

固定式URAx UHF HOST SDK API说明 V1.0

Fixed URx UHF HOST SDK API Documentation V1.0

一、SDK概览

I.SDK Overview

二、详细API说明

II. Detail API Documentation

1 初始化连接

1 Initialize Connection

1.1 getInstance()

1.2 init(Context context)

1.3 getConnectStatus()

1.4 setConnectionStatusCallback(ConnectionStatusCallback btStatusCallback)

1.5 getUsbDeviceList(Context context)

1.6 setUart(String uart)

1.7 setIPAndPort(String ip, int port)

1.8 setUsbDevice(UsbDevice currDevice)

2 断开连接

2 Disconnect

3 获取UHF信息

3 Get UHF Info

3.1 getVersion()

3.2 getAndroidDeviceHardwareVersion()

3.3 getTemperature()

3.4 getUHFSERVICEVersion()

4 天线设置

4 Antenna Settings

4.1 getAntennaPower(AntennaEnum ant)

4.2 List<AntennaPowerEntity> getAntennaPower()

4.3 setAntennaPower(AntennaEnum ant, int power)

4.4 List<AntennaState> getANT()

4.5 setANT(List<AntennaState> antState)

4.6 List<AntennaConnectState> getAntennaConnectState()

5 频率设置

5 Frequency Settings

5.1 getFrequencyMode()

5.2 setFrequencyMode(int freMode)

6 标签盘点

6 Tag Inventory

6.1 startInventoryTag()

6.2 stopInventory()

6.3 inventorySingleTag()

6.4 setInventoryCallback(IUHFInventoryCallback)

6.5 setEPCMode()

6.6 setEPCAndTIDMode()

6.7 setEPCAndTIDUserMode(int,int)

6.8 setEPCAndTIDUserMode(InventoryModeEntity)

6.9 getEPCAndTIDUserMode()

6.10 setFilter(int,int,int,String)

6.11 isInventorying()

6.12 setFastInventoryMode(FastInventoryEntity)

6.13 getFastInventoryMode()

7 数据读写

7 Data Read/Write

7.1 readData(String,int,int,int)

7.2 readData(String,int,int,int,String,int,int,int)

7.3 writeData(String,int,int,int,String)

7.4 writeData(String,int,int,int,String,int,int,int,String)

7.5 writeDataToEpc(String,String)

7.6 writeDataToEpc(String,int,int,int,String,String)

7.7 blockWriteData(String,int,int,int,String,int,int,int,String)

7.8 eraseData(String,int,int,int)

7.9 eraseData(String,int,int,int,String,int,int,int)

7.10 uhfBlockPermalock(String,int,int,int,String,int, int,int,int,int,byte[])

8 标签销毁和锁定

8 Tag Kill、 Lock

8.1 killTag(String)

8.2 killTag(String,int,int,int,String)

8.3 lockMem(String,String)

8.4 lockMem(String,int,int,int,String,String)

8.5 generateLockCode(ArrayList,int)

9 RF链路、 FastID、 TagFocus、 GEN2

9 RFLink、 FastID、 TagFocus、 GEN2

9.1 setRFLink(int)

9.2 getRFLink()

9.3 setFastID(boolean)

9.4 getFastId()

9.5 setTagFocus(boolean)

9.6 getTagFocus()

9.7 setGen2(Gen2Entity)

9.8 getGen2()

10 蜂鸣器

10 Buzzer

10.1 enableBeep()

10.2 disableBeep()

10.3 isEnableBeep()

10.4 setBuzzerOn(int)

10.5 setBuzzerOff()

11 GPIO控制

11 GPIO Control

11.1 output1On()

11.2 output1Off()

11.3 output2On()

11.4 output2Off()

11.5 output3On()

11.6 output3Off()

11.7 output4On()

11.8 output4Off()

11.9 outputOnAndOff(List<GPOEntity>)

11.10 setGPISStateCallback(IGPISStateCallback)

11.11 setGPISStateReverse(bool)

12 以太网、WIFI

12 Ethernet、WIFI

12.1 getEthernetMac()

12.2 setEthernetConfigInfo(ReaderIPEntity)

12.3 getEthernetIpConfig()

12.4 setEthernetIpDynamicAssign()

12.5 openWifi()

12.6 closeWifi()

12.7 getWifiInfo()

12.7 setWifiConfigInfo(WifiConfig)

12.8 getWifiIpConfig()

12.9 setUHFSservicePort(int)

12.10 getReaderCurrentIp()

13 升级功能

13 Upgrade Functions

13.1 readyUpgradeUHFSservice()

13.2 setUpgradeUHFSserviceData(byte[])

13.3 startUpgradeUHFSservice()

13.4 setUpgradeProgress(IUpgradeProgress)

14 其他功能

14 Other Functions

14.1 factoryReset()

14.2 rebootDevice()

14.3 setCsn(String)

14.4 getCsn()

15 外部相关类

15 External related classes

15.1 InventoryParameter

15.2 InventoryParameter.ResultData

15.3 InventoryModeEntity

15.4 InventoryModeEntity.Builder

15.5 Gen2Entity

15.6 ErrorCode

15.7 FastInventoryEntity

15.8 AntennaPowerEntity

15.9 AntennaConnectState

15.10 AntennaState

15.11 GPIStateEntity

15.12 FilterEntity

15.13 ConnectionStatusCallback

15.14 ReaderIPEntity

15.15 GPOEntity

15.16 IGPIStateCallback

15.17 WifiConfig

15.18 IUpgradeProgress

16 多设备连接

16 Multi-device Connection

16.1 addDevice(String ip, int port)

16.2 getDevice(String ip)

16.3 removeDevice(String ip)

- 16.4 `removeAllDevice()`
- 16.5 `isDeviceExist(String address)`
- 16.6 `getDeviceCount()`
- 16.7 `connectAll()`
- 16.8 `disconnectAll()`
- 16.9 `setConnectionStatusCallbackAll(ConnectionStatusCallback btStatusCallback)`
- 16.10 `setInventoryCallbackAll(IUHFIInventoryCallback btStatusCallback)`

三、机型

III. Device

一、SDK 概览

I. SDK Overview

`com.rscja.deviceapi.RFIDwithUHFA4Host` 是基于 Android 平台的 UHF RFID 读写器主机端开发控制核心类，采用单例模式提供统一操作接口。该类支持三种连接方式：直接串口连接、TCP/IP 网络连接、USB 转串口连接（仅兼容搭载 PL2302 芯片的转接线）；可实现 RFID 标签盘点、读写、参数配置等核心功能。**使用此类接口的前置条件：RFID 读写器必须预装并启用 UHFService 服务。**若需同时连接多台 RFID 读写器，请参考 [多设备连接](#) 章节。

`com.rscja.deviceapi.RFIDwithUHFA4Host` is the core control class for host-side development of UHF RFID readers based on the Android platform, which adopts the singleton pattern to provide a unified operation interface. This class supports three connection methods: direct serial port connection, TCP/IP network connection, and USB-to-serial port connection (only compatible with adapter cables equipped with PL2302 chips); it can implement core functions such as RFID tag inventory, reading/writing, and parameter configuration. **The prerequisite for using the interfaces of this class is that the RFID reader must have the UHFService service pre-installed and enabled.** If you need to connect multiple RFID readers simultaneously, please refer to the [Multi-Device Connection](#) section.

1. 简单易用的单例设计

1. Simple and Easy-to-Use Singleton Design

您无需手动创建该类的实例，只需通过 `getInstance()` 方法即可获取全局唯一的管理对象，确保调用过程简单且资源高效。You do not need to manually create an instance of this class. You can obtain the globally unique management object only through the `getInstance()` method, ensuring that the calling process is simple and resource-efficient.

2. 使用步骤

2. Usage Steps

第一步: 确定连接方式并完成参数配置
Step 1: Determine the connection method and complete parameter configuration

- 若采用网口连接：调用 `setIPAndPort(String ip, int port)` 配置目标设备的 IP 地址和端口号；
- For Ethernet connection: Call `setIPAndPort(String ip, int port)` to configure the IP address and port number of the target device;
- 若采用直接串口连接：调用 `setUart(String uart)` 设定串口路径（如 `/dev/ttyS1`）；

- For direct serial port connection: Call `setUart(String uart)` to set the serial port path (e.g., `/dev/ttyS1`);
- 若采用 USB 转串口连接: 先通过 `getUsbDeviceList(Context context)` 获取可用 USB 设备列表, 再调用 `setUsbDevice(UsbDevice currDevice)` 选定目标设备。
- For USB-to-serial port connection: First obtain the list of available USB devices via `getUsbDeviceList(Context context)`, then call `setUsbDevice(UsbDevice currDevice)` to select the target device.

注意: 三种连接方式互斥, 不可同时启用。配置某一连接的参数时, 系统将自动断开当前已建立的其他连接

Note: The three connection methods are mutually exclusive and cannot be enabled simultaneously. When configuring parameters for one connection, the system will automatically disconnect any other existing connections.

第二步: 通过 `init(Context context)` 连接读写器。

Step 2: Connect the reader-writer via `init(Context context)`

第三步: 连接成功后, 根据需求执行以下操作:

Step 3: After a successful connection, perform the following operations as needed:

- 若需设置参数: 调用对应的函数进行参数配置及读写操作;
If parameter setting is required: Call the corresponding function for parameter configuration and read/write operations.
- 若需盘点标签:
If tag inventory is required:
先调用 `setInventoryCallback (IUHFInventoryCallback inventoryCallback)` 设置标签回调接口, 当有标签数据时会触发该接口回调, 再调用 `startInventoryTag()` 函数启动盘点。
First, call `setInventoryCallback(IUHFInventoryCallback inventoryCallback)` to set the tag callback interface, which will be triggered when there is tag data.
Then call the `startInventoryTag()` function to start the inventory.

注意: 在标签盘点过程中, RFID 模块仅会响应 `stopInventory()` 函数。

Note: During the tag inventory process, the RFID module will only respond to the `stopInventory()` function.

第四步: 退出app调用 `free()` 断开连接, 如果断开之前正在盘点, 请先停止盘点, 再断开连接。

Step 4: When exiting the app, call `free()` to disconnect. If an inventory operation is in process before disconnecting, please stop the inventory first and then disconnect.

3.使用示例

3.Usage Examples

```
public void Test(){
    RFIDWithUHFA4Host rfid=RFIDWithUHFA4Host.getInstance();
    //For Ethernet connection
    rfid.setUart("/dev/xxx");
    //For direct serial port connection:
    rfid.setIPAndPort("192.168.1.1", 8080);
    //For USB-to-serial port connection:
    List<UsbDevice> deviceList=rfid.getUsbDeviceList(context);
    rfid.setUsbDevice(deviceList[0]);
    //Note: The three connection methods are mutually exclusive and cannot be enabled simultaneously.
    // When configuring parameters for one connection,
    // the system will automatically disconnect any other existing connections.

    boolean result=rfid.init(Context context);
    if(!result){
        //connect fail
        return;
    }
    rfid.setConnectionStatusCallback(new ConnectionStatusCallback() {
        @Override
        public void getStatus(ConnectionStatus status, Object device) {
            if(status==ConnectionStatus.CONNECTED){

            }else if(status==ConnectionStatus.DISCONNECTED){
```

```
        } else if(status==ConnectionStatus.CONNECTING){

        }
    }
});
//connect success
//configuration parameters
//rfid.setFrequencyMode(xx)
//rfid.setAntennaPower(AntennaEnum.ANT1,30);
//rfid.setAntennaPower(AntennaEnum.ANT2,30);
rfid.setInventoryCallback(new IUHFInventoryCallback() {
    @Override
    public void callback(UHFTAGInfo uhftagInfo) {
        String epc= uhftagInfo.getEPC();
        String rssi= uhftagInfo.getRssi();
        //.....
    }
});
if(rfid.startInventoryTag()){
    //success
}else{
    //fail
}

rfid.stopInventory();
//.....
//disconnect uhf
rfid.free();
}
```

二、详细API说明

II.Detailed API description

1 初始化连接

1 Initialize Connection

1.1 getInstance()

项目 Item	说明 Description
原型 Prototype	static RFIDWithUHFA4Host getInstance() throws ConfigurationException
描述 Description	通过单例模式创建RFIDWithUHFA4Host对象 Create an RFIDWithUHFA4Host object via the singleton pattern
参数 Parameters	无 N/A
返回 Return	RFIDWithUHFA4Host 实例 RFIDWithUHFA4Host instance
备注 Remark	配置错误异常 Configuration err

1.2 init(Context context)

项目 Item	说明 Description
原型 Prototype	boolean init(Context context)
描述 Description	用于建立与UHF模块的连接。调用本函数前，需先通过 <code>setUart(String uart)</code> 、 <code>setUsbDevice(UsbDevice currDevice)</code> 或 <code>setIPAndPort(String ip, int port)</code> 完成对应连接方式的参数配置 Used to establish a connection with the UHF module. Before calling this function, you must first configure parameters for the corresponding connection method via <code>setUart(String uart)</code> , <code>setUsbDevice(UsbDevice currDevice)</code> or <code>setIPAndPort(String ip, int port)</code>
参数 Parameters	context: Android上下文 context: Android Context
返回 Return	true: 连接成功 true: connection successful false: 连接失败 false: connection failed

1.3 getConnectStatus()

项目 Item	说明 Description
原型 Prototype	ConnectionStatus getConnectStatus()
描述 Description	获取连接状态 Get connection status
参数 Parameters	无 N/A
返回 Return	<code>ConnectionStatus.DISCONNECTED</code> , <code>ConnectionStatus.CONNECTED</code> , <code>ConnectionStatus.CONNECTING</code>

1.4 setConnectionStatusCallback(ConnectionStatusCallback btStatusCallback)

项目 Item	说明 Description
原型 Prototype	void setConnectionStatusCallback(ConnectionStatusCallback btStatusCallback)
描述 Description	监听连接状态回调 Listen for connection status callback
参数 Parameters	btStatusCallback: 连接状态回调 Connection status callback
返回 Return	无 N/A

1.5 getUsbDeviceList(Context context)

项目 Item	说明 Description
原型 Prototype	List<UsbDevice> getUsbDeviceList(Context context)
描述 Description	获取 USB 设备列表 Obtain the list of USB devices
参数 Parameters	context: 上下文 Context
返回 Return	List<UsbDevice>: USB 设备列表 List of USB devices

1.6 setUart(String uart)

项目 Item	说明 Description
原型 Prototype	void setUart(String uart)
描述 Description	用于直接串口连接时，设定目标串口的路径 Used in direct serial port connection to set the path of the target serial port
参数 Parameters	uart: 串口路径（例如 <code>/dev/ttyS1</code> ） Serial port path (e.g., <code>/dev/ttyS1</code>)
返回 Return	无 N/A
备注 Remark	注意：三种连接方式互斥，不可同时启用。配置某一连接的参数时，系统将自动断开当前已建立的其他连接 Note: The three connection methods are mutually exclusive and cannot be enabled simultaneously. When configuring parameters for one connection, the system will automatically disconnect any other existing connections.

1.7 setIPAndPort(String ip, int port)

项目 Item	说明 Description
原型 Prototype	synchronized void setIPAndPort(String ip, int port)
描述 Description	用于网口连接时，配置目标设备的IP地址和端口号 Used in Ethernet connection to configure the IP address and port number of the target device
参数 Parameters	ip: 目标设备的IP地址 IP address of the target device port: 目标设备的端口号 Port number of the target device
返回 Return	无 N/A

项目 Item	说明 Description
备注 Remark	注意：三种连接方式互斥，不可同时启用。配置某一连接的参数时，系统将自动断开当前已建立的其他连接 Note: The three connection methods are mutually exclusive and cannot be enabled simultaneously. When configuring parameters for one connection, the system will automatically disconnect any other existing connections.

1.8 setUsbDevice(UsbDevice currDevice)

项目 Item	说明 Description
原型 Prototype	synchronized void setUsbDevice(UsbDevice currDevice)
描述 Description	用于USB转串口连接时，指定要连接的目标USB设备 Used in USB-to-serial port connection to specify the target USB device to connect
参数 Parameters	currDevice：目标USB设备（需先通过 <code>getUsbDeviceList(Context context)</code> 获取可用设备列表后选择） Target USB device (selected from the list of available devices obtained via <code>getUsbDeviceList(Context context)</code> first)
返回 Return	无 N/A
备注 Remark	注意：三种连接方式互斥，不可同时启用。配置某一连接的参数时，系统将自动断开当前已建立的其他连接 Note: The three connection methods are mutually exclusive and cannot be enabled simultaneously. When configuring parameters for one connection, the system will automatically disconnect any other existing connections.

2 断开连接

2 Disconnect

项目 Item	说明 Description
原型 Prototype	boolean free()
描述 Description	关闭UHF模块 Switch off UHF module
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

3 获取UHF信息

3 Get UHF Info

3.1 getVersion()

项目 Item	说明 Description
原型 Prototype	String getVersion()

项目 Item	说明 Description
描述 Description	获取UHF模块版本 Get UHF module version
参数 Parameters	无 N/A
返回 Return	UHF模块版本 UHF module version

3.2 getAndroidDeviceHardwareVersion()

项目 Item	说明 Description
原型 Prototype	String getAndroidDeviceHardwareVersion()
描述 Description	获取主板版本号 Get the mainboard version
参数 Parameters	无 N/A
返回 Return	获取主板版本号 Get the mainboard version

3.3 getTemperature()

项目 Item	说明 Description
原型 Prototype	synchronized int getTemperature()
描述 Description	获取UHF模块温度 Get UHF module temperature
参数 Parameters	无 N/A
返回 Return	温度,单位: 摄氏度 Temperature,Unit:°C

3.4 getUHFSERVICEVersion()

项目 Item	说明 Description
原型 Prototype	String getUHFSERVICEVersion()
描述 Description	获取设备UHF服务的版本信息 Obtain the version information of the device's UHF service
参数 Parameters	无 N/A

项目 Item	说明 Description
返回 Return	字符串类型：UHF服务版本信息字符串（格式通常为X.X.X，返回null/空字符串表示获取版本信息失败） String type: UHF service version information string (usually in the format of X.X.X returning null/empty string means failure to obtain version information)

4 天线设置

4 Antenna Settings

4.1 getAntennaPower(AntennaEnum ant)

项目 Item	说明 Description
原型 Prototype	int getAntennaPower(AntennaEnum ant)
描述 Description	获取天线功率 Get the antenna power
参数 Parameters	ant: 天线号 ant: Antenna Number
返回 Return	功率值 Power value

4.2 List<AntennaPowerEntity> getAntennaPower()

项目 Item	说明 Description
原型 Prototype	List<AntennaPowerEntity> getAntennaPower()
描述 Description	获取所有天线的功率 Get all antenna power
参数 Parameters	无 N/A
返回 Return	每个天线对应的功率值 The power value corresponding to each antenna

4.3 setAntennaPower(AntennaEnum ant, int power)

项目 Item	说明 Description
原型 Prototype	boolean setAntennaPower(AntennaEnum ant, int power)
描述 Description	设置天线功率 Set the antenna power
参数 Parameters	ant: 天线号 ant: Antenna Number
参数 Parameters	power: 功率值(1-30) power: Power value(1-30)

项目 Item	说明 Description
返回 Return	true:成功 false:失败 true:success false:failure

4.4 List<AntennaState> getANT()

项目 Item	说明 Description
原型 Prototype	List<AntennaState> getANT()
描述 Description	获取所有天线的启用状态 Get the enable state of all antennas
参数 Parameters	无 N/A
返回 Return	每个天线的启用状态 The enable state of each antenna

4.5 setANT(List<AntennaState> antState)

项目 Item	说明 Description
原型 Prototype	boolean setANT(List<AntennaState> antState)
描述 Description	设置所有天线的启用状态 Set the enable state of all antennas
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

4.6 List<AntennaConnectState> getAntennaConnectState()

项目 Item	说明 Description
原型 Prototype	List<AntennaConnectState> getAntennaConnectState()
描述 Description	获取所有天线的连接状态 Get the connection State of all antennas
参数 Parameters	无 N/A
返回 Return	每个天线的连接状态 The connection status of each antenna

5 频率设置

5 Frequency Settings

5.1 getFrequencyMode()

项目 Item	说明 Description
原型 Prototype	int getFrequencyMode()
描述 Description	获取UHF模块频率模式 Get UHF module frequency mode
参数 Parameters	无 N/A
返回 Return	频率模式 Frequency mode 0x01：China Standard(840~845MHz) 0x02：China Standard2(920~925MHz) 0x04：Europe Standard(865~868MHz) 0x08：USA(902-928MHz) 0x16：Korea(917~923MHz) 0x32: Japan(916.8~920.8MHz) 0x33: South Africa(915~919MHz) 0x34: China Taiwan(920~928Mhz) 0x35:Vietnam(918~923MHz) 0x36:Peru(915-928MHz) 0x37:Russia(860-867.6MHz) 0x3B:Malaysia(919-923MHz) 0x3C:Brazil(902 ~ 907.5MHz, 915 ~ 928MHz) 0x3D:ETSI_UPPER(915 ~ 921MHz) 0x3E:Australia(920 ~ 926MHz) 0x3F:Indonesia(920-923MHz) 0x40:Israel(915 ~ 917MHz) 0x41:HK(920 ~ 925MHz) 0x42:New Zealand(920 ~ 928MHz)

5.2 setFrequencyMode(int freMode)

项目 Item	说明 Description
原型 Prototype	boolean setFrequencyMode(int freMode)
描述 Description	设置UHF模块频率模式 Set UHF module frequency mode

项目 Item	说明 Description
参数 Parameters	freMode: 频率模式 freMode: Frequency mode 0x01: China Standard(840~845MHz) 0x02: China Standard2(920~925MHz) 0x04: Europe Standard(865~868MHz) 0x08: USA(902-928MHz) 0x16: Korea(917~923MHz) 0x32: Japan(916.8~920.8MHz) 0x33: South Africa(915~919MHz) 0x34: China Taiwan(920~928Mhz) 0x35:Vietnam(918~923MHz) 0x36:Peru(915-928MHz) 0x37:Russia(860-867.6MHz) 0x3B:Malaysia(919-923MHz) 0x3C:Brazil(902 ~ 907.5MHz, 915 ~ 928MHz) 0x3D:ETSI_UPPER(915 ~ 921MHz) 0x3E:Australia(920 ~ 926MHz) 0x3F:Indonesia(920-923MHz) 0x40:Israel(915 ~ 917MHz) 0x41:HK(920 ~ 925MHz) 0x42:New Zealand(920 ~ 928MHz)
返回 Return	true:成功 false:失败 true:succcess false:failure

6 标签盘点

6 Tag Inventory

6.1 startInventoryTag()

项目 Item	说明 Description
原型 Prototype	boolean startInventoryTag()
描述 Description	开始循环识别标签,在盘点标之前先设置setInventoryCallback(IUHFInventoryCallback)回调接口接收标签数据。开始盘点后UHF模块只响应stopInventory()函数。 Start cyclically identifying tags. Before inventory counting, set the setInventoryCallback(IUHFInventoryCallback) callback interface first to receive tag data. After starting the inventory, the UHF module will only respond to the stopInventory() function.
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:succcess false:failure

6.2 stopInventory()

项目 Item	说明 Description
原型 Prototype	boolean stopInventory()

项目 Item	说明 Description
描述 Description	停止盘点 Stop inventory
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

6.3 inventorySingleTag()

项目 Item	说明 Description
原型 Prototype	UHFTAGInfo inventorySingleTag()
描述 Description	单标签识别 Single tag identification
参数 Parameters	无 N/A
返回 Return	标签信息 Tag information

6.4 setInventoryCallback(IUHFIInventoryCallback)

项目 Item	说明 Description
原型 Prototype	void setInventoryCallback(IUHFIInventoryCallback inventoryCallback)
描述 Description	设置盘点回调接口,在开始盘点之调用 Set the inventory callback interface,Set the inventory callback interface and call it before starting the inventory.
参数 Parameters	inventoryCallback: 盘点回调接口 inventoryCallback: Inventory callback interface
返回 Return	void

6.5 setEPCMode()

项目 Item	说明 Description
原型 Prototype	boolean setEPCMode()
描述 Description	设置为EPC模式，盘点时仅输出标签的EPC数据 Set to EPC mode, only output EPC data of tags when inventorying
参数 Parameters	无 N/A

项目 Item	说明 Description
返回 Return	true:成功 false:失败 true:success false:failure

6.6 setEPCAndTIDMode()

项目 Item	说明 Description
原型 Prototype	boolean setEPCAndTIDMode()
描述 Description	设置为EPC+TID模式，盘点时输出标签的EPC和TID数据 Set to EPC+TID mode, output EPC and TID data of tags when inventorying
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

6.7 setEPCAndTIDUserMode(int,int)

项目 Item	说明 Description
原型 Prototype	boolean setEPCAndTIDUserMode(int user_prt, int user_len)
描述 Description	设置为EPC+TID+USER模式，盘点时输出EPC、TID、USER数据 Set to EPC+TID+USER mode, output EPC, TID, USER data of tags when inventorying
参数 Parameters	user_prt: 用户数据起始地址 user_prt: User data start address user_len: 用户数据长度 user_len: User data length
返回 Return	true:成功 false:失败 true:success false:failure

6.8 setEPCAndTIDUserMode(InventoryModeEntity)

项目 Item	说明 Description
原型 Prototype	boolean setEPCAndTIDUserMode(InventoryModeEntity entity)
描述 Description	设置盘点模式实体（如输出字段等） Set the inventory mode via entity (e.g., output fields)
参数 Parameters	entity: 盘点模式 entity: Inventory mode
返回 Return	true:成功 false:失败 true:success false:failure

项目 Item	说明 Description
备注 Remark	比如：设置盘点EPC+Reserved 数据模式 For example: Set up the EPC+Reserved data mode for inventory <pre>InventoryModeEntity inventoryModeEntity=new InventoryModeEntity.Builder() .setMode(InventoryModeEntity.MODE_EPC_RESERVED) .setReservedLength(0) .setReservedOffset(4).build(); rfidWithUHFUART.setEPCAndTIDUserMode(inventoryModeEntity);</pre>

6.9 getEPCAndTIDUserMode()

项目 Item	说明 Description
原型 Prototype	<code>InventoryModeEntity</code> getEPCAndTIDUserMode()
描述 Description	获取当前盘点模式实体（如当前输出字段配置） Get the current inventory mode entity (e.g., current output fields configuration)
参数 Parameters	无 N/A
返回 Return	盘点模式实体对象 Inventory mode entity object

6.10 setFilter(int,int,int,String)

项目 Item	说明 Description
原型 Prototype	<code>boolean</code> setFilter(int bank, int offset, int length, String data)
描述 Description	设置盘点过滤条件 Set the inventory count filtering conditions
参数 Parameters	bank: 过滤内存 (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) bank: Filter memory (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) offset: 偏移量 offset: Offset length: 长度 length: Length data: 数据 data: Data
返回 Return	true:成功 false:失败 true:success false:failure

6.11 isInventorying()

项目 Item	说明 Description
原型 Prototype	<code>boolean</code> isInventorying()

项目 Item	说明 Description
描述 Description	判断设备是否正在盘点 Judge the device is inventorying or not
参数 Parameters	无 N/A
返回 Return	true:正在盘点 false:未盘点 true:inventorying false:not inventorying

6.12 setFastInventoryMode(FastInventoryEntity entity)

项目 Item	说明 Description
原型 Prototype	boolean setFastInventoryMode(FastInventoryEntity entity)
描述 Description	设置快速盘点模式 Set fast inventory mode
参数 Parameters	entity: 快速盘点实体对象 entity: Fast inventory entity object
返回 Return	true:成功 false:失败 true:success false:failure

6.13 getFastInventoryMode()

项目 Item	说明 Description
原型 Prototype	FastInventoryEntity getFastInventoryMode()
描述 Description	获取快速盘点模式 Get fast inventory mode
参数 Parameters	无 N/A
返回 Return	快速盘点实体对象 FastInventoryEntity

7 数据读写

7 Data Read/Write

7.1 readData(String,int,int,int)

项目 Item	说明 Description
原型 Prototype	String readData(String accessPwd, int bank, int offset, int length)

项目 Item	说明 Description
描述 Description	读取数据 Read data
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password bank: 内存 (IUHF.BANK_RESERVED/EPC/TID/USER) bank: Memory (IUHF.BANK_RESERVED/EPC/TID/USER) offset: 偏移量 offset: Offset length: 长度 length: Length
返回 Return	数据 Data

7.2 readData(String,int,int,int,String,int,int,int)

项目 Item	说明 Description
原型 Prototype	String readData(String accessPwd, int filterBank, int filterOffset, int filterLength, String filterData, int bank, int offset, int length)
描述 Description	读取数据（带过滤条件） Read data with filter condition
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password filterBank: 过滤内存 (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) filterBank: Filter bank (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) filterOffset: 过滤偏移量 filterOffset:Filter offset filterLength: 过滤长度 filterLength:Filter length filterData: 过滤数据 filterData:Filter data bank: 目标内存 (IUHF.BANK_RESERVED/EPC/TID/USER) bank: Memory (IUHF.BANK_RESERVED/EPC/TID/USER) offset: 偏移量 offset: Offset length: 长度 length: Length
返回 Return	数据 Data

7.3 writeData(String,int,int,int,String)

项目 Item	说明 Description
原型 Prototype	boolean writeData(String accessPwd, int bank, int offset, int length, String data)
描述 Description	写入数据 Write data

项目 Item	说明 Description
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password bank: 内存 bank: Memory offset: 偏移量 offset: Offset length: 长度 length: Length data: 数据 data: Data
返回 Return	true:成功 false:失败 true:success false:failure

7.4 writeData(String,int,int,int,String,int,int,int,String)

项目 Item	说明 Description
原型 Prototype	boolean writeData(String accessPwd, int filterBank, int filterOffset, int filterLength, String filterData, int bank, int offset, int length, String writeData)
描述 Description	写入数据（带过滤条件） Write data with filter condition
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password filterBank: 过滤内存（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterBank: Filter Bank（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterOffset: 过滤偏移量 filterOffset:Filter offset filterLength: 过滤长度 filterLength:Filter length filterData: 过滤数据 filterData:Filter data bank: 目标内存（IUHF.BANK_RESERVED/EPC/TID/USER） bank: Memory（IUHF.BANK_RESERVED/EPC/TID/USER） offset: 偏移量 offset:Offset length: 长度 length:Length writeData: 写入数据 writeData:Data
返回 Return	true:成功 false:失败 true:success false:failure

7.5 writeDataToEpc(String,String)

项目 Item	说明 Description
原型 Prototype	boolean writeDataToEpc(String accessPwd, String writeData)

项目 Item	说明 Description
描述 Description	写入数据到EPC Write data to the EPC
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password writeData: 写入数据 writeData: Write data
返回 Return	true:成功 false:失败 true:success false:failure

7.6 writeDataToEpc(String,int,int,int,String,String)

项目 Item	说明 Description
原型 Prototype	boolean writeDataToEpc(String accessPwd, int filterBank, int filterOffset, int filterLength, String filterData, String writeData)
描述 Description	写入数据到EPC（带过滤条件） Write data to the EPC with filter condition
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password filterBank: 过滤内存（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterBank: Filter Bank（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterOffset: 过滤偏移量 filterOffset:Filter offset filterLength: 过滤长度 filterLength:Filter length filterData: 过滤数据 filterData:Filter data offset: 偏移量 offset:Offset length: 长度 length:Length writeData: 写入数据 writeData:Data
返回 Return	true:成功 false:失败 true:success false:failure

7.7 blockWriteData(String,int,int,int,String,int,int,int,String)

项目 Item	说明 Description
原型 Prototype	boolean blockWriteData(String accessPwd, int filterBank, int filterOffset, int filterLength, String filterData, int bank, int offset, int length, String writeData)
描述 Description	块写入数据（带过滤条件） Block write data with filter condition

项目 Item	说明 Description
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password filterBank: 过滤内存 (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) filterBank: Filter Bank (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) filterOffset: 过滤偏移量 filterOffset:Filter offset filterLength: 过滤长度 filterLength:Filter length filterData: 过滤数据 filterData:Filter data bank: 目标内存 (IUHF.BANK_RESERVED/EPC/TID/USER) bank: Memory (IUHF.BANK_RESERVED/EPC/TID/USER) offset: 偏移量 offset:Offset length: 长度 length:Length writeData: 写入数据 writeData:Data

7.8 eraseData(String,int,int,int)

项目 Item	说明 Description
原型 Prototype	boolean eraseData(String accessPwd, int bank, int offset, int length)
描述 Description	擦除数据 Erase data
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password bank: 内存 bank: Memory offset: 偏移量 offset: Offset length: 长度 length: Length
返回 Return	true:成功 false:失败 true:success false:failure

7.9 eraseData(String,int,int,int,String,int,int,int)

项目 Item	说明 Description
原型 Prototype	boolean eraseData(String accessPwd, int filterBank, int filterOffset, int filterLength, String filterData, int bank, int offset, int length)
描述 Description	擦除数据 (带过滤条件) Erase data with filter condition

项目 Item	说明 Description
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password filterBank: 过滤内存 (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) filterBank: Filter Bank (IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER) filterOffset: 过滤偏移量 filterOffset:Filter offset filterLength: 过滤长度 filterLength:Filter length filterData: 过滤数据 filterData:Filter data bank: 目标内存 (IUHF.BANK_RESERVED/EPC/TID/USER) bank: Memory (IUHF.BANK_RESERVED/EPC/TID/USER) offset: 偏移量 offset:Offset length: 长度
返回 Return	true:成功 false:失败 true:success false:failure

7.10 uhfBlockPermalock(String,int,int,int,String, int, int,int,int,byte[])

项目 Item	说明 Description
原型 Prototype	boolean uhfBlockPermalock(String accessPwd, int FilterBank, int FilterStartaddr, int FilterLen, String FilterData, int ReadLock, int uBank, int uPtr, int uRange, byte[] uMaskbuf)
描述 Description	数据块操作 Data block operation
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password FilterBank: 过滤的存储区 FilterBank: Filtered storage area: IUHF.Bank_EPC、IUHF.Bank_TID、IUHF.Bank_USER FilterStartaddr: 过滤的起始地址 FilterStartaddr: Filter start address FilterLen: 过滤数据长度 FilterLen: Filter data length FilterData: 过滤的数据 FilterData: Filtered data ReadLock: 读写锁 ReadLock: Read write lock uBank: 存储区域 uBank: Storage area: IUHF.Bank_RESERVED、IUHF.Bank_EPC、IUHF.Bank_TID、IUHF.Bank_USER uPtr: 存储区域起始地址，默认为0，其中1个块单元为16个块 uPtr: Start address in storage area, default is 0, 1 unit has 16 blocks uRange: 存储区域结束范围，默认为1，其中1个块单元为16个块 uRange: End of storage, default is 1, 1 unit has 16 blocks uMaskbuf: 设置锁MSAK，高位在前，每个bit代表为1个块，默认为2字节 uMaskbuf: Setup lock MSAK, high level is at front, every bit has 1 block, default is 2 bytes.
返回 Return	true:成功 false:失败 true:success false:failure

8 标签销毁和锁定

8 Tag Kill、Lock

8.1 killTag(String)

项目 Item	说明 Description
原型 Prototype	boolean killTag(String killPwd)
描述 Description	杀死标签 Kill tag
参数 Parameters	killPwd: 杀死密码 killPwd: Kill password
返回 Return	true:成功 false:失败 true:success false:failure

8.2 killTag(String,int,int,int,String)

项目 Item	说明 Description
原型 Prototype	boolean killTag(String killPwd, int filterBank, int filterOffset, int filterLength, String filterData)
描述 Description	杀死标签（带过滤条件） Kill tag with filter condition
参数 Parameters	killPwd: 杀死密码 killPwd: Kill password filterBank: 过滤内存（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterBank: Filter bank（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterOffset: 过滤偏移量 filterOffset:Filter offset filterLength: 过滤长度 filterLength:Filter length filterData: 过滤数据 filterData:Filter data
返回 Return	true:成功 false:失败 true:success false:failure

8.3 lockMem(String,String)

项目 Item	说明 Description
原型 Prototype	boolean lockMem(String accessPwd, lockCode)
描述 Description	锁定标签 Lock tag
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password lockCode: 锁码 lockCode: Lock code

项目 Item	说明 Description
返回 Return	true:成功 false:失败 true:success false:failure
备注 Remark	lockCode通过 generateLockCode() 函数获取 lockCode is obtained through the generateLockCode() function

8.4 lockMem(String,int,int,int,String,String)

项目 Item	说明 Description
原型 Prototype	boolean lockMem(String accessPwd, int filterBank, int filterOffset, int filterLength, String filterData, String lockCode)
描述 Description	锁定标签（带过滤与锁码） Lock tag with filter condition and lock code
参数 Parameters	accessPwd: 访问密码 accessPwd: Access password filterBank: 过滤内存（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterBank: Filter bank（IUHF.BANK_EPC/IUHF.BANK_TID/IUHF.BANK_USER） filterOffset: 过滤偏移量 filterOffset:Filter offset filterLength: 过滤长度 filterLength:Filter length filterData: 过滤数据 filterData:Filter data lockCode: 锁码 lockCode:Lock code
返回 Return	true:成功 false:失败 true:success false:failure
备注 Remark	lockCode通过 generateLockCode() 函数获取 lockCode is obtained through the generateLockCode() function

8.5 generateLockCode(ArrayList,int)

项目 Item	说明 Description
原型 Prototype	String generateLockCode(ArrayList lockBank, int lockMode)
描述 Description	生成锁码 Generate lock code
参数 Parameters	lockBank: IUHF.LockBank_ACCESS IUHF.LockBank_KILL IUHF.LockBank_EPC IUHF.LockBank_USER lockMode: IUHF.LockMode_LOCK IUHF.LockMode_OPEN IUHF.LockMode_PLOCK-永久锁定(Permanent LOCK) IUHF.LockMode_POPEN-永久开放(Permanent release)

项目 Item	说明 Description
返回 Return	锁码 Lock code
备注 Remark	比如:同时锁EPC和 USER For example: Lock both the EPC bank and the USER bank at the same time ArrayList lockBank= new ArrayList(); lockBank.add(IUHF.LockBank_EPC); lockBank.add(IUHF.LockBank_USER); String lockCode=rfidWithUHFUART.generateLockCode(lockBank,IUHF.LockMode_LOCK);

9 RF链路、FastID、TagFocus

9 RFLink、FastID、TagFocus

9.1 setRFLink(int)

项目 Item	说明 Description
原型 Prototype	boolean setRFLink(int mode)
描述 Description	设置RFLink Set RFLink
参数 Parameters	mode: 模式 mode: Mode 0-PR ASK/Miller8/160KHz 1-PR ASK/Miller4/250KHz 2-PR ASK/Miller4/320KHz 3-PR ASK/Miller4/640KHz 4-PR ASK/Miller2/320KHz 5-PR ASK/Miller2/640KHz 10-Gen2X/Miller8/160KHz 11-Gen2X/Miller4/250KHz 12-Gen2X/Miller4/320KHz 13-Gen2x/Miller4/640KHz 14-Gen2X/Miller2/320KHz 15-Gen2X/Miller2/640KHz 20-Autoset/Miller8/160KHz 21-Autoset/Miller4/250KHz 22-Autoset/Miller4/320KHz 23-Autoset/Miller4/640KHz 24-Autoset/Miller2/320KHz 25-Autoget/Miller2/640KHz
返回 Return	true:成功 false:失败 true:success false:failure
备注 Remark	Gen2X的链路只支持英频杰M830和M850等最新的标签。 The links of Gen2X only support the latest tags such as Impinj M830 and M850.

9.2 getRFLink()

项目 Item	说明 Description
原型 Prototype	synchronized int getRFLink()
描述 Description	获取RFLink模式 Get RFLink mode
参数 Parameters	无 N/A
返回 Return	-1:失败(failure) 0-PR ASK/Miller8/160KHz 1-PR ASK/Miller4/250KHz 2-PR ASK/Miller4/320KHz 3-PR ASK/Miller4/640KHz 4-PR ASK/Miller2/320KHz 5-PR ASK/Miller2/640KHz 10-Gen2X/Miller8/160KHz 11-Gen2X/Miller4/250KHz 12-Gen2X/Miller4/320KHz 13-Gen2x/Miller4/640KHz 14-Gen2X/Miller2/320KHz 15-Gen2X/Miller2/640KHz 20-Autoset/Miller8/160KHz 21-Autoset/Miller4/250KHz 22-Autoset/Miller4/320KHz 23-Autoset/Miller4/640KHz 24-Autoset/Miller2/320KHz 25-Autoget/Miller2/640KHz

9.3 setFastID(boolean)

项目 Item	说明 Description
原型 Prototype	synchronized boolean setFastID(boolean enalbe)
描述 Description	设置FastID Set Fast ID
参数 Parameters	enalbe: 是否启用 enalbe: Enable
返回 Return	true:成功 false:失败 true:success false:failure

9.4 getFastId()

项目 Item	说明 Description
原型 Prototype	synchronized int getFastId()

项目 Item	说明 Description
描述 Description	获取FastId Get FastId
参数 Parameters	无 N/A
返回 Return	0:关闭 1:打开 -1:失败 0:Close 1:Open -1:failure

9.5 setTagFocus(boolean)

项目 Item	说明 Description
原型 Prototype	synchronized boolean setTagFocus(boolean enalbe)
描述 Description	设置TagFocus Set Tag Focus
参数 Parameters	enalbe: 是否启用 enalbe: Enable
返回 Return	true:成功 false:失败 true:succcess false:failure

9.6 getTagFocus()

项目 Item	说明 Description
原型 Prototype	synchronized int getTagFocus()
描述 Description	获取TagFocus Get TagFocus
参数 Parameters	无 N/A
返回 Return	0:关闭 1:打开 -1:失败 0:Close 1:Open -1:failure

9.7 setGen2(Gen2Entity)

项目 Item	说明 Description
原型 Prototype	boolean setGen2(Gen2Entity gen2Entity)
描述 Description	设置GEN2参数值 Set GEN2 parameter value
参数 Parameters	gen2Entity: GEN2参数实体对象 gen2Entity: GEN2 parameter entity object

项目 Item	说明 Description
返回 Return	true:成功 false:失败 true:success false:failure
备注 Remark	设置之前先调用getGen2()获取Gen2Entity对象，在修改对应参数。目前只支持Gen2Entity.setQuerySession(int)和Gen2Entity.setQueryTarget(int). Call getGen2() first to obtain the Gen2Entity object before setting, and then modify the corresponding parameters. Currently, only Gen2Entity.setQuerySession(int) and Gen2Entity.setQueryTarget(int) are supported.

9.8 getGen2()

项目 Item	说明 Description
原型 Prototype	Gen2Entity getGen2()
描述 Description	获取GEN2参数值 Get GEN2 parameter value
参数 Parameters	无 N/A
返回 Return	GEN2参数值实体对象 GEN2 parameter value entity object

10 蜂鸣器

10 Buzzer

10.1 enableBeep()

项目 Item	说明 Description
原型 Prototype	boolean enableBeep()
描述 Description	启用蜂鸣器，启用后盘点到标签时设备会发出提示音 Enable the buzzer. After enabling, the device will emit a prompt tone when a tag is inventoried
参数 Parameters	无 N/A
返回 Return	boolean: true表示启用成功，false表示启用失败 boolean: true for successful enabling, false for failure

10.2 disableBeep()

项目 Item	说明 Description
原型 Prototype	boolean disableBeep()

项目 Item	说明 Description
描述 Description	禁用蜂鸣器，禁用后盘点到标签时设备不会发出提示音 Disable the buzzer. After disabling, the device will not emit a prompt tone when a tag is inventoried
参数 Parameters	无 N/A
返回 Return	boolean: true表示禁用成功，false表示禁用失败 boolean: true for successful disabling, false for failure

10.3 isEnableBeep()

项目 Item	说明 Description
原型 Prototype	boolean isEnableBeep()
描述 Description	判断蜂鸣器是否启用 Check whether the buzzer is enabled
参数 Parameters	无 N/A
返回 Return	boolean: 启用返回true，禁用返回false true: enabled, false: disabled

10.4 setBuzzerOn(int time)

项目 Item	说明 Description
原型 Prototype	void setBuzzerOn(int time)
描述 Description	触发蜂鸣器鸣叫（立即生效） Trigger buzzer to sound (takes effect immediately)
参数 Parameters	time: 蜂鸣器持续鸣叫时长（单位：毫秒） time: Buzzer sounding duration (Unit: millisecond)
返回 Return	无 N/A

10.5 setBuzzerOff()

项目 Item	说明 Description
原型 Prototype	boolean setBuzzerOff()
描述 Description	停止蜂鸣器鸣叫（立即生效），与setBuzzerOn方法功能对应 Stop buzzer sounding (takes effect immediately), corresponding to the setBuzzerOn method
参数 Parameters	无 N/A

项目 Item	说明 Description
返回 Return	boolean: true表示操作成功, false表示操作失败 boolean: true for operation succeeded, false for operation failed

11 GPIO控制

11 GPIO Control

11.1 output1On()

项目 Item	说明 Description
原型 Prototype	boolean output1On()
描述 Description	使能GPO1通用输出口 Enable GPO1 general-purpose output port
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.2 output1Off()

项目 Item	说明 Description
原型 Prototype	boolean output1Off()
描述 Description	禁用GPO1通用输出口 Disable GPO1 general-purpose output port
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.3 output2On()

项目 Item	说明 Description
原型 Prototype	boolean output2On()
描述 Description	使能GPO2通用输出口 Enable GPO2 general-purpose output port
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.4 output2Off()

项目 Item	说明 Description
原型 Prototype	boolean output2Off()
描述 Description	禁用GPO2通用输出口 Disable GPO2 general-purpose output port
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.5 output3On()

项目 Item	说明 Description
原型 Prototype	boolean output3On()
描述 Description	使能GPO3通用输出口 Enable GPO3 general-purpose output port
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.6 output3Off()

项目 Item	说明 Description
原型 Prototype	boolean output3Off()
描述 Description	禁用GPO3通用输出口 Disable GPO3 general-purpose output port
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.7 output4On()

项目 Item	说明 Description
原型 Prototype	boolean output4On()
描述 Description	使能GPO4通用输出口 Enable GPO4 general-purpose output port

项目 Item	说明 Description
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.8 output4Off()

项目 Item	说明 Description
原型 Prototype	boolean output4Off()
描述 Description	禁用GPO4通用输出口 Disable GPO4 general-purpose output port
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

11.9 outputOnAndOff(List<GPOEntity> list)

项目 Item	说明 Description
原型 Prototype	void outputOnAndOff(List<GPOEntity> list)
描述 Description	批量控制多路GPO通用输出口的通断状态 Batch control the on/off status of multiple GPO general-purpose output ports
参数 Parameters	list: GPO实体对象列表, 包含待控制的GPO端口编号、目标状态等信息 list: List of GPOEntity objects, containing information such as the number of GPO ports to be controlled and target status
返回 Return	无 N/A

11.10 setGPISetCallback(IGPISetCallback uhfGPISetCallback)

项目 Item	说明 Description
原型 Prototype	void setGPISetCallback(IGPISetCallback uhfGPISetCallback)
描述 Description	设置GPI通用输入口状态回调函数 Set GPI general-purpose input port state callback function
参数 Parameters	uhfGPISetCallback: GPI状态回调接口实例, 用于接收GPI端口状态变更通知 uhfGPISetCallback: IGPISetCallback interface instance, used to receive notifications of GPI port state changes

项目 Item	说明 Description
返回 Return	无 N/A
备注 Remark	可以通过 <code>setGPISStateReverse</code> 函数反转GPI状态 The GPI state can be inverted via the <code>setGPISStateReverse</code> function

11.11 setGPISStateReverse(boolean isReverse)

项目 Item	说明 Description
原型 Prototype	void setGPISStateReverse(boolean isReverse)
描述 Description	设置GPI通用输入口电平状态反转规则 Set the level status inversion rule of GPI general-purpose input ports
参数 Parameters	isReverse: 电平状态反转开关（boolean类型） true: 启用电平反转（高/低电平状态互换） false: 禁用电平反转（使用原始电平状态） isReverse: Level status inversion switch (boolean type) true: Enable level inversion (swap high/low level status) false: Disable level inversion (use original level status)
返回 Return	无 N/A

12 以太网、WIFI

12 Ethernet、WIFI

12.1 getEthernetMac()

项目 Item	说明 Description
原型 Prototype	String getEthernetMac()
描述 Description	获取以太网MAC地址 Get Ethernet MAC address
参数 Parameters	无 N/A
返回 Return	字符串类型：设备以太网MAC地址（格式如XX:XX:XX:XX:XX:XX） String type: Device Ethernet MAC address (format: XX:XX:XX:XX:XX:XX)

12.2 setEthernetConfigInfo(ReaderIPEntity ipconfig)

项目 Item	说明 Description
原型 Prototype	boolean setEthernetConfigInfo(ReaderIPEntity ipconfig)

项目 Item	说明 Description
描述 Description	配置设备以太网静态IP网络参数，配置后以太网将采用静态IP模式工作 Configure the device's Ethernet static IP network parameters; the Ethernet will work in static IP mode after configuration
参数 Parameters	ipconfig: ReaderIPEntity实体对象，包含IP地址、网关、子网掩码、DNS等静态IP配置参数（端口参数暂不生效） ipconfig: ReaderIPEntity entity object, containing static IP configuration parameters such as IP address, gateway, subnet mask, DNS (port parameter is not effective temporarily)
返回 Return	boolean类型：true表示静态IP配置生效，false表示静态IP配置失败 boolean type: true means the static IP configuration takes effect, false means the static IP configuration fails

12.3 getEthernetIpConfig()

项目 Item	说明 Description
原型 Prototype	String getEthernetIpConfig()
描述 Description	获取且仅能获取已配置生效的以太网静态IP网络配置信息 Obtain only the configured and active static IP network configuration information of the Ethernet
参数 Parameters	无 N/A
返回 Return	字符串类型：JSON格式的以太网配置数据，包含IP地址、子网掩码、网关、首选DNS信息 String type: JSON-format Ethernet configuration data, including IP address, subnet mask, gateway, and primary DNS information

12.4 setEthernetIpDynamicAssign()

项目 Item	说明 Description
原型 Prototype	boolean setEthernetIpDynamicAssign()
描述 Description	将设备以太网IP获取方式切换为动态分配（DHCP模式） Switch the device's Ethernet IP acquisition method to dynamic assignment (DHCP mode)
参数 Parameters	无 N/A
返回 Return	boolean类型：true表示DHCP模式切换成功，false表示DHCP模式切换失败 boolean type: true means DHCP mode switching is successful, false means DHCP mode switching fails

12.5 openWifi()

项目 Item	说明 Description
原型 Prototype	boolean openWifi()

项目 Item	说明 Description
描述 Description	启用设备Wi-Fi功能，开启后设备可接入Wi-Fi网络 Enable the device's Wi-Fi function; the device can access the Wi-Fi network after being enabled
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

12.6 closeWifi()

项目 Item	说明 Description
原型 Prototype	boolean closeWifi()
描述 Description	禁用设备Wi-Fi功能，关闭后设备将断开所有Wi-Fi连接 Disable the device's Wi-Fi function; the device will disconnect all Wi-Fi connections after being disabled
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

12.7 getWifiInfo()

项目 Item	说明 Description
原型 Prototype	String getWifiInfo()
描述 Description	获取设备周边可扫描到的Wi-Fi网络列表，返回数据为JSON格式 Obtain the list of scannable Wi-Fi networks around the device, and the returned data is in JSON format
参数 Parameters	无 N/A
返回 Return	字符串类型：JSON格式的Wi-Fi列表数据，包含每个Wi-Fi的SSID（网络名称）、state（连接状态）、securityType（加密类型）字段 String type: JSON-format Wi-Fi list data, including SSID (network name), state (connection status), and securityType (encryption type) fields for each Wi-Fi

12.7 setWifiConfigInfo(WifiConfig wificonfig)

项目 Item	说明 Description
原型 Prototype	boolean setWifiConfigInfo(WifiConfig wificonfig)
描述 Description	配置设备Wi-Fi连接信息（含Wi-Fi认证、IP分配模式、代理等），完成Wi-Fi连接及静态IP设置 Configure device Wi-Fi connection information (including Wi-Fi authentication, IP allocation mode, proxy, etc.), complete Wi-Fi connection and static IP configuration

项目 Item	说明 Description
参数 Parameters	wifiConfig: WifiConfig实体对象，封装Wi-Fi SSID、密码、加密类型、IP分配模式、静态IP参数、代理模式等配置信息（属性私有，仅通过get/set方法赋值） wifiConfig: WifiConfig entity object that encapsulates configuration information such as Wi-Fi SSID, password, encryption type, IP allocation mode, static IP parameters, proxy mode (attributes are private and only assigned via get/set methods)
返回 Return	boolean类型：true表示Wi-Fi配置命令发送成功（不代表Wi-Fi连接成功，连接结果取决于配置参数的正确性，可通过 getWifiInfo() 或者 getWifiIpConfig 接口查询Wi-Fi实际连接状态）；false表示Wi-Fi配置命令发送失败 boolean type: true means the Wi-Fi configuration command is sent successfully (does not mean the Wi-Fi connection is successful; the connection result depends on the correctness of the configuration parameters, and the actual Wi-Fi connection status can be queried through the getWifiInfo() or getWifiIpConfig interface); false means the Wi-Fi configuration command sending fails

12.8 getWifiIpConfig()

项目 Item	说明 Description
原型 Prototype	String getWifiIpConfig()
描述 Description	获取当前已连接Wi-Fi的IP网络配置信息，返回数据为JSON格式 Obtain the IP network configuration information of the currently connected Wi-Fi, and the returned data is in JSON format
参数 Parameters	无 N/A
返回 Return	字符串类型： 1. 非空值：JSON格式的Wi-Fi IP配置数据，包含ssid（Wi-Fi名称）、ip（IP地址）、gateway（网关）、subnetMask（子网掩码）、dns1（首选DNS）、dns2（备用DNS）字段； 2. 空值（null/空字符串）：获取Wi-Fi IP配置信息失败（如未连接Wi-Fi、配置信息未获取到等） String type: 1. Non-null value: JSON-format Wi-Fi IP configuration data, including fields of ssid (Wi-Fi name), ip (IP address), gateway (gateway), subnetMask (subnet mask), dns1 (primary DNS), dns2 (secondary DNS); 2. Null value (null/empty string): Failed to obtain Wi-Fi IP configuration information (e.g., not connected to Wi-Fi, configuration information not retrieved, etc.)

12.9 setUHFServicePort(int port)

项目 Item	说明 Description
原型 Prototype	boolean setUHFServicePort(int port)
描述 Description	设置UHF服务的通信端口号，端口默认值为9160 Set the communication port number of the UHF service, the default value of the port is 9160
参数 Parameters	port: 整型数值，待设置的UHF服务端口号（需符合TCP/UDP端口规范，取值范围1-65535） port: Integer value, the communication port number to be set for the UHF service (must comply with TCP/UDP port specifications, value range 1-65535)

项目 Item	说明 Description
返回 Return	boolean类型：true表示UHF服务端口设置成功，false表示UHF服务端口设置失败（如端口值不合法、UHF服务未初始化等） boolean type: true means the UHF service port is set successfully; false means the UHF service port setting fails (e.g., illegal port value, UHF service not initialized, etc.)

12.10 getReaderCurrentIp()

项目 Item	说明 Description
原型 Prototype	ReaderIPEntity getReaderCurrentIp()
描述 Description	获取读写器当前分配的网络IP配置信息（不区分Wi-Fi/以太网网络类型） Obtain the currently assigned network IP configuration information of the reader (regardless of Wi-Fi/Ethernet network type)
参数 Parameters	无 N/A
返回 Return	ReaderIPEntity类型：读写器当前分配的IP配置实体，包含IP地址、网关、子网掩码、DNS、UHFSERVICE端口等信息 【关键说明】 1. 以太网动态IP场景：返回的IP信息可能与getEthernetIpConfig()接口返回的静态配置信息不一致； 2. 以太网静态IP场景：返回的IP信息与getEthernetIpConfig()接口返回的配置信息完全一致； ReaderIPEntity type: The currently assigned IP configuration entity of the reader, including IP address, gateway, subnet mask, DNS, UHFService port and other information 【Key Notes】 1. Ethernet dynamic IP scenario: The returned IP information may be inconsistent with the static configuration information returned by the getEthernetIpConfig() interface; 2. Ethernet static IP scenario: The returned IP information is completely consistent with the configuration information returned by the getEthernetIpConfig() interface
备注 Remark	该接口返回的是 当前分配的 IP信息，而非预配置的IP信息 This interface returns the currently assigned IP information, not the pre-configured IP information

13 升级功能

13 Upgrade Functions

13.1 readyUpgradeUHFSERVICE()

项目 Item	说明 Description
原型 Prototype	void readyUpgradeUHFSERVICE()
描述 Description	UHFSERVICE升级流程第一步：初始化UHFSERVICE升级环境，完成升级前的准备工作 UHFSERVICE upgrade process Step 1: Initialize the UHFSERVICE upgrade environment and complete preparatory work before upgrade
参数 Parameters	无 N/A
返回 Return	无 N/A

13.2 setUpgradeUHFSERVICEData(byte[] data)

项目 Item	说明 Description
原型 Prototype	boolean setUpgradeUHFSERVICEData(byte[] data)
描述 Description	UHFSERVICE升级流程第二步：设置UHFSERVICE升级数据包，为后续升级操作加载升级数据 UHFSERVICE upgrade process Step 2: Set the UHFSERVICE upgrade data package and load upgrade data for subsequent upgrade operations
参数 Parameters	data：字节数组类型，UHFSERVICE升级数据包 data: Byte array type, UHFSERVICE upgrade data package
返回 Return	boolean类型：true表示升级数据包设置成功，false表示升级数据包设置失败（如数据为空、数据格式不合法等） boolean type: true means the upgrade data package is set successfully; false means the upgrade data package setting fails (e.g., empty data, illegal data format, etc.)
备注 Remark	升级数据可以分多次设置，data 数据长度没有限制 The upgrade data can be set multiple times, and there is no limit on the length of the data array

13.3 startUpgradeUHFSERVICE()

项目 Item	说明 Description
原型 Prototype	boolean startUpgradeUHFSERVICE()
描述 Description	UHFSERVICE升级流程第三步：启动UHFSERVICE升级流程，执行固件升级操作 UHFSERVICE upgrade process Step 3: Start the UHFSERVICE upgrade process and execute the firmware upgrade operation
参数 Parameters	无 N/A
返回 Return	boolean类型：true表示升级流程启动成功（不代表升级完成，需通过升级进度回调确认最终结果）；false表示升级流程启动失败（如未完成前期准备、升级包未加载等） boolean type: true means the upgrade process is started successfully (does not mean the upgrade is completed, the final result needs to be confirmed through the upgrade progress callback); false means the upgrade process startup fails (e.g., preliminary preparation not completed, upgrade package not loaded, etc.)

13.4 setUpgradeProgress(IUpgradeProgress iUpgradeProgress)

项目 Item	说明 Description
原型 Prototype	void setUpgradeProgress(IUpgradeProgress iUpgradeProgress)

项目 Item	说明 Description
描述 Description	设置UHFSERVICE升级进度回调接口，用于接收升级过程中的实时进度和状态信息 Set the UHFService upgrade progress callback interface to receive real-time progress and status information during the upgrade process
参数 Parameters	iUpgradeProgress: IUpgradeProgress接口实例，实现该接口可监听升级进度（progress方法）和升级状态信息（message方法） iUpgradeProgress: IUpgradeProgress interface instance, implementing this interface can monitor upgrade progress (progress method) and upgrade status information (message method)
返回 Return	无 N/A

14 其他功能

14 Other Functions

14.1 factoryReset()

项目 Item	说明 Description
原型 Prototype	boolean factoryReset()
描述 Description	恢复出厂设置 Factory reset
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

14.2 rebootDevice()

项目 Item	说明 Description
原型 Prototype	boolean rebootDevice()
描述 Description	触发设备重启操作 Trigger the device reboot operation
参数 Parameters	无 N/A
返回 Return	true:成功 false:失败 true:success false:failure

14.3 setCsn(String csn)

项目 Item	说明 Description
原型 Prototype	boolean setCsn(String csn)

项目 Item	说明 Description
描述 Description	设置设备自定义CSN（序列号），该序列号用于设备标识，默认值为空字符串 Set the custom CSN (Serial Number) of the device, which is used for device identification and has an empty string as the default value
参数 Parameters	csn: 字符串类型，待设置的自定义CSN号 csn: String type, the custom CSN number to be set
返回 Return	true:成功 false:失败 true:success false:failure
备注 Remark	1. 恢复默认值可传入空字符串 ("") ； 2. 设置的CSN需设备重启后可能生效（具体以设备硬件规格为准） 1. Pass an empty string ("") to restore the default value; 2. The set CSN may take effect only after the device is restarted (subject to the device hardware specifications)

14.4 getCsn()

项目 Item	说明 Description
原型 Prototype	String getCsn()
描述 Description	获取设备当前生效的CSN（序列号），若未设置自定义CSN则返回默认空字符串 Obtain the currently effective CSN (Serial Number) of the device, return the default empty string if no custom CSN is set
参数 Parameters	无 N/A
返回 Return	字符串类型： 1. 非空值：设备已设置的自定义CSN号； 2. 空字符串 ("") ：未设置自定义CSN，使用默认值 String type: 1. Non-null value: The custom CSN number set for the device; 2. Empty string (""): No custom CSN is set, the default value is used
备注 Remark	返回的CSN为设备当前实际生效的序列号，与setCsn()接口设置的数值一致 The returned CSN is the currently effective serial number of the device, consistent with the value set by the setCsn() interface

15 外部相关类

15 External related classes

15.1 InventoryParameter类

15.1 InventoryParameter Class

盘点参数类，用于配置UHF标签盘点的相关参数，包括结果数据类型和标签去重规则。

Inventory parameter class, used to configure parameters related to UHF tag inventory, including result data type and tag deduplication rules.

方法 Method	描述 Description	参数 Parameters	返回值 Return
InventoryParameter()	无参构造函数 No-argument constructor	无 N/A	-
InventoryParameter(ResultData resultData)	带结果数据的构造函数 Constructor with result data	resultData:结果数据配置对象 resultData:Result data configuration object	-
InventoryParameter(boolean isUnique)	带去重标识的构造函数 Constructor with uniqueness flag	isUnique:是否去重标识 isUnique:Uniqueness flag	-
InventoryParameter(ResultData resultData, boolean isTagUnique)	带结果数据和去重标识的构造函数 Constructor with result data and uniqueness flag	resultData:结果数据配置对象 isTagUnique:是否去重标识	-
getResultData()	获取结果数据配置 Get result data configuration	无 N/A	结果数据配置对象 ResultData object
setResultData(ResultData resultData)	设置结果数据配置 Set result data configuration	resultData:结果数据配置对象 resultData:Result data configuration object	当前InventoryParameter实例 Current InventoryParameter instance
isTagUnique()	判断是否启用标签去重 Check if tag deduplication is enabled	无 N/A	布尔值（true:去重, false:不去重） Boolean (true:unique, false:not unique)
setTagUnique(boolean tagUnique)	设置是否启用标签去重 Set whether to enable tag deduplication	tagUnique:是否去重标识 tagUnique:Uniqueness flag	void

15.2 InventoryParameter.ResultData内部类

15.2 InventoryParameter.ResultData Inner Class

结果数据配置类，用于指定盘点结果中需要包含的额外数据（频率点和相位点）。

Result data configuration class, used to specify additional data (frequency point and phase point) to be included in inventory results.

方法 Method	描述 Description	参数 Parameters	返回值 Return
isNeedFrequencyPoint()	判断是否需要频率点数据 Check if frequency point data is needed	无 N/A	布尔值（true:需要, false:不需要） Boolean (true:needed, false:not needed)
isNeedPhase()	判断是否需要相位点数据 Check if phase point data is needed	无 N/A	布尔值（true:需要, false:不需要） Boolean (true:needed, false:not needed)
setNeedFrequencyPoint(boolean needFrequencyPoint)	设置是否需要频率点数据 Set whether frequency point data is needed	needFrequencyPoint:是否需要频率点标识 needFrequencyPoint:Frequency point need flag	当前ResultData实例 Current ResultData instance

方法 Method	描述 Description	参数 Parameters	返回值 Return
setNeedPhase(boolean needPhase)	设置是否需要相位点数据 Set whether phase point data is needed	needPhase:是否需要相位点标识 needPhase:Phase point need flag	当前ResultData实例 Current ResultData instance

15.3 InventoryModeEntity类

15.3 InventoryModeEntity Class

盘点模式实体类，用于定义UHF标签盘点时的输出数据模式，包括不同存储区数据的输出配置及访问密码等信息。

Inventory mode entity class, used to define the output data mode during UHF tag inventory, including output configuration of data in different storage areas and access password information.

常量 Constant	描述 Description
MODE_EPC	EPC模式，盘点时仅输出标签的EPC数据，不能指定EPC的起始地址和长度 EPC mode, only output EPC data of the tag, can not specify the start address and length of EPC
MODE_EPC_TID	EPC+TID模式，盘点时输出标签的EPC和TID数据，不能指定EPC和TID的起始地址和长度 EPC+TID mode, output EPC and TID data of the tag, can not specify the start address and length of EPC and TID
MODE_EPC_TID_USER	EPC+TID+USER模式，盘点时输出标签的EPC、TID和USER数据，只能指定USER的起始地址和长度 EPC+TID+USER mode, output EPC, TID and USER data of the tag, can only specify the start address and length of USER
MODE_EPC_TID_M775AUTHENTICATION	EPC+TID+M775模式，盘点时输出标签的EPC、TID和M775认证数据，不能指定EPC、TID和M775认证数据的起始地址和长度 EPC+TID+M775 mode, output EPC, TID and M775 authentication data of the tag, can not specify the start address and length of EPC, TID and M775 authentication data
MODE_EPC_TID_RESERVED	EPC+TID+RESERVED模式，盘点时输出标签的EPC、TID和RESERVED数据，且只能指定RESERVED数据的起始地址和长度 EPC+TID+RESERVED mode, output EPC, TID and RESERVED data of the tag, and can only specify the start address and length of RESERVED data
MODE_EPC_RESERVED	EPC+RESERVED模式，盘点时仅输出标签的EPC和RESERVED数据，且只能指定RESERVED数据的起始地址和长度 EPC+RESERVED mode, output EPC and RESERVED data of the tag, and can only specify the start address and length of RESERVED data
MODE_EPC_MEMORY_BANK_EPC	EPC+MEMORY_BANK_EPC模式，盘点时输出标签的EPC和MEMORY_BANK_EPC数据，且只能指定EPC数据的起始地址和长度 EPC+MEMORY_BANK_EPC mode, output EPC and MEMORY_BANK_EPC data of the tag, and can only specify the start address and length of EPC data

常量 Constant	描述 Description
MODE_EPC_MEMORY_BANK_TID	EPC+MEMORY_BANK_TID模式，盘点时输出标签的EPC和MEMORY_BANK_TID数据，且只能指定TID数据的起始地址和长度 EPC+MEMORY_BANK_TID mode, output EPC and MEMORY_BANK_TID data of the tag, and can only specify the start address and length of TID data
MODE_EPC_MEMORY_BANK_USER	EPC+MEMORY_BANK_USER模式，盘点时输出标签的EPC和MEMORY_BANK_USER数据，且只能指定USER数据的起始地址和长度 EPC+MEMORY_BANK_USER mode, output EPC and MEMORY_BANK_USER data of the tag, and can only specify the start address and length of USER data
MODE_LED_TAG	LED_TAG模式，盘点时点亮LED标签 LED_TAG mode, light up LED tag during inventory
MODE_TEMPERATURE_TAG	TEMPERATURE_TAG模式，盘点时输出标签的温度数据 TEMPERATURE_TAG mode, output temperature data of the tag during inventory

方法 Method	描述 Description	参数 Parameters	返回值 Return
getMode()	获取盘点模式 Get inventory mode	无 N/A	盘点模式（对应上述常量值） Inventory mode (corresponding to the above constant values)
getUserLength()	获取要盘点的User区长度 Get User area data length	无 N/A	User区长度 User area data length
getUserOffset()	获取要盘点的User区起始地址 Get User area start address	无 N/A	User区起始地址 User area start address
getReservedLength()	获取要盘点的RESERVED区长度 Get RESERVED area data length	无 N/A	RESERVED区长度 RESERVED area data length
getReservedOffset()	获取要盘点的RESERVED区起始地址 Get RESERVED area start address	无 N/A	RESERVED区起始地址 RESERVED area start address
getEpcLength()	获取要盘点的EPC区长度 Get EPC area data length	无 N/A	EPC区长度 EPC area data length
getEpcOffset()	获取要盘点的EPC区起始地址 Get EPC area start address	无 N/A	EPC区起始地址 EPC area start address
getTidLength()	获取要盘点的TID区长度 Get TID area data length	无 N/A	TID区长度 TID area data length

方法 Method	描述 Description	参数 Parameters	返回值 Return
getTidOffset()	获取要盘点的TID区起始地址 Get TID area start address	无 N/A	TID区起始地址 TID area start address
getHexAccessPassword()	获取访问密码 Get access password	无 N/A	访问密码（十六进制字符串） Access password (hex string)

15.4 Builder内部类

15.4 InventoryModeEntity.Builder Inner Class

InventoryModeEntity的构建器类，用于便捷地创建和配置InventoryModeEntity实例。

Builder class for InventoryModeEntity, used to conveniently create and configure InventoryModeEntity instances.

方法 Method	描述 Description	参数 Parameters	返回值 Return
setMode(int mode)	设置盘点模式 Set inventory mode	mode: 盘点模式（参考上述常量） mode: Inventory mode (refer to the above constants)	构建器实例（Builder） Builder instance
setUserLength(int userLen)	设置要盘点的User区长度 Set User area data length	userLen: User区长度 userLen: User area data length	构建器实例（Builder） Builder instance
setUserOffset(int userOffset)	设置要盘点的User区起始地址 Set User area start address	userOffset: User区起始地址 userOffset: User area start address	构建器实例（Builder） Builder instance
setReservedLength(int reservedLength)	设置要盘点的RESERVED区长度 Set RESERVED area data length	reservedLength: RESERVED区长度 reservedLength: RESERVED area data length	构建器实例（Builder） Builder instance
setReservedOffset(int reservedOffset)	设置要盘点的RESERVED区起始地址 Set RESERVED area start address	reservedOffset: RESERVED区起始地址 reservedOffset: RESERVED area start address	构建器实例（Builder） Builder instance
setEpcLength(int epcLength)	设置要盘点的EPC区长度 Set EPC area data length	epcLength: EPC区长度 epcLength: EPC area data length	构建器实例（Builder） Builder instance
setEpcOffset(int epcOffset)	设置要盘点的EPC区起始地址 Set EPC area start address	epcOffset: EPC区起始地址 epcOffset: EPC area start address	构建器实例（Builder） Builder instance
setTidLength(int tidLength)	设置要盘点的TID区长度 Set TID area data length	tidLength: TID区长度 tidLength: TID area data length	构建器实例（Builder） Builder instance

方法 Method	描述 Description	参数 Parameters	返回值 Return
setTidOffset(int tidOffset)	设置要盘点的TID区起始地址 Set TID area start address	tidOffset: TID区起始地址 tidOffset: TID area start address	构建器实例 (Builder) Builder instance
setHexAccessPassword(String pwd)	设置访问密码 Set access password	pwd: 访问密码 (十六进制字符串) pwd: Access password (hex string)	构建器实例 (Builder) Builder instance
build()	构建盘点模式实体 Build inventory mode entity	无 N/A	配置完成的InventoryModeEntity实例 Configured InventoryModeEntity instance

15.5 Gen2Entity类

15.5 Gen2Entity Class

Gen2协议参数类，用于配置与Gen2协议相关的各种参数，包括select命令和query命令的参数以及链路频率等。

Gen2 protocol parameter class, used to configure various parameters related to the Gen2 protocol, including parameters of the select command, query command, and link frequency, etc.

方法 Method	描述 Description	参数 Parameters	返回值 Return
checkParameter()	检查所有参数是否都已设置 (均不为-1) Check if all parameters have been set (none are -1)	无 N/A	布尔值 (true:所有参数均已设置, false:存在未设置的参数) Boolean (true:all parameters are set, false:there are unset parameters)
getSelectTarget()	获取select命令的Target参数 Get the Target parameter of the select command	无 N/A	整数 (0:s0, 1:s1, 2:s2, 3:s3, 4:SL) Integer (0:s0, 1:s1, 2:s2, 3:s3, 4:SL)
setSelectTarget(int selectTarget)	设置select命令的Target参数 Set the Target parameter of the select command	selectTarget:整数 (0:s0, 1:s1, 2:s2, 3:s3, 4:SL) selectTarget:Integer (0:s0, 1:s1, 2:s2, 3:s3, 4:SL)	void
getSelectAction()	获取select命令的Action参数 Get the Action parameter of the select command	无 N/A	整数 (0-7, 分别对应不同的匹配和非匹配操作) Integer (0-7, corresponding to different matching and non-matching operations)

方法 Method	描述 Description	参数 Parameters	返回值 Return
setSelectAction(int selectAction)	设置select命令的Action参数 Set the Action parameter of the select command	selectAction:整数 (0-7, 分别对应不同的匹配和非匹配操作) selectAction:Integer (0-7, corresponding to different matching and non-matching operations)	void
getSelectTruncate()	获取select命令的Truncate参数 Get the Truncate parameter of the select command	无 N/A	整数 (0:禁用截断, 1:启用截断) Integer (0:Disable truncation, 1:Enable truncation)
setSelectTruncate(int selectTruncate)	设置select命令的Truncate参数 Set the Truncate parameter of the select command	selectTruncate:整数 (0:禁用截断, 1:启用截断) selectTruncate:Integer (0:Disable truncation, 1:Enable truncation)	void
getQ()	获取Q值 Get the Q value	无 N/A	整数 (0:固定Q算法, 1:动态Q算法) Integer (0:fixed Q algorithm, 1:dynamic Q algorithm)
setQ(int q)	设置Q值 Set the Q value	q:整数 (0:固定Q算法, 1:动态Q算法) q:Integer (0:fixed Q algorithm, 1:dynamic Q algorithm)	void
getStartQ()	获取StartQ值 Get the StartQ value	无 N/A	整数 (0-15) Integer (0-15)
setStartQ(int startQ)	设置StartQ值 Set the StartQ value	startQ:整数 (0-15) startQ:Integer (0-15)	void
getMinQ()	获取MinQ值 Get the MinQ value	无 N/A	整数 (0-15) Integer (0-15)
setMinQ(int minQ)	设置MinQ值 Set the MinQ value	minQ:整数 (0-15) minQ:Integer (0-15)	void
getMaxQ()	获取MaxQ值 Get the MaxQ value	无 N/A	整数 (0-15) Integer (0-15)
setMaxQ(int maxQ)	设置MaxQ值 Set the MaxQ value	maxQ:整数 (0-15) maxQ:Integer (0-15)	void
getQueryDR()	获取query命令的DR参数 Get the DR parameter of the query command	无 N/A	整数 (0:8, 1:64/3) Integer (0:8, 1:64/3)

方法 Method	描述 Description	参数 Parameters	返回值 Return
setQueryDR(int queryDR)	设置query命令的DR参数 Set the DR parameter of the query command	queryDR:整数 (0:8, 1:64/3) queryDR:Integer (0:8, 1:64/3)	void
getQueryM()	获取query命令的M参数 Get the M parameter of the query command	无 N/A	整数 (0:FM0, 1:Miller2, 2:Miller4, 3:Miller8) Integer (0:FM0, 1:Miller2, 2:Miller4, 3:Miller8)
setQueryM(int queryM)	设置query命令的M参数 Set the M parameter of the query command	queryM:整数 (0:FM0, 1:Miller2, 2:Miller4, 3:Miller8) queryM:Integer (0:FM0, 1:Miller2, 2:Miller4, 3:Miller8)	void
getQueryTRext()	获取query命令的TRext参数 Get the TRext parameter of the query command	无 N/A	整数 (0:无导频, 1:使用导频) Integer (0:No pilot, 1:Use pilot)
setQueryTRext(int queryTRext)	设置query命令的TRext参数 Set the TRext parameter of the query command	queryTRext:整数 (0:无导频, 1:使用导频) queryTRext:Integer (0:No pilot, 1:Use pilot)	void
getQuerySel()	获取query命令的sel参数 Get the sel parameter of the query command	无 N/A	整数 (0:所有, 1:所有, 2:~SL, 3:SL) Integer (0:All, 1:All, 2:~SL, 3:SL)
setQuerySel(int querySel)	设置query命令的sel参数 Set the sel parameter of the query command	querySel:整数 (0:所有, 1:所有, 2:~SL, 3:SL) querySel:Integer (0:All, 1:All, 2:~SL, 3:SL)	void
getQuerySession()	获取query命令的session参数 Get the session parameter of the query command	无 N/A	整数 (0:S0, 1:S1, 2:S2, 3:S3) Integer (0:S0, 1:S1, 2:S2, 3:S3)
setQuerySession(int querySession)	设置query命令的session参数 Set the session parameter of the query command	querySession:整数 (0:S0, 1:S1, 2:S2, 3:S3) querySession:Integer (0:S0, 1:S1, 2:S2, 3:S3)	void

方法 Method	描述 Description	参数 Parameters	返回值 Return
getQueryTarget()	获取query命令的Target参数 Get the Target parameter of the query command	无 N/A	整数 (0:A, 1:B) Integer (0:A, 1:B)
setQueryTarget(int queryTarget)	设置query命令的Target参数 Set the Target parameter of the query command	queryTarget:整数 (0:A, 1:B) queryTarget:Integer (0:A, 1:B)	void
getLinkFrequency()	获取链路频率设置 Get the link frequency setting	无 N/A	-
setLinkFrequency(int linkFrequency)	设置链路频率 Set the link frequency	-	void
toString()	返回对象的字符串表示 Return the string representation of the object	无 N/A	包含所有参数信息的字符串 String containing all parameter information

15.6 ErrorCode

15.6 错误码定义

常量名 Constant Name	数值 Value	说明 Description
ERROR_NO_TAG	0x01	找不到标签 No tags found.
ERROR_INSUFFICIENT_PRIVILEGES	0x02	没有权限访问 The Interrogator did not authenticate itself with sufficient privileges for the Tag to perform the operation
ERROR_MEMORY_OVERRUN	0x03	数据区超限 Memory Overrun
ERROR_MEMORY_LOCK	0x04	数据区被锁定 Memory lock
ERROR_TAG_NO_REPLY	0x05	标签没有应答 Tag has no response
ERROR_PASSWORD_IS_INCORRECT	0x06	密码不正确 Incorrect Password
ERROR_RESPONSE_BUFFER_OVERFLOW	0x07	缓冲区溢出 The operation failed because Response buffer overflowed
ERROR_NO_ENOUGH_POWER_ON_TAG	0x0B	标签能量不足 no enough power on tag

常量名 Constant Name	数值 Value	说明 Description
ERROR_OPERATION_FAILED	0xFF	操作失败 Operation failure
ERROR_SEND_FAIL	0xFC	发送失败 Send failure
ERROR_RECV_FAIL	0xFD	接收失败 Receive failure
ERRCODE_SUCCESS	0	成功 Success
ERRCODE_FAILURE	-1	失败 Failure

15.7 FastInventoryEntity类

15.7 FastInventoryEntity Class

快速盘点相关参数类。

Fast Inventory related parameters.

方法 Method	描述 Description	参数 Parameters	返回值 Return
FastInventoryEntity(int cr)	带CR值的构造函数 Constructor with CR value	cr	-
getCr()	获取CR值 Get CR value	无 N/A	0x00:close 0x01:CrID16 0x02:CrStoredCRC 0x03:CrRN16 0x04:CrID32
setCr(int cr)	设置CR值 Set CR value	0x00:close 0x01:CrID16 0x02:CrStoredCRC 0x03:CrRN16 0x04:CrID32	当前FastInventoryEntity实例 Current FastInventoryEntity instance

15.8 AntennaPowerEntity类

15.8 AntennaPowerEntity Class

天线功率实体类，用于封装天线枚举信息与对应功率数据

Antenna power entity class, used to encapsulate antenna enumeration information and corresponding power data

方法 Method	描述 Description	参数 Parameters	返回值 Return
getAnt()	获取天线枚举对象 Get the antenna enumeration object	无 N/A	天线枚举值（AntennaEnum类型） Antenna enumeration value (AntennaEnum type)

方法 Method	描述 Description	参数 Parameters	返回值 Return
getPower()	获取天线功率值 Get the antenna power value	无 N/A	天线功率值（int类型） Antenna power value (int type)

15.9 AntennaConnectState类

15.9 AntennaConnectState Class

天线连接状态类，用于封装天线枚举信息与对应连接状态
Antenna connection status class, used to encapsulate antenna enumeration information and corresponding connection status

方法 Method	描述 Description	参数 Parameters	返回值 Return
getAntennaName()	获取天线名称（枚举值） Get antenna name (enumeration value)	无 N/A	天线枚举值（AntennaEnum类型） Antenna enumeration value (AntennaEnum type)
isConnected()	获取天线连接状态 Get antenna connection status	无 N/A	连接状态（boolean类型，true为已连接，false为未连接） Connection status (boolean type, true for connected, false for disconnected)

15.10 AntennaState类

15.10 AntennaState Class

天线启用状态类，用于封装天线枚举信息与对应启用状态
Antenna enable state class, used to encapsulate antenna enumeration information and corresponding enable state

方法 Method	描述 Description	参数 Parameters	返回值 Return
AntennaState(AntennaEnum antName, boolean enable)	带参构造函数，初始化天线名称和启用状态 Parameterized constructor, initializes antenna name and enable state	antName:天线枚举值（AntennaEnum类型） antName: Antenna enumeration value (AntennaEnum type) enable:启用状态（true为启用，false为不启用） enable: Enable state (true for enabled, false for disabled)	-
getAntennaName()	获取天线名称（枚举值） Get antenna name (enumeration value)	无 N/A	天线枚举值（AntennaEnum类型） Antenna enumeration value (AntennaEnum type)

方法 Method	描述 Description	参数 Parameters	返回值 Return
isEnabled()	获取天线启用状态 Get antenna enable state	无 N/A	启用状态（boolean类型，true为启用，false为不启用） Enable state (boolean type, true for enabled, false for disabled)
setEnabled(boolean enable)	设置天线启用状态 Set antenna enable state	enable:启用状态（true为启用，false为不启用） enable: Enable state (true for enabled, false for disabled)	void

15.11 GPIStateEntity类

15.11 GPIStateEntity Class

GPI端口状态信息实体类，用于封装GPI通用输入口的名称及电平状态信息
GPI port status information entity class, used to encapsulate the name and level status information of GPI general-purpose input ports

方法/常量 Method/Constant	描述 Description	参数 Parameters	返回值 Return
常量 Constant	-	-	-
GPI1	表示1号GPI通用输入口 Represents No.1 GPI general-purpose input port	无 N/A	字符串常量: "GPI1" String constant: "GPI1"
GPI2	表示2号GPI通用输入口 Represents No.2 GPI general-purpose input port	无 N/A	字符串常量: "GPI2" String constant: "GPI2"
GPI3	表示3号GPI通用输入口 Represents No.3 GPI general-purpose input port	无 N/A	字符串常量: "GPI3" String constant: "GPI3"
GPI4	表示4号GPI通用输入口 Represents No.4 GPI general-purpose input port	无 N/A	字符串常量: "GPI4" String constant: "GPI4"
普通方法 Method	-	-	-
getGpiName()	获取GPI端口名称（编号） Get GPI port name (number)	无 N/A	GPI端口名称（String类型） GPI port name (String type)

方法/常量 Method/Constant	描述 Description	参数 Parameters	返回值 Return
setGpiName(String gpiName)	设置GPI端口名称（编号） Set GPI port name (number)	gpiName: GPI端口名称 (String类型) GPI port name (String type)	无 N/A
getGpiState()	获取GPI端口电平状态 Get GPI port level status	无 N/A	GPI端口状态（int类型），0表示低电平，1表示高电平 GPI port status (int type), 0 means low level, 1 means high level
setGpiState(int gpiState)	设置GPI端口电平状态 Set GPI port level status	gpiState: GPI端口状态（int类型），0表示低电平，1表示高电平 GPI port status (int type), 0 means low level, 1 means high level	无 N/A

15.12 FilterEntity类

15.12 FilterEntity Class

过滤器实体类，用于封装RFID标签过滤相关的配置信息，包括过滤存储区、起始地址、数据长度及过滤数据

Filter entity class, used to encapsulate RFID tag filtering related configuration information, including filter storage area, start address, data length and filter data

方法 Method	描述 Description	参数 Parameters	返回值 Return
FilterEntity(int filterBank, int filterOffset, int filterLength, String filterData)	构造函数，初始化过滤器实体对象 Constructor, initializes the filter entity object	filterBank: 存储区标识（int类型，0x00:保留Reserve, 0x01:EPC区, 0x02:TID区, 0x03:用户区USR） filterBank: Storage area identifier (int type, 0x00: Reserve, 0x01: EPC, 0x02: TID, 0x03: USR) filterOffset: 过滤起始地址（int类型） filterOffset: Filter start address (int type) filterLength: 过滤数据长度（int类型，单位：bit；长度为0时表示无过滤） filterLength: Filter data length (int type, unit: bit; length 0 means no filter) filterData: 过滤数据（String类型） filterData: Filter data (String type)	无 N/A

方法 Method	描述 Description	参数 Parameters	返回值 Return
getFilterBank()	获取过滤存储区 Get filter storage area	无 N/A	存储区标识 (int类型, 0x00:保留Reserve, 0x01:EPC区, 0x02:TID区, 0x03:用户区USR) Storage area identifier (int type, 0x00: Reserve, 0x01: EPC, 0x02: TID, 0x03: USR)
setFilterBank(int filterBank)	设置过滤存储区 Set filter storage area	filterBank: 存储区标识 (int类型, 0x00:保留Reserve, 0x01:EPC区, 0x02:TID区, 0x03:用户区USR) filterBank: Storage area identifier (int type, 0x00: Reserve, 0x01: EPC, 0x02: TID, 0x03: USR)	无 N/A
getFilterOffset()	获取过滤起始地址 Get filter start address	无 N/A	起始地址值 (int类型) Start address value (int type)
setFilterOffset(int filterOffset)	设置过滤起始地址 Set filter start address	filterOffset: 过滤起始地址 (int类型) filterOffset: Filter start address (int type)	无 N/A
getFilterLength()	获取过滤数据长度 Get filter data length	无 N/A	数据长度值 (int类型, 单位: bit; 长度为0时表示无过滤) Data length value (int type, unit: bit; length 0 means no filter)
setFilterLength(int filterLength)	设置过滤数据长度 Set filter data length	filterLength: 过滤数据长度 (int类型, 单位: bit; 长度为0时表示无过滤) filterLength: Filter data length (int type, unit: bit; length 0 means no filter)	无 N/A
getFilterData()	获取过滤数据 Get filter data	无 N/A	过滤数据 (String类型) Filter data (String type)
setFilterData(String filterData)	设置过滤数据 Set filter data	filterData: 过滤数据 (String类型) filterData: Filter data (String type)	无 N/A

15.13 ConnectionStatusCallback接口

15.13 ConnectionStatusCallback Interface

连接状态回调接口，用于接收连接状态变化通知

Connection status callback interface, used to receive connection status change notifications

方法 Method	描述 Description	参数 Parameters	返回值 Return
getStatus(final ConnectionStatus status, final T device)	接收连接状态信息 Receive connection status information	status: 状态 status device: 目标设备 target device	无 N/A

15.14 ReaderIPEntity类

15.14 ReaderIPEntity Class

读写器IP信息类，用于封装读写器的IP地址、端口、网关、子网掩码及DNS等网络配置信息

Reader IP information class, used to encapsulate network configuration information of the reader such as IP address, port, gateway, subnet mask and DNS

方法 Method	描述 Description	参数 Parameters	返回值 Return
ReaderIPEntity()	无参构造方法 No-argument constructor	无 N/A	无 N/A
ReaderIPEntity(String ip, int port)	带参构造方法（初始化IP地址和端口） Parameterized constructor (initializes IP address and port)	ip: IP地址（String类型） IP address (String type) port: 端口号（int类型） Port number (int type)	无 N/A
getPort()	获取端口号 Get port number	无 N/A	端口号（int类型） Port number (int type)
setPort(int port)	设置端口号 Set port number	port: 端口号（int类型） Port number (int type)	无 N/A
getIp()	获取IP地址 Get IP address	无 N/A	IP地址（String类型） IP address (String type)
setIp(String ip)	设置IP地址 Set IP address	ip: IP地址（String类型） IP address (String type)	无 N/A
getGateway()	获取网关地址 Get gateway address	无 N/A	网关地址（String类型） Gateway address (String type)
setGateway(String gateway)	设置网关地址 Set gateway address	gateway: 网关地址（String类型） Gateway address (String type)	无 N/A

方法 Method	描述 Description	参数 Parameters	返回值 Return
getSubnetMask()	获取子网掩码 Get subnet mask	无 N/A	子网掩码 (String类型) Subnet mask (String type)
setSubnetMask(String subnetMask)	设置子网掩码 Set subnet mask	subnetMask: 子网掩码 (String类型) Subnet mask (String type)	无 N/A
getDns1()	获取首选DNS地址 Get primary DNS address	无 N/A	首选DNS地址 (String类型) Primary DNS address (String type)
setDns1(String dns1)	设置首选DNS地址 Set primary DNS address	dns1: 首选DNS地址 (String类型) Primary DNS address (String type)	无 N/A

15.15 GPOEntity类

15.15 GPOEntity Class

GPO端口信息实体类，用于封装GPO通用输出口/韦根数据口的名称及状态信息
GPO port information entity class, used to encapsulate the name and status information of GPO general-purpose output ports/Wiegand data ports

方法/常量 Method/Constant	描述 Description	参数 Parameters	返回值 Return
常量 Constant	-	-	-
GPO1	表示1号GPO通用输出口 Represents No.1 GPO general-purpose output port	无 N/A	字符串常量: "GPO1" String constant: "GPO1"
GPO2	表示2号GPO通用输出口 Represents No.2 GPO general-purpose output port	无 N/A	字符串常量: "GPO2" String constant: "GPO2"
GPO3	表示3号GPO通用输出口 Represents No.3 GPO general-purpose output port	无 N/A	字符串常量: "GPO3" String constant: "GPO3"
GPO4	表示4号GPO通用输出口 Represents No.4 GPO general-purpose output port	无 N/A	字符串常量: "GPO4" String constant: "GPO4"
构造方法 Constructor	-	-	-

方法/常量 Method/Constant	描述 Description	参数 Parameters	返回值 Return
GPOEntity(String gpoName, int gpoState)	带参构造方法（初始化GPO端口名称和状态） Parameterized constructor (initializes GPO port name and status)	gpoName: GPO端口名称 (String类型) GPO port name (String type) gpoState: GPO端口状态 (int类型) GPO port status (int type)	无 N/A
普通方法 Method	-	-	-
getGpoName()	获取GPO端口名称 Get GPO port name	无 N/A	GPO端口名称 (String类型) GPO port name (String type)
setGpoName(String gpoName)	设置GPO端口名称 Set GPO port name	gpoName: GPO端口名称 (String类型) GPO port name (String type)	无 N/A
getGpoState()	获取GPO端口状态 Get GPO port status	无 N/A	GPO端口状态 (int类型) GPO port status (int type)
setGpoState(int gpoState)	设置GPO端口状态 Set GPO port status	gpoState: GPO端口状态 (int类型), 0表示关闭(Off), 1表示开启(On) GPO port status (int type), 0 means Off, 1 means On	无 N/A

15.16 IGPIStateCallback接口

15.16 IGPIStateCallback Interface

GPI状态回调接口，用于接收GPI通用输入口状态变更的批量通知
 IGPIStateCallback interface, used to receive batch notifications of GPI general-purpose input port status changes

方法 Method	描述 Description	参数 Parameters	返回值 Return
抽象方法 Abstract Method	-	-	-
callback(List<GPIStateEntity> gpioState)	GPI状态变更回调方法，当GPI端口状态发生变化时触发 GPI status change callback method, triggered when the GPI port status changes	gpioState: GPI状态实体对象列表，包含状态变更的GPI端口名称及电平状态信息 gpioState: List of GPIStateEntity objects, containing the name and level status information of GPI ports with status changes	无 N/A

15.17 WifiConfig类

15.17 WifiConfig Class

Wi-Fi配置信息实体类，用于封装Wi-Fi连接的网络参数、认证方式、IP分配模式及代理配置等信息（类内属性均为私有，仅通过get/set方法访问/修改）
 Wi-Fi configuration information entity class, used to encapsulate network parameters, authentication methods, IP allocation modes and proxy configurations for Wi-Fi connections (all internal attributes are private and can only be accessed/modified

through get/set methods)

方法 Method	描述 Description	参数/返回值说明 Parameter/Return Description	备注/取值说明 Remarks/Value Description
Getter/Setter 方法	-	-	-
getSsid()	获取Wi-Fi网络名称 Get Wi-Fi network name	返回值：Wi-Fi网络名称 (String类型) Return value: Wi-Fi network name (String type)	-
setSsid(String ssid)	设置Wi-Fi网络名称 Set Wi-Fi network name	参数：ssid - Wi-Fi网络名称 (String类型) Parameter: ssid - Wi-Fi network name (String type)	-
getPassword()	获取Wi-Fi认证密码 Get Wi-Fi authentication password	返回值：Wi-Fi认证密码 (String类型) Return value: Wi-Fi authentication password (String type)	当加密类型为2（NOPASS）时，该字 段无效 Invalid when security type is 2 (NOPASS)
setPassword(String password)	设置Wi-Fi认证密码 Set Wi-Fi authentication password	参数：password - Wi-Fi认证 密码（String类型） Parameter: password - Wi-Fi authentication password (String type)	-
getSecurityType()	获取Wi-Fi加密认证 类型 Get Wi-Fi encryption authentication type	返回值：加密认证类型（int类 型） Return value: Encryption authentication type (int type)	0: WEP加密 1: WPA加密 2: 无密码（NOPASS） 3: 无效类型（INVALID）
setSecurityType(int securityType)	设置Wi-Fi加密认证 类型 Set Wi-Fi encryption authentication type	参数：securityType - 加密认 证类型（int类型） Parameter: securityType - Encryption authentication type (int type)	-
getStaticIpConfig()	获取Wi-Fi静态 IP配 置信息 Get Wi-Fi static IP configuration information	返回值：静态IP配置实体 (ReaderIPEntity类型) Return value: Static IP configuration entity (ReaderIPEntity type)	仅当IP分配模式为静态 (isStaticIp=true) 时生效，包含IP地 址、网关、子网掩码、DNS等参数 Only valid when IP allocation mode is static (isStaticIp=true), contains parameters such as IP address, gateway, subnet mask, DNS
setStaticIpConfig(ReaderIPEntity staticIpconfig)	设置Wi-Fi静态IP配 置信息 Set Wi-Fi static IP configuration information	参数：staticIpconfig - 静态IP 配置实体（ReaderIPEntity类 型） Parameter: staticIpconfig - Static IP configuration entity (ReaderIPEntity type)	-
getProxy()	获取Wi-Fi代理模式 Get Wi-Fi proxy mode	返回值：代理模式（int类型） Return value: Proxy mode (int type)	0: 无代理 1: 手动代理 2: 自动代理

方法 Method	描述 Description	参数/返回值说明 Parameter/Return Description	备注/取值说明 Remarks/Value Description
setProxy(int proxy)	设置Wi-Fi代理模式 Set Wi-Fi proxy mode	参数： proxy - 代理模式（int类型） Parameter: proxy - Proxy mode (int type)	-
isStaticIp()	获取Wi-Fi IP分配模式标识 Get Wi-Fi IP allocation mode flag	返回值： IP分配模式标识（boolean类型） Return value: IP allocation mode flag (boolean type)	false: 动态IP（DHCP） true: 静态IP
setStaticIp(boolean isStaticIp)	设置Wi-Fi IP分配模式标识 Set Wi-Fi IP allocation mode flag	参数： isStaticIp - IP分配模式标识（boolean类型） Parameter: isStaticIp - IP allocation mode flag (boolean type)	-

15.18 IUpgradeProgress接口

15.18 IUpgradeProgress Interface

UHF设备升级进度回调接口，用于实时反馈升级过程中的进度值和状态信息
UHF device upgrade progress callback interface, used to real-time feed back the progress value and status information during the upgrade process

方法 Method	说明 Description	参数说明 Parameter Description	返回值 Return Value
progress(int value)	升级进度更新回调方法，实时返回升级进度百分比 Upgrade progress update callback method, returns the upgrade progress percentage in real time	value： 整型数值，升级进度百分比（取值范围0-100，0表示未开始，100表示升级完成） value: Integer value, upgrade progress percentage (value range 0-100, 0 means not started, 100 means upgrade completed)	无 (void) None (void)
message(String msg)	升级状态信息回调方法，返回升级过程中的实时状态提示 Upgrade status information callback method, returns real-time status prompts during the upgrade process	msg： 字符串类型，升级状态描述信息（如"正在下载升级包"、"升级中"、"升级成功"、"升级失败：文件校验错误"等） msg: String type, upgrade status description information (e.g., "Downloading upgrade package", "Upgrading", "Upgrade successful", "Upgrade failed: file verification error", etc.)	无 (void) None (void)

16 多设备连接

Multi-device Connection

`RFIDWithUHFA4Host.ManageMultipleDevice` 多设备管理核心类，用于统一管理多个网口连接的UHF设备。通过维护设备列表（以IP地址为唯一标识），提供设备的添加、获取、移除、批量连接/断开及回调设置等功能，支持多设备的集中化控制与状态管理。

`RFIDWithUHFA4Host.ManageMultipleDevice` Core class for multi-device management, used to uniformly manage multiple UHF devices connected via Ethernet. It maintains a device list (with IP address as the unique identifier) and provides functions such as adding, obtaining, removing devices, batch connecting/disconnecting, and setting callbacks, supporting centralized control and status management of multiple devices.

1. 使用步骤

1. Usage Steps

第一步: 实例化多连接对象

Step 1: Initialize the multi-device manager

```
RFIDWithUHFA4Host.ManageMultipleDevice deviceManager = new  
RFIDWithUHFA4Host.ManageMultipleDevice(context);
```

第二步: 添加设备

Step 2: Add Ethernet devices to the manager (IP address + port number)

```
deviceManager.addDevice("192.168.99.202", 8888);  
deviceManager.addDevice("192.168.99.203", 8888);
```

第三步: 监听连接状态回调

Step 3: Set a unified connection status callback for all devices

```
deviceManager.setConnectionStatusCallbackAll(new ConnectionStatusCallback());
```

第四步: 连接所有设备

Step 4: Connect all added devices in batch

```
deviceManager.connectAll();  
deviceManager.setConnectionStatusCallbackAll(new ConnectionStatusCallback() {  
    @Override  
    public void getStatus(ConnectionStatus status, Object device) {  
        if(device!=null){  
            String ip=(String) device;  
        }  
        if(status==ConnectionStatus.CONNECTED){  
  
        }else if(status==ConnectionStatus.DISCONNECTED){  
  
        } else if(status==ConnectionStatus.CONNECTING){  
  
        }  
    }  
});  
deviceManager.setInventoryCallbackAll(new IUHFInventoryCallback(){  
    @Override  
    public void callback(UHFTAGInfo uhftagInfo) {  
        String ip=uhftagInfo.getIp();  
        String epc=uhftagInfo.getEPC();  
    }  
});
```

第五步: 获取指定设备对象, 对其操作

step 5: Step 5: Get the specified device object and operate on it

```
deviceManager.setInventoryCallbackAll(new IUHFInventoryCallback(){  
    @Override  
    public void callback(UHFTAGInfo uhftagInfo) {  
        String ip=uhftagInfo.getIp();  
        String epc=uhftagInfo.getEPC();  
    }  
});
```

```

IUHFA4 uhf=deviceManager.getDevice("192.168.99.202");
uhf.startInventoryTag();
uhf.stopInventory();
//.....
//.....
uhf.setAntennaPower(AntennaEnum.ANT1,30);
uhf.setEPCMode();
//.....

```

第六步: 断开所有设备连接

Step 6: Disconnect all devices in batch (release resources)

```
deviceManager.disconnectAll();
```

2.使用示例

2.Usage Examples

```

/**
 * Multi-device management test example
 * Demonstrates how to use the ManageMultipleDevice class to add, connect, set callbacks, and perform
 * tag inventory operations for multiple Ethernet-connected UHF devices
 */
public void Test(){
    // 1. Initialize the multi-device manager (requires application context)
    RFIDwithUHFA4Host.ManageMultipleDevice deviceManager = new RFIDwithUHFA4Host.
    ManageMultipleDevice(context);

    // 2. Add Ethernet devices to the manager (IP address + port number)
    // Device 1: IP=192.168.99.202, Port=8888
    deviceManager.addDevice("192.168.99.202", 8888);
    // Device 2: IP=192.168.99.203, Port=8888
    deviceManager.addDevice("192.168.99.203", 8888);

    // 3. Set a unified connection status callback for all devices
    // Used to monitor device connection status changes (connected, disconnected, connecting)
    deviceManager.setConnectionStatusCallbackAll(new ConnectionStatusCallback() {
        @Override
        public void getStatus(ConnectionStatus status, Object device) {
            // The "device" parameter is the device's IP address (String type), used to distinguish
different devices
            if (device != null) {
                // Get the IP of the device with status change
                String ip = (String) device;
            }
            // Execute logic based on connection status
            if (status == ConnectionStatus.CONNECTED) {
                // Device is connected
            } else if (status == ConnectionStatus.DISCONNECTED) {
                // Device is disconnected
            } else if (status == ConnectionStatus.CONNECTING) {
                // Device is connecting
            }
        }
    });

    // 4. Connect all added devices in batch
    deviceManager.connectAll();

    // 5. Set a unified tag inventory callback for all devices
    // Used to receive tag information inventoried by devices
    deviceManager.setInventoryCallbackAll(new IUHFInventoryCallback(){

```

```

@Override
public void callback(UHFTAGInfo uhftagInfo) {
    // Get the device IP and tag EPC from the tag information
    String ip = uhftagInfo.getIp(); // Identify which device the tag comes from
    String epc = uhftagInfo.getEPC(); // EPC code of the tag
    // Process tag data (e.g., display, store, etc.)
}
});

// 6. Control specific devices to start tag inventory separately
// Start tag inventory for Device 1
deviceManager.getDevice("192.168.99.202").startInventoryTag();
// Start tag inventory for Device 2
deviceManager.getDevice("192.168.99.203").startInventoryTag();

// .....
// (Other business logic can be executed here, e.g., waiting for inventory completion)
// .....

// 7. Control specific devices to stop tag inventory separately
// Stop tag inventory for Device 1
deviceManager.getDevice("192.168.99.202").stopInventory();
// Stop tag inventory for Device 2
deviceManager.getDevice("192.168.99.203").stopInventory();

// .....
// (Other business logic can be executed here, e.g., data processing)
// .....

// 8. Disconnect all devices in batch (release resources)
deviceManager.disconnectAll();
}

```

17.1 addDevice(String ip, int port)

项目 Item	说明 Description
原型 Prototype	synchronized boolean addDevice(String ip, int port)
描述 Description	添加网口连接的设备至设备列表。若设备已存在或IP地址无效，将抛出运行时异常 Add a device with Ethernet connection to the device list. Throws a runtime exception if the device already exists or the IP address is invalid
参数 Parameters	ip: 设备的IP地址 IP address of the device port: 设备的端口号 Port number of the device
返回 Return	boolean: true 表示添加成功，false 表示添加失败 true: success, false: failure
异常 Exception	RuntimeException: 设备已存在 (device already exists) 或IP地址无效 (Invalid ip address) 时抛出 Thrown when the device already exists or the IP address is invalid

17.2 getDevice(String ip)

项目 Item	说明 Description
原型 Prototype	synchronized IUHFA4 getDevice(String ip)
描述 Description	根据IP地址从设备列表中获取对应的设备实例 Get the corresponding device instance from the device list by IP address
参数 Parameters	ip: 目标设备的IP地址 IP address of the target device
返回 Return	IUHFA4: 设备实例（若不存在则返回null）。函数详细说明可参考 <code>com.rscja.deviceapi.RFIDwithUHFA4Host</code> ，该类是IUHFA4的实现类。 IUHFA4: Device instance (returns null if not exists). For detailed function description, refer to <code>com.rscja.deviceapi.RFIDwithUHFA4Host</code> , which is an implementation class of IUHFA4.

17.3 removeDevice(String ip)

项目 Item	说明 Description
原型 Prototype	synchronized boolean removeDevice(String ip)
描述 Description	从设备列表中移除指定IP地址的设备，并释放该设备的资源 Remove the device with the specified IP address from the device list and release its resources
参数 Parameters	ip: 待移除设备的IP地址 IP address of the device to be removed
返回 Return	boolean: true 表示移除成功, false 表示设备不存在 true: removal successful, false: device does not exist

17.4 removeAllDevice()

项目 Item	说明 Description
原型 Prototype	synchronized boolean removeAllDevice()
描述 Description	移除设备列表中的所有设备，并释放所有设备的资源 Remove all devices from the device list and release resources of all devices
参数 Parameters	无 N/A
返回 Return	boolean: true 表示移除成功, false 表示移除失败 true: all removed successfully, false: removal failed

17.5 isDeviceExist(String address)

项目 Item	说明 Description
原型 Prototype	synchronized boolean isDeviceExist(String address)

项目 Item	说明 Description
描述 Description	检查指定地址的设备是否已在设备列表中存在 Check if the device with the specified address exists in the device list
参数 Parameters	address: 设备的IP地址 (唯一标识) IP address of the device (unique identifier)
返回 Return	boolean: true 表示设备存在, false 表示设备不存在 true: device exists, false: device does not exist

17.6 `getDeviceCount()`

项目 Item	说明 Description
原型 Prototype	synchronized int getDeviceCount()
描述 Description	获取当前设备列表中的设备总数 Get the total number of devices in the current device list
参数 Parameters	无 N/A
返回 Return	int: 设备总数 Total number of devices

17.7 `connectAll()`

项目 Item	说明 Description
原型 Prototype	synchronized void connectAll()
描述 Description	批量建立设备列表中所有设备的连接 (调用各设备的 init 方法) Establish connections for all devices in the device list in batches (calls the init method of each device)
参数 Parameters	无 N/A
返回 Return	无 N/A

17.8 `disconnectAll()`

项目 Item	说明 Description
原型 Prototype	synchronized void disconnectAll()
描述 Description	批量断开设备列表中所有设备的连接, 并释放资源 (调用各设备的 free 方法) Disconnect all devices in the device list in batches and release resources (calls the free method of each device)
参数 Parameters	无 N/A

项目 Item	说明 Description
返回 Return	无 N/A

17.9 setConnectionStatusCallbackAll(ConnectionStatusCallback btStatusCallback)

项目 Item	说明 Description
原型 Prototype	synchronized void setConnectionStatusCallbackAll(ConnectionStatusCallback btStatusCallback)
描述 Description	为设备列表中所有设备统一设置连接状态回调 Uniformly set the connection status callback for all devices in the device list
参数 Parameters	btStatusCallback: 连接状态回调接口实例 Instance of connection status callback interface
返回 Return	无 N/A

17.10 setInventoryCallbackAll(IUHFIInventoryCallback btStatusCallback)

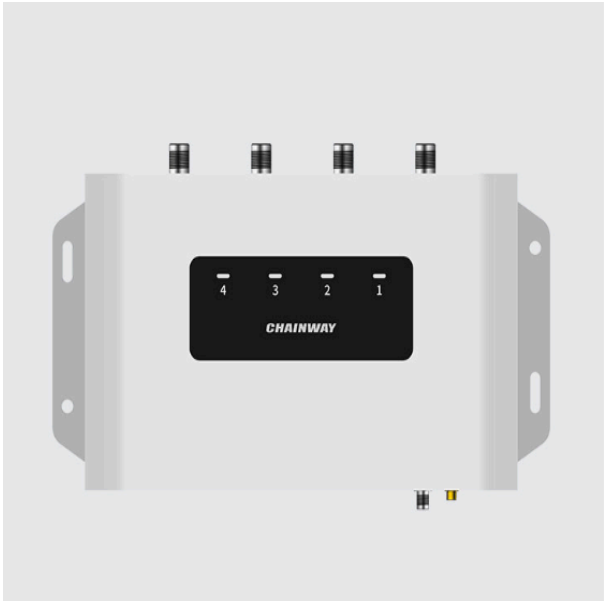
项目 Item	说明 Description
原型 Prototype	synchronized void setInventoryCallbackAll(IUHFIInventoryCallback btStatusCallback)
描述 Description	为设备列表中所有设备统一设置盘点回调 Uniformly set the inventory callback for all devices in the device list
参数 Parameters	btStatusCallback: 盘点回调接口实例 Instance of inventory callback interface
返回 Return	无 N/A

三、机型

III. Device



Reader_A



Reader_B