

Description: Deduct money from Wallet

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, wallet block addresses 1~63

value: Input, amount to deduct

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_decrement(hdev, 1, 100);// Decrease value from wallet block 1 by 100

**int lc_restore(int icdev,unsigned char _fromAdr, unsigned char _toAdr);**

Description: Transfers the value of one value block to another block

Parameter: icdev: Input, connection handle of reader / writer

_fromAdr: Input, original block address of dump value

_toAdr: Input, destination block address of dump value

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_restore(hdev, 4，5);// Move the value in block 4 to block 5

**int lc_readval(int icdev, unsigned char _Adr,unsigned long* value_out);**

Description: Read the amount in your wallet

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, wallet block addresses 1~63

value_out: Output, amount read

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned long value;

extern int hdev;

st = lc_readval(hdev, 1, &value);

**int lc_updateKey(int icdev, unsigned char secNr, unsigned char* pNewKeyA, unsigned char* pNewCtrlW, unsigned char* pNewKeyB);**

Description: Change card password

Parameter: icdev: Input, connection handle of reader / writer

secNr: Input, sector number which key for updating

pNewKeyA: Input, new key-A, 6 bytes

pNewCtrlW: Input, new control word, 4 bytes.

pNewKeyB: Input, new key-B,6 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char n_keyA[6] ={0xff,0xff,0xff,0xff,0xff,0xff};

unsigned char n_ctrlW[4] = {0xff,0x07,0x80,0x69};

unsigned char n_keyB[6]={0xff,0xff,0xff,0xff,0xff,0xff};

extern int hdev;

st = lc_updateKey(hdev, 1, n_keyA, n_ctrlW, n_keyB );// Modify password for sector 1

if(st)printf("Change key error");

else printf("Change key success");

## Ultralight/NTAG Card

**int lc_readTag(int icdev, unsigned char _Adr, unsigned char* _Data);**

Description: Read non-connected Mifare NTAG card

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, page address

_Data: Output, read data, 4 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char bufData[20];

extern int hdev;

st = lc_readTag(hdev, 8, bufData );// Read page 8 data

**int lc_writeTag(int icdev, unsigned char _Adr, unsigned char* _Data);**

Description: Write non-connected Mifare NTAG card

Parameter: icdev: Input, connection handle of reader / writer

_Adr: Input, page address

_Data: Output, write data, 4 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char bufData[5]={0x01,0x02,0x03,0x04};

extern int hdev;

st = lc_writeTag(hdev, 8, bufData );// Write page 8 data

**int lc_tag_authenPwd(int icdev, unsigned char* pKey_in);**

Description: Verify the password of the Mifare NTAG card

Parameter: icdev: Input, connection handle of reader / writer

pKey_in: Input, password, 4 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char defKey[] = {0xff, 0xff,0xff,0xff};

extern int hdev;

st = lc_tag_authenPwd(hdev, defKey);

**int lc_tag_setPwd(int icdev, unsigned char tagType,**

**unsigned char* pPwd_in,**

**unsigned char protectType,**

**unsigned char protectStartAddr,**

**unsigned char maxErrorVerify,**

**BOOL lockConfigAfterSet);**

Description: Set the password for the Mifare NTAG card

Parameter: icdev: Input, connection handle of reader / writer

tagType: Input, NTAG label model number, value and meaning as follows:

1- NTAG213

2- NTAG215

3- NTAG216

pPwd_in: Input, new password, 4 bytes

protectType: Input, data protection, 1-write time to verify, 2-read and write need to verify password first

protectStartAddr: Input, protect data start address page (minimum 4)

maxErrorVerify: Input, the cumulative number of consecutive validation errors, beyond which the label card will be locked; Settable values are 0~7, if set to 0, unlimited number of times

lockConfigAfterSet: Input, whether to lock configuration parameters, 1-lock, 0-no lock; Configuration parameters cannot be changed after lock

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char defKey[] = {0xff, 0xff,0xff,0xff};

extern int hdev;

st = lc_tag_setPwd(hdev,

1,//NTAG213

defKey,

1,// You need to verify your password before writing

4，// Protect from page 4

7，// Seven consecutive validation failures, lock card execution

0 // Do not lock configuration parameters

);

**int lc_halt(int icdev);**

Description: Stop the current Mifare card

Parameter: icdev: Input, connection handle of reader / writer

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_halt(hdev );

**int lc_rawExchange(int icdev,unsigned char slen,unsigned char * sbuff,unsigned char *rrlen,unsigned char * rbuff);**

Description: Device transceve command with Card

Parameter: icdev: Input, connection handle of reader / writer

slen: Input, size of message to be sent, number of bytes

sbuff: Input, send information

rlen: Output, size of returned information, number of bytes

rbuff: Output, return information

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char sendCmd[2]={0x60};//Get version of NTAG

unsigned char revLen;

unsigned char revInfo[128];

st = lc_rawExchange(hdev, 1, sendCmd, &revLen, rbuff);

## ISO14443-4 Agreement Card

**int lc_pro_reset(int icdev,unsigned char *rlen,unsigned char *rbuff);**

Description: CPU card reset operation

Parameter: icdev: Input, connection handle of reader / writer

rlen: Output, size of reset information, number of bytes

rbuff: Output, reset information

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char revLen;

unsigned char info[128];

st = lc_pro_reset(hdev, &revLen, rbuff);

**int lc_pro_commandlink(**

**int icdev,**

**unsigned int slen,**

**unsigned char * sbuff,**

**unsigned int *rlen,**

**unsigned char * rbuff,**

**unsigned char tt,**

**unsigned char FG**

**)**

Description: CPU card APDU instruction exchange

Parameter: icdev: Input, connection handle of reader / writer

slen: Input, size of message to be sent, number of bytes

sbuff: Input, send information (where data is ISO14443-4 compliant)

rlen: Output, size of returned information, number of bytes

rbuff: Output, return information

tt: Input, instruction transfer delay parameter, value range 1-9, recommended value 7

FG: Input, internal segment size for instruction transfer, less than 64, recommended value 59

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char sendCmd[5]={0x00,0x84,0x00,0x00,0x08};

unsigned char revLen;

unsigned char revInfo[128];

st = lc_pro_commandlink(hdev, 5, sendCmd, &revLen, rbuff, 7 , 59);

**int lc_getPBOC_PAN (int icdev,unsigned char slot, unsigned char getMode, char* szPAN);**

Description: Get UnionPay account

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, which slot card locate, 100-RF area, 0-Normal contact slot

getMode: Input, get mode, value 0 is read-only once per card, value 1 is read-repeatedly per card

szPAN: Output, get the UnionPay account, string

Return: < > 0 error code

=0 succeeded

Example: int st;

char szPAN[20];

extern int hdev;

st = lc_getPBOC_PAN (hdev, 100， 0, szPAN);

**int lc_readSSC(int icdev, unsigned char slot, char* pSSC_Info_out);**

Description: Read base information of Social-Security-Card of China

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, which slot card locate, 100-RF area, 0-Normal contact slot

pSSC_Info_out: Output, information got, format as string, every item divised with ‘\r\n’, item struct as bellow:

| Flag | Definition |
|---|---|
| ‘1’ | Specification Version |
| ‘2’ | Card UID |
| ‘3’ | Issue Date |
| ‘4’ | Expiration Date |
| ‘5’ | Issue Area Code |
| ‘8’ | Certification Of Citizenship |
| ‘9’ | Owner’s Name |
| ‘A’ | Owner’s Gender |
| ‘B’ | Owner’s Nation |
| ‘C’ | Birth Date |

Return: < > 0 error code

=0 succeeded

Example: int st;

char szInfo[256];

extern int hdev;

st = lc_readSSC (hdev, 0, szInfo);

## Ultralight-C

**int lc_ultralt_C_authen(int icdev, unsigned char* key);**

Description: Password verification for utralight-C card

Parameter: icdev: Input, connection handle of reader / writer

key: Input, password data, 16 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char arrKey[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

extern int hdev;

st = lc_ultralt_C_authen (hdev, arrKey);

**int lc_ultralt_C_setSafePage(int icdev, int ipage, BOOL readSec);**

Description: Set the Security page of utralight-c card

Parameter: icdev: Input, connection handle of reader / writer

ipage: Input, set security page number, value range: 4 ~ 41

readSec: Input, read permission, 0 - no password verification is required, 1 - password verification is required for read (Note: password verification is required for write operations)

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_ultralt_C_setSafePage (hdev, 4, 0);// Set 4 as security page

**int lc_ultralt_C_changePwd(int icdev, unsigned char* keyold, unsigned char* keynew);**

Description: Update the password of the utralight-C card

Parameter: icdev: Input, connection handle of reader / writer

keyold: Input, current (old) password, 16 bytes

keynew: Input, new password to change, 16 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char oldKey[16]={0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff};

unsigned char newKey[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st = lc_ultralt_C_changePwd (hdev, oldKey, newKey);

**int lc_ultralt_C_lockPage(int icdev, int flag);**

Description: Lock the page of utralight-C card

Parameter: icdev: Input, connection handle of reader / writer

flag: Input, page identification, values and meanings are as follows:

3-15 pages for 3-15 pages

41~43 for 41~43

16 corresponds to PAGE16_ 19, corresponding pages 16~19

20 corresponds to PAGE20_ 23, corresponding pages 20~23

24 corresponds to PAGE24_ 27, corresponding pages 24~27

28 corresponds to PAGE28_ 31, corresponding pages 28~31

32 corresponds to PAGE32_ 35, corresponding pages 32~35

36 corresponds to PAGE36_ 39, corresponding pages 36~39

44 corresponds to PAGE44_ 47, corresponding pages 44~47

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_ultralt_C_lockPage (hdev, 15);// Lock page 15

## Desfire

**int lc_authen_desfire(int icdev,unsigned char keyNo, char* key,unsigned char* sessionKey);**

Description: Desfire card 3DES verification password

Parameter: icdev: Input, connection handle of reader / writer

keyNo: Input, key number to verify

key: Input, 16 byte key

sessionKey: Output, session key returned after successful key validation

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char

curKey[17]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

unsigned char sessionKey[17];

st = lc_authen_desfire(hdev, 1, curKey, sessionKey);

**int lc_authen_desfire_aes(int icdev,unsigned char keyNo, char*key,unsigned char* sessionKey);**

Description: Desfire card AES authentication password

Parameter: icdev: Input, connection handle of reader / writer

keyNo: Input, key number to verify

key: Input, 16 byte key

sessionKey: Output, session key returned after successful key validation

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
char
curKey[17]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0
x0b,0x0c,0x0d,0x0e,0x0f,0x10};
unsigned char sessionKey[17];
st = lc_authen_desfire_aes(hdev, 1, curKey, sessionKey);
```

**int lc_getver_desfire(int icdev,unsigned char* rlen,unsigned char* version);**

Description: Desfire card obtains card manufacturing parameters

Parameter: icdev: Input, connection handle of reader / writer

rlen: Output, return the length of the data

version: Output, return card manufacturing parameters

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
unsigned char revLen;
unsigned char rData[50];
st = lc_getver_desfire(hdev, &revLen, rData);
```

**int lc_getAIDs_desfire(int icdev,unsigned char* rlen,unsigned char* AIDS);**

Description: Desfire Card Obtain Card Application Identification Number

Parameter: icdev: Input, connection handle of reader / writer

rlen: Output, return the length of the data

AIDS: Output, returns all applied identifiers

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char revLen;

unsigned char aids[50];

st = lc_getAIDs_desfire(hdev, &revLen, rData);

**int lc_selectApp_desfire(int icdev,unsigned char* AID);**

Description: Desfire Card Selects Current Application

Parameter: icdev: Input, connection handle of reader / writer

AID: Input, current application ID to select

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char aid[4]={0x01,0x00,0x00};

st = lc_selectApp_desfire(hdev, &revLen, rData);

**int lc_getKeySetting_desfire(int icdev,unsigned char* rlen,unsigned char*setbuf);**

Description: Desfire card gets password configuration

Parameter: icdev: Input, connection handle of reader / writer

rlen: Output, return the length of the data

setbuf: Output, apply (card) master key settings

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
unsigned char revLen;
unsigned char setPara[4];
st   = lc_getKeySetting_desfire(hdev, &revLen, setPara);
```

**int lc_getKeyver_desfire(int icdev,unsigned char keyNo,unsigned char* keyVer);**

Description: Desfire card gets password version information

Parameter: icdev: Input, connection handle of reader / writer

keyNo: Input, key number

keyVer: Output, obtained key version information

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
unsigned char keyVersion[4];
st   = lc_getKeyver_desfire(hdev, 1, keyVersion);
```

**int lc_createApp_desfire(int icdev,unsigned char*AID,unsigned char KeySetting,unsigned char NumOfKey);**

Description: Desfire card creation application

Parameter: icdev: Input, connection handle of reader / writer

AID: Input, application ID to create

KeySetting: Input, apply master key settings

Applying eight Bit bits of the master key means the following:

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| changeKey Access Rights Bit3 | changeKey Access Rights Bit2 | changeKey Access Rights Bit1 | changeKey Access Rights Bit0 | Configuration changeable | Free create/delete | Free directory list access | Allow change master key |

NumOfKey: Input, corresponding number of keys

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char aid[4]={0x02,0x00,0x00};

unsigned char setting = 0xef;

st = lc_createApp_desfire(hdev, aid, setting, 0x0e); //14 sets of keys

**int lc_delAID_desfire(int icdev,unsigned char* AID);**

Description: Desfire Card Delete Application

Parameter: icdev: Input, connection handle of reader / writer

AID: Input, application ID to be deleted

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char aid[];

st = lc_delApp_desfire(hdev, aid);

**int lc_changeKeySetting_desfire(int icdev,unsigned char newSet,char* sessionKey);**

Description: Desfire card changes master key settings

Parameter: icdev: Input, connection handle of reader / writer

newSet: Input, new key settings

Apply 8 Bit bits of master key, which means the following:

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| changeKey Access Rights Bit3 | changeKey Access Rights Bit2 | changeKey Access Rights Bit1 | changeKey Access Rights Bit0 | Configuration changeable | Free create/delete | Free directory list access | Allow change master key |

The card master key has 8 Bit bits, which means the following:

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| RFU | RFU | RFU | RFU | Configuration changeable | Free create/delete Withiout PICC master Key | Free directory list access without PICC master key | Allow change master key |

Each bit of the card-level key setting is described below:

Bit7-Bit4: Reserved, must be set to 0

Bit3: Encoding whether to allow modification of card master key setting information:

0: Configuration can no longer be modified (frozen)

1: If the card master key is certified, you can modify the configuration (default settings)

Bit2: Is it necessary authentication card master key when creating an application/deleting an application

0: Card master key must be successfully certified before an application can be created/deleted

1: Card master keys are not certified and allow applications to be established (default settings)

Successful authentication of the application master key or card master key is required before deleting the application operation

Bit1: Encoding whether an authentication card master key is required for application directory access

0 ： lc_getAIDs_desfire and lc_getKeySetting_desfire Require successful card certification

1 ： lc_getAIDs_desfire and lc_getKeySetting_desfire Authentication card master key is not required (default setting)

Bit0: Is the master key of the encoding card modifiable

0: Card master key can no longer be modified (frozen)

1: Card master key can be modified (current card master key must be certified, default setting)

sesssionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char set = 0x0f;

extern unsigned char sessionKey[];

st = lc_changeKeySetting_desfire(hdev, set, sessionKey);

**int lc_changeKey_desfire(int icdev,unsigned char* sessionKey,unsigned char* curKey,unsigned char keyNo,unsigned char* newkey);**

Description: Desfire Card Change Key (3DES)

Parameter: icdev: Input, connection handle of reader / writer

sessionKey: Input, session key

curKey: Input, current key

keyNo: Input, key number

newkey: Input, new key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char curKey[];

extern unsigned char sessionKey[];

unsigned char newKey[17]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a, 0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st = lc_changeKey_desfire(hdev, sessionKey, curKey,newKey);

**int lc_changeKey_desfire_aes(int icdev,unsigned char* sessionKey,unsigned char* curKey,unsigned char keyNo,unsigned char* newkey, unsigned char keyVersion);**

Description: The desfire card change key (AES) changes the card master key from DES to AES. One of the prerequisites for the success of this function is to verify the key corresponding to the key number, and then use the SessionKey that has been generated to verify the AES key and modify the AES key without obtaining key settings

Parameter: icdev: Input, connection handle of reader / writer

sessionKey: Input, session key

curKey: Input, current key

keyNo: Input, key number

newkey: Input, new key

keyVersion: Input, key version

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char curKey[];

extern unsigned char sessionKey[];

unsigned char

newKey[17]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,

0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st = lc_changeKey_desfire_aes(hdev, sessionKey, curKey,newKey);

**int lc_getFileIDs_desfire(int icdev,unsigned char* rlen,unsigned char* fileIDs);**

Description: Get all file ID numbers for the current application

Parameter: icdev: Input, connection handle of reader / writer

rlen: Output, return the length of the data

fileIDs: Output, returned file ID

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char revLen;

unsigned char fileIDs[20];

st = lc_getFileIDs_desfire(hdev, &revLen, fileIDs);

**int lc_getFileProper(int icdev,unsigned char fileNo,unsigned char* rlen,unsigned char * fileProper);**

Description: Desfire card gets file setup information

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

rlen: Output, return the length of the data

fileProper: Output, return file settings information

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char revLen;

unsigned char fileSet[20];

st = lc_getFileProper_desfire(hdev, &revLen, fileSet);

**int lc_changeFileSetting(int icdev,unsigned char fileNo,unsigned char comSet,unsigned char* accessRight,char* sessionKey);**

Description: Desfire card changes file settings

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

comSet: Input, data transmission, 0-clear text, 1-MAC code verification, 3-DES/3DES encryption

accessRight: Input, access rights

accessRight[0]: The lower half-byte is the permission set to modify access permissions, and the higher half-byte is the permission to read/write access the file

accessRight[1]: The lower half-byte is the write access to the file, the higher half-byte is the read access to the file

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sesssionKey[];

unsigned char comSetting = 0x01;// MAC Code Verification Method

unsigned char accessRights[3]={0x22,0x22};// Validate key 2 for read, write, read, write, and modify settings

st = lc_changeFileSetting(hdev, 0x01,comSetting,accessRights,sessionKey);

**int lc_createDataFile_desfire(int icdev,unsigned char fileNo,unsigned char ComSet,unsigned char* AccessRight,unsigned char* FileSize);**

Description: Desfire card creates standard data files

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

ComSet: Input, data transmission, 0-clear text, 1-MAC code verification, 3-DES/3DES encryption

AccessRight: Input, access rights

accessRight[0] : The lower half-byte is the permission set to modify access permissions, and the higher half-byte is the permission to read/write access the file

accessRight[1] : The lower half-byte is the write access to the file, the higher half-byte is the read access to the file

FileSize: Input, file size

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char comSetting = 0x00;// Clear text transmission

unsigned char accessRights[3]={0x22,0x22};// Validate key 2 for read, write, read, write, and modify settings

unsigned char fSize[4] = {0x20,0x00,0x00};//32 byte length

st = lc_createDataFile_desfire(hdev, 0x01,comSetting,accessRights,fSize);

**int lc_createBackupDataFile_desfire(int icdev,unsigned char fileNo,unsigned char ComSet,unsigned char* AccessRight,unsigned char* FileSize);**

Description: Desfire card creates backup data files

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

ComSet: Input, data transmission, 0-clear text, 1-MAC code verification, 3-DES/3DES encryption

AccessRight: Input, access rights

accessRight[0] : The lower half-byte is the permission set to modify access permissions, and the higher half-byte is the permission to read/write access the file

accessRight[1] : The lower half-byte is the write access to the file, the higher half-byte is the read access to the file

FileSize: Input, file size

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char comSetting = 0x00;// Clear text transmission

unsigned char accessRights[3]={0x22,0x22};// Validate key 2 for read, write, read, write, and modify settings

unsigned char fSize[4] = {0x20,0x00,0x00};//32 byte length

st = lc_createBackupDataFile_desfire(hdev, 0x01,comSetting,accessRights,fSize);

**int lc_createValueFile_desfire(int icdev,unsigned char fileNo,unsigned char ComSet,unsigned char* AccessRight,unsigned char* lowerLimit,unsigned char* upperLimit,unsigned char* value,unsigned char creditEnabled);**

Description: Desfire card create value file

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

ComSet: Input, data transmission, 0-clear text, 1-MAC code verification, 3-DES/3DES encryption

AccessRight: Input, access rights

accessRight[0] : The lower half-byte is the permission set to modify access permissions, and the higher half-byte is the permission to read/write access the file

accessRight[1] : The lower half-byte is the write access to the file, the higher half-byte is the read access to the file

lowerLimit: Input, minimum

upperLimit: Input, maximum

value: Input, set initial value

creditEnable: Input, whether limit is supported

Return: < > 0 error code

=0 succeeded

Example: int st;

```
extern int hdev;
unsigned char comSetting = 0x03;// DES Encrypted Transport
unsigned char accessRights[3]={0x22,0x22};// Validate key 2 for read, write, read, write, and modify settings
unsigned char lower[4]={0x00,0x00,0x00,0x00};// Minimum 0
unsigned char upper[4]={0xff,0xff,0xff,0xff,0x00};// Maximum is 0xFFFFFF
unsigned char value[4]={0x32,0x00,0x00,0x00};// Initial value set to 50 (0x32)
unsigned char bEnable = 0x01;// Supports limitation
st = lc_createValueFile_desfire(hdev,
```

0x02,comSetting,accessRights,lower,upper,value,bEnable);

**int lc_createCsyRecord_desfire(int icdev,unsigned char fileNo,unsigned char comSet,unsigned char* AccessRight,unsigned char* RecordSize,unsigned char* MaxNum);**

Description: Desfire card creates a circular record file

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

ComSet: Input, data transmission, 0-clear text, 1-MAC code verification, 3-DES/3DES encryption

AccessRight: Input, access rights

accessRight[0] : The lower half-byte is the permission set to modify access permissions, and the higher half-byte is the permission to read/write access the file

accessRight[1] : The lower half-byte is the write access to the file, the higher half-byte is the read access to the file

RecordSize: Input, length of each record

MaxNum: Input, length of each record

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char comSetting = 0x03;// DES Encrypted Transport

unsigned char accessRights[3]={0x22,0x22};// Validate key 2 for read, write, read, write, and modify settings

unsigned char recordLen[3]={0x20,0x00,0x00};// Each record defines 32 bytes

unsigned char number[3]={0x10,0x00,0x00};// File definition up to 16

records

st = lc_createCsyRecord_desfire(hdev, 0x03,comSetting,accessRights,recordLen,number);

**int lc_delFile_desfire(int icdev,unsigned char fileNo);**

Description: Desfire card deletes files

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_delFile_desfire(hdev,0x03);

**int lc_write_desfire(int icdev,unsigned char fileNo,unsigned int offset,unsigned int length,unsigned char* data,char*sessionKey);**

Description: Desfire card write data file

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

offset: Input, offset address

length: Input, the length of the data to be written (number of bytes)

data: Input, data to be written

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sessionKey[];

unsigned char wData[]={0x11,0x22,0x33,0x44,0x55,0x66,0x77,0x88};

st=lc_write_desfire(hdev,0x01,0x00,0x08,wData,sessionKey);

Note: If the operation file is a backup data file, the lc_commitTransfer_desfire directive submission operation must be performed for the written data to be valid. You can also call the lc_abortTransfer_desfire directive to cancel all modifications

**int lc_read_desfire(int icdev,unsigned char fileNo,unsigned int offset,unsigned int length,unsigned char* revData,char*sessionKey);**

Description: Desfire card reads data files

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

offset: Input, offset address

length: Input, the length of the data to be read (number of bytes)

revData: Input, data to be read

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sessionKey[];

unsigned char rData[10];

st=lc_read_desfire(hdev,0x01,0x00,0x08,rData,sessionKey);

**int lc_getvalue_desfire(int icdev,unsigned char fileNo,unsigned int* value,char* sessionKey);**

Description: Desfire card get value

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

value: Output, read value

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sessionKey[];

unsigned int value;

st=lc_getvalue_desfire(hdev,0x02,&value,sessionKey);

**int lc_credit_desfire(int icdev,unsigned char fileNo,unsigned int value,char*sessionKey);**

Description: Desfire card increment

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

value: Input, the value to be added

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sessionKey[];

unsigned int value = 100;

st=lc_credit_desfire(hdev,0x02,value,sessionKey);

Note: After this function succeeds, the lc_commitTransfer_desfire function must be called once for the operation to take effect

**int lc_debit_desfire(int icdev,unsigned char fileNo,unsigned int value,char*sessionKey);**

Description: Desfire card subtraction

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

value: Input, value to reduce

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sessionKey[];

unsigned int value = 100;

st=lc_debit_desfire(hdev,0x02,value,sessionKey);

Note: After this function succeeds, the lc_commitTransfer_desfire function must be called once for the operation to take effect

**int lc_writeRecord_desfire(int icdev,unsigned char fileNo,unsigned int offset,unsigned int length,unsigned char* data,char*sessionKey);**

Description: The desfire card writes a record to the record file

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

offset: Input, offset position

length: Input, write data length

data: Input, data to be written

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sessionKey[];

unsigned char wData[8]={0x11,0x22,0x33,0x44,0x55,0x66,0x77,0x88};

st=lc_writeRecord_desfire(hdev,0x03,0,8,wData,sessionKey);

Note: After this function succeeds, the lc_commitTransfer_desfire function must be called once for the operation to take effect

**int lc_readRecord_desfire(int icdev,unsigned char fileNo,unsigned int offset,unsigned int length,unsigned char* revData,unsigned int* SgRecordlen,unsigned int*rlen,char*sessionKey);**

Description: Desfire card reads record file

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

offset: Input, offset position, start record number

length: Input, number of records read from offset address

revData: Output, read data

SgRecordlen: Output, length of a single record

rlen: Output, total length of read data

sessionKey: Input, session key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

extern unsigned char sessionKey[];

unsigned char rData[100];

unsigned int sgLen;

unsigned int revLen;

st=lc_readRecord_desfire(hdev,0x03,0,1,rData,&sgLen,sessionKey);

Note: If both length and offset are set to 0, all records will be read out.

**int lc_clearRecord_desfire(int icdev,unsigned char fileNo);**

Description: The desfire card clears the data of the record file

Parameter: icdev: Input, connection handle of reader / writer

fileNo: Input, file identification

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_clearRecord_desfire(hdev,0x03);

**int lc_commitTransfer_desfire(int icdev);**

Description: Desfire card submits a data transfer

Parameter: icdev: Input, connection handle of reader / writer

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_commitTransfer_desfire(hdev);

**int lc_abortTransfer_desfire(int icdev);**

Description: Desfire card cancels one data transfer

Parameter: icdev: Input, connection handle of reader / writer

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_abortTransfer_desfire(hdev);

**int lc_formatPICC_desfire(int icdev);**

Description: Desfire card formatting

Parameter: icdev: Input, connection handle of reader / writer

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_formatPICC_desfire(hdev);

## Mifare PLus

**int lc_MFPlusL0_WritePerso(int icdev, unsigned char* key, unsigned short keyAddr);**

Description: Card Personalization Settings

Parameter: icdev: Input, connection handle of reader / writer

key: Input, key or data to set, 16 bytes

keyAddr: Input, the block number corresponding to the key or data, meaning as shown in Table 1 of Attachment

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char mastKey[17]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL0_WritePerso(hdev,mastKey, 0x9000);

Note: After successful submission, the card will be automatically upgraded from level 0 to level 1

**int lc_MFPlusL0_CommitPerso(int icdev);**

Description: Submit card personalization settings

Parameter: icdev: Input, connection handle of reader / writer

key: Input, key or data to set, 16 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_MFPlusL0_CommitPerso(hdev);

Note: After successful submission, the card will be automatically upgraded from level 0 to level 1

**int lc_MFPlusL1_AuthenKeyL1(int icdev,unsigned char* key);**

Description: Level 1 card verifies Level 1 key

Parameter: icdev: Input, connection handle of reader / writer

key: Input, Level 1 key, 16 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char keyL1[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL1_AuthenKeyL1(hdev,keyL1);

**int lc_MFPlusL1_SwitchToL2(int icdev,unsigned char* key);**

Description: Level 1 card verifies Level 2 upgrade key and upgrades to Level 2

Parameter: icdev: Input, connection handle of reader / writer

key: Input, Level 2 upgrade key, 16 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char keyL2[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL1_SwitchToL2 (hdev,keyL2);

**int lc_MFPlusL1_SwitchToL3(int icdev,unsigned char* key);**

Description: Level 1 card verifies Level 3 upgrade key and upgrades to Level 3

Parameter: icdev: Input, connection handle of reader / writer

key: Input, Level 3 upgrade key, 16 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char keyL3[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL1_SwitchToL3 (hdev,keyL3);

**int lc_MFPlusL2_SwitchToL3(int icdev,unsigned char* key);**

Description: Level 2 card verifies Level 3 upgrade key and upgrades to Level 3

Parameter: icdev: Input, connection handle of reader / writer

key: Input, Level 3 upgrade key, 16 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char keyL3[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL2_SwitchToL3 (hdev,keyL3);

**int lc_MFPlusL3_AuthenL3Key(int icdev,unsigned char* key, unsigned short keyAddr);**

Description: Level 3 card verifies Level 3 key

Parameter: icdev: Input, connection handle of reader / writer

key: Input, Level 3 upgrade key, 16 bytes

keyAddr: Input, block number corresponding to key or data, as detailed in Appendix 1

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char key[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL3_AuthenL3Key (hdev,key,0x9000);// Verify master key

**int lc_MFPlusL3_AuthenSectorKey(int icdev,unsigned char keyType,unsigned int sectorNo,unsigned char* key);**

Description: Level 3 card verifies sector key

Parameter: icdev: Input, connection handle of reader / writer

keyType: Input, key type, 1-KeyA, 2-KeyB

sectorNo: Input, key number 0~31 or 0~39

key: Input, sector key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char key[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL3_AuthenSectorKey (hdev,1,1,key);// Verify 1 sector KeyA key

**int lc_MFPlusL3_UpdateKey(int icdev, unsigned int keyAddr,unsigned**

**char* newKey);**

Description: Level 3 card update key

Parameter: icdev: Input, connection handle of reader / writer

keyAddr: The block number corresponding to the key or data, as detailed in Appendix 1

newKey: Input, new key

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char key[16]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10};

st=lc_MFPlusL3_UpdateKey (hdev,0x9000,key);// Update master key

**int lc_MFPlusL3_ReadWithPlain(int icdev, unsigned int blkAddr,unsigned char blkNum,unsigned char* rdata);**

Description: Level 3 card-clear reading block data

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, starting block number

blkNum: Input, number of blocks to read (typically 1-3)

rdata: Output, received data

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char rData[49];

st=lc_MFPlusL3_ReadWithPlain (hdev,4,3,readData);

**int lc_MFPlusL3_WriteWithPlain(int icdev, unsigned int**

**blkAddr,unsigned char blkNum,unsigned char* wdata);**

Description: Level 3 card plain text mode for block data writing

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, starting block number

blkNum: Input, the number of blocks to write (typically 1-3)

wdata: Input, data to be written

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char wData[49]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f,0x30};

st=lc_MFPlusL3_WriteWithPlain (hdev,4,3,wData);

**int lc_MFPlusL3_ReadWithEncrypt(int icdev, unsigned int blkAddr,unsigned char blkNum,unsigned char* rdata);**

Description: Level 3 card encryption for reading block data

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, starting block number

blkNum: Input, number of blocks to read (typically 1-3)

rdata: Output, received data

flag: Input, flag, 0 means data is returned after decryption, 1 means data is returned directly without decryption

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char rData[49];

st=lc_MFPlusL3_ReadWithEncrypt (hdev,4,3,readData, 0);

**int lc_MFPlusL3_WriteWithEncrypt(int icdev, unsigned int blkAddr,unsigned char blkNum,unsigned char* wdata,unsigned char flag);**

Description: Write block data with level 3 card encryption

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, starting block number

blkNum: Input, number of blocks to read (typically 1-3)

wdata: Input, data to be written

flag: Input, flag, 0 means data needs to be encrypted before writing, 1 means data has been encrypted

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char wData[49]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f,0x30};

st=lc_MFPlusL3_WriteWithEncrypt (hdev,4,3,wData);

**int lc_MFPlusL3_InitVal(int icdev, unsigned int blkAddr,unsigned long value);**

Description: Level 3 card initialization block is value block

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, block number

value: Input, set initialization value

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_MFPlusL3_InitVal (hdev,4,1000);

**int lc_MFPlusL3_ReadVal(int icdev, unsigned int blkAddr,unsigned long* value);**

Description: Level 3 card reading block (check balance)

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, block number

value: Output, read value

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned lont value;

st=lc_MFPlusL3_ReadVal (hdev,4,&value);

**int lc_MFPlusL3_Increment(int icdev, unsigned int blkAddr,unsigned long value);**

Description: 3-level card value-added (recharge)

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, block number

value: Output, value to add

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_MFPlusL3_Increment (hdev,4,100);

**int lc_MFPlusL3_Decrement(int icdev, unsigned int blkAddr,unsigned long value);**

Description: Level 3 card impairment (deduction)

Parameter: icdev: Input, connection handle of reader / writer

blkAddr: Input, block number

value: Output, value to reduce

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st=lc_MFPlusL3_Decrement (hdev,4,10);

## ISO14443-B Protocol API Details

**int lc_findTypeB (int icdev,unsigned char fSelectCard,unsigned char *response_out, unsigned char* resLen_out);**

Description: Find a TypeB IC card

Parameter: icdev: Input, connection handle of reader / writer

fSelectCard: Input, a flag determine whether select this card when found one.

response_out: Output, card response data (structure 50+PUPI[4]+AppData[4]+Protocol Info[3])

resLen_out: Output, card response data length

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char resetInfo[9];

unsigned char infoSize;

extern int hdev;

st = lc_findTypeB(hdev,1, resetInfo,&infoSize );

**int lc_typeB_command (**

**int icdev,**

**unsigned int slen,**

**unsigned char * sbuff,**

**unsigned int *rlen,**

**unsigned char * rbuff)**

Description: TypeB Card APDU Instruction Exchange

Parameter: icdev: Input, connection handle of reader / writer

slen: Input, size of message to be sent, number of bytes

sbuff: Input, send information

rlen: Output, size of returned information, number of bytes

rbuff: Output, return information

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char sendCmd[5]={0x00,0x36,0x00,0x00,0x08};

unsigned char revLen;

unsigned char revInfo[128];

st = lc_typeB_command (hdev, 5, sendCmd, &revLen, rbuff);

## ISO15693 Protocol API Details

**int lc_find15693 (int icdev,unsigned char reqMode,unsigned char *response_out, unsigned char* resLen_out);**

Description: Find an ISO15693 protocol IC card

Parameter: icdev: Input, connection handle of reader / writer

reqMode: Input, card-seeking mode, set to 36H for each search card, set to 16H for each search card

response_out: Output, find the UID of the card. If you search for multiple cards, the UIDs are listed in turn, each group of UIDs is 8 bytes long

resLen_out: Output, card response data length

Return: < > 0 error code

=0 succeeded

Example: int st;

unsigned char uid[9];

unsigned char rdataLen;

extern int hdev;

st = lc_find15693(hdev,0x36,uid, &rdataLen);

**int lc_15693_command_custom (**

**int icdev,**

**unsigned char cmd,**

**unsigned char MfgCode,**

**unsigned char* pUID,**

**unsigned int sParaLen,**

**unsigned char * psPara,**

**unsigned int *rParaLen,**

**unsigned char * rPara)**

Description: 15693 Catalog Special User Instruction Exchange

Parameter: icdev: Input, connection handle of reader / writer

cmd : Input, function instruction code

MfgCode : Input, card manufacturer code

pUID: Input, UID code of the card being operated on

sParaLen : Input, parameter length sent (number of bytes)

psPara : Input, parameter to function instruction

rParaLen : Output, return the size of the parameter, number of bytes

rPara : Output, returns parameter information

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char mfgCode = 0x04;// Enzhip ICode Type

unsigned char commandCode = 0xB2;// Get random number instruction code

extern unsigned char uid[9];

unsigned int revLen;

unsigned char revInfo[128];

st = lc_15693_command_custom (hdev, mfgCode, commandCode,uid,0,NULL,&revLen,revInfo);

**int lc_15693_select_uid (int icdev,unsigned char* pUID);**

Description: Select 15693 cards to operate

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to be operated on

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

st = lc_15693_select_uid (hdev,uid);

**int lc_15693_readBlock (int icdev,unsigned char* pUID,unsigned char startBlock, unsigned char blockNum, unsigned char* rlen, unsigned char* rbuffer);**

Description: Read block data of 15693 cards

Parameter: icdev: Input, connection handle of reader / writer

pUID : Input, UID of the card to read

startBlock : Input, start block number of reading card

blockNum : Input, number of blocks to read

rlen : Output, length of read data

rbuffer : Output, read block data

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char rData[17];

unsigned char rLen;

unsigned char startBlock = 8;

unsigned char blockNumber = 1;

st = lc_15693_readBlock (hdev,uid,startBlock,blockNumber,&rLen, rData);

**int lc_15693_writeBlock (int icdev,unsigned char* pUID,unsigned char startBlock, unsigned char blockNum, unsigned char wlen, unsigned char* wbuffer);**

Description: Blocks that write data to 15693 cards

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

startBlock: Input, start block number of writing card

blockNum: Input, number of blocks to write

wlen: Input, length of data to be written

wbuffer: Input, data to be written

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char wData[5]={0x11,0x22,0x33,0x44};

unsigned char wLen = 4;

unsigned char startBlock = 8;

unsigned char blockNumber = 1;

st = lc_15693_writeBlock (hdev,uid,startBlock,blockNumber,wLen, wData);

**int lc_15693_lock_block(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char block);**

Description: Lock the specified block of 15693 card

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

block: Input, block number to lock

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

st = lc_15693_lock_block (hdev,uid,8);

**int lc_15693_write_afi(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char AFI);**

Description: Update 15693 card AFI parameters

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

AFI: Input, new value of AFI parameter

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char nAFI = 0x00;

st = lc_15693_write_afi (hdev,uid,nAFI);

**int lc_15693_lock_afi(**

**int icdev,**

**unsigned char *pUID);**

Description: Lock 15693 card AFI parameters

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

st = lc_15693_lock_afi (hdev,uid);

**int lc_15693_write_dsfid(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char DSFID);**

Description: Update 15693 card DSFID parameters

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

DSFID: Input, write updated DSFID

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char nDSFID =0x00;

st = lc_15693_write_dsfid (hdev,uid,nDSFID);

**int lc_15693_lock_dsfid(**

**int icdev,**

**unsigned char *pUID);**

Description: Lock 15693 card DSFID parameters

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

st = lc_15693_lock_dsfid (hdev,uid);

**int lc_15693_get_systemInfo(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char *rlen,**

**unsigned char *rbuffer);**

Description: Get 15693 card system parameters

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

rlen: Output, get the length of the data

rbuffer: Output, get data

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char _rlen;

unsigned char _rbuf[32];

st = lc_15693_get_systemInfo (hdev,uid,&_rlen, _rbuf);

**int lc_15693_get_securityInfo(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char startBlock,**

**unsigned char blockNum,**

**unsigned char *rlen,**

**unsigned char *rbuffer);**

Description: Get 15693 card security parameters

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

startBlock: Input, the starting (block) address of the block to query

blockNum: Input, number of blocks to query

rlen: Output, block security information length

rbuffer: Output, get block security information, 00h-corresponding block not included, 01h-corresponding block protected

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char _rlen;

unsigned char _rbuf[32];

st = lc_15693_get_securityInfo (hdev,uid,4,2, &_rlen, _rbuf);

**int lc_15693set_password(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char pwdType,**

**unsigned char* pPwd);**

Description: Set (verify) the password of 15693 card

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

pwdType: Input, Password Type, 1-Read Password, 2-Write Password,

4-Privacy Password, 8-Destory Password, 16-EAS Password

pPwd: Input, the password to verify

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char pwd[]={0x11,0x22,0x33,0x44};

st = lc_15693set_password (hdev,uid,1,pwd);

**int lc_15693_write_password(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char pwdType,**

**unsigned char* pPwd);**

Description: Reset password of 15693 card

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

pwdType: Input, Password Type, 1-Read Password, 2-Write Password, 4-Privacy Password, 8-Destory Password, 16-EAS Password

pPwd: Input, password to reset

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char pwd[]={0x11,0x22,0x33,0x44};

st = lc_15693_write_password (hdev,uid,1,pwd);

**int lc_15693_icode_64bitProtect(int icdev,unsigned char *pUID);**

Description: ICode card 64-bit protection

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

st = lc_15693_icode_64bitProtect (hdev,uid);

**int lc_15693_protect_page(**

**int icdev,**

**unsigned char *pUID,**

**unsigned char iPage,**

**unsigned char protectStatus);**

Description: ICode card protection page

Parameter: icdev: Input, connection handle of reader / writer

pUID: Input, UID of the card to write

iPage: Input, the page number to protect

protectStatus: Input, protection status, values and meanings are as follows:

00h - Public (unprotected)

01h - read and write protection with read password

10h - Use Write Password to Protect Writing

11h - Use read password for read protection, write password for write protection

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char page = 8;

unsigned char protectSt = 0x10;// Use Write Password to Protect Writing

st = lc_15693_protect_page (hdev,uid, page, protectSt);

## ISO18092/TypeF Protocol API Details

**int lc_findTypeF(int icdev,unsigned char *pIDm_out, unsigned char* pPMm_out);**

Description: Find cards for TypeF protocol

Parameter: icdev: Input, connection handle of reader / writer

pIDm: Output, return card manufacturer UID information

pPMm_out: Output, return vendor parameter information

Return: < > 0 error code

=0 succeeded

Example: int st;

extern unsigned char uid[9];

extern int hdev;

unsigned char bufID[9];

unsigned char bufPara[9];

st = lc_findTypeF (hdev,bufID,bufPara);

**int lc_typeF_command**

**(**

**int icdev,//Handle of device**

**unsigned int slen,//The length of data to send**

**unsigned char * sbuff,//Data to send**

**unsigned int *rlen,//The length of data received**

**unsigned char * rbuff//The data received**

**);**

Description: Card Exchange Instruction for TypeF Protocol

Parameter: icdev: Input, connection handle of reader / writer

slen: Input, length of instruction data sent (number of bytes)

sbuff: Input, send instructions (refer to the card manual for instructions)

rlen: Output, length of response data received (number of bytes)

rbuff: Output, response data received

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char i;

unsigned char bufCmd[9]={0xC0};

unsigned int resLen;

unsigned char bufResp[9];

extern unsigned char IDm[];//TypeF 的 IDm

```
for(i=0;i<8;i++)bufCmd[1+i]=IDm[i];
st   = lc_findTypeF (hdev,9, bufCmd, &resLen, bufResp);
```

## UHF API Details

**int  lc_uhf_inventory(**

**int icdev,**

**unsigned short scanTime,**

**unsigned char *response_out,**

**unsigned char* resLen_out**

**);**

Description: Inventory (Find) all the UHF-tag near around

Parameter: icdev: Input, connection handle of reader / writer

response_out: Output, return EPC information of each tag, which struct of data as below:

EPC1-Len + EPC1 +{EPC2-Len + EPC2+…}

resLen_out: Output, return the length of response_out

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char bufEPCs[50*12];
unsigned char retLen;
st   = lc_uhf_inventory (hdev,200, bufEPCs, &retLen);
```

**int  lc_uhf_readTag**

**(**

**int icdev,**

**unsigned char bTagBank,**

**unsigned int addrWord,**

**unsigned int lenWord,**

**unsigned char *pPwd,**

**unsigned int *rlen,**

**unsigned char * rbuff**

**);**

Description: Read UHF-tag

Parameter:

icdev: Input, connection handle of reader / writer

bTagBank: Input, indicate which bank want to be read, vaule see to Appendix table-5

addrWord: Input, the start address of data to be read, in Word (16 bits)

lenWord: Input, the length of data to be read, in Word (16 bits)

pPwd: Input, the access password used to read

rlen: Output, return the length of rbuff, unit as byte, shall be 2*lenWord if success.

rbuff: Output, return the data readed

Return: < > 0 error code

=0 success

Example:

```
int   st;
extern int hdev;
unsigned char password[4]={0x00,0x00,0x00,0x00};
unsigned char buf_r[2*10];
unsigned char retLen;
st   = lc_uhf_readTag (hdev,3, 0, 2, password, &retLen, buf_r);//Read User Bank
```

**int lc_uhf_writeTag**

**(**

**int icdev,**

**unsigned char bTagBank,**

**unsigned int addrWord,**

**unsigned int lenWord,**

**unsigned char *pPwd,**

**unsigned char * wbuff**

**);**

Description: Write data to UHF-tag

Parameter:

icdev: Input, connection handle of reader / writer

bTagBank: Input, indicate which bank want to be write, vaule see to Appendix table-5

addrWord: Input, the start address of data to be write, in Word (16 bits)

lenWord: Input, the length of data to be write, in Word (16 bits)

pPwd: Input, the access password used to write

wbuff: Input, the data wait for writting

Return: < > 0 error code

=0 success

Example:

```
int   st;
extern int hdev;
unsigned char password[4]={0x00,0x00,0x00,0x00};
unsigned char buf_w[2*2]={0x00,0x01, 0x00,0x02};
st   = lc_uhf_writeTag (hdev,3, 0, 2, password, buf_w);//Write User Bank
```

**int lc_uhf_lockTag**

**(**

**int icdev,**

**unsigned char bTagBank,**

**unsigned char bLockType,**

**unsigned char *pPwd**

**);**

Description: Lock special Bank of UHF-tag

Parameter:

icdev: Input, connection handle of reader / writer

bTagBank: Input, indicate which bank want to be lock, vaule see to Appendix table-5

bLockType: Input, lock type, values see below:

- Open: 0x00, not lock
- Temporary Lock: 0x01, lock temporarily
- Open Forever: 0x02, Keep open, never been changed
- Lock Forever: 0x03, Keep lock, never been changed

pPwd: Input, the access password used to lock

Return: < > 0 error code

=0 success

Example:

```
int   st;
extern int hdev;
unsigned char password[4]={0x00,0x00,0x00,0x00};
st   = lc_uhf_lockTag (hdev,1, 2, password);//Lock EPC Bank
```

**int lc_uhf_killTag**

**(**

**int icdev,**

**unsigned char *pPwd**

**);**

Description: Kill UHF-tag

Parameter:

icdev: Input, connection handle of reader / writer

pPwd: Input, the access password used to kill tag, it shall be changed and not equal the default password.

Return: < > 0 error code

=0 success

Example:

```
int   st;
extern int hdev;
unsigned char password[4]={0x01,0x02,0x03,0x04};
st   = lc_uhf_killTag (hdev, password);
```

# Contact Card Functional API Details

**int lc_iccGetCardState(int icdev,unsigned char slot, unsigned char *pSt);**

Description: Get the state which indicate if a card in slot

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

pSt：output, 1- yes, 0-no

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char icCardState;
st   = lc_iccGetCardState (hdev,0,&icCardState);
if( st == 0){
  if(icCardState == 0)printf("No card in main slot");
  else printf("Card in main slot");
}
```

**int lc_iccSelCardType( int icdev,unsigned char slot,unsigned int cardType);**

Description: Select the card type inserting or will be insert

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

cardType：input, the card type see to appendix

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
st   = lc_iccSelCardType (hdev,0,0x0C);//Select ISO7816 card
```

**int lc_iccSetBaud(int icdev, unsigned char slot, unsigned int baud);**

Description: Setting the baud-rate when communicate with 7816 card, 9600 defaultly using after boot of device.

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

baud: Input,  the baud rate to set

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
st   = lc_iccSetBaud (hdev,0, 9600);
```

**int lc_iccGetATR(int icdev,unsigned char slot, unsigned char *response_out, unsigned char* resLen_out);**

Description: Reset the ISO7816 card and get ATR information.

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

response_out：output, the information of ATR

resLen_out: output, the length of ATR in bytes

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char rATR[33];
unsigned char sizeATR;
st   = lc_iccGetATR (hdev,0,rATR,&sizeATR);
```

**int lc_iccPPS(int icdev, unsigned char slot, unsigned char* pPara);**

Description: Do protocol and parameters exchange with card. This function shall be invoke just after lc_iccGetATR if necessary.

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

pPara: Input, 6 bytes , PPS bytes, start with PPS0, byte useless set to 0

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char para[6]={0x10,0x11,0,0,0,0};//Set to 9600 pbs
st   = lc_iccPPS (hdev,0,para);
```

**int lc_readSIM(int icdev, unsigned char slot, char* pSIM_Info_out);**

Description: Read base information of SIM Card.

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, which slot card locate, 100-RF area, 0-Normal contact slot

pSIM_Info_out: Output, information got, format as string, every item divised with ‘\r\n’, item struct as bellow:

| Flag | Definition |
|---|---|
| ‘1’ | IMSI |
| ‘2’ | ICCID |

Return: < > 0 error code

=0 succeeded

Example: int st;

char szInfo[256];

extern int hdev;

st = lc_readSIM (hdev, 0, szInfo);

**int lc_icc_APDU(int icdev,unsigned char slot,unsigned int slen,unsigned char * sbuff,unsigned int *rlen,unsigned char * rbuff);**

Description: Send command to ISO7816 card, and receive reply (APDU)

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

slen：input, the length of command in byte for sending

sbuff：input, content of command

rlen：output, the length of reply

rbuff：output, reply data

Return: < > 0 error code

=0 succeeded

Example:

int st;

extern int hdev;

unsigned char cmdBuf[5]={0x00,0x84,0x00,0x00,0x08};

```
unsigned char reply[256];
unsigned int replySize;
st  = lc_icc_APDU(hdev,0,5,cmdBuf,&replySize,reply);
```

**int lc_icc_ReadMem(int icdev,unsigned char slot,unsigned int address,unsigned int rLen,unsigned char * rbuff);**

Description: Read logic card or memory card

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

address：input, the start address for reading

rLen：input, the length of data for reading

rbuff：output, data readed

Return: < > 0 error code

=0 succeeded

Example:

```
int  st;
extern int hdev;
unsigned char rData[256];
st  = lc_icc_ReadMem(hdev,0,0,16, rData);
```

**int lc_icc_WriteMem(int icdev,unsigned char slot,unsigned int address,unsigned int wLen,unsigned char * wbuff);**

Description: Write data to logic card or memory card at special address

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

address：input, the start address when writting

wLen：input, the length data for writing in bytes

wbuff：input, the data for writting

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char wData[6]={0x01,0x02,0x03,0x04,0x05,0x06};
st   = lc_icc_WriteMem(hdev,0,0,6, wData);
```

**int lc_icc_VerifyUserPass(int icdev,unsigned char slot,unsigned int passLen,unsigned char * pPassIn);**

Description: Verify password of user zone

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

passLen：input, the length of password for verifying(3-byte case of 4442, 2-byte case of 4428)

pPassIn：input, the pass for verifing

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char pass[3]={0xff,0xff,0xff};
st   = lc_icc_VerifyUserPass(hdev,0,3,pass);
```

**int lc_icc_ReadErrorCounter(int icdev, unsigned char slot,unsigned char* pCnt);**

Description: Read the remain number of chances of verify

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

pCnt: output, the remain number got

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char remainVerifyTimes;
st   = lc_icc_ReadErrorCounter(hdev,0,& remainVerifyTimes);
```

**int lc_icc_UpdateUserPass(int icdev,unsigned char slot,unsigned int passLen,unsigned char * pPassIn);**

Description: Update the password of user zone

Parameter: icdev: Input, connection handle of reader / writer

slot: Input, the slot number, see to appendix

passLen：input, the length of new password(3-byte case of 4442, 2-byte case of 4428)

pPassIn: input, the new password

Return: < > 0 error code

=0 succeeded

Example:

```
int   st;
extern int hdev;
unsigned char pass[3]={0xff,0xff,0xff};
st   = lc_icc_UpdateUserPass(hdev,0,3,pass);
```

# Net Functional API Details

**int lc_setNet_serverIP(int icdev, unsigned char* pIP_in);**

Description: Set the server IP address for Net functionality

Parameter: icdev: Input, connection handle of reader / writer

pIP_in: Input, IP address of server, 4 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char bufIP[5]={192,168,1,16};

st = lc_setNet_serverIP (hdev,bufIP);

**int lc_getNet_serverIP(int icdev, unsigned char* pIP_out);**

Description: Get the currently set IP address of the Net Connection Server

Parameter: icdev: Input, connection handle of reader / writer

pIP_out: Input, get the IP address of the server, 4 bytes

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char bufIP[5];

st = lc_getNet_serverIP (hdev,bufIP);

**int lc_setNet_serverPort(int icdev, int port);**

Description: Set the listening port of the Net connection server

Parameter: icdev: Input, connection handle of reader / writer

port: Input, server's listening port

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_serverPort (hdev, 51006);

**int lc_getNet_serverPort(int icdev, int* pPort);**

Description: Get the listening port of the currently set Net connection server

Parameter: icdev: Input, connection handle of reader / writer

pPort: Output, get the server's listening port

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

int port;

st = lc_setNet_gerverPort (hdev, &port);

**int lc_setNet_URL(int icdev, unsigned char* pURL_in);**

Description: Set the URL to connect when Net is sent in HTTP format

Parameter: icdev: Input, connection handle of reader / writer

pURL_in: Input, URL to connect, string

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char* szURL = "http://134.175.147.135:1234/home.html";

st = lc_setNet_URL (hdev, szURL);

**int lc_getNet_URL(int icdev, unsigned char* pURL_out);**

Description: Get the URL to connect to when the currently set Net is sent in HTTP format

Parameter: icdev: Input, connection handle of reader / writer

pURL_out: Output, URL obtained, string format

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char szURL[265];

st = lc_getNet_URL (hdev, szURL);

**int lc_setNet_heartBeatInterval(int icdev, int cnt_100ms);**

Description: Set the heartbeat interval which terminal device uploading to server.

Parameter: icdev: Input, the linked handle of reader / writer

cnt_100ms: Input, interval value , unit 100 ms.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_heartBeatInterval (hdev, 50);//time is 50*100(ms) = 5000ms

**int lc_getNet_heartBeatInterval(int icdev, int* rCnt_100ms);**

Description: Get the heartbeat interval time.

Parameter: icdev: Input, linked handle of reader / writer

rCnt_100ms: Output, interval value got, unit 100 ms.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

int heartbeat;

st = lc_getNet_heartBeatInterval (hdev, &heartbeat);

**int lc_setNet_paraTitle_Data(int icdev, char* pszTitle_in);**

Description: Set the data parameter’s title , default value “Data” if not set.

Parameter: icdev: Input, the linked handle of reader / writer

pszTitle_in: Input, title for setting, maximum string length is 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_paraTitle_Data (hdev, "Data");

**int lc_getNet_paraTitle_Data(int icdev, char* pszTitle_out);**

Description: Get data parameter' s title.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output,title string, maximum size 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szTitle[9];

st = lc_getNet_paraTitle_Data (hdev, szTitle);

**int lc_setNet_paraTitle_Addr(int icdev, char* pszTitle_in);**

Description: Set the addr parameter' s title , default value "Addr" if not set.

Parameter: icdev: Input, the linked handle of reader / writer

pszTitle_in: Input, title for setting, maximum string length is 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_paraTitle_Addr (hdev, "Addr");

**int lc_getNet_paraTitle_Addr(int icdev, char* pszTitle_out);**

Description: Get addr parameter' s title.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output,title string, maximum size 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szTitle[9];

st = lc_getNet_paraTitle_Addr (hdev, szTitle);

**int lc_setNet_paraTitle_Time(int icdev, char* pszTitle_in);**

Description: Set the time parameter’s title , default value “Time” if not set.

Parameter: icdev: Input, the linked handle of reader / writer

pszTitle_in: Input, title for setting, maximum string length is 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_paraTitle_Time (hdev, “Time”);

**int lc_getNet_paraTitle_Time(int icdev, char* pszTitle_out);**

Description: Get time parameter’s title.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output,title string, maximum size 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szTitle[9];

st = lc_getNet_paraTitle_Time (hdev, szTitle);

**int lc_setNet_paraTitle_DataType(int icdev, char* pszTitle_in);**

Description: Set the data type parameter’s title , default value “DType” if not set.

Parameter: icdev: Input, the linked handle of reader / writer

pszTitle_in: Input, title for setting, maximum string length is 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_paraTitle_DataType (hdev, “DType”);

**int lc_getNet_paraTitle_DataType(int icdev, char* pszTitle_out);**

Description: Get data type parameter’s title.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output,title string, maximum size 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szTitle[9];

st = lc_getNet_paraTitle_DataType (hdev, szTitle);

**int lc_setNet_paraTitle_PackType(int icdev, char* pszTitle_in);**

Description: Set the pack type parameter’s title , default value “PType” if not set.

Parameter: icdev: Input, the linked handle of reader / writer

pszTitle_in: Input, title for setting, maximum string length is 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_paraTitle_PackType (hdev, "PType");

**int lc_getNet_paraTitle_PackType(int icdev, char* pszTitle_out);**

Description: Get pack type parameter' s title.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output,title string, maximum size 8.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szTitle[9];

st = lc_getNet_paraTitle_PackType (hdev, szTitle);

**int lc_setNet_MQTT_clientID(int icdev, char* pszMQTT_client_in);**

Description: Set the client ID which used when MQTT transport

Parameter: icdev: Input, the linked handle of reader / writer

pszMQTT_client_in: Input, client ID, maximum string length is 64.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_MQTT_clientID (hdev, "client1");

**int lc_getNet_MQTT_clientID(int icdev, char* pszMQTT_client_out);**

Description: Get client ID which for MQTT transport.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output, client ID string, maximum size 64.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szClientID[65];

st = lc_getNet_MQTT_clientID (hdev, szClientID);

**int lc_setNet_MQTT_userName(int icdev, char* pszMQTT_userName_in);**

Description: Set the user name which used when MQTT transport

Parameter: icdev: Input, the linked handle of reader / writer

pszMQTT_userName_in: Input, user name, maximum string length is 64.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_MQTT_userName (hdev, "user1");

**int lc_getNet_MQTT_userName(int icdev, char* pszMQTT_userName_out);**

Description: Get user name which for MQTT transport.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output, user name string, maximum size 64.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szUser[65];

st = lc_getNet_MQTT_userName (hdev, szUser);

**int lc_setNet_MQTT_userPass(int icdev, char* pszMQTT_userPass_in);**

Description: Set the user password which used when MQTT transport

Parameter: icdev: Input, the linked handle of reader / writer

pszMQTT_userPass_in: Input, user password, maximum string length is 64.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_MQTT_userPass (hdev, “123456”);

**int lc_getNet_MQTT_userPass(int icdev, char* pszMQTT_userPass_out);**

Description: Get user password which for MQTT transport.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output, user password string, maximum size 64.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szUserPass[65];

st = lc_getNet_MQTT_userPass (hdev, szUserPass);

**int lc_setNet_MQTT_publish_topic(int icdev, char* pszMQTT_publishTopic_in);**

Description: Set the publish topic which used when MQTT transport

Parameter: icdev: Input, the linked handle of reader / writer

pszMQTT_publishTopic_in: Input, publish topic, maximum string length is 128.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setNet_MQTT_publish_topic (hdev, “publish topic”);

**int lc_getNet_MQTT_publish_topic(int icdev, char* pszMQTT_publishTopic_out);**

Description: Get publish topic which for MQTT transport.

Parameter: icdev: Input, linked handle of reader / writer

pszTitle_out: Output, publish topic string, maximum size 128.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szTopic[129];

st = lc_getNet_MQTT_publish_topic (hdev, szTopic);

**int lc_setContent_type(int icdev, unsigned char fContentType);**

Description: Set the content type when uploading to server.

Parameter: icdev: Input, the linked handle of reader / writer

fContentType: Input, content type, which defined below:

0, raw data, field not wrap

1, json format

2, x-www-form-urlencoded format

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setContent_type (hdev, 1);

**int lc_getContent_type(int icdev, unsigned char* fContentType_out);**

Description: Get publish topic which for MQTT transport.

Parameter: icdev: Input, linked handle of reader / writer

fContentType_out: Output, see lc_setContent_type() function.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szTopic[129];

st = lc_getNet_MQTT_publish_topic (hdev, szTopic);

# WIFI Functional API Details

**int lc_setWiFi_STA_Name(int icdev,unsigned char* pSTA_Name);**

Description: Set the station name which for WIFI terminal link.

Parameter: icdev: Input, connection handle of reader / writer

pSTA_Name: Input, string maximum 64 bytes.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setWiFi_STA_Name (hdev,”wifi123”);

**int lc_getWiFi_STA_Name(int icdev, unsigned char* pSTAName_out);**

Description: Get the currently set station name for WIFI terminal link.

Parameter: icdev: Input, connection handle of reader / writer

pSTAName_out: Input, the station name, maximum string 64 bytes.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char szName[65];

st = lc_getWiFi_STA_Name (hdev, szName);

**int lc_setWiFi_STA_Password(int icdev,unsigned char* pSTA_Password);**

Description: Set the station password which for WIFI terminal link.

Parameter: icdev: Input, connection handle of reader / writer

pSTA_Password: Input, string maximum 64 bytes.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setWiFi_STA_Password (hdev,"123456");

**int lc_getWiFi_STA_Password(int icdev, unsigned char* pSTAPass_out);**

Description: Get the currently set station password for WIFI terminal link.

Parameter: icdev: Input, connection handle of reader / writer

pSTAPass_out: Input, the station password, maximum string 64 bytes.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char szPass[65];

st = lc_getWiFi_STA_Password (hdev, szPass);

# Ethernet Functional API Details

**int lc_setETH_localIP(int icdev,unsigned char* pIP_in);**

Description: Set local IP address.

Parameter: icdev: Input, connection handle of reader / writer

pIP_in: Input, device’ s local IP

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char bufIP[4]={192,168,2,88};

st = lc_setETH_localIP (hdev, bufIP);

**int lc_getETH_localIP(int icdev, unsigned char* pIP_out);**

Description: Get the local IP address of device.

Parameter: icdev: Input, connection handle of reader / writer

pIP_out: Output, get the device's local IP

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char bufIP[4];

st = lc_getETH_localIP (hdev, bufIP);

**int lc_setETH_localPort(int icdev,int port);**

Description: Set local listening port.

Parameter: icdev: Input, connection handle of reader / writer

port: Input, device’ s local listen port.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setETH_localPort (hdev, 51006);

**int lc_getETH_localPort(int icdev, int* pPort_out);**

Description: Get the listening port of device.

Parameter: icdev: Input, connection handle of reader / writer

pPort_out: Output, local listen port.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

int port;

st = lc_getETH_localPort (hdev, &port);

**int lc_setETH_gatewayIP(int icdev,unsigned char* pIP_in);**

Description: Set gateway IP address.

Parameter: icdev: Input, connection handle of reader / writer

pIP_in: Input, gateway’ s IP address

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char bufIP[4]={192,168,2,1};

st = lc_setETH_gatewayIP (hdev, bufIP);

**int lc_getETH_gatewayIP(int icdev, unsigned char* pIP_out);**

Description: Get the IP address of gateway.

Parameter: icdev: Input, connection handle of reader / writer

pIP_out: Output, gateway's IP

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char bufIP[4];

st = lc_getETH_gatewayIP (hdev, bufIP);

# Bluetooth Functional API Details

**int lc_setBTName(int icdev, char* szName);**

Description: Set the Bluetooth device’s name.

Parameter: icdev: Input, connection handle of reader / writer

szName: Input, device’s Bluetooth name, maximum string length 22 bytes.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

st = lc_setBTName (hdev, “LCBT”);

**int lc_getBTName(int icdev, char* pszName);**

Description: Get the Bluetooth device’s name.

Parameter: icdev: Input, connection handle of reader / writer

pszName: Output, device’s Bluetooth name, maximum size 22 bytes.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

char szName[23];

st = lc_getBTName (hdev, szName);

**int lc_getBTMAC(int icdev, unsigned char* pMAC);**

Description: Get the Bluetooth device’s MAC.

Parameter: icdev: Input, connection handle of reader / writer

pMAC: Output, device’s Bluetooth MAC, 4 bytes.

Return: < > 0 error code

=0 succeeded

Example: int st;

extern int hdev;

unsigned char bufMAC[6];

st = lc_getBTMAC (hdev, bufMAC);

# Appendix

## Table 1 (Mifare Plus card personalization structure)

| Key or data | Address | Describe |
|---|---|---|
| Mifare data block or value block | 0000h ~ 007Fh | 0 ~ 31 sectors |
| Mifare data block or value block | 0080h ~ 00FFh | 32 ~ 39 sectors |
| MFP Configuration Block | B000h | Define the number of non-carry MAC commands |
| Installation Identification | B001h | Apply to virtual card section |
| ATS Information | B002h | Select answer information,”Answer to select” |
| Domain Configuration Block | B003h | Configure whether proximity checking is enforced and whether RandomID is enabled |
| AES Sector Key | 4000h~403Fh | AES sector keys 0~31 sectors, the second byte defines the sector number and KeyA or KeyB.<br>KeyA = 2* Sector<br>KeyB = 2* Sector + 1<br>For example: 2 Sector KeyA, |

| | | |
|---|---|---|
| | | the corresponding<br>Address: 4004 |
| AES Sector Key | 4040h ~ 404Fh | AES sector keys 0~31 sectors, the second byte defines the sector number and KeyA or KeyB.<br>KeyA = 2* Sector<br>KeyB = 2* Sector + 1 |
| Unique Key | 8000 | Determined by vendor, cannot be changed |
| Master Key | 9000 | Can be used to change hierarchical conversion keys, card configuration keys, MFP value blocks, Install identity, ATS, and itself |
| Card Configuration Key | 9001 | Used to change domain configuration blocks, virtual cards, key and itself |
| Level 2 Conversion Key | 9002 | Upgrade to Level 2 |
| Level 3 Conversion Key | 9003 | Upgrade to Level 3 |
| Level 1 Card Verification Key | 9004 | Level 1 additional AES authentication key |
| Virtual Card Selection Key | A000 | Compute MAC key under virtual card |
| Verify Key Nearby | A001 | Calculate MAC with proximity validation |
| Virtual Card Polling ENC Key | A080 | Select virtual card to poll ENC |
| Virtual Card Polling MAC Key | A081 | Select virtual card to poll MAC |

# Table 2 API return status code

Note: This table only lists status codes with values greater than 255 and less than 255. See

Agreement Overview in the agreement document

| Status code value | | Describe |
|---|---|---|
| Hexadecimal | Decimal | |
| 0120 | 288 | Invalid device handle |
| 013C | 316 | Invalid operation |
| 0140 | 320 | Socket start failed |
| 0141 | 321 | Socket listening failed |
| 0142 | 322 | Socket binding failed |
| 0172 | 370 | This feature is not supported |
| 0173 | 371 | Overtime |
| 0174 | 372 | Overflow |
| 017C | 380 | Instruction error |
| 017D | 381 | Invalid parameter |
| 017F | 383 | Check error |
| 018D | 397 | System busy |
| 01A1 | 417 | Communication transmission error |
| 0201 | 513 | No Tag Nearly |
| 0203 | 515 | None Data |
| 0204 | 516 | Authenticate Failed |
| 0205 | 517 | Invalid Data Given |

## Table 3 Slot Of Contact Card

| SLOT | VALUE | NAME |
|---|---|---|
| SLOT_ICC_MSR | 00 | Main Slot |
| SLOT_ICC_SAM1 | 01 | SAM1 |
| SLOT_ICC_SAM2 | 02 | SAM2 |
| SLOT_ICC_SAM3 | 03 | SAM3 |

## Table 4 Contact Card Type List

| CARD TYPE | VALUE | Remark |
|---|---|---|
| 24CXX | 01 | 24C02/24C04/24C16 |
| 24C64 | 02 | |
| 4442 | 03 | |
| 4428 | 04 | |

| | | |
|---|---|---|
| CPU卡 | 0C | IS07816 Protocol |

## Table 5 UHF Tag Bank

| BANK | VALUE | Remark |
|---|---|---|
| Resvered | 00 | |
| EPC | 01 | |
| TID | 02 | |
| User | 03 | |
| Access Password | 04 | |
| Kill Password | 05 | |