ECx00U&EGx00U 系列
QuecOpen 解码功能开发指导

LTE Standard 模块系列

版本：1.0

日期：2021-09-10

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
-	2021-01-23	Chaos HUANG	文档创建
1.0	2021-09-10	Chaos HUANG	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 QuecOpen®方案下， ECx00U 系列和 EGx00U 模块解码功能相关 API 的使用、参考示例和解码功能调试。

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 解码功能相关 API

2.1. 头文件

解码功能 API 头文件为 *ql_api_decoder.h*，位于 SDK 包 *components\ql-kernel\inc* 目录下。

2.2. 函数概览

表 2：函数概览

函数	说明
ql_qr_decoder_init()	初始化解码功能。
ql_qr_image_decoder()	解码输入的条形码或者二维码图像。
ql_qr_get_decoder_result()	获取解码的结果。
ql_destroy_decoder()	释放解码功能。
ql_get_decoder_version()	获取解码库版本信息。

2.3. API 详解

2.3.1. ql_qr_decoder_init

该函数用于初始化解码功能。

- **函数原型**

```
ql_errcode_decoder_e ql_qr_decoder_init(void)
```

- **参数**

无

- **返回值**

函数执行结果。详情请参考***第 3.1 章***。

2.3.2. ql_qr_image_decoder

该函数用于解码输入的条形码或者二维码图像。

- **函数原型**

```
ql_errcode_decoder_e ql_qr_image_decoder(uint16_t *img_buffer, uint32_t width, uint32_t height)
```

- **参数**

**img_buffer*：
[In] 输入的图像数据。

width:
[In] 输入图像数据的宽度。

height:
[In] 输入图像数据的高度。

- **返回值**

函数执行结果。详情请参考***第 3.1 章***。

2.3.3. ql_qr_get_decoder_result

该函数用于获取解码的结果。

- **函数原型**

```
ql_errcode_decoder_e ql_qr_get_decoder_result (ql_decoder_type_e* type, unsigned char * result);
```

- **参数**

**type*：
[Out] 解码类型。详情参见***第 2.3.3.1 章***。

*result：
[Out] 指向存放扫码结果的 buffer。

- **返回值**

函数执行结果。详情请参考***第 3.1 章***。

2.3.3.1. ql_decoder_type_e

解码类型枚举定义如下：

```
typedef enum
{
    QL_DECODER_TYPE_CODE39 = 0,
    QL_DECODER_TYPE_CODE128,
    QL_DECODER_TYPE_QR_CODE,
    QL_DECODER_TYPE_NONE = 0xff,
}ql_decoder_type_e;
```

- **参数**

参数	描述
QL_DECODER_TYPE_CODE39	条形码 CODE-39 码制
QL_DECODER_TYPE_CODE128	条形码 CODE-128 码制
QL_DECODER_TYPE_QR_CODE	二维码
QL_DECODER_TYPE_NONE	无识别结果

2.3.4. ql_destroy_decoder

该函数用于释放解码功能，同时释放解码申请的内存空间。

- **函数原型**

```
ql_errcode_decoder_e ql_destroy_decoder(void)
```

- **参数**

无

- **返回值**

函数执行结果。详情请参考***第 3.1 章***。

2.3.5. ql_get_decoder_version

该函数用于获取当前解码库的版本信息。

- **函数原型**

```
ql_errcode_decoder_e ql_get_decoder_version(unsigned char* version);
```

- **参数**

version：
[Out] 当前解码库的版本信息

- **返回值**

函数执行结果。详情请参考***第 3.1 章***。

3 错误码

3.1. ql_errcode_decoder_e

函数执行结果的枚举定义如下：

```
typedef enum
{
    QL_DECODER_SUCCESS = QL_SUCCESS,
    QL_DECODER_INIT_ERR                    = 1|QL_COMPONENT_BSP_DECODER,
    QL_DECODER_ERR,
    QL_DECODER_GET_RESULT_ERR,
    QL_DECODER_GET_RESULT_LENGTH_ERR,
    QL_DECODER_DESTROY_ERR,
}ql_errcode_decoder_e;
```

- **参数**

参数	描述
QL_DECODER_SUCCESS	函数执行成功。
QL_DECODER_INIT_ERR	解码功能初始化失败。
QL_DECODER_ERR	解码失败。
QL_DECODER_GET_RESULT_ERR	获取解码结果失败。
QL_DECODER_GET_RESULT_LENGTH_ERR	获取解码结果长度失败。
QL_DECODER_DESTROY_ERR	解码功能释放失败。

4 解码功能开发示例及调试

4.1. APP 解码功能开发示例

4.1.1. 解码静态库使用

模块的解码功能会作为静态库被使用，因此在使用的时候需要开发者自行链接静态库。静态库存放的地址为 *components\newlib\armca5\libql_api_decoder.a*，只需要在需要使用该库的地方链接静态库即可，可参考 APP SDK 中根目录下的 *CmakeLists.txt*，如下图所示：

```
if(CONFIG_APPIMG_LOAD_FLASH)
    set(target ${QL_APP_BUILD_VER})
    add_appimg_flash_ql_example(${target} ql_init.c)
    target_link_libraries(${target} PRIVATE ql_app_nw ql_app_peripheral ql_app_osi ql_app_dev ql_app_sim ql_app_power)
    if(QL_APP_FEATURE_DECODER)
        add_library(ql_decoder_api STATIC IMPORTED)
        set_target_properties(ql_decoder_api PROPERTIES IMPORTED_LOCATION ${SOURCE_TOP_DIR}/components/newlib/armca5/libql_api_decoder.a)
        target_link_libraries(${target} PRIVATE ql_app_decoder ql_decoder_api ${libm_file_name})
    endif()
    if(QL_APP_FEATURE_FTP)
        target_link_libraries(${target} PRIVATE ql_app_ftp)
    endif()
    if(QL_APP_FEATURE_HTTP)
        target_link_libraries(${target} PRIVATE ql_app_http)
    endif()
```

使用解码库的APP　　链接静态库

图 1：链接静态库实例

4.1.2. 解码功能使用示例

QuecOpen SDK 代码中提供了解码功能使用示例，即 *decoder_demo.c* 文件，该文件路径为 *components\ql-application\decoder*。

示例文件中主要包含解码功能初始化、通过摄像头进行扫码、解码和获取解码结果等内容。入口函数为 *ql_decoder_app_init()*，如下图所示：

```
void ql_decoder_app_init(void)
{
    QlDECODERStatus err = QL_OSI_SUCCESS;
    ql_task_t decoder_task = NULL;

    err = ql_rtos_task_create(&decoder_task, QL_DECODER_TASK_STACK_SIZE, QL_DECODER_TASK_PRIO, "decoder DEMO", ql_decoder_demo_thread, NULL, QL_DECODER_TASK_EVENT_CNT);
    if (err != QL_OSI_SUCCESS)
    {
        QL_DECODER_LOG("DECODER demo task created failed");
    }
}
```

图 2：解码功能示例的入口函数

decoder_demo.c 文件先进行解码功能的初始化，然后打开摄像头和 LCD，并打开预览功能，之后开始扫码、解码和获取相关结果。

4.1.3. 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，取消*ql_decoder_app_init()*前的注释即可。

```
static void ql_init_demo_thread(void *param)
{
    QL_INIT_LOG("init demo thread enter, param 0x%x", param);
#if 0
    ql_gpio_app_init();
    ql_gpioint_app_init();
#endif

#ifdef QL_APP_FEATURE_DECODER
    //ql_decoder_app_init();
#endif

#ifdef QL_APP_FEATURE_LEDCFG
    ql_ledcfg_app_init();
#endif

#ifdef QL_APP_FEATURE_AUDIO
    //ql_audio_app_init();
#endif

#ifdef QL_APP_FEATURE_LCD
    //ql_lcd_app_init();
#ifdef QL_APP_FEATURE_LVGL
    //ql_lvgl_app_init();
#endif
```

图 3：设置解码功能示例自启动

4.1.4. 示例使用注意事项

使用该示例时需要把 *ql_ledcfg_app_init()*注释掉，因为解码功能需要使用到摄像头，而摄像头需要的部分功能引脚和 *ql_ledcfg_app_init()*中使用到的引脚重复，从而产生冲突使摄像头无法正常工作。此外，在使用该示例时，请勿同时打开 *ql_camera_app_init()*或 *ql_i2c_demo_init()*，因为这三个示例都使用了摄像头作为演示，若同时打开互相会产生冲突，引起模块异常。

4.2. 解码功能调试

4.2.1. 调试步骤

模块可使用移远通信 LTE OPEN EVB 进行摄像头功能调试，调试步骤如下：

1. 编译软件版本并烧录到模块中。
2. 使用 USB 线连接移远通信 LTE OPEN EVB 的 USB 端口和 PC。PC 的设备管理器 COM 设备中，USB 的 AP Log 端口主要用于显示系统调试信息，如下图所示。
3. 通过 *coolwatcher_debughost.exe* 的 *Trace Tool* 打印的 Log 信息查看该示例的相关信息，通过 Log 信息可以判断解码功能是否运行成功。

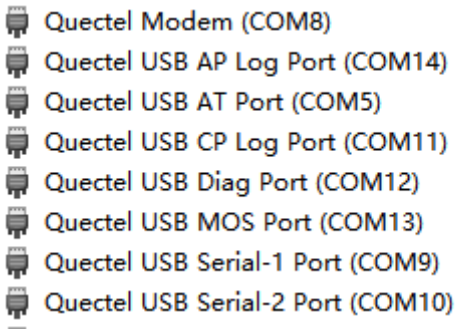

图 4：设备管理器 COM 设备

备注

有关 coolwatcher 工具的获取和使用，请联系移远通信技术支持。

4.2.2. 调试 Log 信息

设置开机后自启动解码功能示例，通过 Log 信息可以判断解码功能是否运行成功。如果解码成功，就能识别到二维码或条形码。否则，无法识别到二维码或条形码。

AP Log 端口调试信息如*图 5* 和*图 6* 所示：

```
[ql_api_qr_code][ql_qr_decoder_init, 131] Qr code Initial success
[ql_api_qr_code][ql_qr_image_decoder, 161] decoder into preview data
[ql_api_qr_code][ql_qr_image_decoder, 179] *** Decoding end***
[ql_api_qr_code][ql_qr_image_decoder, 182] Imgbuffer Decoder failed
[ql_DECODERDEMO][ql_decoder_demo_thread, 98] QR code decode failed
[ql_api_qr_code][ql_qr_image_decoder, 161] decoder into preview data
[ql_api_qr_code][ql_qr_image_decoder, 179] *** Decoding end***
[ql_api_qr_code][ql_qr_image_decoder, 182] Imgbuffer Decoder failed
[ql_DECODERDEMO][ql_decoder_demo_thread, 98] QR code decode failed
```

图 5：二维码或条形码识别失败 Log

```
[ql_api_qr_code][ql_qr_image_decoder, 161] decoder into preview data
[ql_api_qr_code][ql_qr_image_decoder, 179] *** Decoding end***
[ql_api_qr_code][ql_qr_image_decoder, 206] imgbuffer Decoder success
[ql_api_qr_code][ql_qr_get_decoder_result, 247] Get Result Length success
[ql_DECODERDEMO][ql_decoder_demo_thread, 90] Get QR code decode result **345615615131316513513513035313513513513**
[ql_api_qr_code][ql_qr_image_decoder, 161] decoder into preview data
[ql_api_qr_code][ql_qr_image_decoder, 179] *** Decoding end***
[ql_api_qr_code][ql_qr_image_decoder, 206] imgbuffer Decoder success
[ql_api_qr_code][ql_qr_get_decoder_result, 247] Get Result Length success
[ql_DECODERDEMO][ql_decoder_demo_thread, 90] Get QR code decode result **345615615131316513513513035313513513513**
```

图 6：二维码或条形码识别成功 Log

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

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
APP	Application	应用程序
COM	Communication	通信
EVB	Evaluation Board	开发板
LCD	Liquid Crystal Display	液晶显示器
PC	Personal Computer	个人电脑
SDK	Software Development Kit	软件开发工具包
USB	Universal Serial Bus	通用串行总线