
身份证解码 SDK 开发包说明

part2 (离线版部分)

郑州中软高科信息技术有限公司

2024年1月

居民身份证嵌入式安全控制模块接口

API 使用手册

版 本

1.1.3

出版日期

2024年1月

著作权注意事项

本书版权为郑州中软高科信息技术有限公司所有。未经郑州中软高科信息技术有限公司书面同意，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，均由郑州中软高科信息技术有限公司随时作出解释，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。

目录

居民身份证嵌入式安全控制模块接口	2
1. 前言	4
2. 系统要求	4
3. API列表	4
4. API详细说明	6
4.1 端口类API	6
4.1.1 SDT_GetCOMBaud	6
4.1.2 SDT_SetCOMBaud	6
4.1.3 SDT_OpenPort	7
4.1.4 SDT_ClosePort	7
4.2 SAM 类API	8
4.2.1 SDT_ResetSAM	8
4.2.2 SDT_SetMaxRFByte	8
4.2.3 SDT_GetSAMStatus	9
4.2.4 SDT_GetSAMID	9
4.2.5 SDT_GetSAMIDToStr	10
4.2.6 SDT_SetLowPower	10
4.2.7 SDT_StartupChk	11
4.3 身份证卡类API	12
4.3.1 SDT_StartFindIDCard	12
4.3.2 SDT_SelectIDCard	12
4.3.3 SDT_ReadBaseMsg	13
4.3.4 SDT_ReadBaseMsgToFile	14
4.3.5 SDT_ReadBaseFPMsg	14
4.3.6 SDT_ReadBaseFPMsgToFile	15
4.3.7 SDT_ReadNewAppMsg	16
5. API调用方式	17
5.1 调用顺序	17
5.2 C 语言示例程序	17
6. 函数返回码表	21

1. 前言

本手册是居民身份证嵌入式安全控制模块（以下有时以 SAM_A 指代）接口API 的使用说明，适用于 Windows 下版本号为4.0.1.2 的API 动态库（sdtapi.dll）和 Linux 下版本号为 4.1.2 的API 动态库（libsdtapi.so）。

2. 系统要求

使用本动态库的 PC机，必须满足下列条件：

- WindowsXP, Windows7, Windows10, Windows11, Linux (对应具体操作系统环境)；
- 至少一个空闲普通串口或USB 口。

3. API 列表

API 分为三类，在下表中列出。

序号	函数名	功能描述	调用层级号
端口类 API			
1.	SDT_GetCOMBaud	查看 SAM_A 串口当前波特率	1
2.	SDT_SetCOMBaud	设置业务终端与 SAM_A 串口的波特率	1
3.	SDT_OpenPort	打开串口/USB 口	1
4.	SDT_ClosePort	关闭串口/USB 口	2
SAM 类 API			
1.	SDT_ResetSAM	对 SAM_A 复位	1
2.	SDT_SetMaxRFByte	设置 SAM_A 与射频模块一帧通信数据的最大字节数	1
3.	SDT_GetSAMStatus	对 SAM_A 进行状态检测	1
4.	SDT_GetSAMID	读取 SAM_A 的编号，输出为十六进制数值	1
5.	SDT_GetSAMIDToStr	读取 SAM_A 的编号，输出为字符串	1
6.	SDT_SetLowPower	启动低功耗模式	1
7.	SDT_SetLPCDMode	启动 LPCD 模式 1	1
8.	SDT_SetLPCDModeII	启动 LPCD 模式 2	1
9.	SDT_SetMainPowerDomainSleep	主电源域休眠	1
10.	SDT_SetSwitchRFPowerDomain	开关射频电源域	1
11.	SDT_SetRFFieldState	设置射频场开关	1
12.	SDT_GetRFFieldState	读取射频场开关状态	1
13.	SDT_SetRFModParm	设置射频调制参数	1
14.	SDT_GetRFModParm	读取射频调制参数	1
15.	SDT_SetSOFParm	设置射频帧 SOF 控制参数	1
16.	SDT_GetSOFParm	读取射频帧 SOF 控制参数	1
17.	SDT_SetEOFParm	设置射频帧 EOF 控制参数	1

18.	SDT_GetEOFParam	读取射频帧EOF 控制参数	1
19.	SDT_SetEGTParm	设置射频帧EGT 控制参数	1
20.	SDT_GetEGTParm	读取射频帧EGT 控制参数	1
21.	SDT_SetRFRecvSignParm	设置射频接收信号参数	1
22.	SDT_GetRFRecvSignParm	读取射频接收信号参数	1
23.	SDT_SetTestOutPinParm	设置测试输出引脚参数	1
24.	SDT_GetTestOutPinParm	读取测试输出引脚参数	1
25.	SDT_SetRecvConfigParm	设置接收配置参数	1
26.	SDT_GetRecvConfigParm	读取接收配置参数	1
27.	SDT_SetClockDelayAutomaticParm	设置时钟延时自动选择参数	1
28.	SDT_GetClockDelayAutomaticParm	读取时钟延时自动选择参数	1
29.	SDT_SetRecvCircuitResistanceValue	设置接收电路电阻值参数	1
30.	SDT_GetRecvCircuitResistanceValue	读取接收电路电阻值参数	1
31.	SDT_SetR3VoltScanOnRecvCircuit	接收电路使用内部分压 电阻 电 压扫描	1
32.	SDT_SetR2VoltScanOnRecvCircuit	接收电路使用外部分压 电阻 电 压扫描	1
33.	SDT_GetLPCDParam	LPCD 参数读取	1
34.	SDT_WithoutCardParamDebug	LPCD 无卡参数调试	1
35.	SDT_WithoutCardParamConfig	LPCD 无卡参数配置	1
36.	SDT_WithCardParamDebug	LPCD 有卡参数调试	1
37.	SDT_WithCardParamConfig	LPCD 有卡参数配置	1
38.	SDT_WorkParamConfig	LPCD 工作参数配置	1
39.	SDT_GetGALoadState	读取GA码加载状态	1
40.	SDT_GALoad	GA码加载命令	2
41.	SDT_StartupChk	SAM_A_IV启动	见注
身份证卡类 API			
1.	SDT_StartFindIDCard	寻找居民身份证	1
2.	SDT_SelectIDCard	选取居民身份证	2
3.	SDT_ReadBaseMsg	核验居民身份证机读文字信息和相片信息相片	3
4.	SDT_ReadBaseMsgToFile	核验居民身份证机读文字信息和相片信息，并将其存入用户指定文件	3
5.	SDT_ReadBaseFPMsg	核验居民身份证机读文字信息、相片信息和指纹信息	3
6.	SDT_ReadBaseFPMsgToFile	核验居民身份证机读文字信息、相片信息和指纹信息，并将其存入用户指定文件	3
7.	SDT_ReadNewAppMsg	核验追加地址信息	3

注：1.调用 SDT_GALoad 函数可获得启动码，记录得到的启动码，后续可直接使用该启动码调用 SDT_StartupChk 函数；2.加电后，GA码加载状态下需要调用一次SDT_StartupChk 函数后才可使用身份证卡类API进行身份证信息核验。

4. API 详细说明

4.1 端口类API

4.1.1 SDT_GetCOMBaud

查看 SAM_A 串口当前波特率(该函数只用于 SAM_A 采用串口的情形, 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。

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 115200bps。

返回值:

0x90 成功。
0x01 端口打开失败/端口号不合法。
0x05 无法获得该SAM_A 的波特率, 该SAM_A 串口不可用。注:

[in]表示该参数为输入参数, [out]表示该参数为输出参数, 下同。

4.1.2 SDT_SetCOMBaud

设置业务终端及 SAM_A 的串口的波特率(该函数只用于 SAM_A 采用串口的情形, 如果采用 USB 接口则不支持本函数)。

```
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_A 通信的波特率 (SAM_A 出厂时默认值为 115200bps)。业务终端以当前使用的波特率与 SAM_A

通信,发出设置SAM_A新波特率的命令。*uiCurrBaud*的合法取值为下列数值之一: 115200、57600、38400、19200、9600。如果*uiCurrBaud*数值不是这些值 之一,函数返回0x21; 如果已设置的波特率为合法取值之一,但与*uiCurrBaud* 不一致, 则函数返回 0x02,表示设置不成功。

uiSetBaud

[in] 无符号整数,将要设置的 SAM_A 与业务终端通信波特率。*uiSetBaud* 只能取下列值之一: 115200、57600、38400、19200、9600。如果输入*uiSetBaud* 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。
- 0x01 端口打开失败/端口号不合法。
- 0x02 超时, 设置不成功。
- 0x21 *uiCurrBaud*、*uiSetBaud*输入参数数值错误。

4. 1. 3 SDT_OpenPort

打开串口/USB 口。

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

参数说明:

iPort

[in] 整数,表示端口号。串口和USB都只支持 16个,分别为 1~16(十进制) 为 串口, 1001~1016(十进制)为USB 口。

串口	1~16	例如: 1: 串口 1 (COM1) 2: 串口 2 (COM2)
USB 口	1001~1016	例如: 1001: USB1 1002: USB2

返回值:

- 0x90 打开端口成功。
- 0x01 打开端口失败/端口号不合法。

4. 1. 4 SDT_ClosePort

关闭串口/USB 口。

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

参数说明:

iPort

[in] 整数，表示端口号。参见 SDT_OpenPort。

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_A复位。

```
int SDT_ResetSAM (  
    int      iPort,  
    int      iIfOpen  
);
```

参数说明:

iPort

[in] 整数，表示端口号，参见 SDT_OpenPort。

iIfOpen

[in] 整数，0 表示不在该函数内部打开和关闭串口，此时应确保之前调用了 SDT_OpenPort 打开端口，并且应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。

其它 失败(具体含义参见 6 函数返回码表)。

4.2.2 SDT_SetMaxRFByte

设置SAM_A 与射频模块一帧通信数据的最大字节数。

```
int SDT_SetMaxRFByte (  
    int      iPort,  
    unsigned char ucByte,  
    int      bIfOpen  
);
```

参数说明:

iPort

[in] 整数，表示端口号。参见 SDT_OpenPort。

ucByte

[in] 无符号字符，24~255，表示射频适配器一帧通信数据的最大字节数。

iIfOpen

[in] 整数, 参见 SDT_ResetSAM。

返回值:

0x90 成功。

其它 失败(具体含义参见 6 函数返回码表)。

4.2.3 SDT_GetSAMStatus

对 SAM_A 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_OpenPort。

iIfOpen

[in] 整数, 参见 SDT_ResetSAM。

返回值:

0x90 SAM_A 正常。

0x60 自检失败, 不能接收命令。

其它 失败(具体含义参见 6 函数返回码表)。

4.2.4 SDT_GetSAMID

读取 SAM_A 的编号, 输出为十六进制数值。

```
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_OpenPort。

pucSAMID

[out] 无符号字符串指针, SAM_A 编号, 16 字节。该指针指向的存储空间由调用者分配。

iIfOpen

[in] 整数, 参见 SDT_ResetSAM。

返回值:

0x90 成功。
其它 失败(具体含义参见 6 函数返回码表)。

4.2.5 SDT_GetSAMIDToStr

读取 SAM_A 的编号, 输出为字符串。

```
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_OpenPort。

pcSAMID

[out] 字符串指针, SAM_A 编号。该指针指向的存储空间由调用者分配, 不得小于 40 字节。

iIfOpen

[in] 整数, 参见 SDT_ResetSAM。

返回值:

0x90 成功。
其它 失败(具体含义参见 6 函数返回码表)。

4.2.6 SDT_SetLowPower

启动低功耗模式。

```
int SDT_SetLowPower(
    int      iPortID,
    int      iIfOpen
);
```

参数说明:

iPort

[in] 整数, 表示端口号。参见 SDT_OpenPort。

iIfOpen

[in] 整数, 参见 SDT_ResetSAM。

返回值:

0x90 成功。
其他 失败(具体含义参见 6 函数返回码表)。

4.2.7 SDT_StartupChk

SAM_A_IV 启动。

注：1. 模块加电后，GA码加载状态下，需先调用该接口开启SAM核验功能，才能进行身份信息核验操作；

2. 调用SDT_GALoad后，可调用该接口验证启动码有效性。

```
int SDT_StartupChk(  
    int iPort,  
    unsigned char * pucEncCode,  
    int iIfOpen  
);
```

参数说明：

iPort

[in] 整数，表示端口号。参见 SDT_OpenPort。

pucEncCode

[in] 无符号字符指针，16字节。启动码：通过GA码加载命令获得。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值:

0x90 成功。

其他 失败(具体含义参见 6 函数返回码表)。

4.3 身份证卡类API

4.3.1 SDT_StartFindIDCard

寻找居民身份证。

```
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_OpenPort。

pucManaInfo

[out] 无符号字符型指针，4个字节 0x00。该指针指向的存储空间由调用者分配。

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

选取居民身份证。

```
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_OpenPort。

pucManaMsg

[out] 无符号字符型指针，8个字节 0x00。该指针指向的存储空间由调用者分配。

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_ReadBaseMsg

核验居民身份证机读文字信息和相片信息。

```
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_OpenPort。

pucCHMsg

[out] 无符号字符型指针，指向读到的文字信息，其长度由 *puiCHMsgLen* 参数输出。该指针指向的存储空间由调用者分配，不得小于 256 字节。

puiCHMsgLen

[out] 无符号整型数指针，指向读到的文字信息长度，最长 256 字节。

pucPHMsg

[out] 无符号字符型指针，指向读到的相片信息，其长度由 *puiPHMsgLen* 参数输出。该指针指向的存储空间由调用者分配，不得小于 1024 字节。

puiPHMsgLen

[out] 无符号整型数指针，指向读到的相片信息长度，最长 1024 字节。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值:

0x90 读机读文字信息和相片信息成功。

其它 读机读文字信息和相片信息失败(具体含义参见6 函数返回码表)。

4.3.4 SDT_ReadBaseMsgToFile

读取居民身份证机读文字信息和相片信息，将读取到的信息写到输入参数所指定的文件中。

```
int SDT_ReadBaseMsgToFile (  
    int          iPort,  
    char *       pcCHMsgFileName,  
    unsigned int * puiCHMsgFileLen,  
    char *       pcPHMsgFileName,  
    unsigned int * puiPHMsgFileLen,  
    int          iIfOpen  
);
```

参数说明:

iPort

[in] 整数，表示端口号。参见 SDT_OpenPort。

pcCHMsgFileName

[in] 字符型指针，由用户指定的文件名，将读取到的居民身份证机读文字信息写入该文件。

puiCHMsgFileLen

[out] 无符号整型数指针，文件的长度。

pcPHMsgFileName

[in] 字符型指针，由用户指定的文件名，将读取到的居民身份证机读相片信息写入该文件。

puiPHMsgFileLen

[out] 无符号整型数指针，文件的长度。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值:

0x90 读机读文字信息和相片信息成功。

其它 读机读文字信息和相片信息失败（具体含义参见6 函数返回码表）。

4.3.5 SDT_ReadBaseFPMsg

核验居民身份证机读文字信息、相片信息和指纹信息。

注意：该接口只能用于支持读取指纹信息的 SAM_A。

```
int SDT_ReadBaseFPMsg (  
    int          iPort,  
    unsigned char * pucCHMsg,  
    unsigned int * puiCHMsgLen,  
    unsigned char * pucPHMsg,
```

```

    unsigned int * puiPHMsgLen,
    unsigned char * pucFPMsg,
    unsigned int * puiFPMsgLen,
    int          iIfOpen
);

```

参数说明:*iPort*

[in] 整数，表示端口号。参见 SDT_OpenPort。

pucCHMsg[out] 无符号字符型指针，指向读到的机读文字信息，其长度由 *puiCHMsgLen* 参数输出。该指针指向的存储空间由调用者分配，不得小于 256 字节。*puiCHMsgLen*

[out] 无符号整型数指针，指向读到的机读文字信息长度，最长 256 字节。

pucPHMsg[out] 无符号字符型指针，指向读到的机读相片信息，其长度由 *puiPHMsgLen* 参数输出。该指针指向的存储空间由调用者分配，不得小于 1024 字节。*puiPHMsgLen*

[out] 无符号整型数指针，指向读到的机读相片信息长度，最长 1024 字节。

pucFPMsg[out] 无符号字符型指针，指向读到的指纹信息，其长度由 *puiFPMsgLen* 参数输出。该指针指向的存储空间由调用者分配，不得小于 1024 字节。*puiFPMsgLen*

[out] 无符号整型数指针，指向读到的指纹信息长度，最长 1024 字节。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值:

- | | |
|------|--|
| 0x90 | 读机读文字信息、相片信息和指纹信息成功。如果 <i>puiFPMsgLen</i> 指向的无符号整数等于 0，表明身份证中没有指纹信息；如果 <i>puiFPMsgLen</i> 指向的无符号整数大于 0，表明正确地读出了指纹信息。 |
| 0x21 | 错误的命令。可使用 SDT_ReadBaseMsg 接口读取机读文字和相片信息。 |
| 0x37 | 指纹信息验证错误。 |
| 其它 | 读取机读文字信息、相片信息和指纹信息失败(具体含义参见 6 函数返回码表)。 |

4.3.6 SDT_ReadBaseFPMsgToFile

读取居民身份证机读文字信息、相片信息和指纹信息，将读取到的信息写到输入参数所指定的文件中。

注意：该接口只能用于支持读取指纹信息的 SAM_A。

```

int SDT_ReadBaseFPMsgToFile (
    int          iPort,

```

```

char *          pcCHMsgFileName,
unsigned int *  puiCHMsgFileLen,
char *          pcPHMsgFileName,
unsigned int *  puiPHMsgFileLen,
char *          pcFPMsgFileName,
unsigned int *  puiFPMsgFileLen,
int             iIfOpen
);

```

参数说明:*iPort*

[in] 整数，表示端口号。参见 SDT_OpenPort。

pcCHMsgFileName

[in] 由用户指定的文件名，将读取到的居民身份证机读文字信息写入该文件。

puiCHMsgFileLen

[out] 无符号整型数指针，文件的长度。

pcPHMsgFileName

[in] 由用户指定的文件名，将读取到的居民身份证机读相片信息写入该文件。

puiPHMsgFileLen

[out] 无符号整型数指针，文件的长度。

puiFPMsgFileName

[in] 由用户指定的文件名，将读取到的居民身份证指纹信息写入该文件。

puiFPMsgFileLen

[out] 无符号整型数指针，文件的长度。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值:

- 0x90 读机读文字信息、相片信息和指纹信息成功。如果 *puiFPMsgFileLen* 指向的无符号整数等于 0，表明身份证中没有指纹信息；如果 *puiFPMsgFileLen* 指向的无符号整数大于 0，表明正确地读出了指纹信息。
- 0x21 错误的命令。可使用 SDT_ReadBaseMsgToFile 接口读取机读文字和相片信息。
- 0x37 指纹信息验证错误。
- 其它 读取机读文字信息、相片信息和指纹信息失败(具体含义参见 6 函数返回码表)。

4.3.7 SDT_ReadNewAppMsg

核验追加地址信息。

```

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_OpenPort。

pucAppMsg

[out] 无符号字符串，指向读到的追加地址信息，其长度由 *puiAppMsgLen* 参数输出。该指针指向的存储空间由调用者分配，不得小于 70 字节。

puiAppMsgLen

[out] 指向整数的指针，指向读到的追加地址信息长度，最长 70 字节。

iIfOpen

[in] 整数，参见 SDT_ResetSAM。

返回值:

0x90	读取追加地址信息成功。
0x91	居民身份证中无追加地址信息。
其它	读取追加地址信息失败(具体含义参见 6 函数返回码表)。

5. API 调用方式

5.1 调用顺序

API 的调用有先后顺序，每一个 API 都有一个调用层级号。层级号大的函数执行前必须执行比其层级号小 1 的函数，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，以此类推。同一层级函数的调用没有先后顺序。(注：模块加电后，在 GA 码加载状态下，需要调用一次 SDT_StartupChk 接口开启核验功能后才能正确读取完整的身份信息；GA 码未加载状态下，不需要调用 SDT_StartupChk 接口，可直接读取部分身份信息。)

SDT_StartupChk (0)

SDT_StartFindIDCard(1)

SDT_SelectIDCard(2)

SDT_ReadBaseMsg(3)

SDT_ReadNewAppMsg(3)

SDT_ReadBaseFPMsg(3)

SDT_ReadBaseFPMsgToFile(3)

SDT_ReadBaseMsgToFile (3)

5.2 C 语言示例程序

```
//C 语言验证示例程
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 pucEncCode[16]; //启动码
    unsigned char pucManaInfo[4];
    unsigned char pucManaMsg[8];
    unsigned char pucCHMsg[256]; //文字信息最长 256 字节
    unsigned char pucPHMsg[1024]; //相片信息最长 1024 字节
    unsigned char pucFPMsg[1024]; //指纹信息最长 1024 字节
    unsigned int uiCHMsgLen, uiPHMsgLen, uiFPMsgLen;

    iPort=1001; //USB 接口
    iIfOpen=0;
    if(iIfOpen==0)
    {
        iRet=SDT_OpenPort(iPort);
        if(iRet!=0x90)
        {
            printf("打开端口(%d)失败, 返回值为: %02x", iPort, iRet);
            SDT_ClosePort(iPort);
            return;
        }
    }
}
```

/*调用 SDT_StartupChk 接口时, 需要将记录的启动码赋值给 pucEncCode 参数。只有该接口调用成功后, 才能正确读取身份证信息。*/

```
iRet=SDT_StartupChk(iPort, pucEncCode, iIfOpen);
if(iRet!=0x90)
{
    printf("开启 SAM 核验功能失败, 返回值为: %02x\n", iRet);
    return;
}
do //找卡 {
    iRet=SDT_StartFindIDCard(iPort, pucManaInfo, iIfOpen);
    if(iRet==0x9f)
    {
        iRet=SDT_SelectIDCard (iPort, pucManaMsg, iIfOpen);
```

```
        if(iRet!=0x90)
        {
            printf("选卡失败, 返回值为: %02x\n",iRet);
            printf("请重新放卡, 进行找卡、选卡?继续按'y',退出按'n' \n"); }
        else
        {
            break;
        }
    }
else
{
    printf("未找到身份证卡, 请重新放卡。继续按'y',退出按'n' \n"); }
    scanf("%c",
    &cInput); }
while(!(cInput=='n'));
if(cInput=='n')
{
    printf("放弃找卡,退出程序\n");
    return ;
}
/* SDT_ReadBaseMsg()和SDT_ReadBaseFPMsg()都要在SDT_SelectIDCard()成功后
调用。SDT_ReadBaseMsg()只读出文字和相片信息; SDT_ReadBaseFPMsg()读出文字、相片 和
指纹信息。这两个接口调用没有顺序要求。*/
iRet=SDT_ReadBaseMsg(iPort,
                    pucCHMsg,
                    &uiCHMsgLen,
                    pucPHMsg,
                    &uiPHMsgLen,
                    iIfOpen);

if(iRet!=0x90)
{
    printf("读取身份证机读文字信息和相片信息失败, 返回值为: %02x\n",iRet);
    if(iIfOpen==0)
        SDT_ClosePort(iPort);
    return;
}
printf("SDT_ReadBaseMsg OK\n");

iRet=SDT_ReadBaseFPMsg(iPort,pucCHMsg,
                        &uiCHMsgLen,
                        pucPHMsg,
                        &uiPHMsgLen,
```

```
        pucFPMsg,  
        &uiFPMsgLen,  
        iIfOpen);  
  
    if(iRet==0x21)  
    {  
        printf("此安全控制模块不支持 SDT_ReadBaseFPMsg 接口!返回值为: %02x\n  
请使用 SDT_ReadBase Msg 接口。 \n",iRet);  
    }  
    elseif(iRet==0x37)  
    {  
        printf("指纹信息验证错误! 返回值为: %02x\n",iRet); }  
    elseif(iRet!=0x90)  
    {  
        printf(" 读取身份证机读文字信息、相片信息和指纹信息失败， 返  
回 值 为: %02x\n",iRet);  
    }  
    else  
    {  
        if(uiFPMsgLen==0) //身份证中没有指纹信息  
        {  
            printf("身份证卡中没有指纹信息\n");  
        }  
        else  
        {  
            printf("SDT_ReadBaseFPMsg  
OK\n"); }  
        }  
    if(iIfOpen==0)  
        SDT_ClosePort(iPort);  
}
```

6. 函数返回码表

返回值	意 义
0x90	操作成功
0x91	居民身份证中无此项内容
0x9F	寻找居民身份证成功
0x01	端口打开失败/端口尚未打开/端口号不合法
0x02	PC 接收超时，在规定的时间内未接收到规定长度的数据
0x03	数据传输错误
0x05	SAM_A 串口不可用，只有 SDT_GetCOMBaud 函数返回
0x09	打开文件失败
0x10	接收业务终端数据的校验和错
0x11	接收业务终端数据的长度错
0x21	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
0x23	越权操作
0x24	无法识别的错误
0x80	寻找居民身份证失败
0x81	选取居民身份证失败
0x31	居民身份证认证 SAM_A 失败
0x32	SAM_A 认证居民身份证失败
0x33	信息验证失败
0x37	指纹信息验证错误
0x3F	信息长度错误
0x40	无法识别的居民身份证类型
0x41	读居民身份证操作失败
0x47	取随机数失败
0x60	SAM_A 自检失败，不能接收命令
0x66	SAM_A 没经过授权，无法使用
0x83	射频场参数配置失败
0x85	射频状态错误
0x86	嵌入式 SAM 认证失败
0x87	上位机认证失败