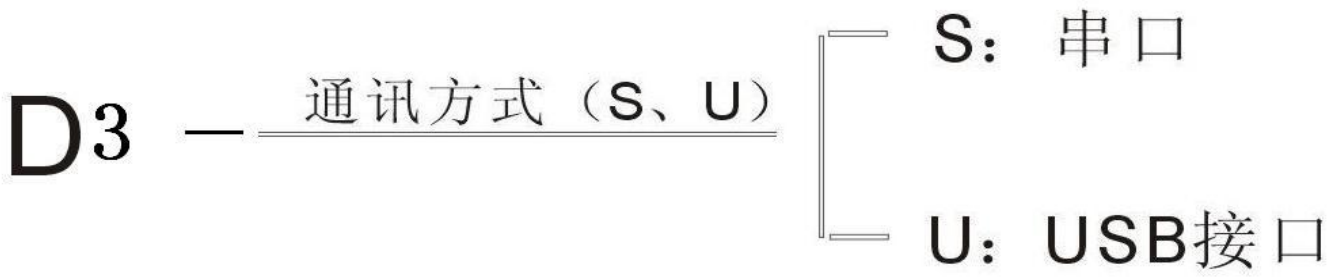
D3非接触式读写器

点击上述图片进入设备介绍

设备使用说明

1、型号说明



2、连接方式

1）、USB接口：将USB线插入计算机的USB接口即可。（如图2）

2）、串口：先关闭计算机电源，将键盘连接线拔下，将键盘线插入读写器后端的键盘口连接端，然后将读写器后端的键盘口端子插入计算机的键盘口，再将串口线一端接至计算机的串口上。（如图3）

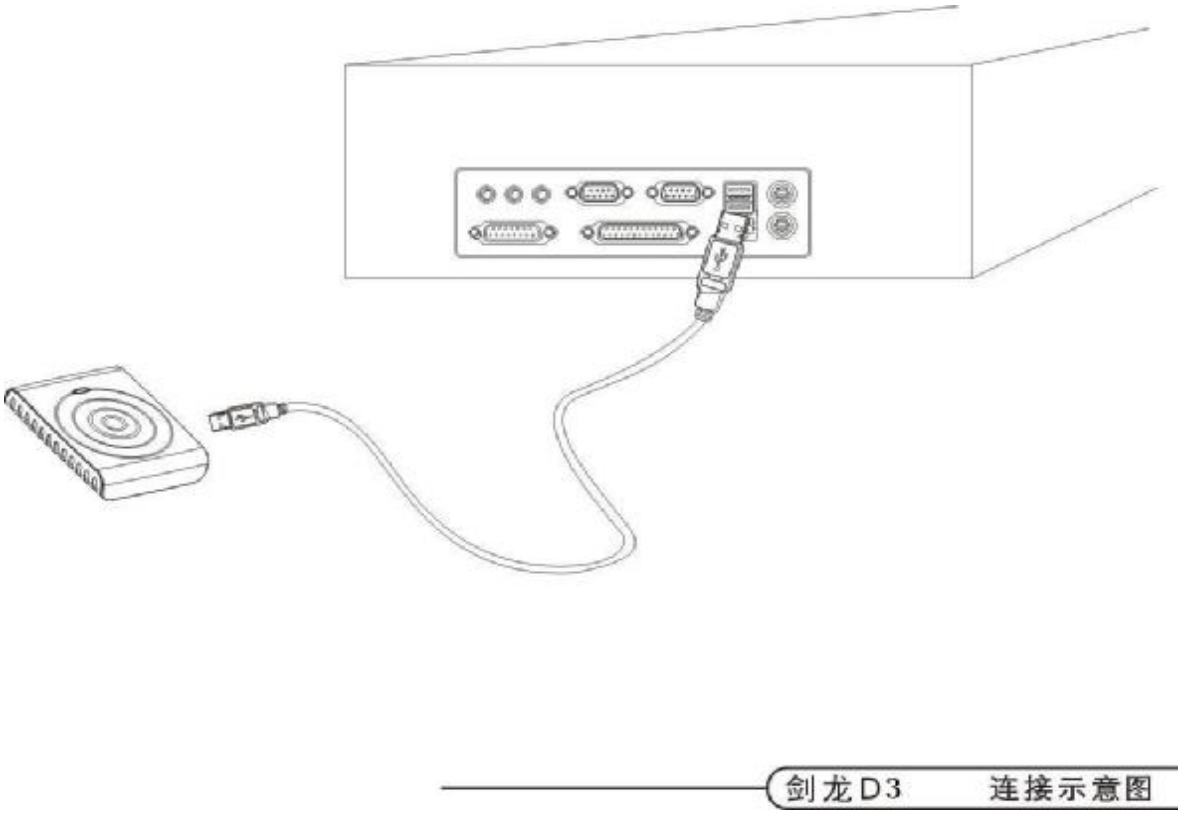


图2.USB接口连接示意图

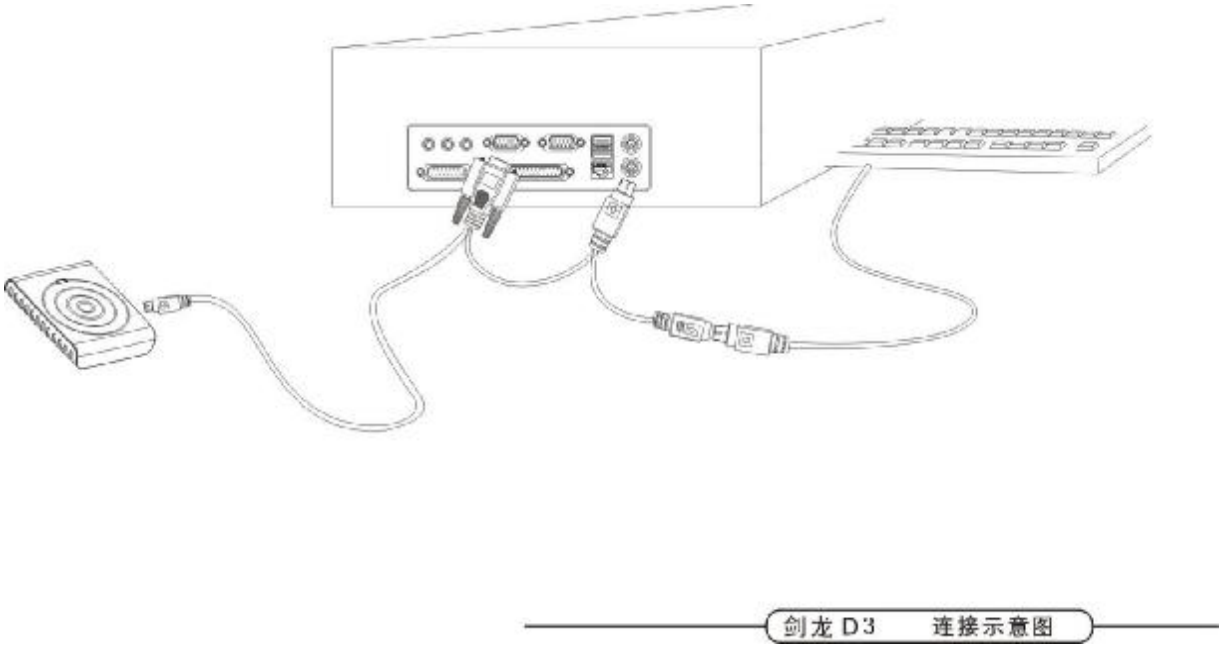


图3.串口连接示意图

3、指示灯

通电灯亮，通迅时灯闪烁。

4、技术指标

通讯接口：采用串口或USB口

电源：键盘口或USB供电，<150 mA

工作环境温度：-20~+60℃

工作相对湿度：<95%

外型尺寸：长x宽x高：123mm*95mm*27mm

5 注意事项

在帮助文档中列出了支持Mifare S50 以及Mifare S70的卡函数，但需要说明的是D3的出厂通用版本仅支持符合ISO-14443 TYPEA 协议的卡片，如果需要支持其他类型卡片的读写器请予事先说明。

此外随着智能卡行业的不断发展、新的卡型的不断推出，D3读写器也在不断的更新来适合发展的需要，如果需要使用D3操作新的卡型，也需要事先咨询一下所需要的版本。

函数使用规则

(1) **首先要调用通讯口初始化函数**dc_init ，其返回值为设备标识符，它将作为其它函数的调用参数。

(2) 调用WINDOWS 32位动态库时，程序退出之前要执行dc_exit (HANDLE icdev) 函数，关闭串口，释放句柄icdev；**否则再次初始化串口将出错**。

(3) 函数调用错误类型，请参照**函数错误类型代码**。所有函数的错误代码均以负数形式返回；Foxpro For Dos例外。

(4) **动态库的位置**应该在声明的相应目录中或缺省的目录当中，否则会有无法寻找到动态库的错误。

(5) 函数的十六进制HEX方式调用中，传入和读出的字符数组是以十六进制字符串的方式进行的，其余参数调用方式相同，所以在函数详细说明中不再列出。

注意：函数详细的使用方法，参考D8\EXAMPLES目录下提供的**范例**。

通 用 函 数

dc_init	dc_exit	dc_card	dc_des
dc_request	dc_anticoll	dc_select	dc_load_key
dc_authentication	dc_halt	dc_read	dc_write
dc_authentication_2	dc_changeb3	dc_initval	dc_readval
dc_check_write	dc_decrement	dc_increment	dc_transfer
dc_HL_authentication	dc_HL_write	dc_HL_read	dc_restore
dc_authentication_pass			

```
int  dc_init(int port,long baud);
```

功 能：初始化通讯口

参 数：port：取值为0～19时，表示串口1～20；为100时，表示USB口通讯，此时波特率无效。

baud：为通讯波特率9600～115200

返 回：成功则返回串口标识符>0，失败返回负值，见错误代码表

例：int icdev;

icdev=dc_init(0,9600);//初始化串口1，波特率9600

int dc_exit(int icdev);

功 能：关闭端口

参 数：icdev：通讯设备标识符

返 回：成功返回0

例：dc_exit(icdev);

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

int dc_card(int icdev,unsigned char _Mode,unsigned long *_Snr);

功 能：寻卡，能返回在工作区域内某张卡的序列号(该函数包含了dc_request,dc_anticoll,dc_select的整体功能)

参 数：icdev：通讯设备标识符

_Mode：寻卡模式

_Snr：返回的卡序列号

返 回：成功则返回 0

例：int st;

unsigned long snr;

st=dc_card(icdev,0,&snr);

注：选择IDLE模式，在对卡进行读写操作，执行dc_halt()指令中止卡操作后，只有当该卡离开并再次进入操作区时，读写器才能够再次对它进行操作。

相关HEX函数：

__int16 __stdcall dc_card_hex(HANDLE icdev,unsigned char _Mode,unsigned char *snrstr)

int dc_request(int icdev,unsigned char _Mode,unsigned int *TagType);

功 能：寻卡请求

参 数：icdev：通讯设备标识符

_Mode：寻卡模式

Tagtype：卡类型值，详情见附录TagType特征值

返 回：成功则返回 0

例：int st;

unsigned int *tagtype;

st=dc_request(icdev,0,tagtype);

int dc_anticoll(int icdev,unsigned char _Bcnt,unsigned long *_Snr);

功 能：防卡冲突，返回卡的序列号

参 数：icdev：通讯设备标识符

_Bcn： 设为0

_Snr：返回的卡序列号地址

返 回：成功则返回 0

例：int st;

unsigned long snr;

st=dc_anticoll(icdev,0,&snr);

注：request指令之后应立即调用anticoll，除非卡的序列号已知。

int dc_select(int icdev,unsigned long _Snr,unsigned char *_Size);

功 能：从多个卡中选取一个给定序列号的卡

参 数：icdev：通讯设备标识符

_Snr：卡序列号

_Size：指向返回的卡容量的数据

返 回：成功则返回 0

例： int st,type;

unsigned char size;

unsigned long snr;

dc_request(icdev,0,&type);

dc_anticoll(icdev,0,&snr);

st=dc_select(icdev,snr,&size);

int dc_load_key(int icdev,unsigned char _Mode,unsigned char _SecNr,unsigned char *_NKey);

功 能：将密码装入读写模块RAM中

参 数：icdev：通讯设备标识符

_Mode：装入密码模式，同密码验证模式

_SecNr：扇区号（M1卡：0～15； ML卡：0）

_Nkey：写入读写器中的卡密码

返 回：成功则返回 0

例：//key A and key B

unsigned char password[7]={0xa0,0xa1,0xa2,0xa3,0xa4,0xa5};

/* 装入1扇区的0套A密码 */

```
if((dc_load_key(icdev,0,1,password))!=0)
{
    printf("Load key error!");
    dc_exit(icdev);
}
```

相关HEX函数：

```
__int16 __stdcall dc_load_key_hex(HANDLE icdev,unsigned char _Mode,unsigned char _SecNr,unsigned char *_NKey)
```

```
int dc_authentication(int icdev,unsigned char _Mode,unsigned char _SecNr)
```

功 能：核对密码函数

参 数：icdev: dc_init返回的设备描述符

_Mode：密码验证模式

_SecNr：要验证密码的扇区号

返 回：成功返回0

```
int dc_read(int icdev,unsigned char _Adr,unsigned char *_Data);
```

功 能：读取卡中数据

对于M1卡，一次读一个块的数据，为16个字节；

对于ML卡，一次读出相同属性的两页（0和1，2和3，...），为8个字节

参 数：icdev：通讯设备标识符

_Adr：M1卡——块地址（0～63）,MS70(0-255)；

ML卡——页地址（0～11）

_Data：读出数据

返 回：成功则返回 0

例：int st;

unsigned char data[16];

st=dc_read(icdev,4,data); //读M1卡块4的数据

相关HEX函数：

__int16 __stdcall dc_read_hex(HANDLE icdev,unsigned char _Adr,char *_Data)

int dc_write(int icdev,unsingned char _Adr,unsigned char *_Data);

功 能：向卡中写入数据

对于M1卡，一次必须写一个块，为16个字节；

对于ML卡，一次必须写一页，为4个字节

参 数：icdev：通讯设备标识符

_Adr：M1卡──块地址（1～63）,M1S70卡—块地址（1-255）；

ML卡──页地址（2～11）

_Data：要写入的数据

返 回：成功则返回0

例：int st;

unsigned char *data="1234567890123456";

st=dc_write(icdev,4,data);//写第四块

相关HEX函数：

__int16 __stdcall dc_write_hex(HANDLE icdev,unsigned char _Adr,char *_Data)

int dc_halt(int icdev)

功 能：中止对该卡操作

参 数：icdev：通讯设备标识符

返 回：成功则返回0

例：st=dc_halt(icdev);

说明：使用dc_card()函数时，有个_Mode参数，如果_Mode=0则在对卡进行操作完毕后，执行dc_halt();则该卡进入HALT模式，则必须把卡移开感应区再进来才能寻得这张卡。

int dc_des(unsigned char *key,unsigned char *sour,unsigned char *dest,__int16 m)

功 能：DES算法加解密函数

参 数：key：密钥

sour：要加解密的数据

dest：加解密后的数据

m：加解密模式，m=1时，为加密；m=0时，为解密过程

返 回：成功返回0

相关hex函数

__int16 __stdcall dc_des_hex(unsigned char *key,unsigned char *sour,unsigned char *dest, __int16 m);

int dc_changeb3(int icdev,unsigned char _SecNr,unsigned char *_KeyA,unsigned char _B0,unsigned char _B1,unsigned char _B2,unsigned char _B3,unsigned char _Bk,unsigned char *_KeyB);

功 能：修改块3的数据(当为M1S70卡时，当扇区号大于31时，修改块15的数据(即一个扇区的最后一块))

参 数：icdev：通讯设备标识符

_SecNr：扇区号（M1:0～15,M1S70:0-39）

_KeyA：密码A

_B0：块0控制字（当一扇区有16块时，对应为块 0- 4的控制字），低3位（D2D1D0）对应C10、C20、C30

_B1：块1控制字（当一扇区有16块时，对应为块 5- 9的控制字），低3位（D2D1D0）对应C11、C21、C31

_B2：块2控制字（当一扇区有16块时，对应为块10-14的控制字），低3位（D2D1D0）对应C12、C22、C32

_B3：块3控制字（当一扇区有16块时，对应为块15的控制字），低3位（D2D1D0）对应C13、C23、C33

_Bk：保留参数，取值为0

_KeyB：密码B

返 回：成功则返回 0

例：int st;

```
unsigned char keya;
unsigned char keyb;
memset(keya,0xff,6);
memset(keyb,0xff,6);
 st=dc_changeb3(icdev,keya,0x00,0x00,0x00,0x01,0,keyb);
```

__int16 __stdcall dc_authentication_passaddr(HANDLE icdev,unsigned char _Mode, unsigned char blockAddr, unsigned char *passbuff)

功 能：核对密码函数，用此函数时，可以不用执行dc_load_key()函数

参 数：icdev: dc_init返回的设备描述符

_Mode：密码验证模式

blockAddr：要验证密码的块地址号

passbuff:密码字符串

返 回：成功返回0

相关HEX函数：

```
__int16 __stdcall dc_authentication_passaddr_hex(HANDLE icdev,unsigned char
_Mode, unsigned char blockAddr, unsigned char *passbuff)
```

```
int dc_initval(int icdev,unsigned char _Adr,unsigned long _Value);
```

功 能：初始化块值

参 数：icdev：通讯设备标识符

_Adr：块地址

_Value：初始值

返 回：成功则返回 0

例：

```
int st;
unsigned long value;
value=1000;                    /* 给value赋值*/
st=dc_initval(icdev,1,value);        /*将块1的值初始化为1000*/
```

注：在进行值操作时，必须先执行初始化值函数，然后才可以读、减、加的操作。

```
int dc_increment(int icdev,unsigned char _Adr,unsigned long _Value);
```

功 能：块加值

参 数：icdev：通讯设备标识符

_Adr：块地址

_Value：要增加的值

返 回：成功则返回 0;

例：int st;

unsigned long value;

value=10;

st=dc_increment(icdev,1,value); /*将块1的值增加value*/

dc_readval(int icdev,unsigned char _Adr,unsigned long *_Value);

功 能：读块值

参 数：icdev：通讯设备标识符

_Adr：块地址

_Value：读出值的地址

返 回：成功则返回 0

例：int st;

unsigned long value;

st=dc_readval(icdev,1,&value); /*读出块1的值，放入value*/

int dc_decrement(int icdev,unsigned char _Adr,unsigned long _Value);

功 能：块减值

参 数：icdev：通讯设备标识符

_Adr：块地址

_Value：要减的值

返 回：成功则返回 0

例：int st;

unsigned long value;

value=10;

st=dc_decrement(icdev,1,value);　　/*将块1的值减少value*/

int dc_HL_authentication(int icdev,unsigned char reqmode,unsigned long snr,unsigned char authmode,unsigned char secnr);

功 能：高级验证（无需调用寻卡函数）

参 数：icdev：通讯设备标识符

reqmode：寻卡模式

snr：卡序列号（在寻卡模式为2时使用）

authmode：密码验证模式

secnr：扇区号

返 回：成功则返回 0

例：st=dc_HL_authentication(icdev,0,snr,0,1);

int dc_HL_read(int icdev,unsigned char _Mode,unsigned char _Adr,unsigned long _Snr,unsigned char *_Data, unsigned long *_NSnr);

功 能：高级读函数

参 数：icdev：通讯设备标识符

_Mode：寻卡模式

_Adr：块地址

_Snr：卡的序列号

_Data：读出的数据

_NSnr：返回卡的序列号

返 回：成功则返回 0

例：if((dc_HL_read(icdev,1,5,snr,HLdata,&Rsnr))!=0)

{

printf("read HL Rvalue wrong");

}

相关HEX函数：

__int16 __stdcall dc_HL_read_hex(HANDLE icdev,unsigned char _Mode,unsigned char _Adr,unsigned long _Snr,unsigned char *_Data,unsigned long *_NSnr);

int dc_HL_write(int icdev,unsigned char _Mode,unsigned char _Adr,unsigned long *_Snr,unsigned char *_Data);

功 能：高级写函数

参 数：icdev：通讯设备标识符

_Mode：寻卡模式

_Adr：块地址

_Snr：卡的序列号地址

_Data：写入的数据

返 回：成功则返回 0

例：if((dc_HL_write(icdev,1,5,Snr,HLdata))!=0)

{

printf("dc_HL_write wrong");

}

相关HEX函数

__int16 __stdcall dc_HL_write_hex(HANDLE icdev,unsigned char _Mode,unsigned char _Adr, unsigned long *_Snr, unsigned char *_Data);

int dc_restore(int icdev,unsigned char _Adr);

功 能：回传函数，将EEPROM中的内容传入卡的内部寄存器

参 数：icdev：通讯设备标识符

_Adr：要进行回传的块地址

返 回：成功返回0

例：int st;

st=dc_restore(icdev,1);

注：用此函数将某一块中的数值传入内部寄存器，然后用dc_transfer()函数将寄存器中数据再传送到另一块中去，实现块与块之间数值传送。该函数只用于值块。

int dc_transfer(int icdev,unsigned char _Adr);

功 能：传送，将寄存器的内容传送到EEPROM中

参 数：icdev：通讯设备标识符

_Adr：要传送的块地址

返 回：成功返回0

例：dc_restore(icdev,1);

dc_transfer(icdev,2);

上两行实现将块1的内容传送到块2。

注：见dc_restore()的说明。

int dc_authentication_2(int icdev,unsigned char _Mode,unsigned char KeyNr,unsigned char Adr);

功 能：卡验证密码

参 数：icdev：通讯设备标识符

_Mode：密码验证模式

KeyNr：扇区地址

Adr：块地址

返 回：成功则返回 0

例：int st;

st=dc_authentication_2(icdev,0,3,3);

int dc_authentication_pass(int icdev, unsigned char _Mode, unsigned char Addr, unsigned char *passbuff)

功 能：核对密码函数，用此函数时，可以不用执行dc_load_key()函数(只支持M1和S70卡)

参 数：icdev: dc_init返回的设备描述符

_Mode：密码验证模式

Addr：要验证密码的扇区号

passbuff:密码字符串

返 回：成功返回0

相关HEX函数：

__int16 __stdcall dc_authentication_pass_hex(HANDLE icdev,unsigned char _Mode, unsigned char _Addr,unsigned char *passbuff)

int dc_card_double(int icdev,unsigned char _Mode,unsigned char *_Snr);

功 能：寻卡，和dc_card差别为多调了dc_anticoll2，dc_select2

参 数：icdev：通讯设备标识符

_Mode：寻卡模式

_Snr：返回的卡序列号(8 字节)

返 回：成功则返回 0

例：int st;

unsigned char snr[8];

```
st=dc_card_double(icdev,0,snr);
```

注：选择IDLE模式，在对卡进行读写操作，执行dc_halt()指令中止卡操作后，只有当该卡离开并再次进入操作区时，读写器才能够再次对它进行操作。

相关HEX函数：

```
__int16 __stdcall dc_card_double_hex(HANDLE icdev,unsigned char
_Mode,unsigned char *snrstr)
```

设备操作函数

dc_beep	dc_getver	dc_srd_eeprom
dc_swr_eeprom	hex_a	a_hex
dc_reset		

```
int dc_beep(int icdev,unsigned int _Msec);
```

功 能：蜂鸣

参 数：icdev：通讯设备标识符

unsigned int _Msec：蜂鸣时间，单位是10毫秒

返 回：成功则返回 0

例：int st;

```
st=dc_beep(icdev,10);                /*鸣叫100毫秒*/
```

```
int dc_getver(int icdev,unsinged char *buff);
```

功 能：读取硬件版本号

参 数：icdev:通讯设备标识符

buff：存放版本号的缓冲区，长度3字节(包括结束字符'\0')。

返 回：成功则返回 0

例：unsigned char buff[3];

dc_getver(icdev,buff);

int dc_srd_eeprom(int icdev,int offset,int length,unsigned char *rec_buffer);

功 能：读取读写器备注信息

参 数：icdev：通讯设备标识符

offset：偏移地址（0～1278）

length：读取信息长度（1～1279）

rec_buffer：读取到的信息

返 回：成功则返回 0

例：int st;

unsigned char buffer[100];

st=dc_srd_eeprom(icdev,0,100,buffer);

相关HEX函数：

__int16 __stdcall dc_srd_eepromhex(HANDLE icdev,__int16 offset, __int16 lenth, unsigned char *rec_buffer);

int dc_swr_eeprom(int icdev,int offset,int length,unsigned char* buffer);

功 能：向读写器备注区中写入信息

参 数：icdev：通讯设备标识符

offset：偏移地址（0～1278）

length：写入信息长度（1～1279）

buffer：要写入的信息

返 回：成功则返回 0

相关HEX函数：

__int16 __stdcall dc_swr_eepromhex(HANDLE icdev,__int16 offset,__int16 length, unsigned char* send_buffer)

__int16 a_hex(unsigned char *a,unsigned char *hex,__int16 len)

功 能：字符串转换函数，十六进制字符转换成普通字符(长转短)。

参 数：a ：要转换的字符

hex：转换后的字符

len：字符a的长度

返 回：成功则返回 0

void hex_a(unsigned char *hex,unsigned char *a,__int16 len)

功 能：字符串转换函数，普通字符转换成十六进制字符(短转长)。

参 数：hex：要转换的字符

a ：转换后的字符

len：字符hex的长度

__int16 __stdcall dc_reset(HANDLE icdev,unsigned __int16 _Msec)

说明：射频复位函数

调用： icdev ----通讯设备端口标识符

_Msec ----复位时间,单位为毫秒(此值为0时是关闭射频,为1,2...为复位1毫秒,2毫秒...)

返回： <0 错误。其绝对值为错误号

=0 成功。

举例：st=dc_reset(icdev,2)

32位Windows库函数错误代码

返回值（负数）	错误类型
0x10(016)	通讯错误
0x11(017)	超时错误
0x20(032)	打开端口错误
0x21(033)	获得端口参数错误
0x22(034)	设置端口参数错误
0x23(035)	关闭端口出错
0x24(036)	端口被占用
0x30(048)	格式错误
0x31(049)	数据格式错误
0x32(050)	数据长度错误
0x40(064)	读错误
0x41(065)	写错误
0x42(066)	无接收错误
0x50(080)	不够减错误
0x73(115)	取版本号错误

返回值（正数）	错误类型
0x01(001)	未放置卡片或认证错误
0x02(002)	数据校验错误
0x03(003)	数值为空错误
0x04(004)	认证失败
0x05(005)	奇偶校验错误
0x06(006)	读写设备与卡片通讯错误
0x08(008)	读卡序列号错误
0x09(009)	密码类型错误
0x0a(010)	卡片尚未被认证
0x0b(011)	读卡操作比特数错误
0x0c(012)	读卡操作字节数错误

0x0f(015)	写卡操作失败
0x10(016)	增值操作失败
0x11(017)	减值操作失败
0x12(018)	读卡操作失败
0x13(019)	传输缓冲区溢出
0x15(021)	传输帧错误
0x17(023)	未知的传输需求
0x18(024)	防冲突错误
0x19(025)	感应模块复位错误
0x1a(026)	非认证接口
0x1b(027)	模块通讯超时
0x3c(060)	非正常操作
0x64(100)	错误的数据
0x7c(124)	错误的参数值

判断卡型的特征值

卡型	特征值1	特征值2
MIFARE 1	4	136
S70	2	24

在执行dc_request函数时使用的TagType值是特征值1，执行dc_select函数时使用的size值是特征值2。