DOTR-900

D.O.Tel

Modification

No	Date	Change Point	Writer
1	2012-08-01	Initialization Manual	Miss.Ji
	2013-10-22	API added	Miss.Ji

Content

1. Introduction.

This document describes API function definition and usage of DOTR-900.

The DOTR-900 is UHF reader and is connected to various mobile devices by Bluetooth.

This document includes example code and how to use demo application to help understanding easily for developer.

2. Development Tool

Development tool and language

- .Net Framework 2.0
- Visual Studio 2005 C++, C#, Visual Basic

Development Platform

Download the SDK to develop DOTH-300 application according to using platform, WinCE and WM.

- **WINCE SDK Download**
- **Windows Mobile SDK Download**
- **Microsoft ActiveSync Download**

3. Providing Package

The package is provided for developing application.

Device	Document	Header File	Using DLL
R900	R900 Manual	R900LIB.h Rfid_def.h	R900LIB.dll UHFAPI.dll Uhfrfid.dll

4. Function Definition and Example Code

4.1 R900LIB_UploadInventory

This function is to upload the recoded TAG information in Reader.

BOOL R900LIB_UploadInventory(DWORD (*UploadInv)(typeInventoried *), UINT32 index, UINT32 count)

Parameters

DWORD (*UploadInv)(typeInventoried *)
A callback function to get the read TAG information
UINT32 id
An index value of stored TAG
UINT32 count
A number of TAG information

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

If index is 0 and count is 0, all TAG information can be read.
TAG information is delivered by callback function.

Example Code

C++

#include "r900lib.h"

R900Manager.cpp

```
typedef struct {
WORD handle;
BYTE *id;
WORD len;
int count;
WORD wb_rssi;
WORD nb_rssi;
WORD lna_gain;
float rssidb;
tag_t first_time;
tag_t last_time;
} typeInventoried;

DWORD cbCountInventory(typeInventoried *pTagInfo)
{
   g_iReaderTagCount++;
   return 0;
}

//Load all TAG information
if ( !R900LIB_UploadInventory(cbCountInventory, 0, 0) )
      return -1;
```

C#

FormInventory.cs

```
public static UHFAPI_NET.UHFAPI_NET R900APP;
R900APP = new UHFAPI_NET.UHFAPI_NET();
```

Form1.R900APP.R900LIB_UploadInventory();

4.2 R900LIB_ClearInventory

This function is to remove the stored TAG information in Reader.

BOOL R900LIB_ClearInventory()

Parameters

None.

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

The DOTR-900 must be in the connection state.

Example Code

C++

#include "r900lib.h"

R900Manager.cpp

```
int ClearInventory(CWnd *thisWnd)
{
   //Load all TAG information
   return R900LIB_ClearInventory();
}
```

C#

FormInventory.cs

public static UHFAPI_NET.UHFAPI_NET R900APP;

R900APP = new UHFAPI_NET.UHFAPI_NET();

Form1.R900APP.R900LIB_ClearInventory();

4.3 R900LIB_RefreshStatus

This function is to update the status of Reader.

BOOL R900LIB_RefreshStatus()

Parameters

None

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

The processed value is stored to library.

The status of trigger button can be read by R900LIB_IsTriggerEvent().

Example Code

C++

#include "r900lib.h"

R900Manager.cpp

```
BOOL GetTriggerState(BOOL refresh)
{
  BOOL status;
  if ( refresh )
      R900LIB_RefreshStatus();
  R900LIB_IsTriggerEvent(&status);

  return status;
}
```

C#

Class1.cs

```
unsafe
{
        if (_R900LIB_RefreshStatus())
        {
          Int32 trigger_on;
          _R900LIB_IsTriggerEvent(&trigger_on);
          evtR900TriggerEvent((bool)(trigger_on != 0));
            }
```

}

4.4 R900LIB_IsTriggerEvent

This function is to check the status of trigger button.

BOOL R900LIB_IsTriggerEvent(BOOL *status)

Parameters

BOOL *status

Indicate the current status of Trigger button

Return Values

Return TRUE when the status of trigger button is changed

Return FALSE when the status of trigger button is not changed

Remarks

The Status can be NULL

Example Code

C++

#include "r900lib.h"

RfidhostDlg.cpp

```
BOOL trigger_on;
if ( R900LIB_IsTriggerEvent(&trigger_on) )
    ModuleTriggerEventService(trigger_on);
```

C#

Class1.cs

```
if (evtR900TriggerEvent != null)
{
   Int32 trigger;

   if (_R900LIB_IsTriggerEvent(&trigger))
       evtR900TriggerEvent((bool)(trigger!=0));
```

}

4.5 R900LIB_IsLinkEvent

This function is to check the current connection state.

BOOL R900LIB_IsLinkEvent (BOOL *status)

Parameters

BOOL *status

If this parameter is TRUE, it indicates the connection state

Return Values

Return TRUE when the status is changed

Return FALSE when the status is not changed

Remarks

The Status can be NULL

4.6 R900LIB_SetBuzzerVolume

This function is to set a value for buzzer volume.

BOOL R900LIB_SetBuzzerVolume(UINT32 value, BOOL nv)

Parameters

UINT32 value

A value to set

BOOL nv : change non-volatile memory

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

The value is one of three values.
0 ; mute
1 ; low volume
2 : high volume

4.7 R900LIB_GetBuzzerVolume

This function is to get a value for buzzer volume.

BOOL R900LIB_GetBuzzerVolume(UINT32 *value)

Parameters

UINT32 *value

A pointer to get a vlaue

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

The value is one of three values.
0 ; mute
1; low volume
2: high volume

Example Code

C++

```
#include "r900lib.h"

BOOL GetBuzzerVolume(UINT32 *value)
{
    return R900LIB_GetBuzzerVolume(value);
}
UINT32 lvlBuzzer;

GetBuzzerVolume(&lvlBuzzer);
```

4.8 R900LIB_SetPowerOffDelay

This function is to set a value for auto power off delay.

BOOL R900LIB_SetPowerOffDelay(UINT32 value, BOOL nv)

Parameters

UINT32 value

A value to set (in secs)

BOOL nv : change non-volatile memory

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

Example Code

C++

```
#include "r900lib.h"

BOOL SetAutoOffDelay(UINT32 value)
{
    return R900LIB_SetPowerOffDelay(value, FALSE);
}
UINT32 secAutoOff;
SetAutoOffDelay(secAutoOff);
```

4.9 R900LIB_GetPowerOffDelay

This function is to get a value for auto power off delay.

BOOL R900LIB_GetPowerOffDelay(UINT32 *value)

Parameters

UINT32 *value

A pointer to get a value

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

Example Code

C++

```
#include "r900lib.h"

BOOL GetAutoOffDelay(UINT32 *value)
{
    return R900LIB_GetPowerOffDelay(value);
}

UINT32 secAutoOff;
GetAutoOffDelay(&secAutoOff);
```

4.10 R900LIB_GetBatteryMeter

This function is to get a value for battery remains.

BOOL R900LIB_GetBatteryMeter(UINT32 *value)

Parameters

UINT32 *value

A pointer to get (% value)

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

Example Code

C++

```
#include "r900lib.h"

BOOL GetBatteryLevel(UINT32 *value)
{
   return R900LIB_GetBatteryMeter(value);
}
UINT32 lvlBattery;
GetBatteryLevel(&lvlBattery);
```

4.11 R900LIB_RunBatteryMoniter

This function is to trigger an event when battery remains is changed.

BOOL R900LIB_RunBatteryMoniter(BOOL run)

Parameters

BOOL run

If this parameter is TRUE, an event is triggered and an event is stopped if it is FALSE.

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

4.12 R900LIB_Beep

This function is to run a buzzer.

BOOL R900LIB_Beep(BOOL on)

Parameters

BOOL on

If this parameter is TRUE, beep for buzzer is on and beep is off if it is FALSE.

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

Example Code

C++

```
#include "r900lib.h"

BOOL BeepReader(BOOL on)
```

```
{
   return R900LIB_Beep(on);
}

BeepReader(on);
```

4.13 R900LIB_ReaderOff

This function is to turn off the connection with R900.

BOOL R900LIB_ReaderOff()

Parameters

None

Return Values

Return TRUE when success

Return FALSE when failure

Remarks

Example Code

C++

```
#include "r900lib.h"

BOOL ReaderOff()
{
   R900LIB_ReaderOff();
   // close handle
   if ( g_hLinkProc )
   {
     CloseHandle( g_hLinkProc );
     g_hLinkProc = NULL;
     }
   return UHFAPI_Close();
}

ReaderOff();
```

4.14 R900LIB_GetMacAddress

This function gets bluetooth Mac address

BOOL R900LIB_GetMacAddress(LPWSTR addr)

Parameters

LPWSTR addr

Get to Bluetooth Mac address(ex – 001122334455)

Return Values

Return TRUE when success

Return FALSE when failure

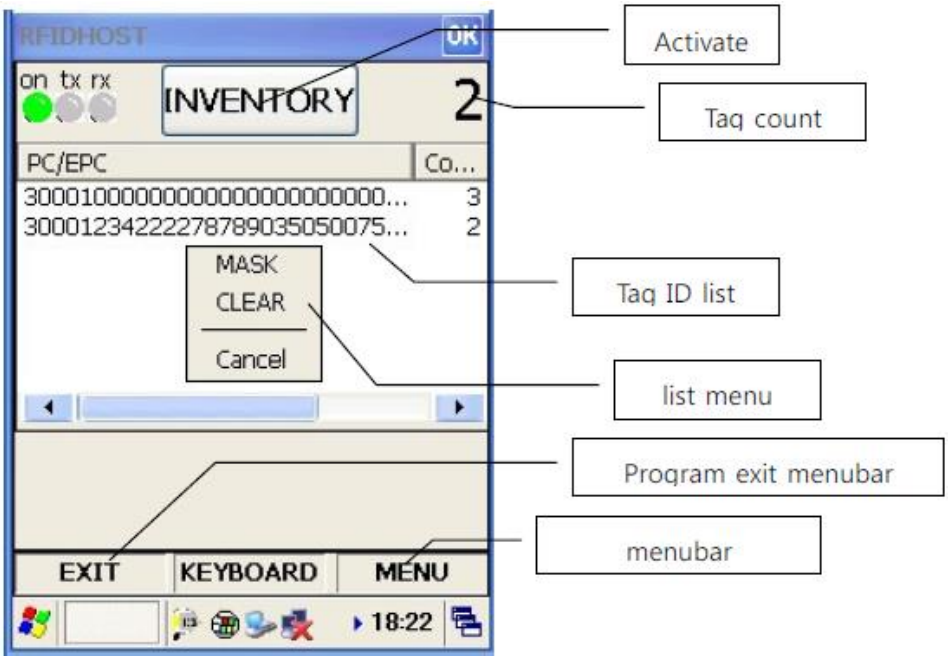
Remarks

The R900 must be linked to the host.

5. Specification for DEMO Application

This chapter describes main operation of RFID Demo application.

5.1 MENU

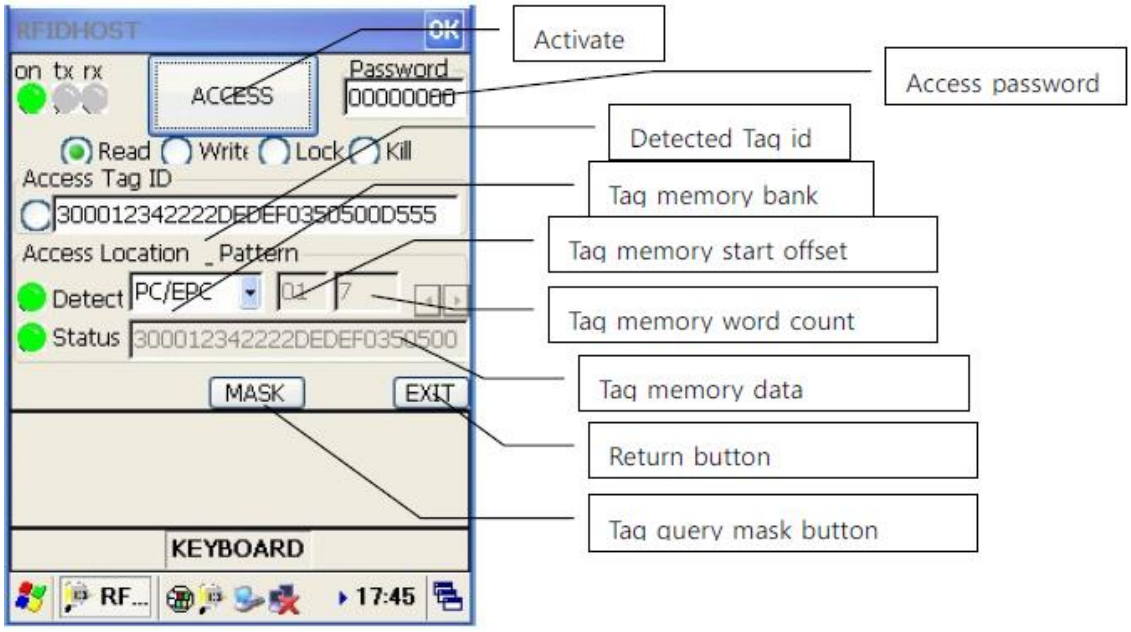


[Main Screen]

No.	UI	Description
(1)	Inventory	When push this button, a button is changed to [STOP] button. If push [STOP] button, process is stopped.
(2)	Tag count↵	The accumulated count.
(3)	Tag ID list↵	PC/EPC value appears in Tag ID list.
(4)	list menu↵	When click Tag list, List Menu appears. Through this menu, Mask can be set or inventory list can be erased.

(5)	Exit menubar	When push "EXIT" in left menu bar, program is terminated. When push "OK" in title bar, program is hidden to tray.
(6)	menubar	"MENU" icon in right menu bar is used for reading/writing the TAG etc.

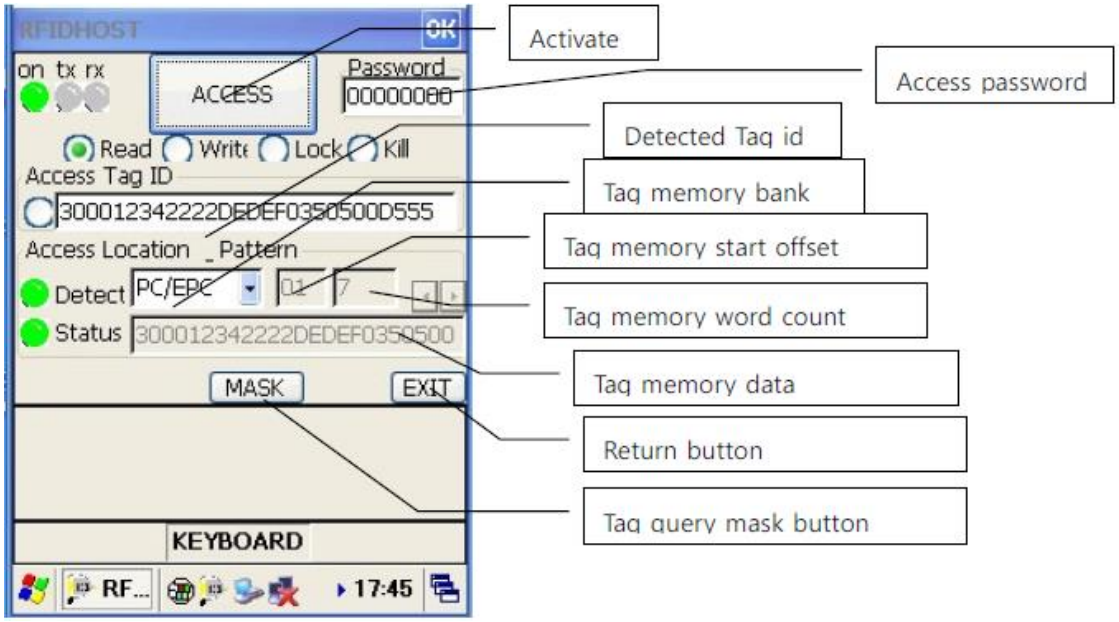
5.2 Composition and specification for Read



[READ Screen]

No.	UI	Description
(1)	ACCESS	Read a data from memory after recognizing the TAG. Select single_tag in config screen before running.
(2)	on	When happen an error, lamp is change to Red color.
(3)		When select a button left of Tag ID window, the TAG of selected ID is recognized and is accessed.

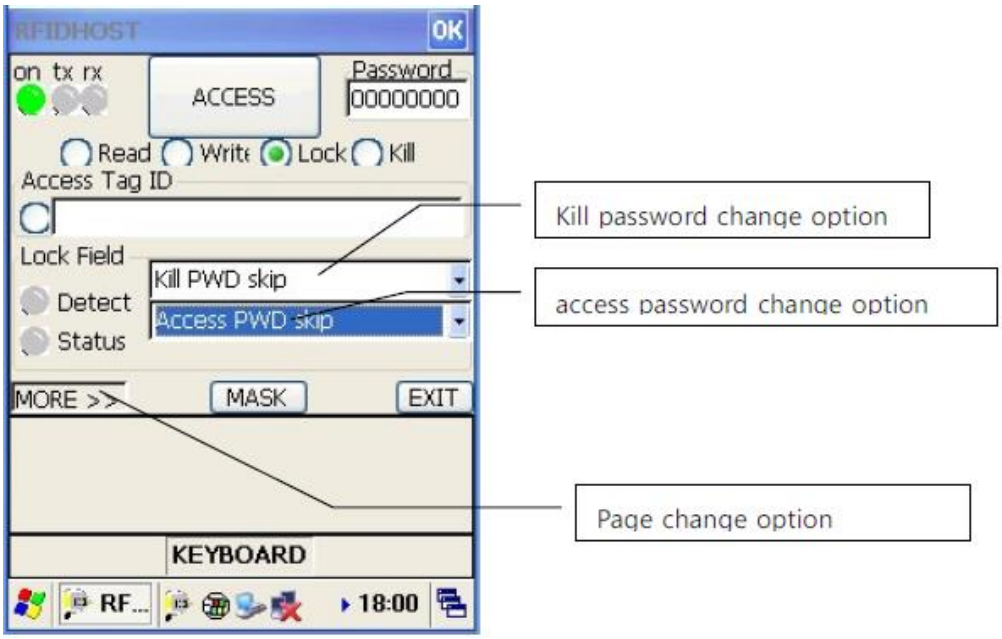
5.3 Composition and specification for Write

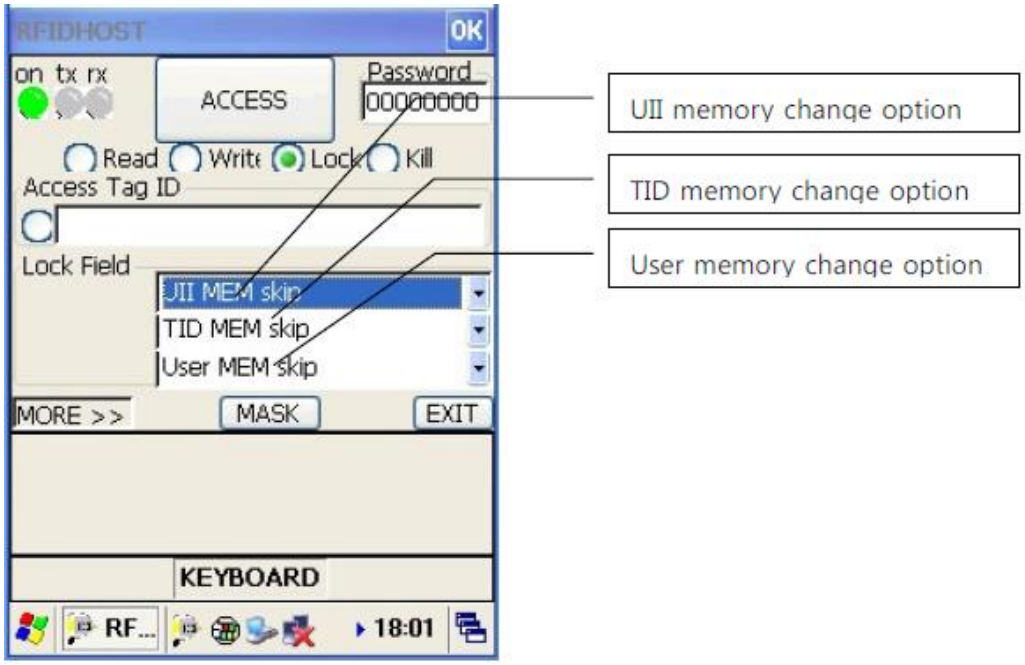


[WRITE Screen]

※ **This operation resembles to Read operation except selecting the Tag data**

5.4 Composition and specification for Lock



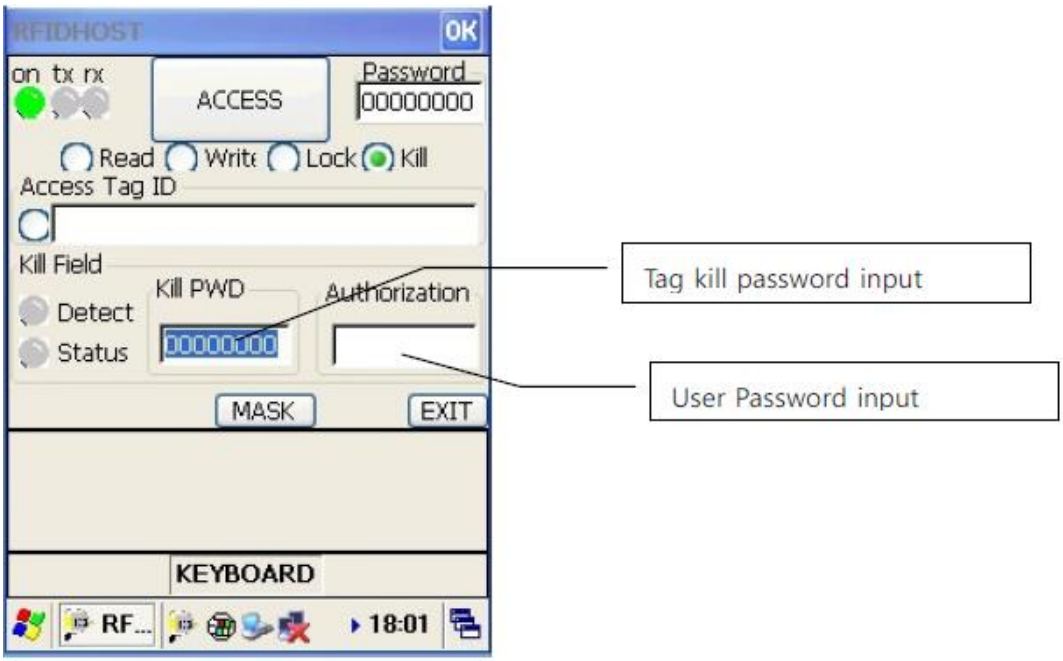


[Lock Screen]

※ **Push [ACCESS] button after selecting memory mode to change.**

If select [xxx skip], the lock status of a corresponding memory is not changed.

5.5 Composition and specification for Kill



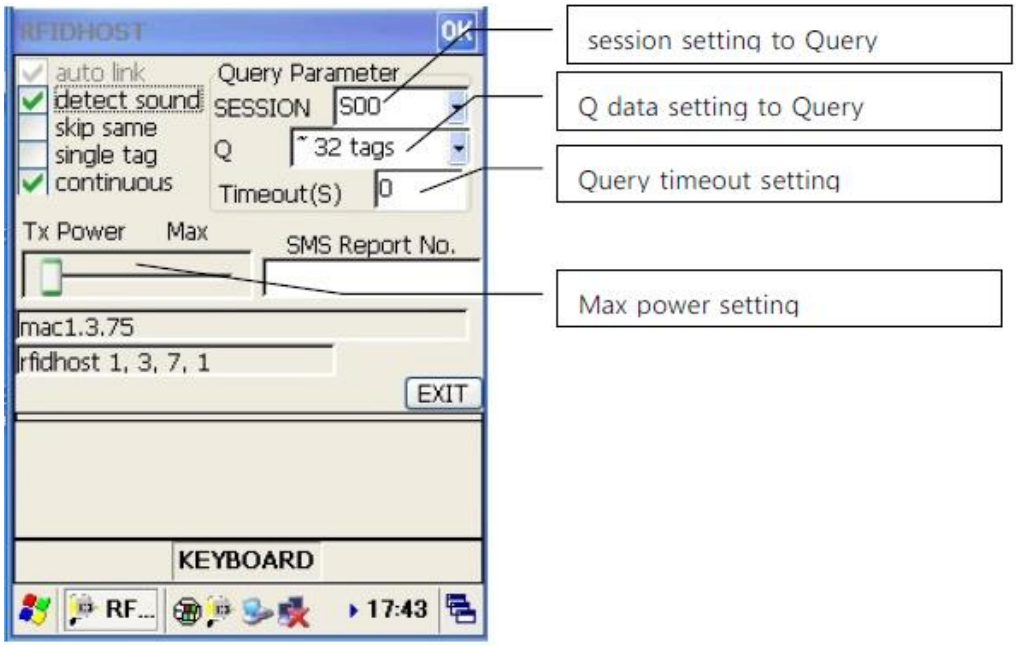
[KILL Screen]

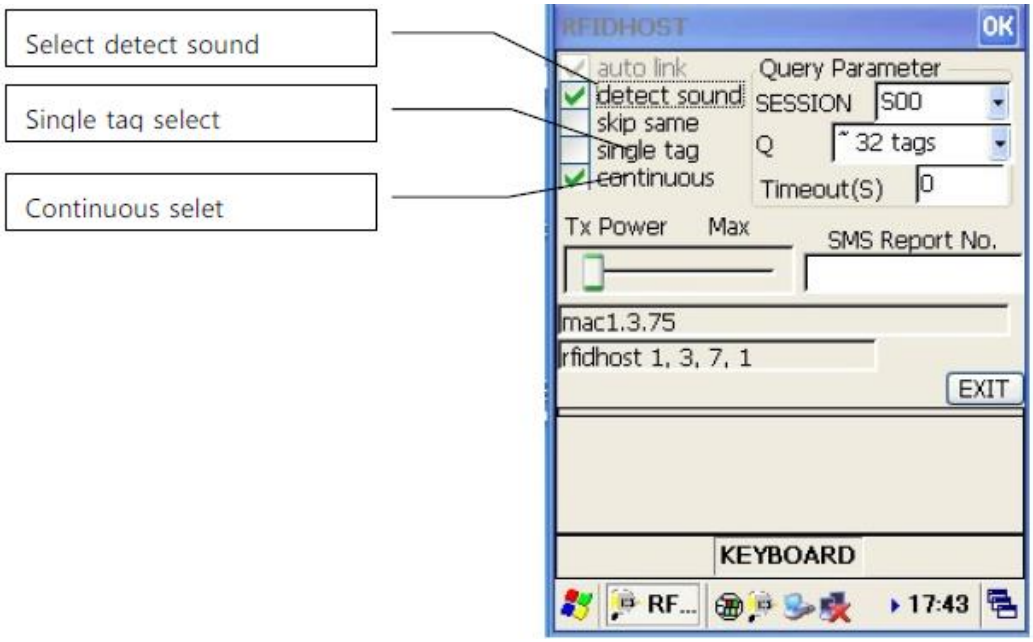
※ **Execute after inputting the kill password.**

User password is for preventing destruction of the TAG and is "tagkiller".

The killed tag can't be recovered.

5.6 Composition and specification for Config



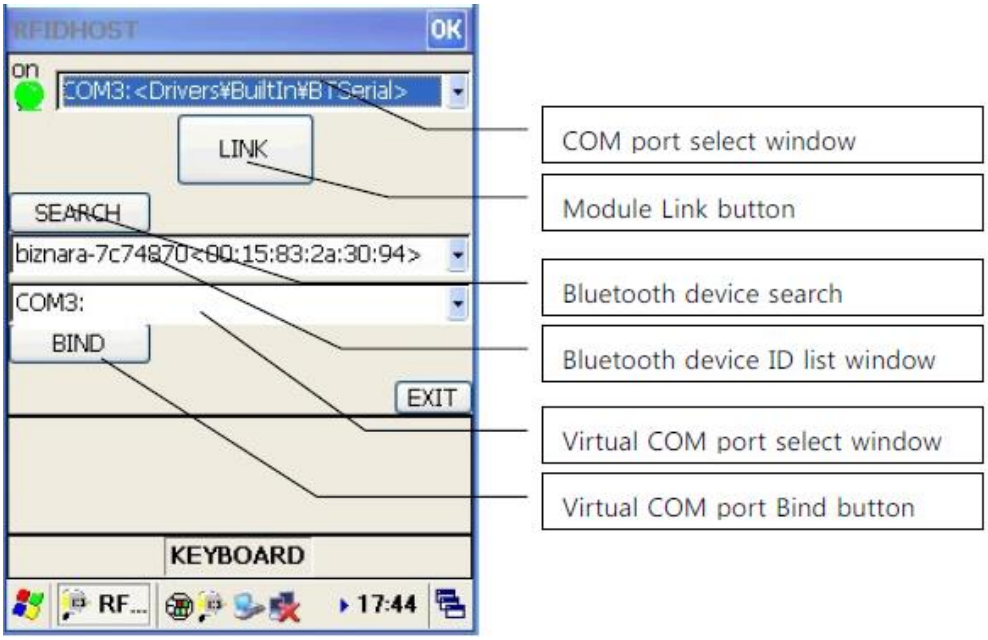


[Config Screen]

No.	UI	Description
(1)	session setting to query	Set a parameter value when query the TAG. This value can be set to inventory and access both.
(2)	query timeout setting	Timeout value (seconds)
(3)	Max power setting	Maximum decrease output power (max -9dB)

※ Select the single tag when ACCESS

5.7 Composition and specification for Link



[Link Screen]

No.	UI	Description
(1)	COM port 선택창	Select the com port and push [LINK] button. When success, screen is switched.
(2)	Bluetooth device 검색 실행 버튼	When connection with Bluetooth, Bluetooth device is connected to virtual COM port. Push [Bluetooth Device inquiry] button to search

		"HQ_UHF_Reader" device. Select the "HQ_UHF_Reader" device in Bluetooth device ID list and then select comport in Virtual COM port. And, push the [BIND] button.