



# A76xx Series Open SDK\_ SIM 卡 API\_使用说明

LTE 模组

芯讯通无线科技(上海)有限公司  
上海市长宁区临虹路289号3号楼芯讯通总部大楼  
电话: 86-21-31575100  
技术支持邮箱: support@simcom.com  
官网: www.simcom.com

名称:	A76xx Series Open SDK_SIM卡API_使用说明
版本:	V1.00
类别:	应用文档
状态:	已发布

## 版权声明

本手册包含芯讯通无线科技（上海）有限公司（简称：芯讯通）的技术信息。除非经芯讯通书面许可，任何单位和个人不得擅自摘抄、复制本手册内容的部分或全部，并不得以任何形式传播，违反者将被追究法律责任。对技术信息涉及的专利、实用新型或者外观设计等知识产权，芯讯通保留一切权利。芯讯通有权在不通知的情况下随时更新本手册的具体内容。

本手册版权属于芯讯通，任何人未经我公司书面同意进行复制、引用或者修改本手册都将承担法律责任。

### 芯讯通无线科技(上海)有限公司

上海市长宁区临虹路289号3号楼芯讯通总部大楼

电话：86-21-31575100

邮箱：simcom@simcom.com

官网：www.simcom.com

了解更多资料，请点击以下链接：

<http://cn.simcom.com/download/list-230-cn.html>

技术支持，请点击以下链接：

<http://cn.simcom.com/ask/index-cn.html> 或发送邮件至 [support@simcom.com](mailto:support@simcom.com)

版权所有 © 芯讯通无线科技(上海)有限公司 2023，保留一切权利。

## Version History

Version	Date	Owner	What is new
V1.00	2022-11-03		第一版

SIMCom  
Confidential

## About this Document

本文档适用于 A1803S open 系列、A1603 open 系列、A1606 open 系列。

SIMCom  
Confidential

# 目录

版权声明 .....	2
Version History .....	3
About this Document .....	4
目录 .....	5
缩略语 .....	7
1 变量定义 .....	8
1.1 SC_simcard_err_e 枚举（头文件：#include “simcom_simcard.h”） .....	8
1.2 SC_SIMCARD_CMD_TYPE 枚举（头文件：#include “simcom_simcard.h”） .....	8
1.3 SC_HotSwapCmdType_e 枚举（头文件：#include “simcom_simcard.h”） .....	8
1.4 UrclndMsg_st 结构体（头文件：#include “simcom_simcard.h”） .....	9
1.5 HotSwapFileType_st 结构体（头文件：#include “simcom_simcard.h”） .....	9
1.6 scSimcardState 枚举（头文件：#include “simcom_simcard.h”） .....	9
1.7 Hplmn_st 结构体（头文件：#include “simcom_simcard.h”） .....	9
1.8 STK（头文件：#include “simcom_simcard.h”） .....	10
1.9 sSIM_Switch 结构体（头文件：#include “simcom_soft_sim.h”） .....	10
1.10 CrsmResponse_st 结构体（头文件：#include “simcom_soft_sim.h”） .....	10
2 API 介绍 .....	11
1.1 sAPI_SimcardPinGet 获取 SIM 卡状态 .....	11
1.2 sAPI_SimcardPinSet 设置 SIM 卡的 pin 或 puk .....	11
1.3 sAPI_SysGetImsi 获取 SIM 卡 imsi .....	11
1.4 sAPI_SysGetIccid 获取 SIM 卡 ICCID .....	12
1.5 sAPI_SysGetHplmn 获取 SIM 卡 Hplmn .....	12
1.6 sAPI_SimcardHotSwapMsg SIM 卡热插拔配置 .....	12
1.7 sAPI_SimcardSwitchMsg 设置 SIM1 或 SIM2 为主卡 .....	13
1.8 sAPI_SysGetBindSim 获取接口目前操作那一张 SIM 卡 .....	13
1.9 sAPI_SysSetBindSim 设置接口操作哪张 SIM 卡 .....	13
1.10 sAPI_SimLockSet 锁定/解锁 SIM 卡 .....	14
1.11 sAPI_SysGetSubscriber 获取电话号码 .....	14
1.12 sAPI_SoftSimSwitch 使能/失能软卡 SoftSim 功能 .....	14
1.13 sAPI_StkService STK 服务 .....	14
1.14 sAPI_VsimOperate 使能/失能 VSIM 功能 .....	15
1.15 sAPI_SysGetDualSimType 查询双卡模式 .....	15
1.16 sAPI_SysSetDualSimType 设置双卡模式 .....	16
1.17 sAPI_SimRestrictedAccess 访问 SIM 数据库 .....	16

<b>3 示例 .....</b>	<b>18</b>
3.1 使用 sAPI_SysGetBindSim 获取接口操作的哪一张 SIM 卡 .....	18
3.2 使用 sAPI_SysSetBindSim 选择接口操作的哪一张 SIM 卡 .....	18
3.3 使用 sAPI_SimcardPinGet 获取 SIM 卡状态 .....	19
3.4 使用 sAPI_SimcardPinSet 操作 PIN/PUK 码 .....	19
3.5 使用 sAPI_SysGetImsi 获取 IMSI .....	21
3.6 使用 sAPI_SysGetIccid 获取 ICCID .....	21
3.7 使用 sAPI_SysGetHplmn 获取 HPLMN .....	22
3.8 使用 sAPI_SimcardHotSwapMsg 配置 SIM 卡热插拔功能 .....	22
3.9 使用 sAPI_SimcardSwitchMsg 设置主卡 .....	24
3.10 使用 sAPI_SimRestrictedAccess 操作 SIM 数据库 .....	25
<b>4 附录 .....</b>	<b>27</b>

## 缩略语

IMSI	国际移动用户识别码( International Mobile Subscriber Identity)
ICCID	集成电路卡识别码( Integrate circuit card identity)
PIN	个人识别密码( Personal Identification Number)
PUK	个人识别密码解锁码( Personal Identification Number Unlock Key)

SIMCom  
Confidential

## 1 变量定义

### 1.1 SC\_simcard\_err\_e 枚举（头文件：#include “simcom\_simcard.h”）

```
typedef enum {  
    SC_SIM_RETURN_SUCCESS,  
    SC_SIM_RETURN_FAIL,  
    SC_SIM_RETURN_UNKNOW  
}SC_simcard_err_e;
```

### 1.2 SC\_SIMCARD\_CMD\_TYPE 枚举( 头文件 :#include “simcom\_simcard.h”)

```
typedef enum{  
    SC_SIMCARD_INVALID,  
    SC_SIMCARD_SWITCHCARD,  
    SC_SIMCARD_HOTPLUG,  
    SC_SIMCARD_URCIND,  
    SC_SIMCARD_TYPE_MAX  
}SC_SIMCARD_CMD_TYPE;
```

### 1.3 SC\_HotSwapCmdType\_e 枚举（头文件：#include “simcom\_simcard.h”）

```
typedef enum{  
    SC_HOTSWAP_INVALID,  
    SC_HOTSWAP_QUERY_STATE,  
    SC_HOTSWAP_QUERY_LEVEL,  
    SC_HOTSWAP_SET_SWITCH,  
    SC_HOTSWAP_SET_LEVEL,  
    SC_HOTSWAP_QUERY_OUTSTATE,  
    SC_HOTSWAP_SET_OUTSWITCH,  
    SC_HOTSWAP_TYPE_MAX  
}SC_HotSwapCmdType_e;
```



## 1.4 UrclndMsg\_st 结构体 ( 头文件 : #include “simcom\_simcard.h” )

```
typedef struct
{
    UINT32 status;
    UINT32 mask;
}UrclndMsg_st;
```

## 1.5 HotSwapFileType\_st 结构体 ( 头文件 : #include “simcom\_simcard.h” )

```
typedef struct
{
    BOOL PlugInEnable;           /*SIM1 1:plug in is enabled 0: plug in is disabled*/
    BOOL trigger;                /*SIM1 1:high level trigger 0:Low level trigger*/
    BOOL PlugOutEnable;          /*SIM1 1:plug out is enabled 0: plug out is disable*/
    BOOL PlugInEnable_1;         /*SIM2 1:plug in is enabled 0: plug in is disabled*/
    BOOL trigger_1;              /*SIM2 1:high level trigger 0:Low level trigger*/
    BOOL PlugOutEnable_1;        /*SIM2 1:plug out is enabled 0: plug out is disable*/
}HotSwapFileType_st;
```

## 1.6 scSimcardState 枚举 ( 头文件 : #include “simcom\_simcard.h” )

```
enum scSimcardState
{
    SC_SIM_READY = 0,
    SC_SIM_PIN,
    SC_SIM_PUK,
    SC_SIM_BLOCKED,
    SC_SIM_REMOVED,
    SC_SIM_CRASH,
    SC_SIM_NOINSRETED,
    SC_SIM_UNKNOW
};
```

## 1.7 Hplmn\_st 结构体 ( 头文件 : #include “simcom\_simcard.h” )

```
typedef struct
{
    char spn[64];/*Service provider name from SIM*/
    char mcc[4];/*Mobile Country Code*/
    char mnc[4];/*Mobile Network Code*/
}Hplmn_st;
```

## 1.8 STK（头文件：#include “simcom\_simcard.h”）

```
#define SC_SIM_MAX_CMD_DATA_SIZE    261
#define SC_NULL_TERMINATOR_LENGTH   1

typedef INT32 scStkOpt; /*refer _scStkOpt*/

typedef struct
{
    scStkOpt opt;
    char StkData[SC_SIM_MAX_CMD_DATA_SIZE * 2 + SC_NULL_TERMINATOR_LENGTH];
    INT32 len;
}StkMsg_st;
```

## 1.9 sSIM\_Switch 结构体（头文件：#include “simcom\_soft\_sim.h”）

```
typedef struct
{
    BOOL v_on; /*TRUE: Send the APDU data from modem to embedded application.FALSE: It is directed
to the SIM card.*/
}sSIM_Switch;
```

## 1.10 CrsmResponse\_st 结构体（头文件：#include “simcom\_soft\_sim.h”）

```
typedef struct
{
    int sw1;
    int sw2;
    char data[2 * 261 + 50];
}CrsmResponse_st;
```

## 2API 介绍

### 1.1 sAPI\_SimcardPinGet 获取 SIM 卡状态

接口:	SC_simcard_err_e sAPI_SimcardPinGet(UINT8 *gcpin)
输入:	无
输出:	gcpin: 当前 SIM 卡的状态。状态值请参考 scSimcardState 枚举
返回值:	成功: 0 失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

### 1.2 sAPI\_SimcardPinSet 设置 SIM 卡的 pin 或 puk

接口:	SC_simcard_err_e sAPI_SimcardPinSet(char *oldpin, char *newpin, int new)
输入:	oldpin: 旧的 PIN 码, 至少 4 位数 newpin: 新的 PIN 码, 至少 4 位数 new: 是否需要新的 PIN 码 (1: 需要 0: 不需要)
输出:	无
返回值:	成功: 0 失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

### 1.3 sAPI\_SysGetImsi 获取 SIM 卡 imsi

接口:	SC_simcard_err_e sAPI_SysGetImsi(char *ImsiValue)
输入:	无
输出:	ImsiValue: SIM 卡的 IMSI
返回值:	成功: 0 失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

## 1.4 sAPI\_SysGetIccid 获取 SIM 卡 ICCID

<b>接口:</b>	SC_simcard_err_e sAPI_SysGetIccid(char *IccidValue)
<b>输入:</b>	无
<b>输出:</b>	IccidValue: SIM 卡的 ICCID
<b>返回值:</b>	成功: 0      失败: 1 (参考 SC_simcard_err_e)
<b>NOTE:</b>	无

## 1.5 sAPI\_SysGetHplmn 获取 SIM 卡 Hplmn

<b>接口:</b>	SC_simcard_err_e sAPI_SysGetHplmn(char *HplmnValue)
<b>输入:</b>	无
<b>输出:</b>	HplmnValue: SIM 卡的 HPLMN
<b>返回值:</b>	成功: 0      失败: 1 (参考 SC_simcard_err_e)
<b>NOTE:</b>	无

## 1.6 sAPI\_SimcardHotSwapMsg SIM 卡热插拔配置

<b>接口:</b>	SC_simcard_err_e sAPI_SimcardHotSwapMsg (SC_HotSwapCmdType_e opt, UINT8 param, sMsgQRef msgQ)
<b>输入:</b>	<p>opt : SC_HOTSWAP_QUERY_STATE: 查询 SIM 卡热插拔使能状态 (插入)</p> <p>SC_HOTSWAP_QUERY_LEVEL: 查询 SIM 卡插入识别电平</p> <p>SC_HOTSWAP_SET_SWITCH: 设置 SIM 卡热插拔使能状态 (插入)</p> <p>SC_HOTSWAP_SET_LEVEL: 设置 SIM 卡插入识别电平</p> <p>SC_HOTSWAP_QUERY_OUTSTATE : 查询 SIM 卡热插拔使能状态 (拔出)</p> <p>SC_HOTSWAP_SET_OUTSWITCH : 设置 SIM 卡热插拔使能状态 (拔出)</p> <p>param:</p> <p>opt 为 SC_HOTSWAP_SET_SWITCH/SC_HOTSWAP_SET_OUTSWITCH 时 (1: 使能 0: 失能); opt 为 SC_HOTSWAP_SET_LEVEL 时 (1: 高电平触发模式 0: 低电平触发模式); opt 为其他的时候, 此参数无效</p>

	msgQ: 结果将通过 msgQ 返回
输出:	无
返回值:	成功: 0      失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

## 1.7 sAPI\_SimcardSwitchMsg 设置 SIM1 或 SIM2 为主卡

接口:	SC_simcard_err_e sAPI_SimcardSwitchMsg(UINT8 status, sMsgQRef msgQ)
输入:	status: 0:SIM1    1:SIM2 msgQ: 结果将通过 msgQ 返回
输出:	无
返回值:	成功: 0      失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

## 1.8 sAPI\_SysGetBindSim 获取接口目前操作那一张 SIM 卡

接口:	SC_simcard_err_e sAPI_SysGetBindSim(UINT8 *bindsim)
输入:	bindsim: 0:SIM1    1:SIM2
输出:	无
返回值:	成功: 0      失败: 1 (参考 SC_simcard_err_e)
NOTE:	默认 SIM1

## 1.9 sAPI\_SysSetBindSim 设置接口操作哪张 SIM 卡

接口:	SC_simcard_err_e sAPI_SysSetBindSim(UINT8 bindsim)
输入:	bindsim: 0:SIM1    1:SIM2
输出:	无
返回值:	成功: 0      失败: 1 (参考 SC_simcard_err_e)
NOTE:	默认 SIM1

## 1.10 sAPI\_SimLockSet 锁定/解锁 SIM 卡

接口:	SC_simcard_err_e sAPI_SimLockSet(int mode, char *pin)
输入:	mode: 1:LOCK 2:UNLOCK pin: SIM 卡的 PIN 码
输出:	无
返回值:	成功: 0 失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

## 1.11 sAPI\_SysGetSubscriber 获取电话号码

接口:	SC_simcard_err_e sAPI_SysGetSubscriber (char *SubscriberValue)
输入:	SubscriberValue: SIM 卡的电话号码
输出:	无
返回值:	成功: 0 失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

## 1.12 sAPI\_SoftSimSwitch 使能/失能软卡 SoftSim 功能

接口:	scSoftSimRet sAPI_SoftSimSwitch(sSIM_Switch enSwitch)
输入:	enSwitch: 0:失能 1:使能
输出:	无
返回值:	成功: 0 失败: -1 (参考_scSoftSimRet)
NOTE:	无

## 1.13 sAPI\_StkService STK 服务

接口:	SC_simcard_err_e sAPI_StkService(scStkOpt opt,char *data,StkResult
-----	--------------------------------------------------------------------

输入:	*output)
	opt: SC_STK_CMD_ENABLE_SIMAT SC_STK_CMD_DOWNLOAD_PROFILE SC_STK_CMD_GET_CAP_INFO SC_STK_CMD_GET_PROFILE SC_STK_CMD_SEND_ENVELOPE SC_STK_CMD_PROACTIVE SC_STK_CMD_SETUP_CALL SC_STK_CMD_DISPLAY_INFO SC_STK_CMD_END_SESSION SC_STK_CMD_ENABLE_AUTORESP_PROACTIVE data: 数据
输出:	output: 返回结果和数据
返回值:	成功: 0     失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

### 1.14sAPI\_VsimOperate 使能/失能 VSIM 功能

接口:	SC_simcard_err_e sAPI_VsimOperate(UINT8 opt, UINT8 *data)
输入:	opt: 0:查询 VSIM 使能状态    1:使能 VSIM(1:使能    2:失能)
输出:	data: 接口返回数据
返回值:	成功: 0     失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

### 1.15sAPI\_SysGetDualSimType 查询双卡模式

接口:	SC_simcard_err_e sAPI_SysGetDualSimType(UINT8 *type)
输入:	无
输出:	type: 双卡模式

	0: DUAL SIM DUAL STANDBY 1: DUAL SIM SINGLE STANDBY 3: DUAL SIM DUAL STANDBY FP
返回值:	成功: 0     失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

### 1.16sAPI\_SysSetDualSimType 设置双卡模式

接口:	SC_simcard_err_e sAPI_SysSetDualSimType(UINT8 type)
输入:	type: 双卡模式  0: DUAL SIM DUAL STANDBY 1: DUAL SIM SINGLE STANDBY 3: DUAL SIM DUAL STANDBY FP
输出:	无
返回值:	成功: 0     失败: 1 (参考 SC_simcard_err_e)
NOTE:	无

### 1.17sAPI\_SimRestrictedAccess 访问 SIM 数据库

接口:	SC_simcard_err_e sAPI_SimRestrictedAccess(int cmd, int fileId, int p1, int p2, int p3, char *data, char * pathid, CrsmResponse_st *response)
输入:	cmd: 176 READ BINARY 178 READ RECORD 192 GET RESPONSE 214 UPDATE BINARY 220 UPDATE RECORD 242 STATUS 203 RETRIEVE DATA 219 SET DATA  fileId: SIM 上基本数据文件的标识符  READ BINARY p1: Offset high (0...255) p2: Offset low (0...255) p3: Length (0...255) READ RECORD



p1: Rec. No. (0...255)  
 p2: Mode "02" = next record  
       "03" = previous record  
       "04" = absolute mode/current mode, the record number is given in  
             P1 with P1='00' denoting the current record.  
 p3: Length (0...255)  
 GET RESPONSE  
 p1: "00"  
 p2: "00"  
 p3: Length (0...255)  
 UPDATE BINARY  
 p1: Offset high (0...255)  
 p2: Offset low (0...255)  
 p3: Length (0...255)  
 UPDATE RECORD  
 p2: Rec. No. (0...255)  
 p2: Mode "02" = next record  
       "03" = previous record  
       "04" = absolute mode/current mode, the record number is given in  
             P1 with P1='00' denoting the current record. Length  
 p3: (0...255)  
 STATUS  
 p1: "00"  
 p2: "00"  
 p3: Length (0...255)

data: 当读取时，为 NULL

      当写入时，此为写入数据

pathid: 包含 SIM/UICC 中的基本文件的路径

response: 结果返回结构体

**输出:** 无

**返回值:** 成功: 0      失败: 1 (参考 SC\_simcard\_err\_e)

**NOTE:** 无

## 3 示例

### 3.1 使用 sAPI\_SysGetBindSim 获取接口操作的哪一张 SIM 卡

```
#include "simcom_simcard.h"

unsigned int SimCardApiTestFunc (void)
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOWN;
    UINT8 bindsim= 0;

    ret = sAPI_SysGetBindSim(&bindsim);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
        //.....The action after success
    }
    else
    {
        //.....The action after fail
    }
    return 0;
}
```

### 3.2 使用 sAPI\_SysSetBindSim 选择接口操作的哪一张 SIM 卡

```
#include "simcom_simcard.h"

unsigned int SimCardApiTestFunc (void)
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOWN;
    UINT8 bindsim= 0;//0:SIM1 1:SIM2

    ret = sAPI_SysSetBindSim(bindsim);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
```

```
        //.....The action after success
    }
    else
    {
        //.....The action after fail
    }
    return 0;
}
```

### 3.3 使用 sAPI\_SimcardPinGet 获取 SIM 卡状态

```
#include "simcom_simcard.h"

unsigned int SimCardApiTestFunc (void)
{
    SC_simcard_err_e ret = SC_SIM_RTEURN_UNKNOW;
    UINT8 cpin = 0;
    /* 先选择获取哪张 SIM 卡的状态
    UINT8 bindsim= 0;//0:SIM1 1:SIM2
    ret = sAPI_SysSetBindSim(bindsim);
    */

    ret = sAPI_SimcardPinGet(&cpin);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
        //.....The action after success
    }
    else
    {
        //.....The action after fail
    }
    return 0;
}
```

### 3.4 使用 sAPI\_SimcardPinSet 操作 PIN/PUK 码

1、假如 sAPI\_SimcardPinGet 获取到 SIM 卡状态为 SC\_SIM\_PIN 或 SC\_SIM\_PIN\_PUK

```
#include "simcom_simcard.h"
```

```
unsigned int SimCardApiTestFunc (void)
```

```
{
    SC_simcard_err_e ret;
    char oldpin[20]={"1234"};//PIN 码或 PUK 码
    char newpin[20]={0};
    int new = 0;
    /* 先选择获取哪张 SIM 卡的状态
    UINT8 bindsim= 0;//0:SIM1  1:SIM2
    ret = sAPI_SysSetBindSim(bindsim);
    */

    ret = sAPI_SimcardPinSet(oldpin, newpin, new);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
        //.....The action after success
    }
    else if(ret == SC_SIM_RETURN_FAIL);
    {
        //.....The action after fail
    }
    return 0;
}
```

## 2、如果想修改 PIN 码

```
#include "simcom_simcard.h"
```

```
unsigned int SimCardApiTestFunc (void)
{
    SC_simcard_err_e ret;
    char oldpin[20]={"1234"};
    char newpin[20]={"0000"};
    int new = 1;
    /* 先选择获取哪张 SIM 卡的状态
    UINT8 bindsim= 0;//0:SIM1  1:SIM2
    ret = sAPI_SysSetBindSim(bindsim);
    */

    ret = sAPI_SimcardPinSet(oldpin, newpin, new);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
        //.....The action after success
    }
    else if(ret == SC_SIM_RETURN_FAIL)
    {
        //.....The action after fail
    }
}
```

```
    return 0;  
}
```

### 3.5 使用 sAPI\_SysGetImsi 获取 IMSI

```
#include "simcom_simcard.h"  
  
unsigned int SimCardApiTestFunc (void)  
{  
    SC_simcard_err_e ret = SC_SIM_RTEURN_UNKNOW;  
    char imsi[32] = {0};  
    /* 先选择获取哪张 SIM 卡的状态  
    UINT8 bindsim= 0;//0:SIM1  1:SIM2  
    ret = sAPI_SysSetBindSim(bindsim);  
    */  
  
    ret = sAPI_SysGetImsi(&imsi[0]);  
    if(ret == SC_SIM_RETURN_SUCCESS)  
    {  
        //.....The action after success  
    }  
    else if(ret == SC_SIM_RETURN_FAIL)  
    {  
        //.....The action after fail  
    }  
}
```

### 3.6 使用 sAPI\_SysGetIccid 获取 ICCID

```
#include "simcom_simcard.h"  
  
unsigned int SimCardApiTestFunc (void)  
{  
    SC_simcard_err_e ret = SC_SIM_RTEURN_UNKNOW;  
    char iccid [32] = {0};  
    /* 先选择获取哪张 SIM 卡的状态  
    UINT8 bindsim= 0;//0:SIM1  1:SIM2  
    ret = sAPI_SysSetBindSim(bindsim);  
    */
```

```
ret = sAPI_SysGetIccid (&iccid[0]);
if(ret == SC_SIM_RETURN_SUCCESS)
{
    //.....The action after success
}
else if(ret == SC_SIM_RETURN_FAIL)
{
    //.....The action after fail
}
}
```

### 3.7 使用 sAPI\_SysGetHplmn 获取 HPLMN

```
#include "simcom_simcard.h"

unsigned int SimCardApiTestFunc (void)
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOWN;
    Hplmn_st hplmn= 0;
    /* 先选择获取哪张 SIM 卡的状态
    UINT8 bindsim= 0;//0:SIM1  1:SIM2
    ret = sAPI_SysSetBindSim(bindsim);
    */

    ret = sAPI_SysGetHplmn(&hplmn);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
        //.....The action after success
    }
    else
    {
        //.....The action after fail
    }
    return 0;
}
```

### 3.8 使用 sAPI\_SimcardHotSwapMsg 配置 SIM 卡热插拔功能

#### 1、查询热插拔使能状态（插入/拔出）

```
#include "simcom_simcard.h"
```

```
#include "simcom_os.h"
```

```
unsigned int SimCardApiTestFunc (void)
```

```
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOWN;
    SC_HotSwapCmdType_e opt = SC_HOTSWAP_QUERY_STATE; //插入检测
    sMsgQRef simcard_test_msgQ;
    SIM_MSG_T msg;
    SC_STATUS status;
    /* 先选择获取哪张 SIM 卡的状态
    UINT8 bindsim= 0; //0:SIM1 1:SIM2
    ret = sAPI_SysSetBindSim(bindsim);
    */

    ret = sAPI_SimcardHotSwapMsg(opt,0, simcard_test_msgQ);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
        memset(&msg,0,sizeof(msg));
        status = sAPI_MsgQRecv(simcard_test_msgQ,&msg, \
                               SIMCARD_URC_RECV_TIME_OUT);
        /* msg.arg2 的值就是查询的 SC_HOTSWAP_QUERY_STATE 的状态 */
        //.....The action after success
    }
    else
    {
        //.....The action after fail
    }
    return 0;
}
```

获取 SC\_HOTSWAP\_QUERY\_OUTSTATE 的状态和 SC\_HOTSWAP\_QUERY\_LEVEL 的值与上述方法类似。

## 2、设置 SC\_HOTSWAP\_SET\_SWITCH(打开/关闭热插拔)

```
#include "simcom_simcard.h"
```

```
#include "simcom_os.h"
```

```
unsigned int SimCardApiTestFunc (void)
```

```
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOWN;
    SC_HotSwapCmdType_e opt = SC_HOTSWAP_SET_SWITCH;
    sMsgQRef simcard_test_msgQ;
    SIM_MSG_T msg;
    SC_STATUS status;
    int param=0;
    /* 先选择获取哪张 SIM 卡的状态
    UINT8 bindsim= 0; //0:SIM1 1:SIM2
```

```
ret = sAPI_SysSetBindSim(bindsim);
*/

if((param == 0) || (param == 1))
{
    ret = sAPI_SimcardHotSwapMsg(opt, param, simcard_test_msgQ);
    if(ret == SC_SIM_RETURN_SUCCESS)
    {
        memset(&msg,0,sizeof(msg));
        status = sAPI_MsgQRecv(simcard_test_msgQ,&msg, \
                                SIMCARD_URC_RECV_TIME_OUT);
        //.....The action after success
    }
    else
    {
        //.....The action after fail
    }
}
return 0;
}
```

设置 SC\_HOTSWAP\_SET\_LEVEL 和 SC\_HOTSWAP\_SET\_OUTSWITCH 和上述方法类似。

### 3.9 使用 sAPI\_SimcardSwitchMsg 设置主卡

```
#include "simcom_simcard.h"

unsigned int SimCardApiTestFunc(void)
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOW;
    UINT8 simcardSwitch = 1;//SIM1
    ret = sAPI_SimcardSwitchMsg(simcardSwitch, NULL);
    if(SC_SIM_RETURN_SUCCESS == ret)
    {
        //.....The action after success
    }
    else if(SC_SIM_RETURN_FAIL == ret)
    {
        //.....The action after fail
    }
    return 0;
}
```



### 3.10 使用 sAPI\_SimRestrictedAccess 操作 SIM 数据库

#### 1、READ BINARY

```
#include "simcom_simcard.h"
```

```
unsigned int SimCardApiTestFunc(void)
```

```
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOWN;
    int cmd=0, fileId=0, p1=0, p2=0, p3=0;
    char data[255 * 2 + 1]={0}, pathid[100 * 2 + 1]={0};
    CrsmResponse_st response;

    cmd=176;
    fileId=28539;

    ret = sAPI_SimRestrictedAccess(cmd,fileId,p1,p2,p3,data,pathid,&response);
    if(SC_SIM_RETURN_SUCCESS == ret);
    {
        //.....The action after success
    }
    else if(SC_SIM_RETURN_FAIL == ret);
    {
        //.....The action after fail
    }
    return 0;
}
```

#### 2、UPDATE BINARY

```
#include "simcom_simcard.h"
```

```
unsigned int SimCardApiTestFunc(void)
```

```
{
    SC_simcard_err_e ret = SC_SIM_RETURN_UNKNOWN;
    int cmd=214, fileId=28539, p1=0, p2=0, p3=48;
    char data[255 * 2 + 1]={"64F00064F01064F02064F06064F07064F00254F50064F051"};
    pathid[100 * 2 + 1]={"7FFF6F7B"};
    CrsmResponse_st response;

    ret = sAPI_SimRestrictedAccess(cmd,fileId,p1,p2,p3,data,pathid,&response);
    if(SC_SIM_RETURN_SUCCESS == ret)
    {
        //.....The action after success
    }
    else if(SC_SIM_RETURN_FAIL == ret)
    {

```

```
//.....The action after fail  
}  
return 0;  
}
```

SIMCom  
Confidential

## 4 附录

开机所有的 API 接口均默认操作 SIM1，若想 API 操作 SIM2，需要用接口切换

