915MHz RFID 读卡器
动态库使用指南

目录

1. 函数库说明

动态库一共有 8 个文件，包括 adpcom.dll, adpcom.lib, adpnet.dll, adpnet.lib, adppub.dll, adppub.lib, adpusb.dll, adpusb.lib,。动态库是 Microsoft Windows 的接口标准，流行的软件开发工具 VC、VB、VF、Delphi、C++ Builder、Power Builder 等均可使用。

该函数库仅适用于读卡器工作在被动模式或者应答模式下；当读卡器工作在主动模式下时，该函数库可能会造成数据冲突或者重叠错误；

2. 串行接口设备

仅适用于采用 RS-232 或 RS-485 联接读写器.
选用 adpcom.dll, adpcom.lib 动态库;

2.1. 设备管理函数

2.1.1. ap_open (连接串口设备)

HANDLE _stdcall ap_open (int nPort, int nBaud)

功 能： 初始化串口
参 数： nPort：串口号，取值为 1～250
nBaud：为通讯波特率 1200～115200,（**非定制类型默认值：9600**）
返 回： 成功则返回串口句柄，失败返回值，见错误代码表

例： int m_handle;
m_handle = ap_open (2,9600);//初始化串口 1，波特率 9600

如果是 WIN32 程序则 m_handle 为设备句柄，见下例：

```
HANDLE m_handle;
m_handle = ap_open (2, 9600);
if(m_handle<0)
    MessageBox("ap_open error");
```

2.1.2. ap_close (释放串口)

void _stdcall ap_close (HANDLE m_handle)

功 能： 释放串口
参 数： m_handle：通讯设备标识符
返 回： 无

例：　　ap_close (m_handle);

注：在 WIN32 环境下 m_handle 为串口的设备句柄，必须释放后才可以再次连接。

2.1.3. ap_getaddress (获取设备通讯地址及版本信息)

int _stdcall ap_getaddress (HANDLE m_handle, int *m_oAddress, int *m_oVer)

功 能：　获取设备通讯地址
参 数：　m_handle：通讯设备标识符
　　m_iAddress：获取的设备通讯地址指针地址
　　m_oVer：获取的设备版本信息指针地址

返 回：　成功返回 0

例：　　int m_iAddress;
　　int st;
　　st = ap_getaddress (m_handle, m_iAddress, m_oVer);

2.1.4. ap_setaddress (设置设备通讯地址)

int _stdcall ap_setaddress (HANDLE m_handle, int m_iAddress , int idata)

功 能：　设置设备通讯地址
参 数：　m_handle：通讯设备标识符
　　m_iAddress：设备当前通讯地址
　　idata：待设置的设备通讯地址

返 回：　成功返回 0

例：　　int idata = 65534;
　　int st;
　　st = ap_setaddress (m_handle, m_iAddress, idata);

2.1.5. ap_getconfig (获取设备基本参数)

int _stdcall ap_getconfig (HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char *oSize)

功 能：　获取设备基本参数
参 数：　m_handle：通讯设备标识符
　　m_iAddress：当前通讯地址
　　oData：返回基本参数指针地址（见基本参数表）
　　oSize：返回基本参数字节数指针地址

返 回：　成功返回 0

例：
```
unsigned char    oData[28];
unsigned char    oSize;
int st;
st = ap_getconfig (m_handle, m_iAddress, oData, oSize);
```

2.1.6. ap_setconfig (设置设备基本参数)

int _stdcall ap_setconfig (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iSize)

功 能： 设置设备基本参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待设置基本参数指针地址（见基本参数表）
iSize：待设置基本参数字节数

返 回： 成功返回 0

例：
```
unsigned char    iData[] = {0x01,};
unsigned char    iSize = 28;
int st;
st = ap_setconfig (m_handle, m_iAddress, iData, iSize);
```

2.1.7. ap_gettcpip (获取设备网络通讯参数)

int _stdcall ap_gettcpip (HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char *oSize)

功 能： 获取设备网络通讯参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回参数指针地址（见 TCPIP 参数表）
oSize：返回参数字节数指针地址

返 回： 成功返回 0

例：
```
unsigned char    oData[20];
unsigned char    oSize;
int st;
st = ap_gettcpip (m_handle, m_iAddress, oData, oSize);
```

2.1.8. ap_settcpip (设置设备网络通讯参数)

int _stdcall ap_settcpip (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iSize)

功 能： 设置设备网络通讯参数

参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待设置参数指针地址（见 TCPIP 参数表）
iSize：待设置参数字节数

返 回： 成功返回 0

例： unsigned char iData[]={0x01,};
unsigned char iSize=20;
int st;
st = ap_setconfig (m_handle, m_iAddress, iData, iSize);

2.2. ISO18000-6B协议标签操作函数

2.2.1. ap_identify6b (识别IS0180000-6B协议标签)

int _stdcall ap_identify6b (HANDLE m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 IS0180000-6B 协议标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

例： unsigned char oData[12];
unsigned char oSize;
int st;
st = ap_identify6b (m_handle, m_iAddress, oData, oSize);

注: 获取标签中唯一 ID 号，获取数据 12 个字节,一般前 8 个字节为卡号,后 4 个字节默认为 0;

2.2.2. ap_read6b (读取IS0180000-6B协议标签中数据)

int _stdcall ap_read6b (HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char iAddr, unsigned char iSize)

功 能： 读取 IS0180000-6B 协议标签中数据
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
iAddr：待获取数据地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

例： unsigned char oData[12];

```
unsigned char     iAddr = 18;
unsigned char     iSize = 12;
int st;
st = ap_read6b (m_handle, m_iAddress, oData, iAddr, oSize);
```

注: **获取标签中自定义数据,地址开始位置从 0 开始;**

2.2.3. ap_write6b (写入数据到IS0180000-6B协议标签中)

int _stdcall ap_write6b (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iAddr, unsigned char iSize)

功 能： 写入数据到 IS0180000-6B 协议标签中
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：返回数据
iAddr：待写入数据地址
iSize：待写入数据字节数

返 回： 成功返回 0

例：
```
unsigned char     iData [2]={0x01,0x02};
unsigned char     iAddr = 18;
unsigned char     iSize = 2;
int st;
st = ap_write6b (m_handle, m_iAddress, iData, iAddr, iSize);
```

注: **写入数据到标签中,地址开始位置从 18 开始,前面位置数据为不可修改区;**

2.3. EPC(GEN 2)协议标签操作函数

2.3.1. ap_identify6c (识别EPC(GEN 2)协议标签)

int _stdcall ap_identify6c (HANDLE m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 EPC(GEN 2)协议标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

例：
```
unsigned char     oData[12];
unsigned char     oSize;
int st;
st = ap_identify6c (m_handle, m_iAddress, oData, oSize);
```

注: **获取标签中 EPC 区 12 字节数据;**

2.3.2. ap_identify6cmult (识别EPC(GEN 2)协议多标签)

int _stdcall ap_identify6cmult (HANDLE m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 EPC(GEN 2)协议多标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

例：
```
unsigned char    oData[12];
unsigned char    oSize;
int st;
st = ap_identify6c (m_handle, m_iAddress, oData, oSize);
```

注: 获取多个标签中 EPC 区 12 字节数据;

2.3.3. ap_read6c (读取EPC(GEN 2)协议标签中数据)

int _stdcall ap_read6c (HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char iMem, unsigned char iAddr, unsigned char iSize)

功 能： 读取 EPC(GEN 2)协议标签中数据
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
iMem：待获取数据块地址
iAddr：待获取数据地址
oSize：返回数据字节数

返 回： 成功返回 0

例：
```
unsigned char    oData[12];
unsigned char    iMem = 1;
unsigned char    iAddr = 2;
unsigned char    iSize = 2;
int st;
st = ap_read6c (m_handle, m_iAddress, oData, iMem, iAddr, oSize);
```

卡片存储划分:

块名称	存储内容	块地址	字节	容量	读/写
Reserved(保留区)	存储 KILL PASSWORD 和 ACCESS PASSWORD	00H	8	64bits	只读
EPC(EPC 区)	存取 EPC 号码	01H	12	96bits	可读 可写
TID(TID 区)	存取标签识别号码,	02H	24	196bits	只读

	每个 TID 号码应该是唯一的				
USER(USER 区)	存取用户自定义的数据	03H	64	512bits	可读 可写

注: **EPC 区数据地址从 2 开始;**
任意区中,每个地址存储 2 个字节;
例如:
EPC 区卡号: 01 02 03 04 05 06 07 08 09 10 11 12;
iMem = 1; iAddr = 2; iSize = 4;
获取数据为: 01 01 02 03 04 (前面 01 为天线号,一体化读卡器默认 01,多通道读卡器为通道号)

iMem = 1; iAddr = 3; iSize = 4;
获取数据为: 01 03 04 05 06 ;

iMem = 1; iAddr = 4; iSize = 4;
获取数据为: 01 05 06 07 08 ;

类推…

2.3.4. ap_write6c (写入数据到EPC(GEN 2)协议标签中)

int _stdcall ap_write6c (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iMem, unsigned char iAddr, unsigned char iSize)

功 能： 写入数据到 EPC(GEN 2)协议标签中
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待写入数据指针地址
iMem：待写入数据块地址
iAddr：待写入数据地址
iSize：待写入数据字节数

返 回： 成功返回 0

例：
```
unsigned char    iData [2]={0x01,0x02};
unsigned char    iMem = 1;
unsigned char    iAddr = 2;
unsigned char    iSize = 2;
int st;
st = ap_write6b (m_handle, m_iAddress, iData, iMem, iAddr, iSize);
```

2.3.5. ap_encrypt (EPC(GEN 2)协议标签加密)

int _stdcall ap_encrypt (HANDLE m_handle, int m_iAddress)

功 能： EPC(GEN 2)协议标签加密
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址

返 回： 成功返回 0

例：　　int st;
　　st = ap_encrypt (m_handle, m_iAddress);

3. 网络接口设备

仅适用于采用 TCPIP 联接读写器,读卡器仅做服务器端使用.
选用 adpnet.dll, adpnet.lib 动态库;

3.1. 设备管理函数

3.1.1. an_open (连接网络设备)

SOCKET _stdcall an_open (char * ulAddress, int iPort)

功 能：　连接网络设备
参 数：　ulAddress：远程 IP 地址，默认值 192.168.1.115
　　iPort：远程 IP 端口,默认值 49152
返 回：　成功则返回网络连接句柄，失败返回值，见错误代码表

例：　　int m_handle;
　　m_handle = an_open (“192.168.1.115”, 49152);

3.1.2. an_close (释放网络)

void _stdcall an_close (SOCKET m_handle)

功 能：　释放串口
参 数：　m_handle：通讯设备标识符
返 回：　无

例：　　an_close (m_handle);

注：在 WIN32 环境下 m_handle 网络的设备句柄，必须释放后才可以再次连接。

3.1.3. an_getaddress (获取设备通讯地址及版本信息)

int _stdcall an_getaddress (SOCKET m_handle, int *m_oAddress, int *m_oVer)

功 能：　获取设备通讯地址
参 数：　m_handle：通讯设备标识符
　　m_iAddress：获取的设备通讯地址指针地址
　　m_oVer：获取的设备版本信息指针地址

返 回： 成功返回 0

3.1.4. an_setaddress (设置设备通讯地址)

int _stdcall an_setaddress (SOCKET m_handle, int m_iAddress , int idata)

功 能： 设置设备通讯地址
参 数： m_handle：通讯设备标识符
m_iAddress：设备当前通讯地址
idata：待设置的设备通讯地址

返 回： 成功返回 0

3.1.5. an_getconfig (获取设备基本参数)

int _stdcall an_getconfig (SOCKET m_handle, int m_iAddress , unsigned char *oData, unsigned char *oSize)

功 能： 获取设备基本参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回基本参数指针地址（见基本参数表）
oSize：返回基本参数字节数指针地址

返 回： 成功返回 0

3.1.6. an_setconfig (设置设备基本参数)

int _stdcall an_setconfig (SOCKET m_handle, int m_iAddress , unsigned char *iData, unsigned char iSize)

功 能： 设置设备基本参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待设置基本参数指针地址（见基本参数表）
iSize：待设置基本参数字节数

返 回： 成功返回 0

3.1.7. an_gettcpip (获取设备网络通讯参数)

int _stdcall an_gettcpip(SOCKET m_handle, int m_iAddress , unsigned char *oData, unsigned char *oSize)

功 能： 获取设备网络通讯参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回参数指针地址（见 TCPIP 参数表）
oSize：返回参数字节数指针地址

返 回： 成功返回 0

注: 网络版设置 TCPIP 参数时,读卡器将重启,需要重新连接;

3.1.8. an_settcpip (设置设备网络通讯参数)

int _stdcall an_settcpip (SOCKET m_handle, int m_iAddress , unsigned char *iData, unsigned char iSize)

功 能： 设置设备网络通讯参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待设置参数指针地址（见 TCPIP 参数表）
iSize：待设置参数字节数

返 回： 成功返回 0

3.2. ISO18000-6B协议标签操作函数

3.2.1. an_identify6b (识别IS0180000-6B协议标签)

int _stdcall an_identify6b (SOCKET m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 IS0180000-6B 协议标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

3.2.2. an_read6b (读取IS0180000-6B协议标签中数据)

int _stdcall an_read6b (SOCKET m_handle, int m_iAddress , unsigned char *oData, unsigned char iAddr, unsigned char iSize)

功 能： 读取 IS0180000-6B 协议标签中数据
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址

oData：返回数据指针地址
iAddr：待获取数据地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

3.2.3. an_write6b (写入数据到IS0180000-6B协议标签中)

int _stdcall an_write6b (SOCKET m_handle, int m_iAddress , unsigned char *iData, unsigned char iAddr, unsigned char iSize)

功 能： 写入数据到 IS0180000-6B 协议标签中
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：返回数据
iAddr：待写入数据地址
iSize：待写入数据字节数

返 回： 成功返回 0

3.3. EPC(GEN 2)协议标签操作函数

3.3.1. an_identify6c (识别EPC(GEN 2)协议标签)

int _stdcall an_identify6c (SOCKET m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 EPC(GEN 2)协议标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

3.3.2. an_identify6cmult (识别EPC(GEN 2)协议多标签)

int _stdcall an_identify6cmult (SOCKET m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 EPC(GEN 2)协议多标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

3.3.3. an_read6c (读取EPC(GEN 2)协议标签中数据)

int _stdcall an_read6c (SOCKET m_handle, int m_iAddress , unsigned char *oData, unsigned char iMem, unsigned char iAddr, unsigned char iSize)

功 能： 读取 EPC(GEN 2)协议标签中数据
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
iMem：待获取数据块地址
iAddr：待获取数据地址
oSize：返回数据字节数

返 回： 成功返回 0

3.3.4. an_write6c (写入数据到EPC(GEN 2)协议标签中)

int _stdcall an_write6c (SOCKET m_handle, int m_iAddress , unsigned char *iData, unsigned char iMem, unsigned char iAddr, unsigned char iSize)

功 能： 写入数据到 EPC(GEN 2)协议标签中
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待写入数据指针地址
iMem：待写入数据块地址
iAddr：待写入数据地址
iSize：待写入数据字节数

返 回： 成功返回 0

3.3.5. an_encrypt (EPC(GEN 2)协议标签加密)

int _stdcall an_encrypt (SOCKET m_handle, int m_iAddress)

功 能： EPC(GEN 2)协议标签加密
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址

返 回： 成功返回 0

4. USB接口设备

仅适用于采用 USB 联接读写器.
选用 adpusb.dll, adpusb.lib 动态库;

4.1. 设备管理函数

4.1.1. ad_open (连接USB设备)

HANDLE __stdcall ad_open ()

功 能： 连接 USB 设备
返 回： 成功则返回设备连接句柄，失败返回值，见错误代码表

例：
```
int m_handle = -1;
While (m_handle != 0)
{
    m_handle = ad_open ();
}
```

4.1.2. ad_close (释放USB设备)

void __stdcall ad_close(HANDLE m_handle)

功 能： 释放串口
参 数： m_handle：通讯设备标识符
返 回： 无

例： ad_close (m_handle);

4.1.3. ad_exitprogram (退出编程模式)

int _stdcall ad_exitprogram (HANDLE m_handle)
功 能： 退出编程模式
参 数： m_handle：通讯设备标识符
返 回： 无

例： ad_close (m_handle);

4.1.4. ad_getaddress (获取设备通讯地址及版本信息)

int _stdcall ad_getaddress (HANDLE m_handle, int *m_oAddress, int *m_oVer)

功 能： 获取设备通讯地址
参 数： m_handle：通讯设备标识符
m_iAddress：获取的设备通讯地址指针地址
m_oVer：获取的设备版本信息指针地址

返 回： 成功返回 0

4.1.5. ad_setaddress (设置设备通讯地址)

int _stdcall ad_setaddress(HANDLE m_handle, int m_iAddress , int idata)

功 能： 设置设备通讯地址
参 数： m_handle：通讯设备标识符
m_iAddress：设备当前通讯地址
idata：待设置的设备通讯地址

返 回： 成功返回 0

4.1.6. ad_getconfig (获取设备基本参数)

int _stdcall ad_getconfig(HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char *oSize)

功 能： 获取设备基本参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回基本参数指针地址（见基本参数表）
oSize：返回基本参数字节数指针地址

返 回： 成功返回 0

4.1.7. ad_setconfig (设置设备基本参数)

int _stdcall ad_setconfig (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iSize)

功 能： 设置设备基本参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待设置基本参数指针地址（见基本参数表）
iSize：待设置基本参数字节数

返 回： 成功返回 0

4.1.8. ad_getoutstatus (获取输出模式参数)

int _stdcall ad_getoutstatus (HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char *oSize)

功 能： 获取设备网络通讯参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回参数指针地址（见输出模式参数表）
oSize：返回参数字节数指针地址

返 回： 成功返回 0

4.1.9. ad_setoutstatus (设置输出模式参数)

int _stdcall ad_setoutstatus (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iSize)

功 能： 设置设备网络通讯参数
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待设置参数指针地址（见输出模式参数表）
iSize：待设置参数字节数

返 回： 成功返回 0

4.2. ISO18000-6B协议标签操作函数

4.2.1. ad_identify6b (识别IS0180000-6B协议标签)

int _stdcall ad_identify6b(HANDLE m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 IS0180000-6B 协议标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

4.2.2. ad_read6b (读取IS0180000-6B协议标签中数据)

int _stdcall ad_read6b (HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char iAddr, unsigned char iSize)

功 能： 读取 IS0180000-6B 协议标签中数据
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
iAddr：待获取数据地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

4.2.3. ad_write6b (写入数据到IS0180000-6B协议标签中)

int _stdcall ad_write6b (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iAddr, unsigned char iSize)

功 能： 写入数据到 IS0180000-6B 协议标签中
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：返回数据
iAddr：待写入数据地址
iSize：待写入数据字节数

返 回： 成功返回 0

4.3. EPC(GEN 2)协议标签操作函数

4.3.1. ad_identify6c (识别EPC(GEN 2)协议标签)

int _stdcall ad_identify6c (HANDLE m_handle, int m_iAddress, unsigned char *oData, unsigned char *oSize)

功 能： 识别 EPC(GEN 2)协议标签
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
oSize：返回数据字节数指针地址

返 回： 成功返回 0

4.3.2. ad_read6c (读取EPC(GEN 2)协议标签中数据)

int _stdcall ad_read6c (HANDLE m_handle, int m_iAddress , unsigned char *oData, unsigned char iMem, unsigned char iAddr, unsigned char iSize)

功 能： 读取 EPC(GEN 2)协议标签中数据
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
oData：返回数据指针地址
iMem：待获取数据块地址
iAddr：待获取数据地址
oSize：返回数据字节数

返 回： 成功返回 0

4.3.3. ad_write6c (写入数据到EPC(GEN 2)协议标签中)

int _stdcall ad_write6c (HANDLE m_handle, int m_iAddress , unsigned char *iData, unsigned char iMem, unsigned char iAddr, unsigned char iSize)

功 能： 写入数据到 EPC(GEN 2)协议标签中
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址
iData：待写入数据指针地址
iMem：待写入数据块地址
iAddr：待写入数据地址
iSize：待写入数据字节数

返 回： 成功返回 0

4.3.4. ad_encrypt (EPC(GEN 2)协议标签加密)

int _stdcall ad_encrypt (HANDLE m_handle, int m_iAddress)

功 能： EPC(GEN 2)协议标签加密
参 数： m_handle：通讯设备标识符
m_iAddress：当前通讯地址

返 回： 成功返回 0

5. 错误代码表

错误代码	错误状态	错误代码	错误状态	错误代码	错误状态

201	打开失败	211	连接 USB 失败	252	通讯地址错误
202	取参数失败	212	进入编程模式失败	253	协议不完整
203	设置参数失败	213	退出编程模式失败	256	数据转换错误
204	设置超时失败			259	数据不完整
205	发送数据失败				
206	接收数据失败				
207	关闭失败				
208	发送超时				
209	接收超时				

6. 附录参数表

6.1. 基本参数表

参数	说明	参考值
Para1	功率大小	可调节读卡器读取标签的距离 默认值：30 参考值：（十进制格式）0~30
Para2	跳频使能	可设置定频或者跳频方式 默认值：1 参考值：（十进制格式）1-定频，2,-跳频
Para3	定频值	默认值：110（915MHz） 参考值：（十进制格式）0~200(860MHz ~ 960MHz)
Para4	跳频值 1	默认值：84（902MHz） 参考值：（十进制格式）0~200(860MHz ~ 960MHz)
Para5	跳频值 2	默认值：93（906.5MHz） 参考值：（十进制格式）0~200(860MHz ~ 960MHz)
Para6	跳频值 3	默认值：102（911MHz） 参考值：（十进制格式）0~200(860MHz ~ 960MHz)
Para7	跳频值 4	默认值：110（915MHz） 参考值：（十进制格式）0~200(860MHz ~ 960MHz)
Para8	跳频值 5	默认值：119（919.5MHz） 参考值：（十进制格式）0~200(860MHz ~ 960MHz)
Para9	跳频值 6	默认值：130（925MHz） 参考值：（十进制格式）0~200(860MHz ~ 960MHz)
Para10	工作模式	应答方式：读卡器停止工作，上位机发送命令，读卡器工作，并根据指令动作； 主动方式：读卡器正常工作，当识别到标签时主动上送数据； 被动方式：读卡器正常工作，识别到标签时标签缓存在读卡器中，上位机发送命令来获取该标签数据； 默认值：2 参考值：（十进制格式）1-应答方式 2-主动方式 3-被动方式
Para11	定时发送间隔	默认值：10（*1ms） 参考值：（十进制格式）5~255（* 1ms）

Para12	外部触发方式	默认值：0 参考值：（十进制格式）0-关闭 2-低电平有效
Para13	输出方式	默认值：1 参考值：（十进制格式） 1- RS232 2- RS485 3- TCPIP 4- CANBUS 5- SYRIS 6- Wiegand26 7- Wiegand34
Para14	韦根参数 1-数据偏移	具体参考 Wiegand 协议 默认值：0 参考值：0~20
Para15	韦根参数 2 – 输出周期	具体参考 Wiegand 协议 默认值：30（* 10ms） 参考值：0~255（* 10ms）
Para16	韦根参数 3 –脉冲宽度	具体参考 Wiegand 协议 默认值：10（* 10us） 参考值：0~255（* 10us）
Para17	韦根参数 4 – 脉冲周期	具体参考 Wiegand 协议 默认值：15（* 100us） 参考值：0~255（* 100us）
Para18	天线设置	一字节数据，低 4 位表示 4 路天线， 举例：使用天线 1: 01H（二进制 0000 0001） 使用天线 3: 04H（二进制 0000 0100） 使用天线 1 和天线 3: 05H（二进制 0000 0101）
Para19	读卡类别	默认值：16 参考值：（十进制格式） 1-ISO18000-6B 单卡 16-EPC(GEN 2) 单卡 17-EPC(GEN 2) + ISO18000-6B 32-EPC(GEN 2) 多卡 64-EPC(GEN 2)+其他分区
Para20	相同 ID 输出间隔	默认值：1s 参考值：（十进制格式）0~255s
Para21	蜂鸣器	默认值：1 参考值：（十进制格式）0-禁止 1-使能
Para22	识读其他分区选择	读卡类别为【EPC(GEN 2)+其他分区】时，此参数为其他分区选择： 默认值：1 参考值：（十进制格式） 1-TID 区（全球唯一号码区） 2-USER 区（用户自定义数据区）
Para23	识读其他分区地址	读卡类别为【EPC(GEN 2)+其他分区】时，此参数为其他分区数据获取起始地址选择： 默认值：0 参考值：（十进制格式）0~31
Para24	识读其他分区长度	读卡类别为【EPC(GEN 2)+其他分区】时，此参数为其他分区数据获取长度选择： 默认值：2 参考值：（十进制格式）1~12
Para25	加密功能使能	使能读卡器加密读卡；

		默认值：0 参考值：（十进制格式） 0-通用版，不加密； 1-读卡器加密；
Para26 Para27	加密密码	默认值：0000 参考值：（十进制格式）0000~9999 举例：密码 0123(十进制) = 00H 7BH（十六进制）
Para28	最大读卡数量	默认值：32 参考值：（十进制格式）10~64

6.2. TCPIP参数表

参数	说明	参考值
Para1 Para2 Para3 Para4	IP 地址（4 字节）	默认值：192.168.5.105 举例： IP = 192.168.5.105 表示为 C0 A8 05 69H
Para5 Para6 Para7 Para8	子网掩码（4 字节）	子网掩码是用于屏蔽 IP 地址的一部分以区别网络标识和主机标识，并说明该 IP 地址是在局域网上，还是在远程网上。 默认值：255.255.255.0 举例： SubNet Mask = 255.255.255.0 表示为 FF FF FF 00H
Para9 Para10 Para11 Para12	默认网关（4 字节）	默认值：192.168.5.1 举例： Gateway = 192.168.5.1 表示为 C0 A8 05 01H
Para13 Para14	IP 端口	默认值：49152 举例： IP Port = 49152 表示为 C0 00H
Para15 Para16 Para17 Para18 Para19 Para20	物理地址（6 字节）	默认值：5E-45-A2-6C-30-1E 举例： MAC = 5E-45-A2-6C-30-1E 表示为 5E 45 A2 6C 30 1EH

6.3. 输出模式参数表

参数	说明	参考值
Para1	输出类型	输出的数据类型 默认值：0 参考值：（十进制格式） 0-Decimal (1747988) 1-Hex (1AAC14) 2-Wiegand (02644052)
Para2	输出位数	可设置定频或者跳频方式 默认值：8 参考值：（十进制格式） 8-输出位数 8 位(01747988)

		9-输出位数 9 位(001747988) 10-输出位数 10 位(0001747988)
Para3	是否带回车符	默认值：0 参考值：（十进制格式） 0-不带回车符 1-带回车符

7. DELPHI 调用函数

//---RS232

function ap_open (nPort : Integer; nBaud : Integer): Integer; stdcall; external adpcom.dll';

function ap_close(m_hCom : Integer):bool; stdcall; external adpcom.dll';

function ap_getaddress(m_hCom : Thandle; oAddress : PDWORD; oVer : PDWORD) : integer; stdcall; external adpcom.dll'; //获取设备通讯地址

function ap_setaddress(m_hCom : Thandle; iAddress : Integer; iData : Integer) : integer; stdcall; external adpcom.dll'; //设置设备通讯地址

function ap_getconfig(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external adpcom.dll'; //获取基本参数

function ap_setconfig(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iSize : Integer) : integer; stdcall; external adpcom.dll'; //设置基本参数

function ap_gettcpip(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external adpcom.dll'; //获取 TCPIP 参数

function ap_settcpip(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iSize : Integer) : integer; stdcall; external adpcom.dll'; //设置 TCPIP 参数

function ap_identify6b(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external adpcom.dll'; //获取 6B 卡号

function ap_read6b(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; iAddr : Integer; iSize : Integer) : integer; stdcall; external adpcom.dll'; //获取 6B 数据

function ap_write6b(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iAddr : Integer; iSize : Integer) : integer; stdcall; external adpcom.dll'; //设置 6B 数据

function ap_identify6c(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external adpcom.dll'; //获取 6c 卡号

function ap_identify6cmult(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external adpcom.dll'; //获取 6c 卡号 (多卡)

function ap_read6c(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; iMem : Integer; iAddr : Integer; iSize : Integer) : integer; stdcall; external adpcom.dll'; //获取 6c 数据

function ap_write6c(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iMem : Integer; iAddr : Integer; iSize : Integer) : integer; stdcall; external adpcom.dll'; //设置 6c 数据

function ap_encrypt(m_hCom : Thandle; iAddress : Integer) : integer; stdcall; external adpcom.dll'; //加密 6c 卡

function ap_softreset(m_hCom : Thandle; iAddress : Integer) : integer; stdcall; external adpcom.dll'; //软件复位设备

function ap_getautocard(m_hCom : Thandle; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external adpcom.dll'; //获取自动上送卡号

//---TCPIP

function an_open(ip : string; port : Integer): Integer; stdcall; external adpnet.dll';

function an_close(m_hCom : Integer):bool; stdcall; external adpnet.dll';

function an_getaddress(m_hCom : Thandle; oAddress : PDWORD; oVer : PDWORD) : integer; stdcall; external adpnet.dll'; //获取设备通讯地址

function an_setaddress(m_hCom : Thandle; iAddress : Integer; iData : Integer) : integer; stdcall; external 'adpnet.dll'; //设置设备通讯地址

function an_getconfig(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpnet.dll'; //获取基本参数

function an_setconfig(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iSize : Integer) : integer; stdcall; external 'adpnet.dll'; //设置基本参数

function an_gettcpip(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpnet.dll'; //获取 TCPIP 参数

function an_settcpip(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iSize : Integer) : integer; stdcall; external 'adpnet.dll'; //设置 TCPIP 参数

function an_identify6b(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpnet.dll'; //获取 6B 卡号

function an_read6b(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpnet.dll'; //获取 6B 数据

function an_write6b(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpnet.dll'; //设置 6B 数据

function an_identify6c(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpnet.dll'; //获取 6c 卡号

function an_identify6cmult(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpnet.dll'; //获取 6c 卡号 (多卡)

function an_read6c(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; iMem : Integer; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpnet.dll'; //获取 6c 数据

function an_write6c(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iMem : Integer; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpnet.dll'; //设置 6c 数据

function an_encrypt(m_hCom : Thandle; iAddress : Integer) : integer; stdcall; external 'adpnet.dll'; //加密 6c 卡

function an_getautocard(m_hCom : Thandle; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpnet.dll'; //获取自动上送卡号

//---USB

function ad_open(): Integer; stdcall; external 'adpusb.dll';

function ad_close(m_hCom : Integer):bool; stdcall; external 'adpusb.dll';

function ad_getaddress(m_hCom : Thandle; oAddress : PDWORD; oVer : PDWORD) : integer; stdcall; external 'adpusb.dll'; //获取设备通讯地址

function ad_setaddress(m_hCom : Thandle; iAddress : Integer; iData : Integer) : integer; stdcall; external 'adpusb.dll'; //设置设备通讯地址

function ad_getconfig(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpusb.dll'; //获取基本参数

function ad_setconfig(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iSize : Integer) : integer; stdcall; external 'adpusb.dll'; //设置基本参数

function ad_getoutstatus(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpusb.dll'; //获取输出类型参数

function ad_setoutstatus(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iSize : Integer) : integer; stdcall; external 'adpusb.dll'; //设置输出类型参数

function ad_identify6b(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpusb.dll'; //获取 6B 卡号

function ad_read6b(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpusb.dll'; //获取 6B 数据

function ad_write6b(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpusb.dll'; //设置 6B 数据

function ad_identify6c(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpusb.dll'; //获取 6c 卡号

function ad_read6c(m_hCom : Thandle; iAddress : Integer; oData : PDWORD; iMem : Integer; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpusb.dll'; //获取 6c 数据

function ad_write6c(m_hCom : Thandle; iAddress : Integer; iData : PDWORD; iMem : Integer; iAddr : Integer; iSize : Integer) : integer; stdcall; external 'adpusb.dll'; //设置 6c 数据

function ad_encrypt(m_hCom : Thandle; iAddress : Integer) : integer; stdcall; external 'adpusb.dll'; //加密 6c 卡

function ad_getautocard(m_hCom : Thandle; oData : PDWORD; oSize : PDWORD) : integer; stdcall; external 'adpusb.dll'; //获取自动上送卡号

8. C#.NET 调用函数

```
#region ---RS232---
[DllImport("adpcom.dll")]
public static extern int ap_open(int iPort, int iBaudrate); // ap_open - 打开串口
[DllImport("adpcom.dll")]
public static extern void ap_close(int iHandle);    // ap_close - 关闭串口
[DllImport("adpcom.dll")]
public static extern int ap_getaddress(int iHandle, ref int oAddress, ref int oVer);// ap_getaddress - 获取地址
[DllImport("adpcom.dll")]
public static extern int ap_setaddress(int iHandle, int iAddress, int iData);// ap_setaddress - 设置地址
[DllImport("adpcom.dll")]
public static extern int ap_getconfig(int iHandle, int iAddress, IntPtr oData, ref byte oSize);// ap_getconfig - 获取参数
[DllImport("adpcom.dll")]
public static extern int ap_setconfig(int iHandle, int iAddress, byte[] iData, byte iSize);// ap_setconfig - 设置参数
[DllImport("adpcom.dll")]
public static extern int ap_gettcpip(int iHandle, int iAddress, IntPtr oData, ref byte oSize);// ap_gettcpip - 获取tcpip参数
[DllImport("adpcom.dll")]
```

```
        public static extern int ap_settcpip(int iHandle, int iAddress, byte[] iData, byte iSize);// ap_settcpip - 设置tcpip参数
        [DllImport("adpcom.dll")]
        public static extern int ap_identify6b(int iHandle, int iAddress, IntPtr oData, ref byte oSize);// ap_identify6b - 识别6b卡号
        [DllImport("adpcom.dll")]
        public static extern int ap_read6b(int iHandle, int iAddress, IntPtr oData, byte iAddr, byte iSize);// ap_read6b - 读6b数据
        [DllImport("adpcom.dll")]
        public static extern int ap_write6b(int iHandle, int iAddress, byte[] iData, byte iAddr, byte iSize);// ap_write6b - 写6b数据
        [DllImport("adpcom.dll")]
        public static extern int ap_identify6c(int iHandle, int iAddress, IntPtr oData, ref byte oSize);// ap_identify6c - 识别6C卡号
        [DllImport("adpcom.dll")]
        public static extern int ap_identify6cmult(int iHandle, int iAddress, IntPtr oData, ref byte oSize);// ap_identify6cmult - 识别6C卡号-多卡
        [DllImport("adpcom.dll")]
        public static extern int ap_read6c(int iHandle, int iAddress, IntPtr oData, byte iMem, byte iAddr, byte iSize);// ap_read6c - 读6C数据
        [DllImport("adpcom.dll")]
        public static extern int ap_write6c(int iHandle, int iAddress, byte[] iData, byte iMem, byte iAddr, byte iSize);// ap_write6c - 写6C数据
        [DllImport("adpcom.dll")]
        public static extern int ap_encrypt(int iHandle, int iAddress);// ap_encrypt - 加密

        [DllImport("adpcom.dll")]
        public static extern int ap_softreset(int iHandle, int iAddress);// ap_softreset – 软重启设备

        [DllImport("adpcom.dll")]
        public static extern int ap_getautocard(int iHandle, IntPtr oData, ref byte oSize);//ap_getautocard - 获取自动上送卡号
        #endregion

        #region ---TCPIP---
        [DllImport("adpnet.dll")]
        public static extern int an_open(string ip, int port);
        [DllImport("adpnet.dll")]
        public static extern int an_close(int iHandle);
        [DllImport("adpnet.dll")]
        public static extern int an_getaddress(int iHandle, ref int oAddress, ref int oVer);
        [DllImport("adpnet.dll")]
        public static extern int an_setaddress(int iHandle, int iAddress, int iData);
        [DllImport("adpnet.dll")]
        public static extern int an_getconfig(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpnet.dll")]
```

```
        public static extern int an_setconfig(int iHandle, int iAddress, byte[] iData, byte iSize);
        [DllImport("adpnet.dll")]
        public static extern int an_gettcpip(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpnet.dll")]
        public static extern int an_settcpip(int iHandle, int iAddress, byte[] iData, byte iSize);
        [DllImport("adpnet.dll")]
        public static extern int an_identify6b(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpnet.dll")]
        public static extern int an_read6b(int iHandle, int iAddress, IntPtr oData, byte iAddr, byte iSize);
        [DllImport("adpnet.dll")]
        public static extern int an_write6b(int iHandle, int iAddress, byte[] iData, byte iAddr, byte iSize);
        [DllImport("adpnet.dll")]
        public static extern int an_identify6c(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpnet.dll")]
        public static extern int an_identify6cmult(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpnet.dll")]
        public static extern int an_read6c(int iHandle, int iAddress, IntPtr oData, byte iMem, byte iAddr, byte
iSize);
        [DllImport("adpnet.dll")]
        public static extern int an_write6c(int iHandle, int iAddress, byte[] iData, byte iMem, byte iAddr, byte
iSize);
        [DllImport("adpnet.dll")]
        public static extern int an_encrypt(int iHandle, int iAddress);
        [DllImport("adpnet.dll")]
        public static extern int an_getautocard(int iHandle, IntPtr oData, ref byte oSize);
        #endregion

        #region ---USB---
        [DllImport("adpusb.dll")]
        public static extern int ad_open();
        [DllImport("adpusb.dll")]
        public static extern int ad_close(int iHandle);
        [DllImport("adpusb.dll")]
        public static extern int ad_exitprogram(int iHandle);
        [DllImport("adpusb.dll")]
        public static extern int ad_getaddress(int iHandle, ref int oAddress, ref int oVer);
        [DllImport("adpusb.dll")]
        public static extern int ad_setaddress(int iHandle, int iAddress, int iData);
        [DllImport("adpusb.dll")]
        public static extern int ad_getconfig(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_setconfig(int iHandle, int iAddress, byte[] iData, byte iSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_getoutstatus(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_setoutstatus(int iHandle, int iAddress, byte[] iData, byte iSize);
```

```
        [DllImport("adpusb.dll")]
        public static extern int ad_identify6b(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_read6b(int iHandle, int iAddress, IntPtr oData, byte iAddr, byte iSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_write6b(int iHandle, int iAddress, byte[] iData, byte iAddr, byte iSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_identify6c(int iHandle, int iAddress, IntPtr oData, ref byte oSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_read6c(int iHandle, int iAddress, IntPtr oData, byte iMem, byte iAddr, byte
iSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_write6c(int iHandle, int iAddress, byte[] iData, byte iMem, byte iAddr, byte
iSize);
        [DllImport("adpusb.dll")]
        public static extern int ad_encrypt(int iHandle, int iAddress);
        [DllImport("adpusb.dll")]
        public static extern int ad_getautocard(int iHandle, IntPtr oData, ref byte oSize);
        #endregion
```