ECx00U&EGx00U 系列

QuecOpen 矩阵键盘开发指导

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
-	2021-01-11	Neo KONG	初始版本
1.0	2021-09-14	Neo KONG	受控版本

目录

表格索引

图片索引

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，ECx00U 系列和 EGx00U 模块矩阵键盘的开发指导。

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 矩阵键盘相关 API

2.1. 头文件

矩阵键盘相关 API 头文件为 *ql_keypad.h*，位于 QuecOpen SDK 的 *components\ql-kernel\inc* 目录下。ECx00U 系列 QuecOpen 和 EGx00U QuecOpen 模块最多可支持 6 × 5 矩阵键盘，共计 30 个按键。

2.2. API 详解

2.2.1. ql_keypad_init

该函数用于初始化矩阵键盘。

- **函数原型**

```
ql_errcode_keypad_e ql_keypad_init(ql_keyeventcb_t cb, ql_keypad_out_e keyrow[QL_KEYPAD_ROW_LENGTH], ql_keypad_in_e keycol[QL_KEYPAD_COL_LENGTH])(void)
```

- **参数**

cb：

[In] 待注册的回调函数。注册完成后，每次按键（按下或松开），均会上报所设置的键值信息。请参考 ***第 2.2.1.1 章***。

keyrow：

[In] 矩阵键盘输出通道信息配置。未启用的通道可设置为 *QL_KP_OUT_NO_VALID*，其中 QL_KEYPAD_ROW_LENGTH = 6。请参考***第 2.2.1.3 章***。

keycol：

[In] 矩阵键盘输入通道信息配置。未启用的通道可设置为 *QL_KP_IN_NO_VALID*，其中 QL_KEYPAD_COL_LENGTH = 5。请参考***第 2.2.1.4 章***。

- **返回值**

请参考***第 2.2.1.5 章***。

2.2.1.1. ql_keyeventcb_t

待注册回调函数，用于上报矩阵键盘事件。

- **函数原型**

```
typedef void (*ql_keyeventcb_t)(ql_keymatrix_t key);
```

- **参数**

key：
[In] 按键信息。请参考***第 2.2.1.2 章***。

- **返回值**

无

2.2.1.2. ql_keymatrix_t

按键信息结构体定义如下：

```
typedef struct
{
    ql_keymap_e keymap;
    uint32_t keyout;
    uint32_t keyin;
    ql_keystate_e keystate;
}ql_keymatrix_t;
```

- **参数**

类型	参数	描述
ql_keymap_e	*keymap*	按键 ID。范围：1~30。请参考***第 2.2.2.1 章***。
uint32_t	*keyout*	按键对应的输出通道。范围：0~5。
uint32_t	*keyin*	按键对应的输入通道。范围：1~5。
ql_keystate_e	*keystate*	按键状态。 *1* 该按键未按下 *2* 该按键已按下

2.2.1.3. ql_keypad_out_e

矩阵键盘输出通道枚举定义如下：

```
typedef enum
{
    QL_KP_OUT0 = 0,
    QL_KP_OUT1,
    QL_KP_OUT2,
    QL_KP_OUT3,
    QL_KP_OUT4,
    QL_KP_OUT5,
    QL_KP_OUT_NO_VALID
}ql_keypad_out_e;
```

- **参数**

参数	描述
QL_KP_OUT0	键盘输出通道 0
QL_KP_OUT1	键盘输出通道 1
QL_KP_OUT2	键盘输出通道 2
QL_KP_OUT3	键盘输出通道 3
QL_KP_OUT4	键盘输出通道 4
QL_KP_OUT5	键盘输出通道 5
QL_KP_OUT_NO_VALID	键盘无效输出通道

2.2.1.4. ql_keypad_in_e

矩阵键盘输入通道枚举定义如下：

```
typedef enum
{
    QL_KP_IN1=1,
    QL_KP_IN2,
    QL_KP_IN3,
    QL_KP_IN4,
    QL_KP_IN5,
    QL_KP_IN_NO_VALID
}ql_keypad_in_e;
```

- **参数**

参数	描述
QL_KP_IN1	键盘输入通道 1
QL_KP_IN2	键盘输入通道 2
QL_KP_IN3	键盘输入通道 3
QL_KP_IN4	键盘输入通道 4
QL_KP_IN5	键盘输入通道 5
QL_KP_IN_NO_VALID	键盘无效输入通道

2.2.1.5. ql_errcode_keypad_e

矩阵键盘错误码枚举定义如下：

```
typedef enum
{
    QL_KEYPAD_SUCCESS = QL_SUCCESS,
    QL_KEYPAD_INVALID_PARAM_ERR = 1|QL_KEYPAD_ERRCODE_BASE
}ql_errcode_keypad_e;
```

- **参数**

参数	描述
QL_KEYPAD_SUCCESS	函数执行成功
QL_KEYPAD_INVALID_PARAM_ERR	无效参数

2.2.2. ql_keypad_state

该函数用于获取指定按键的状态。

- **函数原型**

```
ql_errcode_keypad_e ql_keypad_state(uint32_t *pressed, ql_keymap_e id)
```

- **参数**

pressed：
[In] 按键状态。
0 按键处于松开状态
1 按键处于按下状态

id：
[In] 按键 ID。请参考***第 2.2.2.1 章***。

- **返回值**

请参考***第 2.2.1.5 章***。

2.2.2.1. ql_keymap_e

按键枚举定义如下：

```
typedef enum
{
    QL_KEY_MAP_1 = 1,
    QL_KEY_MAP_2,
    QL_KEY_MAP_3,
    QL_KEY_MAP_4,
    QL_KEY_MAP_5,
    QL_KEY_MAP_6,
    QL_KEY_MAP_7,
    QL_KEY_MAP_8,
    QL_KEY_MAP_9,
    QL_KEY_MAP_10,
    QL_KEY_MAP_11,
    QL_KEY_MAP_12,
    QL_KEY_MAP_13,
    QL_KEY_MAP_14,
    QL_KEY_MAP_15,
    QL_KEY_MAP_16,
    QL_KEY_MAP_17,
    QL_KEY_MAP_18,
    QL_KEY_MAP_19,
    QL_KEY_MAP_20,
    QL_KEY_MAP_21,
    QL_KEY_MAP_22,
    QL_KEY_MAP_23,
    QL_KEY_MAP_24,
```

```
    QL_KEY_MAP_25,
    QL_KEY_MAP_26,
    QL_KEY_MAP_27,
    QL_KEY_MAP_28,
    QL_KEY_MAP_29,
    QL_KEY_MAP_30,
    QL_KEY_MAP_MAX_COUNT   ///< total count
} ql_keymap_e;
```

- **参数**

参数	描述
QL_KEY_MAP_1	按键 1
QL_KEY_MAP_2	按键 2
QL_KEY_MAP_3	按键 3
QL_KEY_MAP_4	按键 4
QL_KEY_MAP_5	按键 5
QL_KEY_MAP_6	按键 6
QL_KEY_MAP_7	按键 7
QL_KEY_MAP_8	按键 8
QL_KEY_MAP_9	按键 9
QL_KEY_MAP_10	按键 10
QL_KEY_MAP_11	按键 11
QL_KEY_MAP_12	按键 12
QL_KEY_MAP_13	按键 13
QL_KEY_MAP_14	按键 14
QL_KEY_MAP_15	按键 15
QL_KEY_MAP_16	按键 16
QL_KEY_MAP_17	按键 17
QL_KEY_MAP_18	按键 18
QL_KEY_MAP_19	按键 19

QL_KEY_MAP_20	按键 20
QL_KEY_MAP_21	按键 21
QL_KEY_MAP_22	按键 22
QL_KEY_MAP_23	按键 23
QL_KEY_MAP_24	按键 24
QL_KEY_MAP_25	按键 25
QL_KEY_MAP_26	按键 26
QL_KEY_MAP_27	按键 27
QL_KEY_MAP_28	按键 28
QL_KEY_MAP_29	按键 29
QL_KEY_MAP_30	按键 30

3 示例

3.1. 开发示例

3.1.1. 示例说明

QuecOpen SDK 中提供了矩阵键盘相关的示例，即 *components\ql-application\peripheral\keypad_demo.c* 文件。该文件中主要包含矩阵键盘功能的初始化和按键状态信息的获取。入口函数为 *ql_keypad_app_init()*，如下图所示。

```
void ql_keypad_app_init()
{
    QlOSStatus err = QL_OSI_SUCCESS;
    ql_task_t keypad_task = NULL;

    err = ql_rtos_task_create(&keypad_task, 1024, APP_PRIORITY_NORMAL, "ql_keypaddemo", ql_keypad_demo_thread, NULL, 1);
    if( err != QL_OSI_SUCCESS )
    {
        QL_KEYPADDEMO_LOG("keypad demo task created failed");
    }
}
```

该示例文件中首先配置矩阵键盘输入和输出通道，然后调用 *ql_keypad_init()*初始化矩阵键盘、传递输入输出通道信息和回调函数，当指定按键按下或松开时，系统调用回调函数 *ql_keypad_callback()*输出按键信息，包含按键位置、按键对应的输入输出通道和按键状态。

```
void ql_keypad_demo_thread(void *param)
{
    QlOSStatus err = 0;
    uint32_t *pressed;
    ql_keypad_out_e row[QL_KEYPAD_ROW_LENGTH] = {QL_KP_OUT0, QL_KP_OUT1, QL_KP_OUT2, QL_KP_OUT3, QL_KP_OUT4, QL_KP_OUT5};
    ql_keypad_in_e  col[QL_KEYPAD_COL_LENGTH] = {QL_KP_IN1,QL_KP_IN2, QL_KP_IN3, QL_KP_IN_NO_VALID, QL_KP_IN_NO_VALID};
    pressed = (uint32_t *)malloc(sizeof(uint32_t));

    err = ql_keypad_init(ql_keypad_callback, row, col);
    if(err == QL_KEYPAD_SUCCESS)
    {
        while(1)
        {
            printf("keypad_test \r\n");
            err = ql_keypad_state(pressed, QL_KEY_MAP_15);
            if(err == QL_KEYPAD_SUCCESS)
            {
                printf("keymap15 pressed: %d", *pressed);
            }
            else
            {
                return;
            }
            ql_rtos_task_sleep_s(3);
        }
    }
    free(pressed);
    err = ql_rtos_task_delete(NULL);
    if(err != QL_OSI_SUCCESS)
    {
        QL_KEYPADDEMO_LOG("task deleted failed");
    }

} « end ql_keypad_demo_thread »
```

3.1.2. 示例自启动

上述示例在 *ql_init_demo_thread()*线程中默认不启动，如下图所示。如需自启动该示例，只需取消 *ql_keypad_app_init()*前的注释即可。

```
#ifdef QL_APP_FEATURE_CAMERA
    //ql_camera_app_init();
#endif

#ifdef QL_APP_FEATURE_SPI_FLASH
    //ql_spi_flash_demo_init();
#endif

#if defined (QL_APP_FEATURE_WIFISCAN)
    //ql_wifiscan_app_init();
#endif

#ifdef QL_APP_FEATURE_FOTA_HTTP
»   »   //ql_fota_http_app_init();
#endif

#ifdef QL_APP_FEATURE_DECODER
    //ql_decoder_app_init();
#endif

#ifdef QL_APP_FEATURE_KEYPAD
    //ql_keypad_app_init();
#endif
```

3.2. 调试

3.2.1. 调试准备

模块可通过移远通信 LTE OPEN EVB 板调试矩阵键盘功能。

编译固件版本后烧录到模块中，使用 USB 线连接 LTE OPEN EVB 板的 USB 端口和 PC。USB 的 AP Log 端口主要用于显示系统调试信息，通过 *coolwatcher_usb.exe* 的 *Trace Tool* 打印的 Log 信息可以查看该示例的相关信息。

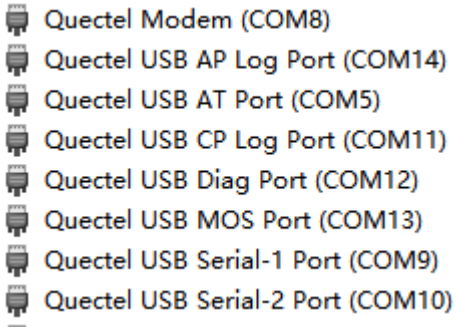

图 1：COM 设备图

3.2.2. 功能调试

设置开机自动启动矩阵键盘示例，然后按下或松开按键，可以通过 Log 信息看到打印出的按键信息。

AP Log 端口调试信息如下图所示：

Index	Received	Tick	Level	
183	19:28:38.925	20054	QOPN/I	keymap17 pressed: 0
199	19:28:41.925	3670	QOPN/I	keymap17 pressed: 0
209	19:28:43.543	30207	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:29 keyout:5 keyin:3 pressd:1
210	19:28:43.878	35696	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:29 keyout:5 keyin:3 pressd:2
218	19:28:50.925	20054	QOPN/I	keymap17 pressed: 0
226	19:28:51.759	33777	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:27 keyout:5 keyin:1 pressd:1
229	19:28:51.883	35787	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:27 keyout:5 keyin:1 pressd:2
279	19:28:53.925	3670	QOPN/I	keymap17 pressed: 0
281	19:28:55.255	25512	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:9 keyout:1 keyin:3 pressd:1
282	19:28:55.380	27582	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:9 keyout:1 keyin:3 pressd:2
286	19:28:56.926	52822	QOPN/I	keymap17 pressed: 0
289	19:28:59.925	36438	QOPN/I	keymap17 pressed: 0
304	19:29:01.561	63291	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:17 keyout:3 keyin:1 pressd:1
328	19:29:11.927	36438	QOPN/I	keymap17 pressed: 1
340	19:29:14.926	20054	QOPN/I	keymap17 pressed: 1
341	19:29:15.097	22937	QOPN/I	[ql_KEYPADDEMO][ql_keypad_callback, 49] keymap:17 keyout:3 keyin:1 pressd:2
347	19:29:17.932	3670	QOPN/I	keymap17 pressed: 0
351	19:29:26.924	20054	QOPN/I	keymap17 pressed: 0

图 2：矩阵键盘 Log 信息

4 附录 参考文档及术语缩写

表 2：参考文档

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

表 3：术语缩写

缩写	英文全称	中文全称
AP	Application Processor	应用处理器
API	Application Programming Interface	应用程序编程接口
APP	Application	应用程序
COM	Communication	通信
EVB	Evaluation Board	评估板
IoT	Internet of Things	物联网
PC	Personal Computer	个人电脑
RTOS	Real-time Operation System	实时操作系统
SDK	Software Development Kit	软件开发工具包
USB	Universal Serial Bus	通用串行总线