

## OCPU FAQ

**FAQ** 就是 Frequently Asked Questions ( 常见问题 ) , 本文档中记录一些客户经常遇到的问题, 后续客户遇到问题, 推荐用户先看看 **FAQ**, 看自己的问题是否符合 **FAQ** 中的情况, 如果符合的话 **FAQ** 中是有相应解答的, 这样就不用人工服务了, 节省了大家的时间。

## SDK 初次使用部分

### 1、SDK 版本

#### 1.1、客户从哪里获取我们 SDK 版本包和相关资料?

可以从我司 FAE 或者代理商获取到 SDK 的版本包和相关资料, [亦可发邮件到 support@quectel.com](mailto:support@quectel.com) 获取。

#### 1.2、客户获取 SDK 版本包时需要注意哪些?

- 1) 需要了解自己的需求, 根据自己的需求选择软件版本。是否需要 BT 功能, TTS 功能, UFS 需要多大, 支持 SD 卡等。
- 2) 提供模块的软件版本号 (ATI 可查询) 给我司, SDK 版本包和软件版本包是需要对应的。

#### 1.3、模块型号 (按照软件版本分)

我司模块分为标准版本、纯 open cpu 版本、二合一版本

**标准模块:** 只支持标准 AT 命令, 客户外接 MCU 通过串口向模块发送 AT 指令, 实现信息交互。

eg: M35F、M95F、M26TTS

**纯 OCPU 版本:** 模块不支持标准 AT, 客户可以在模块上进行二次开发, 如果需要发送 AT 命令, 也是通过 OCPU 传给模块底层处理。

eg: M50F\_OCPU、M10F\_OCPU

**二合一版本:** 同时支持标准模块和 OPEN CPU.

eg: M26BT、M26BPU、M66、M66DS、MC20、MC20FC、MC20TTS、MC20E、MC20U、MC60、MC60E

#### 特别提醒:

- 1、不同软件版本对应不同的 SDK, 不可以混用。
- 2、CA 和 CB 版本的模块软件版本不能通用, 但 SDK 版本是一样的。

3、不同版本支持的功能项不一致，客户需要按照自己项目需求选型模块

## 2、SDK 编译、基本配置

### 2.1、GCC 编译器的版本号？

arm-none-eabi-gcc-4.7.2

### 2.2、是否支持 win10 下安装，如何安装？

右击 exe 软件--->弹出属性设置--->兼容性设置--->选择 win7 模式--->应用,确定--->开始安装

(Right click of the exe ---- > jump to attribute setting --- > jump to compatible setting --- >run as compatible mode (win7 mode) ---- >apply the setting and run the setup--->Start installation software)

### 2.3、32 位和 64 位操作系统下，\make\gcc\gcc\_makefile 中的路径有何不同？

#### 2.3.1 路径不对报错提示如下：

```
ld\gcc\obj\custom\fota/src\fota_http.o build\gcc\obj\custom\fota/src\fota_http_c
ode.o build\gcc\obj\custom\fota/src\fota_main.o build\gcc/build.log build\gcc/AP
PGS3MDM32A01.map build\gcc/APPGS3MDM32A01.bin build\gcc/APPGS3MDM32A01.elf, ...>
failed.
make (e=2): 系统找不到指定的文件。
make: *** [clean] Error 2
```

#### 2.3.2 路径更改：

1) 如果选择默认条件成功安装的 GCC,那么路径如下：

**32 位操作系统：**

GCC\_INSTALL\_PATH=C:\Program Files\CodeSourcery\Sourcery\_CodeBench\_Lite\_for\_ARM\_EABI

**64 位操作系统：((X86)之前有空格)**

GCC\_INSTALL\_PATH=C:\Program Files (x86)\CodeSourcery\Sourcery\_CodeBench\_Lite\_for\_ARM\_EABI

2) 如果是自定义的路径，根据安装时实际选择路径更改。

### 2.4、编译出错哪里查看错误信息？

客户查看 log 文件，路径：build\gcc\build.log

### 2.5、编译其他的 example 在哪里更改宏

\make\gcc\gcc\_makefile

```

#-----
# Configure GCC installation path, and GCC version.
# To execute "arm-none-eabi-gcc -v" in command line can get the current gcc version
#-----
GCC_INSTALL_PATH=C:\Program Files (x86)\CodeSourcery\Sourcery_CodeBench_Lite_for_ARM_EABI
GCC_VERSION=4.7.2

#-----
#use the following path for 32-bit operating system
#-----
#GCC_INSTALL_PATH=C:\Program Files\CodeSourcery\Sourcery_CodeBench_Lite_for_ARM_EABI
C_PREDEF=-D __CUSTOMER_CODE__
#-----

```

## 2.6、如何关闭 OCPU，使模块在标准模式运行。

### 2.6.1、重新烧录软件版本(FW)

如果没有软件版本包，可以联系我司 FAE。

### 2.6.2、烧录 APP bin

在\custom\config\custom\_sys\_cfg.c 中更改为 APP\_DISABLE,重新编译，烧录。

```

00044: /******
00045: /* Configure the enable of application. */
00046: /* The possible values are 'APP_DISABLE' and 'APP_ENABLE', and the */
00047: /* default value is 'APP_ENABLE'. */
00048: /* APP_ENABLE: the application will work after downloading into module*/
00049: /* APP_DISABLE: the application will not work after downloading.In this*/
00050: /* case, the module still work as a STANDARD module. */
00051: /******
00052: static const ST_AppEnable appEnableCfg = {
00053:     APP_ENABLE
00054: };
00055:

```

## 2.7、设置 debug 口模式，抓取 catcher log。

在\custom\config\custom\_sys\_cfg.c 中

```

00088: /******
00089: /* Define the working mode of serial debug port (UART2). */
00090: /* */
00091: /* The serial debug port (UART2) may work as a common serial port, as */
00092: /* well as work as a special debug port that can debug some issues */
00093: /* underlay application. */
00094: /* Under the special debug mode, please refer to the document */
00095: /* "Catcher_Operation_UGD" for the usage of debug port */
00096: /******
00097: static const ST_DebugPortCfg debugPortCfg = {
00098:     //BASIC_MODE // Set the serial debug port (UART2) to a common serial port
00099:     ADVANCE_MODE // Set the serial debug port (UART2) to a special debug port
00100: };
00101:

```

**注：**

- 1、当 debug 口设置成 ADVANCE\_MODE 时，APP 中不能通过 QI\_UART\_Open 打开 debug 口输出文本 log。
- 2、可以通过接口 QI\_Debug\_Trace 把文本 log 发送到 catcher 中,此时 MOD 选择 QL。（如果 MOD 选择过多,这种方式可能会丢失 log）

MOD_QL	OpenCPU_GS3_SDK_V1.4
MOD_QL	Ql_SetPortOwner,phyport=2,id=0,mod_id=175,mod=175,TST_PORT=2
MOD_QL	[OCPU]Fail to Ql_SetPortOwner(), ret=-7
MOD_QL	Fail to open serial port[11], ret=-7
MOD_QLTASK1	MOD_QL MSG_ID_MULTITASK_EVENT_IND

## 2.8、配置看门狗引脚

在\custom\config\custom\_sys\_cfg.c 中

```
00074: /******
00075:  /* Define the GPIO pin for external watchdog.
00076:  /* NOTES:
00077:  /* Customer may specify two GPIOs if needed.
00078:  /******
00079: static const ST_ExtWatchdogCfg wtdCfg = {
00080:     PINNAME_PCM_OUT, // Specify a pin which connects to the external watchdog
00081:     PINNAME_END      // Specify another pin for watchdog if needed
00082: };
```

我司模块可以设置两个看门狗引脚，给其中任何一个引脚喂狗，都可以完成喂狗动作。一般我们选择一个引脚来喂狗，另一个引脚设置成 **PINNAME\_END**。如果客户实际应用没有看门狗，请把两个引脚都配置成 **PINNAME\_END**。

## 2.9、配置 PWRKEY 引脚工作模式(需后续工作)

在\custom\config\custom\_sys\_cfg.c 中

```
00056: static const ST_PowerKeyCfg pwrkeyCfg = {
00057:     TRUE, // working mode for power-on on PWRKEY pin
00058:     /*
00059:     Module automatically powers on when feeding a low level to POWER_KEY pin.
00060:
00061:     When set to FALSE, the callback that Ql_PwrKey_Register registers will be triggered. Application must
00062:     call Ql_LockPower () to lock power supply, or module will lose power when the level of PWRKEY pin goes high.
00063:     */
00064:     TRUE, // working mode for power-off on PWRKEY pin
00065:     /*
00066:     Module automatically powers off when feeding a low level to POWER_KEY pin.
00067:
00068:     When set to FALSE, the callback that Ql_PwrKey_Register registers will be triggered.
00069:     Application may do post processing before switches off the module.
00070:     */
00071: };
00072: ;
```

**pwrkeyCfg 作用:**

当 **power-off** 模式设置成 FALSE 后，每次操作模块的 PWRKEY 引脚都会上报一次 callback，并上报电平信息，客户可以自己来决定 PWRKEY 的机制，而不再是我们原有的拉低超过 1000ms 模块关机。

eg:

1. 客户收到拉低 PWRKEY 的 callback，先把内部工作处理完，再自己 POWER DOWN.

**注:**

当 **power-on** 模式设置成 FALSE 后，只会上报一次拉低模块开机的 callback，并不会如注释所述，拉高会关机，Ql\_LockPower()也未见起作用，**后期需要继续调查**。

**例子参考**

**example\_pwrkey** 文件夹

## 2.10、提前配置 GPIO

在\custom\config\ custom\_gpio\_cfg.h 中

```
00055:
00056: #if 0 // If needed, config GPIOs here
00057: GPIO_ITEM(PINNAME_NETLIGHT, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00058: GPIO_ITEM(PINNAME_DTR, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00059: GPIO_ITEM(PINNAME_RI, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00060: GPIO_ITEM(PINNAME_DCD, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00061: GPIO_ITEM(PINNAME_CTS, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00062: GPIO_ITEM(PINNAME_RTS, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00063: GPIO_ITEM(PINNAME_PCM_CLK, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00064: GPIO_ITEM(PINNAME_PCM_SYNC, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00065: GPIO_ITEM(PINNAME_PCM_IN, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00066: GPIO_ITEM(PINNAME_PCM_OUT, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00067: GPIO_ITEM(PINNAME_SD_CMD, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00068: GPIO_ITEM(PINNAME_SD_CLK, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00069: GPIO_ITEM(PINNAME_SD_DATA, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00070: GPIO_ITEM(PINNAME_SIM2_DATA, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00071: GPIO_ITEM(PINNAME_SIM2_RST, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00072: GPIO_ITEM(PINNAME_SIM2_CLK, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00073: GPIO_ITEM(PINNAME_GPIO0, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00074: GPIO_ITEM(PINNAME_GPIO1, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00075: GPIO_ITEM(PINNAME_GPIO2, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00076: GPIO_ITEM(PINNAME_GPIO3, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00077: GPIO_ITEM(PINNAME_GPIO4, PINDIRECTION_OUT, PINLEVEL_LOW, PINPULLSEL_PULLUP)
00078:
00079: #endif
```

**作用:**

从模块开机到 APP 跑起来, 大约需要 3s, 时间会比较久, 如果客户想在内核期间就配置 gpio 的电平状态, 可以通过这个配置表来实现。

**模块 GPIO 配置共有如下四个阶段:**

**1) 硬件初始化阶段**

3.11.4. AT命令

3.12. SPI和I2C接口

3.12.1. SPI接口

3.12.2. I2C接口

3.13. (U)SIM卡接口

3.14. SD卡接口

3.15. ADC模数转换

3.16. 外部中断

3.17. PWM

3.18. GPIO

3.19. 4线串口

3.20. 网络状态指示

3.21. EASYTM Autonomous AG

3.22. EPOTM Offline AGPS技术

3.23. 秒定

3.24. Multi-tone AIC

3.25. LOCUS技术

3.26. PPS VS. NMEA (1PPS)功能

4 天线接口

4.1. GSM天线接口

4.1.1. 参考设计

4.1.2. RF输出功率

4.1.3. RF接收灵敏度

4.1.4. 工作频率

QUECTEL

MC20-OpenCPU 系列硬件设计手册

表 35: GPIO 引脚列表

引脚号	引脚名	Mode	Reset		Output Driving
			I/O	PU/PD	
7	SD_CMD	Mode 2	I	PD	4mA
8	SD_CLK	Mode 2	I	PD	4mA
9	SD_DATA	Mode 2	I	PD	4mA

参考《MC20-OpenCPU 系列硬件设计手册》3.18 节 GPIO

**2) 内核 GPIO 初始化**

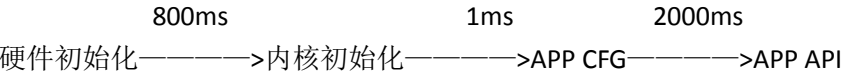
**3) APP 的 CFG 配置**

SDK 中的\custom\config\ custom\_gpio\_cfg.h

**4) APP 的 API 接口配置**

具体客户代码中的 GPIO 配置

**大致时间**



**2.11、增加一个 task**

在\custom\config\ custom\_task\_cfg.h 中

```

00050: /*-----*/
00051: |      Task Entry Function | Task Id Name      | Task Stack Size (Bytes) | Default Value1 | Default Value2
00052: |-----|
00053: TASK_ITEM(proc_main_task,    main_task_id,    10*1024, DEFAULT_VALUE1, DEFAULT_VALUE2)
00054: TASK_ITEM(proc_reserved1,    reserved1_id,    5*1024, DEFAULT_VALUE1, DEFAULT_VALUE2)
00055: TASK_ITEM(proc_reserved2,    reserved2_id,    5*1024, DEFAULT_VALUE1, DEFAULT_VALUE2)
00056: TASK_ITEM(proc_subtask1,     subtask1_id,     5*1024, DEFAULT_VALUE1, DEFAULT_VALUE2)
00057:

```

#### 注意:

- 1) 默认的三个任务 (proc\_main\_task, proc\_reserved1, proc\_reserved2), 建议客户不更改, 顺序也不移动, 因为我们用户库(app\_start.lib)中会用到这三个任务, 如果更改会影响代码运行。
- 2) 如果客户需要新增 task, 可以在后面按照格式增加。
- 3) 如果客户新增的 task 中会用到 file system, 则 stack size 必须设置最少 5K, 防止栈溢出。

## 3、APP 下载

### 3.1、下载工具

我司 2G 模块的下载工具统一使用 QFLASH。目前推荐版本 QFLASH4.0 。

### 3.2、QFlash 下载流程

以我司开发板为例:

- 1) PWRKEY 引脚一直拉低(S201 拨到 ON)
- 2) 打开 QFLASH, 加载下载文件, 点击 START
- 3) 开发板上电(S202 拨到 ON)

### 3.3、QFlash 工具参数配置

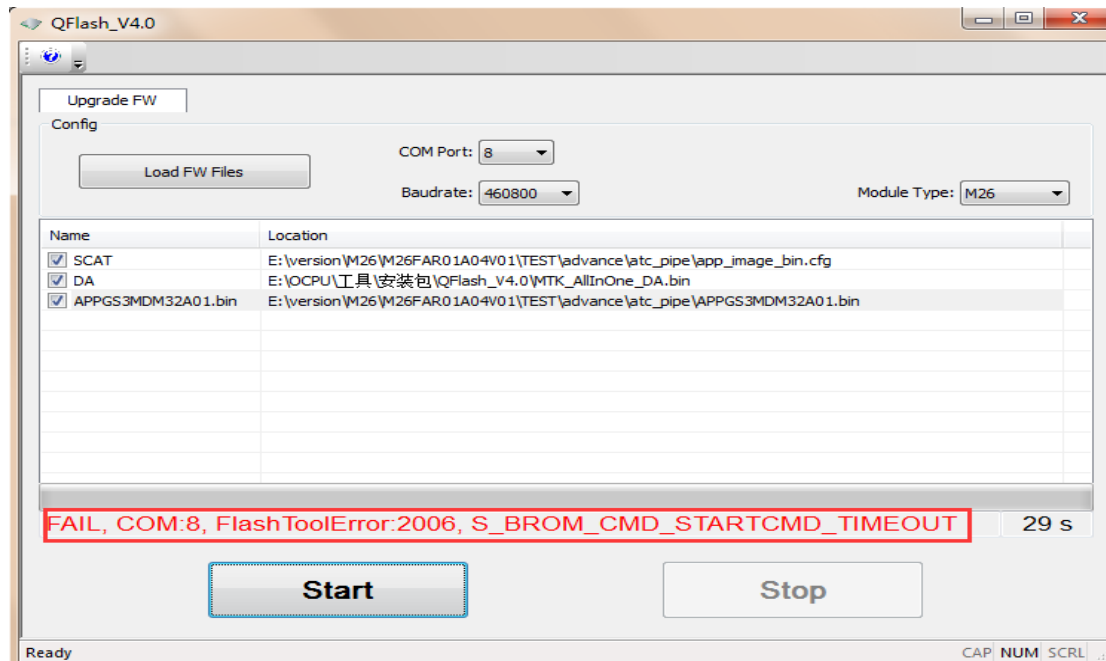


#### 波特率

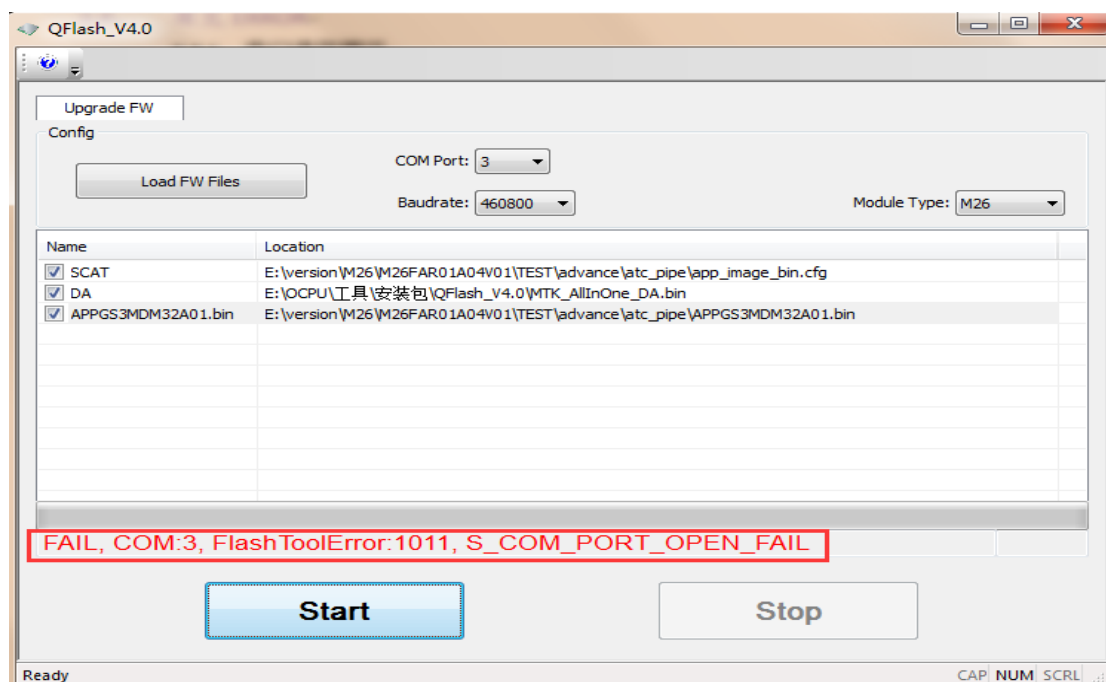
模块支持 9600-460800，推荐 460800，另客户选择波特率时，也需要考虑自身设计硬件电路支持波特率上限。

## 3.4、常见 ERROR

### 3.4.1、串口选择错误

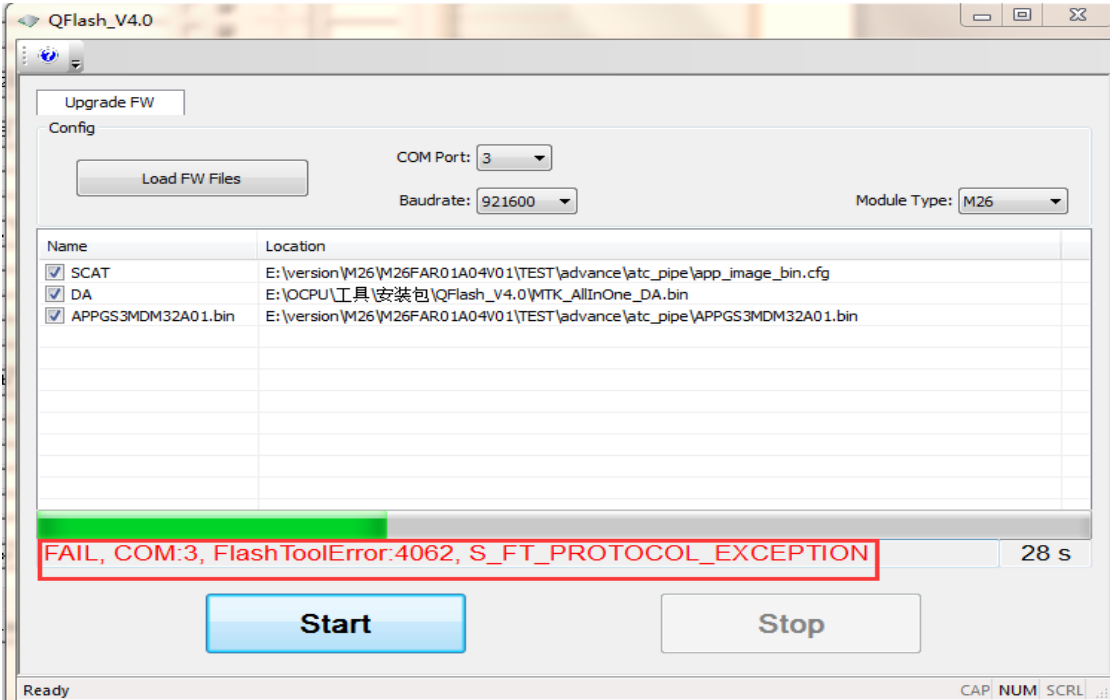


### 3.4.2、串口被占用

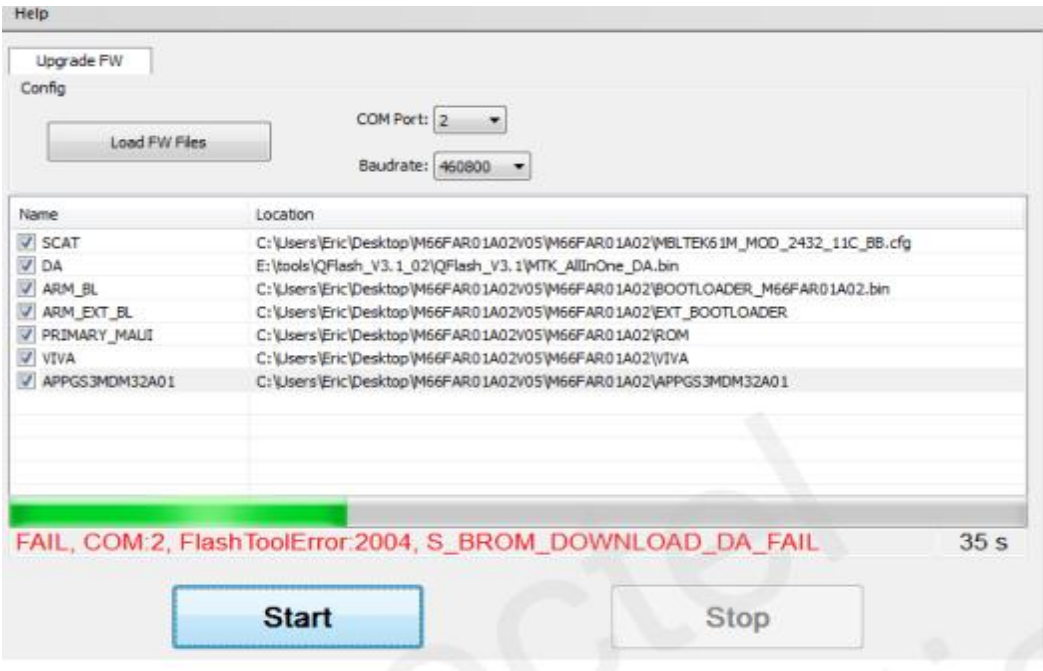




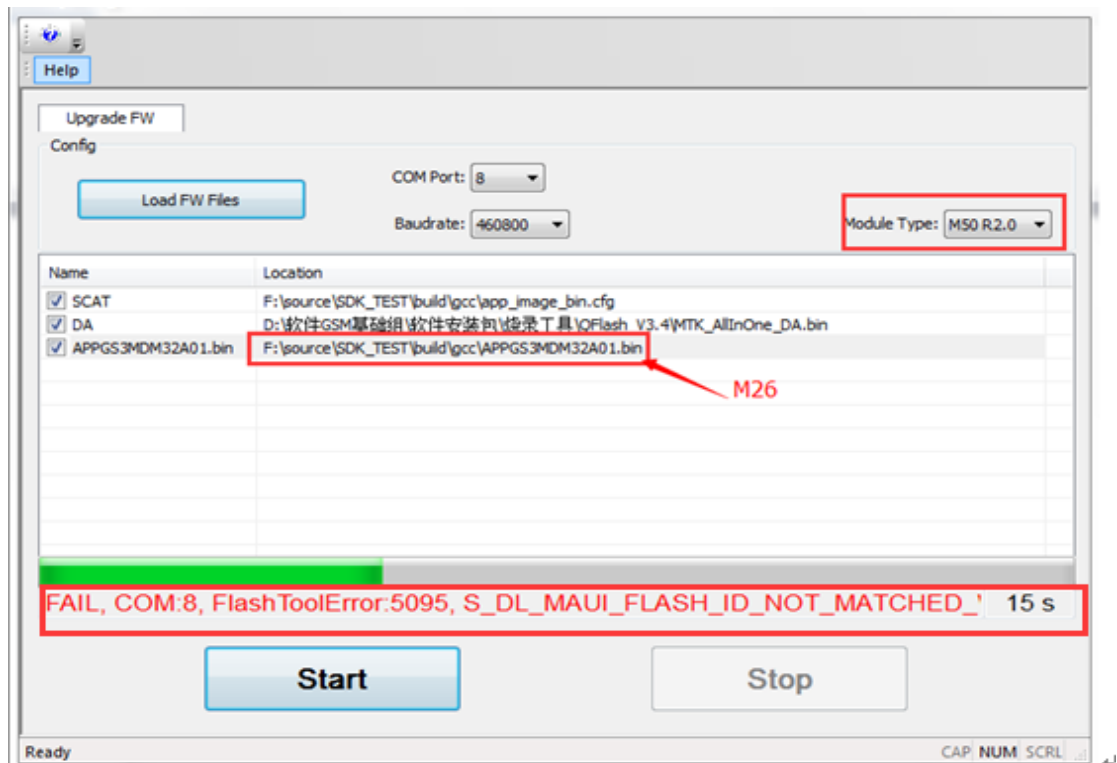
3.4.3、波特率过高



3.4.4、VBAT 供电不稳



### 3.4.5、选错 module type



Module type M50 实际模块 M26

Module type M26 烧写的程序不是对应的 M26 也是这种情况

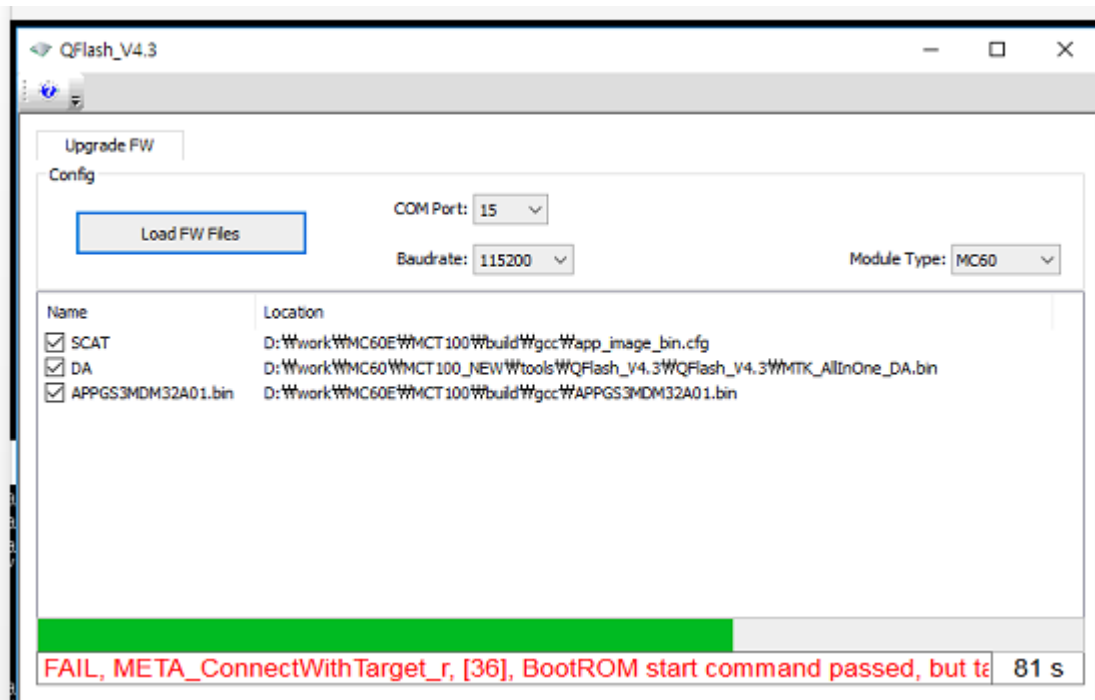
### 3.4.6、固件与 SDK 版本不匹配（烧录正常，运行不正常）

一般可以烧录成功，但是 SDK 不能正常运行，或者还是运行以前的 SDK 代码。

eg1: MC20 的固件烧录了 MC20E 的 SDK 包，可以烧录成功，但是 SDK 跑到 ril is ready,就会卡住。

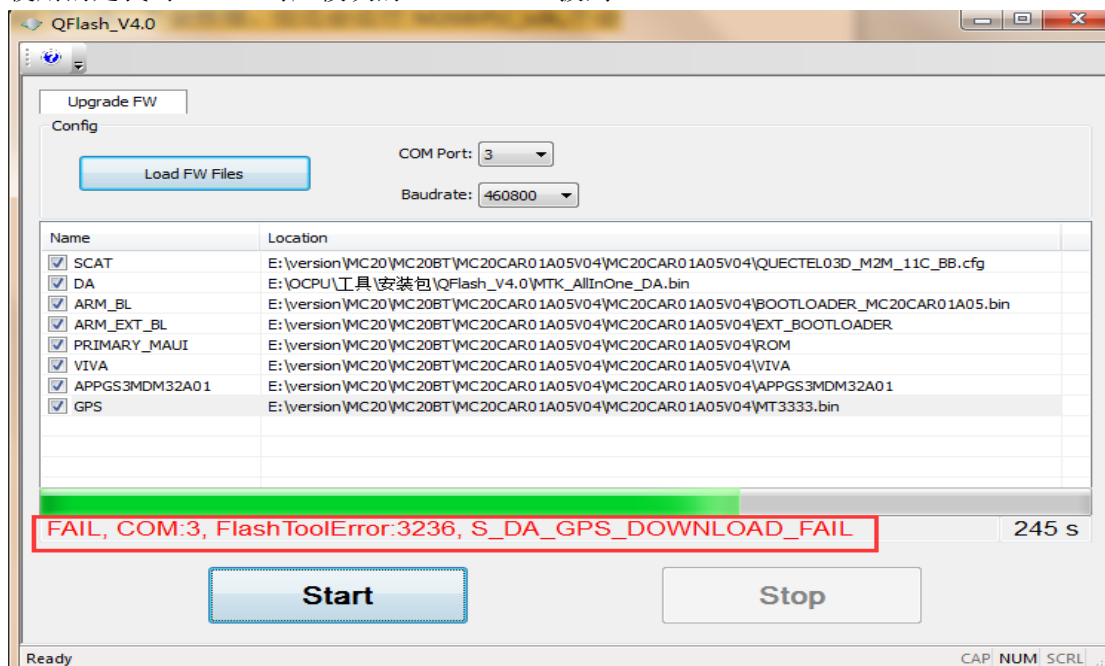
eg2: M26BPU 的内核烧 M26BT 的 sdk，不会报错，但还会运行 M26BPU\_sdk,不会擦除。

eg3: 客户用 MC60 的 APP BIN 烧录 MC60E，然后再用 MC60E 的 APP BIN ,烧录发现不会成功，经重新烧录 MC60E 内核,问题解决，报错信息如下：

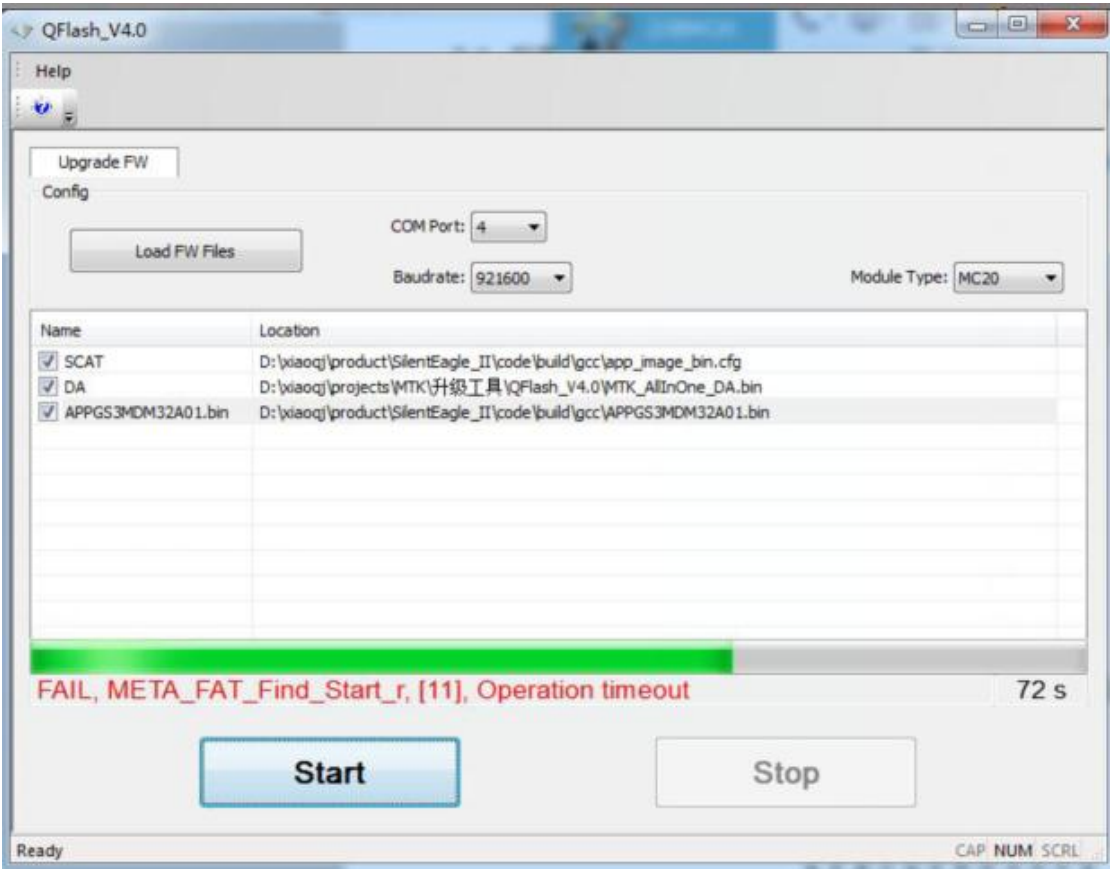


### 3.4.7、GPS 的开关未打开

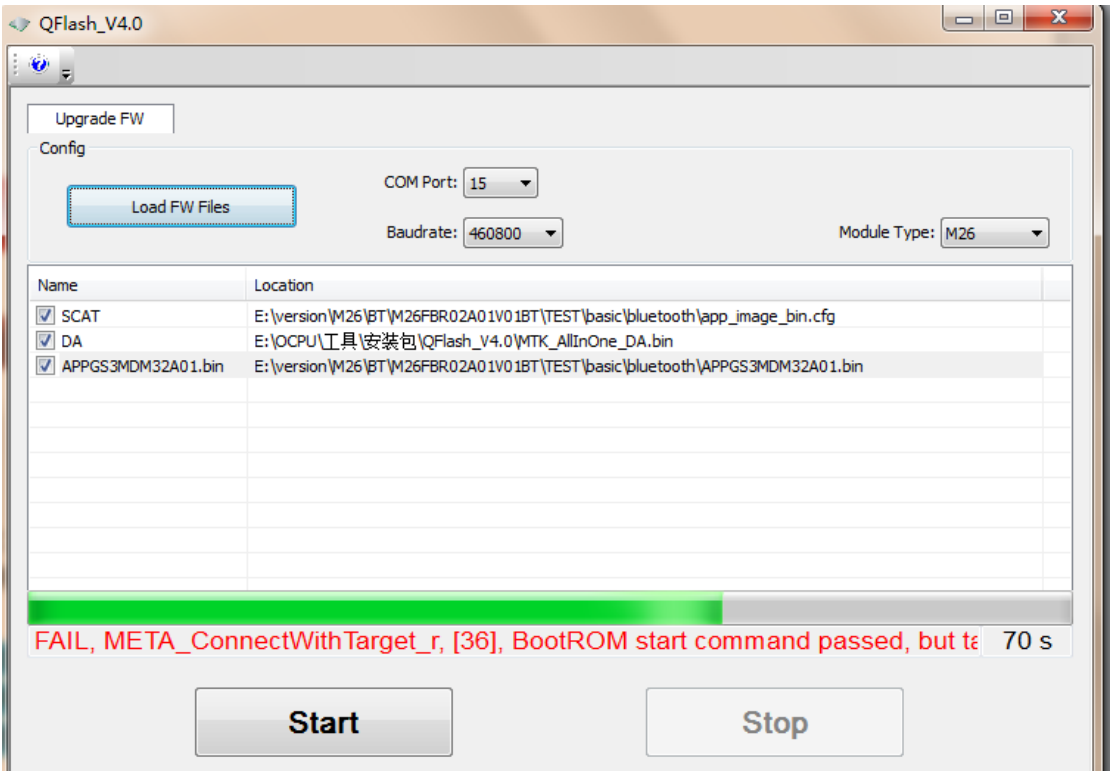
GNSS\_TX,GNSS\_RX 需要和 AUX\_RX,AUX\_TX 连接起来，这样才能跟新 GPS 的 bin 文件。如果使用的是我司 TE-A，对应模块的 S101, S102 拨到 ON



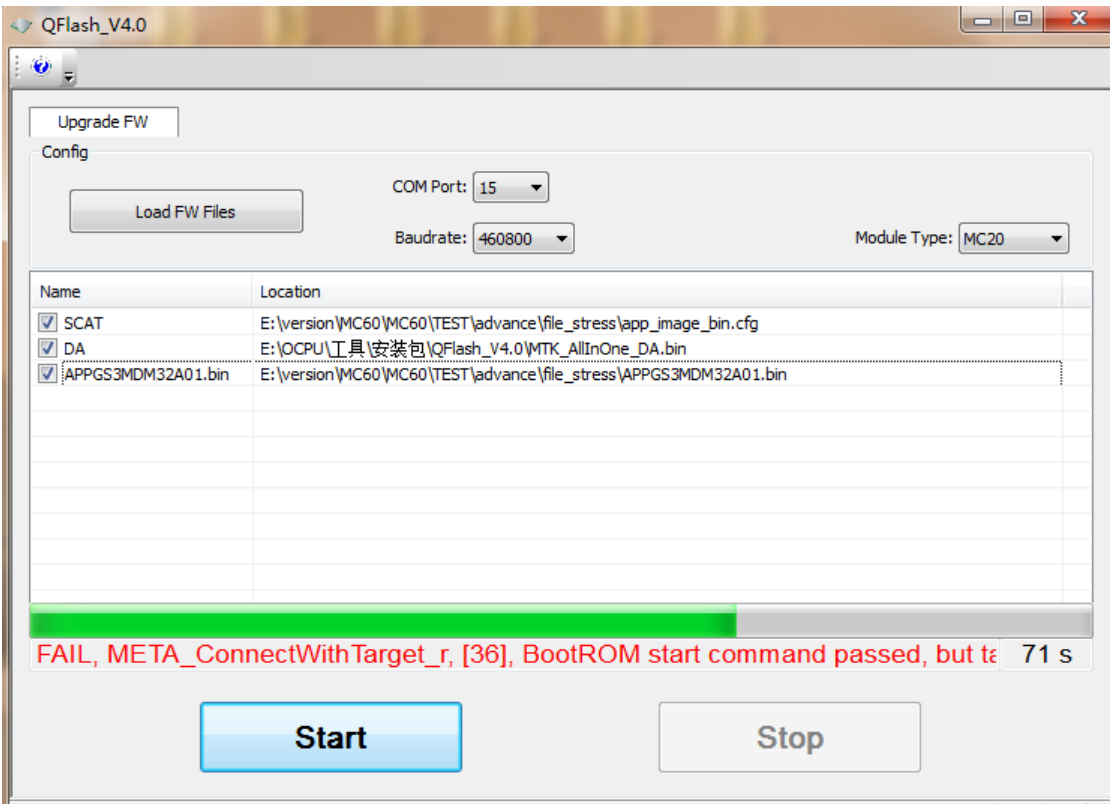
3.4.8、数据线不稳定或者数据线与 PC 驱动不匹配



3.4.9、固件版本和 SDK 包不匹配（烧录错误）



M26TTS 的固件烧录 BT 的 SDK



MC60E 烧录 MC20 的

# 常用工具介绍

## 1、抓包工具(catcher)

### 1.1、工具作用

Catcher 是一个在 PC 端的工具，当模块出现异常或者 DUMP 时，通过应用层的 APP LOG 无法定位问题时，就需要客户抓取相应的 catcher log 帮助分析。通常以下几类问题需要抓取 catcher log.

- 1) 模块异常重启/死机
- 2) 网络相关异常，找网、注网、TCP 通讯等
- 3) 定位内核问题/调试内核代码

### 1.2、工具操作步骤

参考文档《Catcher\_Operation\_UGD\_V1.0.pdf》、《how to carther a log.pdf》

### 1.3、MOD 选择

SIM 问题: MOD\_SIM

网络问题: MOD\_APPTCPIP、MOD\_RLC、MOD\_LLC、MOD\_SOC、MOD\_TCPIP

GPS 问题: MOD\_GPS

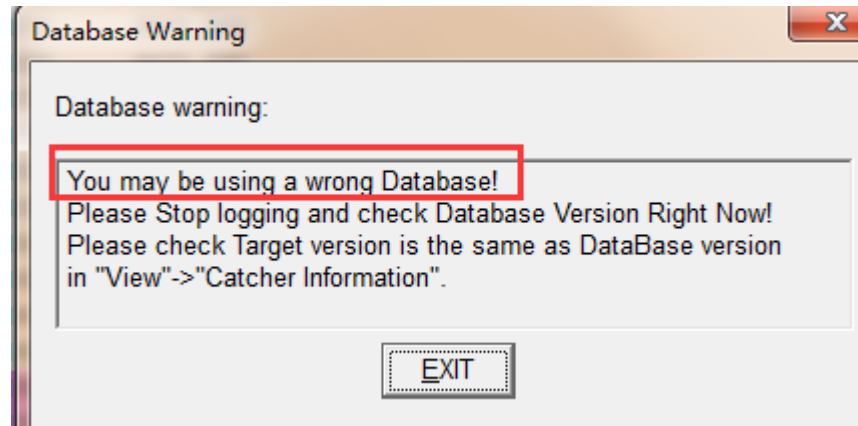
Opencpu 问题: 选择 MOD\_QL、MOD\_QLTASK1....

不清楚该选哪些: 勾(2G)Field Trial

**注:** 对于客户程序有几个 task，把相应 TASK MOD 全勾上。

## 1.3、异常错误

### 1.3.1、选择的 database 与模块实际烧录的版本不符



解决办法:

在下面 log 中找到模块实际烧录的版本号，然后根据这个版本号选择对应的 database。

Sys Trace				
Index	Frame #	Time	Local Time	Message
0		50615	15:04:29:381 2017/11/03	[Test Mode Info] Current Test Mode: 0, Special Setting: 0x0
1		50615	15:04:29:382 2017/11/03	Chip SW second version: 01
2		50615	15:04:29:382 2017/11/03	Bin file SW second version: 01
3		50615	15:04:29:382 2017/11/03	Product: 11CW1418IOTMP QUECTEL03D_M2M_11C Version: MC20CAR01A08 BuildTime: 2017/08/28 10:41
4		50615	15:04:29:382 2017/11/03	SWLA is not running now
5		50615	15:04:29:382 2017/11/03	[Info] Catcher connects

### 1.3.2、没有 catcher log 信息吐出

- 1、如果是 opencpu 方案，确认 debug 口（UART2）是否配置成 ADVANCE MODE
- 2、模块串口的 RX TX GND 都需要引出
- 3、硬件接触不良，可以先使用我们 main.c，把串口改到 UART2 吐信息到 QCOM，看看收发是否成功

## 2、串口工具(QCOM)

### 2.1、工具作用

串口调试工具，作为上位机软件，客户可以通过 QCOM 来和模块进行数据交互。抓取 APP 层的应用 log，辅助分析问题。

## 2.2、异常错误

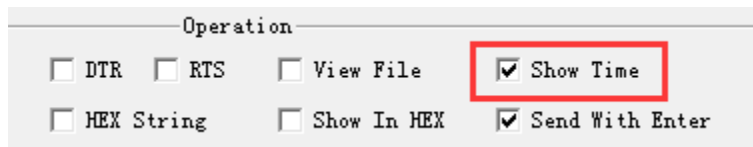
### 2.2.1、串口选择错误

### 2.2.2、串口被占用

2.2.2、没有勾选回车换行，导致 AT 不通。

## 2.3 、注意

1) 建议客户在使用 QCOM 工具时，勾选如下对话框，使 log 增加时间戳，便于分析问题。



2) 如果需要跑压力测试，可以把 log 保存在本地 txt 文档中，避免 log 丢失。

## 3、FOTA 升级包制作工具(OpenCPU\_FOTA\_Package\_Tool)

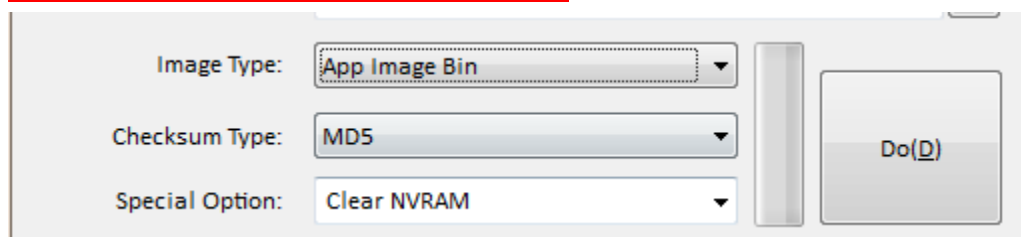
### 3.1、工具作用

在 SDK 编译生成 BIN 文件的头和尾部增加一些数据位(CRC 校验，文件长度等)，制作生成一个 FOTA\_BIN 文件，为后面进行 fota 升级做准备。

### 3.2、工具操作步骤

参考文档《Quectel\_OpenCPU\_FOTA\_Application\_Note\_V1.0.pdf》

注：工具的配置选择默认就可，不需要更改。





### 3.3、异常错误

3.3.1、客户发现我们的 FOTA TOOLS 工具，在很多系统(win7,win8,win10 32 位&&64 位)不能使用。

根据本地复现的错误事件提示，发现少了 X86 库和 VC 库集。在 32 位系统上安装 X86 库即可，在 64 位系统上两个都需要安装。

## SYSTEM 部分

### 1、 RIL

#### 1.1、RIL 还未初始化成功就操作 AT 命令返回-5

**注：**我们的 Ql\_SecureData\_Read/Ql\_SecureData\_Store 接口都是通过操作 AT 命令来实现的，所以需要等到 RIL 初始化后才能调这些接口。

#### 1.2、对于我们没有增加的 AT 命令，客户如何处理

客户可以参考我们已经实现的 RIL 接口和 AT 命令手册，自己封装一个接口

### 2、 URC

#### 2.1、某些 URC 在 SDK 中并没有实现，如何处理

客户可以参考我们已经实现的 URC 处理接口和 AT 命令手册，自己封装

#### 2.2、URC 消息处理

默认 URC 消息往主 task 中发送，所以主 task 一定需要增加接收消息的接口（Ql\_OS\_GetMessage），用户也可在 RIL\_URC.c 中更改 URC 消息发送的 task id,往其他 task 发送 URC 消息，建议不要更改。

#### 2.3、收到 type（msg.param1）为 101 的 URC 如何处理

这类 URC 被称为未定义的 URC，即我们 opencpu 收到这类 URC 时，并没有找到相应 URC 处理的接口，对此进行解析处理。

用户如果需要解析这类 URC，可以自己来增加对应的接口（在 RIL\_URC.c 中），如果不需要关心这类 URC，也可以直接忽略。

## 2.4、URC 上报的条件

URC 的上报都是有一定条件的，比如网络状态的“\r\n+CREG:”，需要网络状态改变时触发。“\r\nALARM RING\r\n”，需要闹铃时间到达，其他 URC 也是类似。

# 3、MESSAGE

## 3.1、s32 Ql\_OS\_GetMessage(ST\_MSG\* msg)

这是一个很重要的接口，在 opencpu 中，可以通过这个接口来处理当前 task 收到的消息。

它有如下的特点：

- 1、可以处理当前 task 或者其他 task 发来的消息
- 2、TIMER、EINT、ADC 事件的处理，也要依赖于在这个接口中根据不同的消息来调用 callback
- 3、当 task 阻塞在这个接口的时候，任务会被挂起
- 4、task 只有来消息时，会继续执行一次，其余时间都在这个接口处阻塞。

## 3.2、发送消息时，一定要注册 taskid

想知道某个 task 的 id,有如下方式：

1. 在\custom\config\ custom\_task\_cfg.h 中，查找对应的 task 配置

/*----- Task Entry Function   Task Id Name   Task Stack Size (Bytes)   Default Value1   Default Value2  -----*/				
TASK_ITEM(proc_main_task,	main_task_id,	10*1024,	DEFAULT_VALUE1,	DEFAULT_VALUE2)
TASK_ITEM(proc_reserved1,	reserved1_id,	5*1024,	DEFAULT_VALUE1,	DEFAULT_VALUE2)
TASK_ITEM(proc_reserved2,	reserved2_id,	5*1024,	DEFAULT_VALUE1,	DEFAULT_VALUE2)

2. 通过接口（Ql\_OS\_GetActiveTaskId）来获取当前的 task id

**注：**如果往一个不存在或者错误的 task id 发送消息，可能会导致系统异常。

## 3.3、每个 task 的消息队列，最大只能保存 30 条消息，如果消息队列溢出，模块将会死机或者重启

用户实际使用时，如果有频繁的消息需要处理，切记不要在 task 中做任何阻塞或者 Ql\_Sleep 动作。

这类问题可以抓 dump 信息：

```
文件(F)  编辑(E)  格式(O)  查看(V)  帮助(H)
== 本分析报告仅供参考，如有特殊case，请联系SS ==

== 异常报告 ==
详细描述: 发送消息时，QLT3的外部消息队列已满
可能原因: 1. 疯狂发送消息导致溢出，2. 高优先级抢占CPU导致低优先级没机会执行(文件系统操作会占用大量CPU资源，)
软件版本: M26FAR01A04
硬件版本: M26FA
当前进程: QL -> 请查看进程分析部分
```

## 4、EVENT、MUTEX

### 4.1、MUTEX 不支持重复锁的机制。

### 4.2、MUTEX 创建后不会释放。

## 5、WTD

### 5.1、opencpu 方案中什么情况需要增加外部看门狗芯片

Opencpu 的用户中，如果用户的设备应用环境比较恶劣，拆装比较复杂，返厂升级成本较高等情况下，用户一定需要增加外部看门狗。

**注：**一般的除了用户设备外部已经有 MCU 可以模拟看门狗时序，其他情况都建议增加外部看门狗。

### 5.2、为什么不选择软件看门狗的方案

当客户的应用 APP 发生了程序跑飞、数组越界、指针错误等，程序会直接死机，导致软件重启的逻辑不再执行。

### 5.3、opencpu 方案推荐的看门狗设计方案介绍

- 1、硬件设计：客户在设置看门狗电路之前，一定要严格参考我们的看门狗文档（Quectel\_OpenCPU\_Watchdog\_Application\_Note\_V1.0.pdf）
- 2、软件设计：为了避免程序中由于某些阻塞或者复杂业务导致不能及时喂狗，我们推荐用户使用我们的逻辑看门狗

```
// Initialize external watchdog:
// specify the GPIO pin (PINNAME_NETLIGHT) and the overflow time is 600ms.
ret = Ql_WTD_Init(0, PINNAME_NETLIGHT, 600);
if (0 == ret)
{
    APP_DEBUG("\r\n<--OpenCPU: watchdog init OK!-->\r\n");
}

// Create a logic watchdog, the interval is 3s
wtdid = Ql_WTD_Start(3000);
```

实际的喂狗引脚 →

实际喂狗时间：取决于客户看门狗芯片的选择

这个数值取决于当前task最大一次任务轮询所需要的时间，可以尽量设置大些，用户只需要起个定时器，在逻辑看门狗到来之前，进行喂狗动作即可

注：具体可以参考 SDK 中的 example\_watchdog.c

## 5.4、可以在每个 task 中都增加一个看门狗吗？

可以，每个 task 都可以设置一个逻辑看门狗，当其中一个 task 的看门狗没有及时喂狗，就会导致模块重启。

## 5.5、看门狗芯片的选型

用户可以使用我们文档中推荐的看门狗型号（TPS3823-33DBVR）也可以自行选购，需要注意的是，用户选择的看门狗复位时间一定要大于 1000ms，因为我们模块开机时，喂狗的时间间隔最大会达到 1000ms。

## 5.6、看门狗引脚配置一致。

在\custom\config\custom\_sys\_cfg.c 中

```
00074: /******
00075: /* Define the GPIO pin for external watchdog.
00076: /* NOTES:
00077: /* Customer may specify two GPIOs if needed.
00078: /******
00079: static const ST_ExtWatchdogCfg wtdCfg = {
00080:     PINNAME_PCM_OUT, // Specify a pin which connects to the external watchdog
00081:     PINNAME_END // Specify another pin for watchdog if needed
00082: };
```

这个地方配置的看门狗引脚，是为了告诉模块开机和 fota 升级在 bootlooder 阶段的喂狗引脚

```
// Initialize external watchdog:
// specify the GPIO pin (PINNAME_PCM_OUT) and the overflow time is 600ms.
ret = Ql_WTD_Init(0, PINNAME_PCM_OUT, 600);
```

这个接口的配置引脚，是告诉模块，在 task 运行期间，模块的喂狗引脚

注：实际应用中，这两个引脚需保持一致

## 6、TASK

### 6.1、TASK 中不能及时处理业务

如果调用阻塞接口或者 QI\_Sleep 接口，task 会阻塞，导致无法继续处理其他业务或者消息

### 6.2、获取 taskid 不正确

**问题现象：**客户发现在如下的 callback 中获取 TASKid,返回的是 6,实际在 main task(应该为 0)

```
#ifdef __GNSS_NEW_API__
iRet = QI_GNSS_PowerOn(RMC_EN | GGA_EN, Customer_NMEA_Hdlr, NULL); // Also can use ALL_NMEA_EN to
#else
iRet = RIL_GPS_Open(1);
#endif
```

**问题原因：**事实上这个 callback 接口并不是在当前的 task 中运行的，而是传递给内核的 GPS\_task 加载，所以返回错误的 task id.

**注：**我们的一些通过 register 注册的接口，比如 TIMER、ADC、EINT 这些接口的 callback 都是在注册的 task 中运行

默认两个 task 的位置不能改变

### 6.3、栈溢出导致模块死机

**问题现象：**当客户的代码运行到某一个位置时出现死机，

**问题原因：**客户 task 中定义了大量的局部大数组，导致栈空间不断减小，当到栈底时就会死机。

**问题解决：**使用全局变量来代替这些局部变量，去除不必要的大数组，必须要时使用 malloc。

### 6.4、task 中如何获取 us 级别的延迟

```
// 1: 7us
// 5: 15us
// 20: 30us
// 100: 300us
// 1000: 1300us

void delayus (volatile u32 time)
{
    while(--time);
}
```

## 6.5、task 调度

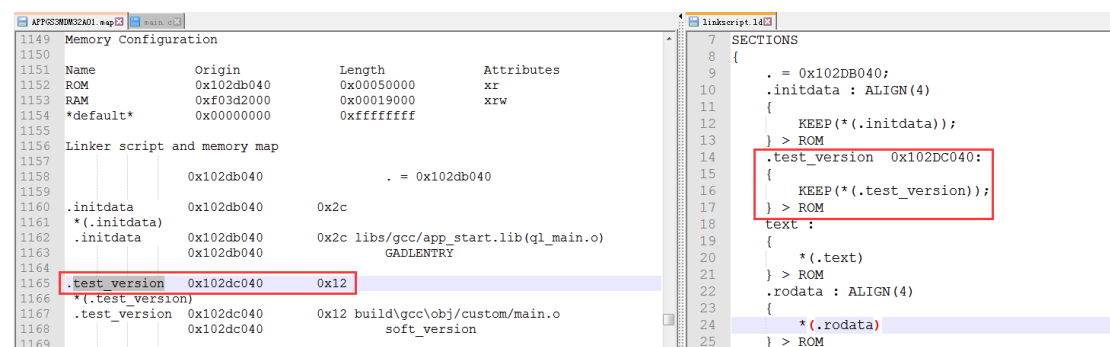
如果任务中不增加接口（`Ql_OS_GetMessage`）来挂起任务，而是采用如下的方式来做 `while`（1）循环，会由于当前 `task` 占用大量的系统资源，导致模块死机。如果客户需要使用这样的轮询方式，可以在 `while`（1）中增加延迟接口。

```
void proc_subtask1 (s32 taskId)
{
    while(TRUE)
    {
        // Q1_Sleep(5);
    }
}
```

## 8、其他

### 8.1、客户想在 ROM 中的某个地址写入数据

更改我们的 linkscript.ld 文件如下:



在代码中定义段，如下：

```
__attribute__((section(".test_version")))
const char soft_version[] = "mc20 dd test v100";
```

实际 bin 结果:

[illegible]

00001010	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	mc20_dd_test_v10
00001020	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	0 0 Yá0 Yá
00001030	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	0 Yá Qá °. i.
00001040	6D 63 32 30 5F 64 64 5F	74 65 73 74 5F 76 31 30	úyy° Yá Yá 0 á
00001050	30 00 00 00 00 00 00 00	30 00 9F E5 30 10 9F E5	Qá i.úyy° Y/á
00001060	30 20 9F E5 02 00 51 E1	08 00 B0 B8 08 00 A1 B8	5. =ðP =ðP =ð
00001070	FB FF FF BA 20 10 9F E5	20 20 9F E5 00 30 A0 E3	æ8=ððµWFNFÈFà' *
00001080	02 00 51 E1 08 00 A1 B8	FC FF FF BA 1E FF 2F E1	ø # * ði\ÄT 3"B
00001090	90 35 2E 10 00 20 3D F0	50 20 3D F0 50 20 3D F0	úN 4 FµFøFø4 4 G
000010A0	F8 38 3D F0 F0 B5 57 46	4E 46 45 46 E0 B4 03 2A	, f ð \$ x p 3
000010B0	0E D8 00 23 00 2A 04 D0	CC 5C C4 54 01 33 93 42	
000010C0	FA D1 1C BC 90 46 99 46	A2 46 F0 BC 02 BC 08 47	
000010D0	82 18 83 07 08 D0 03 1C	03 24 0D 78 1D 70 01 33	

**注：**客户最好还是不要固定段地址，如果前一个段和当前段空间预留不足，**编译会提示超出范围**，如果预留过大，**会造成 ROM 的浪费**。建议还是使用我们默认的动态管理方式。

## 8.2、我们模块是大端还是小端模式

我们 2G 平台的所有模块都是小端模式。

## 8.3、调用 QI\_Reset()重启模块需要拉低 PWRKEY 吗？

不需要，调用 QI\_Reset()接口后，模块的 pwrkey 引脚不管拉高还是拉低，都不影响重启。

## 8.4、针对 MC20A06 和 A07 版本获取的时间不一致问题

### 问题描述：

客户反馈, MC60CAR01A06 版本和 MC60CAR01A07 版本获取时间的信息有差异。

A06 版本获取到的是 UTC 时间

A07 版本获取到的是本地(UTC+时区)时间

### 问题解决：

为了保证两个版本的兼容性，在 SDK 中需保证 timezone 的值一直为 0，这样无论两个版本获取到的是本地时间还是 UTC 时间，实际上时间都是相等的。

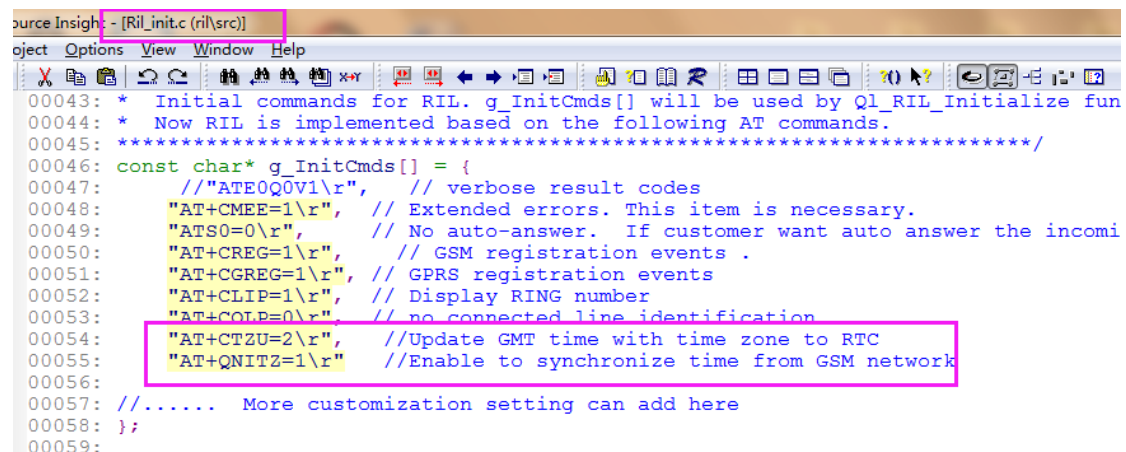
### 基本思路：

- 1.设置"AT+CTZU=2"， 更新 UTC 时间到 RTC 中。
- 2.设置"AT+QNITZ=1"， 使能基站同步网络时间的功能
- 3.当模块收到基站下发的网络时间，会上报 URC(+QNITZ....).
- 4.模块收到这个 URC 时，先保存获取到的 timezone 值(后面如果需要获取本地时间可以自己计算)。
- 5.把 timezone 设置为 0 后，再重新 set 下时间信息，保证现在写入的 RTC 中的时区为 0



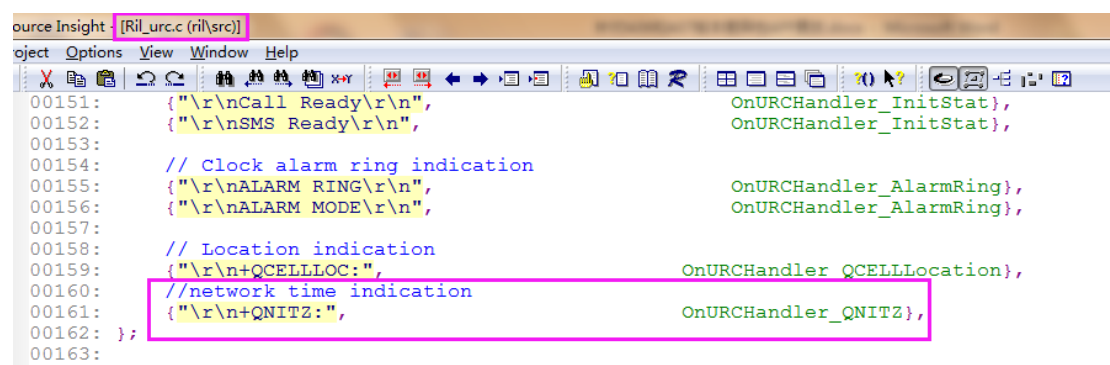
## 代码实现

1.在 ril init 阶段增加两个配置。



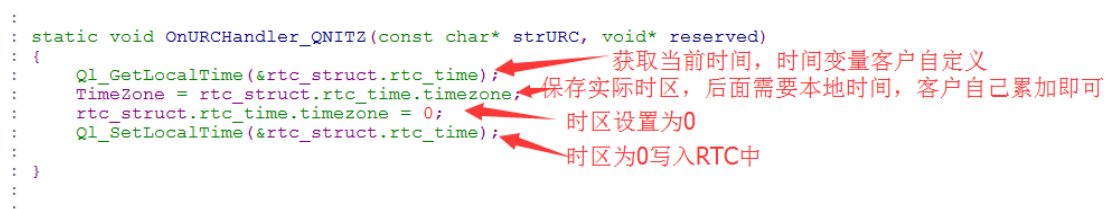
```
00043: * Initial commands for RIL. g_InitCmds[] will be used by Ql_RIL_Initialize fun
00044: * Now RIL is implemented based on the following AT commands.
00045: *****/
00046: const char* g_InitCmds[] = {
00047:     // "ATE0Q0V1\r", // verbose result codes
00048:     "AT+CMEE=1\r", // Extended errors. This item is necessary.
00049:     "ATS0=0\r", // No auto-answer. If customer want auto answer the incomi
00050:     "AT+CREG=1\r", // GSM registration events .
00051:     "AT+CGREG=1\r", // GPRS registration events
00052:     "AT+CLIP=1\r", // Display RING number
00053:     "AT+COLP=0\r", // no connected line identification
00054:     "AT+CTZU=2\r", //Update GMT time with time zone to RTC
00055:     "AT+QNITZ=1\r", //Enable to synchronize time from GSM network
00056:
00057: //..... More customization setting can add here
00058: };
00059:
```

2.在 ril urc 中增加对于 QNITZ 的处理的接口



```
00151: {"\r\nCall Ready\r\n", OnURCHandler_InitStat},
00152: {"\r\nSMS Ready\r\n", OnURCHandler_InitStat},
00153:
00154: // Clock alarm ring indication
00155: {"\r\nALARM RING\r\n", OnURCHandler_AlarmRing},
00156: {"\r\nALARM MODE\r\n", OnURCHandler_AlarmRing},
00157:
00158: // Location indication
00159: {"\r\n+QCELLLOC:", OnURCHandler_QCELLLoc},
00160: //network time indication
00161: {"\r\n+QNITZ:", OnURCHandler_QNITZ},
00162: };
00163:
```

3.上报 QNITZ 接口的实现。



```
: static void OnURCHandler_QNITZ(const char* strURC, void* reserved)
: {
:     Ql_GetLocalTime(&rtc_struct.rtc_time);
:     Timezone = rtc_struct.rtc_time.timezone;
:     rtc_struct.rtc_time.timezone = 0;
:     Ql_SetLocalTime(&rtc_struct.rtc_time);
: }
:
```

获取当前时间，时间变量客户自定义  
保存实际时区，后面需要本地时间，客户自己累加即可  
时区设置为0  
时区为0写入RTC中

注:客户在代码处理中，如果需要调用 Ql\_SetLocalTime 接口，则 timezone 都必须为 0，如果设置其他非 0 值，还是会导致版本差异性。timezone 的数值客户可以自己设置个全局变量来管理。

## 8.5、如何添加用户自己的 lib 库

1、用 gcc 自带的 ar 工具把当前的所有.o 文件打包成库

```
arm-none-eabi-ar -r test.lib *.o
```



## 2、makefile 中增加编译的库

```
USERLIB += libs/gcc/app_start.lib \  
          libs/gcc/test.lib
```

## 8.6、编译时，发现报错 `_sbrk` 未定义

客户调用了 C 标准库的 `strtod` 接口，这些接口我们默认增加的 `lib` 库中缺少部分依赖接口的实现。客户可以参考源码自己来实现这类接口。

**注：**我们已经实现了部分的标准库接口，并使用 `QL` 开头，用户如果用到这类标准库，需要使用 `QL` 开头，而不能直接调用标准库。

# 休眠模式部分

## 1、睡眠模式介绍

在嵌入式应用中，系统的功耗越来越受到人们的重视，这一点对于需要电池供电的便携式系统尤为明显，降低系统功耗，延长电池的寿命，就是降低系统的运营成本。系统功耗的最小化需要从软、硬件两方面入手，下面我们重点介绍软件实现。

模块有 `32K` 和 `13M` 两个时钟，当模块进入睡眠模式后，只有 `32K` 时钟仍然工作，所以睡眠模式又称为慢时钟模式。下面将介绍模块如果进入和退出睡眠模式，以及需要注意的事项

## 2、如何进入睡眠模式

模块可以通过下面两种方式其一进入睡眠模式，需要强调的是系统并不会立即进入睡眠，而是依赖于当前网络及系统所有任务执行状态，只有当系统处于 `IDLE` 状态后才会自动进入睡眠。所以如果需要模块快速进入睡眠模式，需要关闭系统中频繁的 `timer` 等中断。

2.1、AT+QSCCLK(标准下推荐方案)

AT+QSCCLK 慢时钟配置	
测试命令 AT+QSCCLK=?	响应 +QSCCLK: (<n>取值列表)  OK
查询命令 AT+QSCCLK?	响应 +QSCCLK:<n>  OK
配置命令 AT+QSCCLK=<n>	响应 OK

<n>	
0	禁用慢时钟
1	启用慢时钟，拉高 DTR 生效，拉低 DTR 唤醒
2	启用慢时钟，5 秒内串口无数据交互即进入睡眠模式；有数据时唤醒

特别提醒：

模式2，第一次串口数据会被丢弃，仅仅起唤醒作用。建议此模式下先发一个AT 来唤醒模块再继续其他业务。

2.2、QI\_SleepEnable(Opencpu 推荐方案)

```

/*****
* Function:   QI_SleepEnable
*
* Description:
*             Set the module into sleep mode at once
*
* Return:
*             QL_RET_OK indicates this function successes.
*             QL_RET_NOT_SUPPORT this function not support.
*****/
s32 QI_SleepEnable(void);
```

在 opencpu 模式下，除了设置 AT 命令 (AT+QSCCLK)，还可以通过接口 (QI\_SleepEnable) 使模块进入睡眠模式，调用此接口，不需要配置模块的 DTR 引脚。但是当模块进入睡眠模式后，DTR 引脚电平跳变会唤醒模块。

3、如何退出睡眠模式

- 在标准模式退出睡眠，可以通过 AT 命令 (AT+QSCCLK=0)。
- 使用 Opencpu 方案，既可以通过 AT 命令 (AT+QSCCLK=0)，也可以通过

如下接口（QI\_SleepDisable）退出睡眠模式。

```

/*****
* Function:    QI_SleepDisable
*
* Description:
*             Exit the sleep mode
*
* Return:
*             QL_RET_OK indicates this function successes.
*             QL_RET_NOT_SUPPORT this function not support.
*****/
s32 QI_SleepDisable(void);

```

需要注意的是，当模块进入睡眠模式（AT+QSClk=2 除外），模块串口 RX 被 closed，不再接收数据，所以如果发送命令退出睡眠模式，先要唤醒模块，当模块被唤醒后，及时通过“AT+QSClk=0”或者 QI\_SleepDisable 退出睡眠模式。否则当系统进入 IDLE 模式后，会再次自动进入睡眠。

3.1、以下方式可以唤醒模块：

- 语音
- 收到短信或者彩信
- GPRS 数据（如 server 下发 TCP 数据）
- 外部中断(包括 ALARM, TIMER,EINT)
- 串口来数据（仅 AT+QSClk=2 模式下起作用）

特别提醒：

在AT+QSClk=1模式下,可以通过将 DTR 引脚拉低20ms直接退出睡眠模式，不需发送 AT+QSClk=0.

4、耗流

下面是模块典型应用的耗流情况(MC20E 为例)

● GSM 工作但不传输数据功耗（关闭 BT，GPS）

关机模式	220	uA
睡眠模式 @DRX=5	1.2	mA
睡眠模式 @DRX=5	0.8	mA
最少功能模式		
AT+CFUN=0 空闲模式	13	mA
睡眠模式	0.68	mA
AT+CFUN=4 空闲模式	13	mA
睡眠模式	0.73	mA

## ● GSM+BT 工作功耗（关闭 GPS）

模块 RF 状态	GSM 模式	蓝牙状态	耗流
全功能 (AT+CFUN=1)	空闲模式	关闭	13.01mA
		广播	13.59mA
	睡眠模式	关闭	1.42mA
		广播	2.06mA
关闭 RF 发送和接收功能 (AT+CFUN=4)	空闲模式	关闭	12.51mA
		广播	13.08mA
	睡眠模式	关闭	0.7mA
		广播	1.32mA
最少功能 (AT+CFUN=0)	空闲模式	关闭	12.47mA
		广播	13.04mA
	睡眠模式	关闭	0.64mA
		广播	1.26mA

## ● GSM 传输数据（关闭 BT、GPS）

### GPRS 数据传输

#### 数据传输模式，GPRS（3 收，2 发）Class 12

GSM850	@功率等级 5，<550mA，典型值 363mA
	@功率等级 12，典型值 131mA
	@功率等级 19，典型值 91mA
EGSM900	@功率等级 5，<550mA，典型值 356mA
	@功率等级 12，典型值 132mA
	@功率等级 19，典型值 92mA
DCS1800	@功率等级 0，<450mA，典型值 234mA
	@功率等级 7，典型值 112mA
	@功率等级 15，典型值 88mA
PCS1900	@功率等级 0，<450mA，典型值 257mA
	@功率等级 7，典型值 119mA
	@功率等级 15，典型值 89mA

## ● GPS 单独供电工作（关闭 BT、GSM）

参数	条件	典型值	单位
I <sub>VCC</sub> @捕获	@VCC=3.3V (GPS)	25	mA
I <sub>VCC</sub> @跟踪	@VCC=3.3V (GPS)	19	mA
I <sub>VCC</sub> @捕获	@VCC=3.3V (GPS+BeiDou)	29	mA
I <sub>VCC</sub> @跟踪	@VCC=3.3V (GPS+BeiDou)	26	mA
I <sub>VCC</sub> @Standby	@VCC=3.3V	0.3	mA
I <sub>BCKP</sub> @Backup	@V <sub>BCKP</sub> =3.3V	14	uA

### 特别提醒：

实网下由于干扰和当地运营商DRX等策略的不同，功耗会略微大于实验室环境，另上述测试数据客户可以参考我司硬件手册。

## 5、注意事项

### 5.1、睡眠模式，模块串口不能接收数据（AT+QSCLK=2 除外）

模块在睡眠模式下，仍会定时和基站进行数据交互，所以如果频繁的往串口发送数据，会发现偶尔可以接收到串口数据，但时间很短，大约只有几十毫秒，这个时间取决于当地基站设置的 DRX 值。

### 5.2、睡眠模式，必须关闭 GP-TIMER 和 GPS

频繁的中断或者 GPRS 数据，会导致模块很难进入睡眠模式或者进入后会马上唤醒

### 5.3、串口 RX 不能拉低

所有串口的 RX 都不能外部硬件拉低，否则模块不能进入睡眠模式

### 5.4、标准模式 DTR 推荐设计

标准模式下，为了更方便的切换模块的睡眠/唤醒模式，建议客户通过外部 MCU 控制 DTR 电平。

## 5.5、Opencpu 方案，task 中增加接口（QI\_OS\_GetMessage）

当模块有消息需要处理时，会通过这个接口来处理，没有消息时 task 挂起，可以立刻进入睡眠模式。

## 5.6、进入睡眠模式前关闭频繁中断（如 TIMER、EINT）

如果有频繁中断，模块将很难进入睡眠模式，或者会频繁唤醒，导致功耗降不下去。例，如果应用中增加了周期 1s 的 timer，模块功耗降不会明显下降，可以延长周期或者关闭 timer。

### 特别提醒：

我们测试的睡眠模式功率，是裸模块下的数据，如果客户在自己 PCB 测试，需要减去外设和供电电路的功耗。

## 6、opencpu 常见唤醒机制

在 opencpu 下，客户可以通过如下方式来唤醒模块：

- 1、timer （客户在模块睡眠前启动一个 timer，比如定时 10min，模块会每隔 10min 唤醒一次，模块唤醒后，可以去进行相关业务，业务处理完，继续睡眠，等待下次唤醒）
- 2、模块收到 GPRS 数据 （客户通过服务器下发数据唤醒模块，模块唤醒后把相关业务数据上报给服务器后，继续睡眠）
- 3、EINT （客户通过触发外部中断来唤醒模块，这个方式比较适合模块外部有 MCU,可以控制模块外部引脚电平的跳变，特殊的场景也可以使用按键来触发）
- 4、电话/短信/彩信 （客户在产品出货时，有记录产品使用的 SIM 卡号，通过拨打电话或者发送短信来唤醒模块，这种方式实际使用的比较少）

## FOTA 升级部分

### 1、简介

为了满足客户 APP BIN 的 OTA(over the air)升级功能，我司提供了两种 OTA 方式，FTP 和 HTTP，客户可以自建一个 FTP/HTTP 服务器，通过我们的 fota 工具制作对应的升级包，放到服务器上，从而实现产品功能更新和问题修复。

### 1)客户需要搭建 FTP/HTTP 服务器

- 2)把编译生成的 APP BIN 通过工具制作成升级包，并放到搭建好的服务器上
- 3) 模块注册网络
- 4) 连接 FTP 服务器，download APP 到模块的 UFS/RAM 中
- 5) 读取前面的文件分包写入到 fota temp 区，统计 length(QI\_FOTA\_WriteData)
- 6) 设置升级标志位(QI\_FOTA\_Finish)
- 7) 把标志位等信息写入到最好一个 block(2Kb)(QI\_FOTA\_Update)
- 8)开始重启
- 9)重启后,在 extbootloader 阶段先判断升级标志位，如果成立，开始搬迁 temp 区数据到 APP
- 10)搬完继续正常开机流程。

## 4、常见客户问题

### 4.1、未做升级包

客户直接把 SDK 生成的 bin 文件放到服务器,未做升级包，FOTA 升级时检验错误返回 9999

### 4.2、fota 升级导致无法及时喂狗，模块复位

客户的逻辑看门狗时间设置过短（3s）,模块 FTP 流程耗时过长（下载 FIEL 比较耗时，FTP 的下载速率 2K/s）,造成 task 的阻塞时间过长，不能及时喂狗，造成模块重启。客户可以通过更改逻辑喂狗时间，解决问题。

### 4.3、linux 搭建 FOTA 服务器

**问题现象：**客户使用 Linux 平台搭建的服务器，FOTA 升级运行到 getsize 时返回-605（get size error）

**问题原因：**文件路径不对，客户用了根目录，实际应该用用户 home 目录，如，/home/yy/

### 4.4、FOTA 升级时，fota\_init 返回 error(-1)

**可能问题原因：**

- 1、客户的外部看门狗引脚设置重复，导致参数判断出错
- 2、我们引脚数有增减，导致客户设置（PINNAME\_END）出错。



## 4.5、增加 AT+QFTPCFG=6,1 命令

**问题现象：**FTP 连接成功，但是 downloading ftp file 时失败

**问题解决：**加一条配置命令(AT+QFTPCFG=6,1)，在被动模式下，强制模块使用原 FTP 服务器地址建立数据连接，而不是使用服务器分配的地址。

## 4.6、场景问题

**问题现象：**FTP 使用场景 1 进行 FOTA 升级不成功，在场景 0 下可以正常运行。

**问题原因：**发现在 M26A0 以前的版本，当使用场景 1 进行 FTP 配置后，FTP 的连接却使用场景 0 处理，此 bug 在 M26A04 以后版本已经修复。

## 4.7、FTP 服务器搭建问题

可以使用 FTP 工具先验证客户的 FTP 服务器是否能正常下载，上传数据。

## 4.8、QI\_FOTA\_WriteData 返回-9999 错误

**问题现象：**客户使用 HTTP 服务器进行 FOTA 升级时，获取到的数据长度不对，写数据返回 -9999

**问题原因：**客户传入了错误的文件路径，导致下载的文件不对。

# SIM 卡部分

## 1、常见客户问题

### 1.1、SIM 卡欠费

如果怀疑 SIM 卡欠费，可以拨打运营商电话：

[2017-12-05\_13:27:44] AT+QAUDCH=1

[2017-12-05\_13:29:11:005]OK

[2017-12-05\_13:29:13:095]AT+QMIC=0,8

[2017-12-05\_13:29:13:095]OK

[2017-12-05\_13:29:16:605]AT+CLVL=60

[2017-12-05\_13:29:16:605]OK

[2017-12-05\_13:29:17:572]AT+CLIP=1

[2017-12-05\_13:29:17:572]OK

[2017-12-05\_13:29:20:100]ATD10010;

[2017-12-05\_13:29:20:100]OK

## 1.2、写电话号码到 SIM 卡

某些 SIM 卡出厂可能没有写入电话卡号码，导致使用 AT+CNUM 命令无法读取到正确的 SIM 卡号。

解决如下：

- 1)部分卡运营商没有将电话号码写入 SIM 中，需要客户手动发送 AT+CPBS="ON",AT+CPBW=1,XXXXXX 写入号码)
- 2)客户向运营商购买 SIM 卡时，要求运营商写入电话号码。

## 1.3 SIM 卡热插拔后不能自动找卡

模块上电过程中，如果拔卡，再次插入 SIM 卡，需要切换 CFUN 才会重新找卡

## 1.4 发送特殊字符到模块，会出现乱码

把读取短信的模式设置成 LIB\_SMS\_CHARSET\_8859\_1

## 1.5 开机无法找到 SIM 卡

现象描述：开机后模块返回 +CPIN: NOT READY or NOT INSERTED。从以下几方面分析解决：

- 1) SIM 卡与卡座接触不良，可以尝试在 SIM 卡上增加垫片。
- 2) SMT 焊接不良，可以通过万用表测试模块与 SIM 卡焊盘之间的连通性。
- 3) SIM 卡已损坏，可以将 SIM 卡放在 EVB 或者手机上测试，确认是否正常。
- 4) 假如 VBAT 电源走线过于靠近 SIM 信号线，可能会由于 VBAT 电源线上的纹波太大，干扰到 SIM 卡各个信号线，导致 SIM 卡无法识别，可尝试切断附近的 VBAT 线，通过其他途径单独给模块供电，看问题是否消失。
- 5) 假如客户的 SIM 卡座和模块确实离得很远，走线也比较长，各个信号线也没有地屏蔽处理，很有可能导致 SIM 卡无法识别，可以尝试使用较短的飞线直接连接到卡座。
- 6) SIM 卡信号线上并联的 ESD 器件寄生电容需要不大于 50pF，过大可能会导致 SIM 卡无法识别，可尝试直接去掉该 ESD 看问题是否消失。
- 7) 天线摆放位置以及射频走线不合理也会干扰到 SIM 卡，从而导致 SIM 卡无法识别，客户可以使用 AT+CFUN=4,1 关闭模块射频发射和接收，确定 SIM 卡工作是否正常，若正常则表示 SIM 卡受到 RF 干扰。为了尽可能的消除 RF 干扰，SIM\_DATA,SIM\_CLOCK,SIM\_RST,SIM\_VDD 并联滤波电容。具体参考模块硬件设计手册。
- 8) 周边环境有干扰，确认测试现场周围是否有超强电/磁场存在，比如高压输电线、大功率无线设备等， 可以尝试使用一个屏蔽罩盖住 SIM 卡以及 SIM 卡各个信号线的走线，或者使用地屏蔽线处理 SIM 卡信号线，看问题是否消失。

## 1.6 使用过程中出现掉卡的问题

现象描述：模块开机返回+CPIN: READY, 过一会儿后模块返回 +CPIN: NOT READY。

可能原因：

1) RF 干扰，可以通过以下方法确认：

- a) 可以尝试使用 AT+CFUN=4,1 关闭模块射频发射和接收，看看问题是否仍然存在。
- b) 可以尝试把天线靠近 SIM 卡和 SIM 信号线，看看问题出现概率是否有所增加。
- c) 确认测试现场周围是否有超强电/磁场存在，比如高压输电线、大功率无线设备等。
- d) 可尝试通过并联 15~33PF 电容来滤除射频干扰。

2) 硬件设计存在问题。可以尝试如下办法确认：

- a) 假如 VBAT 电源走线过于靠近 SIM 信号线，当刚开机的时候纹波可能不是很大，模块能正常找到 SIM 卡，但是当模块开始注册并同步网络时，纹波增大可能直接干扰到 SIM 卡的信号线，导致掉卡。可尝试切断附件的 VBAT 线，通过其他途经单独给模块供电。
- b) 确认该卡工作是几伏，1.8V 还是 3V？有些情况下，1.8V 的卡更容易被干扰，可尝试使用 AT 命令 AT+QSIMVOL 锁定 3V 看问题是否依然存在。

3) SIM 卡质量比较差。可以尝试如下办法确认：

- a) 换张卡看看是否有同样的问题。
- b) 把问题卡放在 Quectel EVB 上确认是否有同样问题。

## 1.7 哪些原因可能造成模块开机后无法注册网络？

- 1) 确认模块是否找到 SIM 卡（AT+CPIN?）；
- 2) 确认 SIM 卡是否欠费；
- 3) 确认模块工作频段（AT+QBAND）和工作模式（AT+CFUN）是否正确；
- 4) 确认模块射频信号是否正常（AT+CSQ）；
- 5) 查询模块 IMEI 号（AT+GSN），确认 IMEI 号是否合法。由于有些客户会改写 IMEI 号，导致部分区域网络认为此 IMEI 非法，从而禁止模块注册。

## 1.8 文本模式和 PDU 模式有什么区别？

文本模式和 PDU 模式是提供给客户编辑短信的不同方式。

- 1) 使用 Text 模式时，需要使用 AT 命令设置相关参数（CMGF,CSCS,CSMP）；
- 2) 使用 PDU 模式时，需要自己组串，各项参数都包含在 PDU 串（SUBMIT/ DELIVER/ STATUS REPORT）中，例如短信中心号码、收件人号码、有效期等。

## 1.9 普通短信内容最大长度是多少？

根据 3GPP 规范，普通短信的最大长度为 1120bits，当采用不同的编码方式时，普通短信包含的最大字符长度则有所不同，具体如下：

- 1) 采用 7 位 GSM 编码，可以输入  $1120\text{bits}/7\text{ bit}=160$  个字符；
- 2) 采用 8 位 GSM 编码，可以输入  $1120\text{bits}/8\text{ bit}=140$  个字符；
- 3) 采用 16 位 UCS2 编码，可以输入  $1120\text{bits}/16\text{ bit}=70$  个字符。

### 1.10 当短信已存满，新来短信时模块如何处理？

当短信已经存满，新来短信时模块将无法接收。

如果设置 `AT+QEXTUNSOL="SM",1`，当短信存满后则会上报 `+TSMSINFO:322` 提示。

### 1.11 国内号码互发短信时，加上国际号“+86”为什么某些地区会发送失败？

因为国内有些地区基站不识别国际号“+86”，会认为号码错误而导致无法发送的问题。

### 1.12 开机阶段一些 SMS 相关命令为什么会执行失败？

模块开机后，会对 SIM 卡进行初始化，因为 SMS 用到的很多参数是跟 SIM 卡相关的，这些参数要等待 SIM 卡初始化成功后才可以执行。一般来说 SIM 卡可以在开机后 20s 左右完成初始化，但是有的 SIM 卡由于电气特性等原因导致初始化速度很慢，可能需要 40s 甚至更长时间。可以通过 `AT+QINISTAT` 命令查询 SIM 卡关于短信部分是否初始化完毕，当 `QINISTAT` 返回为 3 后，表示已经初始化完毕，这时再执行 SMS 相关命令就不会出现返回失败的问题。

#### 备注

在 SIM 卡初始化完成前，以下命令会返回失败：QMGDA CGSMS CSMS CMGL CSAS CRES CMSS CSCB CMGW CMGD CMGR CPMS。
---

### 1.13 模块关机，给模块发短信，模块开机需要一定时间才能收到。这个时间能否缩短？

短信中心转发会判断模块的开关机状态，如果关机会有重发机制，所以这个时间由短信中心控制我们无法更改。当模块开机短信功能初始化完成，网络正常情况下就会在这个时间内收到短信中心重发的短信。

### 1.14 +CME +CMS 错误的区别

CMS 是短信中心的返回错误，CME 是设备返回的错误。

## 网络部分

### 1、TCP、UDP

我们模块一共支持最大同时建立 6 路 socket，所以用户最大可以同时建立 6 路的 TCP 或者 UDP。

## 找卡、注网

- 1、s32 RIL\_SIM\_GetSimState(s32 \*stat)-----AT+CPIN?
- 2、s32 RIL\_NW\_GetGSMState(s32 \*stat)-----AT+CREG?
- 3、s32 RIL\_NW\_GetGPRSState(s32 \*stat)-----AT+CGREG?

#### 常见返回值

NW\_STAT\_SEARCHING: 刚开机上电时, ME 搜索附近运营商准备注册

NW\_STAT\_REGISTERED : 本地网络 注册成功

NW\_STAT\_REGISTERED\_ROAMING: 漫游网络 注册成功 //这种情况客户代码中容易忽略

NW\_STAT\_REG\_DENIED : 注册被拒绝 // 欠费、GPRS 业务未开通等(可能基站不支持, 需要基站优化, 自动选择基站的三个指令)

**注:** 客户虽然使用的是本地卡, 测试环境也是在本地, 但设备实际用在外地或者运动环境比如车载, 会出现漫游的情况, 建议最好实际代码考虑漫游注册的情况。

### 注册、激活、域名解析

4、s32 QI\_GPRS\_Register(u8 contextId, ST\_PDPContxt\_Callback\* callback\_func, void\* customParam);

5、s32 QI\_GPRS\_Config(u8 contextId, ST\_GprsConfig\* cfg);

6、s32 QI\_GPRS\_Activate(u8 contextId);

7、s32 QI\_GPRS\_GetDNSAddress(u8 contextId, u32\* firstAddr, u32\* seconAddr)

**注:** IP 地址和 DNS 服务器都可以由 DHCP 服务器分配。DHCP 服务器最大的好处就是可以防止局域网内电脑的 IP 地址冲突, 防止 IP 冲突, 网络不稳定。如果我们手动分配 IP 地址, 电脑多了, 很多可能会分到重复的 IP 地址, 改起来就麻烦了。使用了 DHCP 服务器自动分配 IP 的功能, 就可以避免这个麻烦, 由路由器统一分配 IP 地址。DHCP 服务器把每个 IP 地址只分给一台电脑, 这样就能保证局域网的稳定性。

8、s32 QI\_GPRS\_GetLocalIPAddress(u8 contextId, u32\* ipAddr)

**FUNC:** 获取模块本地 IP 地址

**注:** 步骤 7、8 中的数值在激活中已经获取, 这两个函数功能只是获取, 查询功能。

9、s32 QI\_IpHelper\_ConvertIpAddr (u8 \*addrString, u32\* ipAddr)

**FUNC:** 检查 IP 地址并把 string 类型的 IP 转化成 integer

10、s32 QI\_IpHelper\_GetIPByHostName(u8 contextId, u8 requestId, u8\*hostname, Callback\_IpHelper\_GetIpByName callback\_GetIpByName)

**FUNC:** 通过域名获取 IP 地址

MSG\_ID\_APP\_SOC\_GET\_HOST\_BY\_NAME\_IND---->opencpu\_ind\_gethostipbyname----> 调用 callback

### Socket 业务

## 先简单介绍下 MTK socket

MTK socket 主要有三种模式: block(阻塞), non-block(非阻塞), Asynchronous(异步), 组合方式 也只有三种 1 block , 2 non-block, 3 non-block + Asynchronous。

block 模式下, 调用相应的函数(接受或者发送数据), 如果这个函数动作没有完成(没有发送或者接受完成), 那么函数就不会返回, 那么调用的整个 task, 就会阻塞, 进行不了任何动作。如果在 MMI MOD 里面直接用这个, 那是很危险的, 会出现手机没有响应这种假死现象, 所以几乎不用这个模式。

non-block 模式下, 调用相应的函数, 可能返回 ok 或者 block, 大部分情况下返回 block, 表示数据还没有处理完毕, 但是函数会立即返回。但是什么时候表示数据 处理完成呢, 这也是一个比较头疼的事情。这个时候要配合 select 函数来一起使用, 这样就需要自己轮询去查询相应的 socket 是否可以使用了。一般也不用, 效率比较低。

non-block + Asynchronous 模式: 这个模式推荐使用, 可以编成工作中几乎就用这种方式, non-block, 就不会阻塞, 不会让应用看起来假死, Asynchronous 模式, 那么当使用函数返回 block 时, app 只要注册相应的回调函数, 当数据处理完毕了, 就会收到相应的通知, 不用自己去轮询, 效率也就高了。—————>

下面接口使用的就是这种模式

11、s32 Ql\_SOC\_Register(ST\_SOC\_Callback cb, void\* customParam);

12、s32 Ql\_SOC\_Create(u8 contextId, u8 socketType);

13、s32 Ql\_SOC\_Connect(s32 socketId, u32 remoteIP, u16 remotePort);

client 向 server 发送三次连接请求而均未得到 server 回复时, 连接失败

MOD_TCPIP	MOD_SOC	TCPIP_SOC_SAP	MSG_ID_SOC_TCPIP_CONNECT_CNF
MOD_SOC	MOD_QL	SOC_APP_SAP	MSG_ID_APP_SOC_NOTIFY_IND
!!!			
Dec	Oct	Bit	Enum
1	0001	00000001	
12	0000014	0000000000001100	
0	0000	00000000	
8	0010	00001000	
0	0000	00000000	
-1	0377	11111111	SOC_CONNECT KAL_FALSE SOC_ERROR
0	000000000000	00000000000000000000000000000000	

server 的 IP 或 port 出错, server 未开启, server 拒绝 client 连接设置等都会出现上述情况

MOD_QL	MOD_TCPIP	TCPIP_SOC_SAP	MSG_ID_SOC_TCPIP_CONNECT_REQ
MOD_TCPIP	MOD_RATDM	TCM_TCPIP_SAP	MSG_ID_PS_DATA_REQ
MOD_APPTCPIP			soc_connect=-2
MOD_RATDM	MOD_TCPIP	RATDM_TCPIP_SAP	MSG_ID_PS_DATA_IND
MOD_TCPIP	MOD_SOC	TCPIP_SOC_SAP	MSG_ID_SOC_TCPIP_CONNECT_CNF
MOD_L4C			idle:60
MOD_TCPIP			tcp_input_rtt update, rtt = 3 and reset rttshift value
MOD_TCPIP	MOD_RATDM	TCM_TCPIP_SAP	MSG_ID_PS_DATA_REQ
MOD_SOC	MOD_QL	SOC_APP_SAP	MSG_ID_APP_SOC_NOTIFY_IND

三次握手



Dec	Oct	Bit	Enum
1	0001	00000001	
12	0000014	0000000000001100	
0	0000	00000000	
8	0010	00001000	
1	0001	00000001	
0	0000	00000000	
0	00000000000	00000000000000000000000000000000	

SOC\_CONNECT  
KAL\_TRUE  
SOC\_SUCCESS

## 14、s32 Ql\_SOC\_Send(s32 socketId,u8\* pData,s32 dataLen);

```
15、s32 QI_SOC_Rcv(s32 socketId, u8* pBuffer, s32 bufferLen);
```

Oct	Bit	Enum
0001	00000001	
0000014	0000000000001100	
0000	00000000	
0001	00000001	SOC_READ
0001	00000001	KAL_TRUE
0000	00000000	SOC_SUCCESS

```
16、s32 Ql_SOC_GetAckNumber(s32 socketId, u64* ackNum);
```

MOD_TCPIP	MOD_SOC	TCPIP_SOC_SAP	MSG_ID_SOC_TCPIP_CLOSE_IND
MOD_TCPIP	MOD_RATDM	TCM_TCPIP_SAP	MSG_ID_PS_DATA_REQ
MOD_SOC	MOD_QL	SOC_APP_SAP	MSG_ID_APP_SOC_NOTIFY_IND
MOD_APPTCPIP			opencpu_ind_socnotify pre
MOD_APPTCPIP			opencpu_callback_socket_close contextid=0, sock=1, result=0, error_cause=-15, detail_cause=0
!!!			
	Dec	Oct	Bit
1		0001	00000001
12		0000014	00000000000001100
1		0001	00000001
16		0020	00010000
0		0000	00000000
-15		0361	11110001
0		00000000000	00000000000000000000000000000000
			SOC_CLOSE KAL_FALSE SOC_CONNRESET



- 1) 退出 socket
- 2) 删除 socket
- 3) 复制 callback 接口到全局变量中

### 1.3、参考用例

Example\_tcpclient.c ----->TCP 长连接  
Example\_tcpdemo.c ----->TCP 短连接  
Example\_tcpserver.c ----->TCP, 模块作为服务器

Example\_udpclient.c ----->UDP, 模块作为客户端  
Example\_udpserver.c ----->UDP, 模块作为服务器

### 1.4、常见客户问题

#### 1.4.1、域名异常导致解析失败

**问题现象:** 使用 Ql\_IpHelper\_GetIPByHostName 返回域名解析失败

**问题分析:** 使用 nslookup 工具检查当前域名是否正常, 如果异常反馈给客户的网络管理员

```
C:\Users\allan>nslookup www.baidu.com
服务器: UnKnown
Address: 192.168.23.251

非权威应答:
名称: www.a.shifen.com
Addresses: 115.239.211.112
          115.239.210.27
Aliases: www.baidu.com
```

正常

```
C:\Users\allan>nslookup www.test123.com
服务器: UnKnown
Address: 192.168.23.251

DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
*** 请求 UnKnown 超时
```

异常

**问题原因:** 客户的域名异常

**1.4.2、主动执行 QI\_GPRS\_Deactivate 会上报 callback，主动 closed socket 不会上报 callback**

**1.4.3、当使用 UDP 时，服务器发送的数据超过 RCV\_LEN 的数据时，调用 QI\_SOC\_RecvFrom 来获取 RCV\_LEN 大小数据，直接返回-2。**

MTK 接口内部处理，当一次获取的数据 LEN 小于实际上本地 buffer 中已经保存的数据时，会返回-2，建议客户增加自己的 RCV\_LEN 来解决这个问题

**1.4.4、TCP/UDP 接收数据的方式有哪些？**

- 1) 非透传模式下有两种方式：直接从串口输出数据和通过 AT 命令提取数据；
- 2) 透传模式下只有一种方式：直接从串口输出数据。

**1.4.5、 TCP 连接中模块的 IP 地址会如何变化？**

模块和服务器建立 TCP 连接之后，模块的 IP 地址是不会变的，可以通过 AT+QILOCIP 获得。如果执行 AT+QICLOSE 后，再通过 AT+QIOPEN 建立连接 IP 地址也不会变。但是执行 AT+QIDEACT 后，再通过 AT+QIOPEN，IP 地址就会变化。原因是 AT+QICLOSE 只是把当前连接给关闭了，但是 PDP 场景还是激活的，而 AT+QIDEACT 则把当前场景给关闭了，再次连接时需要重新激活场景，激活场景的时候会重新获得 IP 地址，至于网络分配给模块的 IP 地址是根据运营商的配置而决定的。

**1.4.6 模块 QI\_SOC\_Send 返回正确，但是为什么服务器还没有收到发送过去的的数据？**

QI\_SOC\_Send 返回正确的 length,表示模块上层发送的数据已经送入了底层 TCP socket 对应的 buffer 中，这并不代表数据已经发送到了服务器，如果想知道数据是否发送到了服务器可以通过 QI\_SOC\_GetAckNumber 查询。

**1.4.7 模块在哪些情况下会上报 CLOSED？**

模块在下面三种情况下会上报 CLOSED：

- 1) 服务器主动关闭了 TCP 链接；
- 2) 网络异常发送 RST 包中断模块和服务器的 TCP 链接。
- 3)其他

## 2、QuecLocator

### 2.1、调用基站定位接口之前，需要设置场景

### 2.2、我司数据库没有当前模块注册的基站位置信息，导致无法定位成功

目前也没有好的办法，联系我们的数据库维护的同事，把相关的数据信息手动添加

### 2.3、位置信息的精度

一般在市内可以达到 1KM 以内，郊区大概 3-5KM.

注：模块获取的是当前注册的基站位置信息，并不是模块的当前位置信息

## 3、QNTTP

### 3.1、客户使用 AT+QNTTP 返回-2

问题原因：网上公用的 NTP 服务器故障率很高，有些服务器连续连接会出现连接不上。

问题解决：推荐客户使用以下阿里云的 NTP 服务器。

```
ntp1.aliyun.com  
ntp2.aliyun.com  
ntp3.aliyun.com  
ntp4.aliyun.com  
ntp5.aliyun.com  
ntp6.aliyun.com  
ntp7.aliyun.com
```

注：客户可以购买收费的 NTP 服务器

### 3.2、网上免费 NTP 服务器不稳定，客户端应对方法推荐

- 1、尽量使用域名的 NTP 服务器，避免 IP 更改不能使用
- 2、可以设置多个 NTP 服务器，作为备用
- 3、可以自己搭建一个时间服务器，通过 TCP 方式获取当前的时间。

## 4、QNITZ

### 4.1、模块注册成功，并未发现有时间同步

部分地区运营商不支持该功能，具体可以咨询当地运营商

### 4.2、什么时候模块会上报时间同步的 URC

每次模块重新注册网络后，会上报时间同步的 URC

**注：**部分地区的运营商会连续上报两次 URC

### 4.3、NITZ 功能是否需要 GPRS 流量

不会产生数据流量

### 4.4、QNITZ 的 URC 上报

有时刚配置的需要先固定波特率 115200

## 5、HTTP

### 5.1、我们仅有一路 http 资源，同时使用两路会报错。

比如进行 http post 的时候，还通过基站定位获取位置信息。基站定位使用的也是 http，所以基站定位会报错、失败。

### 5.2、通过 HTTP 发送中文到服务器。

通过如下方式实现：GET 请求的数据会附在 URL 之后，以?分割 URL 和传输数据，参数之间以&相连， 如：

<http://api.efxnow.com/DEMOWebServices2.8/Service.asmx/Echo?Message=%E4%BD%A0%E5%A5%BD>。如果数据是英文字母/数字，原样发送，如果是空格，转换为+，如果是中文，则把字符转换为 UTF-8，每个字符前面加上'%'。如 你好，则得出：%E4 %BD%A0%E5%A5%BD。客户照此方法成功实现功能。

### 5.3、opencpu 中使用 HTTP POST，一次最多只能 POST 2K 字节。

我们通过调用 QI\_UART\_Write 经过 VIRTUAL\_PORT3 向内核中写数据，而虚拟串口的 buffer

上限为 2K(2048).

## 5.4、HTTP 发送 json 格式数据

如果用户需要发送 json 格式的数据包，用户需要自己来组包，然后通过 opencpu 接口发送出去。

如下的组包格式：

```
POST /processorder.php HTTP/1.1
Host: demo-myiothub.azure-devices.net
Accept: */*
User-Agent: QUECTEL_MODULE
Connection: Keep-Alive
Content-Type: application/json
Content-Length:128
```

```
{"HTime\":\"0\",\"ID\": \"8\",\"Temperature\": \"25\",\"Pressure\": \"65\",\"Latitude\": \"19.126480\",\"Longitude\": \"73.01101\", \"Acceleration\":\"0\"}
```

## 5.5、POST 过后，换成不同的 URL download 失败

在 post 的前配置了用户自定义的 http 头，然后进行 download 的时候没有把这个配置取消，导致连接的时候被拒绝，连接失败返回 3815。

# 6、SMTP

## 6.1、客户反馈无法使用 M35F 模块通过 gmail 无法正常发送邮件，返回-554

Gmail 服务器限制在 NAME 中出现“@”字符，其他特殊字符可能也有限制，需要实际测试。

# 7、其他

## 7.1、如何锁频点

```
[2017-07-12_10:28:35:853]AT+QOPS -----搜索当前附近基站信息
[2017-07-12_10:28:56:035]+QOPS: 2,"CHINA UNICOM GSM","UNICOM","46001"---联通
[2017-07-12_10:28:56:035]1,5504,2B53,37,40,111-----这些都是频点信息
[2017-07-12_10:28:56:097]2,5504,56E3,27,36,109
[2017-07-12_10:28:56:097]3,5504,2B55,34,35,123
[2017-07-12_10:28:56:097]4,5504,56E1,35,31,114
```

[2017-07-12\_10:28:56:097]5,5504,44A3,2C,30,643  
[2017-07-12\_10:28:56:097]6,5504,56E2,34,28,121  
[2017-07-12\_10:28:56:097]7,5504,582B,1C,30,110  
[2017-07-12\_10:28:56:097]8,5504,47A7,35,28,122  
[2017-07-12\_10:28:56:097]9,5504,2871,1F,28,120  
[2017-07-12\_10:28:56:097]10,5504,2B54,1C,27,118  
[2017-07-12\_10:28:56:097]+QOPS: 3,"CHINA MOBILE","CMCC","46000"---移动  
[2017-07-12\_10:28:56:105]1,550A,37A8,30,47,48  
[2017-07-12\_10:28:56:105]2,550A,01FB,0F,38,82  
[2017-07-12\_10:28:56:105]3,550A,6D46,1E,57,5  
[2017-07-12\_10:28:56:105]4,550A,2BB9,3E,36,45  
[2017-07-12\_10:28:56:105]5,5665,01EB,0E,34,84  
[2017-07-12\_10:28:56:105]6,550A,2E3C,22,33,88  
[2017-07-12\_10:28:56:105]7,550A,5DF7,20,34,1  
[2017-07-12\_10:28:56:105]8,550A,2BB7,0F,39,90  
[2017-07-12\_10:28:56:105]9,550A,37A6,37,32,51  
[2017-07-12\_10:28:56:105]10,550A,3A40,2B,31,89  
[2017-07-12\_10:28:56:105]+QOPS: 1,"46020","46020","46020"  
[2017-07-12\_10:28:56:114]1,4103,0105,2C,5,1015  
[2017-07-12\_10:28:56:114]2,4103,0202,2D,8,1005  
[2017-07-12\_10:28:56:114]OK

[2017-07-12\_10:30:12:461]AT+QLOCKF=2,0,114  
---模块锁到 114 频点上，参数 2 的意思:开启锁频功能并开机自动切换到上次锁定的频点  
[2017-07-12\_10:30:12:461]OK  
[2017-07-12\_10:30:23:656]AT+QPOWD=0 ---关机  
[2017-07-12\_10:30:23:656]OK  
[2017-07-12\_10:30:47:326]AT+CGREG?  
[2017-07-12\_10:30:47:326]+CGREG: 0,1----已注册上网

[2017-07-12\_10:30:47:326]OK  
[2017-07-12\_10:31:09:635]AT+QENG=1,0 ----查询当前小区  
[2017-07-12\_10:31:09:635]OK  
[2017-07-12\_10:31:11:752]AT+QENG?  
[2017-07-12\_10:31:11:752]+QENG: 1,0

[2017-07-12\_10:31:11:752]+QENG: 0,460,01,5504,56e1,114,53,-83,59,59,5,12,x,x,x,x,x,x  
-----当前模块注册频点为 114  
[2017-07-12\_10:31:11:752]OK

# 外设部分

## 1、GPIO

## 2、ADC

### 2.1、ADC 使能后，没有 callback 上报

我们的 ADC 检测是通过发消息的机制来触发的，当 ADC 检测到数据，会发送消息到注册 ADC 的 task，在这个 task 中，客户必须调用接口（QI\_OS\_GetMessage）来处理消息，才会上报 callback

注：软件 timer、普通 EINT、也是同样的机制，硬件 timer 和 fast eint 直接中断触发，不过发送消息

### 2.2、ADC 是否可以采集大于 2.8V 的电压

可以，但是用户需要外部做分压处理，保证到达模块 ADC0 引脚的电平在 0-2800mv 之间，通过比例间接算出当前电压。

### 2.3、ADC 采样是多少位的 D/A

10bit

## 3、ENIT

### 3.1、模块支持的最小消抖时间是多少？

50ms,即模块只能检测到大于 50ms 的电平变化

### 3.2、我们的 EINT 是电平还是上升沿触发

电平触发

两者区别：

电平触发：在高或低电平保持的时间内触发

边沿触发：由高到低或由低到高这一瞬间触发

### 3.3、高/低电平是否都能触发中断

是的，只要电平发生跳变，并且大于消抖时间，都会触发中断，如果客户需要其中的一个电平，可以在 callback 中做个逻辑，只当高/低电平时才做业务。

### 3.4、注册 EINT 时返回-16

在注册 EINT 之前，这个引脚已经做其他功能，比如 GPIO，SD 等

**注：**用户受其他 MCU 外部中断配置影响，设置 EINT 之前，需要先初始化 GPIO，我们 opencpu 是不需要的。

## 4、PWM

### 4.1、如何输出指定频率 PWM 波形

$PWM\ frequency = (pwmSrcClk / pwmDiv) / (lowPulseNum + highPulseNum)$ .

PWM frequency: 频率

pwmSrcClk: 时钟源

pwmDiv: 分频

lowPulseNum: 低电平比例

highPulseNum: 高电平比例

### 4.2、opencpu 中仅支持一路 PWM（NETLIGHT 引脚）

## 5、IIC

### 5.1、返回-34 错误

- 1、客户的硬件连接有问题，比如虚焊等
- 2、从器件地址错误

**举个例子：**

使用芯片的 datasheet 信息



Table 9.I2C Address

SAD6	SAD5	SAD4	SAD3	SAD2	SAD1	SAD0	W/R
0	1	0	0	1	1	SAO	0/1



Table 10.SAD+Read/Write patterns

Command	SAD[6:1]	SAD[0]=SA0	R/W	SAD+R/W
Read	010011	0	1	01001101(4dh)
Write	010011	0	0	01001100(4ch)
Read	010011	1	1	01001111(4fh)
Write	010011	1	0	01001110(4eh)

**应该使用的器件地址:**我们的 IIC 是需要传入 8 位(0X4C)地址的,有些客户只传入 7 位(0X26) 没有包含最后一个读写位。

## 5.2、代码中对于读写接口是否要写入不同的地址（根据最后一个读写位）

不需要,我们的代码会根据你调用的接口不同,来更改最后一位的值,用户只需要统一传入 0 或者 1 即可

## 5.3、如何实现控制多路的 IIC

我们最多支持 **6 路**的 IIC,客户初始化（仅一次）后,需要支持多少路 IIC 就配置几次即可。  
**注意:**如果客户需要挂载多路的 IIC,需要确定这几路的 IIC 器件地址必须不同,因为我们是根据不同的器件地址来进行操作的。

## 5.4、IIC 操作时,返回-34,从波形上面可以看出返回的 ACK 大约有 1v 左右

上拉电阻选择过大,把 10K 电阻更改成 4.7K

# 6、SPI

## 6.1、使用我们 example\_SPI,读取寄存器数据全为 0

客户硬件电路设计时,MISO 和 MOSI 引脚设置反了

**注:**一般芯片上标注 DO(MISO)、DI(MOSI)

## 6.2、使用我们的 SPI 时,写入 512 个字节后,读出来时,只有 256 个字节是成功的

查看 datasheet 发现芯片规定一页为 256 个字节

6.3、SPI 可以在 Init 时定义一个 clk 管脚，只是不要使用特定的 PINNAME\_PCM\_CLK。然后通过操作 CS 管脚选择不同的设备，从而实现用硬件的管脚和硬件 spi 挂载多个设备。SDK 中 Config 时 channel 只能是 1 应该是描述错误这个 channel 可以 0~254，但必须和 init 时一致。

## TIME

### 1、timer

#### 1.1、简介

我们 2G 模块，提供了两种类型的 timer。一种是软件 timer，一种是快速 timer（GP\_TIMER）。在 opencpu 中每个 task 可以最多同时使用 10 个软件 timer，一个工程只能有 1 个 GP\_TIMER。软件 timer 是通过传递消息来调用的，GP\_TIMER 直接通过中断来触发的，所以优先级要比软件 timer 高。

#### 1.2、常见客户问题

##### 1.2.1、软件 timer 定时不准的问题

因为软件 timer 是通过发送消息的方式来触发 callback，所以如果 timeout 后，当前 task 不能及时处理 message，将会导致定时延迟。

比如代码中有如下操作会造成延迟：

- 1、Ql\_Sleep()
- 2、Mutex、EVENT 等阻塞接口
- 3、复杂业务'

##### 1.2.2、GP timer 频繁开关导致无法正常工作

##### 1.2.3、GP timer 的 callback 中不能增加锁操作

MTK 规定禁止在 HISR 里获取互斥量时等待，比如做锁的操作  
这个问题导致的死机，可以通过抓取 dump 信息看到提示：

```
== 异常报告 ==
详细描述: 禁止在HISR里获取互斥量时等待
软件版本: MC20CBR01A02_TTS
硬件版本: MC20
当前进程: DRVHISR -> 请查看进程分析部分
```

```
== 进程检查 ==
HISR: DRVHISR (激活1次, 优先级:1, 栈最大使用66%)
调用栈: SP: 0xF01D0850, LR: 0x10034AD9, PC: 0x10034AD9
        Callback_GPTimer(参数2:0x10034ACD = Callback_GPTimer(), 参数3:0xA0050180, 参数4:0x4000)
        GPTI_HISR(参数1:0x10025E5D = GPTI_HISR(), 参数4:0x10034ACD = Callback_GPTimer())
        DRV_HISR(参数4:0)
        TCT_HISR_Shell()
        == 栈结束 ==
```

**注:**我们这个打印接口 (QI\_Debug\_Trace) 底层实现有增加锁, 所以不能在 GP\_callback 中增加。

#### 1.2.4、stop timer 时返回-4

**问题原因:** 用户在 mian task register 的 timer, 然后再 subtask 中 stop, 发现不起作用, 返回 -4 的 error

**注:** 我们的软 timer 都是根据 task id 来操作的, 所以需要在当前 task 来 register start stop 但是 GP timer 可以在任意的 task 中操作, 不受这个限制。

#### 1.2.5、如何获取准确的时间计数

**问题现象:** 用户想通过 timer 来确认 A 动作开始到 A 动作结束一共花费多少 ms

**问题分析:** 如果客户对时间精度要求过高不可选用软件 timer/GP\_timer 来做。因为:

- 1、我们 timer 的时间单位是 4.6125ms, 误差本身就很大
- 2、软件 timer 如果 task 阻塞时, 会延迟, 不会及时处理

**问题解决:** 用户可以通过接口 (QI\_GetMsSincePwrOn) 来获取 A 动作一共花费 ms 数。

**注:** 此接口存在当超过 0x7CF FFFF 会溢出, 重新从 0 开始计时。这个问题最新版本 (参考 RN) 的固件已解决。

## 2、Alarm

### 2.1、一个任务中最多可以同时建立几个闹铃?

一个, 如果建立的闹铃未删除或者虽然删除但是新建的闹铃和上一个闹铃时间重合, 都不能成功。

### 2.2、如果想连续设置闹铃, eg: 每隔 2、4、6 小时响铃.....如何实现?

每次闹铃 ring 以后, 删除该闹铃, 设置下一个闹铃

## 2.3、两个或以上任务可以同时建立相同或不同的闹铃吗？

不能，同一时间内 OCPU 只能建最多一个闹铃，理由参考 2.2.1

## 2.4、模块处于睡眠模式下，闹铃是否能够唤醒

可以，URC 正常上报 ring。

## 2.5、设置闹铃的时间要大于本地当前时间

此处和设置时间是否一致无关，设置闹铃的时间要大于本地当前时间，内核代码中做了限制。

## 2.6、实现定时开关机操作

定时开机，需要给模块持续供电

## 2.7、设置 Alarm 时，年份格式注意是两位

Eg:

AT+QALARM=0,"2014/12/01,10:01:10+02",1,0 格式错误，年份格式出错

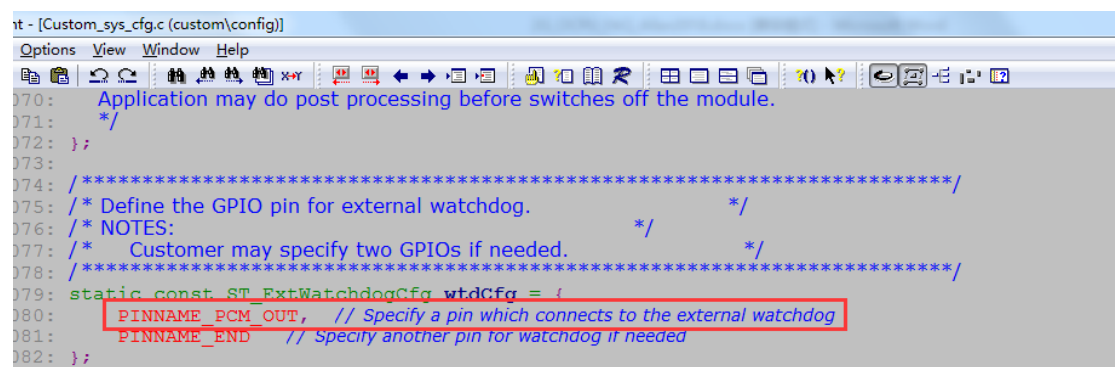
AT+QALARM=0,"14/12/01,10:01:10+02",1,0 格式正确，年、月、日

# FILE 部分

## 1、客户问题

### 1.1、M66 模块在 OPENCPU 中读取 SD 文件返回-35。

SDK 中默认的看门狗引脚 PCM\_OUT,导致 SD 的引脚被占用，无法正常操作。把看门狗喂狗引脚更换成其他引脚，可以解决问题。



```
it - [Custom_sys_cfg.c (custom\config)]
Options View Window Help
Application may do post processing before switches off the module.
070:  */
071:  */
072:  };
073:
074:  /**
075:   * Define the GPIO pin for external watchdog.
076:   * NOTES:
077:   * Customer may specify two GPIOs if needed.
078:   */
079:  static const ST_ExtWatchdogCfg_wtdCfg = {
080:      PINNAME_PCM_OUT, // Specify a pin which connects to the external watchdog
081:      PINNAME_END // Specify another pin for watchdog if needed
082:  };
```

## 1.2、最大支持多大的 SD 卡容量

### 3.9. SD Card Interface

The module provides an SD card interface that supports many types of memory, such as Memory Stick, SD/MCC card, and T-Flash or Micro SD card. The following are the main features of SD card interface.

- Only support 1bit serial mode
- Not support the SPI mode for SD memory card
- Not support multiple SD memory cards
- Not support hot plug
- The data rate up to 48MHz in serial mode
- Up to 32GB maximum memory card capacity

## 1.3、最大支持多少个文件

最大可以增加多少文件个数，我们并没有限制，只要不超过文件系统的内存空间即可。

但是同时最大只能打开 15 个文件。

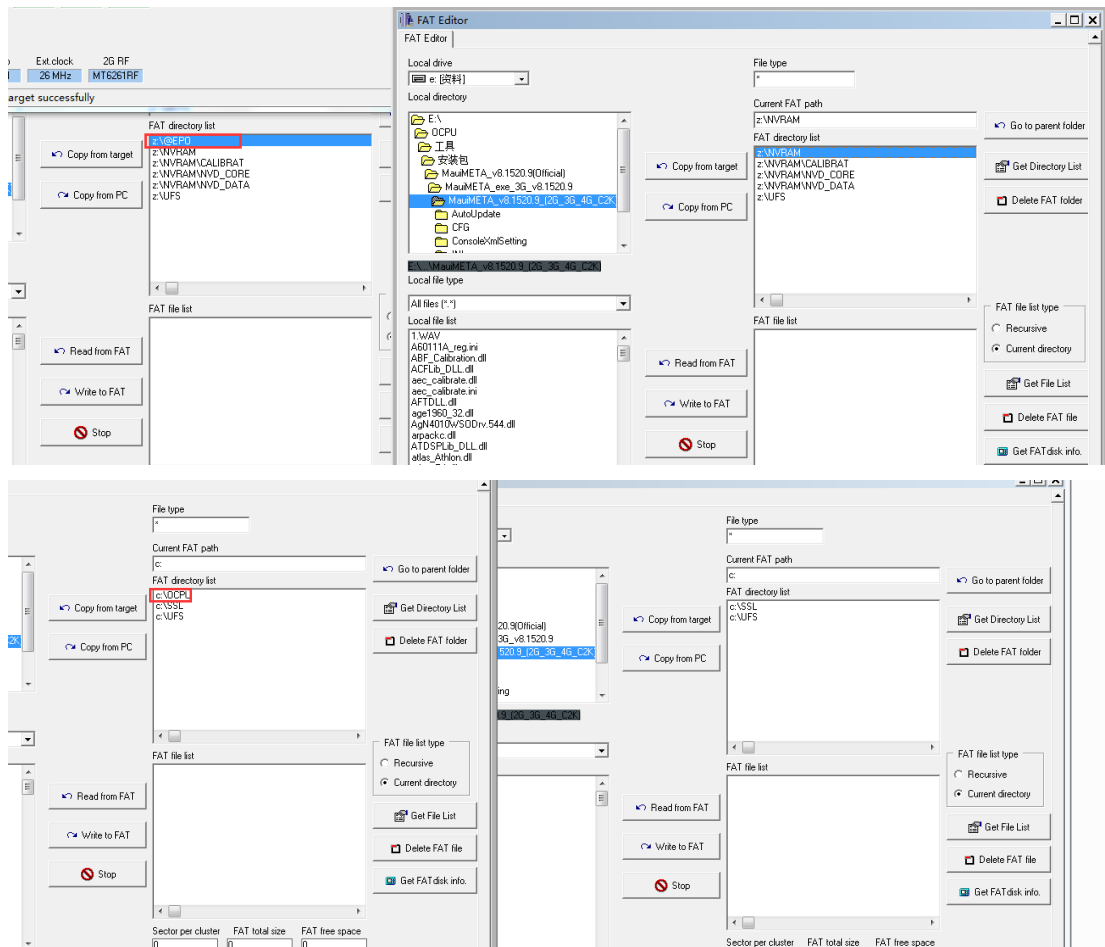
## 1.4、当调用 QI\_FS\_GetSize 接口时，返回-35

当调用 QI\_FS\_GetSize 接口时，如果 file 已经打开，需要先 closed file，因为在这个接口中，也会调用 open 接口。

## 1.5、当文件不存在时，open 时，需要增加 create 的属性

## 1.6、相同版本烧录后发现 UFS 剩余空间相差几百字节

通过 META 工具可以看出，其中一个模块跑过 opencpu，里面包含了 open 和 EPO 的路径，这个路径是不会删除的，所以占用了几百字节。



**META 工具使用方法：**选择 COM 口后，点击 **Reconnect**，重启模块即可

## 1.7、使用已经初始化的 SD 卡，发现已经默认生成” picture” 文件夹

我们的模块会默认生成这个文件夹，用来给 MMS 使用

## 1.8 模块 FLASH 的使用寿命

一般情况下，Flash 的擦除寿命大约是 10 万次。为了延长 FLASH 读写寿命，我们模块采用 **动态平均读写技术**，可以将数据平均写入到 FLASH 的不同区域中，而避免因重复写入同一区域可能造成的 FLASH 损坏。

因此，FLASH 的每个存储区域都获得了相同的使用机会，从而延长了使用寿命。

# AUDIO 部分

## 1、客户问题

### 1.1、部分模块不支持把音频添加到代码中播放的功能。

由于空间不足，（M26、MC20E）不支持把音频添加到代码中播放的功能（AT+QPLAYMEM）

注：具体还需查询项目如下的宏开关

`_QUECTEL_AUDIO_RESOURCES_FLAG_`

### 1.2、操作 record 功能时，返回-1

我们 example 默认是保存在 UFS 中，而 M26 没有 UFS，所以会报错，改成 RAM:后成功。

### 1.2、保存录音文件到 SD 卡中，返回 error

路径出错，这个路径 SD: \\test.amr 应该改为 SD: test.amr

### 1.3、TTS 和通话优先级哪个高？

TTS 播放过程中如果来电，TTS 会中断播放，并且当来电结束后 TTS 也不会再继续播放；目前 Quectel 模块支持在通话中播放 TTS 文本，通过 AT+QWTTS 命令实现，双方都可以听到，而且播放内容支持 ASCII 和 UNICODE 格式。

### 1.4、通过 spi 外扩 flash，其中的音频文件不能直接播放，需要拷贝至我们的文件系统中播放

# GNSS 部分

## 1、简介

MC20/MC60 模块集成了 GSM 和 GNSS 双系统，在网络交互的同时，实现 GNSS 系统的快速、精准定位；同时模块内置了业界领先的 EPO 技术，大大减少了模块在冷启动模式下所需的定位时间；借助 Reference-location 信息，还可以实现定位更为迅速的秒定功能。关于 EPO

和秒定的流程，请用户严格按照我们 SDK 中关于相关内容介绍的文档。

## 2、常见客户问题

### 2.1、GNSS 返回 7103 错误

一般原因如下：

- 1、AT+QGNSSSC=1 后，需要等待 3s 才能读取 NMEA 语句
- 2、GPS 和 GSM 串口未正常连接
- 3、GSM 发生重启后，GPS 还在正常工作的情况。可以监控 GPS 的 TX 看看有没有 NMEA 数据吐出。

### 2.2、GNSS 比 GSM 先上电，导致异常

如果 GPS 比 GSM 先上电，会导致 PMTK 命令无法发送到 GPS 模块执行(有 OK 返回，但是没有 PMTK 开头的返回，说明没有执行成功)，EPO，秒定功能无法使用。

### 2.3、使用坐标系

WGS-84 坐标系

### 2.4、EPO 导致数据流量明显增加

如果客户的本地时间更新策略有误，导致 EPO 数据经常需要更新，就会导致模块的数据流量明显增加

EPO 数据大小：

6H数据：4KB

6天数据：96KB

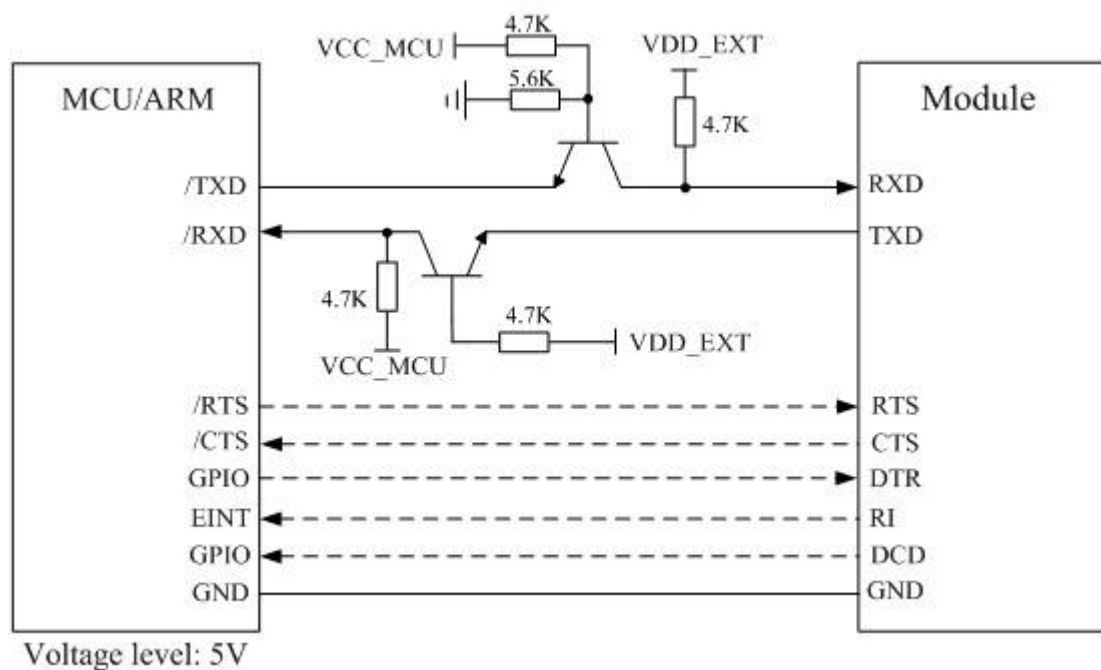
### 2.5、opencpu 模式只支持 All-in-one 模式

All-in-one 方式，GNSS 的主 UART 与 GSM 的 AUX port 连接，NMEA 语句是通过 GSM 的主串口发 AT+QGNSSRD?或 AT+QGNSSRD="NMEA/GGA"来读取；OCPU 下直接通过 RIL API 或 Callback 获取。

### 2.6、秒定的定位耗时

OpenSky 环境下，GNSS 在冷启动模式下定位耗时约为 4.5 秒（参考值）。





## 2.7、冷启动、温启动、热启动

TTFF	Cold start	Warm start	Hot start
Almanac data	Not valid	Valid	Valid
Ephemeris data	Not valid	Not valid	Valid
Position data	Not valid or >100KM	position<100kM,no accurate view of satallite	Valid
UTC time	Not valid	Accuracy<20s Restart time>2H	Restart time<2H
Standard Time(-130dBm)	<35s	≤30s	≤1s

## 2.8、定位点总是与实际位置偏差几百米或几千米。

1. 是否使用的是百度地图或其他国内地图，这个直接输入经纬度会有偏差，因为有规定国内必须有偏差，需用相关地图的 API 进行误差消除。

2. 导入 google 地图时，需要使用正确的坐标单位,要么设置 google 地球的单位为度分，要么进行如下转换：

纬度： ddmm.mmmm

经度： dddmm.mmmm

如果需要转换为度格式，如下转换：

纬度 =  $dd + (mm.mmmm/60)$  度，同时注意北纬是正数，南纬是负数

经度 = ddd + (mm.mmmm/60) 度，同时注意东经是正数，西经是负数

## 2.9、有些客户模块的漂移很大，和什么有关？如何解决？

主要与测试环境（如存在遮挡、多径等）、PCB 天线设计（干扰等）有关。

可在相同环境使用功分器将信号一分为二供客户板子与 EVB 对比测试，若客户板子仍然性能较差，则需要检查 PCB。若都有漂移，则需与竞品对比，并获取竞品的 NMEA 语句和我司模块的 GPS 的 debug log（PMTK299,1 开启 debug 语句输出）。

## 2.10、All-in-one 方式下操作 EPO 有两个操作流程，流程 1 和流程 2 有什么区别？

1.流程 1 是在 GSM 开机上报 RDY 之后即可发送 AT+QGNSSC=1 打开 GPS，使 GPS 提前工作搜星。等到时间同步之后再使能 EPO，使能 EPO 之后必须发送 AT+QGEPOAID 来触发下载和 Aiding（GSM 通过 AUX PORT push EPO 数据给 GPS）。

2.流程 2 是 GSM 开机后并不立即打开 GPS，而是等到时间同步之后使能 EPO，在使能 EPO 之后发送 AT+QGNSSC=1 来下载和触发 Aiding（GSM 通过 AUX PORT push EPO 数据给 GPS）。

## 2.11、AT+QNTF 已经同步时间成功或 AT+CCLK 设置 UTC 时间了，AT+QGNSSST?为什么查询出来的一直是 0？

NITZ 开机会自动同步时间，如果同步成功会修改 AT+QGNSSST?查询的变量为 1；all-in-one 下 GPS 定位了，也会同步时间，同时也会修改该变量为 1；其他同步方式不会修改 AT+QGNSSST?查询的结果。因此，如果运行商不支持 NITZ，GPS 又没有开启或定位，则 AT+QGNSSST?查询的结果会一直是 0。

当 Network 的 CS 域注册上之后，AT+CREG?查询到+CREG: 0,1 时，如果 AT+QGNSSST?仍然查询到的是 0，则表明运营商不支持 NITZ。等到 PS 域注册上之后就可以网络激活并进行 QNTF 同步时间了，当 QNTF 返回成功后，即可进行使能 EPO 操作，因为 EPO 使能的前提是时间已经同步，并非仅仅依赖于 AT+QGNSSST?的查询结果。

## 2.12、使用 EPO 的定位时间

约 15s。

## 2.13、什么是秒定？秒定的定位时间是多长？

在使能 EPO 之前，先通过 AT+QREFLOC=<latitude>,<logitude>设置大概位置（20km 以内）的经纬度，即为秒定流程。秒定在 open-Sky（开阔环境）下大约 5s 定位。

## 2.14、MC20/MC60 模块能否访问固定 IP 地址的服务器来下载 EPO 文件？能否使用第三方服务器下载？

少量特定版本支持。

# 电源部分

## 1、常见客户问题

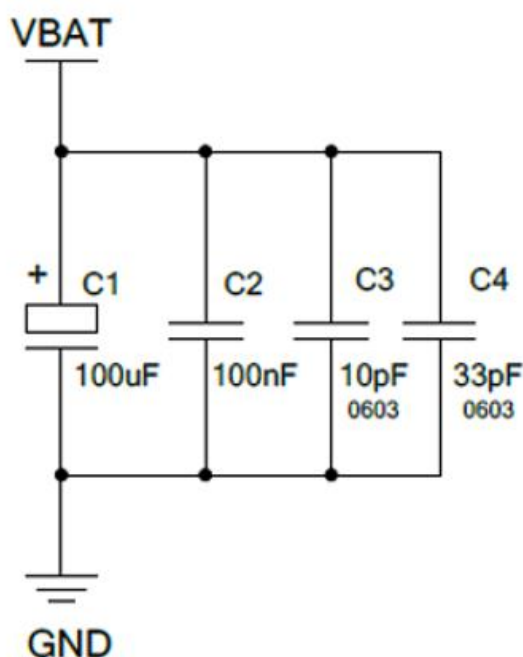
### 1.1、基本要求

对于 2G 模块，在最大发射功率等级下模块的峰值电流会达到 1.6A，这会引起 VBAT 端电压的跌落。为确保模块能够稳定正常工作，建议模块 VBAT 端的最大跌落电压不应超过 400mV。

### 1.2、减少电压跌落

VBAT 电压输入范围为 3.3V~4.6V。为保证 VBAT 电压不会跌落到 3.3V 以下，在靠近模块 VBAT 输入端，建议并联一个低 ESR（ESR=0.7Ω）的 100uF 的钽电容，以及 100nF、33pF（0603 封装）和 10pF（0603 封装）滤波电容。VBAT 输入端参考电路如下图所示。

同时建议 VBAT 的 PCB 走线尽量短且足够宽，以减小 VBAT 走线的等效阻抗，确保在最大发射功率时大电流下不会产生太大的电压跌落。建议 VBAT 走线宽度不少于 2mm。原则上走线越长，线宽越宽。



### 1.3、模块在开机后，不断重启

如果用户的设备出现在开机后的一段时间内，频繁的重启，就需要考虑是否因为在注册网络阶段，由于电压跌落造成的，可以通过示波器来监控 VBAT 的电压。

注：有时为了测试客户设备，外部 VBAT 飞线，需要注意飞线不能过长。

### 1.4 高低压报警、高低压关机

以 MC20CA 模块为例

高压关机：4.6V

高压报警：4.5V （连续报警 6 次，模块关机）

低压报警：3.5V （连续报警 8 次，模块关机）

低压关机：3.1V

注：

- 1、不同型号的数值设置可能不一样
- 2、高低压报警超过一定次数后，模块会关机。

3、现在代码中超过 4.5V 会有 WARNING 提示，超过 4.6V 会有 POWER DOWN 提示并且 6 次后会关机。低于 3.5V 会有 WARNING 提示，超过 3.1V 会有 POWER DOWN 提示并且 8 次后会关机。

### 1.5 能否将 5V 电压通过二极管降压后给模块供电？

不建议采用此方式，因为二极管压降随电流变化而变化，可能会因模块供电电压不稳定而导致模块工作异常。

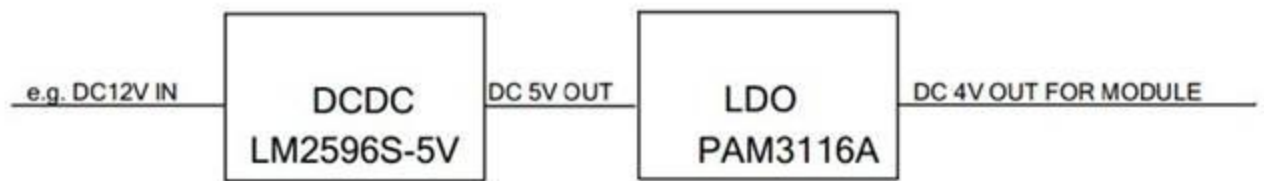
### 1.6 能否通过标准 USB 给模块供电？

标准 USB 接口最大供电电流是 500mA，不能满足模块峰值电流的要求。如果客户一定要选择 USB 供电方式，可以按照以下步骤操作：

- 1) 通过 AT+QGPCLASS 配置模块 GPRS 等级为 8（该命令需要重启模块生效）。GPRS 等级配置为 8 时，uplink slot 只有 1 个。
- 2) 靠近模块 VBAT 端增加 2 个低 ESR 值的 2200uF 大电容。
- 3) 建议将 VBAT 电压调节在 4.2V~4.5V 之间，以增加电压跌落余量。
- 4) 在弱信号环境（比如 CSQ 值小于 14）下测试，如果模块能够正常使用，则表示供电正常

### 1.7 车载 12V 电源给模块供电使用方案？

建议使用两级转换，第一级经 DCDC 转换为 5V，第二级经 LDO 转换为 4V，如图 1 所示。



## 串口部分

### 1、常见客户问题

#### 1.1、串口数量

**MC20/MC60** 系列模块, 由于 AUX 串口默认和 GNSS 的 UART 通讯, 所以只有 MAIN 口和 DEBUG 口两个 UART。

**其他 2G 模块:** 有三个串口: MAIN、DEBUG、AUX

**注:** 为了方便 APP 和内核之间进行通讯, 我们还提供了三个虚拟串口, 其中虚拟串口 3 默认提供给 RIL 接口使用。所以用户如果只需要和底层进行数据交互, 可以使用虚拟串口 1 和虚拟串口 2。

#### 1.2、各串口功能

##### MAIN 串口 (UART1):

- 1、下载 APP BIN
- 2、打印应用 log
- 3、和外设进行通讯

##### Debug 串口 (URAT2)

- 1、抓取 catcher log (advance mode)
- 2、打印应用 log (basic mode)
- 3、和外设进行通讯 (basic mode)

##### AUX 串口 (UART3)

- 1、打印应用 log
- 2、和外设进行通讯

**注:** MC20/MC60 系列模块此口默认和 GPS 连接, 用户不可控制。

##### 虚拟串口 1 (VIRTUAL\_PORT1)

和底层进行数据交互

##### 虚拟串口 2 (VIRTUAL\_PORT2)

和底层进行数据交互

### 虚拟串口 3 (VIRTUAL\_PORT3)

RIL 功能占用，用来交互 AT

## 1.3、串口 buffer 必须要读空

当应用层收到读取串口 buffer 的 callback 时，在 callback 中需要 while (1) 读取 buffer 中的数据，直到把 buffer 数据读空，才返回，否则串口再次来数据将无法接收。可以参考我们 SDK 中的 main.c

## 1.4、底层 buffer 发送消息的条件

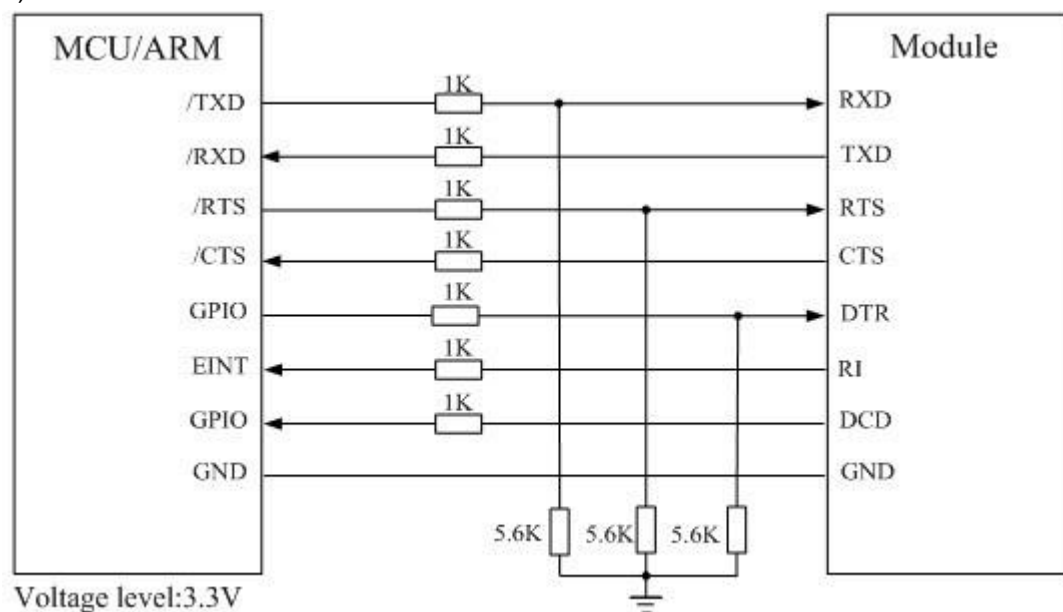
当收到的 buffer 数据超过最大阈值的 2/3 或者连续 4 个字节时间间隔没有收到数据时。

## 1.5、当模块通过 QI\_Sleep\_Enable 进入睡眠模式后，将不再能接收数据

如果用户需要发送数据给模块，可以先通过外部中断，唤醒模块，最好能增加个延迟，然后再发送数据，保证模块当前已经被唤醒。

## 1.6、串口电平如何匹配

1) 如果 MCU 的串口电平是 3.3V，匹配电路如图所示：



2) 如果 MCU 的串口电平是 3V，则将图 3 中的 5.6K 电阻换成 15K。

# 程序设计部分

## 1、常见客户问题

### 1.1、取值范围与打印问题

对应的数据类型需要用对应的类型打印。

例如客户 u32 的数据，范围是 0xFFFFFFFF，打印出来返回-1。客户使用%d、%ld 打印。u32 位无符号 int 型，打印应使用%u。

### 1.2、FTP 下载文件

OPEN 里接口 RIL\_FTP\_QFTPGET，当参数 filesize 设置为 0 时默认最大下载文件大小为 100K，所以当下载大于 100K 文件时应把 filesize 参数做适应性修改（改成对应文件大小即可）。

### 1.3、程序栈溢出

给 task 设置的栈空间小了会导致栈溢出，修改栈空间大小可解决。

### 1.4、域名解析

推荐使用 QI\_GPRS\_GetDNSAddress，然后域名解析。

有的客户使用 QI\_GPRS\_SetDNSAddress 设置 8.8.8.8 或者 114.114.114.114 会出现失败。

### 1.5、关于用户安全数据区

用户安全数据区数据写入后重新烧版本和 app 都不会清掉。

### 1.6、socket 编程

QI\_SOC\_ConnectEx 如果 75s 没有连上会有超时，关闭 socket。

### 1.7、QI\_UART\_SetDCBConfig 设置失败。

使用 API 配置时必须先注册和打开串口才可以配置。