目录

OUR_MIFARE.dll 动态库使用说明

一、动态库简介

动态库 OUR_MIFARE.dll 用 VC 开发，编译成 32 位 Release 实体或 64 位 Release 实体，适用于 WIN2000、XP、2003、win7 的 32 或 64 位、win8 编译和运行环境。本文档下面的源代码例子是 C++的调用格式。

本动态库是本公司 USB 接口 IC 卡读写器的配套文件，必须和读写器一起使用。

OUR_MIFARE.dll 支持在软件运行中可以随时更换 USB 接口。本公司免驱型和有驱型读写器使用的 OUR_MIFARE.dll 不同，但函数的调用一样，如果客户程序开发好了，免驱型和有驱型互换时，只需更换对应的 OUR_MIFARE.dll。

推荐使用动态调用的方法使用本动态库。

本读写器支持的 IC 卡内部存储结构知识：

1、ISO14443 TYPE A 标准的 Mifare One 系列卡（含 S50 卡），容量为 1K 字节。Mifare One 卡的内部存储结构分为 16 个扇区，从第 0 区到第 15 区，每个扇区有独立的密码和权限控制字，可作为独立的电子钱包，每个扇区有 4 个块，序号为第 0 块、第 1 块、第 2 块、第 3 块，每块 16 个字节，第 3 块是卡的密码和权限控制字专用块，禁止在此存放数据。第 0 块、第 1 块、第 2 块可随意存放数据。但第 0 区的第 0 块已被固化了 IC 卡出厂信息，此块只能读出信息，不可更改。

每张卡都有一个全球统一的 4 个字节的序列号。许多公司销售的读卡器只有读取卡序列号的功能，也能用于一般的考勤系统。但这类读卡器跟本公司的读写器有着本质的不同，本公司的读写器不仅能读序列号，而且还有服务于 IC 卡所有用途的功能，比如选中卡、认证、读卡、写卡、改密码、休眠卡能功能。

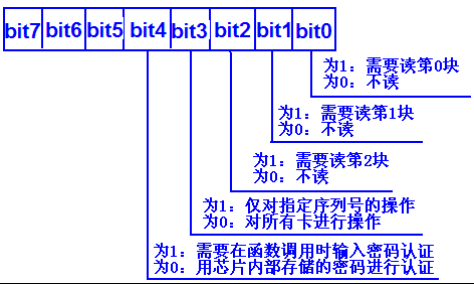
2、ISO14443 TYPE A 标准的 S70 卡，容量为 4K 字节。共 40 个扇区，也就是 40 个独立电子钱包，其中 0~31 扇区为普通区，区定义与以上的 S50 卡相同。32~39 区为大数据区，每区有 16 个块，第 15 块是卡的密码和权限控制字专用块，第 0~14 块为数据块，每块 16 个字节，共有 224 个字节可用。

二、易用函数

大部份软件、一卡通公司，只要使用以下二个函数完全可以达到软件需求，比如读出信息、读出金额、扣费等等。

轻松读卡：

函数名：piccreadex

功能	超强读卡，使用此函数可以一次性读整个区的第 0 块、第 1 块、第 2 块共 3 块的信息，并且返回卡序列号。
原始声明	unsigned char __stdcall piccreadex(unsigned char ctrlword,unsigned char *serial,unsigned char area,unsigned char keyA1B0,unsigned char *picckey,unsigned char *piccdata0_2)
输入	1、ctrlword 读卡的控制字，ctrlword 是一个字节，相当于八个位，每个位只有 0 和 1 两种状态： 

		推荐使用方法如下：
		//先定义以下常量
		#define BLOCK0_EN 0x01
		#define BLOCK1_EN 0x02
		#define BLOCK2_EN 0x04
		#define NEEDSERIAL 0x08
		#define EXTERNKEY 0x10
		举例：
		//以下控制字含义：读块 0、块 1、块 2，仅读指定序列号的卡，需要每次指定密码
		Ctrlword = BLOCK0_EN + BLOCK1_EN + BLOCK2_EN + NEEDSERIAL+ EXTERNKEY
		//以下控制字含义：读块 0、块 2，可读任意卡，需要每次指定密码
		Ctrlword = BLOCK0_EN + BLOCK2_EN + EXTERNKEY
		//以下控制字含义：读块 0、块 2，可读任意卡，启用芯片内部密码
		Ctrlword = BLOCK0_EN + BLOCK2_EN
		
		2、serial 只需指向一个至少已分配了 4 个 char 空间的可写数组 unsigned char * 指针，serial 的下标由 0 开始。如果在控制字中没有指定 NEEDSERIAL ，则 Serial 数组的内容无需赋值，因为此数组仅用于返回值。如果指定了 NEEDSERIAL，则必须为数组的内容赋值。
		3、area 是需要读出的区号，则 0-15 中的某个数。
		4、keyA1B0 为 0 时以 B 密码来认证，为非 0 时以 A 密码来认证。刚出厂的卡以 A 密码来认证。
		
		5、*picckey 指向存放卡密码的数组(6 个 char 的密码数组)。
		
		6、*piccdata0_2 是指向下标个数大于 48 的字节数组，用于存放 3 个块的数据，其中下标 0~15 存放作为读出的块 0 的数据，下标 16~31 存放作为读出的块 1 的数据，下标 32~47 存放作为读出的块 2 的数据，

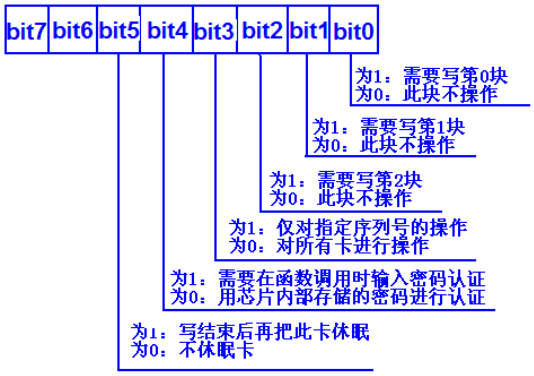
返回		返回 unsigned char 值，并将卡本块的数据传值到*piccdata 指向的数组中。
返回值说明	0	操作成功，读出的数据有效。
	1	0~2 块都没读出来，可能刷卡太块。但卡序列号已被读出来。
	2	第 0 块已被读出，但 1~2 块读取失败。卡序列号已被读出来。
	3	第 0、1 块已被读出，但 2 块读取失败。卡序列号已被读出来。
	8	寻卡错误，根本就没有卡在感应区，*serial 无效。
	9	有多张卡在感应区，寻卡过程中防冲突失败，*serial 无效。
	10	该卡可能已被休眠，无法选中，但卡序列号已被读出，*serial 数组中的数据有效。
	11	密码装载失败。
	12	密码认证失败。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注	请要本公司网站下载各种开发工具的例子源代码，本函数的声明和输入参数格式可以直接套用，谢谢使用	

释	

轻松写卡：

函数名：Piccwriteex

功能	超强写卡，使用此函数可以一次性写整个区的第 0 块、第 1 块、第 2 块信息。
原始声明	unsigned char __stdcall piccwriteex(unsigned char ctrlword,unsigned char *serial,unsigned char area, unsigned char keyA1B0,unsigned char *picckey,unsigned char *piccdata0_2)

输入	1、ctrlword 读卡的控制字，ctrlword 是一个字节，相当于八个位，每个位只有 0 和 1 两种状态： 
	推荐使用方法如下：
	//先定义以下常量
	#define BLOCK0_EN 0x01
	#define BLOCK1_EN 0x02
	#define BLOCK2_EN 0x04
	#define NEEDSERIAL 0x08
	#define EXTERNKEY 0x10
	#define NEEDHALT 0x20
	举例：
	//以下控制字含义：读块 0、块 1、块 2，仅写指定序列号的卡，需要每次指定密码，写成功后休眠卡
	Ctrlword = BLOCK0_EN + BLOCK1_EN + BLOCK2_EN + NEEDSERIAL+ EXTERNKEY + NEEDHALT
	//以下控制字含义：写块 0、块 2，可写任意卡，需要每次指定密码, 写成功后休眠卡
	Ctrlword = BLOCK0_EN + BLOCK2_EN + EXTERNKEY + NEEDHALT
	//以下控制字含义：写块 0、块 2，可写任意卡，启用芯片内部密码，写成功后休眠卡
	Ctrlword = BLOCK0_EN + BLOCK2_EN + NEEDHALT
	
	2、serial 只需指向一个至少已分配了 4 个 char 空间的可写数组 unsigned char * 指针，serial 的下标由 0 开始。如果在控制字中没有指定 NEEDSERIAL ，则 Serial 数组的内容无需赋值，因为此数组仅用于返回值。如果指定了 NEEDSERIAL，则必须为该数组的内容赋值。
	
	3、area 是需要准备写的区号，则 0-15 中的某个数。
	
	4、keyA1B0 为 0 时以 B 密码来认证，为非 0 时以 A 密码来认证。刚出厂的卡以 A 密码来认证。
	
	5、*picckey 指向存放卡密码的数组(6 个 char 的密码数组)。
	
	6、*piccdata0_2 是指向下标个数大于 48 的字节数组，用于存放 3 个块的数据，其中下标 0~15 存放作为准备写的块 0 的数据，下标 16~31 存放作为准备写的

		块 1 的数据，下标 32~47 存放作为准备写的块 2 的数据，
返回		返回 unsigned char 值，并将卡本块的数据传值到*piccdata 指向的数组中。
返回值说明	0	操作成功，写卡数据有效。
	1	0~2 块都没写进去，可能刷卡太块。
	2	第 0 块已写进去，但 1~2 块写失败。
	3	第 0、1 块已被写进去，但 2 块读写失败。
	8	寻卡错误，根本就没有卡在感应区，*serial 无效。
	9	有多张卡在感应区，寻卡过程中防冲突失败，*serial 无效。
	10	该卡可能已被休眠，无法选中，但卡序列号已被读出，*serial 数组中的数据有效。
	11	密码装载失败。
	12	密码认证失败
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释	请要本公司网站下载各种开发工具的例子源代码，本函数的声明和输入参数格式可以直接套用，谢谢使用	

以上函数是在已知卡密码的情况下操作。如果需要更改卡密码可通过 piccchangesinglekey 函数，快速地更改卡密码，piccchangesinglekey 详细介绍请查看下面第 12 页。

让读写器发出声音

函数名：pcdbeep

功能		让读写器发出声响。
原始声明		unsigned char __stdcall pcdbeep(unsigned long xms)
输入		xms 为响声的时间长度，单位为 2 毫秒
返回		返回 unsigned char 值。
返回值说明	0	操作成功，。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出

		现这个错误的。
	其他	未知错误
例子及注释	请要本公司网站下载各种开发工具的例子源代码，本函数的声明和输入参数格式可以直接套用，谢谢使用	

三、可作为软件加密狗，轻松实现自己软件的产权保护。

函数名：pcdgetdevicenumber

功能		返回本读写器独一无二的设备编号，此编号固化在芯片中，并通过加密的方式传输。根据此编号可在本公司网站查询是否真正为本公司的质保产品。因为可以返回全球唯一的设备编号，所以只需增加少量的算法，本读写器也可作软件加密狗用。
原始声明		unsigned char __stdcall pcdgetdevicenumber(unsigned char *devicenumber)
输入		devicenumber 只需指向一个至少已分配了 4 个 char 空间的可写数组 unsigned char *指针， serial 的下标由 0 开始。因为此数组仅用于返回设备编号。
返回		返回 unsigned char 值，并将寻到的卡的序列号传值到*serial 数组。
返回值说明	0	操作成功，* devicenumber 数组中的数据有效。
	12	读取设备编号失败。
	9	有多张卡在感应区，寻卡过程中防冲突失败，*serial 无效。
	10	该卡可能已被休眠，无法选中，但卡序列号已被读出，*serial 数组中的数据有效。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释	请要本公司网站下载各种开发工具的例子源代码，本函数的声明和输入参数格式可以直接套用，谢谢使用	

功能

四、其他函数大全

函数名：piccrequest

功能		寻卡并返回该卡的序列号
原始声明		unsigned char __stdcall piccrequest(unsigned char *serial)
输入		形参*serial 只需指向一个至少已分配了 4 个 char 空间的可写数组 unsigned char *指针， serial 的下标由 0 开始。Serial 数组的内容无需赋值，因为此数组仅用于返回值。
返回		返回 unsigned char 值，并将寻到的卡的序列号传值到*serial 数组。
返回值说	0	操作成功，*serial 数组中的数据有效。
	8	寻卡错误，根本就没有卡在感应区，*serial 无效。
	9	有多张卡在感应区，寻卡过程中防冲突失败，*serial 无效。
	10	该卡可能已被休眠，无法选中，但卡序列号已被读出，*serial 数组中的数据有

明		效。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误

例子及注释

```
        /*******************{{piccrequest 函数使用特例*********************************/
        //卡序列号缓冲
        unsigned char myserial[4];
        unsigned char status;
        //函数指针声明
        unsigned char (__stdcall *piccrequest)(unsigned char *serial);
        //提示当前目录
        AnsiString FileName=ExtractFilePath(Application->ExeName);
         //如果中没有"\"字符，就加进去
        if(FileName.SubString(FileName.Length(),1) != "\\")
        {
              FileName += "\\";
        }

        //调用读卡函数，如果没有寻到卡返回 1，拿卡太快返回 2，没注册发卡机返回 4，没有驱动程序返回 3
        FileName += "OUR_MIFARE.dll";
        if(!FileExists(FileName))
        {//如果文件不存在
              ShowMessageb("无法在应用程序的文件夹找到 IC 卡读写卡器动态库");
              return; //返回
        }
        HINSTANCE hDll;
        //加载动态库
        hDll=LoadLibrary(FileName.c_str());
        //提取动态库
        piccrequest = (unsigned char (__stdcall *piccrequest)(unsigned char *serial))GetProcAddress(hDll,"piccread");
        //调用函数
        status = piccrequest(myserial);
        //返回值处理
        switch(status)
        {
              case 0:
                    //TO-DO 相应的处理，请在以下加入代码
                    break;
              case 1:
                    break;
              //...
         }
```

注：原始声明指动态库的 VC 源码内的声明。

函数名：piccrequestex

功能		寻卡并选中指定序列号的 IC 卡，必须指定序列号
原始声明		unsigned char __stdcall piccrequestex (unsigned char *serial)
输入		形参*serial 只需指向一个至少已分配了 4 个 char 空间的数组 unsigned char *指针， serial 的下标由 0 开始, serial 数组的值为需要寻卡选卡的卡序列号。
返回		返回 unsigned char 值，并将寻到的卡的序列号传值到*serial 数组。
返回值说明	0	操作成功，*serial 数组中的数据有效。
	8	寻卡错误，根本就没有卡在感应区，*serial 无效。
	9	有多张卡在感应区，寻卡过程中防冲突失败，*serial 无效。
	10	该卡可能已被休眠，无法选中，但卡序列号已被读出，*serial 数组中的数据有效。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误

例子及注释		/**********{{piccrequestex 函数使用特例**********/
		//卡序列号缓冲
		unsigned char myserial[4];
		unsigned char status;
		//函数指针声明
		unsigned char (__stdcall *piccrequestex)(unsigned char *serial);
		//提示当前目录
		AnsiString FileName=ExtractFilePath(Application->ExeName);
		//如果中没有"\"字符，就加进去
		if(FileName.SubString(FileName.Length(),1) != "\\")
		{
		FileName += "\\";
		}
		
		//调用读卡函数，如果没有寻到卡返回 1，拿卡太快返回 2，没注册发卡机返回 4，没有驱动程序返回 3
		FileName += "OUR_MIFARE.dll";
		if(!FileExists(FileName))
		{//如果文件不存在
		ShowMessageb("无法在应用程序的文件夹找到 IC 卡读写卡器动态库");
		return; //返回
		}
		HINSTANCE hDll;
		//加载动态库
		hDll=LoadLibrary(FileName.c_str());
		//提取动态库
		piccrequestex = (unsigned char (__stdcall ***piccrequestex**)(unsigned char *serial))GetProcAddress(hDll," piccrequeste");

```
//调用函数
Myserial[0] = 0x18;
Myserial[1] = 0x18;
Myserial[2] = 0x18;
Myserial[3] = 0x18;
//调用函数
status = piccrequestex (myserial);
//返回值处理
switch(status)
{
        case 0:
                //TO-DO 相应的处理，请在以下加入代码
                break;
        case 1:
                break;
        //...
 }
```

函数名：piccauthkey1

功能		密码认证方式 1,用外部密码认证,必须指定外部密码。本函数必须在 piccrequest 或 piccrequestex 函数执行之后运行，并且要紧接着调用，中途不能调用其他函数。
原始声明		unsigned char __stdcall piccauthkey1(unsigned char *serial,unsigned char area,unsigned char keyA1B0,unsigned char *picckey)
输入		1、*serial 指向存放选中卡序列号的数组，此序列号必须是选中卡的。 2、area 是需要认证的区号，0-15 中的某个数。 3、keyA1B0 为 0 时以 B 密码来认证，为非 0 时以 A 密码来认证。刚出厂的卡以 A 密码来认证。 4、*picckey 指向存放卡密码的数组(6 个 char 的密码数组)。
返回		返回 unsigned char 值。
返回值说明	0	操作成功，该卡的密码已认证通过，可以进运读写操作了。
	11	密码装载失败。
	12	密码认证失败。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释		

函数名：piccauthkey2

功能	密码认证方式 2，用芯片内部密码认证，该密码存放在芯片的只写区域（用

		pcdwritekeytoe2 写密码），认证时通过芯片内部纳米线路传输，保密性极强，能小心剖开芯片，再用连上纳米线路拦截恐怕只有请外星人出马了。本函数必须在 piccrequest 或 piccrequestex 函数执行之后运行，并且要紧接着调用，中途不能调用其他函数。
原始声明		unsigned char __stdcall piccauthkey2(unsigned char *serial,unsigned char area,unsigned char keyA1B0)
输入		1、*serial 指向存放选中卡序列号的数组(4 个 char)，此序列号必须是选中卡的。 2、area 是需要认证的区号，0-15 中的某个数。 3、keyA1B0 为 0 时以 B 密码来认证，为非 0 时以 A 密码来认证。刚出厂的卡以 A 密码来认证。
返回		返回 unsigned char 值。
	0	操作成功，该卡的密码已认证通过，可以进运读写操作了。
	11	密码装载失败。
	12	密码认证失败。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释		

函数名：pcdwritekeytoe2

功能		将密码写入芯片内部保密性极高的只写区域，此函数写入密码仅仅是为了 piccauthkey2 函数的使用。
原始声明		unsigned char __stdcall pcdwritekeytoe2(unsigned char area,unsigned char keyA1B0,unsigned char *picckey)
输入		1、area 是需要认正的区号，0-15 中的某个数。 2、keyA1B0 为 0 时以 B 密码来认证，为非 0 时以 A 密码来认证。刚出厂的卡以 A 密码来认证。 3、*picckey 指向存放卡密码的数组(6 个 char 的密码数组)。
返回		返回 unsigned char 值。
	0	写密码成功
	15	写密码错误
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会

		出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释		

函数名：piccread

功能		读出一块的数据，也就是 16 个字节。必须在执行 piccrequest 或 Piccrequestex 函数，接着执行 piccauthkey1 或 piccauthkey2 函数，然后执行 piccread 才能成功读出一块的数据。
原始声明		unsigned char __stdcall piccread(unsigned char block,unsigned char *piccdata)
输入		1、block 是 IC 卡的绝对块号，当需要读 IC 卡的第 x 区的第 y 块时，绝对块号必须是 block = x * 4 + y。 2、*piccdata 是指向下标个数大于 16 的数组，作为返回 16 个字节的卡数据的缓存。
返回		返回 unsigned char 值，并将卡本块的数据传值到*piccdata 指向的数组中。
返回值说明	0	操作成功，读出的数据有效。
	13	读本块失败，原因是本块所对应的区还没通过密码认证。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释		

函数名：piccwrite

功能	写入一块的数据，也就是 16 个字节。必须在执行 piccrequest 或 Piccrequestex 函数，接着执行 piccauthkey1 或 piccauthkey2 函数，然后执行 piccwrite 才能成功写进一块的数据。
原始声明	unsigned char __stdcall piccwrite(unsigned char block,unsigned char *piccdata)
输入	1、block 是 IC 卡的绝对块号，当需要读 IC 卡的第 x 区的第 y 块时，绝对块号必须是 block = x * 4 + y。 2、*piccdata 是指向下标个数大于 16 的数组，特别提醒在调用 piccwrite 之前，必须对 piccdata 数组明确赋值，千万不能写进不明数据，特别对是存放卡权限的第 3 块，更要明确写入，否则极有可能导致卡作废。
返回	返回 unsigned char 值。

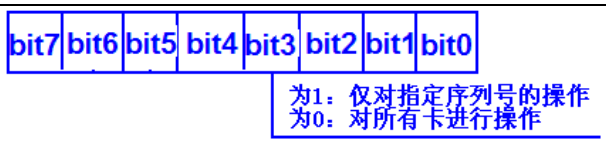
返回值说明	0	操作成功，写进数据有效。
	14	写本块失败，原因是本块所对应的区还没通过密码认证。。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释		

函数名：picchalt

功能		休眠选中卡，卡一旦被休眠，将不能再次被选中、不能被读被写。必须得拿卡离开感应区，再次放卡才能再次选中和读写卡。休眠卡功能一般用在只能操作一次的情况下，比如刷一次卡只扣一次钱的情况。
原始声明		unsigned char __stdcall picchalt()
输入		不用输入
返回		返回 unsigned char 值。
返回值说明	0	操作成功，。
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释		

函数名：piccchangesinglekey

功能	改单区密码。
原始声明	unsigned char __stdcall piccchangesinglekey(unsigned char ctrlword,unsigned char *serial,unsigned char area , unsigned char keyA1B0,unsigned char *piccoldkey,unsigned char *piccnewkey)
输入	1、ctrlword 读卡的控制字，ctrlword 是一个字节，相当于八个位，每个位只有 0 和 1 两种状态：

<table>
<tr><td colspan="2"></td><td>


推荐使用方法如下：
//先定义以下常量
#define NEEDSERIAL　0x08
举例：
//以下控制字含义：仅更改指定序列号的卡密码
Ctrlword = NEEDSERIAL
//以下控制字含义：更改任意卡密码
Ctrlword = 0

2、serial 只需指向一个至少已分配了 4 个 char 空间的可写数组 unsigned char * 指针，serial 的下标由 0 开始。如果在控制字中没有指定 NEEDSERIAL ，则 Serial 数组的内容无需赋值，因为此数组仅用于返回值。如果指定了 NEEDSERIAL，则必须为该数组的内容赋值。

3、area 是需要准备写的区号，则 0-15 中的某个数。

4、keyA1B0 为 0 时以 B 密码来认证，为非 0 时以 A 密码来认证。刚出厂的卡以 A 密码来认证。

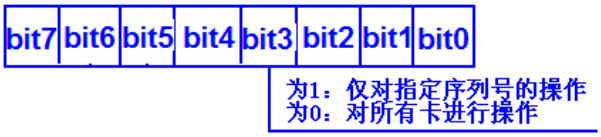
5、* piccoldkey 指向存放卡原始密码的数组(6 个 char 的密码数组)。

6、* piccnewkey 指向存放卡新密码(也就是准备改成的密码)的数组(6 个 char 的密码数组)。</td></tr>
<tr><td colspan="2">返回</td><td>返回 unsigned char 值，并将卡本块的数据传值到*piccdata 指向的数组中。</td></tr>
<tr><td rowspan="12">返回值说明</td><td>0</td><td>操作成功，写卡数据有效。</td></tr>
<tr><td>8</td><td>寻卡错误，根本就没有卡在感应区，*serial 无效。</td></tr>
<tr><td>9</td><td>有多张卡在感应区，寻卡过程中防冲突失败，*serial 无效。</td></tr>
<tr><td>10</td><td>该卡可能已被休眠，无法选中，但卡序列号已被读出，*serial 数组中的数据有效。</td></tr>
<tr><td>11</td><td>密码装载失败。</td></tr>
<tr><td>12</td><td>密码认证失败</td></tr>
<tr><td>13</td><td>刷卡太快</td></tr>
<tr><td>14</td><td>刷卡太快</td></tr>
<tr><td>22</td><td>动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。</td></tr>
<tr><td>24</td><td>操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。</td></tr>
<tr><td>27</td><td>USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。</td></tr>
<tr><td>28</td><td>USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。</td></tr>
<tr><td>其他</td><td>未知错误</td></tr>
<tr><td>例子及注释</td><td colspan="2">请要本公司网站下载各种开发工具的例子源代码，本函数的声明和输入参数格式可以直接套用，谢谢使用</td></tr>
</table>

修改卡片扇区密码

函数名：piccchangesinglekeyex

功能		改单区密码，可同时修改 A 密码，或密码权限访问字(密码权限编码)或 B 密码。
原始声明		unsigned char __stdcall piccchangesinglekeyex(unsigned char ctrlword,unsigned char *serial,unsigned char area,unsigned char keyA1B0,unsigned char *piccoldkey,unsigned char *piccdata)

输入		1、ctrlword 读卡的控制字，ctrlword 是一个字节，相当于八个位，每个位只有 0 和 1 两种状态：
		 
		
		推荐使用方法如下：
		//先定义以下常量
		#define NEEDSERIAL　0x08
		举例：
		//以下控制字含义：仅更改指定序列号的卡密码
		Ctrlword = NEEDSERIAL
		//以下控制字含义：更改任意卡密码
		Ctrlword = 0
		
		2、serial 只需指向一个至少已分配了 4 个 char 空间的可写数组 unsigned char * 指针，serial 的下标由 0 开始。如果在控制字中没有指定 NEEDSERIAL ，则 Serial 数组的内容无需赋值，因为此数组仅用于返回值。如果指定了 NEEDSERIAL，则必须为该数组的内容赋值。
		
		3、area 是需要准备写的区号，则 0-15 中的某个数。
		
		4、keyA1B0 为 0 时以 B 密码来认证，为非 0 时以 A 密码来认证。刚出厂的卡以 A 密码来认证。
		
		5、* piccoldkey 指向存放卡原始密码的数组(6 个 char 的密码数组)。
		
		6、* piccdata 指向存放卡新 A 密码(也就是准备改成的密码)、密码权限访问字、新 B 密码的数组、指定更改项目的标志(17 个 char 的数组)。其中新 A 密码 6 个字节；密码权限访问字 4 个字节；新 B 密码 6 个字节；指定更改项目的标志为 1 个字节，这个字节为 3 是表示同时更改 A、B、 密码权限访问字，为 2 表示密码权限访问字不更改，只改 A、B 密码，为 0 表示只改 A 密码

返回		返回 unsigned char 值，并将卡本块的数据传值到*piccdata 指向的数组中。
返回值说明	0	操作成功，写卡数据有效。
	8	寻卡错误，根本就没有卡在感应区，*serial 无效。
	9	有多张卡在感应区，寻卡过程中防冲突失败，*serial 无效。
	10	该卡可能已被休眠，无法选中，但卡序列号已被读出，*serial 数组中的数据有效。
	11	密码装载失败。
	12	密码认证失败
	13	刷卡太快
	14	刷卡太快
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确

		的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	30	密码权限访问字(密码权限编码)校验错误，不允许更改，以免造成卡永久损坏。
	其他	未知错误
例子及注释	请要本公司网站下载各种开发工具的例子源代码，本函数的声明和输入参数格式可以直接套用，谢谢使用	

写 UID 卡号（0 区 0 块）

函数名：piccwriteserial

功能		写 UID 卡的 0 扇区第 0 块，总计 16 个字节。 。
原始声明		unsigned char __stdcall piccwriteserial (unsigned char ctrlword,unsigned char *serial, unsigned char keyA1B0,unsigned char *picckey,unsigned char *piccdata)
输入		1、*piccdata 是指向下标个数为 16 的数组，前 4 个字节为要写入的卡号，第 5 字节为前 4 个字节的异或和。
返回		返回 unsigned char 值。
返回值说明	0	操作成功，写进数据有效。
	18	非 UID 卡
	22	动态库或驱动程序异常，解决方法是退出程序，拔出 IC 卡读写器，重装驱动程序再插上 IC 卡读写器重试，或者重新拷贝动态库 OUR_MIFARE.dll 到正确的位置。
	24	操作超时。可能是电脑中毒导致 USB 帧传递调度缓慢，或者是 IC 卡读写器有问题，解决方法是重启电脑或重新拔插 IC 卡读写器。
	27	USB 传输不稳定导致传输的字符不全。不需理会这个错误，因为基本上是不会出现这个错误的。
	28	USB 传输不稳定导致 CRC 校验错。不需理会这个错误，因为基本上是不会出现这个错误的。
	其他	未知错误
例子及注释		