

# **USB E7umf Library**

**Function manual**

# Contents

- 1. UHF\_CONNECT .....2
- 2. UHF\_DISCONNECT .....2
- 3. UHF\_READ.....3
- 4. UHF\_WRITE .....4
- 5. UHF\_ACTION .....5
- 6. UHF\_SETACCESSPASSWORD .....5
- 7. UHF\_LOCKMEMORY .....6
- 8. UHF\_INVENTORY .....8
- 9. UHF\_KILLTAG.....9

# 1. uhf\_connect

**int uhf\_connect(int port, long baud)**

## Description

Connect reader.

## Parameters

port:

100: USB interface

0: Serial port(COM1)

1: Serial port(COM2)

2: Serial port(COM3)

...

baud: Baud rate(9600-115200)

## Return Value

>0 is device handle, otherwise connect failed

## Example

```
int icdev;
```

```
icdev = uhf_connect(100, 115200); // USB port
```

# 2. uhf\_disconnect

**int uhf\_disconnect(int icdev)**

## Description

Disconnect reader.

## Parameters

icdev: Handle of reader.

## Return Value

=0 correct

other error

### Example

```
int st;  
st = uhf_disconnect(icdev);
```

## 3. uhf\_read

**int uhf\_read (int icdev, unsigned char infoType, unsigned char address, unsigned int rlen, unsigned char\* pData)**

### Description

Read data from UHF tag, read data according to the infoType, address, rlen and other parameters.

### Parameters

icdev: Handle of reader.

infoType: 1: EPC

2: TID

3: USER

4: reserved

address: start address

rlen: length of the data to read(will get rlen\*4 bytes data)

pData: Data read

### Return Value

<>0 error, the absolute value is error code

= 0 read data correctly

### Example

```
unsigned char[50] DataBuffer;  
int st;  
  
st = uhf_read(icdev, 1, 0, 8, DataBuffer);  
/* Read EPC 8 words data to DataBuffer start from address 0(will get 32  
bytes data)*/
```

## 4. uhf\_write

**int uhf\_write (int icdev, unsigned char infoType, unsigned char address, unsigned int wlen, unsigned char\* pData)**

### Description

Write UHF tag

### Parameters

icdev: Handle of reader.

infoType: 1: EPC

2: TID

3: USER

4: reserved

address: start address

wlen: length of the data to write(will write wlen\*4 bytes data)

pData: Data for write

### Return Value

<>0 error, the absolute value is error code

= 0 write data correctly

### Example

```
unsigned char[50] DataBuffer;

int st;

int i;

for(i=0;i<50;i++)
    DataBuffer[i] = 0x35;

st = uhf_write(icdev, 1, 2, 6, DataBuffer);

/* Write 6 words data to EPC start from address 2(will write 24 bytes
data)*/
```

## 5. uhf\_action

**int uhf\_action (int icdev, unsigned char action, unsigned char time)**

### Description

Control buzzer and led

### Parameters

icdev: Handle of reader.

action: 1: Beep

2: Red led on

4: Green led on

8: Yellow led on

time: Unit: 10ms

### Return Value

<>0 error, the absolute value is error code

= 0 Correct execution

### Example

```
int st;
```

```
st = uhf_action(icdev, (1 | 4), 50);
```

```
/* beep and green led on 500ms*/
```

## 6. uhf\_setAccessPassword

**int uhf\_setAccessPassword (int icdev, unsigned char\* AccessPassword)**

### Description

Set Access Password

### Parameters

icdev: Handle of reader.

AccessPassword: Access password of tag

### Return Value

<>0 error, the absolute value is error code

= 0 Correct execution

### Example

```
int st;  
  
unsigned char password[10]={0x31,0x32, 0x33,0x34, 0x35,0x36,  
0x37,0x38};  
  
st = uhf_setAccessPassword(icdev, password);
```

## 7. uhf\_lockMemory

Lock function is used to:

- Lock individual passwords (Kill pwd & Access pwd) – preventing or allowing subsequent reads and writes of that password
- Lock individual memory banks(EPC, TID, USER) – preventing or allowing subsequent writes to that memory bank
- Permalock – make the lock status permanently unchangeable for a password or memory bank

mask field bits

		Kill pwd		Access pwd		EPC memory		TID memory		User memory	
11	10	9	8	7	6	5	4	3	2	1	0
0	0	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write

bit = 0: Ignore the associated Action field and retain the current lock setting.

bit = 1: Implement the associated Action field and overwrite the current lock setting.

action field bits

		Kill pwd		Access pwd		EPC memory		TID memory		User memory	
11	10	9	8	7	6	5	4	3	2	1	0
0	0	pwd read/write	perma lock	pwd read/write	perma lock	pwd write	perma lock	pwd write	perma lock	pwd write	perma lock

bit = 0: Deassert lock for the associated memory location.

bit = 1: Assert lock or permalock for the associated memory location.

#### Action-field functionality

pwd-write	permalock	Description
0	0	Associated memory bank is writeable
0	1	Associated memory bank is permanently writeable and may never be locked
1	0	Associated memory bank is writeable if ACCESS password is correct
1	1	Associated memory bank is not writeable

pwd-read/write	permalock	Description
0	0	Associated password location is readable and writeable
0	1	Associated password location is readable and writeable and may never be locked
1	0	Associated password location is readable and writeable if ACCESS password is correct
1	1	Associated password location is not readable and writeable

In memory lock operation, need ACCESS pwd to protect writable or readable/writable, we need `uhf_setAccessPassword` function to set ACCESS operation, please refer to `uhf_setAccessPassword` function.

### **int uhf\_lockMemory (int icdev, unsigned char\* lockSetting)**

#### **Description**

Lock Memory

#### **Parameters**

icdev: Handle of reader.

lockSetting: Lock setting, 6 bytes ASCII character

#### **Return Value**

<>0 error, the absolute value is error code

= 0 Correct execution

#### **Example**



```

int st;

unsigned char lockSetting[10]={0x30,0x30, 0x32,0x30, 0x30,0x32}; //002,
002, lock user bank

st = uhf_lockMemory(icdev, lockSetting);

```

## 8. uhf\_inventory

```

int uhf_inventory (int icdev, unsigned int *tagCount, unsigned int
*dataLen, unsigned char *pData)

```

### Description

Multiple EPC reads.

### Parameters

icdev: Handle of reader.

tagCount: EPC count read

dataLen: length of the pData

pData: Data read ( data1\_length(1 byte), EPC1\_readcount(1 byte),  
EPC1(data1\_length-1 bytes), data2\_length(1 byte), EPC2\_readcount(1 byte),  
EPC2(data2\_length-1 bytes)... )

### Return Value

<>0 error, the absolute value is error code

= 0 read data correctly

### Example

```

unsigned char[2] tagCount;

unsigned char[2] dataLen;

unsigned char[50] DataBuffer;

int st;

st = uhf_read(icdev, tagCount, dataLen, DataBuffer);

```

## 9. uhf\_killTag

**int uhf\_killTag(int icdev, unsigned char\* KillPassword)**

### Description

Kill tag

### Parameters

icdev: Handle of reader.

KillPassword: Kill password of tag

### Return Value

<>0 error, the absolute value is error code

= 0 Correct execution

### Example

```
int st;

unsigned char password[10]={0x31,0x32, 0x33,0x34, 0x35,0x36,
0x37,0x38};

st = uhf_killTag(icdev, password);
```