

神思第二代居民身份证验证机具接口函数说明

版本 V2.6.2 2017-12-20

函数原型:

int UCommand1(BYTE *pCmd, int *parg0, int *parg1, int *parg2)

*pCmd:

- 0x41: 初始化端口
- 0x42: 关闭端口
- 0x43: 验证卡
- 0x44: 读基本信息
- 0x45: 读最新住址信息
- 0x46: 仅读文字信息
- 0x47: 读基本信息但不进行图像解码
- 0x48: 读外国人永久居留证基本信息
- 0x49: 读二代身份证基本信息或外国人永久居留证基本信息

1. 初始化端口

*parg0 : 串口号, 取值 1~16

USB 口 取值 1001~1016

*parg0=0 时, 自动查找端口范围串口 1~8, USB1001~1016

2. 关闭端口

parg0 无效

3. 验证卡

parg0 无效

4. 读卡内信息

//读基本信息

生成 wz.txt 身份信息文本文件 unicode 编码

生成 wx.txt 身份信息文本文件 GBK 编码

如果设备支持读取指纹, 并且身份证中存在指纹信息, 则生成 fp.bin 指纹信息文件

生成 zp.bmp 图像文件

//只读文字信息

生成 wz.txt 身份信息文本文件 unicode 编码

生成 wx.txt 身份信息文本文件 GBK 编码

如果设备支持读取指纹, 并且身份证中存在指纹信息, 则生成 fp.bin 指纹信息文件

//读最新住址信息

生成 NewAdd.txt 文件 unicode 编码

生成 NewAddgb.txt 文件 GBK 编码

//读取基本信息, 且图像不解码

生成 wz.txt 身份信息文本文件 unicode 编码

生成 wx.txt 身份信息文本文件 GBK 编码

生成 zp.wlt 图像压缩文件

如果设备支持读取指纹, 并且身份证中存在指纹信息, 则生成 fp.bin 指纹信息文件

当*parg2 为 char * 时, 指定二代证的存储文件

路径必须是全路径, 即当作 char*指针时, 第二字节为"."时, 认为传入了文件名

/该文件名不要包含后缀，因为要以该文件名生成文本格式文件、已转换的文本格式文件、Wlt 格式文件、Bmp 格式文件

//如 parg2 指示为 C:\temp\id
//则生成 C:\temp\idwz.txt Unicode 格式文本
// C:\temp\idwx.txt 已转换文本
// C:\temp\idpic.wlt 未解码图片
// C:\temp\idzp.bmp 已解码图片
// C:\temp\idfp.bin 指纹信息文件

注意：当身份证中没有指纹数据时，返回 62171 为成功；
当身份证中含有指纹数据时，返回 62172 为成功；
当读取外国人永久居留证时，返回 62173 为成功。

文字信息采用 GB 13000 的 UCS-2 进行存储，各项目分配如下：

项目	长度（字节）	说明
姓名	30	汉字
性别	2	代码
民族	4	代码
出生	16	年月日：YYYYMMDD
住址	70	汉字和数字
公民身份号码	36	数字
签发机关	30	汉字
有效期起始日期	16	年月日：YYYYMMDD
有效期截止日期	16	年月日：YYYYMMDD 有效期为长期时存储“长期”
最新住址(NewAdd.txt)	70	汉字和数字

错误信息详细见下表

-1	相片解码错误
-2	Wlt 文件后缀错误
-3	Wlt 文件打开错误
-4	Wlt 文件格式错误
-5	软件未授权
-6	设备连接错误
-7	设备不正确（非法设备）
-8	文件存储失败
-9	加载通讯函数错误
-10	端口操作失败
-11	解码失败
-13	非二代身份证
-14	非外国人永久居留身份证
-15	未读二代身份证
-16	未读外国人永久居留身份证
2	接收数据超时

62171	读取身份证信息正确（无指纹）
62172	读取身份证信息正确（带指纹）
62173	读取外国人永久居留身份证正确

类 别	返回值 (16 进制)	意 义
与 ID 卡相关	80	找卡不成功
	81	选卡不成功
	31	卡认证机具失败
	32	机具认证卡失败
	33	信息验证错误
	34	尚未找卡，不能进行对卡的操作
	40	无法识别的卡类型
	41	读卡操作失败
	50	写卡操作失败
	61	用户登录失败

5. 如果身份证，读卡成功后，可以使用一下函数直接获得文字信息

```

int GetName      (char *buf);      读取姓名
int GetSex       (char BYTE *buf);  读取性别编码
int GetFolk      (char BYTE *buf);  读取民族编码
int GetSexGB     (char BYTE *buf);  读取性别
int GetFolkGB    (char BYTE *buf);  读取民族
int GetBirth     (char BYTE *buf);  读取出生
int GetAddr      (char BYTE *buf);  读取住址
int GetIDNum     (char BYTE *buf);  读取公民身份号码
int GetDep       (char BYTE *buf);  读取签发机关
int GetBegin     (char BYTE *buf);  读取有效期起
int GetEnd       (char BYTE *buf);  读取有效期止
int GetNewAddr   (char BYTE *buf);  读取最新地址
int GetBmpPath   (char BYTE *buf);  读取头像图片路径

```

6. 如果外国人永久居留证，读卡成功后，可以使用一下函数直接获得文字信息

```

int FID_GetEnName (char *buf);      读取英文姓名
int FID_ GetChName (char *buf);      读取中文姓名
int FID_ GetNationality (char *buf);  读取国籍
int FID_GetChNationality(char *pbuf)  读取国籍中文简称
int FID_GetSex     (char BYTE *buf);  读取性别编码
int FID_GetSexGB   (char BYTE *buf);  读取性别
int FID_GetBirth   (char BYTE *buf);  读取出生
int FID_GetIDNum   (char BYTE *buf);  读取公民身份号码
int FID_GetDep     (char BYTE *buf);  读取签发机关
int FID_GetBegin   (char BYTE *buf);  读取有效期起
int FID_GetEnd     (char BYTE *buf);  读取有效期止
int FID_ GetVersion (char BYTE *buf);  读取证件版本号

```

int GetBmpPath (char BYTE *buf); 读取头像图片路径

各函数返回值为读取得信息的字节数

buf 为传入的字节数组，用于取得返回值。返回的数据文字信息为 GBK 编码，可以直接显示。

要读取这些信息，必须在读卡内信息成功后才有效！

注意：1. 端口初始化后，在退出程序时必须调用关闭端口。

2. 要读取卡内信息，必须先认证卡，成功后才能够读取。

7. 将 Unicode 字符串文本转换为 GBK 文本

// pwc 为 Unicode 编码文本

// pc 为接收转换结果的字符缓冲区

// nsize 为 pc 缓冲区大小

// 返回结果 >0 成功，为 pc 中字符数

// 其他失败

int UnicodeToGBK(wchar_t *pwc, char *pc, int nsize);

//=====

8. 将身份证的 256 字节 Unicode 字符串文本文件，转换存储为 GBK 文本文件

每项信息占用一行

//sfileSrc 为从身份证读出的 unicode 编码文件的文件名

//sfileDest 为转换为 GBK 文本的文件名，即目标文件

//nDict=1 转换时会将“性别”和“民族”的编码转换为文字

//nDict=0 不转换，保持“性别”和“民族”的数字编码

//返回 1 成功

//其他失败

int ID2CovToGBK(char * sfileSrc,char *sfileDest,int nDict);

9. 获取设备编号：

(1) int PASCAL GetSAMIDToStr(BYTE *id)

输入 id 为接收数据的缓冲区，建议 char buf[64]

返回字符串，如“05.03-20130202-0000022342-1111111111”

返回值为 id 字符串的长度

(2) int PASCAL GetSamID(BYTE *id)

输入 id 为接收数据的缓冲区，建议 char buf[64]

返回 nRet=1 成功，id 中为 16 字节的二进制数据，各式转换：

unsigned long d1,d2,d3,d4,d5;

char sID[64];

d1=id[0]+id[1]*0x100;

d2=id[2]+id[3]*0x100;

d3=id[4]+id[5]*0x100+id[6]*0x10000+id[7]*0x1000000;

d4=id[8]+id[9]*0x100+id[10]*0x10000+id[11]*0x1000000;

d5=id[12]+id[13]*0x100+id[14]*0x10000+id[15]*0x1000000;

sprintf(sID,"%02d.%02d-%08d-%010d-%010d",d1,d2,d3,d4,d5);

推荐操作流程：

