

居民身份证验证安全控制模块接口 API 使用手册

信息产业部数据通信科学技术研究数据所

2004 年 11 月

信息产业部数据通信科学技术研究数据所

居民身份证验证安全控制模块接口

API 使用手册

版 本

1.0

出版日期

2004年11月

著作权注意事项

本书版权为信息产业部数据通信科学技术研究数据所所有。未经信息产业部数据通信科学技术研究数据所书面同意，任何公司、单位或个人，不得用任何手段复制本手册的部分或全部内容。

对印刷错误的更正，所述信息谬误的勘误，以及产品的改进，均由信息产业部数据通信科学技术研究数据所随时作出解释，恕不预先通知，修正内容将编入再版说明书中。

商 标

所有在本手册使用的商标为该商标所有人的资产。

1. 前言	1
2. 系统要求	1
3. API 列表	1
4. API 详细说明	2
4.1 端口类 API	2
4.1.1 SDT_GetCOMBaud	2
4.1.2 SDT_SetCOMBaud	2
4.1.3 SDT_OpenPort	3
4.1.4 SDT_ClosePort	3
4.2 SAM 类 API	4
4.2.1 SDT_ResetSAM	4
4.2.2 SDT_SetMaxRFByte	4
4.2.3 SDT_GetSAMStatus	5
4.2.4 SDT_GetSAMID	5
4.2.5 SDT_GetSAMIDToStr	6
4.3 身份证卡类 API	6
4.3.1 SDT_StartFindIDCard	7
4.3.2 SDT_SelectIDCard	7
4.3.3 SDT_ReadMngInfo	7
4.3.4 SDT_ReadBaseMsg	8
4.3.5 SDT_ReadNewAppMsg	9
5. API 调用说明	9
5.1 调用顺序	9
5.2 例子程序(以 C 语言为例)	10
6. 函数返回码列表	11

1. 前言

本应用程序接口(API)用于二代证验证系统的验证终端应用系统的开发。

2. 系统要求

使用本 API 的 PC 机，必须满足下列条件：

- Windows 98 , Windows 2000 Pro, Windows 2000 Server, WinXP
- 至少 32 兆内存 (32M RAM or Larger)
- 至少 10 兆空闲硬盘空间 (10M Free Hard Disk Space or Larger)
- 至少一个空闲普通串口或 USB 口。

3. API 列表

加密 API 分为下列几类，在下面各表中列出。

端口API		
序号	函数名	功能描述
1.	SDT_GetCOMBaud	查看串口当前波特率
2.	SDT_SetCOMBaud	设置串口上 SAM_V 的波特率
3.	SDT_OpenPort	打开串口/USB 口
4.	SDT_ClosePort	关闭串口/USB 口

SAM 类 API		
序号	函数名	功能描述
5.	SDT_ResetSAM	对 SAM_V 复位
6.	SDT_SetMaxRFByte	设置射频适配器最大通信字节数
7.	SDT_GetSAMStatus	对 SAM_V 进行状态检测
8.	SDT_GetSAMID	读取 SAM_V 的编号 (十六进制)
9.	SDT_GetSAMIDToStr	读取 SAM_V 的编号 (字符串格式)

身份证卡类 API		
序号	函数名	功能描述
10.	SDT_StartFindIDCard	开始找卡
11.	SDT_SelectIDCard	选卡
12.	SDT_ReadMngInfo	读取卡体管理号
13.	SDT_ReadBaseMsg	读取证/卡固定信息
14.	SDT_ReadNewAppMsg	读取追加信息

4. API 详细说明

4.1 端口类 API

4.1.1 SDT_GetCOMBaud

查看串口当前波特率(该函数只用于 SAM_V 采用 RS232 串口的情形,如果采用 USB 接口则不支持该 API)。

```
int SDT_GetCOMBaud (
    int          iPort,
    unsigned int * puiBaudRate
);
```

参数说明：

iPort

[in] 整数,表示端口号。此处端口号必须为 1-16,表示串口,参见 SDT_Login。

puiBaudRate

[out] 无符号整数指针,指向普通串口当前波特率,默认情况下为 115200。

返回值：

0x90	成功
0x1	端口打开失败/端口号不合法
0x5	无法获得该 SAM_V 的波特率,该 SAM_V 串口不可用。

4.1.2 SDT_SetCOMBaud

设置 SAM_V 的串口的波特率(该函数只用于 SAM_V 采用 RS232 串口的情形,如果采用 USB 接口则不支持该 API),设置成功后,在该 SAM_V 和主机注册表中都记录设置后的波特率,保证在 SAM_V 重新启动和该套 API 被重新调用时采用设置后的波特率。该函数调用成功后,需要延时 5 毫秒,然后才能继续与 SAM_V 通信。

```
int SDT_SetCOMBaud (
    int          iPort,
    unsigned int uiCurrBaud,
    unsigned int uiSetBaud
);
```

参数说明：

iPort

[in] 整数,表示端口号。此处端口号必须为 1-16,表示串口。

uiCurrBaud

[in] 无符号整数，调用该 API 前已设置的业务终端与 SAM_V 通信的波特率 (SAM_V 出厂时默认，业务终端与 SAM_V 通信的波特率为 115200)。业务终端以该波特率与 SAM_V 通信，发出设置 SAM_V 新波特率的命令。ui CurrBaud 只能为下列数值之一：115200，57600，38400，19200，9600。如果 ui CurrBaud 数值不是这些值之一，函数返回 0x21；如果已设置的波特率与 ui CurrBaud 不一致，则函数返回 0x02，表示不能设置，调用 API 不成功。

ui SetBaud

[in] 无符号整数，将要设置的 SAM_V 与业务终端通信波特率。ui SetBaud 只能取下列值之一：115200，57600，38400，19200，9600，如果输入 ui SetBaud 参数不是这些数值之一，函数返回 0x21，设置不成功，保持原来的波特率不变。

返回值：

0x90	成功
0x1	端口打开失败/端口号不合法。
0x2	超时，设置不成功。
0x21	ui CurrBaud、ui SetBaud 输入参数数值错误。

4.1.3 SDT_OpenPort

打开串口/USB。

```
int SDT_OpenPort(
    int iPort
);
```

参数说明：

iPort

[in] 整数，表示端口号。1-16（十进制）为串口，1001-1016（十进制）为 USB 口，缺省的一个 USB 设备端口号是 1001。

返回值：

0x90	打开端口成功
1	打开端口失败/端口号不合法

4.1.4 SDT_ClosePort

关闭串口/USB。

```
int SDT_ClosePort (
    int iPort
);
```

参数说明：

iPort
[in] 整数，表示端口号。

返回值：
0x90 关闭端口成功。
0x01 端口号不合法

4.2 SAM 类 API

4.2.1 SDT_ResetSAM

对 SAM_V 复位。
int SDT_ResetSAM (
 int iPort,
 int ifOpen
);

参数说明：

iPort
[in] 整数，表示端口号。根据 SAM_V 使用的接口不同(分为普通串口 SAM_V 和 USB 口 SAM_V)，分别使用不同的端口号（目前串口和 USB 都只支持 16 个，即串口 0001-0016 和 USB1001-1016）：

普通串口 SAM_V	0001 – 0016(十进制)	例如： 0001：串口 1(COM1) 0002：串口 2(COM2)
USB 口 SAM_V	1001 – 1016(十进制)	例如： 1001：USB1 1002：USB2

ifOpen
[in] 整数，0 表示不在该函数内部打开和关闭串口，此时确保之前调用了 SDT_OpenPort 来打开端口，并且在不需要与端口通信时，调用 SDT_ClosePort 关闭端口；非 0 表示在 API 函数内部包含了打开端口和关闭端口函数，之前不需要调用 SDT_OpenPort，也不用再调用 SDT_ClosePort。

返回值：
0x90 成功
其他 失败（具体含义参见返回码表）

4.2.2 SDT_SetMaxRFByte

设置射频适配器最大通信字节数。
int SDT_SetMaxRFByte (

```

int          iPort,
unsigned char ucByte,
int          iIfOpen
);

```

参数说明：

iPort

[in] 整数，表示端口号。参见 SdT_ResetSAM。

ucByte

[in] 无符号字符，24-255，表示射频适配器最大通信字节数。

iIfOpen

[in] 整数，参见 SdT_ResetSAM。

返回值：

0x90 成功
其他 失败（具体含义参见返回码表）

4.2.3 SdT_GetSAMStatus

对 SAM_V 进行状态检测。

```

int SdT_GetSAMStatus (
    int          iPort,
    int          iIfOpen
);

```

参数说明：

iPort

[in] 整数，表示端口号。参见 SdT_ResetSAM。

iIfOpen

[in] 整数，参见 SdT_ResetSAM。

返回值：

0x90 SAM_V 正常
0x60 自检失败，不能接收命令
其他 命令失败（具体含义参见返回码表）

4.2.4 SdT_GetSAMID

读取 SAM_V 的编号。

```

int SdT_GetSAMID (

```



```

    int          iPort,
    unsigned char * pucSAMID,
    int          iIfOpen

);

```

参数说明：

iPort

[in] 整数，表示端口号。参见 SDT_ResetSAM。

pucSAMID

[out] 无符号字符串指针，SAM_V 编号，16 字节。

返回值：

0x90 成功
其他 失败（具体含义参见返回码表）

4.2.5 SDT_GetSAMIDToStr

读取 SAM_V 的编号。

```

int SDT_GetSAMIDToStr (
    int          iPort,
    char *       pcSAMID,
    int          iIfOpen
);

```

参数说明：

iPort

[in] 整数，表示端口号。参见 SDT_ResetSAM。

pcSAMID

[out] 字符串指针，SAM_V 编号。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值：

0x90 成功
其他 失败（具体含义参见返回码表）

4.3 身份证卡类 API

4.3.1 SDT_StartFindIDCard

开始找卡。

```
int SDT_StartFindIDCard (  
    int          iPort ,  
    unsigned char * pucManaInfo,  
    int          iIfOpen  
);
```

参数说明：

iPort

[in] 整数，表示端口号。参见 SDT_ResetSAM。

pucManaInfo

[out] 无符号字符指针，证/卡芯片管理号，4 个字节。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值：

0x9f 找卡成功

0x80 找卡失败

4.3.2 SDT_SelectIDCard

选卡。

```
int SDT_SelectIDCard (  
    int          iPort ,  
    unsigned char * pucManaMsg,  
    int          iIfOpen  
);
```

参数说明：

iPort

[in] 整数，表示端口号。参见 SDT_ResetSAM。

pucManaMsg

[out] 无符号字符指针，证/卡芯片序列号，8 个字节。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值：

0x90 选卡成功

0x81 选卡失败

4.3.3 SDT_ReadMngInfo

读取卡体管理号。

```
int SDT_ReadMngInfo(  
    int          iPort ,  
    unsigned char * pucManageMsg,  
    int          iIfOpen  
);
```

参数说明：

iPort

[in] 整数，表示端口号。参见 SDT_ResetSAM。

pucManageMsg

[out] 无符号字符指针，卡体管理号, 28 字节。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值：

0x90 读成功

其他 读失败（具体含义参见返回码表）

4.3.4 SDT_ReadBaseMsg

读取证/卡固定信息。

```
int SDT_ReadBaseMsg (  
    int          iPort ,  
    unsigned char * pucCHMsg ,  
    unsigned int  * puiCHMsgLen ,  
    unsigned char * pucPHMsg ,  
    unsigned int  * puiPHMsgLen ,  
    int          iIfOpen  
);
```

参数说明：

iPort

[in] 整数，表示端口号。参见 SDT_ResetSAM。

pucCHMsg

[out] 无符号字符指针，指向读到的文字信息。

puiCHMsgLen

[out] 无符号整型数指针，指向读到的文字信息长度。

pucPHMsg

[out] 无符号字符指针，指向读到的照片信息。

puiPHMsgLen

[out] 无符号整型数指针，指向读到的照片信息长度。

iIfOpen

[in] 整数，参见 SdT_ResetSAM。

返回值：

0x90 读固定信息成功

其他 读固定信息失败（具体含义参见返回码表）

4.3.5 SdT_ReadNewAppMsg

读取追加信息。

```
int SdT_ReadNewAppMsg (
    int          iPort ,
    unsigned char * pucAppMsg,
    unsigned int * puiAppMsgLen,
    int          iIfOpen
);
```

参数说明：

iPort

[in] 整数，表示端口号。参见 SdT_ResetSAM。

pucAppMsg

[out] 无符号字符串，指向读到的追加信息。

puiAppMsgLen

[out] 指向整数的指针，指向读到的追加信息长度。

iIfOpen

[in] 整数，参见 SdT_ResetSAM。

返回值：

0x90 读取追加信息成功

其他 读取追加信息失败（具体含义参见返回码表）

5. API 调用说明

5.1 调用顺序

有些 API 的调用有一定先后顺序。我们把这些有先后顺序关系的 API 按不同层次列表如下。（2）级函数执行前必须执行其上的（1），即（1）级函数最先执行，然后可以执行（2）级函数，然后可以执行（3）级函数，然后依次为（4）级、（5）级函数。同一级函数没有先后顺序。其他未列出的函数全是（1）级函数，调用没有先后顺序。

SdT_StartFindIDCard(1)

SdT_SelectIDCard(2)

SDT_ReadBaseMsg(3)

5.2 例子程序(以 C 语言为例)

```
//验证例子程序
#include <stdio.h>
#include <string.h>
#include "sdtapi.h" //动态连接库头文件

void main()
{
    char cInput;
    int iRet;      //返回码
    int iPort;     //端口号
    int iIfOpen;   //是否需要打开端口
    unsigned char pucManaInfo[4];
    unsigned char pucManaMsg [8];
    unsigned char pucCHMsg[512]; //文字信息
    unsigned char pucPHMsg[1024]; //照片信息
    unsigned char pucFPMsg[1024]; //指纹信息
    unsigned int uiCHMsgLen,uiPHMsgLen,

    iPort=1001; //USB 设备
    iIfOpen=0;

    if(iIfOpen==0)
    {
        iRet=SDT_OpenPort(iPort);
        if(iRet!=0x90)
        {
            printf("SDT_OpenPort error, error code is: %02x", iRet);
            SDT_ClosePort(iPort);
            return;
        }
    }

    do //找卡
    {
        iRet=SDT_StartFindIDCard(iPort, pucManaInfo, iIfOpen);
        if(iRet==0x9f)
        {
            iRet=SDT_SelectIDCard (iPort, pucManaMsg,iIfOpen);
            if(iRet==0x90)
                break;
        }
    }
    printf("尚未找到卡,你想继续找卡?继续按\"y\",退出按\"n\" \"n\");
```

```

        scanf("%c", &cInput);
    }
    while(!(cInput=='n'));

    iRet=SDT_ReadBaseMsg(iPort,
                        pucCHMsg,
                        &uiCHMsgLen,
                        pucPHMsg,
                        &uiPHMsgLen,
                        iIfOpen);

    if(iRet!=0x90)
    {
        printf("SDT_ReadBaseMsg  error, error code is: %02x", iRet);
        if(iIfOpen==0)
            SDT_ClosePort(iPort);
        return;
    }
    printf("SDT_ReadBaseMsg OK");
    if(iIfOpen==0)
        SDT_ClosePort(iPort);
    return;
}

```

6. 函数返回码列表

API 返回值列表:

返回值 (16 进制)	意 义
90	操作成功
91	证/卡中此项无内容
9F	寻找找证/卡成功
01	端口打开失败/端口尚未打开/端口号不合法
02	PC 接收超时，在规定的时间内未接收到规定长度的数据。
03	数据传输错误
05	该 SAM_V 串口不可用，只在 SDT_GetCOMBaud 时才有可能返回
10	接收业务终端数据的校验和错
11	接收业务终端数据的长度错。
21	接收业务终端的命令错误，包括命令中的各种数值或逻辑搭配错误
23	越权操作
24	无法识别的错误
80	寻找证/卡失败
81	选取证/卡失败

31	证/卡证认 SAM_V 失败
32	SAM_V 认证证/卡失败
33	信息验证错误
40	无法识别的卡类型
41	读证/卡操作失败
47	取随机数失败
60	SAM_V 自检失败，不能接收命令
66	SAM_V 没经过授权,无法使用

SDT telecom