

ECx00U&EGx00U 系列

QuecOpen PSM 应用指导

LTE Standard 模块系列

版本：1.0

日期：2022-02-11

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。

隐私声明

为实现移远通信产品功能，特定设备数据将会上传至移远通信或第三方服务器（包括运营商、芯片供应商或您指定的服务器）。移远通信严格遵守相关法律法规，仅为实现产品功能之目的或在适用法律允许的情况下保留、使用、披露或以其他方式处理相关数据。当您与第三方进行数据交互前，请自行了解其隐私保护和数据安全政策。

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 2022，保留一切权利。

Copyright © Quectel Wireless Solutions Co., Ltd. 2022.

文档历史

修订记录

版本	日期	作者	变更表述
-	2021-12-28	Felix Lin	文档创建
1.0	2022-02-11	Felix Lin	受控版本

目录

文档历史	3
目录	4
表格索引	6
图片索引	7
1 引言	8
1.1. 适用模块	8
2 PSM API	9
2.1. 头文件	9
2.2. 函数概览	9
2.3. API 详解	9
2.3.1. ql_psm_sleep_enable	9
2.3.1.1. ql_errcode_sleep	10
2.3.1.2. periodic_TAU_time (T3412)	11
2.3.1.3. active_time (T3324)	12
2.3.2. ql_psm_register_enter_cb	12
2.3.2.1. ql_psm_enter_callback	13
3 示例	14
3.1. APP PSM 开发示例	14
3.1.1. PSM 应用程序示例说明	14
3.1.2. 应用程序示例自启动	15
3.2. APP PSM 调试及测试	16
3.2.1. APP PSM 调试	16
3.2.1.1. 调试准备	16
3.2.1.2. 使用模块 USB 接口抓取开机 AP Log	17
3.2.1.3. 使用模块 USB 接口抓取开机 CP Log	17
3.2.1.4. 使用模块调试串口抓取 AP Log	18
3.2.1.4.1. 模块开机进入 PSM 的 AP Log	19
3.2.1.4.2. RTC 唤醒的 AP Log	19
3.2.2. APP PSM 测试	20
3.2.2.1. 测试准备	20
3.2.2.1.1. 设置 RTC 唤醒	20
3.2.2.1.2. 设置串口无数据交互时进入 PSM 的时间	22
3.2.2.1.3. 使能增强型休眠模式	22
3.2.2.1.4. 新增 TCP 数据发送	23
3.2.2.2. 测试案例	24
3.2.2.2.1. 测试示例	24
3.2.2.2.2. 测试数据	24
3.2.2.2.3. 测试结论	25
3.2.2.3. 注意事项	25

4	附录 参考文档及术语缩写	26
---	--------------------	----

表格索引

表 1: 适用模块.....	8
表 2: 函数概览.....	9
表 3: EC600U-CN PSM 唤醒平均耗流数据	24
表 4: 参考文档.....	26
表 5: 术语缩写	26

图片索引

图 1: ql_psm_app_init().....	14
图 2: ql_psm_demo_thread()	15
图 3: ql_init_demo_thread()	15
图 4: COM 设备图	16
图 5: coolwatcher_usb.exe 关键字过滤	16
图 6: USB AP Log	17
图 7: USB CP Log	18
图 8: ql_psm_demo_thread()	19
图 9: 调试串口模块开机进入 PSM 的 AP Log.....	19
图 10: 调试串口 RTC 唤醒的 AP Log	20
图 11: 注册 RTC 回调	21
图 12: RTC 回调中重置 RTC 唤醒	21
图 13: 设置串口无数据交互时进入 PSM 的时间	22
图 14: 使能增强型休眠模式.....	22
图 15: PSM 开机添加 TCP 数据发送	23
图 16: PSM 唤醒添加 TCP 数据发送	23
图 17: EC600U-CN PSM 唤醒平均耗流.....	24

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 模块和省电模式（Power Saving Mode，以下简称 PSM）相关 API、开发示例及调试与测试流程。

备注

1. 模块可通过以下方式从 PSM 中被唤醒：
 - 1) TAU 周期请求定时器（T3412）唤醒；
 - 2) RTC 唤醒；
 - 3) 拉低 PWRKEY 唤醒；
 - 4) 中断唤醒引脚 PSM_EXT_INT* 被拉高（仅 EC600U 系列模块支持）。
2. 模块进入 PSM 条件如下：
 - 1) 运营商和 (U)SIM 卡支持 PSM；
 - 2) 使能 PSM 后，模块在驻网成功的情况下，无任何数据交互动作后，会自动进入 PSM。
 - 3) 网络下发的 T3412 值与 T3324 值之间的差值大于 5 分钟。

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 PSM API

2.1. 头文件

`ql_power.h` 文件即为 PSM API 的头文件，位于 `\components\ql-kernel\inc` 目录下。若无特别说明，本文档所涉及头文件均在该目录下。

2.2. 函数概览

表 2：函数概览

函数	说明
<code>ql_psm_sleep_enable()</code>	使能 PSM
<code>ql_psm_register_enter_cb()</code>	注册 PSM 回调函数

2.3. API 详解

2.3.1. ql_psm_sleep_enable

该函数用于使能 PSM。

- 函数原型

```
ql_errcode_sleep ql_psm_sleep_enable(uint8_t nSim, bool psm_enable, const char
*periodic_TAU_time, const char *active_time)
```

● 参数

nSim:

[In] (U)SIM 卡 ID。

- 0 (U)SIM 卡1
- 1 (U)SIM 卡 2

psm_enable:

[In] 关闭/使能 PSM。

- FALSE 关闭
- TRUE 使能

periodic_TAU_time:

[In] TAU 周期请求定时器（T3412）的值，8 位（1 字节）格式。详情参考第 2.3.1.2 章。

active_time:

[In] 激活定时器（T3324）的值，8 位（1 字节）格式。详情参考第 2.3.1.3 章。

● 返回值

详见第 2.3.1.1 章。

2.3.1.1. ql_errcode_sleep

```
typedef enum
{
    QL_SLEEP_SUCCESS                = QL_SUCCESS,
    QL_SLEEP_INVALID_PARAM          = (QL_COMPONENT_PM_SLEEP << 16) | 1000,
    QL_SLEEP_LOCK_CREATE_FAIL       = (QL_COMPONENT_PM_SLEEP << 16) | 1001,
    QL_SLEEP_LOCK_DELETE_FAIL       = (QL_COMPONENT_PM_SLEEP << 16) | 1002,
    QL_SLEEP_LOCK_LOCK_FAIL         = (QL_COMPONENT_PM_SLEEP << 16) | 1003,
    QL_SLEEP_LOCK_UNLOCK_FAIL       = (QL_COMPONENT_PM_SLEEP << 16) | 1004,
    QL_SLEEP_LOCK_AUTOSLEEP_FAIL    = (QL_COMPONENT_PM_SLEEP << 16) | 1005,
    QL_SLEEP_PARAM_SAVE_FAIL        = (QL_COMPONENT_PM_SLEEP << 16) | 1006,
    QL_SLEEP_EXECUTE_FAIL           = (QL_COMPONENT_PM_SLEEP << 16) | 1007,
}ql_errcode_sleep
```

● 成员

成员	描述
QL_SLEEP_SUCCESS	操作成功
QL_SLEEP_INVALID_PARAM	参数无效

QL_SLEEP_LOCK_CREATE_FAIL	创建休眠锁失败
QL_SLEEP_LOCK_DELETE_FAIL	删除休眠锁失败
QL_SLEEP_LOCK_LOCK_FAIL	锁定休眠锁失败
QL_SLEEP_LOCK_UNLOCK_FAIL	释放休眠锁失败
QL_SLEEP_LOCK_AUTOSLEEP_FAIL	启动自动休眠失败
QL_SLEEP_PARAM_SAVE_FAIL	参数保存失败
QL_SLEEP_EXECUTE_FAIL	操作失败

2.3.1.2. periodic_TAU_time (T3412)

参考 3GPP TS 24.008 V17.3.0 (2021-06) Table 10.5.163a/3GPP TS 24.008，内容如下：

GPRS Timer 3 value (octet 3)

Bit 5 到 Bit 1 表示二进制编码的定时器值。

Bit 6 到 Bit 8 定义 GPRS 定时器的定时器值单位，如下：

Bits

8 7 6

0 0 0 值以 10 分钟的倍数递增

0 0 1 值以 1 小时的倍数递增

0 1 0 值以 10 小时的倍数递增

0 1 1 值以 2 秒的倍数递增

1 0 0 值以 30 秒的倍数递增

1 0 1 值以 1 分钟的倍数递增

1 1 0 值以 320 小时的倍数递增（参考 3GPP TS 24.008 V17.3.0 (2021-06) Table 10.5.163a/3GPP TS 24.008 中的 NOTE 1）

1 1 1 值表示定时器已停用（参考 3GPP TS 24.008 V17.3.0 (2021-06) Table 10.5.163a/3GPP TS 24.008 中的 NOTE 2）

2.3.1.3. active_time (T3324)

参考 3GPP TS 24.008 V17.3.0 (2021-06) Table 10.5.163/3GPP TS 24.008 和 Table 10.5.172/3GPP TS 24.008, 内容如下:

Timer value (octet 3)

Bit 5 到 Bit 1 表示二进制编码的定时器值。

Bit 6 到 Bit 8 定义 GPRS 定时器的定时器值单位, 如下:

Bits

8 7 6

0 0 0 值以 2 秒的倍数递增

0 0 1 值以 1 分钟的倍数递增

0 1 0 值以 1/10 小时的倍数递增

1 1 1 表示定时器已停用。

其他值表示 1 分钟的倍数。

2.3.2. ql_psm_register_enter_cb

该函数用于注册 PSM 回调函数。

● 函数原型

```
ql_errcode_sleep ql_psm_register_enter_cb(ql_psm_enter_callback cb, void *ctx)
```

● 参数

cb:

[In] 回调函数指针。详情请参考第 2.3.2.1 章。

ctx:

[In] 回调函数指针参数。详情请参考第 2.3.2.1 章。

● 返回值

详见第 2.3.1.1 章。

2.3.2.1. ql_psm_enter_callback

定义函数类型，用于定义 PSM 回调函数。

- 函数类型

```
typedef void (*ql_psm_enter_callback)(void *ctx);
```

- 参数

ctx:

[In] PSM 回调函数参数指针。

3 示例

本章节主要介绍在 APP 侧如何使用上述 API 进行低功耗开发以及简单的调试，并演示如何进入和唤醒 PSM。

3.1. APP PSM 开发示例

3.1.1. PSM 应用程序示例说明

ECx00U 系列和 EGx00U 模块 QuecOpen SDK 代码中提供了 PSM 的示例，即 *psm_demo.c* 文件，位于 `components\ql-application\power` 目录下。

psm_demo.c 文件中主要包含如何进入和唤醒 PSM 功能等内容。入口函数为 *ql_psm_app_init()*，如下图所示。

```
void ql_psm_app_init(void)
{
    ... QLOSStatus err = QL_OSI_SUCCESS;
    ... err = ql_rtos_task_create(&psm_task, 4096, APP_PRIORITY_NORMAL, "ql_psmdemo", ql_psm_demo_thread, NULL, 3);
    ... if(err != QL_OSI_SUCCESS)
    ... {
    ...     QL_PSMDEMO_LOG("psm_demo_task_created_failed", err);
    ...     return;
    ... }
    ... return;
}
```

图 1: ql_psm_app_init()

psm_demo 流程如下：

1. 注册 PSM 回调函数（用户可使用此回调函数来保存模块进入 PSM 前的数据），网络通知回调函数（告知驻网时 RTC 更新状态和 PDP 激活状态）；
2. 使能 PSM，将 **AT+CFUN=0** 切换为 **AT+CFUN=1**；
3. 等待网络更新 RTC；
4. RTC 更新完成后，激活 1 路 PDP，打印 IP 地址，同时设置 RTC 唤醒；
5. 使能自动休眠，模块准备进入 PSM。满足进入 PSM 的条件后，模块进入 PSM；
6. RTC 唤醒会定时将模块从 PSM 中唤醒，*psm_demo* 确认开机原因为 PSM 唤醒，则打印相关信息，同时打印当前激活 PDP 的 IP 地址（进入 PSM 后，网络仍处于注册状态，所以该地址与第一次上电开机激活的 PDP 地址相同），并重置 RTC 唤醒；
7. 满足进入 PSM 的条件后，模块将重新进入 PSM。

```
static void ql_psm_demo_thread(void *param)
{
    uint8_t source = 0;
    int ret = 0, retry_count = 0;
    int profile_idx = 1; // range 1 to 7
    ql_log_set_port(QL_LOG_PORT_UART);
    ql_nw_reg_status_info_s *nw_info = (ql_nw_reg_status_info_s *)calloc(1, sizeof(ql_nw_reg_status_info_s));
    ql_data_call_info_s *info = (ql_data_call_info_s *)calloc(1, sizeof(ql_data_call_info_s));

    if(nw_info == NULL || info == NULL)
    {
        QL_PSMDEMO_LOG("calloc fail!");
        goto PSM_EXIT;
    }

    if(QL_SLEEP_SUCCESS != ql_psm_register_enter_cb(ql_psm_cb, NULL))
    {
        QL_PSMDEMO_LOG("register psm callback fail!");
        goto PSM_EXIT;
    }

    if(QL_NW_SUCCESS != ql_nw_register_cb(ql_psm_nw_notify_cb))
    {
        QL_PSMDEMO_LOG("register network callback fail!");
        goto PSM_EXIT;
    }
}
```

图 2: ql_psm_demo_thread()

3.1.2. 应用程序示例自启动

上述应用程序示例在 `ql_init_demo_thread()` 线程中默认不启动。如下图所示，需要测试（即需要应用程序示例自启动）时，请取消 `ql_init_demo_thread()` 前的注释。

```
359: #ifdef QL_APP_FEATURE_BT_HFP
360:     //ql_bt_hfp_demo_init();
361: #endif
362: #ifdef QL_APP_FEATURE_BT_A2DP_AVRCP
363:     //ql_bt_a2dp_avrcp_app_init();
364: #endif
365: #endif
366:     //ql_sim_app_init();
367:     //ql_power_app_init();
368: #ifdef QL_APP_FEATURE_PSM
369:     ql_psm_app_init();
370: #endif
371:
372: #ifdef QL_APP_FEATURE_CLOUDOTA
373:     //ql_cloudota_app_init();
374: #endif
375:
376: #ifdef QL_APP_FEATURE_PWK
377:     //ql_pwrkey_app_init();
378: #endif
```

图 3: ql_init_demo_thread()

3.2. APP PSM 调试及测试

本章节以 EC600U-CN 模块为例, 介绍如何使用移远通信 LTE OPEN EVB 进行 PSM 功能调试和测试。

备注

1. PSM 调试仅做 PSM 功能演示, 不包含相关测试项目。
2. PSM 测试仅针对移远通信测试案例和指导用户操作, 不在 PSM 示例中体现。

3.2.1. APP PSM 调试

3.2.1.1. 调试准备

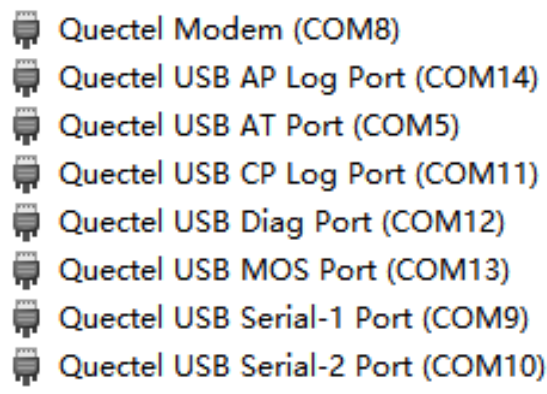


图 4: COM 设备图

其中, *coolwatcher_usb.exe* 和 *coolwatcher_debughost.exe* 里的 Trace tool 功能打印的 Log 信息可以进行关键字过滤, 以查看需要的 Log 信息, 且支持通过添加 “|” 同时搜索多个关键字。有关 Log 抓取详细信息, 请参考文档 [2]。

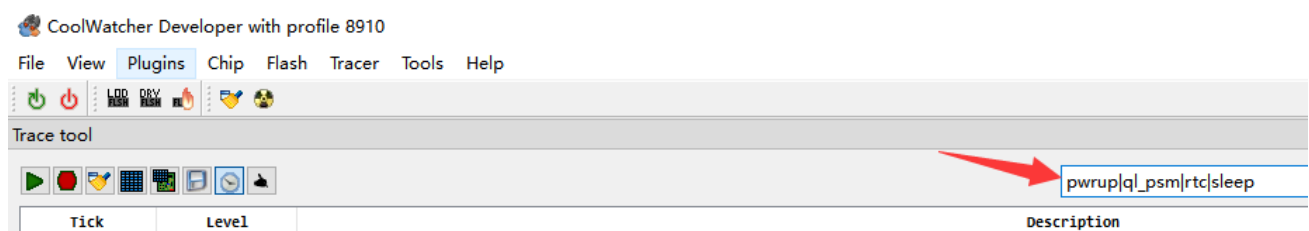


图 5: coolwatcher_usb.exe 关键字过滤

3.2.1.2. 使用模块 USB 接口抓取开机 AP Log

- 关键字过滤: pwrup|ql_psm|rtc|sleep
- 步骤: 模块通过 USB 数据线连接电脑, 打开 *coolwatcher_usb.exe* 抓取相应 AP Log。
- 目的: 确认 *psm_demo* 是否运行正常。

RTCA/I	RTC store nv length/12038	
RTCA/I	RTC set alarm day/7885 hour/13 min/39 sec/49	
RTCA/D	RTC intr 0x00001000	1 开机原因: 引脚复位
QUEC/I	atEngineCreateUartChannel,328 Init at uart port 827609685	
QUEC/I	quec_init_thread,889 ### quec init done... pwrup=2 ###	
QOPN/I	[ql_POWER][ql_psm_sleep_enable, 585] enter 0,1,(null),(null)	2 使能PSM
QOPN/I	[ql_POWER][ql_psm_sleep_enable, 650] nvWriten=0x00, bitmap=0x00	
QOPN/I	[ql_PSM][ql_psm_set_cfun, 204] cfun: 1	3 CFUN切换
RTCA/I	RTC unset alarm	
RTCA/I	RTC write second 7885/13:38:58	
QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 181] Sim0 data reg status changed, current status is 0	
RTCA/I	RTC set alarm day/7885 hour/13 min/39 sec/49	
RTCA/I	RTC store nv length/12038	
RTCA/D	RTC intr 0x00001000	
RTCA/D	RTC intr 0x00001000	
QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 181] Sim0 data reg status changed, current status is 0	
QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 181] Sim0 data reg status changed, current status is 1	
RTCA/I	RTC unset alarm	
RTCA/I	RTC write second 7885/13:39:3	4 等待驻网后RTC更新
RTCA/I	RTC set alarm day/7885 hour/13 min/39 sec/49	
RTCA/I	RTC store nv length/12038	
QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 174] time update!	5 终端PDP激活
QOPN/I	[ql_PSM][ql_psm_demo_thread, 350] info.v4.addr.ip: 10.0.161.53	
QOPN/I	[ql_RTC][ql_rtc_print_time, 305] 2021-08-03 13:39:03 [TUS]	
QOPN/I	[ql_RTC][ql_rtc_set_alarm, 338] ql rtc set alarm success!	6 输出RTC, 设置并使能RTC唤醒
QOPN/I	[ql_RTC][ql_rtc_print_time, 305] 2021-08-03 13:40:03 [TUS]	
RTCA/I	RTC set alarm day/7885 hour/13 min/40 sec/3	
RTCA/I	RTC store nv length/12038	
QOPN/I	[ql_RTC][ql_rtc_enable_alarm, 378] ql rtc enable alarm success!	
QOPN/I	[ql_PSM][ql_psm_demo_thread, 363] psm enable!	7 使能自动休眠
QUEC/I	quec_pmsource_auto_sleep,324 start auto sleep mode	
RTCA/D	RTC intr 0x00001100	
QUEC/I	quec_print_os_pm_state,825 source CP not allow sleep	8 无法进入休眠原因
QUEC/I	quec_print_os_pm_state,825 source URT1 not allow sleep	
QUEC/I	quec_print os pm state,825 source USB2 not allow sleep	

图 6: USB AP Log

3.2.1.3. 使用模块 USB 接口抓取开机 CP Log

- 步骤: 模块通过 USB 数据线连接电脑, 打开 *ArmTracer.exe* 抓取相应 CP Log。
- 目的: 查看终端上报的用户设置的 T3324 和 T3412 定时器信息和网络下发的 T3324 和 T3412 定时器信息, 确认模块是否可以正常进入 PSM。

Sim1	0x4422	0x1e403	0x329	0x973	↑ EMM_ATTACH_REQUEST	09:16:50:28.909
Sim1	0x4492	0x1e449	0x32a	0x977	↑ rrcConnectionRequest	09:16:50:28.913
Sim1	0x452a	0x1e49a	0x32e	0x98c	↑ rrcConnectionSetup	09:16:50:28.934
Sim1	0x4555	0x1e4e6	0x32f	0x98d	↑ rrcConnectionSetupComplete	09:16:50:28.936
Sim1	0x4a81	0x1f7e7	0x34e	0x9d1	↓ dlInformationTransfer	09:16:50:29.79
Sim1	0x4a91	0x1f7f3	0x34e	0x9d1	↓ EMM_AUTHENTICATION_REQUEST	09:16:50:29.80
Sim1	0x4bb5	0x1f971	0x353	0x9d1	↓ sib5	09:16:50:29.103
Sim1	0x5169	0x2063e	0x37e	0x9d1	↑ EMM_AUTHENTICATION_RESPONSE	09:16:50:29.303
Sim1	0x5172	0x20647	0x37e	0x9d1	↑ ulInformationTransfer	09:16:50:29.304
Sim1	0x5294	0x2083b	0x385	0xb1c	↓ dlInformationTransfer	09:16:50:29.335
Sim1	0x52a5	0x20855	0x385	0xb1e	↓ EMM_SECURITY_MODE_COMMAND	09:16:50:29.336
Sim1	0x52ab	0x20861	0x386	0xb1e	↓ EMM_SECURITY_MODE_COMPLETE	09:16:50:29.337
Sim1	0x52c0	0x20865	0x386	0xb1f	↓ ulInformationTransfer	09:16:50:29.337
Sim1	0x571f	0x2107a	0x3a1	0xb9d	↓ dlInformationTransfer	09:16:50:29.463
Sim1	0x5738	0x21087	0x3a1	0xb9e	↓ ESM_INFORMATION_REQUEST	09:16:50:29.464
Sim1	0x5742	0x2108a	0x3a1	0xb9e	↓ ESM_INFORMATION_RESPONSE	09:16:50:29.464
Sim1	0x574d	0x21094	0x3a1	0xb9e	↓ ulInformationTransfer	09:16:50:29.465
Sim1	0x5b4b	0x2180c	0x3bd	0xc1f	↓ ueCapabilityEnquiry	09:16:50:29.593
Sim1	0x5b57	0x21816	0x3bd	0xc20	↓ ueCapabilityRAT_Container	09:16:50:29.595
Sim1	0x5b58	0x21818	0x3bd	0xc21	↓ ueCapabilityInformation	09:16:50:29.595
Sim1	0x5c05	0x21a55	0x3c2	0xc37	↓ securityModeCommand	09:16:50:29.617
Sim1	0x5c0b	0x21a60	0x3c2	0xc38	↓ securityModeComplete	09:16:50:29.618
Sim1	0x5c09	0x21a64	0x3c2	0xc38	↓ rrcConnectionReconfiguration	09:16:50:29.618
Sim1	0x5c0f	0x21a63	0x3c3	0xc3a	↓ rrcConnectionReconfigurationComplete	09:16:50:29.620
Sim1	0x5d22	0x21a89	0x3c3	0xc3a	↓ EMM_ATTACH_ACCEPT	09:16:50:29.621
Sim1	0x5d23	0x21a89	0x3c3	0xc3a	↓ ESM_ACTIVATE_DEFAULT_EFS_BEARER_C...	09:16:50:29.621
Sim1	0x5d67	0x21aa8	0x3c3	0xc3e	↓ EMM_ATTACH_COMPLETE	09:16:50:29.622
Sim1	0x5d68	0x21aa9	0x3c3	0xc3e	↓ ESM_ACTIVATE_DEFAULT_EFS_BEARER_C...	09:16:50:29.622
Sim1	0x5d77	0x21ab3	0x3c4	0xc3d	↓ ulInformationTransfer	09:16:50:29.623
Sim1	0x5f0f	0x21b4c	0x3c6	0xc46	↓ rrcConnectionReconfiguration	09:16:50:29.632
Sim1	0x5f1f	0x21b5c	0x3c6	0xc47	↓ rrcConnectionReconfigurationComplete	09:16:50:29.633
Sim1	0x6053	0x21c43	0x3cb	0xc5e	↓ measurementReport	09:16:50:29.656
Sim1	0x606f	0x21c41	0x3cb	0xc5f	↓ measurementReport	09:16:50:29.657
Sim1	0x76a2	0x2462b	0x457	0xc44	↓ measurementReport	09:16:50:30.302
Sim1	0x7774	0x24812	0x454	0xf02	↓ rrcConnectionReconfiguration	09:16:50:30.332
Sim1	0x7787	0x2481a	0x454	0xf02	↓ rrcConnectionReconfigurationComplete	09:16:50:30.332
Sim1	0x183e4	0x3e403	0x9f5	0xd2	↓ rrcConnectionRelease	09:16:50:36.940
Sim1	0x187e7	0x3f45e	0xd07	0x126	↓ MasterInformationBlock	09:16:50:37.24
Sim1	0x1882e	0x3f53d	0xd0a	0x133	↓ systemInformationBlockType1	09:16:50:37.38

图 7：USB CP Log

3.2.1.4. 使用模块调试串口抓取 AP Log

- 环境准备：将模块调试串口和主串口连接电脑，打开 `coolwatcher_debughost.exe` 准备抓取相应 AP Log。
- 目的：由于模块的 USB 接口通过 USB 数据线连接电脑，从而导致模块无法休眠，如需抓取休眠和唤醒的 AP Log，需配置调试串口抓取 AP Log。
- 步骤：
 - 1) 开机查看 AP Log 和 CP Log，确认(U)SIM 卡和运营商支持 PSM 后，配置调试串口抓取 AP Log。
 - 2) 在 PSM 示例中添加头文件 `ql_log.h`，在 PSM 线程开始时调用 `ql_log_set_port(QL_LOG_PORT_UART)`，重启后即可通过调试串口抓取 AP Log。

```
static void ql_psm_demo_thread(void *param)
{
    uint8_t source = 0;
    int ret = 0, retry_count = 0;
    int profile_idx = 1; //range 1 to 7
    ql_log_set_port(QL_LOG_PORT_UART);
    ql_nw_reg_status_info_s *nw_info = (ql_nw_reg_status_info_s *)calloc(1, sizeof(ql_nw_reg_status_info_s));
    ql_data_call_info_s *info = (ql_data_call_info_s *)calloc(1, sizeof(ql_data_call_info_s));
    if(nw_info == NULL || info == NULL)
    {
        QL_PSMDEMO_LOG("calloc fail!");
    }
}
```

图 8: ql_psm_demo_thread()

3.2.1.4.1. 模块开机进入 PSM 的 AP Log

通过模块调试串口抓取的 AP Log 如下所示。

21:55:59.167	17147	RTCA/D	RTC intr 0x00001100	1 开机原因
21:55:59.369	21466	QUEC/I	atEngineCreateUartChannel,328 Init at uart port 827609685	
21:56:01.287	52574	QUEC/I	quec_init_thread,889 ### quec init done... pwrup=1 ###	2 使能PSM
21:56:01.295	52583	QOPN/I	[ql_POWER][ql_psm_sleep_enable, 585] enter 0,1,(null),(null)	
21:56:01.302	52589	QOPN/I	[ql_POWER][ql_psm_sleep_enable, 650] nvWrite=0x00, bitmap=0x00	
21:56:01.308	52597	QOPN/I	[ql_PSM][ql_psm_set_cfun, 204] cfun: 1	3 CFUN切换
21:56:01.337	52708	RTCA/I	RTC unset alarm	
21:56:01.338	52709	RTCA/I	RTC write second 7885/13:53:20	
21:56:01.342	52834	RTCA/I	RTC set alarm day/7885 hour/13 min/54 sec/17	
21:56:01.344	53055	RTCA/I	RTC store nv length/12102	
21:56:01.356	53073	QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 181] Sim0 data reg status changed, current status is 0	
21:56:01.477	55035	RTCA/D	RTC intr 0x00001100	
21:56:01.590	57367	QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 181] Sim0 data reg status changed, current status is 0	
21:56:04.653	41842	QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 181] Sim0 data reg status changed, current status is 1	
21:56:04.671	42250	RTCA/I	RTC unset alarm	4 RTC更新
21:56:04.671	42251	RTCA/I	RTC write second 7885/13:56:4	
21:56:04.676	42343	QOPN/I	[ql_PSM][ql_psm_nw_notify_cb, 174] time update!	
21:56:04.676	42411	RTCA/I	RTC store nv length/12102	
21:56:04.806	42576	QOPN/I	[ql_PSM][ql_psm_demo_thread, 350] info.v4.addr.ip: 10.31.133.105	5 PDP激活
21:56:04.808	42578	QOPN/I	[ql_RTC][ql_rtc_print_time, 305] 2021-08-03 13:56:04 [TUS]	
21:56:04.808	42581	QOPN/I	[ql_RTC][ql_rtc_set_alarm, 330] ql rtc set alarm success!	
21:56:04.810	42581	QOPN/I	[ql_RTC][ql_rtc_print_time, 305] 2021-08-03 13:57:04 [TUS]	6 更新RTC并重置RTC唤醒
21:56:04.810	42583	RTCA/I	RTC set alarm day/7885 hour/13 min/57 sec/4	
21:56:04.814	43499	RTCA/I	RTC store nv length/12166	
21:56:04.814	43507	QOPN/I	[ql_RTC][ql_rtc_enable_alarm, 378] ql rtc enable alarm success!	
21:56:04.815	43507	QOPN/I	[ql_PSM][ql_psm_demo_thread, 363] psm enable!	7 使能自动休眠
21:56:04.819	43516	QUEC/I	quec_pmsource_auto_sleep,324 start auto sleep mode	
21:56:04.822	44795	RTCA/D	RTC intr 0x00001100	

图 9: 调试串口模块开机进入 PSM 的 AP Log

3.2.1.4.2. RTC 唤醒的 AP Log

模块在进入 PSM 时，所有 Log 均停止打印。可以通过是否打印 Log 判断模块是否进入 PSM。同时，所有无法进入 PSM 原因的 AP Log 均会被打印。RTC 将模块从 PSM 唤醒后，之前设置的 PDP 仍处于激活状态，唤醒时获取到 PDP 激活的 IP 地址和第一次开机激活的 IP 地址相同。RTC 唤醒 AP Log 如下：

21:57:07.859	41428	KERN/D	psm: set sleep time 1324120 at 69680	
21:57:08.4			init_thread,889 ### quec init done... pwrup=6 ###	1 开机因为RTC时钟唤醒
21:57:08.4			PSM[q1_psm_demo_thread, 281] PSM WAKEUP!	
21:57:08.4			PSM[q1_psm_demo_thread, 285] info.v4.addr.ip: 10.31.133.105	2 获取到的IP地址与第一次开机激活的IP地址一致
21:57:08.4			RTC[q1_rtc_print_time, 305] 2021-08-03 13:57:07 [TUS]	
21:57:08.449	51358	QOPN/I	[q1_RTC][q1_rtc_set_alarm, 338] q1 rtc set alarm success!	
21:57:08.450	51358	QOPN/I	[q1_RTC][q1_rtc_print_time, 305] 2021-08-03 13:58:07 [TUS]	
21:57:08.451	51360	RTCA/I	RTC set alarm day/7885 hour/13 min/58 sec/7	
21:57:08.478	51671	KERN/D	psm: set sleep time 1323500 at 70305	
21:57:08.478	52271	RTCA/I	RTC store nv length/12230	
21:57:08.481	52294	QOPN/I	[q1_RTC][q1_rtc_enable_alarm, 378] q1 rtc enable alarm success!	
21:57:08.616	53500	RTCA/D	RTC intr 0x00001000	
21:57:08.845	58206	KERN/D	psm: set sleep time 1323100 at 70704	4 尝试重新进入PSM
21:57:08.851	58218	KERN/D	psm: set sleep time 1323100 at 70705	
21:57:08.855	58228	KERN/D	psm: set sleep time 1323100 at 70705	
21:57:08.864	58241	KERN/D	psm: set sleep time 1323100 at 70706	
21:57:08.873	58251	KERN/D	psm: set sleep time 1323100 at 70707	
21:57:08.882	58263	KERN/D	psm: set sleep time 1323100 at 70707	
21:57:08.891	58273	KERN/D	psm: set sleep time 1323100 at 70708	
21:57:08.897	58284	KERN/D	psm: set sleep time 1323090 at 70709	
21:57:10.380	17869	QUEC/I	quec_print_os_pm_state,825 source URT1 not allow sleep	5 无法进入PSM的原因
21:57:10.446	17869	QUEC/I	quec_print_os_pm_state,825 source CP not allow sleep	
21:57:10.756	24019	KERN/D	psm: set sleep time 1321190 at 72617	
21:57:10.760	24031	KERN/D	psm: set sleep time 1321190 at 72618	
21:57:10.765	24041	KERN/D	psm: set sleep time 1321180 at 72619	
21:57:10.769	24052	KERN/D	psm: set sleep time 1321180 at 72619	
21:57:10.774	24070	KERN/D	psm: set sleep time 1321180 at 72620	
21:57:10.778	24080	KERN/D	psm: set sleep time 1321180 at 72621	

图 10: 调试串口 RTC 唤醒的 AP Log

3.2.2. APP PSM 测试

本章节介绍如何修改原始 *psm_demo* 代码，以演示完整的 PSM 数据收发过程。其中，如需抓取相应的休眠唤醒 Log，需配置 EC600U-CN 的调试串口，再通过 *coolwatcher_debughost.exe* 里的 Trace tool 功能查看相关的 AP Log。详情参考第 3.2.1.4 章。

备注

本章节涉及到的 RTC 相关 API，请参考文档 [3]；其他 API，请参考文档 [4]。

3.2.2.1. 测试准备

3.2.2.1.1. 设置 RTC 唤醒

RTC 唤醒可将模块从 PSM 中唤醒，只在模块上电和模块从 PSM 唤醒后设置一次，软件上设置 RTC 唤醒时间为 1 分钟。模块第一次开机时，由于各种原因，在 RTC 唤醒前无法进入 PSM，导致随后进入 PSM 后无法被唤醒。因此，需要注册 RTC 回调，在 RTC 回调中重置 RTC 唤醒时间。设置 RTC 回调的作用如下：

1. 进入 PSM 前，RTC 唤醒计时未结束。待进入 PSM 后，RTC 唤醒源仍存在（此时，由于模块关机的原因，不会进入注册的 RTC 回调）。模块总能被 RTC 唤醒。
2. 进入 PSM 前，RTC 唤醒计时已结束，则进入 RTC 回调重新设置 RTC 唤醒时间，保证在进入 PSM 后 RTC 唤醒源总会存在，避免上述 PSM 无法唤醒的问题。

```

---if(nw_info == NULL || info == NULL)
---{
---    QL_PSMDEMO_LOG("calloc fail!");
---    goto ↓PSM_EXIT;
---}
---if(QL_SLEEP_SUCCESS != ql_rtc_register_cb(ql_rtc_callback))
---{
---    QL_PSMDEMO_LOG("register rtc callback fail!");
---    goto ↓PSM_EXIT;
---}

```

图 11: 注册 RTC 回调

```

284: void ql_rtc_callback(void)
285: {
286:     int ret = 0;
287:     int64_t time_sec = 0;
288:     ql_rtc_time_t tm = {0};
289:     ret = ql_rtc_get_time(&tm);
290:     if(ret != QL_RTC_SUCCESS)
291:     {
292:         QL_PSMDEMO_LOG("get time err");
293:         return;
294:     }
295:     ql_rtc_print_time(tm);
296:     ql_rtc_conv_sec_time(&time_sec, &tm);
297:     time_sec += 10;
298:     ql_sec_conv_rtc_time(&time_sec, &tm);
299:     ret = ql_rtc_set_alarm(&tm);
300:     if(ret != QL_RTC_SUCCESS)
301:     {
302:         QL_PSMDEMO_LOG("set alarm err");
303:         return;
304:     }
305:     ql_rtc_print_time(tm);
306:     //Enable RTC alarm
307:     ret = ql_rtc_enable_alarm(1);
308:     if(ret != QL_RTC_SUCCESS)
309:     {
310:         QL_PSMDEMO_LOG("enable alarm err");
311:         return;
312:     }
313: }
314: /*end ql_rtc_callback*/
315:

```

图 12: RTC 回调中重置 RTC 唤醒

3.2.2.1.2. 设置串口无数据交互时进入 PSM 的时间

相关 API 的头文件为 `ql_uart.h`，位于 SDK 包中的 `components\ql-kernel\inc` 目录下。

```

...../*****
.....//TCP
.....ql_send_tcp_packet(info);
...../*****
.....//set alarm
.....if(!ql_psm_set_alarm())
.....{
.....    QL_PSMDEMO_LOG("alarm set fail!");
.....    goto ↓PSM_EXIT;
.....}
.....QL_PSMDEMO_LOG("psm enable!");
.....ql_uart_set_sleep_delay_time(1);
.....ql_autosleepex_enable(true);

```




图 13: 设置串口无数据交互时进入 PSM 的时间

3.2.2.1.3. 使能增强型休眠模式

将代码中 `ql_autosleep_enable(true)` 替换为 RRC 快速释放接口：

```

#define QL_PSMDEMO_NO_DATA_DEFAULT      1          //RRC 释放 1 秒
#define QL_PSMDEMO_PUNISH_TIME_DEFAULT  30
ql_autosleepex_enable(QL_ALLOW_SLEEP,QL_PSMDEMO_NO_DATA_DEFAULT,QL_PSMDEMO_
PUNISH_TIME_DEFAULT);

```

```

.....ql_send_tcp_packet(info);
...../*****
.....//set alarm
.....if(!ql_psm_set_alarm())
.....{
.....    QL_PSMDEMO_LOG("alarm set fail!");
.....    goto ↓PSM_EXIT;
.....}
.....QL_PSMDEMO_LOG("psm enable!");
.....ql_uart_set_sleep_delay_time(1);
.....ql_autosleepex_enable(QL_ALLOW_SLEEP,QL_PSMDEMO_NO_DATA_DEFAULT,QL_PSMDEMO_PUNISH_TIME_DEFAULT);
.....}«end if QL_SLEEP_SUCCESS==ql...»
.....else
.....{

```

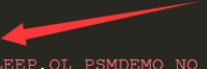


图 14: 使能增强型休眠模式

3.2.2.1.4. 新增 TCP 数据发送

在原有 API 的示例上，加入 TCP 发包程序。详情请参考 `socket_demo.c`（SDK 包路径 `\components\ql-application\socket`）。其中，`ql_send_tcp_packet()`为自定义测试函数。

```

...
}
//TCP
ql_send_tcp_packet(info);
//set alarm
if(!ql_psm_set_alarm())
{
    QL_PSMDEMO_LOG("alarm set fail!");
    goto PSM_EXIT;
}
QL_PSMDEMO_LOG("psm enable!");
ql_autosleepex_enable(QL_ALLOW_SLEEP, QL_PSMDEMO_NO_DATA_DEFAULT, QL_PSMDEMO_PUNISH_TIME_DEFAULT);
}
else
{
    QL_PSMDEMO_LOG("psm enable fail!");
    goto PSM_EXIT;
}
}
}
PSM_EXIT:

```

1 第一次开机发送TCP包

2 使能增强型休眠模式

图 15: PSM 开机添加 TCP 数据发送

```

if(0==ql_get_powerup_reason(&source) && QL_PWRUP_PSM_WAKEUP==source)
{
    QL_PSMDEMO_LOG("PSM WAKEUP!");
    ret=ql_get_data_call_info(QL_PSMDEMO_SIM_ID, profile_idx, info);
    if(QL_DATACALL_SUCCESS==ret && info->v4.state)
    {
        QL_PSMDEMO_LOG("info.v4.addr.ip: %s", ip4addr_ntoa(&(info->v4.addr.ip)));
    }
    else{
        QL_PSMDEMO_LOG("ql_get_data_call_info fail!");
        goto PSM_EXIT;
    }
}
//TCP
ql_send_tcp_packet(info);
//Reset alarm for PSM wake up
if(!ql_psm_set_alarm())
{
    QL_PSMDEMO_LOG("alarm set fail!");
    goto PSM_EXIT;
}
//ql_autosleepex_enable(QL_ALLOW_SLEEP, QL_PSMDEMO_NO_DATA_DEFAULT, QL_PSMDEMO_PUNISH_TIME_DEFAULT);
}
}
}

```

1 PSM唤醒后重新发送TCP包

图 16: PSM 唤醒添加 TCP 数据发送

3.2.2.2. 测试案例

3.2.2.2.1. 测试示例

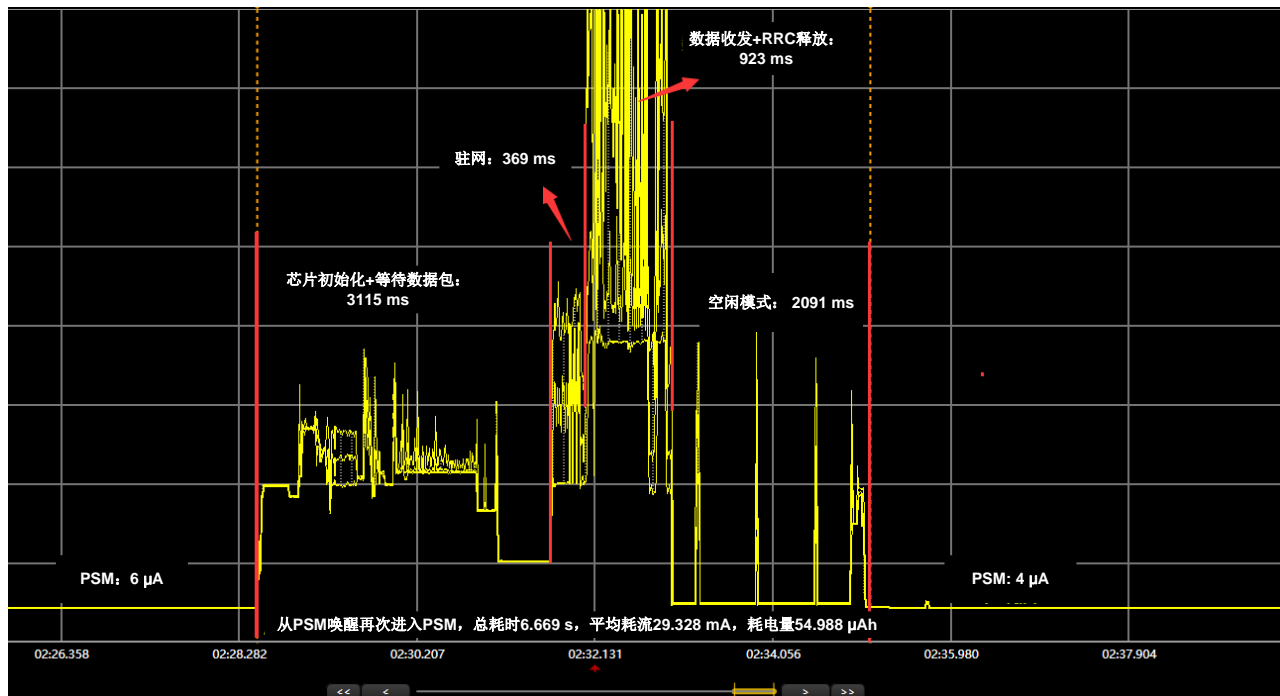


图 17: EC600U-CN PSM 唤醒平均耗流

3.2.2.2.2. 测试数据

表 3: EC600U-CN PSM 唤醒平均耗流数据

模式	持续时间 (ms)	平均耗流 (mA)	耗电量 (μAh)
PSM	-	0.006	-
芯片初始化+等待数据包	3115	31.077	26.89
驻网	369	52.979	5.435
数据收发+RRC 释放	923	77.112	19.776
空闲模式	2091	4.208	2.524
PSM	-	0.004	-
数据汇总	6669	29.328	54.988

3.2.2.3. 测试结论

模块从 PSM 唤醒到再次进入 PSM 总共经过 4 个阶段，分别是：模块的初始化，模块驻网，模块发送 TCP 数据及模块再次进入 PSM 的 T3324 等待时间。各阶段耗流情况如下：

- 模块的初始化持续时间为 3115 ms，平均耗流为 31.077 mA，耗电量为 26.890 μ Ah；
- 模块驻网持续时间为 369 ms，平均耗流为 52.979 mA，耗电量为 5.435 μ Ah；
- 模块发送 TCP 数据持续时间为 923 ms，平均耗流为 77.112 mA，耗电量为 19.766 μ Ah；
- 模块进入 PSM 的 T3324 等待持续时间为 2091 ms，平均耗流为 4.208 mA，耗电量为 2.524 μ Ah。当模块再次进入 PSM 后，耗流维持在 6 μ A 以内，比较稳定。

模块从 PSM 唤醒到再次进入 PSM 总耗时为 6669 ms，平均耗流为 29.328 mA，耗电量为 54.988 μ Ah。

3.2.2.3. 注意事项

启用增强型休眠模式后，用户发起数据交互时，会给基站产生额外的信令负担，所以可能存在终端被网络侧惩罚不能上线的情况。但目前基站对这些信令交互没有明确的限制。

启用 PSM 后，模块会上报用户设置的 T3412 定时器和 T3324 定时器的值作为参考，当运营商支持 PSM，由网络决定是否采用上报的定时器值，最终下发到模块。模块无法预知网络行为，若网络下发的 T3412 与 T3324 之间的差值小于 5 分钟，则模块不会进入 PSM。也就是说，即使模块上报的定时器值满足上述条件，但若网络下发值不满足上述条件，模块也不会进入 PSM。

4 附录 参考文档及术语缩写

表 4：参考文档

文档名称
[1] Quectel_ECx00U&EGx00U 系列_QuecOpen_快速开发指导
[2] Quectel_ECx00U&EGx00U 系列_QuecOpen_Log_抓取指导
[3] Quectel_ECx00U&EGx00U 系列_QuecOpen_RTC_API_参考手册
[4] Quectel_ECx00U&EGx00U 系列_QuecOpen_低功耗 API_参考手册

表 5：术语缩写

缩写	英文全称	中文全称
AP	Application Processor	应用程序处理器
API	Application Programming Interface	应用程序编程接口
APP	Application	应用程序
CP	Coprocessor	协同处理器
EVB	Evaluation Board	评估板
IoT	Internet of Things	物联网
LTE	Long-Term Evolution	长期演进
PSM	Power Saving Mode	省电模式
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

TAU	Tracking Area Update	跟踪区更新
USB	Universal Serial Bus	通用串行总线