RFID Development Documentation

V2.1

Version control

**Record of updating**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **author** | **version** | **description** |
| 2022-08-25 | TangTeng | v1.1 | This is a documentation of rfid’s api |
| 2022-11-24 | TangTeng | V2.0 | Add some useful methods and delete some methods,almost every method has been changed.so when you are ready to use those methods you need pay attention to the changing |
| 2023-2-1 | TangTeng | V2.1 | Added setting mask, setting link, and reading tag data through TID. Carry TID when short-distance support inventory |

**Table of contents**

[1.Common method usage process 4](#_Toc857)

[1.1 initialization 4](#_Toc23258)

[1.2 Inventory 4](#_Toc3117)

[1.3 read tag 4](#_Toc2402)

[1.4 write tag 5](#_Toc29740)

[1.5 get the power of output 5](#_Toc2751)

[1.6 set the power of output 5](#_Toc1950)

[2. Interface’s information 5](#_Toc16105)

[The callback of 6B tag: 5](#_Toc67)

[onInventoryTag 6](#_Toc27454)

[onInventoryTagEnd 6](#_Toc26028)

[3. RFID api 8](#_Toc3926)

[3.1 init【initialization】 8](#_Toc21415)

[3.2 release【release resources】 8](#_Toc24117)

[3.3 sendCommand【send instructions】 8](#_Toc9660)

[3.4 disConnect【disconnect rfid module】 8](#_Toc4226)

[3.5 isConnected【Determine the connection status of the module】 9](#_Toc26442)

[3.6 cancelAccessEpcMatch【clear EPC match】 9](#_Toc19481)

[3.7 clearTagMask【clear TAG mask setting】 9](#_Toc5866)

[3.8 startInventory【User define session and target inventory】 10](#_Toc27815)

[3.9 stopInventory【stop tag inventory】 10](#_Toc8975)

[3.10 getAccessEpcMatch【Query match EPC status.】 10](#_Toc2671)

[3.11 getFrequencyRegion【Query frequency region.】 10](#_Toc24632)

[3.12 getFirmwareVersion【Get Reader Firmware Version.】 11](#_Toc30735)

[3.13 getIdentifier【query identifier】 11](#_Toc28201)

[3.14 getInventoryBuffer【Get tag data and keep buffer】 11](#_Toc30575)

[3.15 getInventoryBufferTagCount【Query tag quantity in buffer】 12](#_Toc27533)

[3.16 getOutputPower【Query output power】 12](#_Toc10064)

[3.17 getReaderIdentifier【Get Reader Identifier】 12](#_Toc5091)

[3.18 getReaderTemperature【Query internal temperature】 12](#_Toc30216)

[3.19 iso180006BInventory【Inventory 18000-6B Tag】 13](#_Toc429)

[3.20 iso180006BLockTag【Lock 18000-6B Tag】 13](#_Toc2931)

[3.21 iso180006BQueryLockTag【Query 18000-6B Tag】 13](#_Toc24372)

[3.22 iso180006BReadTag【Read 18000-6B Tag】 14](#_Toc27428)

[3.23 iso180006BWriteTag【Write 18000-6B Tag】 14](#_Toc25191)

[3.24 killTag【Kill Tag】 14](#_Toc22869)

[3.25 lockTag【Lock Tag】 15](#_Toc5268)

[3.26readTag【read tag】 15](#_Toc8237)

[3.27 realTimeInventory【Inventory Tag(Read Time Mode)】 16](#_Toc11537)

[3.28 rest【Reset the specified address reader.】 16](#_Toc78)

[3.29 setAccessEpcMatch【Set Access EPC match】 17](#_Toc15887)

[3.30 setFrequencyRegion【Set frequency region】 17](#_Toc11984)

[3.31 setOutputPower【Set output power(Method 1)】 17](#_Toc11569)

[3.32 setTagMask【Set the mask filter the Tag】 18](#_Toc5137)

[3.33 setTrigger【Reader Trigger mode】 19](#_Toc32152)

[3.34 writeTag【Write Tag】 19](#_Toc23041)

[3.35 registerCallback【register CallBack】 20](#_Toc24445)

[3.36 unregisterCallback【Unregister callback】 20](#_Toc3539)

[3.37 getModule【get module】 20](#_Toc22943)

[3.38 getModuleFirmware【Get module firmware information】 20](#_Toc30783)

[3.39 writeEpc【write epc number】 21](#_Toc17158)

[3.40 readOnceTag【single tag query】 21](#_Toc10416)

[3.41 writeTagByTid【Write data to each storage area through tid】 21](#_Toc14310)

[3.42 setScanInterval【Set the inventory interval】 22](#_Toc21273)

[3.43 getScanInterval【Read Inventory Interval】 22](#_Toc19120)

[3.44 setQueryMode【config query model】 23](#_Toc23882)

[3.45 getQueryMode【get query mode】 23](#_Toc31376)

[3.46 GetReaderType【get rfid mode】 23](#_Toc8177)

[3.47 startRead【start inventory】 23](#_Toc741)

[3.48 scanRfid【Take inventory over a period of time】 24](#_Toc16563)

[3.49 setInventoryParameter【configure inventory parameters】 24](#_Toc29563)

[3.50 getDeviceId【Gets the serial number】 24](#_Toc30208)

[3.51 setCustomRegion【Set custom bands】 24](#_Toc16266)

[3.52 getCustomRegion【Get a custom band】 25](#_Toc19085)

[3.53 readDataByTid【Tag data is read via TID】 25](#_Toc8655)

[3.54 addMask【add mask】 26](#_Toc26010)

[3.55 clearMask【clear mask】 26](#_Toc26162)

[3.56 setProfile【Set the link configuration information】 26](#_Toc18031)

[3.57 setRange【set the distance】 27](#_Toc28119)

[3.58 getRange【get the distance】 27](#_Toc5299)

**1.Common method usage process**

* 1. **initialization**

|  |
| --- |
| **private void initRfid() {**  **// 在异步回调中拿到RFID实例**  **USDKManager.getInstance().init(BaseApplication.getContext(),new USDKManager.InitListener() {**  **@Override**  **public void onStatus(USDKManager.STATUS status) {**  **if ( status == USDKManager.STATUS.SUCCESS) {**  **Log.d(TAG, "initRfid() success.");**  **mRfidManager = USDKManager.getInstance().getRfidManager();**  **}else {**  **Log.d(TAG, "initRfid fail.");**  **}**  **}**  **});**  **}** |

* 1. **Inventory**

After initializing **RfidManager’s** instance,you can make an inventory of cargo by using **customizedSessionTargetInventory,**and you can get the result from the callback method of **OnInventoryTag**,which is defined in the interface of **IRfiCallbcak**.

**steps：**

1. Start working：

Rfidmanager.**customizedSessionTargetInventory**(btSession,btTarget,btRepeat);

1. Implements the method **onInventoryTag** of the interface .
   1. **read tag**

You can get the tag’s information by using the **readTag** method of **RfidManager**.

**steps：**

1. Read tag：rfidmanager.**readTag**(epc,btMemBank,btWordAdd,btWordCnt,btAryPassWord)，the method will return what you want to get from the target tag.
   1. **write tag**

You can write the tag by using the **writeTag** method of **RfidManger**。

Steps：

1. Write tag：Rfidmanager.**writeTag**(epc,btAryPassWord,btMemBank，btWordAdd，btWordCnt，btAryData),the method will return the result of the operation.

**1.5 get the power of output**

You can get current value of output by using

**RfidManager**.**getOutputPower**,using the **getOutputPower** method to send command.

1. Send the command of getting the power of output

**rfidmanager**.**getOutputPower**()

**1.6 set the power of output**

You can get current value of output by using the **setOutputPower** method ,using the **getOutputPower** method to send command.

1. Set output power：**rfidmanager**.**setOutputPower**()
2. **Interface’s information**

public interface IRfidCallback {

void onInventoryTag(String EPC,String TID,String RSSI);

void onInventoryTagEnd();

}

### The callback of 6B tag:

ITag6BCallback {

public void onRead6BTag( String strData) ;

public void onWrite6BTag( byte nWriteLen) ;

public void onLock6BTag(byte nStatus) ;

public void onLockQuery6BTag( byte nStatus) ;

public void onExeCMDStatus(byte cmd, byte status);

public void onInventory6BTag( String strUID );

}

**The details about methods of the callback interface。**

note：there are some parameters do not work,they have a default value,it is necessary to focus on them in this version。

onInventoryTag

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onInventoryTag( String epc\_data, String tid, String rssi); | | |
| **description** | Get the result of making an inventory of cargo | | |
| **parameter** | **name** | **type** | **note** |
| epc | String | epc |
| tid | String | tid |
| rssi | String | rssi |

**onInventoryTagEnd**

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onInventoryTagEnd(); | | |
| **description** | The method is used when you finish counting | | |
| **parameters** | **name** | **type** | **note** |

**6B callback:**

#### **onRead6BTag**

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onRead6BTag( String strData); | | |
| **description** | The result callback of reading 6b-tag | | |
| **parameters** | name | type | note |
| strData | String |  |

#### **onWrite6BTag**

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onWrite6BTag( byte nWriteLen); | | |
| **description** | The result callback of writing 6b-tag | | |
| **parameters** | name | type | note |
| nWriteLen | byte |  |

#### **onLock6BTag**

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onLock6BTag( byte nStatus); | | |
| **description** | The result callback of locking 6b-tag | | |
| **parameters** | name | type | note |
| nStatus | byte | 0：success 1：fail |

#### **onLockQuery6BTag**

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onLockQuery6BTag( byte nStatus); | | |
| **description** | The result callback of querying 6b-tag | | |
| **parameters** | name | type | note |
| nStatus | byte | 0：locked；1：unlocked |

#### **onInventory6BTag**

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onInventory6BTag( String strUID ); | | |
| **description** | The result callback of inventorying 6b-tag | | |
| **parameters** | name | type | note |
| strUID | String |  |

#### **onExeCMDStatus**

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | void onExeCMDStatus(byte cmd, byte result); | | |
| **description** | The result callback of sending cmd | | |
| **parameters** | name | type | note |
| cmd | byte |  |
| result | byte |  |

1. **RFID api**

3.1 init【initialization】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void init(); | | |
| **description** | Initialize RFIDManager’s intance。 | | |
| **parameters** | **name** | **type** | **note** |
|  |  |  |
| **return** |  | | |
| **Reference Code** | None | | |

3.2 release【release resources】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void release(); | | |
| **description** | release resources | | |
| **parameters** | **name** | **type** | **note** |
|  |  |  |
| **return** |  | | |
| **Reference Code** |  | | |

3.3 sendCommand【send instructions】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int sendCommand(byte[] btCMDPackage); | | |
| **description** | send instructions | | |
| **parameters** | **name** | **type** | **note** |
| btCMDPackage | byte[] | instructions |
| **return**  **(int)** | Success：0,fail：-1； | | |
| **Reference Code** |  | | |

3.4 disConnect【disconnect rfid module】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void disConnect(); | | |
| **description** | disconnect rfid module。 | | |
| **parameters** | **name** | **type** | **note** |
|  |  |  |
| **return**  **(void)** |  | | |
| **Reference Code** | None | | |

3.5 isConnected【Determine the connection status of the module】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public boolean isConnected(); | | |
| **description** | Determine the connection status of the module | | |
| **parameters** | **name** | **type** | **note** |
|  |  |  |
| **return**  **(boolean)** | success：true ；fail：false ； | | |
| **Reference Code** |  | | |

3.6 cancelAccessEpcMatch【clear EPC match】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int cancelAccessEpcMatch(); | | |
| **description** | clear EPC match | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.7 clearTagMask【clear TAG mask setting】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int clearTagMask(); | | |
| **description** | clear TAG mask setting | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.8 startInventory【User define session and target inventory】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int customizedSessionTargetInventory( byte btSession); | | |
| **description** | User define session and target inventory | | |
| **parameters** | **name** | **type** | **note** |
| btSession | **byte** | **Desired session ID.** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.9 stopInventory【stop tag inventory】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int stopInventory(); | | |
| **description** | stop tag inventory | | |
| **parameters** | **name** | **type** | **note** |
|  |  |  |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.10 getAccessEpcMatch【Query match EPC status.】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getAccessEpcMatch(); | | |
| **description** | Query match EPC status. | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.11 getFrequencyRegion【Query frequency region.】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public RfidDate getFrequencyRegion(); | | |
| **description** | Query frequency region. | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(**RfidDate **)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.12 getFirmwareVersion【Get Reader Firmware Version.】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public String getFirmwareVersion(); | | |
| **description** | Get Reader Firmware Version. | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.13 getIdentifier【query identifier】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public byte[ ] getIdentifier(); | | |
| **description** | Query identifier。 | | |
| **parameters** | **name** | **type** | **note** |
|  |  |  |
| **return**  **(byte[ ])** | Reader Address | | |
| **Reference Code** |  | | |

3.14 getInventoryBuffer【Get tag data and keep buffer】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getInventoryBuffer(); | | |
| **description** | Get tag data and keep buffer | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.15 getInventoryBufferTagCount【Query tag quantity in buffer】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getInventoryBufferTagCount(); | | |
| **description** | Query tag quantity in buffer | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.16 getOutputPower【Query output power】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getOutputPower(); | | |
| **description** | Query output power。 | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.17 getReaderIdentifier【Get Reader Identifier】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getReaderIdentifier(); | | |
| **description** | Get Reader Identifier | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(byte)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.18 getReaderTemperature【Query internal temperature】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public RfidDate getReaderTemperature(); | | |
| **description** | Query internal temperature。 | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** |  | | |
| **Reference Code** |  | | |

3.19 iso180006BInventory【Inventory 18000-6B Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int iso180006BInventory(); | | |
| **description** | Inventory 18000-6B Tag | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.20 iso180006BLockTag【Lock 18000-6B Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int iso180006BLockTag(byte[] btAryUID,byte btWordAdd); | | |
| **description** | Lock 18000-6B Tag。 | | |
| **parameters** | **name** | **type** | **note** |
| btAryUID | **byte[]** | **Operated Tag's UID, 8 bytes** |
| btWordAdd | **byte** | **Locked address** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.21 iso180006BQueryLockTag【Query 18000-6B Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int iso180006BQueryLockTag(  byte[] btAryUID,  byte btWordAdd); | | |
| **description** | Query locked 18000-6B Tag。 | | |
| **parameters** | **name** | **type** | **note** |
| btAryUID | **byte[]** | **Operated Tag's UID, 8 bytes** |
| btWordAdd | **byte** | **Locked address** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.22 iso180006BReadTag【Read 18000-6B Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int iso180006BReadTag(  byte[] btAryUID,  byte btWordAdd,  byte btWordCnt); | | |
| **description** | Read 18000-6B Tag。 | | |
| **parameters** | **name** | **type** | **note** |
| btAryUID | **byte[]** | **Operated Tag's UID, 8 bytes** |
| btWordAdd | **byte** | **Read data first address** |
| btWordCnt | **byte** | **Read data length** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.23 iso180006BWriteTag【Write 18000-6B Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int iso180006BWriteTag(  byte[] btAryUID,  byte btWordAdd,  byte btWordCnt,  byte[] btAryBuffer); | | |
| **description** | Write 18000-6B Tag。 | | |
| **parameters** | **name** | **type** | **note** |
| btAryUID | **byte[]** | **Operated Tag's UID, 8 bytes** |
| btWordAdd | **byte** | **Write data first address** |
| btWordCnt | **byte** | **Write data length** |
| btAryBuffer | **byte[]** | **Write data** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.24 killTag【Kill Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int killTag(byte[] btAryPassWord); | | |
| **description** | Kill Tag | | |
| **parameters** | **name** | **type** | **note** |
| btAryPassWord | **byte[]** | **Kill password,4 bytes** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.25 lockTag【Lock Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int lockTag(  byte[] btAryPassWord,  byte btMemBank,  byte btLockType); | | |
| **description** | Lock Tag | | |
| **parameters** | **name** | **type** | **note** |
| btAryPassWord | **byte[]** | **Access password, 4 bytes.** |
| btMemBank | **byte** | **Tag memory bank(0x01:User Memory, 0x02:TID Memory, 0x03:EPC Memory, 0x04:Access Password, 0x05:Kill Password)** |
| btLockType | **byte** | **Lock operation type(0x00:Open, 0x01:Lock, 0x02:Permanent open, 0x03:Permanent lock)** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.26readTag【read tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int readTag(  byte btMemBank,  byte btWordAdd,  byte btWordCnt,  byte[] btAryPassWord); | | |
| **description** | Read Tag. Attention: If two tags have the same EPC, but different read data, then these two tags are considered different tags. | | |
| **parameters** | **name** | **type** | **note** |
| btReadId | **byte** | **Reader Address** |
| btAryPassWord | **byte[]** btWordCnt | **Access password,4 bytes.** |
| btMemBank | byte | **Tag memory bank(0x00:RESERVED, 0x01:EPC, 0x02:TID, 0x03:USER)** |
| btWordAdd | **Byte[** | **Read start address,please see the tag's spec for more information.** |
| btWordCnt | **byte** | **Read data length,Data length in WORD(16bits) unit. Please see the tag's spec for more information.** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.27 realTimeInventory【Inventory Tag(Read Time Mode)】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int realTimeInventory(byte btRepeat); | | |
| **description** | Inventory Tag(Read Time Mode). Attention: The hardware has a dual CPU architecture, main CPU is responsible for tag inventory, and assistant CPU is responsible for data management. Inventory and data transfer are parallel and simultaneous. So the data transfer via serial port doesn't affect the efficiency of reader. | | |
| **parameters** | **name** | **type** | **note** |
| btRepeat | **byte** | **Repeat time of inventory round. When Repeat = 255, The inventory duration is minimized. For example, if the RF field only has one or two tags, the inventory duration could be only 30-50 mS, this function provides a possibility for fast antenna switch applications on multi-ant devices.** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.28 rest【Reset the specified address reader.】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int reset(); | | |
| **description** | Reset the specified address reader. | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.29 setAccessEpcMatch【Set Access EPC match】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setAccessEpcMatch(  byte btEpcLen,  byte[] btAryEpc); | | |
| **description** | Set Access EPC match(EPC match is effective,until next refresh). | | |
| **parameters** | **name** | **type** | **note** |
| btEpcLen | byte | **Length of EPC** |
| btAryEpc | byte[] | **EPC, Length equals EpcLen.** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.30 setFrequencyRegion【Set frequency region】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setFrequencyRegion(  byte btRegion,  byte btStartRegion,  byte btEndRegion); | | |
| **description** | Set frequency region(system default frequencies). | | |
| **parameters** | **name** | **type** | **note** |
| btRegion | byte | **Spectrum regulation(0x01:FCC, 0x02:ETSI, 0x03:CHN).** |
| btStartRegion | byte | **Start frequency of the spectrum.** |
| btEndRegion | byte | **End frequency of the spectrum,Setup the range of the RF output spectrum. The rules are: 1,Start frequency and end frequency should be in the range of the specified regulation. 2,Start frequency should be equal or lower than end frequency. 3, End frequency equals start frequency means use single frequency point.** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.31 setOutputPower【Set output power(Method 1)】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setOutputPower(byte btOutputPower); | | |
| **description** | This command consumes more than 100mS. If you want you change the output power frequently, please use Cmd\_set\_temporary\_output\_power command, which doesn't reduce the life of the internal flash memory. | | |
| **parameters** | **name** | **type** | **note** |
| btReadId | **byte** | **Reader Address（公共地址是：0xFF）** |
| btOutputPower | byte | **RF output power:**  **New firmware power range: 10-30**  **Old firmware power range: 0-33**  **Short-Range Version Power Range: 0-26** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.32 setTagMask【Set the mask filter the Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setTagMask(  Int Membank,  Int StartAdd,  Int MaskLen,  String maskValue); | | |
| **description** | Set the mask filter the Tag | | |
| **parameters** | **name** | **type** | **note** |
| Membank | int | **The select mask region,EPC,TID or USER.** |
| StartAdd | int | **The mask start address.** |
| btMaskLen | int | **The mask length .** |
| maskValue | String | **The mask value.** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.33 setTrigger【Reader Trigger mode】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setTrigger(boolean enable); | | |
| **description** | Reader Trigger mode. | | |
| **parameters** | **name** | **type** | **note** |
| enable | boolean | **enable to operate Reader** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.34 writeTag【Write Tag】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int writeTag(String epc  byte[] btAryPassWord,  byte btMemBank,  byte btWordAdd,  byte btWordCnt,  byte[] btAryData); | | |
| **description** | Write Tag | | |
| **parameters** | **name** | **type** | **note** |
| **epc** | **String** | **epc** |
| btAryPassWord | byte[] | **Access password, 4 bytes.** |
| btMemBank | byte | **Tag memory bank(0x00:RESERVED, 0x01:EPC, 0x02:TID, 0x03:USER)** |
| btWordAdd | byte | **Write start address,WORD(16 bits). When write EPC area, notice that EPC starts from address 02, the first two 2 words are for PC+CRC.** |
| btWordCnt | byte | **WORD(16 bits), please see the tag's spec for more information.** |
| btAryData | byte[] | **Write data, btWordCnt\*2 bytes.** |
| **return**  **(int)** | success：0；fail：-1； | | |
| **Reference Code** |  | | |

3.35 registerCallback【register CallBack】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void registerCallback(com.ubx.usdk.rfid.aidl.IRfidCallback cb); | | |
| **description** | Unregister CallBack。 | | |
| **parameters** | **name** | **type** | **note** |
| cb | com.ubx.usdk.rfid.aidl.IRfidCallback | **Callback** |
| **return**  **(void)** |  | | |
| **Reference Code** |  | | |

3.36 unregisterCallback【Unregister callback】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void unregisterCallback(com.ubx.usdk.rfid.aidl.IRfidCallback cb); | | |
| **description** | Unregister callback | | |
| **parameters** | **name** | **type** | **note** |
| cb | com.ubx.usdk.rfid.aidl.IRfidCallback | **Callback** |
| **return**  **(void)** |  | | |
| **Reference Code** |  | | |

3.37 getModule【get module】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public java.lang.String getModule(); | | |
| **description** | get module | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(String)** | Module’s information | | |
| **Reference Code** |  | | |

3.38 getModuleFirmware【Get module firmware information】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public String getModuleFirmware(); | | |
| **description** | Get module firmware information | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(String)** | module firmware information | | |
| **Reference Code** |  | | |

**3.39 writeEpc【write epc number】**

|  |  |  |  |
| --- | --- | --- | --- |
| **define1** | public int WriteEPC(byte epclen,byte epc[],byte Password[]); | | |
| **description** | Randomly rewrite the EPC number of a label by broadcasting | | |
| **parameters** | **name** | **type** | **note** |
| epclen | byte | length of epc |
| epc | byte[] | Input, the EPC number of the label, the size is epclen\*2 |
| Password | byte[] | Input, the access code for the tag, 4 bytes |
| **return**  **(int)** | successreturn0。 | | |
| **Reference Code** |  | | |

**3.40 readOnceTag****【single tag query】**

|  |  |  |  |
| --- | --- | --- | --- |
| define | public String readTagOnce( byte AdrTID, byte LenTID) | | |
| **description** | This command is used to read the EPC or TID of the tag at a time | | |
| **parameters** | **name** | **type** | **note** |
| AdrTID | byte | Read the starting address of the TID |
| LenTID | byte | Read the TID length, if the length is 0, read the EPC number |
| **return**  **(String)** | Success:Label EPC or TID，fail:null | | |
| **Reference Code** | 。 | | |

**3.41 writeTagByTid【Write data to each storage area through tid】**

|  |  |  |  |
| --- | --- | --- | --- |
| **define3** | public int writeTagByTid(String TIDStr,byte Mem,  byte WordPtr,byte Password[],String wdata); | | |
| **description** | Write data to each memory area. | | |
| **parameters** | **name** | **type** | **note** |
| TIDStr | String | The tid number of the label, a hexadecimal string, the length must be an integer multiple of 4 |
| Mem | byte | memory area to be written,   1. 0- Password area, the first 3 words are the destruction password, the last 3 words are the access password 2. 1-EPC area 3. 2-TID area 4. User area |
| WordPtr | int | Write start word address |
| Password | String | The access password of the label, in hexadecimal string format。 |
|  | WData | String | The data to be written, a hexadecimal string, the length must be an integer multiple of 4 |
| **return**  **(int)** | Success:0,fail:-1 | | |
| **Reference Code** |  | | |

3.42 setScanInterval【Set the inventory interval】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setScanInterval( int interval) | | |
| **description** | Set the query parameters to be used when inventory is enabled | | |
| **parameters** | **name** | **type** | **note** |
| interval | int | Inventory interval |
| **return**  **(void)** |  | | |
| **Reference Code** |  | | |

3.43 getScanInterval【Read Inventory Interval】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getScanInterval() | | |
| **description** | Read the query used when inventory is enabled parameters | | |
| **parameters** | **name** | **type** | **note** |
|  |  |  |
| **return**  **(**int**)** | ReaderParameter | | |
| Reference **Code** |  | | |

3.44 setQueryMode【config query model】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void setQueryMode(int mode) | | |
| **description** | Config query model | | |
| **parameters** | **name** | **type** | **note** |
| Mode | Int、 | 0：epc  1：epc+tid  2：epc+fasttid |
| **return**  **(void)** |  | | |
| Reference **Code** | none | | |

3.45 getQueryMode【get query mode】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getQueryMode() | | |
| **description** | Get the query mode | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | 0：epc  1：epc+tid  2：epc+fasttid | | |
| Reference **Code** | none | | |

3.46 GetReaderType【get rfid mode】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int GetReaderType() | | |
| **description** | Get rfid mode | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | 80：the mode of short distance  others：the mode of long distance mode | | |
| Reference **Code** | none | | |

3.47 startRead【start inventory】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int startRead() | | |
| **description** | You can configure your own inventory parameters before calling the method | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(int)** | 0:success  -1:fail | | |
| Reference **Code** | none | | |

3.48 scanRfid【Take inventory over a period of time】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void scanRfid() | | |
| **description** | Take inventory over a period of time | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(void)** |  | | |
| Reference **Code** | none | | |

3.49 setInventoryParameter【configure inventory parameters】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void setInventoryParameter(Rfidparameter parameter) | | |
| **description** | configure inventory parameters | | |
| **parameters** | **name** | **type** | **note** |
| **return**  **(void)** |  | | |
| Reference **Code** | none | | |

3.50 getDeviceId【Gets the serial number】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public String getDeviceId() | | |
| **description** | Gets the serial number | | |
| **parameters** | **name** | **Type** | **note** |
| **return**  **(**String **)** | the serial number | | |
| Reference **Code** | none | | |

3.51 setCustomRegion【Set custom bands】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setCustomRegion(byte flags, int band, int FreSpace, int FreNum, int StartFre); | | |
| **description** | Set custom bands | | |
| **parameters** | **name** | **Type** | **note** |
|  | flags | byte | **Whether to save the setting information ID when powered**  **0：save**  **1：clear** |
| int | band | **Reserved fields**  **固定为：0xff** |
| int | FreSpace | **Band spacing** |
| int | FreNum | **Number of frequency points** |
| int | StartFre | **Starting band** |
| **return**  **(int)** | 0:success | | |
| Reference **Code** |  | | |

3.52 getCustomRegion【Get a custom band】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public CustomRegionBean getCustomRegion(); | | |
| **description** | Get a custom band information | | |
| **parameters** | **name** | **Type** | **note** |
| **return**  **(**CustomRegionBean **)** | Null:fail  public class CustomRegionBean {  public int [] band=new int[1];//  public int [] FreSpace=new int[1];//  public int [] FreNum=new int[1];//  public int [] StartFre=new int[1];//d  public CustomRegionBean(){}  } | | |
| Reference **Code** |  | | |

3.53 readDataByTid【Tag data is read via TID】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public String readDataByTid(String TIDStr, byte Mem, byte WordPtr, byte Num, String Password) | | |
| **description** | Tag data is read via TID | | |
| **parameters** | **name** | **Type** | **note** |
| TIDStr | String | **tid** |
| Mem | byte | **The region that needs to be read** |
| WordPtr | byte | **The start address of the data to be read** |
| Num | byte | **Length of data** |
| Password | String | **Access password** |
| **return**  **(String)** | Null:fail | | |
| Reference **Code** |  | | |

3.54 addMask【add mask】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void addMask(int mem,int startAddress,int len,String data) | | |
| **description** |  | | |
| **parameters** | **name** | **Type** | **note** |
| mem | int | **Filter area** |
| startAddress | int | **Start address** |
| len | int | **Length of data** |
| data | String | **Filter data** |
| **return**  **(void)** |  | | |
| Reference **Code** |  | | |

3.55 clearMask【clear mask】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public void clearMask(); | | |
| **description** | Clear so the mask | | |
| **parameters** | **name** | **Type** | **note** |
| **return**  **(**void**)** |  | | |
| Reference **Code** |  | | |

3.56 setProfile【Set the link configuration information】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setProfile(byte param); | | |
| **description** | Set the link configuration information | | |
| **parameters** | **name** | **Type** | **note** |
| param | byte | **Generally, the optional parameters of this parameter need to be provided according to the device information.**  **If the type of reader**  **is（0x21,0x23,0x28,0x36,0x37）**  **parm：**  **0:40K, FM0,25us**  **1:250K,M4, 25us**  **2:300K,M4, 25us**  **3:400K,FM0,6.25us**  **Other**  **11:640K,FM0,7.5u**  **1:640K, M2,7.5us**  **15:640K, M4,7.5us**  **12:320K, M2, 15us**  **3:320K, M2, 20us**  **5:320K, M4, 20us**  **7:250K, M4, 20us**  **13:160K, M8, 20us** |
| **Void** |  | | |
| Reference **Code** |  | | |

3.57 setRange【set the distance】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int setRange(int range); | | |
| **description** | Set the distance at which the label can be identified | | |
| **parameters** | **name** | **Type** | **note** |
| Int | range | **0~100** |
| **Int** | 0:success; | | |
| Reference **Code** |  | | |

3.58 getRange【get the distance】

|  |  |  |  |
| --- | --- | --- | --- |
| **define** | public int getRange(); | | |
| **description** | get the distance at which the label can be identified | | |
| **parameters** | **name** | **Type** | **note** |
|  |  |  |
| **Int** | -1:fail;  Other:the distance | | |
| Reference **Code** |  | | |