

分立器读写器 SDK 接口说明文档(C++)

(libDiscrete.dll 库)

编制: _____ 日期: _____

复核: _____ 日期: _____

批准: _____ 日期: _____

深圳市捷通科技有限公司研发部

(2021-03-30 更新)

附修订记录

版本	修订内容	修订人/修订时间
V3.01		
V3.02	1.更新频率读取和设置(修改欧标和韩标)	2021-03-30

目录

一、引言.....	7
1.1、目的.....	7
1.2、适用范围.....	7
1.3、定义.....	7
1.4、版权说明.....	7
二、概述.....	8
2.1、部署说明.....	8
2.2、读写器基本硬件框架.....	8
2.3、SDK API 总体流程.....	8
2.4、读写器基本流程.....	11
三、开发包文件说明.....	13
3.1、常用的定义.....	13
四、SDK 接口说明.....	14
五、软件连接与断开连接.....	15
5.1、快速上手.....	16
5.2、功能：连接设备(串口或 IP(TCP Client)).....	16
5.3、功能：断开连接设备.....	17
5.4、功能：TCP Server 连接设备.....	17
5.5、功能：启动监听(TCP Server).....	17
5.6、功能：停止监听(TCP Server).....	18
六、基本操作.....	18
6.1、功能：连续读卡.....	19
6.2、功能：停止连续读卡.....	19
6.3、功能：查询版本号.....	19
6.4、功能：单次读卡.....	20
七、标签操作.....	20
7.1、快写标签.....	20
7.1.1、功能：EPC 区快写标签.....	20

7.2、标签读写	21
7.2.1、功能：指定区域读标签	21
7.2.2、功能：指定区域写标签	22
7.3、标签操作区域加锁和解锁	22
7.3.1、功能：标签指定区域加锁	23
7.3.2、功能：标签指定区域解锁	23
7.4、标签销毁(慎用).....	23
7.4.1、功能：标签销毁	24
7.5、初始化标签	24
7.5.1、功能：标签初始化	24
7.6、标签授权	25
7.6.1、功能：标签授权	25
7.6.2、功能：查询标签授权码	25
7.6.3、功能：更新标签授权码	26
八、读写器参数配置	26
8.1、工作模式	26
8.1.1、功能：查询工作模式	27
8.1.2、功能：更新工作模式	28
8.1.3、功能：查询相邻判别(是否启用).....	28
8.1.4、功能：更新相邻判别(是否启用).....	28
8.1.5、功能：查询相邻判别时间	29
8.1.6、功能：更新相邻判别时间	29
8.1.7、功能：查询定时间隔时间 (定时模式).....	29
8.1.8、功能：更新定时间隔时间 (定时模式).....	30
8.1.9、功能：查询定时间隔速度 (定时模式).....	30
8.1.10、功能：更新定时间隔速度 (定时模式).....	30
8.1.11、功能：查询触发延迟时间(触发模式).....	31
8.1.12、功能：更新触发延迟时间(触发模式).....	31
8.1.13、功能：查询触发延迟开关(触发模式).....	31
8.1.14、功能：更新触发延迟开关(触发模式).....	32

8.2、通讯方式	32
8.2.1、功能：查询通讯方式	33
8.2.2、功能：更新通讯方式	33
8.2.3、查询脉冲宽度(韦根).....	33
8.2.4、更新脉冲宽度(韦根).....	34
8.2.5、查询脉冲周期(韦根).....	34
8.2.6、更新脉冲周期(韦根).....	34
8.2.7、查询韦根协议 (韦根).....	35
8.2.8、更新韦根协议 (韦根).....	35
8.2.9、查询韦根值 (韦根).....	35
8.2.10、更新韦根值 (韦根).....	36
8.2.11、功能：查询串口波特率	36
8.2.11、功能：更新串口波特率	37
8.3、天线配置	37
8.3.2、功能：更新天线参数配置	38
8.4、设备功率	38
8.4.1、功能：查询设备功率	38
8.4.2、功能：更新设备功率	39
8.5、设备频率	39
8.5.1、功能：查询设备频率	40
8.5.2、功能：更新频率	41
8.5.1、频率参数表	41
8.6、读卡模式(单标签和多标签).....	42
8.6.1、功能：查询读卡模式	43
8.6.2、功能：更新读卡模式	43
8.7、设备编号	43
8.7.1、功能：查询设备编号	44
8.7.2、功能：更新设备编号	44
九、其它操作	45
9.1、蜂鸣器配置(声音).....	45

9.1.1、功能：查询蜂鸣器状态	45
9.1.2、功能：更新蜂鸣器状态	45
9.2、继电器配置(GPIO)	46
9.2.1、功能：查询继电器配置(暂无).....	46
9.2.2、功能：更新继电器模式(GPIO 主动和被动)	46
9.2.3、功能：更新继电器主动闭合时间(GPIO 主动触发).....	47
9.2.4、功能：更新继电器两路触发 (GPIO 被动触发)	47
9.3、指定标签读和写区域	47
9.3.1、功能：查询标签读和写区域	48
9.3.2、功能：更新标签读和写区域	48
9.4、标签数据输出格式	48
9.4.1、功能：查询标签数据输出格式(仅限双 USB 设备)	49
9.4.2、功能：更新标签数据输出格式(仅限双 USB 设备)	49
9.4.3、功能：查询标签数据输出格式(非双 USB 设备)	50
9.4.4、功能：更新标签数据输出格式(非双 USB 设备)	50

一、引言

1.1、目的

使用本开发包，用户可以高效、方便地完成应用软件开发。

1.2、适用范围

本文档适用于捷通分立器系列产品型号设备。

本文档面向的读者对象为：

系统设计师、

软件工程师、

测试工程师、

技术支持

1.3、定义

定义	名称	描述
分立器读写器	Reader	读写器接收和发送的电子信号
标签	Tag	标签定时发送电子信号

1.4、版权说明

文档的所有内容，包括文字、图片均为原创。对未经许可，擅自使用在商业用者，本公司保留追究其法律责任的权利。未经授权，使用者不得擅自添加、修改、删除本文档内容，不得以网络、光盘等方式进行传播。如若违返，后果自负。

二、概述

欢迎您选用深圳捷通科技有限公司提供的(分立器)读写器，为方便用户开发应用程序，本公司特提供软件开发包，以动态链接库形式提供给用户使用。

开发包支持VC、VB、C#、VB.NET、C++ Builder Java 和Delphi 等多种语言的开发。

2.1、部署说明

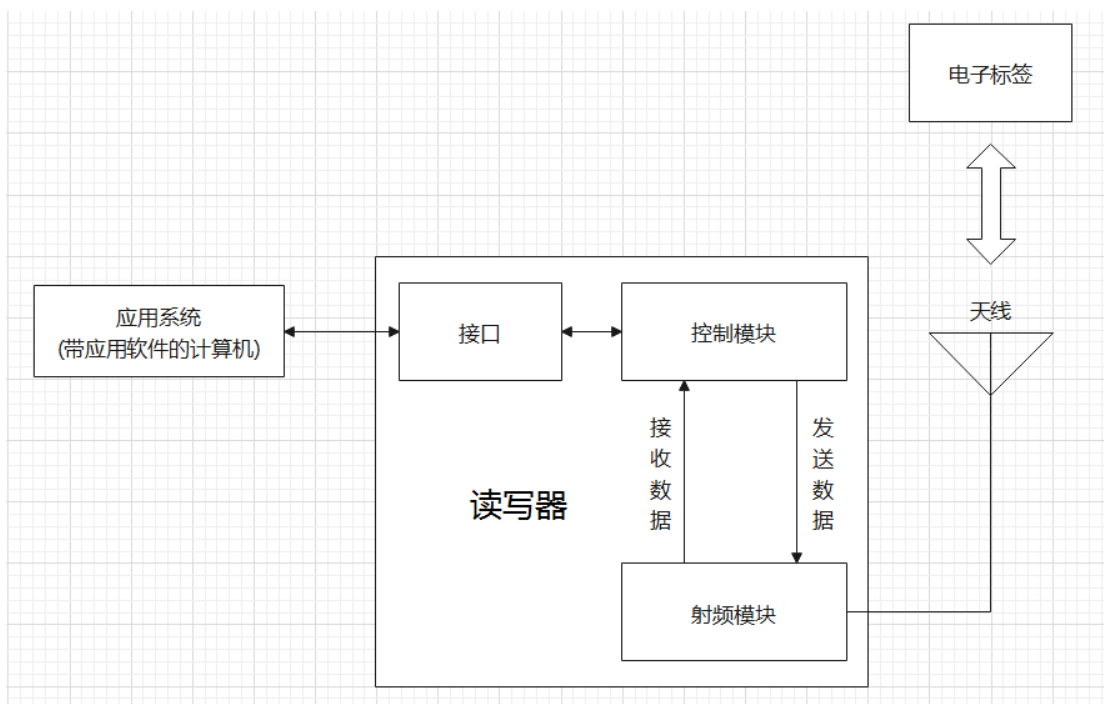
开发包使用VC++语言开发完成，共3个文件（LibDiscrete.h、LibDiscrete.dll），分别是头文件、导出库文件和动态链接库(32位和64位)。

用户在使用时，需要根据实际存放的路径引入头文件和导出库文件，动态链接库文件和最终的可执行文件放在同一目录即可。

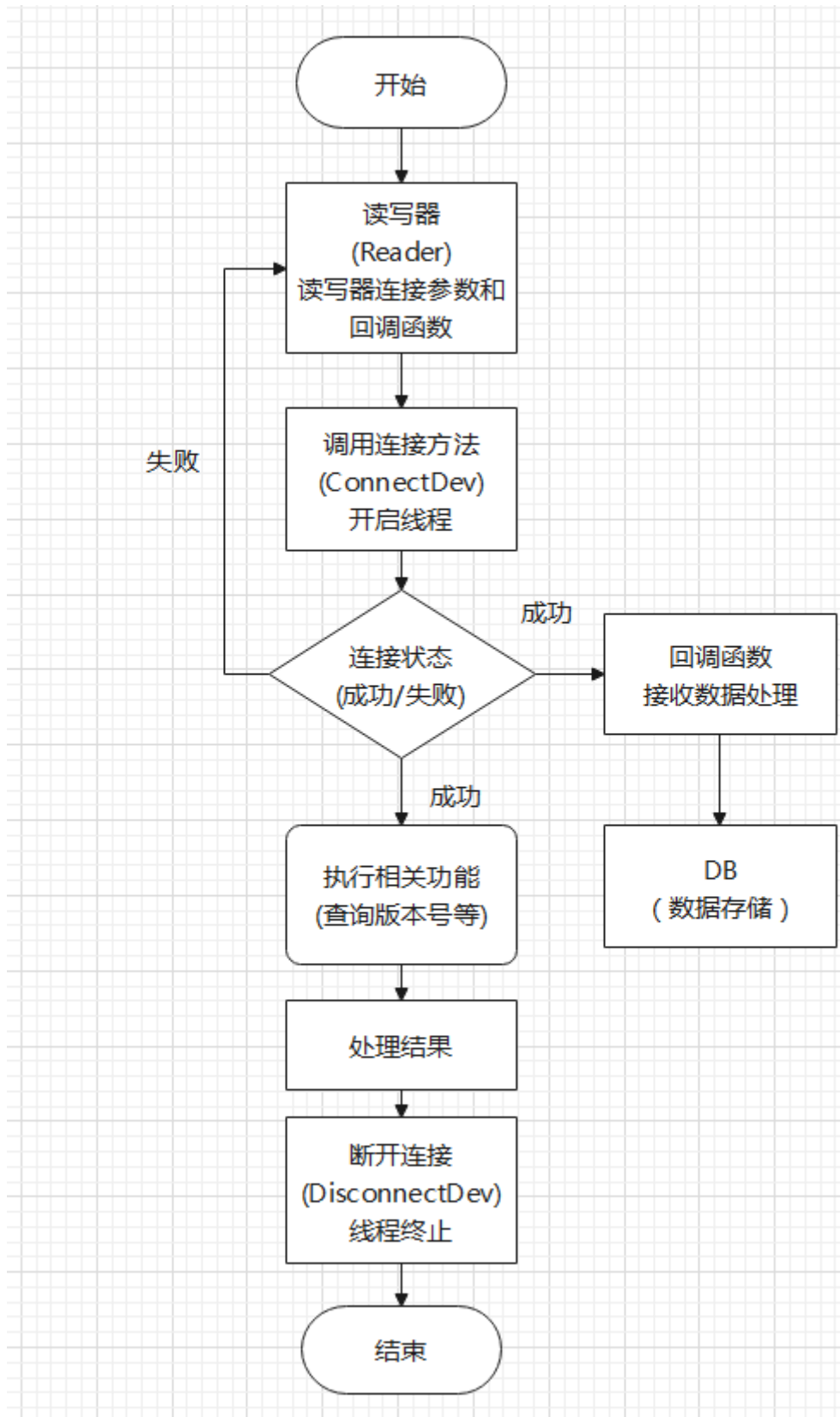
适用操作系统：Windows

2.2、读写器基本硬件框架

发送通道、接收通道、载波产生器、时钟产生电路、天线



2.3、SDK API 总体流程



上图中，读写器建立连接后，调用 SDK 接口发送指令方式：

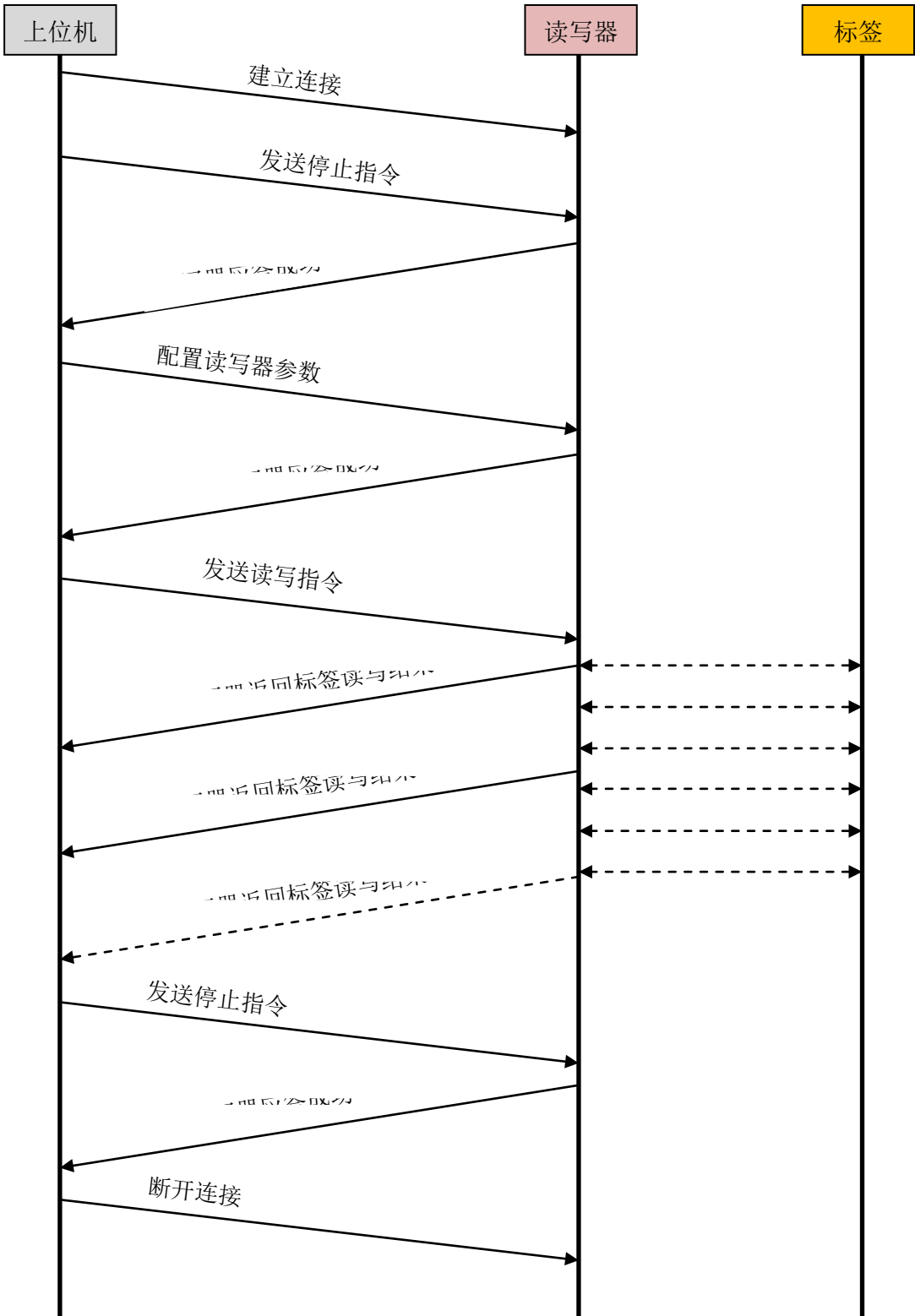
SDK 接口向读写器发送消息后，回应结果同步等待，等待时间不可指定，默认 1000 毫秒。SDK 接口内部与设备的数据处理为异步，可调用 SDK 接口发送指令后，上位机在不同的线程无需等待，读写器执行。

对于读写器读主动上传的消息，通过回调函数，异步接收读写器获取到的信息；

回调信息如以下几种：

标签扫描数据、(后续增加)GPIO 触发数据、心跳、红外触发信号等。

2.4、读写器基本流程



上位机通过指定连接方式（例如：串口，网口 **TCP Cleint** 或 **TCP Server** 等连接方式）向读写器发起连接，连接成功后，上位机无需向读写器发送任何指令，程序内部已做处理，连接上设备后内部已创建了线程优点主要有：

读写器连接后处理数据层进入待命状态，可随时响应发送的指令；

向读写器发送指令后的响应，来确认读写器当前状态是否正常可用。

如果需要重新连接，建议主动（断开连接）释放之前的线程，已免出现连接或其它问题。(在已经连接成功的前提下)

三、开发包文件说明

开发可能有更新与文档有差异。

主要有以下几种文件类型：Demo、Doc、SDK、SourceCode

Demo：存放带 UI 或是执行文件(例如：exe 等)。

Doc：存放文档，(例如：设备的通讯协议，SDK 接口文档等)。

SDK：存放 dll .h 等文件。

SourceCode：存放测试用例和源码。

3.1、常用的定义

```
typedef struct OutputInfoStruct
{
    BYTE id_data[200];           //id 编号
    BYTE data_len;               //id 数据长度
    BYTE host[32];               //通信方式(例如：COM1 or 192.168.1.200)
    BYTE host_len;               //通信方式名称的长度
    BYTE baud_rate_or_port [32]; //波特率(串口)或端口(IP)
    char antNo[10];              //天线号
    char device_id[10];           //设备号
    bool connect_exception;       //网络异常
}OutputInfoStruct;

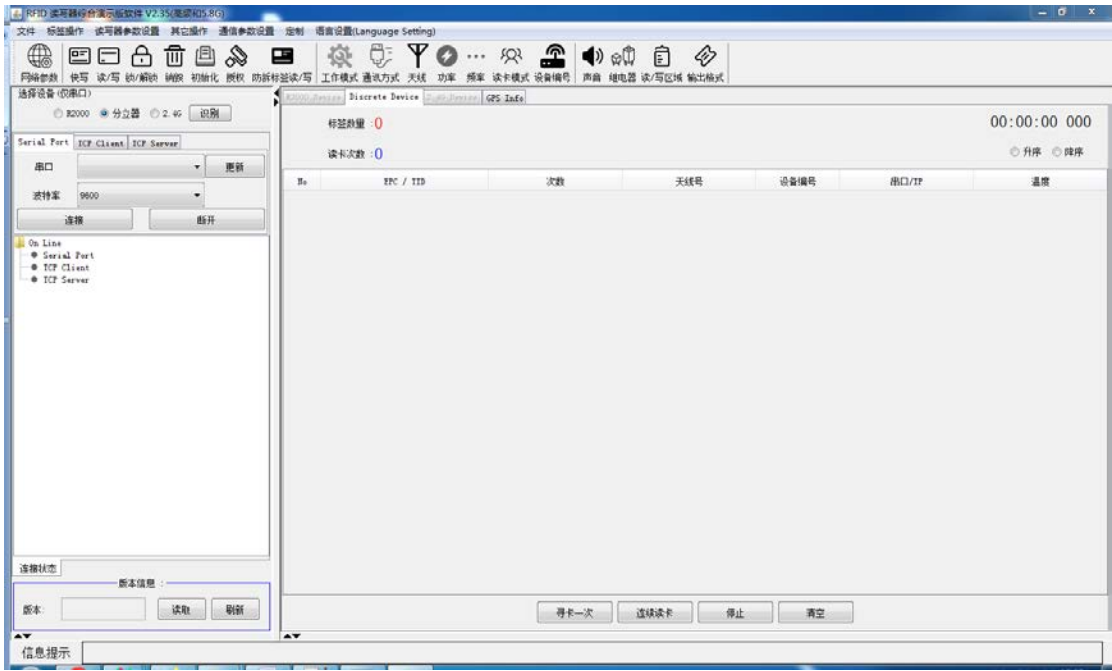
typedef struct SocketInfoStruct
{
    BYTE IP[32];                 //通信方式(例如：COM1 or 192.168.1.200)
    BYTE host_len;               //通信方式名称的长度
    unsigned short port;         //端口
    int socket;                  //套字节
    struct sockaddr_in srvaddr;
}SocketInfoStruct;

//返回值
#define R_SUCCESS                (1)           // 正确结果
#define R_FAILURE                (0)           // 错误结果
```

四、SDK 接口说明

SDK 接口目前支持串口和网口(TCP Client)，支持网口 TCP Server(监听端口)

文档中的 UI 参考图片，均来自 Java 综合版 Demo 软件。



五、软件连接与断开连接

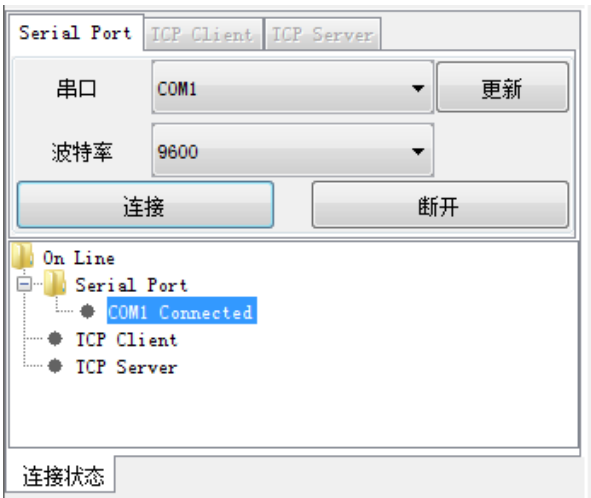


图 5.1.1、串口连接示例

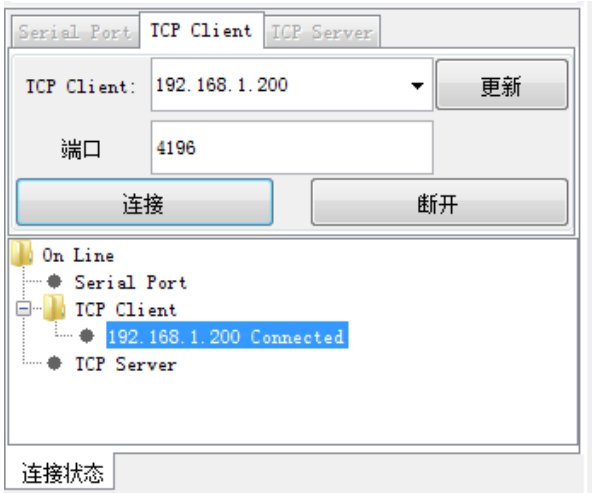


图 5.1.2、网口 TCP Clie 连接示例



图 5.1.3、网口 TCP Server 连接示例

5.1、快速上手

输出参数结构体

```
typedef struct OutputInfoStruct
{
    BYTE id_data[200];           //id 编号
    BYTE data_len;               //id 数据长度
    BYTE host[32];               //通信方式(例如： COM1 or 192.168.1.200)
    BYTE host_len;               //通信方式名称的长度
    BYTE baud_rate_or_port [32]; //波特率(串口)或端口(IP)
    char antNo[10];              //天线号
    char device_id[10];          //设备号
    bool connect_exception;      //网络异常(异常提示详情参考示例)
}OutputInfoStruct;

typedef void (CALLBACK * FUN_DATA_CB) (const HANDLE hDev,OutputInfoStruct * outputInfo);
```

5.2、功能：连接设备(串口或 IP(TCP Client))

功能说明：

```
int ConnectDev(HANDLE& hDev, FUN_DATA_CB f, char * host, int baudrateOrPort);
```

函数名称		返回值类型	说明		
ConnectDev		int	连接设备（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE &	hDev		句柄
2	参数 2	FUN_DATA_CB	f		回调函数
3	参数 3	char *	host	例：“COM1” 例：“192. 168. 1. 200”	串口 或IP
4	参数 4	int	baudrateOrPort	波特率：“115200” 端口：“8000”	波特率 或端口

5.3、功能：断开连接设备

功能说明：

```
int DisconnectDev(HANDLE hDev);
```

函数名称		返回值类型	说明		
DisconnectDev		int	断开连续（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

5.4、功能：TCP Server 连接设备

功能说明：需启动监听后，在回调函数中调用

```
int TCPServerConnect (HANDLE& hDev,
                      int socket,FUN_DATA_CB f, char * host, unsigned short port);
```

函数名称		返回值类型	说明		
ConnectDev		int	连接设备（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE &	hDev		句柄
2	参数 1	int	socket		监听产生的 socket
3	参数 2	FUN_DATA_CB	f		回调函数
4	参数 3	char *	host	例:"192.168.1.200"	IP
5	参数 4	unsigned short	port		监听产生的端口

5.5、功能：启动监听(TCP Server)

功能说明：

```
int StartListening(HANDLE & hServerSocket, FUN_SOCKET_CB f, unsigned short port);
```

函数名称		返回值类型	说明		
StartListening		int	启动监听 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hServerSocket		句柄 (Server Socket)
2	参数 2	FUN_SOCKE_CB	f		回调函数
3	参数 3	unsigned short	port		端口 (0~65535)

```
typedef struct SocketInfoStruct
{
    BYTE IP[32];                //通信方式(例如: COM1 or 192.168.1.200)
    BYTE host_len;              //通信方式名称的长度
    unsigned short port;        //端口
    int socket;                 //套字节
    struct sockaddr_in srvaddr;
}SocketInfoStruct;

typedef void (CALLBACK * FUN_SOCKET_CB)(SocketInfoStruct * outputInfo);
```

5.6、功能：停止监听(TCP Server)

功能说明：

```
int StopListening (HANDLE hServerSocket);
```

函数名称		返回值类型	说明		
StopListening		int	停止监听 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hServerSocket		句柄 (Server Socket)

六、基本操作

6.1、功能：连续读卡

功能说明：

```
int BeginInv(HANDLE hDev);
```

函数名称		返回值类型	说明		
BeginInv		int	连续读卡（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

6.2、功能：停止连续读卡

功能说明：

```
int StopInv(HANDLE hDev);
```

函数名称		返回值类型	说明		
StopInv		int	停止连续读卡（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

6.3、功能：查询版本号

功能说明：

```
int QueryVersion(HANDLE hDev, char* pVer);
```

函数名称		返回值类型	说明		
QueryVersion		int	查询版本号（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	char *	pVer		存放版本号信息

6.4、功能：单次读卡

功能说明：

```
int ReadSingleTag (HANDLE hDev,  
    BYTE * data, BYTE & data_len, BYTE & device_no, BYTE & antenna_no, BYTE * IP_or_comm);
```

函数名称		返回值类型	说明		
ReadSingleTag		int	单次读卡 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE *	data		数据 (EPC or TID)
3	参数 3	BYTE &	data_len		数据长度 (EPC or TID)
4	参数 4	BYTE &	device_no		设备号
5	参数 5	BYTE &	antenna_no		天线号
6	参数 6	BYTE *	IP_or_comm		{"COM1"} or {"192. 168. 1. 200"}

七、标签操作

7.1、快写标签

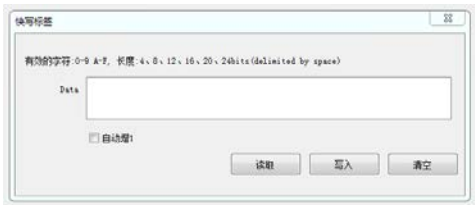


图 7. 1、快写标签功能参考

7.1.1、功能：EPC 区快写标签

功能说明：

```
int FastWriteTag (HANDLE hDev, BYTE data [], BYTE length);
```

函数名称	返回值类型	说明
------	-------	----

FastWriteTag		int	EPC 区快写标签(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE []	data	有效字符 (0-9, A-F)	写入数据
3	参数 3	BYTE	length	(长度:4、8、12、16、 20、24 BYTE、) 注意: 2个字符为1个BYTE	数据长度

7.2、标签读写

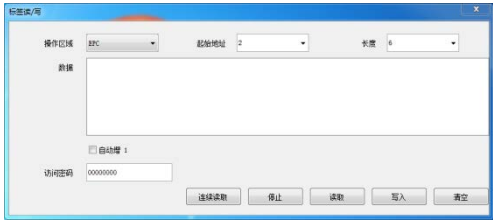


图 7.2、标签读写功能参考

7.2.1、功能：指定区域读标签

功能说明：

```
int ReadTagData (HANDLE hDev,  
                BYTE bank, BYTE begin, BYTE length, BYTE * data, BYTE & data_len);
```

函数名称		返回值类型		说明	
ReadTagData		int		指定区域读标签(成功 1 失败 0)	
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	bank	0. Reserver(保留区) 1. EPC 2. TID 3. USER(用户区)	操作区域
3	参数 3	BYTE	begin	与操作区域有关	起始地址
4	参数 4	BYTE	length	与操作区域有关	长度

5	参数 5	BYTE *	data	注意： 2个字符为1个BYTE	数据
6	参数 6	BYTE &	data_len	数据长度 = (长度 * 2)	数据长度

7.2.2、功能：指定区域写标签

功能说明：

```
int WriteTagData (HANDLE hDev, BYTE bank, BYTE begin, BYTE length, BYTE data[]);
```

函数名称		返回值类型	说明		
WriteTagData		int	指定区域写标签 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	bank	0. Reserver (保留区) 1. EPC 2. TID (只读不可写) 3. USER (用户区)	操作区域
3	参数 3	BYTE	begin	与操作区域有关	起始地址
4	参数 4	BYTE	length	与操作区域有关	长度
5	参数 5	BYTE []	data	数据长度 = (长度 * 2) 注意： 2个字符为1个BYTE	数据

7.3、标签操作区域加锁和解锁

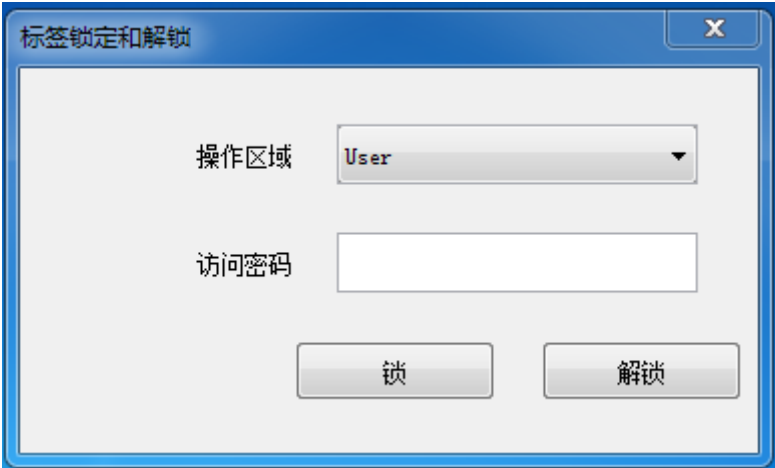


图 7.3、标签加锁和解锁功能参考

7.3.1、功能：标签指定区域加锁

功能说明：

```
int LockTag (HANDLE hDev, BYTE lock_type, BYTE password[]);
```

函数名称		返回值类型	说明		
LockTag		int	标签指定区域加锁（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	lock_type	0. USER 1. TID 2. EPC 3. 访问密码 4. 灭活密码 5. 全部	操作区域
3	参数 3	BYTE[]	password	密码长度(2个BYTE) 例: {0xFF, 0xFF}	密码

7.3.2、功能：标签指定区域解锁

功能说明：

```
int UnLockTag (HANDLE hDev, BYTE lock_type, BYTE password[]);
```

函数名称		返回值类型	说明		
UnLockTag		int	标签指定区域解锁（成功 1 失败 0）		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	lock_type	6. USER 7. TID 8. EPC 9. 访问密码 10. 灭活密码 11. 全部	操作区域
3	参数 3	BYTE[]	password	密码长度(2个BYTE) 例: {0xFF, 0xFF}	密码

7.4、标签销毁(慎用)

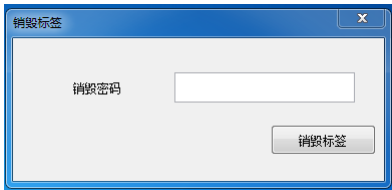


图 7.4、标签销毁功能参考

7.4.1、功能：标签销毁

功能说明：

```
int KillTag (HANDLE hDev, BYTE password[]);
```

函数名称		返回值类型	说明		
KillTag		int	标签销毁(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE[]	password	密码长度(2个BYTE) 例: {0xFF, 0xFF}	销毁密码

7.5、初始化标签

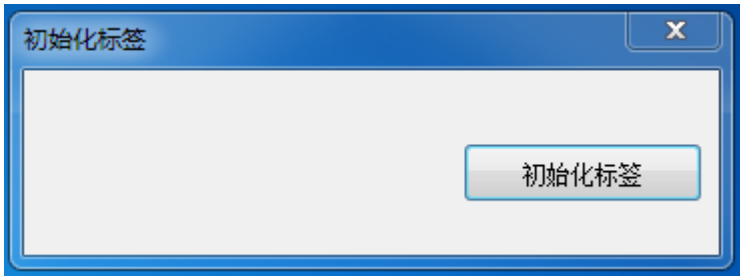


图 7.5、初始化标签功能参考

7.5.1、功能：标签初始化

功能说明：

```
int InitializeTag (HANDLE hDev);
```

函数名称		返回值类型	说明		
InitializeTag		int	标签初始化(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

7.6、标签授权

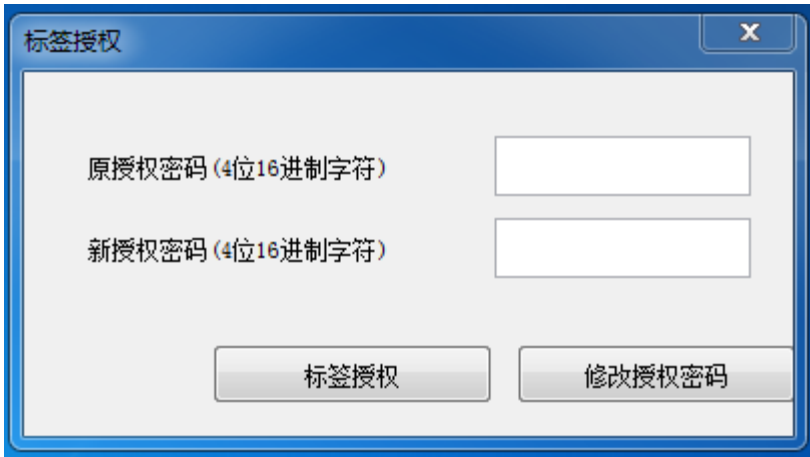


图 7.6、标签授权功能参考

7.6.1、功能：标签授权

功能说明：

```
int TagAuther (HANDLE hDev);
```

函数名称		返回值类型	说明		
TagAuther		int	标签授权 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

7.6.2、功能：查询标签授权码

功能说明：

```
int QueryAutherPwd (HANDLE hDev, BYTE * pwd, BYTE & pwd_len);
```

函数名称		返回值类型	说明		
QueryAutherPwd		int	查询标签授权码 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE *	pwd	例: {0xFF, 0xFF}	密码
3	参数 3	BYTE	pwd_len	密码长度 (2个BYTE)	密码长度

7.6.3、功能：更新标签授权码

功能说明：

```
int UpdateAutherPwd(HANDLE hDev, BYTE pwd[], BYTE pwd_len);
```

函数名称		返回值类型	说明		
UpdateAutherPwd		int	更新标签授权码(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE []	pwd	例: {0xFF, 0xFF}	密码
3	参数 3	BYTE	pwd_len	密码长度(2个BYTE)	密码长度

八、读写器参数配置

8.1、工作模式

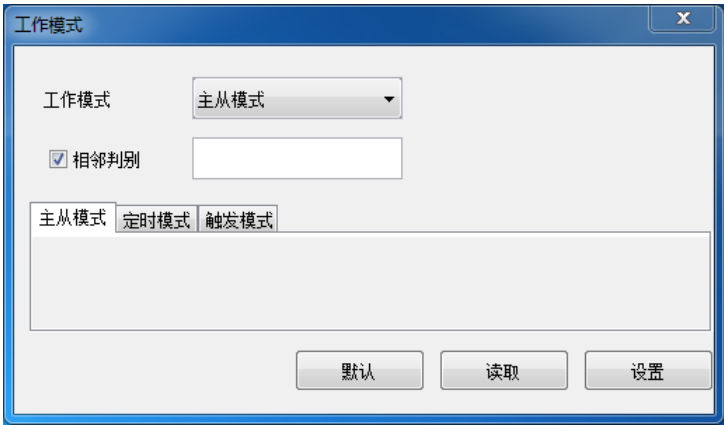


图 8.1.1、工作模式(主从模式) 功能参考

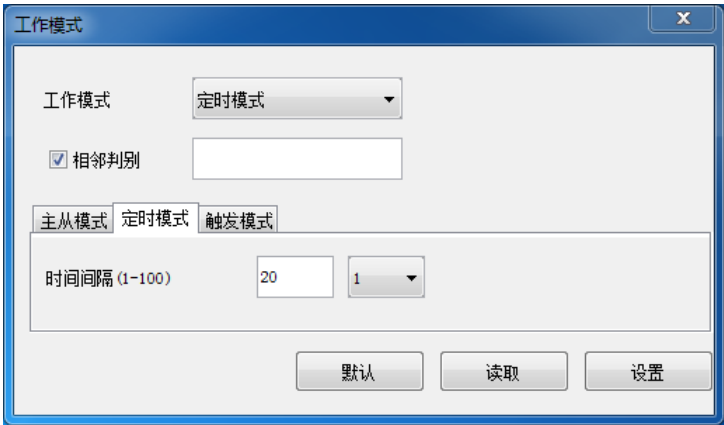


图 8.1.2、工作模式(定时模式) 功能参考

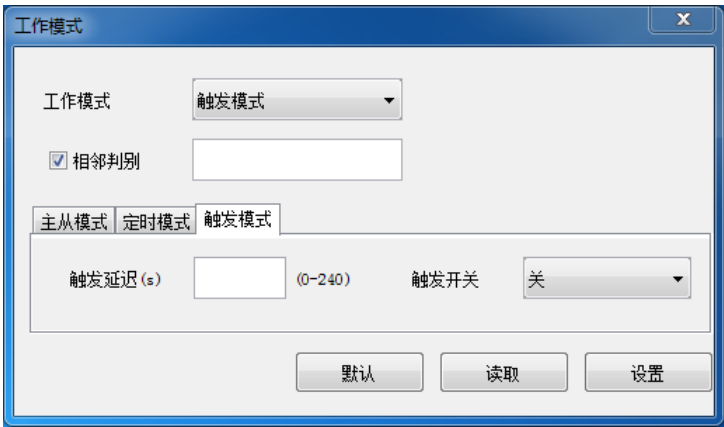


图 8.1.3、工作模式(触发模式) 功能参考

8.1.1、功能：查询工作模式

功能说明：

```
int QueryWorkMode(const HANDLE hDev, BYTE & workMode);
```

函数名称		返回值类型	说明		
QueryWorkMode		int	查询工作模式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	workMode	0:主从模式 1:定时模式 2:触发模式	工作模式

8.1.2、功能：更新工作模式

功能说明：

```
int UpdateWorkMode(const HANDLE hDev, BYTE workMode);
```

函数名称		返回值类型	说明		
UpdateWorkMode		int	更新工作模式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	workMode	0:主从模式 1:定时模式 2:触发模式	工作模式

8.1.3、功能：查询相邻判别(是否启用)

功能说明：

```
int QueryAdjacentDiscriminantStatus (HANDLE hDev, BYTE & status);
```

函数名称		返回值类型	说明		
QueryAdjacentDiscriminantStatus		int	查询相邻判别(启用) (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	status	0:不启用 1:启用	相邻判别 状态

8.1.4、功能：更新相邻判别(是否启用)

功能说明：

```
int UpdateAdjacentDiscriminantStatus (HANDLE hDev, BYTE status);
```

函数名称		返回值类型	说明		
UpdateAdjacentDiscriminantStatus		int	更新相邻判别(是否启用) (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

2	参数 2	BYTE	status	0:不启用 1:启用	相邻判别 状态
---	------	------	--------	---------------	------------

8.1.5、功能：查询相邻判别时间

功能说明：

```
int QueryAdjacentDiscriminantTime (HANDLE hDev, BYTE time);
```

函数名称		返回值类型	说明		
QueryAdjacentDiscriminantTime		int	查询相邻判别时间 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	time	0:取消相邻判别 0~240:单位(秒s)	相邻判别 时间

8.1.6、功能：更新相邻判别时间

功能说明：

```
int UpdateAdjacentDiscriminantTime (HANDLE hDev, BYTE time);
```

函数名称		返回值类型	说明		
UpdateAdjacentDiscriminantTime		int	更新相邻判别时间 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	time	0:取消相邻判别 0~240:单位(秒s)	相邻判别 时间

8.1.7、功能：查询定时间隔时间 (定时模式)

功能说明：

```
int QueryTimeInterval (HANDLE hDev, BYTE & time);
```

函数名称		返回值类型	说明		
QueryTimeInterval		int	查询定时间隔时间 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

2	参数 2	BYTE &	time	0:没有延迟 1~100:单位(秒s)	定时间隔 时间
---	------	--------	------	------------------------	------------

8.1.8、功能：更新定时间隔时间 (定时模式)

功能说明：

int UpdateTimeInterval (HANDLE hDev, BYTE time);

函数名称		返回值类型	说明		
UpdateTimeInterval		int	更新定时间隔时间(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	time	0:没有延迟 1~100:单位(秒s)	定时间隔 时间

8.1.9、功能：查询定时间隔速度 (定时模式)

功能说明：

int QueryTimeIntervalSpeed (HANDLE hDev, BYTE & time);

函数名称		返回值类型	说明		
QueryTimeIntervalSpeed		int	查询定时间隔速度(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	time	0:代表1 1:代表10(秒s)	定时间隔 速度

8.1.10、功能：更新定时间隔速度 (定时模式)

功能说明：

int UpdateTimeIntervalSpeed (HANDLE hDev, BYTE time);

函数名称		返回值类型	说明		
UpdateTimeIntervalSpeed		int	更新定时间隔速度(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述

1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	time	0:代表1 1:代表10(秒s)	定时间隔 速度

8.1.11、功能：查询触发延迟时间(触发模式)

功能说明：

```
int QueryTriggerDelayTime (HANDLE hDev, BYTE & time);
```

函数名称		返回值类型	说明		
QueryTriggerDelayTime		int	查询触发延迟时间(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	time	0:没有延迟 0~240:单位(秒s)	

8.1.12、功能：更新触发延迟时间(触发模式)

功能说明：

```
int UpdateTriggerDelayTime (HANDLE hDev, BYTE time);
```

函数名称		返回值类型	说明		
UpdateTriggerDelayTime		int	更新触发延迟时间(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	time	0:没有延迟 0~240:单位(秒s)	

8.1.13、功能：查询触发延迟开关(触发模式)

功能说明：

```
int QueryTriggerSwitch (HANDLE hDev, BYTE & status);
```

函数名称		返回值类型	说明		
QueryTriggerSwitch		int	查询触发延迟开关(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述

1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	status	0:关 1:高电平	触发开关 状态

8.1.14、功能：更新触发延迟开关(触发模式)

功能说明：

```
int UpdateTriggerSwitch (HANDLE hDev, BYTE & status);
```

函数名称		返回值类型	说明		
UpdateTriggerSwitch		int	更新触发延迟开关(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	status	0:关 1:高电平	触发开关 状态

8.2、通讯方式

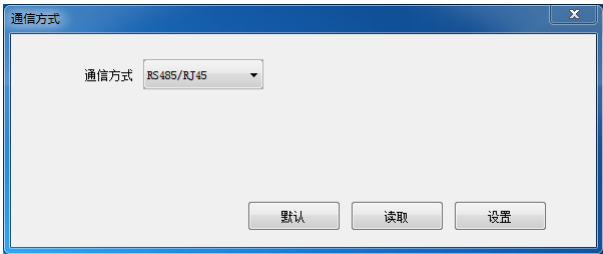


图 8.2.1、通讯方式(RS485/RJ45)功能参考

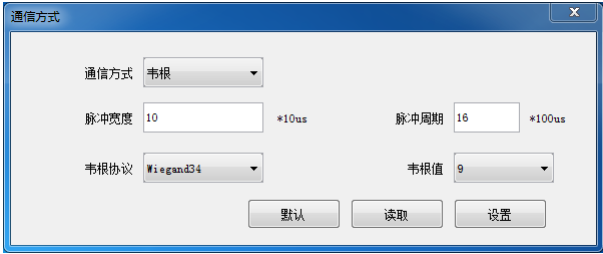


图 8.2.2、通讯方式(韦根)功能参考

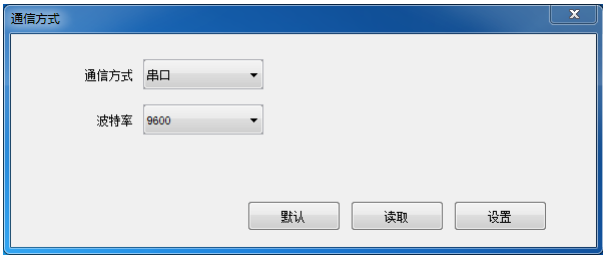


图 8.2.2、通讯方式(串口)功能参考

8.2.1、功能：查询通讯方式

功能说明：

```
int QueryOutputMode (HANDLE hDev, BYTE & output_mode);
```

函数名称		返回值类型	说明		
QueryTriggerSwitch		int	查询通讯方式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	output_mode	0:RS485/RJ45 1:韦根 2:串口	

8.2.2、功能：更新通讯方式

功能说明：

```
int UpdateOutputMode (HANDLE hDev, BYTE output_mode);
```

函数名称		返回值类型	说明		
UpdateOutputMode		int	更新通讯方式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	output_mode	0:RS485/RJ45 1:韦根 2:串口	

8.2.3、查询脉冲宽度(韦根)

功能说明：

```
int QueryWiegandPulseWidth (HANDLE hDev, BYTE & wiegand_pulse_width);
```

函数名称		返回值类型	说明		
QueryWiegandPulseWidth		int	查询韦根脉冲宽度(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	wiegand_pulse_width	1~255	脉冲宽度

8.2.4、更新脉冲宽度(韦根)

功能说明:

```
int UpdateWiegandPulseWidth (HANDLE hDev, BYTE wiegand_pulse_width);
```

函数名称		返回值类型	说明		
UpdateWiegandPulseWidth		int	更新韦根脉冲宽度(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	wiegand_pulse_width	1~255	脉冲宽度

8.2.5、查询脉冲周期(韦根)

功能说明:

```
int QueryWiegandPulseCycle (HANDLE hDev, BYTE & wiegand_pulse_cycle);
```

函数名称		返回值类型	说明		
QueryWiegandPulseCycle		int	查询脉冲周期(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	wiegand_pulse_cycle	1~255(*100us)	脉冲周期

8.2.6、更新脉冲周期(韦根)

功能说明:

```
int UpdateWiegandPulseCycle (HANDLE hDev, BYTE wiegand_pulse_cycle);
```

函数名称		返回值类型	说明		
UpdateWiegandPulseCycle		int	更新脉冲周期 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	wiegand_pulse_cycle	1~255(*100us)	脉冲周期

8.2.7、查询韦根协议 (韦根)

功能说明:

int QueryWiegandProto (HANDLE hDev, BYTE & wiegand_proto);

函数名称		返回值类型	说明		
QueryWiegandProto		int	查询脉冲周期 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	wiegand_proto	0:Wiegand26 1:Wiegand34 2:Wiegand32 3:Wiegand66	韦根协议

8.2.8、更新韦根协议 (韦根)

功能说明:

int UpdateWiegandProto (HANDLE hDev, BYTE wiegand_proto);

函数名称		返回值类型	说明		
UpdateWiegandProto		int	更新韦根协议 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	wiegand_proto	0:Wiegand26 1:Wiegand34 2:Wiegand32 3:Wiegand66	韦根协议

8.2.9、查询韦根值 (韦根)

功能说明:

```
int QueryWiegandValue (HANDLE hDev, BYTE & wiegand_value);
```

函数名称		返回值类型	说明		
QueryWiegandValue		int	查询韦根值(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	wiegand_proto	0~9	韦根值

8.2.10、更新韦根值 (韦根)

功能说明:

```
int UpdateWiegandValue (HANDLE hDev, BYTE wiegand_value);
```

函数名称		返回值类型	说明		
UpdateWiegandValue		int	查询韦根值(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	wiegand_proto	0~9	韦根值

8.2.11、功能：查询串口波特率

功能说明:

```
int QueryBaudRate (HANDLE hDev, BYTE & baud_rate);
```

函数名称		返回值类型	说明		
QueryBaudRate		int	查询串口波特率(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	baud_rate	0: 9600 1: 19200 2: 38400 3: 57600 4: 115200	

8.2.11、功能：更新串口波特率

功能说明：

```
int UpdateBaudRate (HANDLE hDev, BYTE baud_rate);
```

函数名称		返回值类型	说明		
UpdateBaudRate		int	更新串口波特率 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	baud_rate	0: 9600 1: 19200 2: 38400 3: 57600 4: 115200	

8.3、天线配置



图 8.3、天线配置功能参考

8.3.1、功能：查询天线参数配置

功能说明：

```
int QueryAntenna (HANDLE hDev, bool * antenna, BYTE & antenna_len);
```

函数名称		返回值类型	说明		
QueryAntenna		int	查询天线参数配置 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	bool *	antenna	bool [8]	
3	参数 3	BYTE	antenna_len	8	

8.3.2、功能：更新天线参数配置

功能说明：

```
int UpdateAntenna (HANDLE hDev, bool antenna[]);
```

函数名称		返回值类型	说明		
UpdateAntenna		int	更新天线参数配置(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	bool []	antenna	bool [8]	
3	参数 3	BYTE	antenna_len	8	

8.4、设备功率

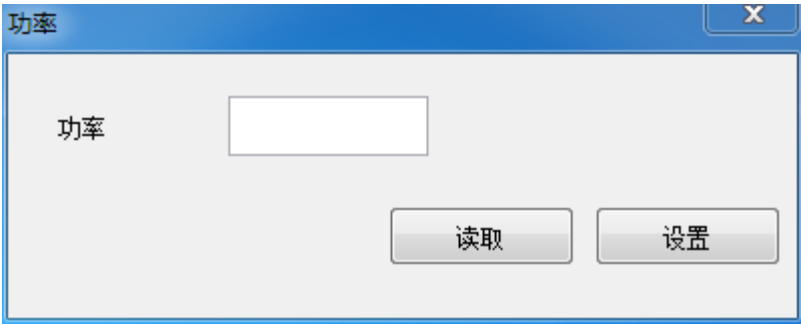


图 8.4、设备功率功能参考

8.4.1、功能：查询设备功率

功能说明：

```
int QueryPower (HANDLE hDev, BYTE & power);
```

函数名称		返回值类型	说明		
QueryTagType		int	查询设备功率(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄

2	参数 2	BYTE &	power	1~150	
---	------	--------	-------	-------	--

8.4.2、功能：更新设备功率

功能说明：

```
int UpdatePower (HANDLE hDev, BYTE & power);
```

函数名称		返回值类型	说明		
UpdatePower		int	更新设备功率(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	power	1~150	

8.5、设备频率

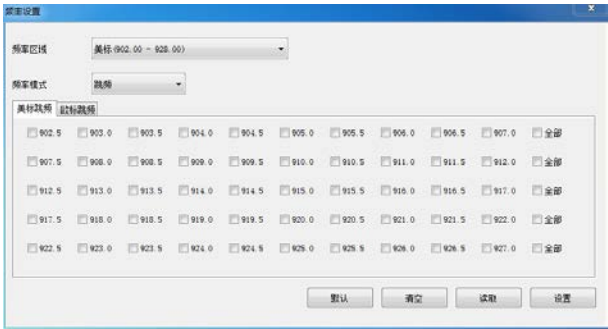


图 8.5.1、设备频率(美标跳频) 功能参考

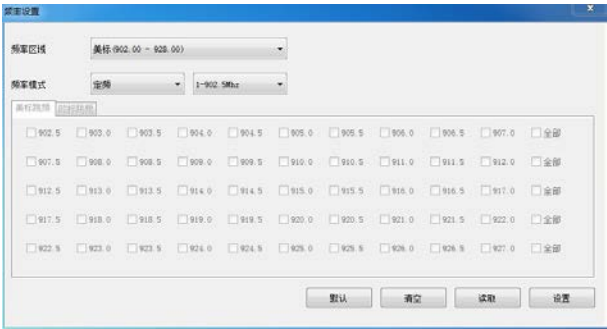


图 8.5.2、设备频率(美标定频) 功能参考

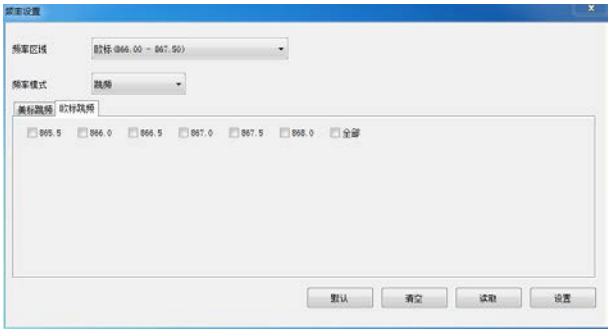


图 8.5.3、设备频率(欧标跳频) 功能参考



图 8.5.4、设备频率(欧标定频)功能参考

8.5.1、功能：查询设备频率

功能说明：

```
int QueryFrequencyPoints (HANDLE hDev, BYTE & national_standard,
                           BYTE & fixed_or_hopping_freq, BYTE * frequency_point, BYTE & frequency_point_len);
```

函数名称		返回值类型	说明		
QueryFrequencyPoints		int	查询设备频率 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	national_standard	0: 美标 (902.50~927.00) 1: 欧标 (865.50~868.00) 2: 韩标 (917.30~920.30)	
3	参数 3	BYTE &	fixed_or_hopping_freq	0: 跳频 1: 定频	

4	参数 4	BYTE *	frequency_point	附表：频率参数表	
5	参数 5	BYTE &	frequency_point_len	附表：频率参数表	

8.5.2、功能：更新频率

功能说明：

```
int UpdateFrequencyPoints (HANDLE hDev, BYTE national_standard,
                           BYTE fixed_or_hopping_freq, BYTE frequency_point[], BYTE frequency_point_len);
```

函数名称		返回值类型	说明		
QueryFrequencyPoints		int	查询设备频率 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	national_standard	0:美标 (902. 50~927. 00) 1:欧标 (865. 50~868. 00)	
3	参数 3	BYTE	fixed_or_hopping_freq	0:跳频 1:定频	
4	参数 4	BYTE []	frequency_point	附表：频率参数表	
5	参数 5	BYTE	frequency_point_len	附表：频率参数表	

8.5.1、频率参数表

频率区域	定频或跳频	长度
美标 902. 50~927. 00		
	定频	(只1个，选下标值对应的频率) 例：设置定频频点为907. 0，下标值为9(frequency_point[0] = 9) frequency_point_len = 1
	跳频	frequency_point[50] (50个，选下标对应的频率(0 or 1为选中)) frequency_point_len = 50

0	1	2	3	4	5	6	7	8	9
902.5	903.0	903.5	904.0	904.5	905.0	905.5	906.0	906.5	907.0
10	11	12	13	14	15	16	17	18	19
907.5	908.0	908.5	909.0	909.5	910.0	910.5	911.0	911.5	912.0
20	21	22	23	24	25	26	27	28	29
912.5	913.0	913.5	914.0	914.5	915.0	915.5	916.0	916.5	917.0
30	31	32	33	34	35	36	37	38	39
917.5	918.0	918.5	919.0	919.5	920.0	920.5	921.0	921.5	922.0
40	41	42	43	44	45	46	47	48	49
922.5	923.0	923.5	924.0	924.5	925.0	925.5	926.0	926.5	927.0
频率区域		定频或跳频		长度					
欧标 865.50~868.00									
		定频		(只1个，选 下标值 对应的频率) 例: 设置定频频点为866.5, 下标为2 (frequency_point[0] = 2) frequency_point_len = 1					
		跳频		frequency_point[6] (6个，选下标对应的频率(0 or 1选中)) frequency_point_len = 6					
0	1	2	3	4	5				
865.5	866.0	866.5	867.0	867.5	868.0				

8.6、读卡模式(单标签和多标签)

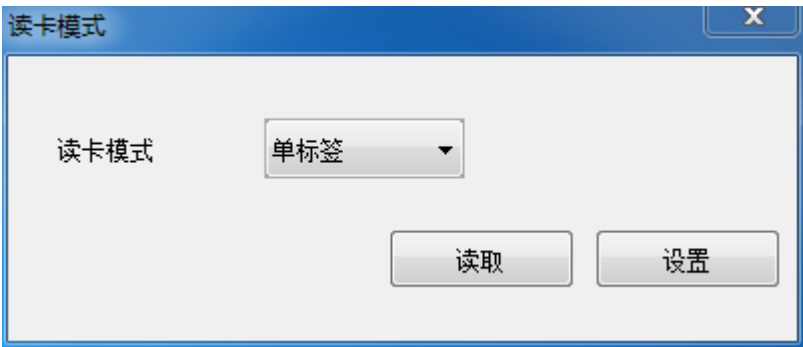


图 8.6、读卡模式功能参考

8.6.1、功能：查询读卡模式

功能说明：

```
int QuerySingleOrMultiTag (HANDLE hDev, BYTE & single_or_multi);
```

函数名称		返回值类型	说明		
QuerySingleOrMultiTag		int	查询读卡模式 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	single_or_multi	0: 单标签 1: 多标签	

8.6.2、功能：更新读卡模式

功能说明：

```
int UpdateSingleOrMultiTag (HANDLE hDev, BYTE single_or_multi);
```

函数名称		返回值类型	说明		
UpdateSingleOrMultiTag		int	更新读卡模式 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	single_or_multi	0: 单标签 1: 多标签	

8.7、设备编号

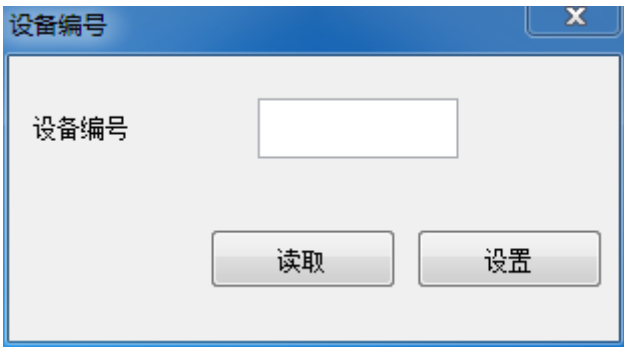


图 8.7、设备编号功能参考

8.7.1、功能：查询设备编号

功能说明：

```
int QueryDeviceId (HANDLE hDev, BYTE & device_id);
```

函数名称		返回值类型	说明		
QueryDeviceId		int	查询设备编号 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	device_id	0~255	设备编号

8.7.2、功能：更新设备编号

功能说明：

```
int UpdateDeviceId (HANDLE hDev, BYTE device_id);
```

函数名称		返回值类型	说明		
UpdateDeviceId		int	更新设备编号 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	device_id	0~255	设备编号

九、其它操作

9.1、蜂鸣器配置(声音)

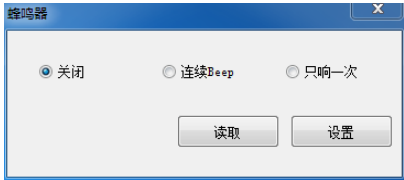


图 9. 1、蜂鸣器功能参考

9.1.1、功能：查询蜂鸣器状态

功能说明：

```
int QueryBuzzer(HANDLE hDev, BYTE & status);
```

函数名称		返回值类型	说明		
QueryBuzzer		int	查询蜂鸣器状态 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	status	0: 关闭 1: 打开	

9.1.2、功能：更新蜂鸣器状态

功能说明：

```
int UpdateBuzzer(HANDLE hDev, BYTE status);
```

函数名称		返回值类型	说明		
UpdateBuzzer		int	更新蜂鸣器状态 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	status	0: 关闭 1: 打开 2: 只响一次	

9.2、继电器配置(GPIO)



图 9.2.1、继电器(主动触发)功能参考



图 9.2.2、继电器(被动触发) 功能参考

9.2.1、功能：查询继电器配置(暂无)

9.2.2、功能：更新继电器模式(GPIO 主动和被动)

功能说明：

```
int UpdateRelayInitiativeOrPassivity (HANDLE hDev, bool mode);
```

函数名称		返回值类型	说明		
UpdateRelayInitiativeOrPassivity		int	更新继电器模式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	bool	mode	0:被动触发	

				1:主动触发	
--	--	--	--	--------	--

9.2.3、功能：更新继电器主动闭合时间(GPIO 主动触发)

功能说明：

```
int UpdateRelayInitiativeCloseTime (HANDLE hDev, BYTE time);
```

函数名称		返回值类型	说明		
UpdateRelayInitiativeCloseTime		int	更新继电器主动闭合时间(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	time	建议值 (1、5、10)	

9.2.4、功能：更新继电器两路触发 (GPIO 被动触发)

功能说明：

```
int UpdateRelay (HANDLE hDev, BYTE on_or_off_1, BYTE on_or_off_2);
```

函数名称		返回值类型	说明		
UpdateRelay		int	更新继电器主动闭合时间(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	on_or_off_1	0:关闭 1:打开	
3	参数 3	BYTE	on_or_off_2	0:关闭 1:打开	

9.3、指定标签读和写区域

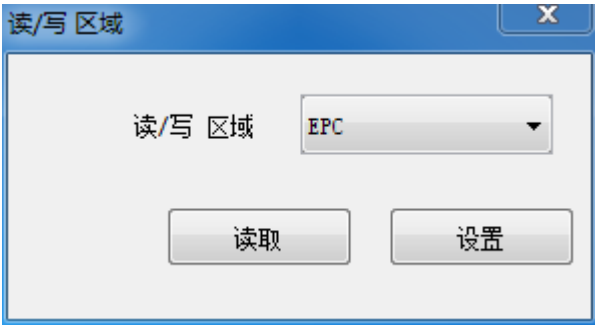


图 9.3、标签读和写区域功能参考

9.3.1、功能：查询标签读和写区域

功能说明：

```
int QueryEpcOrTid (HANDLE hDev, BYTE & operation_area);
```

函数名称		返回值类型	说明		
QueryDataOutputFormat		int	查询标签读和写区域 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	operation_area	0:EPC 1:TID(仅读)	

9.3.2、功能：更新标签读和写区域

功能说明：

```
int UpdateEpcOrTid (HANDLE hDev, BYTE & operation_area);
```

函数名称		返回值类型	说明		
UpdateEpcOrTid		int	更新标签读和写区域 (成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	operation_area	0:EPC 1:TID(仅读)	

9.4、标签数据输出格式

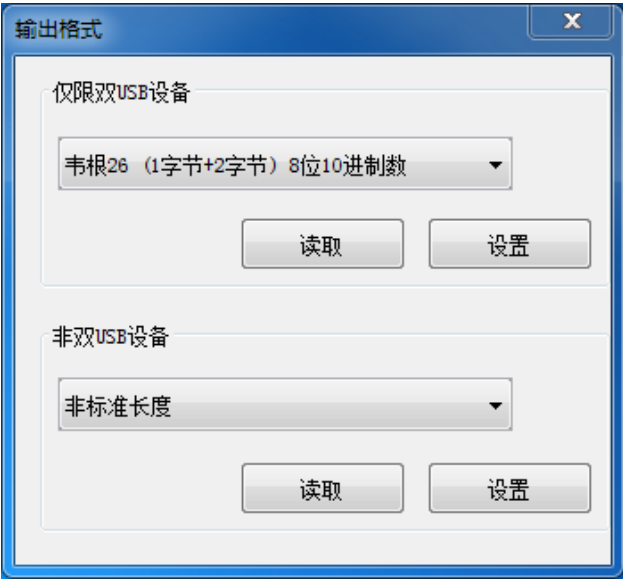


图 9.4、标签数据输出格式

9.4.1、功能：查询标签数据输出格式(仅限双 USB 设备)

功能说明：

```
int QueryUsbOutputFormat (HANDLE hDev, BYTE & format);
```

函数名称		返回值类型	说明		
QueryDataOutputFormat		int	查询标签数据输出格式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	format	0: 韦根26(1字节+2字节)8位10进制 1: 韦根26(3字节)8位10进制 2: 00+3字节16进制 3: 4字节16进制 4: 12字节16进制 5: 12字节(TID) 6: 0000+前3字节16进制	

9.4.2、功能：更新标签数据输出格式(仅限双 USB 设备)

功能说明：

```
int UpdateUsbOutputFormat (HANDLE hDev, BYTE format);
```

函数名称		返回值类型	说明		
UpdateUsbOutputFormat		int	查询标签数据输出格式(成功 1 失败 0)		

序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	format	0: 韦根26(1字节+2字节)8位10进制 1: 韦根26(3字节)8位10进制 2: 00+3字节16进制 3: 4字节16进制 4: 12字节16进制 5: 12字节(TID) 6: 0000+前3字节16进制	

9.4.3、功能：查询标签数据输出格式(非双 USB 设备)

功能说明：

int QueryNotUsbOutputFormat (HANDLE hDev, BYTE & format);

函数名称		返回值类型	说明		
QueryNotUsbOutputFormat		int	查询标签数据输出格式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE &	format	0: 非标准长度 1: 12字节	

9.4.4、功能：更新标签数据输出格式(非双 USB 设备)

功能说明：

int UpdateNotUsbOutputFormat (HANDLE hDev, BYTE format);

函数名称		返回值类型	说明		
UpdateNotUsbOutputFormat		int	更新标签数据输出格式(成功 1 失败 0)		
序号	参数	类型	变量名	参数范围	描述
1	参数 1	HANDLE	hDev		句柄
2	参数 2	BYTE	format	0: 非标准长度 1: 12字节	