NLite22G 多功能加密锁 API

开发手册

广州耐特电子有限公司

2008 年 8 月

广州耐特电子有限公司尽最大努力使这篇文章中的内容完善且正确。对于由这篇文档导致的任何形式的直接或间接损失不负有责任。这篇文章的内容会跟随产品的升级而有所变化。

Tel：(020) 82315664　39885802　　Fax：(020) 82315664-803　　http:// www.NLite.net.cn

软件开发协议

广州耐特电子有限公司（以下简称耐特电子）的所有产品，包括但不限于：开发工具包，磁盘，光盘，硬件设备和文档，以及未来的所有定单都受本协议的制约。如果您不愿接受这些条款，请在收到后的 7 天内将开发工具包寄回耐特电子，预付邮资和保险。我们会把货款退还给您，但要扣除运费和适当的手续费。

1. 许可使用

您可以将本软件合并、连接到您的计算机程序中，但其目的只是保护该程序。您可以以存档为目的复制合理数量的拷贝。

2. 禁止使用

除在条款 1 中特别允许的之外，不得复制、反向工程、反汇编、反编译、修改、增加、改进软件、硬件和产品的其它部分。禁止对软件和产品的任何部分进行反向工程，或企图推导软件的源代码。禁止使用产品中的磁性或光学介质来传递、存储非本产品的原始程序或由耐特电子提供的产品升级的任何数据。禁止将软件放在服务器上传播。

3. 有限担保

耐特电子保证在自产品交给您之日起的 12 个月内，在正常的使用情况下，硬件和软件存储介质没有重大的工艺和材料上的缺陷。

4. 修理限度

当根据本协议提出索赔时，耐特电子唯一的责任就是根据耐特电子的选择，免费进行替换或维修。耐特电子对更换后的任何产品部件都享有所有权。

保修索赔单必须在担保期内写好，在发生故障 14 天内连同令人信服的证据交给耐特电子。当将产品返还给耐特电子或耐特电子的授权代理商时，须预付运费和保险。

除了在本协议中保证的担保之外，耐特电子不再提供特别的或隐含的担保，也不再对本协议中所描述的产品负责，包括它们的质量，性能和对某一特定目的的适应性。

5. 责任限度

不管因为什么原因，不管是因合同中的规定还是由于刑事的原因，包括疏忽的原因，而使您及任何一方受到了损失，由我方产品所造成的损失或该产品是起诉的原因或与起诉有间接关系，耐特电子对您及任何一方所承担的全部责任不超出您购买该产品所支付的货款。在任何情况下，耐特电子对于由于您不履行责任所导致的损失，或对于数据、利润、储蓄或其它的后续的和偶然的损失，即使耐特电子被建议有这种损失的可能性，或您根据第 3 方的索赔而提出的任何索赔均不负责任。

6. 协议终止

当您不能遵守本协议所规定的条款时，将终止您的许可和本协议。但条款 2, 3，4，5 将继续有效。

广州耐特电子有限公司

地址:广州市天河区中山大道 268 号天河广场天盛 25F
电话:020-82315664 39885802 **传真**: 020-39885802-803
网址: http://www.NLite.net.cn

产品列表

多功能加密锁	
NLite101	NLite101 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；用户密码最大长度为 128 位，提供 256 个字节读写空间;提供 64 位自定产品编码；Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。
NLite112	NLite112 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；提供**双操作员**即时超级用户及普通用户，超级用户可自定义各操作员的密码校验次数；每个密码最大长度为 128 位，提供 **512 个字节**读写空间;提供 64 位自定产品编码；提供自己定义 **34 个字节的设备制造商信息**。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。
NLite124	NLite124 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；提供双操作员即时超级用户及普通用户，超级用户可自定义各操作员的密码校验次数；每个密码最大长度为 128 位，提供 **1024 个字节大容量读写空间**; 超级用户可以对各块（每 64 字节为一块）的**用户读写权限进行设置**,提供 64 位自定产品编码；提供 OTP 设置，提供自己定义 34 个字节的设备制造商信息。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。
NLite132	NLite132 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；用户密码最大长度为 128 位，提供 512 个字节读写空间;提供 64 位自定产品编码；这个产品最大特点是提供**多用户多扇区**操作，即是提供一个超级用户加四个普通用户,每个普通用户只能操作自己所对扇区 128 字节。方便实现”**一锁通**”,即用一个加密锁可以加密**多个产品**或一个加密锁提供给**多个开发厂商**同时使用。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。
NLite141	NLite141 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；提供双操作员即时超级用户及普通用户，超级用户可自定义各操作员的密码校验次数；可自定义软件使用模式（试用时间，试用次数，试用周期，试用天数，及超级模式）；每个密码最大长度为 128 位，提供 56 个字节读写空间; 超级用户可以对普通**用户读写权限进行设置**,提供 64 位自定产品编码；提供自己定义 34 个字节的设备制造商信息；提供检测失效机制。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。

多功能加密锁

NLite22F	NLite22F 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；提供双操作员即时超级用户及普通用户，超级用户可自定义各操作员的密码校验次数；每个密码最大长度为 128 位，提供 **4096 个字节大容量读写空间**；超级用户可以设置前 16 块（每 64 字节为一块）的用户读写权限,提供 64 位自定产品编码；提供自己定义 34 个字节的设备制造商信息。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。
NLite23F	NLite23F 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；用户密码最大长度为 128 位，提供 4096 个字节读写空间;提供 64 位自定产品编码；这个产品最大特点是提供**多用户多扇区**操作，即是提供一个超级用户加四个普通用户,每个普通用户只能操作自己所对扇区 1024 字节。方便实现”**一锁通**”,即用一个加密锁可以加密**多个产品**或一个加密锁提供给**多个开发厂商**同时使用。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。
NLite22G	NLite22F 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；提供双操作员即时超级用户及普通用户，超级用户可自定义各操作员的密码校验次数；每个密码最大长度为 128 位，提供 **8192 个字节超大容量读写空间**；超级用户可以设置前 16 块（每 64 字节为一块）的用户读写权限,提供 64 位自定产品编码；提供自己定义 34 个字节的设备制造商信息。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。
NLite23G	NLite23G 加密锁有 48 位全球唯一硬件 ID，支持在同一台 PC 上插入多把加密锁；用户密码最大长度为 128 位，提供 8192 个字节读写空间;提供 64 位自定产品编码；这个产品最大特点是提供**多用户多扇区**操作，即是提**供一个超级用户加八个普通用户**,每个普通用户只能操作自己所对扇区 1024 字节。方便实现”**一锁通**”,即用一个加密锁可以加密**多个产品**或一个加密锁提供给**多个开发厂商**同时使用。Windows 桌面系列、WINCE、Linux 等多种操作系统平台下都无需要安装驱动程序。提供 DLL、LIB、COM、Active、Http 接口程序及多种实例程序；支持程序、多媒体、网站、文档等加密消费等开发应用。

新产品在不断增加中,更多产品信息,请登录 http://www.NLite.net.cn 网站或直接联系我们
(+86)020-82315664　　39885802

目录

1 NLite-22G 概述

NLite-22G 加密锁是一款可以支持软件保护应用和身份认证应用的大容量增强型加密锁，提供 **8192 字节超大容量存储空间**，为前 16 块数据（每块数据为 64 字节，共分为 16 块）**提供读写权限控制**。免驱动的 USB 设备。由于该加密锁使用**双用户**对加密锁权限进行管理，即是超级用户及普通用户，超级用户拥有加密锁的全部权限(一般这个密码由开发商掌握)，普通用户只能执行匿名权限操作、获取当前读写权限操作、读写操作；对于受保护的软件，通过它使软件更安全，可以保护该软件不被非法复制和非授权访问或使用。当使用多功能锁加密保护您的软件后，启动所加密保护的程序时，此时若多功能锁不存在或对某个应用模块的访问已超过预先设定的次数，程序会发出错误信息，从而终止，这就达到了加密保护软件的目的。NLite-22G 多功能锁还可以应用在各种安全系统身份认证领域，包括网站系统、OA 办公系统、信息查询系统等。通过 NLite-22G 多功能锁的使用替换传统的用户名和密码，保证了系统的登录安全。NLite-22G 多功能锁提供多种 API 接口调用方式。

- **LIB 静态库方式** 该方式好处是开发后编译的文件无须附带其它文件，也无须向系统注册相关文件。不好的地方是现在只支持 **Visual C++, Boland C++**两种开发语方；操作方法是在开发软件时把 NT22G.LIB 及 NT22G.H 文件包含到工程中，在程序中调用各个函数，最后编译工程就可以了。
- **DLL 动态库方式** 该方式是大多数编程序语言所支持的一种开发方式，用户只需在程序中进行声明或引用并调用相应的函数即可，程序在运行时必须保证该 **NT22G.DLL** 文件在系统目录下或与 **EXE** 文件在同一目录下。
- **COM 组件方式** 该方式也是绝大大多数编程语言所支持的开发方式 **COM** 组件是提供了所有的应用程序 **API**，在运行时必须保证该 **NT22GC.DLL** 组件存在并已注册。
- **Active 控件方式** 该方式也是绝大大多数编程语言所支持的开发方式 **OCX** 控件是提供了所有的应用程序 **API**，在运行时必须保证该 **NT22G.OCX** 控件存在并已注册。

本文档详细主要说明API函数的接口定义以及API中的常量定义，有关NLite-22G多功能锁的详细介绍，请参阅《NLite-22G多功能锁用户手册》。

2 产品专用术语介绍

2.1 产品编号

产品编码主要是提供给使用加锁的软件开发厂商作为为了**区分不同的软件产品**使用的不同编号加密锁。产生产品编号的方法是，由软件厂商输入 1 至 56 位字符长度的产品密码，然后把产品密码送到加密锁中，由加密锁调用内部**硬件 3DES** 进行运算，并把新的 16 字节产品编码设置加密锁中该过程是不可逆。也就是说只有知道产品密的的人才可以产生该产品编号，所以产品编号还有一个功能是**防止加密锁被复制**。该加密锁生成产品编码的权限为超级用户权限，要生成产品编号除了要知道产品密码外还要知道超级用户密码(16 字节)，所以该型号的加密锁**更难复制**。

2.2 设备编号

设备编号是由加密锁，加密芯片在制作过程中固化在加密芯片中的一组硬件编号。不会产生相同的编号，全球唯一。该编号可以用作跟踪各个加密设备使用情况，如防代理商窜货等等功能。也可以用来作为加密算法中的一个加密条件，增强产品的安全性。

2.3 产品密码

产品密码是用来产生产品编号的一个字符串，它的有效范围 1-56 个字符串，只要输入顺序及长度有所变化则产生的产品编码将会面目全非,所在请您在设置产品编号是一定要记住这个密码,否则我们也爱莫能助。

2.4 超级用户密码

超级用户密码是保障超级权限的一个密码，由于这个密码权限非常高级，一般应掌握在开发商核心管理人员手中。如果校验的密码不正确，加密锁的操作权限只能进行各种匿名权限操作。如果校验正确，则加密锁将把权限提升超级用户权限，可以进行全部无限制操作，如设置显示信息、设置产品编号、修改普通用户密码，设置密码校验次数、读写数据等。

2.5 普通用户密码

普通用户密码是保障普通用户权限的一个密码。如果校验的密码不正确，加密锁的操作权限只能进行各种匿名权限操作。如果校验正确，则加密锁将把权限提升普通用户权限，可以修改普通用户密码，读写数据及各种种匿名权限操作等。

2.6 产品显示信息

产品显示信息，主要是在设备管理器中属性页,显示如图 2-2 设备属性页显示信息或在第一次使用加密锁时在 WINDOWS 右下角弹出的显示生产厂商信息如图 2-1 插入时显示信息。我们在这里为开发商提供显示加密锁个性信息或广告信息，而不是我们生产厂商信息；为软件开发商提供更好的 OEM 软件服务。

显示信息为 17 个汉字或 17 个字符串，请输入信息不要超过该长度。

图 2-1 插入时显示信息

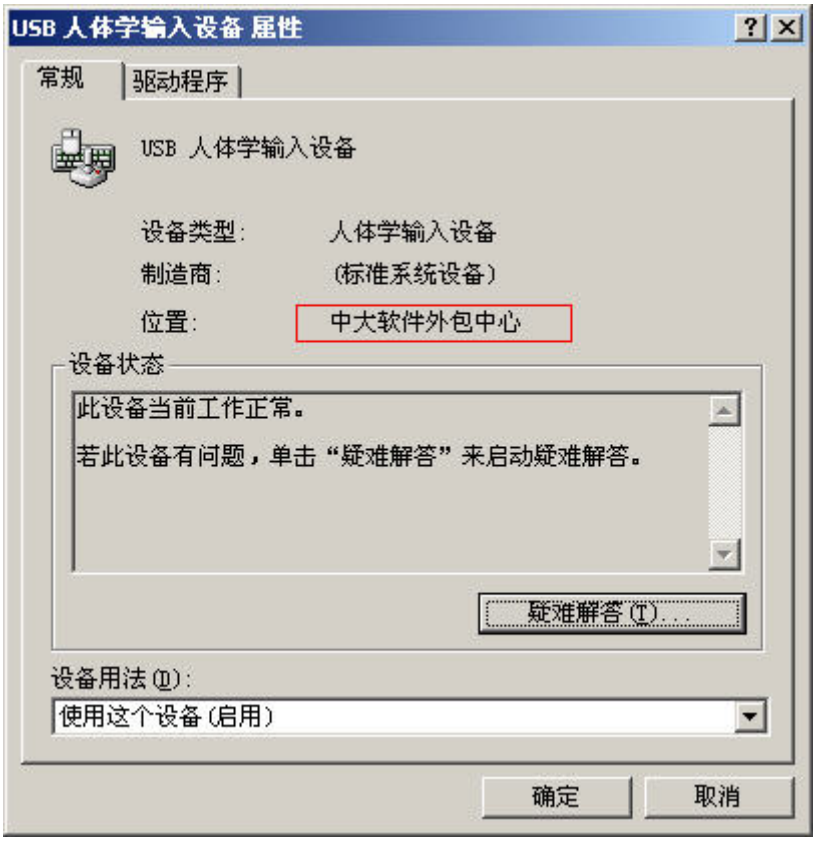

图 2-2 设备属性页显示信息

2.7 密码校验次数

密码校验次数是校验密码是连续错误最大次数，每校验错一次则加密锁自动减一，当减至 0 次时则对该密码锁死，如果在未锁死之前校验正确则自动恢复密码校验次数为最大次；例如设用户密码校验次为 3 次，如果连续执行 3 次校验错误，则加锁则把用户密码锁死。如果在第 3 或第 2 次时输入正确，则把校验次数恢复到 3 次。

当普通用户密码锁死后，可以通过校验超级用户进行恢复。如果超级密户密码锁死，该加密锁只能报废。当密码校验次数设为 0 次时则为不限制错误次数。即是可以任意校验。

2.8 读写权限

读写权限主是超级用户设置普通用户对存储空间的数据块写操作进行限制，增强产品的安全性。超级用户可以任何时候读、写任意数据块数据。普通用户则授该读写权限限制，如果某

块数数据为只读操作时(权限值为 0 时)，普通用户只能进行读数据，而不能写数据。

读写权限的功能一般是软件开发商，在加密锁上设置一些关键数据(如代商信息、开发口令、各种授权编码等)，而该数据不让用户权限进行修改。则应把相关数据的权限块设置为只读。

读写权限的设置只有超级用户能够进行设置。

3 调用流程图

API 函数调用的流程图如图 3-1 函数调用总体流程图，具体的各个操作请参考相应的例子程序。

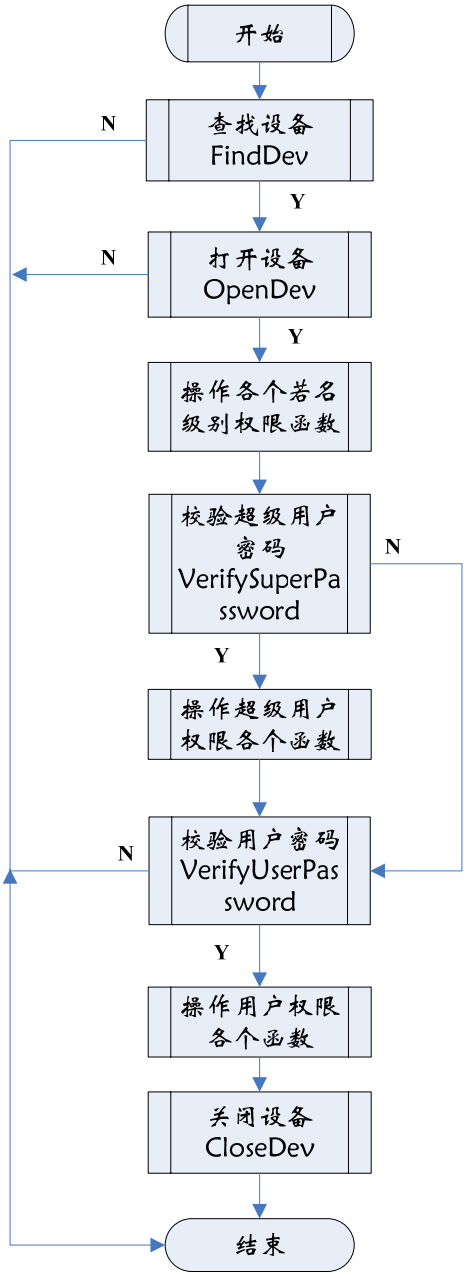

图 3-1 函数调用总体流程图

4 产品出厂默认设置

- 产品编号(PID)初始值为 16 个字符“0”，即是:“0000000000000000”；
- 超级用户密码(SPIN)实始值为 16 个字符“8”，即是:“8888888888888888”；
- 普通用户密码(UPIN)实始值为 16 个字符“8”，即是:“8888888888888888”；
- 显示信息为“www.NLite.net.cn”；
- 读写权限为全部可以读写，没有进行只读限制,即是:“1111111111111111”。
- 超级用户校验次数为 0 次即是不限次数，普通用户密码校验次数为 0 次也不限次数；
- 数据存储空间全为 0XFF；

5 API 常量说明

- OP_OK=0 表示操作成功;
- MODE_ERR=1 表示当前操作权限不够;
- EEPROM_ERR=2 表示当前写数据失败(可能当前写的范围里有只读块);
- PIN_ERR=3 表示校验密码失败;
- CHK_ZERO=4 表示没有校验次数(加密位已速烧断);
- PARA_ERR=5 表示输入的参数不正确，请检查输入参数;
- CMD_ERR=6 表示输入的命令错误，或当前设备不支持该命令;
- SEND_ERR=-1 表示发送数据失败,请检查 USB 是否已接收;
- REC_ERR=-2 表示接收数据失败，请检查 USB 是否已接收;
- EN_ERR=-3 表示写数据失败，请检查输入数据是否正确
- DE_ERR=-4 表示读数据失败,请重新操作一次;
- READ_ERR=-5 表示读数据失败;
- OPEN_ERR=-6 表示设备已打开,请不要再打开设备;
- OPENNOT_ERR=-7 表示设备没有打开，请重新打开设备;
- PASS_ERR=-8 表示校验密码错误;

6 API 函数说明

6.1 设置通信参数设置 NT22GFindDev

函数原型	short __stdcall NT22GFindDev(const unsigned char *pPidData);
功能	该接口主要功能是查找指定产品标识(PID)的在线设备设备数
输入参数	pPidData 16 字节产品标识(PID)
输出参数	无
权限类别	匿名
返回值	查找到设备个数（>0 为正确）

使用示例	int m_count;
	unsigned char PidData[16];
	CString Str;
	
	Str="0000000000000000";
	memcpy(PidData,m_EFindProductID,sizeof(PidData));
	m_count=NT22GFindDev(PidData);
	if(m_count==0)
	{
	AfxMessageBox("没有设备!", MB_ICONINFORMATION);
	return;
	}

6.2 打开设备 NT22GOpenDev

函数原型	HANDLE __stdcall NT22GOpenDev(const unsigned char *pPidData,short Index);
功能	该接口主要功能是打开指定索引的设备。
输入参数	pPidData 16 字节产品标识字符串 Index 相应的设备索引
输出参数	无
权限类别	匿名
返回值	相应的设备操作句柄（>0 为正确）

使用示例	short Index;
	unsigned char PidData[16];
	CString Str;
	
	Index=0;
	Str="0000000000000000";
	memcpy(PidData,m_EFindProductID,sizeof(PidData));
	Index=m_ComDevList.GetCurSel();
	handle=NT22GOpenDev(PidData,Index);
	if((int)handle<1)
	{
	AfxMessageBox("打开设备失败!", MB_ICONINFORMATION);
	return;
	}

6.3 打开指示灯 NT22GLedOpen

函数原型	short __stdcall NT22GLedOpen(HANDLE handle);
功能	打开当前设备指示灯
输入参数	handle 操作设备句柄
输出参数	无
权限类别	匿名
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	m_st=NT22GLedClose(handle); //LED灭
	if(m_st ==0)
	{
	AfxMessageBox("打开指示为成功!", MB_ICONINFORMATION);
	return;
	}

6.4 关闭指示灯 NT22GLedClose

函数原型	short __stdcall NT22GLedClose(HANDLE handle);
功能	关闭当前设备指示灯
输入参数	handle 操作设备句柄
输出参数	无
权限类别	匿名
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	
	m_st=NT22GLedClose(handle); //LED灭
	if(m_st ==0)
	{
	AfxMessageBox("关闭指示为成功!", MB_ICONINFORMATION);
	return;
	}

6.5 获取设备编号 NT22GGetDID

函数原型	short __stdcall NT22GGetDID(HANDLE handle,unsigned char *pDidData);
功能	取产品设备编号
输入参数	handle 操作设备句柄
输出参数	pDidData 12 字节设备编号
权限类别	匿名
返回值	返回操作状态(0 为正确)

使用示例	unsigned char DevData[12];
	Short m_st;
	
	m_st=NT22GGetDID(handle,DevData); //取DID值
	if(m_st==0)
	{
	AfxMessageBox("获取设备编号成功!", MB_ICONINFORMATION);
	return;
	}

6.6 获取产品标识 NT22GGetPID

函数原型	short __stdcall NT22GGetPID(HANDLE handle,unsigned char *pPidData);
功能	获取产品标识编号
输入参数	handle 操作设备句柄
输出参数	pPidData 16 字节产品编号
权限类别	匿名
返回值	返回操作状态(0 为正确)
使用示例	unsigned char PidData[16]; Short m_st;

	m_st=NT22GGetPID(handle,PidData);
	if(m_st==0)
	{
	AfxMessageBox("获取产品标识编号成功!", MB_ICONINFORMATION);
	return;
	}

6.7 校验超级用户 NT22GVerifySuperPassword

函数原型	short __stdcall NT22GVerifySuperPassword(HANDLE handle,const unsigned char *pSuperPassword,short nLen);
功能	这里的主要功能是校验超级用户密码，校验正确则把当前操作权限提升级为超级用户级
输入参数	handle 操作设备句柄, pSuperPassword 最大 16 字节密码, nLen 密码长度
输出参数	无
权限类别	匿名
返回值	密码的校验次数(>0 正确)

使用示例	short VerifyNum;
	unsigned char Password[16];
	Short m_st;
	CString m_EVerifyPass;
	
	m_EVerifyPass="8888888888888888";
	memcpy(Password,m_EVerifyPass,m_EVerifyPass.GetLength());
	VerifyNum= NT22GVerifySuperPassword (handle,Password,m_EVerifyPass.GetLength());
	if(VerifyNum==0)
	{
	AfxMessageBox("对不起，密码校验次数为0!", MB_ICONINFORMATION);
	return;
	}
	
	if(VerifyNum<0)
	{
	AfxMessageBox("对不起，密码校验失败!", MB_ICONINFORMATION);
	return;
	}
	
	AfxMessageBox("校验密码成功!",MB_OK|MB_ICONINFORMATION);

6.8 校验用户密码 NT22GVerifyUserPassword

函数原型	short __stdcall NT22GVerifyUserPassword(HANDLE handle,const unsigned char *pUserPassword,short nLen);
功能	这里的主要功能是校验用户密码，校验正确则把当前操作权限提升级为用户级
输入参数	handle 操作设备句柄,UserPassword 最大 16 字节密码, nLen 密码长度
输出参数	无
权限类别	匿名
返回值	密码的校验次数(>0 正确)

使用示例	short VerifyNum;
	unsigned char Password[16];
	Short m_st;
	CString m_EVerifyPass;
	
	m_EVerifyPass="8888888888888888";
	memcpy(Password,m_EVerifyPass,m_EVerifyPass.GetLength());
	VerifyNum=NT22GVerifyUserPassword(handle,Password,m_EVerifyPass.GetLength());
	if(VerifyNum==0)

	{
	AfxMessageBox("对不起，密码校验次数为0!", MB_ICONINFORMATION);
	return;
	}
	
	if(VerifyNum<0)
	{
	AfxMessageBox("对不起，密码校验失败!", MB_ICONINFORMATION);
	return;
	}
	
	AfxMessageBox("校验密码成功!",MB_OK|MB_ICONINFORMATION);

6.9 修改超级用户密码 NT22GSetSuperPassword

函数原型	short __stdcall NT22GSetSuperPassword(HANDLE handle,const unsigned char *pSuperPassword,short nLen);
功能	修改超级用户密码
输入参数	handle 操作设备句柄, pSuperPassword 最大为 16 字节密码, nLen 密码长度
输出参数	无
权限类别	超级用户权限
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	unsigned char Password[16];
	CString m_EUserNewPass;
	
	m_EUserNewPass ="8888888888888888";
	memcpy(Password,m_EUserNewPass,m_EUserNewPass.GetLength());
	m_st=NT22GSetSuperPassword(handle,Password,m_EUserNewPass.GetLength());
	if(m_st ==0)
	{
	AfxMessageBox("修改密码成功!", MB_ICONINFORMATION);
	return;
	}

6.10 修改用户密码 NT22GSetUserPassword

函数原型	short __stdcall NT22GSetUserPassword(HANDLE handle,const unsigned char *pUserPassword,short nLen);
功能	修改用户密码
输入参数	handle 操作设备句柄,UserPassword 最大为 16 字节密码, nLen 密码长度
输出参数	无
权限类别	用户或超级用户权限
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	unsigned char Password[16];
	CString m_EUserNewPass;
	
	m_EUserNewPass ="8888888888888888";
	memcpy(Password,m_EUserNewPass,m_EUserNewPass.GetLength());
	m_st=NT22GSetUserPassword(handle,Password,m_EUserNewPass.GetLength());
	if(m_st ==0)
	{
	AfxMessageBox("修改密码成功!", MB_ICONINFORMATION);
	return;
	}

6.11 设置数据块读写权限参数 NT22GSetAuthorityData

函数原型	short __stdcall NT22GSetAuthorityData(HANDLE handle,short SuperNum,short UserNum,const unsigned char *pRWSwitch);
功能	该接口主要功能是设置数据块的用户读写权限参数及设置超级用户密码及普通用户密码校验次数(0 为不限制)
输入参数	handle 操作设备句柄, SuperNum 超级密码重试次数，UserNum 用户密码重试次数(0 为不限), pRWSwitch 16 字节块读写权限参数
输出参数	无
权限类别	超级用户权限
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	Short ESuperNum=0;
	Short m_EUserNum =0;
	unsigned char RWSwitch[16];
	
	memset(RWSwitch, 0x31,sizeof(RWSwitch)); //全部有权
	m_st=NT22GSetAuthorityData(handle,m_ESuperNum,m_EUserNum,RWSwitch);
	if(m_st==0)
	{
	AfxMessageBox("设置读写权限参数成功!", MB_ICONINFORMATION);
	return;
	}

6.12 获取数据块读写权限参数 NT22GGetAuthorityData

函数原型	short __stdcall NT22GGetAuthorityData(HANDLE handle,unsigned char *pRWSwitch);
功能	获取当前设备的各块数据读写权限情况,’0’为只读 ‘1’ 为读写
输入参数	handle 操作设备句柄
输出参数	pRWSwitch 块读写权限参数操作 16 字节('0'或'1')
权限类别	用户权限
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	unsigned char RWSwitch[16];
	
	memset(RWSwitch, 0,sizeof(RWSwitch));
	m_st=NT22GGetAuthorityData(handle, RWSwitch);
	if(m_st==0)
	{
	AfxMessageBox("获取读写权限参数成功!", MB_ICONINFORMATION);
	return;
	}

6.13 设置显示信息 NT22GSetDispInfo

函数原型	short __stdcall NT22GSetDispInfo(HANDLE handle,const unsigned char *pDispData,short nLen);
功能	该接口主要功能是加密锁属性页显示的信息(17 个汉字或字符串)
输入参数	handle 操作设备句柄,DispData 显示信息 34 字节数据 nLen 显示长度
输出参数	无
权限类别	超级用户权限
返回值	返回操作状态(0 为正确)
使用示例	Short m_st; unsigned char DispData[34];

	m_EDispInfo="www.nlite.net.cn";
	memcpy(DispData,m_EDispInfo,m_EDispInfo.GetLength());
	m_st=NT22GSetDispInfo(handle,DispData,m_EDispInfo.GetLength());
	if(m_st==0)
	{
	AfxMessageBox("设置显示信息成功!", MB_ICONINFORMATION);
	return;
	}

6.14 设置产品标识 NT22GSetPID

函数原型	short __stdcall NT22GSetPID(HANDLE handle,const unsigned char *pProductPassword,short nBuffLen,unsigned char *pPidData);
功能	该接口主要功能是根据输入的产品密码产生设置新的产品标识(PID)
输入参数	handle 操作设备句柄,pProductPassword 1-56 字节 nLen 密码长度
输出参数	pPidData 运算后得到的 16 字节新产品标识
权限类别	超级用户权限
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	unsigned char PidData[6];
	unsigned char ProductPassword[56];
	
	memset(PidData,0,sizeof(PidData));
	memset(ProductPassword,0,sizeof(ProductPassword));
	m_EProductPass= "NLite22G_Demo";
	memcpy(ProductPassword,m_EProductPass,m_EProductPass.GetLength());
	m_st=NT22GSetPID(handle,ProductPassword,m_EProductPass.GetLength(),PidData);
	if(m_st==0)
	{
	AfxMessageBox("设置产品标识成功!", MB_ICONINFORMATION);
	return;
	}

6.15 向设备写数据 NT22GDevWrite

函数原型	short __stdcall NT22GDevWrite(HANDLE handle,short StartAdd,const unsigned char *pWriteBuff, short nBuffLen);
功能	该接口主要功能是从指定的地址开始写指定长度的数据
输入参数	handle 操作设备句柄,StartAdd 开始地址 pWriteBuff 要写的数据,nBuffLen 数据长度
输出参数	无
权限类别	用户或超级用户
返回值	返回操作状态(0 为正确)

使用示例	Short m_st;
	short nLen, m_EAdd;
	unsigned char WriteData[256];
	CString m_ MData;
	
	m_MData= "1234567890";
	nLen=m_MData.GetLength();
	m_EAdd=0;
	if(m_EAdd+nLen>256)
	nLen=256-m_EAdd;
	
	memcpy(WriteData,m_MData,nLen);
	m_st=NT22GDevWrite(handle,m_EAdd,WriteData,nLen);
	if(m_st==0)
	{

	AfxMessageBox("写数据成功!", MB_ICONINFORMATION); return; }

6.16 从设备读数据 NT22GDevRead

函数原型	short __stdcall NT22GDevRead(HANDLE handle,short StartAdd,unsigned char *pReadBuff, short nBuffLen);
功能	该接口主要功能是从指定的地址开始读指定长度的数据
输入参数	handle 操作设备句柄,StartAdd 开始地址,nBuffLen 数据长度
输出参数	pReadBuff 读到的数据
权限类别	用户或超级
返回值	返回操作状态(0 为正确)

使用示例	unsigned char ReadBuff[256];
	Short m_st;
	short nLen, m_EAdd;
	
	memset(ReadBuff,0,sizeof(ReadBuff));
	m_ELen=100;
	m_EAdd=0;
	if((m_EAdd+m_ELen)>256)
	m_ELen=256-m_EAdd;
	
	m_st=NT22GDevRead(handle,m_EAdd,ReadBuff,m_ELen);
	if(m_st==0)
	{
	AfxMessageBox("读取数据成功!", MB_ICONINFORMATION);
	return;
	}

6.17 关闭设备 NT22GCloseDev

函数原型	short __stdcall NT22GCloseDev(HANDLE handle);
功能	关闭设备,把权限恢复到匿名级
输入参数	handle 操作设备句柄
输出参数	无
权限类别	匿名
返回值	返回操作状态(0 为正确)
使用示例	NT22GCloseDev(handle);

7 联系我们

用户反馈

我们非常欢迎用户对我们的产品的任何反馈，您可将您对我们的产品的任何建议和意见发送 到以下 **EMAIL** 邮箱或打电话与我们直接联系，我们会给您满意的答复。

电话：(020)82315664-805 39885802-805

EMAIL：Manager@ NLite.net.cn

申请试用

如果您对我们的产品感兴趣，您可先申请试用，在您测试通过后再进行购买。如果您希望进行产品试用，可以到下网站填写试用单或直接打电话与我们联系：

试用单网址：http://www. NLite.net.cn/Trial/index.htm

电话：(020) 82315664-801 39885802-801

传真：(020) 82315664-803 39885802-803

EMAIL: Trial@NLite.net.cn

QQ：985562662

技术支持

我们提供了多种方式的技术支持服务，您可通过如下方式与我们的技术人员咨询您所关注的技术问题：

电话：(020) 82315664-802 39885802-802

传真：(020) 82315664-803 39885802-803
EMAIL: Support@NLite.net.cn
QQ：99801189

购买产品

如果您希望咨询我们产品价格或正式购买我们的产品，可以通过如下方式与我们的销售人员 联系：

电话：(020) 82315664-801 39885802-801

传真：(020) 82315664-803 39885802-803

EMAIL: Sales@NLite.net.cn
QQ：120423252