

ECx00U&EGx00U 系列

QuecOpen RTC API 参考手册

LTE Standard 模块系列

版本：1.0

日期：2021-08-03

状态：受控文件



上海移远通信技术股份有限公司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 51086236 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，可随时登陆如下网址：
<http://www.quectel.com/cn/support/technical.htm> 或发送邮件至：support@quectel.com。

前言

上海移远通信技术股份有限公司提供该文档内容用以支持其客户的产品设计。客户须按照文档中提供的规范、参数来设计其产品。因未能遵守有关操作或设计规范而造成的损害，上海移远通信技术股份有限公司不承担任何责任。在未声明前，上海移远通信技术股份有限公司有权对该文档进行更新。

免责声明

上海移远通信技术股份有限公司尽力确保开发中功能的完整性、准确性、及时性或效用，但不排除上述功能错误或遗漏的可能。除非其他有效协议另有规定，否则上海移远通信技术股份有限公司对开发中功能的使用不做任何暗示或明示的保证。在适用法律允许的最大范围内，上海移远通信技术股份有限公司不对任何因使用开发中功能而遭受的损失或损害承担责任，无论此类损失或损害是否可以预见。

保密义务

除非上海移远通信技术股份有限公司特别授权，否则我司所提供文档和信息的接收方须对接收的文档和信息保密，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。未经上海移远通信技术股份有限公司书面同意，不得获取、使用或向第三方泄露我司所提供的文档和信息。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，上海移远通信技术股份有限公司有权追究法律责任。

版权申明

本文档版权属于上海移远通信技术股份有限公司，任何人未经我司允许而复制转载该文档将承担法律责任。

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
-	2021-07-19	Neo KONG	文档创建
1.0	2021-08-03	Neo KONG	受控版本

目录

文档历史	2
目录	3
表格索引	4
图片索引	5
1 引言	6
1.1. 适用模块	6
2 RTC 相关 API.....	7
2.1. 头文件.....	7
2.2. 函数概览.....	7
2.3. 函数详解	8
2.3.1. ql_rtc_set_time	8
2.3.1.1. ql_errcode_rtc_e	8
2.3.1.2. ql_rtc_time_t.....	9
2.3.2. ql_rtc_get_time.....	9
2.3.3. ql_rtc_print_time.....	10
2.3.4. ql_rtc_set_alarm.....	10
2.3.5. ql_rtc_get_alarm	11
2.3.6. ql_rtc_enable_alarm.....	11
2.3.7. ql_rtc_register_cb.....	11
2.3.7.1. ql_rtc_cb.....	12
2.3.8. ql_rtc_get_localtime	12
3 示例	13
3.1. 开发示例	13
3.2. 功能调试	15
4 附录 参考文档及术语缩写	17

表格索引

表 1: 适用模块	6
表 2: 函数概览	7
表 3: 参考文档	17
表 4: 术语缩写	17

图片索引

图 1: ql_rtc_app_init()	13
图 2: ql_rtc_demo_thread().....	14
图 3: ql_init_demo_thread()	15
图 4: COM 设备.....	16
图 5: RTC log 信息.....	16

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 RTC 相关 API、RTC 开发示例及相关调试。

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 RTC 相关 API

2.1. 头文件

RTC API 的头文件为 `ql_api_rtc.h`，位于 SDK 包的 `\components\ql-kernel\inc` 目录下。若无特别说明，本文档所涉头文件均位于该目录下。

2.2. 函数概览

表 2：函数概览

函数	说明
<code>ql_rtc_set_time()</code>	设置 RTC 时间
<code>ql_rtc_get_time()</code>	获取当前 RTC 时间
<code>ql_rtc_print_time()</code>	打印 RTC 时间
<code>ql_rtc_set_alarm()</code>	设置警报时间
<code>ql_rtc_get_alarm()</code>	获取警报时间
<code>ql_rtc_enable_alarm()</code>	启用警报功能
<code>ql_rtc_register_cb()</code>	注册警报回调函数
<code>ql_rtc_get_localtime()</code>	获取本地时间（含时区）

2.3. 函数详解

2.3.1. ql_rtc_set_time

该函数用于设置 RTC 时间。

- 函数原型

```
ql_errcode_rtc_e ql_rtc_set_time(ql_rtc_time_t *tm)
```

- 参数

tm:

[In] RTC 时间结构体指针。详情请参考第 2.3.1.2 章。

- 返回值

详情请参考第 2.3.1.1 章。

2.3.1.1. ql_errcode_rtc_e

RTC 功能组件的结果码定义如下：

```
typedef enum
{
    QL_RTC_SUCCESS = QL_SUCCESS,
    QL_RTC_INVALID_PARAM_ERR = 1|QL_RTC_ERRCODE_BASE,
    QL_RTC_SET_TIME_ERROR,
    QL_RTC_SET_CB_ERR,
    QL_RTC_ENABLE_ALARM_ERR,
    QL_RTC_REMOVE_ALARM_ERR
}ql_errcode_rtc_e
```

- 参数

参数	描述
QL_RTC_SUCCESS	函数执行成功
QL_RTC_INVALID_PARAM_ERR	输入无效参数
QL_RTC_SET_TIME_ERROR	设置 RTC 时间错误

<code>QL_RTC_SET_CB_ERR</code>	注册回调函数错误
<code>QL_RTC_ENABLE_ALARM_ERR</code>	启用警报错误
<code>QL_RTC_REMOVE_ALARM_ERR</code>	删除警报错误

2.3.1.2. ql_rtc_time_t

RTC 时间结构体定义如下：

```
typedef struct ql_rtc_time_struct {
    int tm_sec;
    int tm_min;
    int tm_hour;
    int tm_mday;
    int tm_mon;
    int tm_year;
    int tm_wday;
}ql_rtc_time_t
```

● 参数

类型	参数	描述
int	<code>tm_sec</code>	秒。范围：0~59。
int	<code>tm_min</code>	分。范围：0~59。
int	<code>tm_hour</code>	时。范围：0~23。
int	<code>tm_mday</code>	日。范围：1~31。
int	<code>tm_mon</code>	月。范围：1~12。
int	<code>tm_year</code>	年。范围：2000~2100。
int	<code>tm_wday</code>	一周中的第几天。范围：0~6，0 为周日。

2.3.2. ql_rtc_get_time

该函数用于获取当前 RTC 时间。

● 函数原型

```
ql_errcode_rtc_e ql_rtc_get_time(ql_rtc_time_t *tm)
```

- 参数

tm:

[Out] RTC 时间结构体指针。详情请参考第 2.3.1.2 章。

- 返回值

详情请参考第 2.3.1.1 章。

2.3.3. ql_rtc_print_time

该函数用于打印 RTC 时间。

- 函数原型

```
ql_errcode_rtc_e ql_rtc_print_time(ql_rtc_time_t tm)
```

- 参数

tm:

[In] RTC 时间结构体。详情请参考第 2.3.1.2 章。

- 返回值

详情请参考第 2.3.1.1 章。

2.3.4. ql_rtc_set_alarm

该函数用于设置警报时间。

- 函数原型

```
ql_errcode_rtc_e ql_rtc_set_alarm (ql_rtc_time_t* tm)
```

- 参数

tm:

[In] RTC 时间结构体指针。详情请参考第 2.3.1.2 章。

- 返回值

详情请参考第 2.3.1.1 章。

2.3.5. ql_rtc_get_alarm

该函数用于获取警报时间。

- 函数原型

```
ql_errcode_rtc_e ql_rtc_get_alarm (ql_rtc_time_t* tm)
```

- 参数

tm:

[In] RTC 时间结构体指针。详情请参考第2.3.1.2章。

- 返回值

详情请参考第2.3.1.1章。

2.3.6. ql_rtc_enable_alarm

该函数用于启用警报功能。

- 函数原型

```
ql_errcode_rtc_e ql_rtc_enable_alarm (unsigned char on_off)
```

- 参数

on_off:

[In] 使能参数，即是否启用警报功能。

1 启用警报

0 删除警报

- 返回值

详情请参考第2.3.1.1章。

2.3.7. ql_rtc_register_cb

该函数用于注册警报回调函数。

- 函数原型

```
ql_errcode_rtc_e ql_rtc_register_cb (ql_rtc_cb cb)
```

- 参数

cb:

[In] 警报回调函数指针。详情请参考第2.3.7.1章。

- 返回值

详情请参考第2.3.1.1章。

2.3.7.1. ql_rtc_cb

该函数为警报回调函数。

- 函数类型

```
typedef void (*ql_rtc_cb)(void)
```

2.3.8. ql_rtc_get_localtime

该函数用于获取本地时间（含时区）。

- 函数原型

```
ql_errcode_rtc_e ql_rtc_get_localtime(ql_rtc_time_t *tm)
```

- 参数

tm:

[Out] RTC 时间结构体指针。详情请参考第2.3.1.2章。

- 返回值

详情请参考第2.3.1.1章。

3 示例

本章节主要介绍在 APP 侧如何使用上述 API 进行 RTC 开发以及相关调试。

3.1. 开发示例

QuecOpen SDK 包中提供了 RTC API 示例，即 `ql_rtc_demo.c`，位于 `\components\ql-application\rtc` 目录下。文件中主要包含如何获取、设置、打印时间及如何设置警报功能。入口函数为 `ql_rtc_app_init()`，该函数启动 `ql_rtc_demo_thread()` 任务，如下图所示。

```
void ql_rtc_app_init()
{
    QIOSStatus err = QL_OSI_SUCCESS;
    ql_task_t rtc_task = NULL;

    err = ql_rtos_task_create(&rtc_task, 1024, APP_PRIORITY_NORMAL, "ql_rtcdemo", ql_rtc_demo_thread, NULL, 1);
    if( err != QL_OSI_SUCCESS )
    {
        QL_RTCDEMO_LOG("rtc demo task created failed");
    }
}
```

图 1: ql_rtc_app_init()

该应用程序示例将依次完成：获取当前时间并打印、设置新的时间、获取更新后的时间并打印、设置及启用警报功能。到达警报时间时，程序将自动调用 `ql_rtc_test_callback()` 回调函数。

```
static void ql_rtc_demo_thread()
{
    QL_RTCDEMO_LOG("ql rtc demo enter! \n");

    ql_rtc_time_t tm;

    //get current time
    ql_rtc_get_time(&tm);
    ql_rtc_print_time(tm);

    //2020-12-22 16:50:30 [WED]
    tm.tm_year = 2020;
    tm.tm_mon = 12;
    tm.tm_mday = 22;
    tm.tm_wday = 2;
    tm.tm_hour = 16;
    tm.tm_min = 50;
    tm.tm_sec = 30;
    ql_rtc_print_time(tm);

    //set time
    ql_rtc_set_time(&tm);

    ql_rtc_get_time(&tm);
    ql_rtc_print_time(tm);

    //set & enable alarm
    >> tm.tm_sec += 20;
    >> ql_rtc_set_alarm(&tm);
    >> ql_rtc_register_cb(ql_rtc_test_callback);
    >> ql_rtc_enable_alarm(1);

    >> while (1)
    >> {
    >>     ql_rtc_get_time(&tm);
    >>     QL_RTCDEMO_LOG("=====print current time=====\\r\\n");
    >>     ql_rtc_print_time(tm);
    >>     ql_rtc_task_sleep_s(5);
    >> }
} « end ql_rtc_demo_thread »
```

图 2: ql_rtc_demo_thread()

如下图所示，上述应用程序示例已在 `ql_init_demo_thread()` 线程中默认不启动。需要测试时请取消注释。

```
#ifdef QL_APP_FEATURE_GNSS
    //ql_gnss_app_init();
#endif

#ifdef QL_APP_FEATURE_SPI
    //ql_spi_demo_init();
#endif

#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_HTTP_FOTA
    //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

#ifdef QL_APP_FEATURE_RTC
    //ql_rtc_app_init();
#endif
```




图 3: ql_init_demo_thread()

3.2. 功能调试

ECx00U 系列和 EGx00U QuecOpen 模块可通过移远通信 LTE OPEN EVB 进行 RTC 功能调试。

将编译版本烧录到模块中，使用 USB 线连 EVB 的 USB 端口和 PC。使用 coolwatcher_debughost.exe 中的 Trace Tool 功能抓取 log 后，可通过 USB 的 AP Log 端口（即下图中 Quectel USB AP Log Port (COM14)）查看该应用程序示例的调试信息。

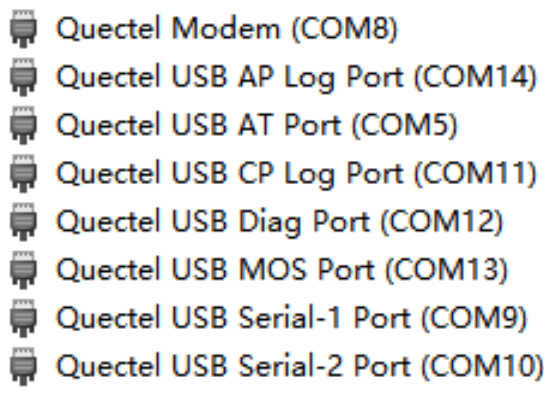


图 4: COM 设备

模块开机后自动启动 `ql_rtc_app_init()`。AP Log 端口输出的调试信息如下图所示:

Index	Received	Tick	Level	
257	11:11:18.351	15298	RTCA/I	RTC store nv length/6
819	11:11:21.558	828	QOPN/I	[ql_RTCDEMO][ql_rtc_demo_thread, 64] ql rtc demo enter!
820	11:11:21.558	830	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 17:03:18 [TUS]
821	11:11:21.558	830	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:30 [TUS]
822	11:11:21.558	831	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:30 [TUS]
826	11:11:21.558	836	RTCA/I	RTC set alarm day/7661 hour/16 min/50 sec/50
827	11:11:21.558	861	RTCA/I	RTC store nv length/70
828	11:11:21.558	866	QOPN/I	[ql_RTCDEMO][ql_rtc_demo_thread, 97] =====print current time=====
829	11:11:21.581	867	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:30 [TUS]
832	11:11:21.649	2939	RTCA/D	RTC intr 0x00001000
841	11:11:26.445	17105	QOPN/I	[ql_RTCDEMO][ql_rtc_demo_thread, 97] =====print current time=====
842	11:11:26.504	17105	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:34 [TUS]
1037	11:11:31.436	33490	QOPN/I	[ql_RTCDEMO][ql_rtc_demo_thread, 97] =====print current time=====
1038	11:11:31.488	33490	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:39 [TUS]
1046	11:11:36.444	49873	QOPN/I	[ql_RTCDEMO][ql_rtc_demo_thread, 97] =====print current time=====
1047	11:11:36.503	49873	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:44 [TUS]
1094	11:11:41.449	721	QOPN/I	[ql_RTCDEMO][ql_rtc_demo_thread, 97] =====print current time=====
1095	11:11:41.449	722	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:49 [TUS]
1096	11:11:41.449	838	QOPN/I	[ql_RTCDEMO][ql_rtc_test_callback, 49] ql rtc test callback come in!
1097	11:11:41.449	839	QOPN/I	[ql_RTCDEMO][ql_rtc_test_callback, 57] =====callback print alarm time=====
1098	11:11:41.449	839	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:50 [TUS]
1099	11:11:41.449	839	RTCA/I	RTC unset alarm
1100	11:11:41.505	862	RTCA/I	RTC store nv length/6
1124	11:11:46.440	17106	QOPN/I	[ql_RTCDEMO][ql_rtc_demo_thread, 97] =====print current time=====
1125	11:11:46.501	17106	QOPN/I	[ql_RTC][ql_rtc_print_time, 219] 2020-12-22 16:50:54 [TUS]

图 5: RTC log 信息

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_快速开发指导

表 4: 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
EVB	Evaluation Board	评估板
IoT	Internet of Things	物联网
LTE	Long-Term Evolution	长期演进
RTC	Real-Time Clock	实时时钟
RTOS	Real-Time Operating 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