Table of Contents

1. Overview

This document is used to explain the use of functions related to UHF DLL library file (uhfapi. DLL). Through this DLL library, users can easily write their own windows applications, connect the applicable functional devices through USB, TCP or serial ports, support one to many links, and view the working status of the currently connected devices through the connection status query interface.

2. Operating Environment

Applicable operating system:
Windows XP/7/8/10
It depends on DLL files "hidapi. DLL" and uhfapi.dll. If you need to call the interface in a static environment, you also need to:
Add the file "uhfapi. Lib", uhfapi h ”, HFReader.h

3. General

- **Log**
  If printing debugging is required
  Log can call setloglevel to enable debugging information output at different levels.
  If you need to save the detailed communication command between PC and device, you can call savelogfile to save the transfer command file. The file is saved in the same level directory of DLL and named with date and time.
- **Notes**
  During the continuous inventory tag, the reader does not respond to other operations. If other commands need to be executed, call stop counting first, stop the continuous inventory tag, and then call other operations.
- **Return Value**
  Return value is less than 0, means err code.

4. Interface

4.1 Communication Interface

- TCPConnect
  Interface Description: use TCP to connect to the specified address port.
  int TCPConnect(const char * hostaddr,int hostport);
  Parameters:
  Hostaddr [in]: IP address
  Hostport [in]: port number, 0 ~ 65535
  Return: 0 - success, other failures refer to the error code definition
  TCPDisconnect
  Interface Description: disconnect TCP connection
  void TCPDisconnect();
  Parameters: None

- UsbOpen
  Interface Description: use USB to open the device
  int UsbOpen();
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- UsbClose
  Interface Description: turn off USB device
  void UsbClose();
  Parameters: None
  Return: None
- ComOpen
  Interface Description: use 115200 default baud rate to open the serial port device.
  int ComOpen(int port);
  Parameters:
  Port [in]: serial port number loaded by the serial port device, 0 ~ n, for example: COM8 serial port number is 8.
  Return: 0 - success, other failures refer to the error code definition.
- ComOpenWithBand
  Interface Description: use the custom baud rate to open the serial port device.
  int ComOpenWithBaud(int port, int baudrate);
  Parameters:
  Port [in]: serial port number loaded by the serial port device, 0 ~ n, for example: COM8 serial port number is 8.
  Baudrate [in]: baud rate, such as 9600, 19200, 38400, 57600, 115200, 230400…..
  Return: 0 - success, other failures refer to the error code definition.
- ClosePort
  Interface Description: close the serial port
  int ClosePort();
  Parameters: None
  Return: None
- LinkGetInfo
  Interface Description: get the status information of all current connections.
  int LinkGetInfo(char *info, int len);
  Info [out]: receive connection status information in JSON format.
  Len [in]: limit the maximum receiving length of info.
  Return: the effective length of info should be greater than 0. If it is less than or equal to 0, it is a failure.
- LinkGetSelected
  Interface Description: get the ID of the current active link.
  int LinkGetSelected();
  Parameters: None
  Return: ID of the current activity, greater than and including 0. Refer to the error code definition for other failures.
- LinkSelect
  Interface Description: select the link with the specified ID as the current active link.
  int LinkSelect(int id);
  Parameters:
  ID [in]: obtained using linkgetinfo interface, 0 ~ 635535.
  Return: 0 - success, other failures refer to the error code definition.
- LinkCloseAll
  Interface Description: close all open connections.
  void LinkCloseAll();

Parameters: None
Return: None

- SendBytes
  Interface Description: send data to the currently connected device, and the data packet is encapsulated by the user.
  int SendBytes(const unsigned char *sendBytes, int sendLen);
  Parameters:
  Sendbytes [in]: the pointer address of the sent data.
  Sendlen [in]: length of sent data.
  Return: the length of successfully sent data should be greater than 0, and less than or equal to 0 indicates that sending data failed.
- RecvBytes
  Interface Description: receive the data of the current linked device. The data format is not processed, and the user can analyze it by himself.
  int RecvBytes(unsigned char *recvBytes, int recvLen);
  Parameters:
  Recvbytes [out]: pointer address of received data.
  Recvlen [in]: limit the maximum receive length of recvbytes.
  Return: the length of successfully sent data should be greater than 0, and less than or equal to 0 indicates that sending data failed.

4.2 UHF Interface

- UHFGetReaderVersion
  Interface Description: obtain the hardware version number.
  int UHFGetReaderVersion(unsigned char *version);
  Parameters:
  Version [out]: length (1 byte) + version number.
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetSoftwareVersion
  Interface Description: obtain the version number of UHF module software.
  int UHFGetSoftwareVersion(unsigned char *version);
  Parameters:
  Version [out]: length (1 byte) + version number.
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetDeviceID
  Interface Description: obtain the version number of UHF module software.
  int UHFGetDeviceID(unsigned int *id);
  Parameters:
  ID [out]: ID number of the module, 32bits
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetIp
  Interface Description: set the local IP and port.
  int UHFSetIp (unsigned char* ip, unsigned char *port, unsigned char *mask, unsigned char *gate);
  Parameters:
  IP [in]: IP address, string, such as "192.168.1.101"
  Port [in]: port number, expressed in string, such as "5084"
  Mask [in]: mask, 4bytes
  Gate [in]: gateway, 4bytes
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetIp

Interface Description: read the network information of UHF host.
int UHFGetIp (unsigned char* ipbuf, unsigned char *postbuf, unsigned char *mask, unsigned char *gate);
Parameters:
IP [out]: IP address, string, e.g. "192.168.1.101"
Port [out]: port number, expressed in string, such as "5084"
Mask [out]: mask, 4bytes
Gate [out]: gateway, 4bytes
Return: 0 - success, other failures refer to the error code definition.

- UHFSetDestIp
  Interface Description: set the target address of UHF equipment, which is used to actively forward the equipment inventory data to the target address using UDP protocol.
  int UHFSetDestIp (unsigned char* ip, unsigned char *port);
  Parameters:
  IP [in]: IP address, string, such as "192.168.1.101"
  Port [in]: port number, expressed in string, such as "5084"
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetDestIp
  Interface Description: obtain the target address actively output by UHF equipment.
  int UHFGetDestIp (unsigned char* ip, unsigned char *port);
  Parameters:
  IP [out]: IP address, string, e.g. "192.168.1.101"
  Port [out]: port number, expressed in string, such as "5084"
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetPower
  Interface Description: set the transmission power of UHF antenna.
  int UHFSetPower (unsigned char save,unsigned char uPower);
  Parameters:
  Save [in]: 1 - save after power failure, 0 - do not save after power failure
  UPower [in]: power (DB)
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetAntennaPower
  Interface Description: set the transmit and receive power of UHF designated antenna.
  int UHFSetAntennaPower (unsigned char save, unsigned char num, unsigned char read_power, unsigned char write_power);
  Parameters:
  Save [in]: 1 - save after power failure, 0 - do not save after power failure.
  Num [in]: antenna number, value range 1 ~ n, n is not greater than the total number of antennas of the current equipment.
  read_ Power [in]: received power of antenna (DB).
  write_ Power [in]: transmission power of antenna (DB).
  Return: 0 - success, other failures refer to the error code definition.

- UHFGetPower
  Interface Description: obtain UHF antenna power.
  int UHFGetPower (unsigned char *uPower);
  Parameters:
  UPower [out]: antenna power (DB), 8bits
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetAntennaPower
  Interface Description: obtain the power of all UHF antennas.
  int UHFGetAntennaPower (unsigned char *pPowerInfo, unsigned short *nBytesReturned);
  Parameters:
  Powerinfo [in]: power configuration information of each antenna, data format {[antenna number (1byte) + read power (1byte) + write power (1byte)],...}, length is 3 * n, n is the number of antennas.
  Nbytesreturned [in]: length of power information, unit: byte
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetJumpFrequency
  Interface Description: Setup fixed frequency mode.
  int UHFSetJumpFrequency(unsigned char nums,unsigned int *frequencys);
  Parameters:
  Nums [in]: number of fixed frequencies.
  Frequencies [in]: frequency point array, unit: kHz, for example: {920125921250...}
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetJumpFrequency
  Interface Description: obtain current fixed frequency status and fixed frequency table.
  int UHFGetJumpFrequency(unsigned int *frequencys);
  Parameters:
  Freqbuf [out]: frequencies [0] - number of fixed frequencies, frequencies [1 ~ n] frequency point array, for example: {2920125921250}
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetGen2
  Interface Description: set Gen2 parameters
  int UHFSetGen2 (unsigned char Target,unsigned char Action, unsigned char T,unsigned char Q, unsigned char StartQ,unsigned char MinQ, unsigned char MaxQ,unsigned char D,unsigned char C,unsigned char P, unsigned char Sel,unsigned char Session,unsigned char G,unsigned char LF);
  Parameters:
  Target [in]: target parameter of select command, 0-s0, 1-s1, 2-s2, 3-s3, 4-sl
  Action [in]: action parameter of select command.

	Matching	Non-Matching
0	assert SL or inventoried →A	de-assert SL or inventoried → B
1	assert SL or inventoried → A	do nothing
2	do nothing	de-assert SL or inventoried → B
3	negate SL or (A → B, B → A)	do nothing
4	de-assert SL or inventoried → B	de-assert SL or inventoried → A
5	de-assert SL or inventoried → B	do nothing
6	do nothing	de-assert SL or inventoried → A
7	do nothing	negate SL or (A → B, B → A)

T [in]: truncate parameter of select command, 0 - Disable truncation, 1 - enable truncation
Q [in]: 0-fixed Q algorithm, 1-dynamic Q algorithm
startQ[in]：0~15
MinQ[in]：0~15
MaxQ[in]：0~15
D [in]: Dr parameter of query command, 0-84,1-64 / 3
C [in]: m parameter of query command, 0 - FM0, 1 - miller2, 2 - miller4, 3 - miller8
P [in]: trext parameter of query command, 0 - no pilot, 1 - use pilot
SEL [in]: sel parameter of query command, 0-all, 1-all, 2 - ~ SL, 3-sl
Session [in]: session parameters of the query command, 0-s0, 1-s1, 2-s2, 3-s3
G [in]: target parameter of query command, 0-A, 1-B
LF [in]: link frequency setting: 0-40khz, 1-160khz, 2-200khz, 3-250khz, 4-300khz, 5-320khz, 6-400khz, 7-640khz
Return: 0 - success, other failures refer to the error code definition.

- UHFGetGen2
  Interface Description: get the current Gen 2 parameter setting.
  int UHFGetGen2 (unsigned char *Target, unsigned char *Action,unsigned char *T,unsigned char *Q, unsigned char *StartQ,unsigned char *MinQ, unsigned char *MaxQ,unsigned char *D, unsigned char *Coding,unsigned char *P, unsigned char *Sel,unsigned char *Session,unsigned char *G,unsigned char *LF);
  Parameter:
  Target [out]: target parameter of select command, 0-s0, 1-s1, 2-s2, 3-s3, 4-sl
  Action [out]: the action parameter of the select command.

	Matching	Non-Matching
0	assert SL or inventoried →A	de-assert SL or inventoried → B
1	assert SL or inventoried → A	do nothing
2	do nothing	de-assert SL or inventoried → B
3	negate SL or (A → B, B → A)	do nothing
4	de-assert SL or inventoried → B	de-assert SL or inventoried → A
5	de-assert SL or inventoried → B	do nothing
6	do nothing	de-assert SL or inventoried → A
7	do nothing	negate SL or (A → B, B → A)

  T [out]: truncate parameter of select command, 0 - Disable truncation, 1 - enable truncation.
  Q [out]: 0-fixed Q algorithm, 1-dynamic Q algorithm
  startQ[out]：0~15
  MinQ[out]：0~15
  MaxQ[out]：0~15
  D [out]: Dr parameter of query command, 0-84,1-64 / 3
  C [out]: m parameter of query command, 0 - FM0, 1 - miller2, 2 - miller4, 3 - miller8.
  P [out]: trext parameter of query command, 0 - no pilot, 1 - use pilot.
  SEL [out]: sel parameter of query command, 0-all, 1-all, 2 - ~ SL, 3-sl.
  Session [out]: the session parameters of the query command, 0-s0, 1-s1, 2-s2, 3-s3.
  G [out]: target parameter of query command, 0-A, 1-B.
  LF [out]: link frequency setting: 0-40khz, 1-160khz, 2-200khz, 3-250khz, 4-300khz, 5-320khz, 6-400khz, 7-640khz.

Return: 0 - success, other failures refer to the error code definition.

- UHFSetCW
  Interface Description: continuous carrier setting.
  int UHFSetCW(unsigned char enable);
  Parameters:
  Enable [in]: 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetANT
  Interface Description: antenna switch setting.
  int UHFSetANT(unsigned char save,unsigned char *status);
  Parameters:
  Save [in]: 0 - do not save after power failure, 1 - save after power failure.
  Status [in]: 2bytes, 16bits, high byte in front, bit [0 ~ 15] corresponding to antenna number 1 ~ 16, 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetANT
  Interface Description: get antenna on status
  int UHFGetANT(unsigned char *status);
  Parameters:
  Status [out]: 2bytes, 16bits, high byte in front, bit [0 ~ 15] corresponding to antenna number 1 ~ 16, 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetRegion
  Interface Description: frequency band area setting.
  int UHFSetRegion(unsigned char save, unsigned char region);
  Parameters:
  Save [in]: 0 - do not save after power failure, 1 - save after power failure
  Region [in]: band region, 0x01-china1, 0x02-china2, 0x04 Europe, 0x08 USA, 0x16 Korea, 0x32 Japan
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetRegion
  Interface Description: obtain frequency band area.
  int UHFGetRegion(unsigned char *region);
  Parameters:
  Region [out]: band region, 1byte, 0x01-china1, 0x02-china2, 0x04 Europe, 0x08 USA, 0x16 Korea, 0x32 Japan
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetTemperature
  Interface Description: obtain UHF module temperature.
  int UHFGetTemperature(unsigned int *temperature);
  Parameters:
  Temperature [out]: Fahrenheit temperature, 32bits
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetTemperatureProtect
  Interface Description: set temperature protection switch.
  int UHFSetTemperatureProtect(unsigned char enable);
  Parameters:
  Enable [in]: 0 - turn off temperature protection, 1 - enable temperature protection.
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetTemperatureProtect
  Interface Description: obtain temperature protection status
  int UHFGetTemperatureProtect(unsigned char *enable);

Parameters:
Enable [out]: temperature protection status, 1byte, 0-not enabled, 1-enabled.
Return: 0 - success, other failures refer to the error code definition.

- UHFSetBeep
  Interface Description: set the buzzer working mode of UHF equipment.
  int UHFSetBeep(unsigned char mode);
  Parameters:
  Mode [in]: buzzer switch, 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetBeep
  Interface Description: obtain the working mode of buzzer of UHF equipment.
  int UHFGetBeep(unsigned char* mode);
  Parameters:
  Mode [out]: 1byte, buzzer switch, 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetTempVal
  Interface Description: set temperature protection value.
  int UHFSetTempVal (unsigned char tempVal);
  Parameters:
  Tempval [in]: temperature value, unit - Fahrenheit degree Celsius, value range: 50 ℃ ~ 75 ℃
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetTempVal
  Interface Description: obtain temperature protection value.
  int UHFGetTempVal(unsigned char* TempVal);
  Parameters:
  Tempval [out]: 1byte, temperature value, unit - Fahrenheit degree Celsius, value range: 50 ℃ ~ 75 ℃
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetIOControl
  Interface Description: relay and IO control output settings.
  int UHFSetIOControl(unsigned char output1, unsigned char output2 ,unsigned char outStatus);
  Parameters:
  Output1 [in]: GPO 0 status, 0-low level, 1 high level
  Output2 [in]: GPO 1 status, 0-low level, 1 high level
  Outstatus [in]: 0-open, 1-close
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetIOControl
  Interface Description: obtain relay and IO control output status.
  int UHFGetIOControl (unsigned char *statusData);
  Parameters:
  StatusData [out]: statusData [0], GPO 0 status, 0-low level, 1 high level, statusData [1], GPO 1 status, 0-low level, 1 high level
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetANTWorkTime
  Interface Description: set antenna working time.
  int UHFSetANTWorkTime(unsigned char antnum,unsigned char save,unsigned int workTime);
  Parameters:
  Antnum [in]: antenna number, 1 ~ n
  Save [in]: save, 0 - do not save after power failure, 1 - save after power failure
  Worktime [in]: working time, unit: ms, 10 ~ 65535
  Return: 0 - success, other failures refer to the error code definition.

- UHFGetANTWorkTime
  Interface Description: obtain antenna working time.
  int UHFGetANTWorkTime(unsigned char antnum,unsigned int *WorkTime);
  Parameters:
  Antnum [in]: antenna number, 1 ~ n
  Worktime [out]: working time, 32bits, unit: ms, 10 ~ 65535
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetWorkMode
  Interface Description: set UHF working mode.
  int UHFSetWorkMode (unsigned char mode);
  Parameters:
  Mode [in]: working mode, 0-command mode, 1-automatic working mode, 2-trigger working mode
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetWorkMode
  Interface Description: obtain UHF working mode.
  int UHFGetWorkMode(unsigned char* mode);
  Parameters:
  Mode [out]: working mode, 1byte, 0-command mode, 1-automatic working mode, 2-trigger working mode
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetWorkModePara
  Interface Description: set the triggering mode of UHF equipment.
  int UHFSetWorkModePara(unsigned char * param);
  Parameters:
  Param [in]: param [0] - IO trigger, 0x00 indicates trigger input 1, 0x01 indicates trigger input 2,
  Param [1 ~ 2]: trigger working time, high byte first, unit - 10 milliseconds
  Param [3 ~ 4]: trigger time interval, high byte in front, unit - 10 ms, minimum time interval from the last trigger
  Param [5]: tag output mode, 0x00 serial port output, 0x01 UDP output
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetWorkModePara
  Interface Description: obtain the triggering mode of UHF equipment.
  int UHFGetWorkModePara(unsigned char * para);
  Parameters:
  Param [out]: param [0] - IO trigger, 0x00 indicates trigger input 1, 0x01 indicates trigger input 2,
  Param [1 ~ 2]: trigger working time, high byte first, unit - 10 milliseconds
  Param [3 ~ 4]: trigger time interval, high byte in front, unit - 10 ms, minimum time interval from the last trigger
  Param [5]: tag output mode, 0x00 serial port output, 0x01 UDP output
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetRFLink
  Interface Description: set the recommended RF link combination.
  int UHFSetRFLink (unsigned char save,unsigned char mode);
  Parameters:
  Save [in]: save, 0 - do not save after power failure, 1: save after power failure
  mode[in]：0-DSB_ ASK/FM0/40KHZ、1-PR_ ASK/Miller4/250KHZ、2-PR_ ASK/Miller4/300KHZ、3-DSB_ ASK/FM0/400KHZ
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetRFLink

Interface Description: obtain the recommended RF link combination status.
int UHFGetRFLink (unsigned char* mode);
Parameters:
mode[out]：1Byte，0-DSB_ ASK/FM0/40KHZ、1-PR_ ASK/Miller4/250KHZ、2-PR_ ASK/Miller4/300KHZ、3-DSB_ ASK/FM0/400KHZ
Return: 0 - success, other failures refer to the error code definition.

- UHFSetFastID
  Set fastid switch.
  int UHFSetFastID(unsigned char enable);
  Parameters:
  Enable [in]: 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetFastID
  Interface Description: get fastid switch status.
  int UHFGetFastID(unsigned char *enable);
  Parameters:
  Enable [out]: 1byte, 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetTagfocus
  Interface Description: set tag focus switch.
  int UHFSetTagfocus(unsigned char enable);
  Parameters:
  Enable [in]: 0-off, 1-on
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetWorkTime
  Interface Description: set contact working time and waiting time.
  int UHFSetWorkTime(unsigned char save, unsigned char param0 ,unsigned char param1, unsigned char param2,unsigned char param3);
  Parameters:
  Save [in]: 0 - do not save after power failure, 1 - save after power failure
  Param [in]: param [0] * 256 + param [1] indicates the working time, param [2] * 256 + param [3] indicates the waiting time, in milliseconds
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetWorkTime
  Interface Description: obtain continuous card searching and waiting time.
  int UHFGetWorkTime (unsigned char *param);
  Parameters:
  Param: param [0] * 256 + param [1] indicates the working time, param [2] * 256 + param [3] indicates the waiting time, in milliseconds.
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetSoftReset
  Interface Description: reset UHF module.
  int UHFSetSoftReset(void);
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetDualSingleMode
  Interface Description: Switch Single / dual mode.
  int UHFSetDualSingelMode(unsigned char save,unsigned char mode);
  Parameters:
  Save [in]: 0 - do not save after power failure, 1 - save after power failure
  mode[in]：0-Dual、1-Single
  Return: 0 - success, other failures refer to the error code definition.

- UHFGetDualSingleMode
  Interface Description: get single / dual mode
  int UHFGetDualSingelMode(unsigned char *mode);
  Parameters:
  mode[in]：1Byte，0-Dual、1-Single
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetFilter
  Interface Description: set tag filter configuration.
  int UHFSetFilter(unsigned char save,unsigned char bank,unsigned int startAddr,unsigned int dataLen,unsigned char *filterBuf);
  Parameters:
  Save [in]: 0 - do not save after power failure, 1: - save after power failure.
  Bank [in]: data area, 1-epc, 2-tid, 3-user area
  Startaddr [in]: address start, unit - bit
  Datalen [in]: filter data length, unit - bit
  Filterbuf [in]: filter the data of the tag, and the length of byte data (number of bytes * 8) is not less than datalen.
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetEPCTIDMode
  Interface Description: set the format of data acquisition
  int UHFSetEPCTIDMode(unsigned char save,unsigned char mode);
  Parameters:
  Save [in]: 0 - do not save after power failure, 1 - save after power failure
  Mode [in]: 0 - output EPC, 1 - output EPC + TID, 2 - output EPC + TID + user
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetEPCTIDMode
  Interface Description: get tag output format.
  int UHFGetEPCTIDMode(unsigned char *mode);
  Parameters:
  Mode [out]: 1byte, 0-output EPC, 1-output EPC + TID, 2-output EPC + TID + user
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetEPCTIDUSERMode
  Interface Description: get tag output format.
  int UHFSetEPCTIDUSERMode(unsigned char save, unsigned char mode, unsigned char userAddr, unsigned char userLen);
  Parameters:
  Save [in]: 0 - do not save after power failure, 1 - save after power failure
  Mode [in]: 0 - output EPC, 1 - output EPC + TID, 2 - output EPC + TID + user
  Useraddr [in]: the starting address of the user area, unit word (16bits)
  Userlen [in]: read the length of user data, unit - word (16bits)
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetEPCTIDUSERMode
  Interface Description:
  Int UHFGetEPCTIDUSERMode(unsigned char rev 1, unsigned char rev2, unsigned char char * mode);* mode);
  Parameters:
  Rev1 [in]: reserved data, passed in: reserved data, passed in 00
  Rev2 [in]: reserved data, passed in: reserved data, passed in 00
  Mode [out]: 3bytes 3bytes, mode [mode [0], 0, 00 -- output epcepc, 11 -- output EPC + tidpc + TID, 22 -- output EPC + TID + userepc + TID + user, mode [1] mode [1], start address of useruser area, start address of unit area, unit -- word (word (1166 bits)), mode [2] mode [2], read length of useruser data, length of unit data, unit -- word (word (1166 bits))

Return: 33 -- success, other failures refer to success, and other failures refer to error code definition.

- UHFSetDefaultMode
  Interface Description: UHF module restores factory settings.
  int UHFSetDefaultMode();
  Return: 0 - success, other failures refer to the error code definition.
- UHFInventorySingle
  Interface Description: single reading tag.
  int UHFInventorySingle (unsigned char* rLen, unsigned char* rData);
  Parameters:
  Rlen [out]: 1byte, length of tag data
  Rdata [out]: tag data, length rlen [0]
  Return: 0 - success, other failures refer to the error code definition.
- UHFInventory
  Interface Description: Start inventory.
  int UHFInventory();
  Parameters: None
  return:
  0 succeeded, other failures refer to the error code definition.
- UHFInventoryById
  Interface Description: open the equipment inventory with the specified link ID.
  int UHFInventoryById(int id);
  Parameter: ID [in]: link ID, 0 ~ 65535
  Return: 0 - success, other failures refer to the error code definition.
- UHFInventoryTempTag
  Interface Description: calibration data + sensor code + on chip RSSI + temperature code.
  int UHFInventoryTempTag (unsigned char antNum,unsigned char *powerValue);
  Parameters:
  Antnum [in]: antenna number
  Powervalue [in]: power value, 2bytes, high byte first, unit: dBm
  Return: 0 - success, other failures refer to the error code definition.
- UHFInventoryTempTag2
  Interface Description: inventory temperature tag (on chip RSSI + temperature code)
  int UHFInventoryTempTag2 (unsigned char antNum,unsigned char *powerValue);
  Parameters:
  Antnum [in]: antenna number
  Powervalue [in]: power value, 2bytes, high byte first, unit: dBm
  Return: 0 - success, other failures refer to the error code definition.
- UHFStopGet
  Interface Description: stop counting tag.
  int UHFStopGet();
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- UHFStopById
  Interface Description: stop counting equipment with specified link ID
  int UHFStopById(int id);
  Parameter: ID [in]: link ID, 0 ~ 65535
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetTagData
  Interface Description: obtain the data of continuous counting tags.
  int UHFGetTagData(unsigned char *tdata, int recvlen);
  Parameters:

Tdata [out]: tag data, length is positive return value
Recvlen [in]: maximum data length that can be received
Return: greater than 0 is the length of Tdata, 0 means no data, and other failures refer to the error code definition.

- UHF_GetReceived_EX
  Interface Description: obtain the data of continuous counting tags.
  int UHF_ GetReceived_ EX(int* rLen, unsigned char* rData);
  Rlen: tag data length
  Rdata: tag data, length rlen [0]
  Return: 0 - success, other failures refer to the error code definition.
- UHF_GetTempTagReceived
  Interface Description: obtain the data of inventory temperature tag.
  int UHF_ GetTempTagReceived(int* tLen, unsigned char* tData);
  Parameters:
  Tlen [out]: length of tag data obtained
  Tdata [out]: acquired tag data (calibration data + sensor code + on chip RSSI + temp code)
  Return: 0 - success, other failures refer to the error code definition.
- UHF_GetTempTagReceived2
  Interface Description: obtain the data of inventory temperature tag (on chip RSSI + temperature code).
  int UHF_ GetTempTagReceived2(int* tLen, unsigned char* tData);
  Parameters:
  Tlen [out]: length of tag data obtained
  Tdata [out]: acquired tag data
  Return: 0 - success, other failures refer to the error code definition.
- UHFAuthenticate
  Interface Description: verification tag.
  int UHFAuthenticate (unsigned long password, unsigned char filterBank, unsigned short filterAddr, unsigned char * filterData, unsigned short filterLen, unsigned char keyID, unsigned short tLen, unsigned char *tData, unsigned short *recvLen, unsigned char *recvData);
  Parameters:
  Password [in]: 32bits authentication password
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  KeyID: the keyID used by the authenticate command. The default value is 0x00
  tLen：IChallenge_ The data length of tam1 is 10 by default
  tData：IChallenge_ Data of tam1
  Recvlen: output data length, unit - byte
  Recvdata: output data with length of recvlen
  Return: 0 - success, other failures refer to the error code definition.
- UHFAuthenticateCommon
  Interface Description: verification tag, for Impinj m775
  int UHFAuthenticateCommon (unsigned long password, unsigned char filterBank, unsigned short filterAddr, unsigned char * filterData, unsigned short filterLen, unsigned char keyID, unsigned short tLen,
  unsigned char *tData, unsigned short *recvLen, unsigned char *recvData);
  Parameters:
  Password [in]: 32bits authentication password
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit

Filterlen [in]: length of filtered data, unit - bit
Filterdata [in]: filtered data
KeyID: the keyID used by the authenticate command. The default value is 0x00
tLen：IChallenge_ The data length of tam1 is 10 by default
tData：IChallenge_ Data of tam1
Recvlen: output data length, unit - byte
Recvdata: output data with length of recvlen, challenge (6bytes) + tag short TID (8bytes) + tag response (8bytes)
Return: 0 - success, other failures refer to the error code definition.

- UHFReadData
  Interface Description: read tag data area.
  UHFReadData (unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char bank,unsigned int addr, unsigned int readLen, unsigned char* readData, unsigned int* nByesReturned);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Bank [in]: read data area, 1-epc, 2-tid, 3-user area
  Addr [in]: start address of reading, unit word (16bits)
  Readlen [in]: read data length
  ReadData [out]: returned data
  Nbyesreturned [out]: length of returned data, unit - bytes
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetCalibrationData
  Interface Description: read calibration data.
  int UHFGetCalibrationData (unsigned char *epc,unsigned char antNum,unsigned char *powerValue,unsigned char * c_data);
  Parameters:
  EPC: EPC number, 16bytes
  Antnum: antenna number
  Powervalue: power value, 2bytes, high byte first, unit - dBm
  c_ data：Calibration Data，8Bytes
  Return: 8 (bytes) - success, other failures refer to the error code definition.
- UHFGetCalibrationDataEX
  Interface Description: read calibration data, which is different from the above method and can be passed in function words
  int UHFGetCalibrationDataEX (unsigned char mode, unsigned char *epc,unsigned char antNum,unsigned char *powerValue,unsigned char *data);
  Parameters:
  Mode [in]: function word (determined according to the tag)
  EPC [in]: EPC number, 16bytes
  Antnum [in]: antenna number
  Powervalue [in]: power value, 2bytes, high byte first, unit - dBm
  c_ data[out]：Calibration Data，8Bytes
  Return: 8 (bytes) - success, other failures refer to the error code definition.
- UHFGetSensorCode
  Interface Description: read sensor code.
  int UHFGetSensorCode (unsigned char *epc,unsigned char antNum,unsigned char *powerValue,unsigned char * sensorCode);

Parameters:
EPC [in]: EPC number, 16bytes
Antnum [in]: antenna number
Powervalue [in]: power value, 2bytes, high byte first, unit - dBm
sensorCode[out]：sensor code，2Bytes
Return: sensor code length should be 2. Refer to the error code definition for other failures.

- UHFGetOnChipRSSI
  Interface Description: read on chip RSSI.
  int UHFGetOnChipRSSI (unsigned char *epc,unsigned char antNum,unsigned char *powerValue,unsigned char *rssi);
  Parameters:
  EPC [in]: EPC number, 16bytes
  Antnum [in]: antenna number
  Powervalue [in]: power value, 2bytes, high byte first, unit - dBm
  rssi[out]：On-Chip RSSI，2Bytes
  Return: on chip RSSI length, which should be 2. Refer to the error code definition for other failures.
- UHFGetTemperatureCode
  Interface Description: read temperature code
  int UHFGetTempertureCode (unsigned char *epc,unsigned char antNum,unsigned char *powerValue,unsigned char *code);
  Parameters:
  EPC [in]: EPC number, 16bytes
  Antnum [in]: antenna number
  Powervalue [in]: power value, 2bytes, high byte first, unit - dBm
  code [out]：Temperature Code，2Bytes
  Return: the length of the temperature code should be 2. Refer to the error code definition for other failures.
- UHFGetOnChipRSSIAndTempCode
  Interface Description: read on chip RSSI and temperature code
  int UHFGetOnChipRSSIAndTempCode (unsigned char *epc,unsigned char antNum,unsigned char *powerValue,unsigned char *rssiCode);
  Parameters:
  EPC [in]: EPC number, 16bytes
  Antnum [in]: antenna number
  Powervalue [in]: power value, 2bytes, high byte first, unit - dBm
  rssiCode[out]：On-Chip RSSI + Temperature Code，2Bytes
  Return: RSSI + code length, which should be 4. Refer to the error code definition for other failures.
- UHFWriteData
  Interface Description: write data to the specified area of the tag
  int UHFWriteData (unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char bank,unsigned int addr, unsigned char writeLen,unsigned char *writeData);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Bank [in]: data area written, 1-epc, 2-tid, 3-user area
  Addr [in]: start address of writing, unit word (16bits)

Writelen [in]: length of data written
Writedata [in]: data written
Return: 0 - success, other failures refer to the error code definition.

- UHFWriteCalibrationData
  Interface Description: write calibration data
  int UHFWriteCalibrationData (unsigned char *epc,unsigned char antNum,unsigned char *powerValue,unsigned char *c_data);
  Parameters:
  EPC [in]: EPC number, 16bytes
  Antnum [in]: antenna number
  Powervalue [in]: power value, 2bytes, high byte first, unit - dBm
  c_ data[in]：Calibration Data，8Bytes
  Return: 0 - success, other failures refer to the error code definition.
- UHFLockTag
  Interface Description: Lock tag
  bool UHFLockTag(unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char *lockbuf);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Lockconfig [in]: 3bytes, the high byte is the first, 0-9 bits are action bits, and 10-19 bits are mask bits
  Return: true - success, other failures refer to the error code definition.
- UHFGBTagLock
  Interface Description: Lock national military standard tag
  int UHFGBTagLock(unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char memory, unsigned char config, unsigned char action);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Memory [in]: 0x00 tag information area, 0x10 coding area, 0x20 security area
  0x3x user area: 0x30 ~ 0x3f user area No.: 0 ~ 15
  Config [in]: 0 - indicates the configuration store attribute, 1 - indicates the configuration security mode
  Action [in]: used to configure storage area properties or security mode, as follows:
  Configure storage area attributes: 0x00 readable and writable, 0x01 readable and writable, 0x02 unreadable and writable, 0x03 unreadable and writable
  Configure security mode: 0x00 reserved, 0x01 no authentication required, 0x02 authentication required, no secure communication required, 0x03 authentication required, secure communication required
  Return: true - success, other failures refer to the error code definition.
- UHFKillTag
  Interface Description: kill tag
  int UHFKillTag(unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData);

Parameters:
Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
Filteraddr [in]: data length of filter tag, unit - bit
Filterlen [in]: length of filtered data, unit - bit
Filterdata [in]: filtered data
Return: 0 - success, other failures refer to the error code definition.

- UHFBlockWriteData
  Interface Description: write tags by block.
  int UHFBlockWriteData (unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterStartaddr, unsigned int filterLen, unsigned char *filterData, unsigned char bank,unsigned int addr, unsigned int writeLen,unsigned char * writeData);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Bank [in]: data area written, 1-epc, 2-tid, 3-user area
  Addr [in]: start address of writing, unit word (16bits)
  Writelen [in]: length of data written
  Writedata [in]: data written
  Return: 0 - success, other failures refer to the error code definition.
- UHFSetQT
  Interface Description: set QT command parameters.
  int UHFSetQT (unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char QTData);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Qtdata [in]: bit0 indicates close control, 0-off, 1-on, Bit1 = 0 enables private memory map, Bit1 = 1 enables public memory map, and other bits are reserved
  Return: 0 - success, other failures refer to the error code definition.
- UHFGetQT
  Interface Description: get QT parameters.
  int UHFGetQT (unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char *QTData);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Qtdata [out]: 1byte, bit0 indicates close control, 0-off, 1-on, Bit1 = 0 enables private memory map, Bit1 = 1 enables public memory map, and other bits are reserved
  Return: 0 - success, other failures refer to the error code definition.
- UHFReadQT
  Interface Description: QT tag reading operation.

int UHFReadQT(unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char QTData,unsigned char bank,unsigned int addr, unsigned char readLen, unsigned char *readData,unsigned char *nBytesReturned);
Parameters:
Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
Filteraddr [in]: data length of filter tag, unit - bit
Filterlen [in]: length of filtered data, unit - bit
Filterdata [in]: filtered data
Qtdata [in]: bit0,0-no proximity control, 1-enable proximity control, other bits reserved
Bank [in]: read data area, 1-epc, 2-tid, 3-user area
Addr [in]: start address of reading, unit word (16bits)
Readlen [in]: read data length
ReadData [out]: returned data
Nbyesreturned [out]: length of returned data, unit - bytes
Return: 0 - success, other failures refer to the error code definition.

- UHFWriteQT
  Interface Description: QT tag write operation.
  int UHFWriteQT(unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char QTData,unsigned char bank,unsigned int addr, unsigned char writeLen,unsigned char *writeData);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Qtdata [in]: bit0,0-no proximity control, 1-enable proximity control, other bits reserved
  Bank [in]: data area written, 1-epc, 2-tid, 3-user area
  Addr [in]: start address of writing, unit word (16bits)
  Writelen [in]: length of data written
  Writedata [in]: data written
  Return: 0 - success, other failures refer to the error code definition.
- UHFBlockPermalock
  Interface Description: permanent lock tag.
  int UHFBlockPermalock(unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr, unsigned int filterLen, unsigned char *filterData, unsigned char readLock,unsigned char bank,unsigned int addr, unsigned char lockRange,unsigned char *uMaskData);
  Parameters:
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Readlock [in]: bit0 = 1 indicates read, 1 indicates permanent lock, and other bits are reserved
  Bank [in]: locked data area, 1-epc, 2-tid, 3-user area
  Addr [in]: the starting address of the lock, in 16 blocks
  Lockrange [in]: locked block range, unit: 16 blocks
  Umaskdata [in]: block mask data, the high order is the first, two bytes, and 16 bits correspond to 16 blocks. Whether to select
  Return: 0 - success, other failures refer to the error code definition.

- UHFDeactivate
  Interface Description: active or inactive em4124 tag.
  int UHFDeactivate (unsigned int cmd,unsigned char* uAccessPwd, unsigned char filterBank,unsigned int filterAddr,unsigned int filterLen, unsigned char *filterData);
  Parameters:
  cmd[in]:？？
  Uaccesspwd [in]: the access password of the tag: 4bytes, with the high byte in front
  Filterbank [in]: filtered data area, 1-epc, 2-tid, 3-user area
  Filteraddr [in]: data length of filter tag, unit - bit
  Filterlen [in]: length of filtered data, unit - bit
  Filterdata [in]: filtered data
  Return: 0 - success, other failures refer to the error code definition.
- UHFStartLogging
  Interface Description: start reading temperature tag.
  int UHFStartLogging (unsigned char mask_bank, unsigned short mask_addr, unsigned short mask_len, unsigned char *mask_data, float min_temp, float max_temp, unsigned short work_delay, unsigned short work_interval);
  Parameters:
  mask_ Bank [in]: mask data area, 1-epc, 2-tid, 3-user
  mask_ Addr [in]: Mask address, unit - bit
  mask_ Len [in]: mask length, unit - bit
  mask_ Data [in]: mask data
  min_ Temp [in]: minimum limit temperature
  max_ Temp [in]: maximum limit temperature
  work_ Time [in]: the continuous working time of each cycle, unit - MS
  work_ Interval [in]: duration of each work cycle, unit - milliseconds
  Return: 0 - success, other failures refer to the error code definition.
- UHFStopLogging
  Interface Description: stop reading temperature tag.
  int UHFStopLogging (unsigned char mask_bank, unsigned short mask_addr, unsigned short mask_len, unsigned char *mask_data, unsigned long password);
  Parameters:
  mask_ Bank [in]: mask data area, 1-epc, 2-tid, 3-user
  mask_ Addr [in]: Mask address, unit - bit
  mask_ Len [in]: mask length, unit - bit
  mask_ Data [in]: mask data
  Password [in]: access password, 32bits
  Return: 0 - success, other failures refer to the error code definition.
- UHFCheckOpMode
  Interface Description: read the temperature of the current temperature tag.
  int UHFCheckOpMode (unsigned char mask_bank, unsigned short mask_addr, unsigned short mask_len, unsigned char *mask_data);
  Parameters:
  mask_ Bank [in]: mask data area, 1-epc, 2-tid, 3-user
  mask_ Addr [in]: Mask address, unit - bit
  mask_ Len [in]: mask length, unit - bit
  mask_ Data [in]: mask data
  Return: the tag temperature is greater than 0. Refer to the error code definition for other failures.
- UHFSetProtocolType
  Interface Description: set UHF protocol type.
  int UHFSetProtocolType(unsigned char type);

Parameters:
Type [in]: protocol type, 0-iso18000-6c, 1-gb / t29768 national standard protocol, 2-gjb7377.1
Return: 0 - success, other failures refer to the error code definition.

- UHFGetProtocolType
  Interface Description: get protocol type.
  int UHFGetProtocolType (unsigned char *type);
  Parameters:
  Type [out]: 1byte, protocol type, 0-iso18000-6c protocol, 1-gb / t29768 national standard protocol, 2-gjb7377.1 national military standard protocol
  Return: 0 - success, other failures refer to the error code definition.
- UHFDwell
  Interface Description: UHF tag dwell.
  int UHFDwell (unsigned int dwell, unsigned int count);
  Parameters:
  Dwell: dwell mode
  Count: Times
  Return: 0 - success, other failures refer to the error code definition.
- UHF_GetIdleMs
  Interface Description: get the time when all connections are idle.
  unsigned long UHF_ GetIdleMs();
  Parameters: None
  Return: idle time in milliseconds.
- UHF_GetIdleMsById
  Interface Description: get the time when the specified link is continuously idle.
  unsigned long UHF_ GetIdleMsById(int id);
  Parameters:
  ID [in]: ID of the link
  Return: idle time in milliseconds.

4.3 Upgrade

- UHFJump2Boot
  Interface Description: enter bootloader and prepare to upgrade firmware.
  int UHFJump2Boot(unsigned char flag);
  Parameters:
  flag[in]:
  Return: 0 - success, other failures refer to the error code definition.
- UHFStartUpd
  Interface Description: start upgrading.
  int UHFStartUpd();
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- UHFUpdating
  Interface Description: import firmware data with a fixed length of 64bytes.
  int UHFUpdating(unsigned char *buf);
  Parameters:
  BUF [in]: address of imported data pointer, fixed 64 bytes
  Return: 0 - success, other failures refer to the error code definition.
- UHF_Upgrade
  Interface Description: import firmware data with a fixed length of 64bytes.
  int UHF_ Upgrade(const unsigned char *buf, unsigned short length);
  Parameters:
  BUF [in]: address of imported data pointer

Length [in]: length of imported data
Return: 0 - success, other failures refer to the error code definition.

- UHFStopUpdate
  Interface Description: stop downloading firmware.
  int UHFStopUpdate();
  Parameters: None
  Parameters: None
  Return: None
  Return: None

4.4 HF Interfaces

- HFGetVer
  Interface Description: obtain the version of HF module.
  int HFGetVer(char *pcVer, unsigned char *pcVerLen);
  Parameters:
  Pcver [out]: version number, string
  Pcverlen [out]: version number character length, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFResetModule
  Interface Description: reset high frequency module.
  int HFResetModule();
  Parameters: None
  Return: None
- HFBuzzerWork
  Interface Description: turn on the buzzer of high frequency module.
  int HFBuzzerWork(int ms);
  Parameter: Ms [in]: continuous working time, unit: ms
  Return: 0 - success, other failures refer to the error code definition.
- HFSetVolume
  Interface Description: set the volume of HF module.
  int HFSetVolume(int vol);
  Parameters:
  Vol [in]: volume, 0 ~ 4095, the higher the value, the louder the sound
  Return: 0 - success, other failures refer to the error code definition.
- HFReadReg
  Interface Description: read the register status of HF module.
  int HFReadReg(unsigned char reg_addr, unsigned char *val);
  Parameters:
  reg_ Addr [in]: register address
  Val [out]: returned register value, 1byte
  Return: 0 - success, other failures refer to the error code definition.
- HFTurnOnRF
  Interface Description: turn on the high-frequency card reading antenna. The default state is on
  int HFTurnOnRF(void);
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- HFTurnOffRF
  Interface Description: turn off HF antenna.
  int HFTurnOffRF(void);
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- HFRequestTypeA

Interface Description: card searching, applicable to 14443a protocol.
int HFRequestTypeA(int cMode, unsigned char *pcCardType);
Parameters:
Cmode [in]: 0x26 - find the card in idle state, 0x52 - find all the sensed cards
Pccardtype [out]: card type, 2bytes
Return: 0 - success, other failures refer to the error code definition.

- HFAnicollTypeA
  Interface Description: anticoll 14443a card.
  int HFAnticollTypeA(unsigned char *pcSnr, unsigned char *pcSnrLen);
  Parameters:
  Pcsnr [out]: unique serial number of the card
  Pcsnrlen [out]: 1byte, pcsnr length, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFSelectTypeA
  Interface Description: select 14443a card.
  int HFSelectTypeA(unsigned char *SAK);
  Parameters:
  Sak: type after card selection, 1byte
  Return: 0 - success, other failures refer to the error code definition.
- HFActivateTypeA
  Interface Description: activate 14443a card.
  int HFActivateTypeA(int cMode, unsigned char *pcATQA);
  Parameters:
  Cmode [in]: 0x26 - find the card in idle state, 0x52 - find all the sensed cards
  Pcatqa [out]: card type (2bytes) + uid length (1bytes) + uid (n bytes) + selection type (1bytes)
  Return: 0 - success, other failures refer to the error code definition.
- HFHaltTypeA
  Interface Description: halt 14443a card.
  int HFHaltTypeA(void);
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- HFAuthentication
  Interface Description: verify Mifare1 card key, such as S50 and S70 cards.
  int HFAuthentication(unsigned char cMode, unsigned char cBlock, unsigned char *pcKey);
  Parameters:
  Cmode [in]: 0x60 - verify zone a key, 0x61 - verify zone B key
  Cblock [in]: block address, such as S50 0 ~ 63
  Pckey [in]: key, 6bytes
  Return: 0 - success, other failures refer to the error code definition.
- HFReadBlock
  Interface Description: read Mifare1 card by block.
  int HFReadBlock(unsigned char cBlock, unsigned char *bdata);
  Parameters:
  Cblock: block address, such as S50 0 ~ 63
  Bdata: read data, 16bytes
  Return: 0 - success, other failures refer to the error code definition.
- HFWriteBlock
  Interface Description: write Mifare1 card by block.
  int HFWriteBlock(unsigned char cBlock, unsigned char *pcBlockData);
  Parameters:
  Cblock [in]: block address
  Pcblockdata [in]: data written to the block, 16bytes

Return: 0 - success, other failures refer to the error code definition.

- HFInitValue
  Interface Description: initialize e-wallet.
  int HFInitValue(unsigned char cBlock, unsigned long lValue);
  Parameters:
  Cblock [in]: block address
  Lvalue [in]: initial value, 32bits
  Return: 0 - success, other failures refer to the error code definition.
- HFReadValue
  Interface Description: read e-wallet balance.
  int HFReadValue(unsigned char cBlock, unsigned long *plValue);
  Parameters:
  Cblock [in]: block address
  Plvalue [in]: balance, 32bits
  Return: 0 - success, other failures refer to the error code definition.
- HFDecValue
  Interface Description: electronic wallet deduction.
  int HFDecValue(unsigned char blockValue, unsigned char blockResult, unsigned long value);
  Parameters:
  Blockvalue [in]: initial block address
  Blockresult [in]: target block address, the block address where the deduction result needs to be stored
  Value [in]: deduction amount, 32bits
  Return: 0 - success, other failures refer to the error code definition.
- HFIncValue
  Interface Description: e-wallet recharge.
  int HFIncValue(unsigned char blockValue, unsigned char blockResult, unsigned long value);
  Blockvalue [in]: initial block address
  Blockresult [in]: target block address, the block address where the recharge result needs to be stored
  Value [in]: recharge amount, 32bits
  Return: 0 - success, other failures refer to the error code definition.
- HFRestore
  Interface Description: recover e-wallet amount.
  int HFRestore(unsigned char cBlock);
  Parameters:
  Cblock [in]: e-wallet block address
  Return: 0 - success, other failures refer to the error code definition.
- HFTransfer
  Interface Description: convert electronic wallet.
  int HFTransfer(unsigned char cBlock);
  Parameters:
  Cblock [in]: e-wallet block address
  Return: 0 - success, other failures refer to the error code definition.
- HFUlAnticoll
  Interface Description: get ulight tag SN.
  int HFUlAnticoll(unsigned char *pcSnr, unsigned char *pcSnrLen);
  Parameters:
  Pcsnr [out]: SN number
  Pcsnrlen [out]: SN length, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFUlWrite

Interface Description: write ulight tag.
int HFUlWrite(unsigned char cBlock, unsigned char *pcWriteData, unsigned char cWriteLen);
Parameters:
Cblock [in]: block address
Pcwritedata [in]: write data
Cwritelen [in]: length of written data, unit - byte
Return: 0 - success, other failures refer to the error code definition.

- HFResetTypeA
  Interface Description: reset typea card.
  int HFResetTypeA(unsigned char *cardType, unsigned char *pcUid, unsigned char *cUidLen, unsigned char *pcATR, unsigned char *pcATRLen);
  Parameters:
  Cardtype [out]: card type, 2bytes
  Pcuid [out]: uid of the card
  Cuidlen [out]: uid length, unit - byte
  Pcatr [out]: reset information
  Pcatrlen [out]: reset message length, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFRatsTypeA
  Interface Description: reset the awakened 14443a CPU card.
  int HFRatsTypeA(unsigned char *pcATR, unsigned char *pcATRLen);
  Parameters:
  Pcatr [out]: reset information
  Pcatrlen [out]: reset message length, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFNtagGetVersion
  Interface Description: get ntag version information.
  int HFNtagGetVersion(unsigned char *version, unsigned char *vLen);
  Parameters:
  Version [out]: tag version
  Vlen [out]: length of version information, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFNtagAuth
  Interface Description: verify ntag tag key.
  int HFNtagAuth(unsigned char *password);
  Parameters:
  Password [in]: tag key, 4bytes
  Return: 0 - success, other failures refer to the error code definition.
- HFNtagRead
  Interface Description: read sector data of ntag tag, read 4 sectors at a time, 4 bytes per sector.
  int HFNtagRead(unsigned char sector, unsigned char *pdata, unsigned char *dataLen);
  Parameters:
  Sector [in]: start number of sector, 0 ~ n
  Pdata [out]: read data
  Datalen [out]: length of read data, unit - bytes
  Return: 0 - success, other failures refer to the error code definition.
- HFNtagWrite
  Interface Description: write one sector of ntag.
  int HFNtagWrite(unsigned char sector, unsigned char *pdata);
  Parameters:
  Sector [in]: sector number, 0 ~ n

Pdata [in]: data written, 4bytes
Return: 0 - success, other failures refer to the error code definition.

- HFHaltTypeB
  Interface Description: half TypeB card.
  int HFHaltTypeB(void);
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- HFResetTypeB
  Interface Description: reset TypeB card.
  int HFResetTypeB(unsigned char *pcInfo, unsigned char *pcInfoLen);
  Parameters:
  Pcinfo [out]: reset information of TypeB card
  Pcinfolen [out]: reset information length, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFGetUidTypeB
  Interface Description: obtain the second generation ID card uid of Chinese Residents.
  int HFGetUidTypeB(unsigned char *pcUid, unsigned char *cUidLen);
  Parameters:
  Pcuid [out]: ID card physical number UID
  Cuidlen [out]: length of uid data, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HFCpuCommand
  Interface Description: non contact CPU card command interaction, applicable to typea and TypeB cards.
  int HFCpuCommand(unsigned char *pcInCos, unsigned char cInLen, unsigned char *pcOutCos, unsigned char *pcOutLen);
  Parameters:
  Pcincos [in]: command data sent
  Cinlen [in]: length of command data sent, unit - byte
  Pcoutcos [out]: returned result data
  Pcoutlen [out]: length of returned data, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HF15639Inventory
  Interface Description: count 15693 cards.
  int HF15693Inventory(unsigned char cMode, unsigned char AFI, unsigned char *pcData, unsigned char *dataLen);
  Parameters:
  Cmode [in]: counting mode, value 0 ~ 3
  AFI [in]: value of AFI
  PCDATA [out]: card number
  Datalen [out]: 1byte, length of card number, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HF15639StayQuite
  Interface Description: put 15693 card into static state.
  int HF15693StayQuite(void);
  Parameters: None
  Return: 0 - success, other failures refer to the error code definition.
- HF15639Read
  Interface Description: read 15693 card.
  int HF15693Read(unsigned char cMode, unsigned char *pcUid, int cUidLen, int iStartBlock, int cBlockNum, unsigned char *pData, unsigned char *pDataLen);
  Parameters:

Cmode [in]: read mode, value 0 ~ 10
Pcuid [in]: uid data
Cuidlenp [ind: length of uid, unit - byte
Istartblock [in]: block address
Cblocknum [in]: number of blocks
Pdata [out]: read data
Pdatalen [out]: length of read data, unit - byte
Return: 0 - success, other failures refer to the error code definition.

- HF15693Write
  Interface Description: write 15693 card.
  int HF15693Write(unsigned char cMode, unsigned char *pcUid, int cUidLen, int iStartBlock, int cBlockNum, unsigned char *pwData, unsigned char wLen);
  Parameters:
  Cmode [in]: write mode, value 0 ~ 10
  Pcuid [in]: uid data
  Cuidlenp [ind: length of uid, unit - byte
  Istartblock [in]: block address
  Cblocknum [in]: number of blocks
  Pwdata [in]: data written
  Wlen [in]: length of written data, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HF15693LockBlock
  Interface Description: lock the area of 15693 card.
  int HF15693LockBlock(unsigned char cMode, unsigned char *pcUid, unsigned char cUidLen, int iStartBlock, unsigned char cBlockNum);
  Parameters:
  Cmode [in]: write mode, value 0 ~ 10
  Pcuid [in]: uid data
  Cuidlenp [ind: length of uid, unit - byte
  Istartblock [in]: block address
  Cblocknum [in]: number of blocks
  Return: 0 - success, other failures refer to the error code definition.
- HF15693Select
  Interface Description: select 15693 card and get card information.
  int HF15693Select(unsigned char *pcInfo, unsigned char *pcInfoLen);
  Parameters:
  Pcinfo [out]: card information
  Pcinfolen [out]: card information length, 1byte, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HF15693ResetReady
  Interface Description: reset 15693 card and obtain card information.
  int HF15693ResetReady(unsigned char *pcInfo, unsigned char *pcInfoLen);
  Pcinfo [out]: card information
  Pcinfolen [out]: card information length, 1byte, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- HF15693GetSystemInfo
  Interface Description: obtain 15693 card system information.
  int HF15693GetSystemInfo(unsigned char cMode, unsigned char *pcUid, unsigned char cUidLen, unsigned char *pcInfo, unsigned char *pcInfoLen);
  Parameters:
  Cmode [in]: mode, value 0 ~ 9
  Pcuid [in]: uid data

Cuidlen [in]: uid length
Pcinfo [out]: card system information
Pcinfolen [out]: card information length, 1byte, unit - byte
Return: 0 - success, other failures refer to the error code definition.

- HF15693WriteAFI
  Interface Description: write 15693 card AFI.
  int HF15693WriteAFI(unsigned char cMode, unsigned char *pcUid, int cUidLen, unsigned char cAFI);
  Parameters:
  Cmode [in]: mode, value 0 ~ 9
  Pcuid [in]: uid data
  Cuidlen [in]: uid length
  CFI [in]: value of AFI
  Return: 0 - success, other failures refer to the error code definition.
- HF15693LockAFI
  Interface Description: lock 15693afi.
  int HF15693LockAFI(unsigned char cMode, unsigned char *pcUid, unsigned char cUidLen);
  Parameters:
  Cmode [in]: mode, value 0 ~ 9
  Pcuid [in]: uid data
  Cuidlen [in]: uid length
  Return: 0 - success, other failures refer to the error code definition.
- HF15693WriteDsfid
  Interface Description: write 15693 card dsfid.
  int HF15693WriteDsfid(unsigned char cMode, unsigned char *pcUid, unsigned char cUidLen, unsigned char cDSFID);
  Parameters:
  Cmode [in]: mode, value 0 ~ 9
  Pcuid [in]: uid data
  Cuidlen [in]: uid length
  Cdsfid [in]: value of dsfid
  Return: 0 - success, other failures refer to the error code definition.
- HF15693LockDSFID
  Interface Description: lock the value of dsfid.
  int HF15693LockDSFID(unsigned char cMode, unsigned char *pcUid, unsigned char cUidLen);
  Parameters:
  Cmode [in]: mode, value 0 ~ 9
  Pcuid [in]: uid data
  Cuidlen [in]: uid length
  Return: 0 - success, other failures refer to the error code definition.
- HF15693GetMultipleBlock
  Interface Description: read data of multiple blocks of 15693 card.
  int HF15693GetMultipleBlock(unsigned char cMode, unsigned char *pcUid, unsigned char cUidLen, int iStartBlock, unsigned char cBlockNum, unsigned char *pcData, unsigned char *pcDataLen);
  Parameters:
  Cmode [in]: mode, value 0 ~ 9
  Pcuid [in]: uid data
  Cuidlen [in]: uid length
  Istartblock [in]: start block address of read data
  Cblocknum [in]: number of consecutive read blocks
  PCDATA [out]: read data

Pcdatalen [out]: 1byte, length of returned data, unit - byte
Return: 0 - success, other failures refer to the error code definition.

- HF15693TransferCmd
  Interface Description: use command interaction for 15693 card.
  int HF15693TransferCmd(unsigned char *pcInCmd, unsigned char cInLen, unsigned char *pcOutCmd, unsigned char *pcOutLen);
  Parameters:
  Pcincmd [in]: send command
  Cinlen [in]: length of sending command
  Pcoutcmd [out]: receive command
  Pcoutlen [out]: received command length, 1byte, unit - byte
  Return: 0 - success, other failures refer to the error code definition.

4.5 IC card interface

- SmartCardInit
  Interface Description: the initialization card slot is used for the contact. CPU card or PSAM card. It is called before using the reset card
  int SmartCardInit(unsigned char cSlotNum);
  Parameters:
  Cslotnum [in]: card slot number, such as R1 0-contact CPU card, 1-psam card
  Return: 0 - success, other failures refer to the error code definition.
- SmartCardFree
  Interface Description: release the card slot and call it after use.
  int SmartCardFree(unsigned char cSlotNum);
  Parameters:
  Cslotnum [in]: card slot number, such as R1 0-contact CPU card, 1-psam card
  Return: 0 - success, other failures refer to the error code definition.
- SmartCardReset
  Interface Description: reset card.
  int SmartCardReset(unsigned char cSlotNum, unsigned char *pcATR, unsigned char *pcATRLen);
  Cslotnum [in]: card slot number
  Pcatr [out]: reset information
  Pcatrlen [out]: length of reset information, 1byte, unit - byte
  Return: 0 - success, other failures refer to the error code definition.
- SmartCardTransferCmd
  Interface Description: contact card or PSAM card executes commands.
  int SmartCardTransferCmd(unsigned char cSlotNum, unsigned char *pcInCmd, unsigned char cInLen, unsigned char *pcOutCmd, unsigned char *pcOutLen);
  Parameters:
  Cslotnum [in]: card slot number
  Pcincmd [in]: send command
  Cinlen [in]: length of sending command
  Pcoutcmd [out]: return command
  Pcoutlen [out]: returns the length of the command, 1byte, unit - byte
  Return: 0 - success, other failures refer to the error code definition.

4.6 Tool Interface

- SetLogLevel
  Interface Description: set the output level of debugging information. It is not output by default
  void SetLogLevel(int level);
  Parameters:

Level [in]: output level, 0-no output, 1-fatal log output, 2-error log output, 3-BASIC log output, 4-detailed log output. The higher the level, the more detailed the output. The default value is 0
Return: None

- SaveLogFile
  Interface Description: save the log file. The file name is named after the date and time
  void SaveLogFile(int save);
  Parameters:
  Save [in]: save tag
  Return: None
- AESHandle
  Interface Description: AES encryption
  int AESHandle(unsigned char isEnc, unsigned char *key, unsigned short keylen, unsigned char *buf, unsigned long inLen);
  Parameters:
  Isenc [in]: 0-decryption, 1-encryption
  Key [in]: the key should be 16 / 24 / 32bytes
  Keylen [in]: key length, which should be 16 / 24 / 32, unit - byte
  BUF [in out]: processed data
  Inlen [in]: buf length, must be 16 * n, n is a natural number greater than 0, unit: bytes
  Return: length of encrypted and decrypted data, less than or equal to
  0 error.

5. Error Code Definition

Return Values	Names	Definition
0	ERR_SUCCESS	Success
1	ERR_FAILURE	Fail
2	ERR_CONNECT_FAILURE	Connection Fail
3	ERR_OPEN_PORT_FAILURE	Open Port Fail
4	ERR_OPEN_PORT_INVALID	Invalid Port
5	ERR_CONFIG_PORT_ERROR	Config port err
6	ERR_OPEN_USB_FAILURE	Open USB Fail
7	ERR_NOT_CREATE_SOCKET	Create Socket connection Fail
251	ERR_RECV_PACK	Receive pack fail
252	ERR_SEND_FAIL	Send fail
253	ERR_RECV_FAIL	Receive fail
254	ERR_RECV_NODATA	No data receive
255	ERR_OPERA_FAIL	Write fail
-1	ERR_COMMON_FAILURE	Common err

-2	ERR_PARAM_ERROR	Param err
-3	ERR_15693_GET_INFO	Get 15693 data err
-4	ERR_INVALID_RETURN	Return data err