ECx00U&EGx00U 系列

QuecOpen (U)SIM API 参考手册

LTE Standard 模块系列

版本：1.0

日期：2021-09-14

状态：受控文件

上海移远通信技术股份有限公司（以下简称“移远通信”）始终以为客户提供最及时、最全面的服务为宗旨。如需任何帮助，请随时联系我司上海总部，联系方式如下：

上海移远通信技术股份有限公司
上海市闵行区田林路 1016 号科技绿洲 3 期（B 区）5 号楼　　邮编：200233
电话：+86 21 5108 6236　　邮箱：info@quectel.com

或联系我司当地办事处，详情请登录：http://www.quectel.com/cn/support/sales.htm。

如需技术支持或反馈我司技术文档中的问题，请随时登陆网址：
http://www.quectel.com/cn/support/technical.htm 或发送邮件至：support@quectel.com。

前言

移远通信提供该文档内容以支持客户的产品设计。客户须按照文档中提供的规范、参数来设计产品。同时，您理解并同意，移远通信提供的参考设计仅作为示例。您同意在设计您目标产品时使用您独立的分析、评估和判断。在使用本文档所指导的任何硬软件或服务之前，请仔细阅读本声明。您在此承认并同意，尽管移远通信采取了商业范围内的合理努力来提供尽可能好的体验，但本文档和其所涉及服务是在“可用”基础上提供给您的。移远通信可在未事先通知的情况下，自行决定随时增加、修改或重述本文档。

使用和披露限制

许可协议

除非移远通信特别授权，否则我司所提供硬软件、材料和文档的接收方须对接收的内容保密，不得将其用于除本项目的实施与开展以外的任何其他目的。

版权声明

移远通信产品和本协议项下的第三方产品可能包含受移远通信或第三方材料、硬软件和文档版权保护的相关资料。除非事先得到书面同意，否则您不得获取、使用、向第三方披露我司所提供的文档和信息，或对此类受版权保护的资料进行复制、转载、抄袭、出版、展示、翻译、分发、合并、修改，或创造其衍生作品。移远通信或第三方对受版权保护的资料拥有专有权，不授予或转让任何专利、版权、商标或服务商标权的许可。为避免歧义，除了正常的非独家、免版税的产品使用许可，任何形式的购买都不可被视为授予许可。对于任何违反保密义务、未经授权使用或以其他非法形式恶意使用所述文档和信息的违法侵权行为，移远通信有权追究法律责任。

商标

除另行规定，本文档中的任何内容均不授予在广告、宣传或其他方面使用移远通信或第三方的任何商标、商号及名称，或其缩略语，或其仿冒品的权利。

第三方权利

您理解本文档可能涉及一个或多个属于第三方的硬软件和文档（“第三方材料”）。您对此类第三方材料的使用应受本文档的所有限制和义务约束。

移远通信针对第三方材料不做任何明示或暗示的保证或陈述，包括但不限于任何暗示或法定的适销性或特定用途的适用性、平静受益权、系统集成、信息准确性以及与许可技术或被许可人使用许可技术相关的不侵犯任何第三方知识产权的保证。本协议中的任何内容都不构成移远通信对任何移远通信产品或任何其他硬软件、设备、工具、信息或产品的开发、增强、修改、分销、营销、销售、提供销售或以其他方式维持生产的陈述或保证。此外，移远通信免除因交易过程、使用或贸易而产生的任何和所有保证。

免责声明

1) 移远通信不承担任何因未能遵守有关操作或设计规范而造成损害的责任。

2) 移远通信不承担因本文档中的任何因不准确、遗漏、或使用本文档中的信息而产生的任何责任。

3) 移远通信尽力确保开发中功能的完整性、准确性、及时性，但不排除上述功能错误或遗漏的可能。除非另有协议规定，否则移远通信对开发中功能的使用不做任何暗示或法定的保证。在适用法律允许的最大范围内，移远通信不对任何因使用开发中功能而遭受的损害承担责任，无论此类损害是否可以预见。

4) 移远通信对第三方网站及第三方资源的信息、内容、广告、商业报价、产品、服务和材料的可访问性、安全性、准确性、可用性、合法性和完整性不承担任何法律责任。

文档历史

修订记录

版本	日期	作者	变更表述
-	2020-10-28	Braden HE	文档创建
1.0	2021-09-14	Braden HE	受控版本

目录

表格索引

1 引言

移远通信 LTE Standard ECx00U 系列和 EGx00U 模块支持 QuecOpen®方案；QuecOpen®是基于 RTOS 的嵌入式开发平台，可简化 IoT 应用的软件设计和开发过程。有关 QuecOpen®的详细信息，请参考 *文档 [1]*。

本文档主要介绍在 QuecOpen®方案下，移远通信 ECx00U 系列和 EGx00U 模块的(U)SIM 卡 API 的应用指导，包括数据结构、相关 API 和使用流程。

1.1. 适用模块

表 1：适用模块

模块系列	模块
ECx00U	EC200U 系列
	EC600U 系列
EGx00U	EG500U-CN
	EG700U-CN

备注

1. EC200U 系列和 EC600U 系列 QuecOpen 模块支持 2 个(U)SIM 接口，但(U)SIM2 为可选功能，且支持单(U)SIM 卡与双(U)SIM 卡的软件不同。支持双(U)SIM 卡的软件版本带有“_DS”后缀，请注意二者在软件版本上的区别。
2. EG500U-CN 和 EG700U-CN QuecOpen 模块仅支持 1 个(U)SIM 接口。

2 (U)SIM 卡相关 API 操作流程

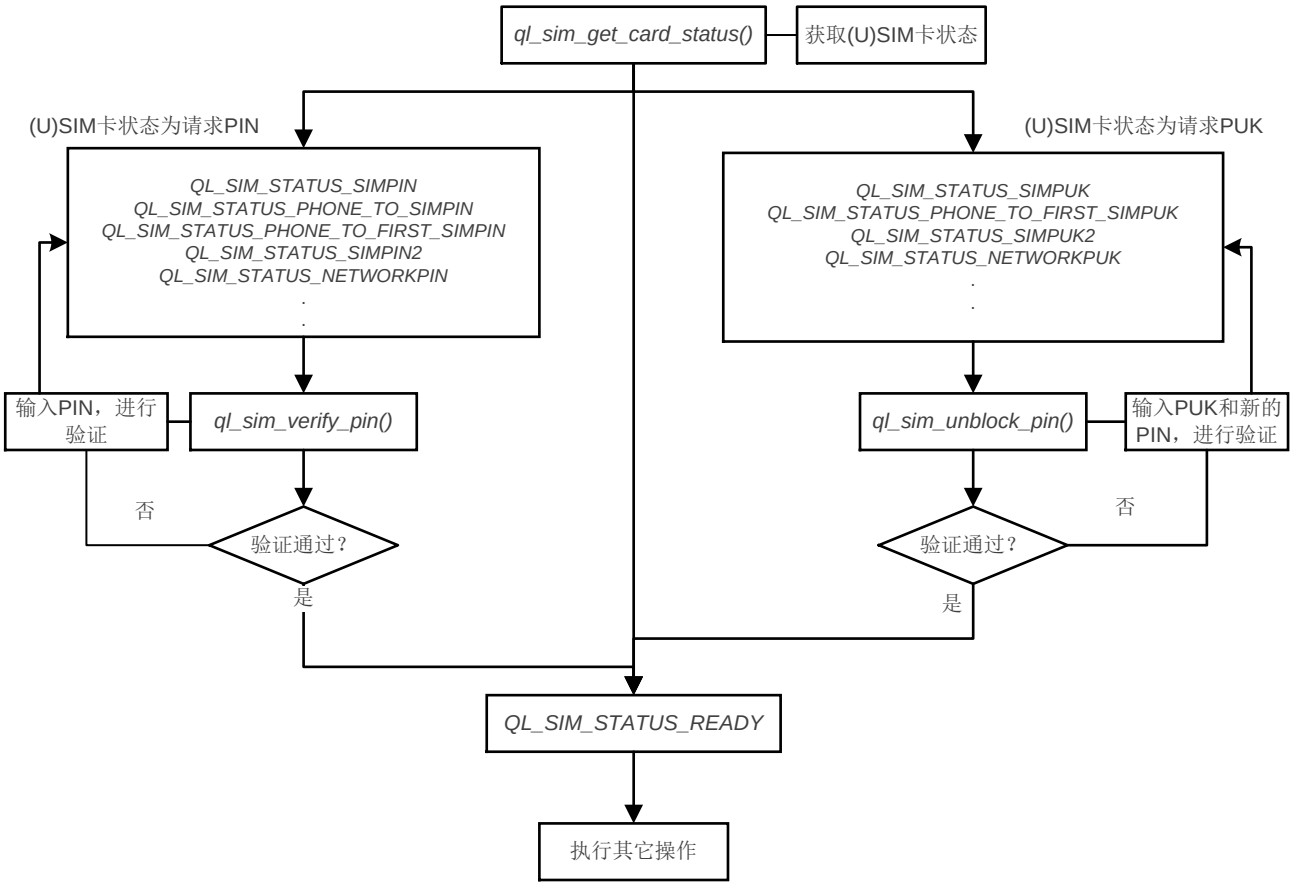

图 1：(U)SIM 卡相关 API 操作流程

上图主要展示获取(U)SIM 卡状态及 PIN 码的验证流程。当(U)SIM 卡状态为请求 PIN/PIN2/PUK/PUK2，需要输入正确的 PIN/PIN2/PUK/PUK2 进行验证。通过验证后，(U)SIM 卡将进入就绪状态从而执行后续操作。

3 (U)SIM 卡相关 API 及数据结构

3.1. 头文件

(U)SIM 卡相关 API 的头文件为 *ql_api_sim.h*，位于 *components\ql-kernel\inc* 目录下。若无特别说明，本文档所提到的头文件均在上述目录下。

3.2. 参考示例

(U)SIM卡相关接口的参考示例位于SDK包的 *components\ql-application\sim\ql_sim_demo.c* 路径下；电话簿相关API接口的参考示例位于SDK包的 *components\ql-application\sim\ql_pbk_demo.c* 路径下。

3.3. 函数概览

表 2：函数概览

函数	说明
ql_sim_get_imsi()	获取(U)SIM 卡的 IMSI
ql_sim_get_iccid()	获取(U)SIM 卡的 ICCID
ql_sim_get_phonenumber()	获取(U)SIM 卡的本机号码
ql_sim_get_card_status()	获取(U)SIM 卡状态信息
ql_sim_enable_pin()	启用(U)SIM 卡 PIN 码验证
ql_sim_disable_pin()	关闭(U)SIM 卡 PIN 码验证
ql_sim_verify_pin()	当(U)SIM 卡状态为请求 PIN 码时输入 PIN 码进行验证
ql_sim_change_pin()	更改(U)SIM 卡的 PIN 码

ql_sim_unblock_pin()	多次错误输入 PIN 码后且(U)SIM 卡状态为请求 PUK 码时，输入 PUK 码和新的 PIN 码进行解锁
ql_sim_read_pbk_item()	获取(U)SIM 卡指定电话簿中的一条或多条电话号码记录
ql_sim_write_pbk_item()	在(U)SIM 卡指定电话簿中写入电话号码记录
ql_sim_set_pbk_encoding()	设置电话簿字符集
ql_sim_get_pbk_encoding()	获取电话簿字符集
ql_pbk_callback_register()	注册电话簿回调函数

3.4. (U)SIM 卡相关 API

3.4.1. ql_sim_get_imsi

该函数用于获取(U)SIM 卡的 IMSI。

- **函数原型**

```
ql_sim_errcode_e ql_sim_get_imsi(uint8_t nSim, char *imsi, size_t imsi_len)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

imsi：
[Out] (U)SIM 卡的 IMSI。

imsi_len：
[In] *imsi* 数组的长度。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.1.1. ql_sim_errcode_e

(U)SIM 卡相关 API 的错误码，表示 API 执行是否成功，若失败则返回错误原因，枚举信息定义如下：

```
typedef enum
{
    QL_SIM_SUCCESS                                  = 0,
    QL_SIM_EXECUTE_ERR                              = 1|QL_SIM_ERRCODE_BASE,
    QL_SIM_MEM_ADDR_NULL_ERR,
    QL_SIM_INVALID_PARAM_ERR,
    QL_SIM_NO_MEMORY_ERR,
    QL_SIM_SEMAPHORE_CREATE_ERR                     = 5|QL_SIM_ERRCODE_BASE,
    QL_SIM_SEMAPHORE_TIMEOUT_ERR,
    QL_SIM_NOT_INSERTED_ERR,
    QL_SIM_CFW_IMSI_GET_REQUEST_ERR,
    QL_SIM_CFW_IMSI_GET_RSP_ERR,
    QL_SIM_CFW_ICCID_GET_REQUEST_ERR                = 10|QL_SIM_ERRCODE_BASE,
    QL_SIM_CFW_PHONE_NUM_GET_REQUEST_ERR,
    QL_SIM_CFW_PHONE_NUM_GET_RSP_NULL_ERR,
    QL_SIM_CFW_STATUS_GET_REQUEST_ERR,
    QL_SIM_CFW_PIN_ENABLE_REQUEST_ERR,
    QL_SIM_CFW_PIN_DISABLE_REQUEST_ERR              = 15|QL_SIM_ERRCODE_BASE,
    QL_SIM_CFW_PIN_VERIFY_REQUEST_ERR,
    QL_SIM_CFW_PIN_CHANGE_REQUEST_ERR,
    QL_SIM_CFW_PIN_UNBLOCK_REQUEST_ERR,
} ql_sim_errcode_e
```

- **参数**

参数	描述
QL_SIM_SUCCESS	函数执行成功
QL_SIM_EXECUTE_ERR	函数执行失败
QL_SIM_MEM_ADDR_NULL_ERR	API 的参数地址为 NULL
QL_SIM_INVALID_PARAM_ERR	API 的参数无效
QL_SIM_NO_MEMORY_ERR	内存申请失败
QL_SIM_SEMAPHORE_CREATE_ERR	创建信号量失败
QL_SIM_SEMAPHORE_TIMEOUT_ERR	等待信号量超时
QL_SIM_NOT_INSERTED_ERR	(U)SIM 卡未插入

QL_SIM_CFW_IMSI_GET_REQUEST_ERR	请求(U)SIM 卡的 IMSI 号被拒
QL_SIM_CFW_IMSI_GET_RSP_ERR	获取(U)SIM 卡的 IMSI 号失败
QL_SIM_CFW_ICCID_GET_REQUEST_ERR	请求(U)SIM 卡的 ICCID 号被拒
QL_SIM_CFW_PHONE_NUM_GET_REQUEST_ERR	请求(U)SIM 卡的本机号码被拒
QL_SIM_CFW_PHONE_NUM_GET_RSP_NULL_ERR	(U)SIM 卡本机号码为空
QL_SIM_CFW_STATUS_GET_REQUEST_ERR	请求(U)SIM 卡状态被拒
QL_SIM_CFW_PIN_ENABLE_REQUEST_ERR	启动(U)SIM 卡 PIN 码验证请求失败
QL_SIM_CFW_PIN_DISABLE_REQUEST_ERR	关闭(U)SIM 卡 PIN 码验证请求失败
QL_SIM_CFW_PIN_VERIFY_REQUEST_ERR	请求 PIN 码验证失败
QL_SIM_CFW_PIN_CHANGE_REQUEST_ERR	更改(U)SIM 卡 PIN 码请求失败
QL_SIM_CFW_PIN_UNBLOCK_REQUEST_ERR	请求 PUK 码解锁失败

3.4.2. ql_sim_get_iccid

该函数用于获取(U)SIM 卡的 ICCID。

- **函数原型**

```
ql_sim_errcode_e ql_sim_get_iccid(uint8_t nSim, char *iccid, size_t iccid_len)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

iccid：
[Out] (U)SIM 卡的 ICCID。

iccid_len：
[In] *iccid* 数组的长度。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.3. ql_sim_get_phonenumber

该函数用于获取(U)SIM 卡的本机号码。

- **函数原型**

```
ql_sim_errcode_e ql_sim_get_phonenumber(uint8_t nSim, char *phonenumber,size_t phonenumber_len)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

phonenumber：
[Out] (U)SIM 卡的本机号码。

phonenumber_len：
[In] *phonenumber* 数组的长度。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.4. ql_sim_get_card_status

该函数用于获取(U)SIM 卡状态信息。

- **函数原型**

```
ql_sim_errcode_e ql_sim_get_card_status(uint8_t nSim, ql_sim_status_e *card_status)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

card_status：
[Out] (U)SIM 卡状态信息。详见***第 3.4.4.1 章***。

- **返回值**

详见*第 3.4.1.1 章*。

3.4.4.1. ql_sim_status_e

(U)SIM 卡状态类型，枚举信息定义如下：

```
typedef enum
{
    QL_SIM_STATUS_READY                 = 0,
    QL_SIM_STATUS_PIN1_READY,
    QL_SIM_STATUS_SIMPIN,
    QL_SIM_STATUS_SIMPUK,
    QL_SIM_STATUS_PHONE_TO_SIMPIN,
    QL_SIM_STATUS_PHONE_TO_FIRST_SIMPIN = 5,
    QL_SIM_STATUS_PHONE_TO_FIRST_SIMPUK,
    QL_SIM_STATUS_SIMPIN2,
    QL_SIM_STATUS_SIMPUK2,
    QL_SIM_STATUS_NETWORKPIN,
    QL_SIM_STATUS_NETWORKPUK            = 10,
    QL_SIM_STATUS_NETWORK_SUBSETPIN,
    QL_SIM_STATUS_NETWORK_SUBSETPUK,
    QL_SIM_STATUS_PROVIDERPIN,
    QL_SIM_STATUS_PROVIDERPUK,
    QL_SIM_STATUS_CORPORATEPIN          = 15,
    QL_SIM_STATUS_CORPORATEPUK,
    QL_SIM_STATUS_NOSIM,
    QL_SIM_STATUS_PIN1BLOCK,
    QL_SIM_STATUS_PIN2BLOCK,
    QL_SIM_STATUS_PIN1_DISABLE          = 20,
    QL_SIM_STATUS_SIM_PRESENT,
    QL_SIM_STATUS_UNKNOW,
}ql_sim_status_e
```

- **参数**

参数	描述
QL_SIM_STATUS_READY	(U)SIM 卡已就绪
QL_SIM_STATUS_PIN1_READY	PIN1 码验证通过
QL_SIM_STATUS_SIMPIN	等待验证(U)SIM 卡的 PIN 码

QL_SIM_STATUS_SIMPUK	等待验证(U)SIM 卡的 PUK 码
QL_SIM_STATUS_PHONE_TO_SIMPIN	(U)SIM 卡个性化检查失败导致(U)SIM 卡被锁定，等待 PCK 码去激活(U)SIM 卡个性化
QL_SIM_STATUS_PHONE_TO_FIRST_SIMPIN	待验证隐藏电话簿条目的密码
QL_SIM_STATUS_PHONE_TO_FIRST_SIMPUK	待验证隐藏电话簿条目的解锁码
QL_SIM_STATUS_SIMPIN2	等待验证(U)SIM 卡的 PIN2
QL_SIM_STATUS_SIMPUK2	等待验证(U)SIM 卡的 PUK2
QL_SIM_STATUS_NETWORKPIN	网络个性化检查失败导致(U)SIM 卡被锁定，等待 NCK 码去激活网络个性化
QL_SIM_STATUS_NETWORKPUK	错误的 NCK 码导致(U)SIM 卡被锁定，需要 MEP 解锁密码
QL_SIM_STATUS_NETWORK_SUBSETPIN	网络子集个性化检查失败导致(U)SIM 卡被锁定，等待 NSCK 码去激活网络子集个性化
QL_SIM_STATUS_NETWORK_SUBSETPUK	错误的 NSCK 码导致(U)SIM 卡被锁定，需要 MEP 解锁密码
QL_SIM_STATUS_PROVIDERPIN	服务提供商个性化检查失败导致(U)SIM 卡被锁定，等待 SPCK 码去激活服务提供商个性化
QL_SIM_STATUS_PROVIDERPUK	错误的 SPCK 码导致(U)SIM 卡被锁定，需要 MEP 解锁密码
QL_SIM_STATUS_CORPORATEPIN	公司个性化检查失败导致(U)SIM 卡被锁定，等待 CCK 码去激活公司个性化
QL_SIM_STATUS_CORPORATEPUK	错误的 CCK 码导致(U)SIM 卡被锁定，需要 MEP 解锁密码
QL_SIM_STATUS_NOSIM	(U)SIM 卡未插入
QL_SIM_STATUS_PIN1BLOCK	不支持
QL_SIM_STATUS_PIN2BLOCK	不支持
QL_SIM_STATUS_PIN1_DISABLE	不支持
QL_SIM_STATUS_SIM_PRESENT	不支持
QL_SIM_STATUS_UNKNOW	未知状态

3.4.5. ql_sim_enable_pin

该函数用于启用(U)SIM 卡 PIN 码验证。

- **函数原型**

```
ql_sim_errcode_e ql_sim_enable_pin(uint8_t nSim, ql_sim_verify_pin_info_s *pt_info)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

pt_info：
[In] PIN 码。详见***第 3.4.5.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.5.1. ql_sim_verify_pin_info_s

PIN 码结构体定义如下：

```
typedef  struct
{
     uint8_t  pin_value[QL_SIM_PIN_LEN_MAX+1];
}ql_sim_verify_pin_info_s
```

- **参数**

类型	参数	描述
uint8_t	*pin_value*	字符串类型。PIN码，最大长度为 QL_SIM_PIN_LEN_MAX（8）字节。

3.4.6. ql_sim_disable_pin

该函数用于关闭(U)SIM 卡 PIN 码验证。

- **函数原型**

```
ql_sim_errcode_e ql_sim_disable_pin(uint8_t nSim, ql_sim_verify_pin_info_s *pt_info)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

pt_info：
[In] PIN 码。详见***第 3.4.5.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.7. ql_sim_verify_pin

该函数用于当(U)SIM 卡状态为请求 PIN 码时输入 PIN 码进行验证。

- **函数原型**

```
ql_sim_errcode_e ql_sim_verify_pin(uint8_t nSim, ql_sim_verify_pin_info_s *pt_info)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

pt_info：
[In] PIN 码。详见***第 3.4.5.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.8. ql_sim_change_pin

该函数用于更改(U)SIM 卡的 PIN 码。

- **函数原型**

```
ql_sim_errcode_e ql_sim_change_pin(uint8_t nSim, ql_sim_change_pin_info_s *pt_info)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

pt_info：
[In] 旧的 PIN 码和新的 PIN 码。详见***第 3.4.8.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.8.1. ql_sim_change_pin_info_s

旧的 PIN 码和新的 PIN 码，结构体定义如下：

```
typedef  struct
{
    uint8_t    old_pin_value[QL_SIM_PIN_LEN_MAX+1];
    uint8_t    new_pin_value[QL_SIM_PIN_LEN_MAX+1];
}ql_sim_change_pin_info_s
```

- **参数**

类型	参数	描述
uint8_t	*old_pin_value*	字符串类型。旧的PIN码，最大长度为QL_SIM_PIN_LEN_MAX（8）字节。
uint8_t	*new_pin_value*	字符串类型。新的PIN码，最大长度为QL_SIM_PIN_LEN_MAX（8）字节。

3.4.9. ql_sim_unblock_pin

该函数用于多次错误输入 PIN 码后且(U)SIM 卡状态为请求 PUK 码时，输入 PUK 码和新的 PIN 码进行解锁。

- **函数原型**

```
ql_sim_errcode_e ql_sim_unblock_pin(uint8_t nSim, ql_sim_unblock_pin_info_s *pt_info)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

pt_info：
[In] PUK 码和新的 PIN 码。详见***第 3.4.9.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.9.1. ql_sim_unblock_pin_info_s

PUK 码和新的 PIN 码，结构体定义如下：

```
typedef  struct
{
    uint8_t   puk_value[QL_SIM_PIN_LEN_MAX+1];
    uint8_t   new_pin_value[QL_SIM_PIN_LEN_MAX+1];
}ql_sim_unblock_pin_info_s
```

- **参数**

类型	参数	描述
uint8_t	*puk_value*	字符串类型。PUK码，最大长度为 QL_SIM_PIN_LEN_MAX（8）字节。
uint8_t	*new_pin_value*	字符串类型。新的PIN码，最大长度为 QL_SIM_PIN_LEN_MAX（8）字节。

3.4.10. ql_sim_read_pbk_item

该函数用于获取(U)SIM 卡指定电话簿中的一条或多条电话号码记录。

- **函数原型**

```
ql_sim_errcode_e ql_sim_read_pbk_item(uint8_t nSim,
                                      ql_sim_pbk_storage_e storage,
                                      int start_index,
                                      int end_index,
                                      uint8_t *username,
                                      uint8_t username_len,
                                      ql_sim_pbk_itemset_info_s *itemset)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

storage：
[In]电话号码在电话簿中的存储位置。详见***第 3.4.10.1 章***。

start_index：
[In] 待读取电话号码记录的起始编号。0 表示不使用编号获取电话号码记录。

end_index：
[In] 待读取电话号码记录的结束编号。

username：
[In] 电话号码中的用户名。当 *start_index* 为 0 时有效。

username_len：
[In] 电话号码中的用户名的长度。当 *start_index* 为 0 时有效。

itemset：
[out] 待读取的电话号码记录列表。详见***第 3.4.10.2 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.10.1. ql_sim_pbk_storage_e

电话号码在电话簿中的存储位置，枚举信息定义如下：

```
typedef enum
{
    QL_SIM_PBK_STORAGE_FD,
    QL_SIM_PBK_STORAGE_LD,
    QL_SIM_PBK_STORAGE_ME,
    QL_SIM_PBK_STORAGE_ON,
    QL_SIM_PBK_STORAGE_SM,
    QL_SIM_PBK_STORAGE_MAX,
} ql_sim_pbk_storage_e
```

- **参数**

参数	描述
QL_SIM_PBK_STORAGE_FD	(U)SIM 卡固定拨号电话簿
QL_SIM_PBK_STORAGE_LD	(U)SIM 卡最新拨号电话簿
QL_SIM_PBK_STORAGE_ME	MT 电话簿
QL_SIM_PBK_STORAGE_ON	(U)SIM 卡（或 MT）本机号码（MSISDNs）列表
QL_SIM_PBK_STORAGE_SM	(U)SIM 卡/UICC 电话簿

3.4.10.2. ql_sim_pbk_itemset_info_s

待读取的电话号码记录列表，结构体定义如下：

```
typedef struct
{
    int item_count;                                          /**< the count of items */
    ql_sim_pbk_item_info_s item[QL_PBK_ITEM_COUNT_MAX]; /**< the list of items */
} ql_sim_pbk_itemset_info_s
```

- **参数**

类型	参数	描述
int	*item_count*	电话号码记录的数量
ql_sim_pbk_item_info_s	*item*	电话号码记录列表，详见***第 3.4.11.1 章***。

3.4.11. ql_sim_write_pbk_item

该函数用于在(U)SIM 卡指定电话簿中写入电话号码记录，可以多次写入，但每次只能写入一条。

- **函数原型**

```
ql_sim_errcode_e ql_sim_write_pbk_item(uint8_t nSim,
                                       ql_sim_pbk_storage_e storage,
                                       ql_sim_pbk_item_info_s *item)
```

- **参数**

nSim：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

storage：
[In] 电话号码在电话簿的存储位置。详见***第 3.4.10.1 章***。

item：
[In] 待写入的电话号码记录。详见***第 3.4.11.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.11.1. ql_sim_pbk_item_info_s

待写入的电话号码记录，结构体定义如下：

```
typedef struct
{
    int index;
    uint8_t username[QL_PBK_USERNAME_LEN_MAX];
    uint8_t phonenum[QL_PBK_PHONENUM_LEN_MAX];
    uint8_t username_len;
} ql_sim_pbk_item_info_s
```

- **参数**

类型	参数	描述
int	*index*	电话号码记录在电话簿中的编号。当写入电话号码记录，*index*为0，表示自动设置电话号码记录编号；*index*为非0时，表示以*index*编号写

		入记录。
uint8_t	*username*	电话号码记录中的用户名。默认长度：32 字符；最大长度：32 字符。采用 *ql_sim_pbk_encoding_e* 枚举中所列出的编码方式，详见***第 3.4.12.1 章***。
uint8_t	*phonenum*	电话号码记录中的电话号码，可能包含“+”的字符串。默认长度：24 字符；最大长度：24 字符。当写入 *phonenum* 为空，则删除 *index* 指定的电话号码记录。
uint8_t	*username_len*	电话号码记录中的用户名长度。

3.4.12. ql_sim_set_pbk_encoding

该函数用于设置电话簿字符集，重启后恢复为 *QL_SIM_PBK_GSM*。

- **函数原型**

```
ql_sim_errcode_e ql_sim_set_pbk_encoding(ql_sim_pbk_encoding_e encoding)
```

- **参数**

encoding：
[In] 电话簿字符集。详见***第 3.4.12.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.12.1. ql_sim_pbk_encoding_e

电话簿字符集，枚举信息定义如下：

```
typedef enum
{
    QL_SIM_PBK_GSM,
    QL_SIM_PBK_HEX,
    QL_SIM_PBK_UCS2,
    QL_SIM_PBK_GBK,
    QL_SIM_PBK_IRA,
    QL_SIM_PBK_ENCODE_MAX,
} ql_sim_pbk_encoding_e
```

- **参数**

参数	描述
QL_SIM_PBK_GSM	GSM 默认字符集
QL_SIM_PBK_HEX	仅由从 00 到 FF 的十六进制字符串组成
QL_SIM_PBK_UCS2	UCS2 字符集
QL_SIM_PBK_GBK	汉字内码扩展规范
QL_SIM_PBK_IRA	国际参考字符集

3.4.13. ql_sim_get_pbk_encoding

该函数用于获取电话簿字符集，默认为 *QL_SIM_PBK_GSM*。

- **函数原型**

```
ql_sim_errcode_e ql_sim_get_pbk_encoding(ql_sim_pbk_encoding_e *encoding)
```

- **参数**

encoding：
[In] 电话簿字符集。详见***第 3.4.12.1 章***。

- **返回值**

详见***第 3.4.1.1 章***。

3.4.14. ql_pbk_callback_register

该函数用于注册电话簿的回调函数。

- **函数原型**

```
void ql_pbk_callback_register(ql_pbk_event_handler_t cb)
```

- **参数**

cb：
[In] 电话簿相关事件回调函数。详见***第 3.4.14.1 章***。

- **返回值**

无

3.4.14.1. ql_pbk_event_handler_t

该函数为电话簿相关事件回调函数。

- **函数原型**

```
typedef void (*ql_pbk_event_handler_t)(uint8_t nSim, int event_id, void *ctx)
```

- **参数**

nSIM：
[In] 所用的(U)SIM 卡。若模块只支持 1 个(U)SIM 接口，此参数可设置为 0。
0 (U)SIM 卡 1
1 (U)SIM 卡 2

event_id：
[In] 电话簿事件 ID。
QL_PBK_INIT_OK_IND 电话簿初始化完成

ctx：
[In] 电话簿事件上下文。

- **返回值**

无

4 附录 参考文档及术语缩写

表 3：参考文档

文档名称
[1] Quectel_ECx00U&EGx00U 系列_QuecOpen_CSDK_快速开发指导

表 4：术语缩写

缩写	英文全称	中文全称
API	Application Programming Interface	应用程序接口
CCK	Corporate Control Key	公司控制密钥
GBK	Chinese Internal Code Specification	汉字内码扩展规范
GSM	Global System for Mobile Communications	全球移动通信系统
HEX	Hexadecimal	十六进制位
ICCID	Integrate circuit card identity	集成电路卡识别码
IMSI	International Mobile Subscriber Identity	国际移动用户识别码
IoT	Internet of Things	物联网
IRA	International Reference Alphabet	7-bit 国际参考字母编码字符集
MEP	Mobile Equipment Personalization	移动设备个性化设置
MT	Mobile Termination	移动终端
NSCK	Network Subset Control Key	网络子集控制密码
NSISDN	Mobile Subscriber ISDN Number	移动台国际用户识别码
PCK	Personalization Control Key	个性化控制密码
PIN	Personal Identification Number	个人识别密码

PUK	Personal Identification Number Unlock Key	个人识别密码解锁码
RTOS	Real-Time Operating System	实时操作系统
SDK	Software Development Kit	软件开发工具包
SIM	Subscriber Identity Module	用户身份识别模块
UCS	Universal Character Set	通用字符集
UICC	Universal Integrated Circuit Card	通用集成电路卡
(U)SIM	(Universal) Subscriber Identity Module	(通用)用户身份识别模块