

ECx00U&EGx00U 系列 设备管理 API 参考手册

LTE Standard 模块系列

版本：1.0

日期：2021-08-31

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软硬件、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。

版权所有 ©上海移远通信技术股份有限公司 2021，保留一切权利。

Copyright © Quectel Wireless Solutions Co., Ltd. 2021.

文档历史

修订记录

版本	日期	作者	变更表述
-	2020-10-20	Jensen FANG/ JoJo YAN	文档创建
1.0	2021-08-31	Jensen FANG/ JoJo YAN	受控版本

目录

文档历史	3
目录	4
表格索引	5
1 引言	6
1.1. 适用模块	6
2 API 调用流程	7
3 设备管理 API 介绍	8
3.1. 头文件	8
3.2. 函数概览	8
3.3. 函数详解	9
3.3.1. ql_dev_get_imei	9
3.3.1.1. ql_errcode_dev_e	9
3.3.2. ql_dev_get_firmware_version	11
3.3.3. ql_dev_get_model	12
3.3.4. ql_dev_get_sn	12
3.3.5. ql_dev_set_modem_fun	13
3.3.6. ql_dev_get_modem_fun	13
3.3.7. ql_dev_get_product_id	14
3.3.8. ql_dev_get_cpu_uid	14
3.3.9. ql_dev_get_temp_value	15
3.3.10. ql_dev_cfg_wdt	15
3.3.11. ql_dev_feed_wdt	15
3.3.12. ql_dev_memory_size_query	16
3.3.12.1. ql_memory_heap_state_t	16
4 示例	17
4.1. 开发示例	17
4.2. 功能调试	18
5 附录 参考文档及术语缩写	19

表格索引

表 1: 适用模块	6
表 2: 函数概览	8
表 3: 参考文档	19
表 4: 术语缩写	19

1 引言

移远通信 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，ECx00U 系列和 EGx00U 模块的设备管理 API、调用流程、相关示例及功能调试。

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

2 API 调用流程

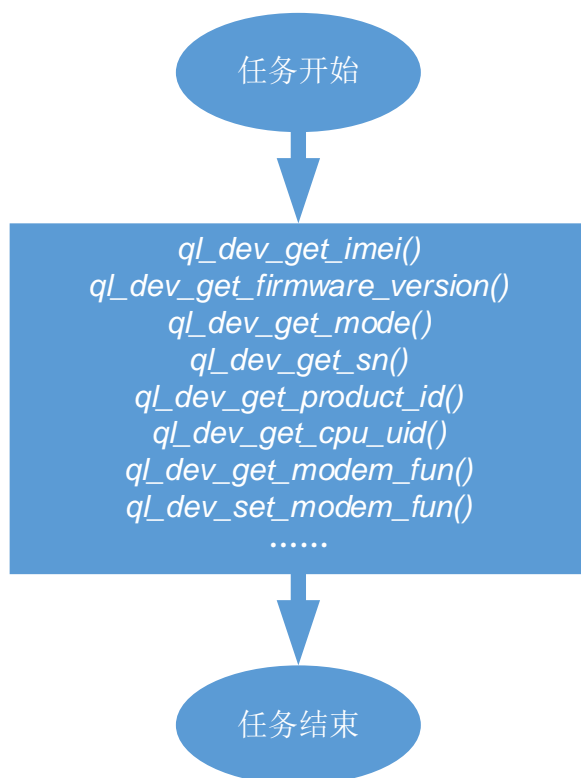


图 1: API 调用流程

3 设备管理 API 介绍

3.1. 头文件

设备管理 API 的头文件为 `ql_api_dev.h`，位于 SDK 包的 `components\ql-kernel\inc` 目录下。若无特别说明，本文档所述头文件均在该目录下。

3.2. 函数概览

表 2：函数概览

函数	说明
<code>ql_dev_get_imei()</code>	获取设备的 IMEI
<code>ql_dev_get_firmware_version()</code>	获取设备的固件版本
<code>ql_dev_get_model()</code>	获取设备型号
<code>ql_dev_get_sn()</code>	获取设备序列号
<code>ql_dev_set_modem_fun()</code>	设置设备 modem 功能
<code>ql_dev_get_modem_fun()</code>	获取设备当前 modem 功能
<code>ql_dev_get_product_id()</code>	获取设备制造商 ID
<code>ql_dev_get_cpu_uid()</code>	获取 CPU 唯一识别码
<code>ql_dev_get_temp_value()</code>	获取芯片温度
<code>ql_dev_cfg_wdt()</code>	配置看门狗（定时器）开关
<code>ql_dev_feed_wdt()</code>	喂系统看门狗（将定时器清零）
<code>ql_dev_memory_size_query()</code>	查询 heap 空间状态信息

3.3. 函数详解

3.3.1. ql_dev_get_imei

该函数用于获取设备的 IMEI。

- 函数原型

```
ql_errcode_dev_e ql_dev_get_imei(char *p_imei, size_t imei_len, uint8_t nSim)
```

- 参数

p_imei:

[Out] 设备的 IMEI。

imei_len:

[In] IMEI 的长度。单位：字节。

nSim:

[In] 当前使用的(U)SIM 卡。

0 (U)SIM 卡 1

1 (U)SIM 卡 2

- 返回值

详见第 3.3.1.1 章。

3.3.1.1. ql_errcode_dev_e

函数执行结果码枚举定义如下：

```
typedef enum
{
    QL_DEV_SUCCESS = QL_SUCCESS,

    QL_DEV_EXECUTE_ERR                                = 1|QL_DEV_ERRCODE_BASE,
    QL_DEV_MEM_ADDR_NULL_ERR,
    QL_DEV_INVALID_PARAM_ERR,
    QL_DEV_BUSY_ERR,
    QL_DEV_SEMAPHORE_CREATE_ERR,
    QL_DEV_SEMAPHORE_TIMEOUT_ERR,

    QL_DEV_CFW_CFUN_GET_ERR                            = 15|QL_DEV_ERRCODE_BASE,
```

```

QL_DEV_CFW_CFUN_SET_CURR_COMM_FLAG_ERR = 18|QL_DEV_ERRCODE_BASE,
QL_DEV_CFW_CFUN_SET_COMM_ERR,
QL_DEV_CFW_CFUN_SET_COMM_RSP_ERR,
QL_DEV_CFW_CFUN_RESET_BUSY                = 25|QL_DEV_ERRCODE_BASE,
QL_DEV_CFW_CFUN_RESET_CFW_CTRL_ERR,
QL_DEV_CFW_CFUN_RESET_CFW_CTRL_RSP_ERR,

QL_DEV_IMEI_GET_ERR                        = 33|QL_DEV_ERRCODE_BASE,

QL_DEV_SN_GET_ERR                          = 36|QL_DEV_ERRCODE_BASE,

QL_DEV_UID_READ_ERR                        = 39|QL_DEV_ERRCODE_BASE,

QL_DEV_TEMP_GET_ERR                        = 50|QL_DEV_ERRCODE_BASE,

QL_DEV_WDT_CFG_ERR                        = 53|QL_DEV_ERRCODE_BASE,

QL_DEV_HEAP_QUERY_ERR                     = 56|QL_DEV_ERRCODE_BASE,

QL_DEV_AUTHCODE_READ_ERR                  = 90|QL_DEV_ERRCODE_BASE,

QL_DEV_READ_WIFI_MAC_ERR                  = 120|QL_DEV_ERRCODE_BASE,

QL_DEV_AUTHCODE_ADDR_NULL_ERR,
} ql_errcode_dev_e

```

● 参数

参数	描述
QL_DEV_SUCCESS	函数执行成功
QL_DEV_EXECUTE_ERR	函数执行失败
QL_DEV_MEM_ADDR_NULL_ERR	指针 NULL 错误
QL_DEV_INVALID_PARAM_ERR	参数错误
QL_DEV_BUSY_ERR	设备繁忙，操作失败
QL_DEV_SEMAPHORE_CREATE_ERR	信号量创建失败
QL_DEV_SEMAPHORE_TIMEOUT_ERR	信号量超时
QL_DEV_CFW_CFUN_GET_ERR	当前功能模式获取失败
QL_DEV_CFW_CFUN_SET_CURR_COMM_FLAG_ERR	当前不支持功能模式设置

QL_DEV_CFW_CFUN_SET_COMM_ERR	功能模式设置失败
QL_DEV_CFW_CFUN_SET_COMM_RSP_ERR	功能模式设置响应异常
QL_DEV_CFW_CFUN_RESET_BUSY	关机忙碌，前一次关机流程进行中
QL_DEV_CFW_CFUN_RESET_CFW_CTRL_ERR	关闭协议栈失败
QL_DEV_CFW_CFUN_RESET_CFW_CTRL_RSP_ERR	关闭协议栈响应异常
QL_DEV_IMEI_GET_ERR	IMEI 获取失败
QL_DEV_SN_GET_ERR	设备序列号获取失败
QL_DEV_UID_READ_ERR	唯一识别码获取失败
QL_DEV_TEMP_GET_ERR	芯片温度获取失败
QL_DEV_WDT_CFG_ERR	看门狗（定时器）开关配置失败
QL_DEV_HEAP_QUERY_ERR	Heap 状态查询失败
QL_DEV_AUTHCODE_READ_ERR	摄像头解码库授权码读取失败
QL_DEV_READ_WIFI_MAC_ERR	读取 Wi-Fi MAC 地址失败
QL_DEV_AUTHCODE_ADDR_NULL_ERR	摄像头解码库授权码地址错误

3.3.2. ql_dev_get_firmware_version

该函数用于获取设备的固件版本。

- 函数原型

```
ql_errcode_dev_e ql_dev_get_firmware_version(char *p_version, size_t version_len)
```

- 参数

p_version:

[Out] 设备的固件版本。

version_len:

[In] 版本号 的长度。单位：字节。

- 返回值

详见第 3.3.1.1 章。

3.3.3. ql_dev_get_model

该函数用于获取设备型号。

- 函数原型

```
ql_errcode_dev_e ql_dev_get_model(char *p_model, size_t model_len);
```

- 参数

p_model:

[Out] 设备型号。

model_len:

[In] 设备型号的长度。单位：字节。

- 返回值

详见第 3.3.1.1 章。

3.3.4. ql_dev_get_sn

该函数用于获取设备序列号。

- 函数原型

```
ql_errcode_dev_e ql_dev_get_sn(char *p_sn, size_t sn_len, uint8_t nSim);
```

- 参数

p_sn:

[Out] 设备序列号。

sn_len:

[In] 设备序列号的长度。单位：字节。

nSim:

[In] 当前使用的(U)SIM 卡。

0 (U)SIM 卡 1

1 (U)SIM 卡 2

- 返回值

详见第 3.3.1.1 章。

3.3.5. ql_dev_set_modem_fun

该函数用于设置设备 modem 功能。

● 函数原型

```
ql_errcode_dev_e ql_dev_set_modem_fun(uint8_t at_dst_fun, uint8_t rst, uint8_t nSim);
```

● 参数

at_dst_fun:

[In] 需要设置的 modem 功能，支持 0/1/4。

- 0 最小功能模式
- 1 全功能模式
- 4 飞行模式

rst:

[In] 设置 modem 功能前是否重启 modem。

- 0 不重启
- 1 重启

nSim:

[In] 当前使用的(U)SIM 卡。

- 0 (U)SIM 卡 1
- 1 (U)SIM 卡 2

● 返回值

详见第 3.3.1.1 章。

3.3.6. ql_dev_get_modem_fun

该函数用于获取设备当前 modem 功能。

● 函数原型

```
ql_errcode_dev_e ql_dev_get_modem_fun(uint8_t *p_function, uint8_t nSim)
```

● 参数

p_function:

[Out] 设备当前 modem 功能，支持 0/1/4。

- 0 最小功能模式
- 1 全功能模式

4 飞行模式

- 返回值

详见第 3.3.1.1 章。

3.3.7. ql_dev_get_product_id

该函数用于获取设备制造商 ID。

- 函数原型

```
ql_errcode_dev_e ql_dev_get_product_id(char *p_product_id, size_t product_id_len)
```

- 参数

p_product_id:

[Out] 设备制造商 ID。

product_id_len:

[In] 设备制造商 ID 的长度。单位：字节。

- 返回值

详见第 3.3.1.1 章。

3.3.8. ql_dev_get_cpu_uid

该函数用于获取 CPU 唯一识别码。

- 函数原型

```
ql_errcode_dev_e ql_dev_get_cpu_uid(unsigned long long *p_chip_id)
```

- 参数

p_chip_id:

[Out] CPU 唯一识别码。

- 返回值

详见第 3.3.1.1 章。

3.3.9. ql_dev_get_temp_value

该函数用于获取芯片温度。

- 函数原型

```
ql_errcode_dev_e ql_dev_get_temp_value(int32_t *ql_temp)
```

- 参数

ql_temp:

[Out] 芯片摄氏温度值。

- 返回值

详见第 3.3.1.1 章。

3.3.10. ql_dev_cfg_wdt

该函数用于配置看门狗（定时器）开关。

- 函数原型

```
ql_errcode_dev_e ql_dev_cfg_wdt(uint8_t opt)
```

- 参数

opt:

[In] 看门狗的开关。

0 关闭

1 打开

- 返回值

详见第 3.3.1.1 章。

3.3.11. ql_dev_feed_wdt

该函数用于喂系统看门狗（将定时器清零）。

- 函数原型

```
ql_errcode_dev_e ql_dev_feed_wdt(void)
```


- 参数

无

- 返回值

详见第 3.3.1.1 章。

3.3.12. ql_dev_memory_size_query

该函数用于查询 heap 空间状态信息。

- 函数原型

```
ql_errcode_dev_e ql_dev_memory_size_query(ql_memory_heap_state_t *ql_heap_state)
```

- 参数

ql_heap_state:

[In] Heap 空间状态信息。详见第 3.3.12.1 章。

- 返回值

详见第 3.3.1.1 章。

3.3.12.1. ql_memory_heap_state_t

Heap 空间状态信息结构体定义如下：

```
typedef struct
{
    uint32_t total_size;
    uint32_t avail_size;
} ql_memory_heap_state_t
```

- 参数

类型	参数	描述
uint32_t	<i>total_size</i>	heap 空间总大小
uint32_t	<i>avail_size</i>	可用 heap 空间大小（可能大于实际值）

4 示例

本章节主要介绍在 APP 侧如何使用上述 API 进行设备管理开发及简单调试。

4.1. 开发示例

ECx00U 系列和 EGx00U QuecOpen 模块的 SDK 代码中提供了设备管理 API 示例文件，即 *ql_dev_demo.c*，位于 *components\ql-application\dev* 路径下。文件中主要包含如何获取模块的 IMEI、设备序列号、设备固件版本等。入口函数为 *ql_dev_app_init()*，如下图所示。

```
void ql_dev_app_init(void)
{
    QIOSStatus.err = QL_OSI_SUCCESS;
    ql_task_t dev_task = NULL;
    err = ql_rtos_task_create(&dev_task, QL_DEV_TASK_STACK_SIZE, QL_DEV_TASK_Prio, "QDEVDEMO", ql_dev_demo_thread, NULL, QL_DEV_TASK_EVENT_CNT);
    if (err != QL_OSI_SUCCESS)
    {
        QL_DEV_LOG("dev demo task created failed");
    }
}
```

如下图所示，上述示例已在 *ql_dev_demo_thread* 线程中默认不启动。需要测试时请取消注释。

```
static void ql_init_demo_thread(void *param)
{
    QL_INIT_LOG("init demo thread enter, param: 0x%x", param);

    /*Caution: If the macro of secure boot and the function are opened, down
    .....the secret key cannot be changed forever*/
    #ifndef QL_APP_FEATURE_SECURE_BOOT
    //ql_dev_enable_secure_boot();
    #endif

    #if 0
    ql_gpio_app_init();
    ql_gpioint_app_init();
    #endif

    #ifndef QL_APP_FEATURE_LEDCFG
    //ql_ledcfg_app_init();
    #endif









    #ifndef QL_APP_FEATURE_AUDIO
    //ql_audio_app_init();
    #endif
    #ifndef QL_APP_FEATURE_HEADSET_DET
    //ql_headset_det_app_init();
    #endif
    #endif

    //ql_dev_app_init();
    //ql_adc_app_init();
    //ql_uart_app_init();
    #endif
```

4.2. 功能调试

ECx00U 系列和 EGx00U QuecOpen 模块可通过移远通信 LTE OPEN EVB 进行设备管理功能调试。

将编译版本烧录到模块中，使用 USB 线连接 LTE OPEN EVB 的 USB 端口和 PC。使用 *cooltools* 抓取 log 后，可通过 USB 的 AP Log 端口查看该示例的调试信息。Log 的抓取方法，请参考文档 [2]。

-  Quectel Modem (COM8)
-  Quectel USB AP Log Port (COM14)
-  Quectel USB AT Port (COM5)
-  Quectel USB CP Log Port (COM11)
-  Quectel USB Diag Port (COM12)
-  Quectel USB MOS Port (COM13)
-  Quectel USB Serial-1 Port (COM9)
-  Quectel USB Serial-2 Port (COM10)

模块开机后自动启动 *ql_dev_app_init()*，通过 log 信息可以查看模块的 IMEI、设备序列号、设备固件版本等信息。

AP Log 端口显示的调试信息如下图所示：

[ql_DEV][ql_dev_demo_thread, 48]	IMEI: 352273017386340
[ql_DEV][ql_dev_demo_thread, 51]	SN: 98612453164038
[ql_DEV][ql_dev_demo_thread, 54]	Product: Quectel
[ql_DEV][ql_dev_demo_thread, 57]	Model: EC200U
[ql_DEV][ql_dev_demo_thread, 60]	FW: EC200UCNAAR02A01M08_OCPU_BETA210512095844
[ql_DEV][ql_dev_demo_thread, 63]	subVer: V01
[ql_DEV][ql_dev_demo_thread, 66]	subVer: V01

5 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
[2] Quectel_ECx00U&EGx00U 系列_QuecOpen_Log_抓取指导

表 4: 术语缩写

缩写	英文全称	中文全称
(U)SIM	(Universal) Subscriber Identity Module	(通用) 用户身份识别模块
AP	Access Point	接入点
API	Application Programming Interface	应用程序编程接口
APP	Application	应用
CPU	Central Processing Unit	中央处理器
EVB	Evaluation Board	评估板
ID	Identifier	标识符
IMEI	International Mobile Equipment Identity	国际移动设备识别码
IoT	Internet of Things	物联网
LTE	Long-Term Evolution	长期演进
MAC	Medium Access Control	媒体访问控制
PC	Personal Computer	个人电脑
RTOS	Real-Time Operating System	实时操作系统
SDK	Software Development Kit	软件开发工具包

UART	Universal Asynchronous Receiver/Transmitter	通用异步收发传输器
USB	Universal Serial Bus	通用串行总线
Wi-Fi	Wireless Fidelity	无线保真（技术）