



IAC-IMX8MM-Kit 功能说明与测试手册

版本号：V2.0
2021年04月

浙江启扬智能科技有限公司版权所有

QIYANG TECHNOLOGY Co., Ltd

Copyright Reserved

版本更新记录

| 版本 | 硬件平台 | 描述 | 日期 | 修订人 |
|-----|---------------------|------|---------|-------|
| 1.0 | IAC-IMX8MM-MB V1.00 | 内部版本 | 2020-02 | zhujh |
| 2.0 | IAC-IMX8MM-MB V1.20 | 更新图片 | 2021-04 | wwx |

目 录

| | |
|--------------------------|----|
| 目 录..... | 3 |
| 阅读前须知：本手册主要介绍接口功能测试..... | 4 |
| 一、前言..... | 4 |
| 公司简介..... | 4 |
| 一、准备工作..... | 5 |
| 二、主板测试..... | 6 |
| 2.1、蜂鸣器测试..... | 6 |
| 2.2、时钟测试..... | 7 |
| 2.3、看门狗测试..... | 9 |
| 2.4、CAN 测试..... | 10 |
| 2.5、GPIO 测试..... | 12 |
| 2.6、显示测试..... | 15 |
| 2.6.1 hdmi 显示..... | 15 |
| 2.6.2 mipi-dsi 显示..... | 16 |
| 2.6.3 lvds 显示..... | 18 |
| 2.7、触摸测试..... | 19 |
| 2.8、USB 测试..... | 20 |
| 2.9、WIFI 测试..... | 22 |
| 2.10、蓝牙测试..... | 24 |
| 2.11、4G 测试..... | 28 |
| 2.12、串口测试..... | 30 |
| 2.13、摄像头测试..... | 33 |
| 2.14、音频测试..... | 34 |
| 2.15、SD 卡测试..... | 35 |
| 2.16、网口测试..... | 36 |
| 2.17、屏背光测试..... | 39 |
| 三、测试小结..... | 40 |

阅读前须知：本手册主要介绍接口功能测试

一、前言

公司简介

浙江启扬智能科技有限公司 2007 年成立于杭州，是一家专注于 ARM 嵌入式产品研发、生产与销售的国家高新技术企业。10 余年的积累与沉淀，成功构建了产品从开发到量产的服务链。

作为公司的核心，启扬研发团队由 30 余位嵌入式工程师组成，致力于为用户提供简单易用的嵌入式硬件、软件工具以及定制化的产品解决方案。已广泛应用于工控、物联网、新零售、医疗、电力、环境监测、充电桩等领域。

设立于诸暨的生产基地为启扬提供了强有力的保障，占地面积 5000 平米，拥有 2 条 SMT 产线，通过并严格遵循 ISO9001 质量管理体系认证指导生产。依托雄厚的生产实力，年产能可达 100 万套，保证用户交期，解决后顾之忧。

启扬拥有完善的销售市场网络，专业的销售和售后团队为用户提供全方位的技术支持与服务。业务已遍及 120 多个国家和地区，成功帮助 2000 多家用户将产品快速高效地推向市场。

研发、产能、市场的结合与延伸，为启扬智能成为专业化、全球化的嵌入式软硬件供应商奠定了坚实的基础。

我们为您提供：

- **多平台软/硬件产品**

NXP、Rockchip、MTK、Renesas、TI、Atmel、Cirrus Logic 等多平台 ARM 开发板/核心板/工控板和周边硬件产品以及支持用户快速二次开发的配套工具与软件资源。

- **定制服务**

充分发挥在 ARM 平台及 Linux、Android、Ubuntu 操作系统上的技术累积，为用户提供量身定制嵌入式产品服务（OEM/ODM）。

感谢您使用启扬智能的产品，我们会尽最大努力为您提供技术协助！祝愿您工作顺利！

一、准备工作

启动系统，连接串口，通过调试串口进入到板子的文件系统中。

```
[ OK ] Started Permit User Sessions.
[ OK ] Started /etc/rc.local Compatibility.
[ OK ] Started Getty on tty1.
[ OK ] Started Serial Getty on ttymxc1.
[ 9.129325] lan743x 0000:01:00.0 eth1: Link is Down
[ 9.129942] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not ready
[ OK ] Reached target Login Prompts.
       Starting Weston Wayland Compositor (on tty7)...
[ OK ] Started Hostname Service.
[ OK ] Started Kernel Logging Service.
[ OK ] Started Weston Wayland Compositor (on tty7).
[ OK ] Reached target Multi-User System.
       Starting Update UTMP about System Runlevel Changes...
[ OK ] Started Update UTMP about System Runlevel Changes.

NXP i.MX Release Distro 4.14-sumo
"*****"
"Version: 2021-01-[ 9.378685] audit: type=1006 audit(1571622066.708:2): pid=
=4294967295 auid=0 tty=(none) old-ses=4294967295 ses=1 res=1
28"
"*****"
imx8mmqiyang ttymxc1

Last login: Mon Oct 21 01:41:06 UTC 2019 on tty7
root@imx8mmqiyang:~# █
```

主板测试程序位于/usr/test目录下，进入该目录，后续测试都在该目录下进行。

```
cd /usr/test/
```

```
ls
```

二、主板测试

2.1、蜂鸣器测试

IAC-IMX8-MB 主板使用 GPIO 4_27 控制主板上的蜂鸣器。

设置为低电平时，蜂鸣器不工作；设置为高电平时蜂鸣器鸣响。

测试原理：

该项测试主要实现蜂鸣器鸣响。

测试步骤和结果：

运行蜂鸣器测试程序 `buzzer_test`

```

root@imx8mmqiyang:/usr/test# ./buzzer_test
Invalid arguments!
Usage: ./buzzer_test <device> [0|1]
  <device> -- for example: /dev/qiyang_buzzer
           0    -- buzzer off.
           1    -- buzzer on.

root@imx8mmqiyang:/usr/test# █
    
```

说明：`buzzer_test <device> 0` 蜂鸣器不工作，可以关闭蜂鸣器

`buzzer_test <device> 1` 蜂鸣器鸣响

1. 打开蜂鸣器，听到主板上的蜂鸣器持续发出蜂鸣声。

```
# ./buzzer_test /dev/qiyang_buzzer 1
```

2. 关闭蜂鸣器。

```
# ./buzzer_test /dev/qiyang_buzzer 0
```

设备节点：

`/dev/qiyang_buzzer`

测试源码：

光盘/测试源码/buzzer_test/buzzer_test.c

驱动代码：

`imx_4.14.98_2.0.0_ga/drivers/misc/buzzer.c`

内核对应选项：

`CONFIG_FSL_IMX8_BUZZER=y`

2.2、时钟测试

IAC-IMX8-MM 主板使用 I2C2 连接底板上的 PCF8563 芯片作为外部硬件时钟，请在测试 RTC 之前，确保已经安上电池。

测试原理：

通过 date 系统命令设置系统时间，用 hwclock 命令把系统时间写入硬件时钟，通过 rtc_test 测试程序读取硬件时钟并打印出来，断电后重启，查看时钟是否准确。

测试步骤和结果：

1. 在板子上执行 date 命令，可查看到当前系统时钟。

date

```
root@imx8mmqiyang:/usr/test# date
Mon Apr 20 06:31:56 UTC 2020
root@imx8mmqiyang:/usr/test#
```

2. 用 date 命令设置系统时钟，比如按当前 PC 的显示时间来设置。

date -s "2020-04-20 14:33:15"

```
root@imx8mmqiyang:/usr/test# date -s "2020-04-20 14:33:15"
Mon Apr 20 14:33:15 UTC 2020
```

3. 用 hwclock 命令把系统时间写入硬件时钟芯片。

hwclock -w

4. 分别用 date 和 hwclock 命令来查看系统和硬件时钟。

```
root@imx8mmqiyang:/usr/test# hwclock -w
root@imx8mmqiyang:/usr/test# date
Mon Apr 20 14:33:58 UTC 2020
root@imx8mmqiyang:/usr/test# hwclock
Mon Apr 20 14:34:02 2020  0.000000 seconds
root@imx8mmqiyang:/usr/test#
```

5. 设置成功之后，执行 rtc_test 测试程序。

./rtc_test /dev/rtc0

```
root@imx8mmqiyang:/usr/test# ./rtc_test /dev/rtc0

          RTC Driver Test Example.
Current RTC date/time is 2020/4/20, 14:34:20.
Current RTC date/time is 2020/4/20, 14:34:21.
Current RTC date/time is 2020/4/20, 14:34:22.
Current RTC date/time is 2020/4/20, 14:34:23.
Current RTC date/time is 2020/4/20, 14:34:24.
Current RTC date/time is 2020/4/20, 14:34:25.
Current RTC date/time is 2020/4/20, 14:34:26.
Current RTC date/time is 2020/4/20, 14:34:27.
Current RTC date/time is 2020/4/20, 14:34:28.
Current RTC date/time is 2020/4/20, 14:34:29.

          *** Test complete ***
root@imx8mmqiyang:/usr/test# █
```

程序打印 10 条当前硬件时间后退出程序，提前退出程序请按 `ctrl+c`

查看是否精准走时，查看有无出现丢秒现象。

6. 断电，过一会再上电，再次查看系统和硬件时钟，看时间有没保存，走时是否精准。
7. 通过和 PC 时间对比之后，发现时间无误差，若需测试长时间的走时精确度，可以分别断电和上电老化测试几天几周或者几个月来测试时钟误差。
本主板在发布之前都经过 1 个月的老化测试，时间误差不超过 2S
对于发货产品均经过 24 小时以上老化测试，时间误差不超过 1S

设备节点：

/dev/rtc

/dev/rtc0

设备树文件：

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码：

imx_4.14.98_2.0.0_ga/drivers/rtc/rtc-pcf8563.c

内核对应选项：

CONFIG_RTC_DRV_PCF8563=y

2.3、看门狗测试

测试原理：

硬件看门狗，GPIO1_IO05 使能看门狗，GPIO1_IO08 执行喂狗操作。

测试步骤和结果：

- 1、运行 watchdog_feed_test 开发板不重启、退出 watchdog_feed_test 开发板过 1.6s 重启

```
# ./watchdog_feed_test /dev/qiyang_watchdog
```

```
root@imx8mmqiyang:/usr/test# ./watchdog_feed_test /dev/qiyang_watchdog
[ 334.557793] watchdog: enable watchdog
```

- 2、运行 watchdog_notfeed_test 开发板过 1.6s 马上重启

```
# ./watchdog_notfeed_test /dev/qiyang_watchdog
```

```
root@imx8mmqiyang:/usr/test# ./watchdog_notfeed_test /dev/qiyang_watchdog
[ 2763.487865] watchdog: enable watchdog

U-Boot SPL 2018.03-00019-gfab7ffc (Apr 20 2020 - 13:55:50 +0800)
power_bd71837_init
DDRINF0: start lpddr4 ddr init
DRAM PHY training for 3000MTS
check ddr4_pmu_train_imem code
check ddr4_pmu_train_imem code pass
check ddr4_pmu_train_dmem code
check ddr4_pmu_train_dmem code pass
Training PASS
DRAM PHY training for 400MTS
```

设备节点：

/dev/qiyang_watchdog

测试源码：

watchdog_feed_test.c

watchdog_notfeed_test.c

设备树文件：

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码：

imx_4.14.98_2.0.0_ga /drivers/misc/fsl_imx8_watchdog.c

内核对应选项：

CONFIG_FSL_IMX8_WATCHDOG=y

2.4、CAN 测试

IAC-IMX8-MB 主板引出了一路 SPI 外接 MCP2515 芯片扩展 CAN 接口，

测试原理：

文件系统中提供了测试 CAN 的方法，使用 CAN 工具进行测试。

can0 的 H、L 分别在 J20 的 1、2 脚

测试步骤和结果：

1. 需使用两路的 CAN 方可进行测试，可将当前的 CAN 连接到其他 CAN 接口，可以打开两块 IMX8 的板子，连接两个串口终端，进入系统配置和打开 CAN

```
# ip link set can0 type can bitrate 125000
```

```
# ifconfig can0 up
```

```
root@imx8mmqiyang:~# ip link set can0 type can bitrate 125000
root@imx8mmqiyang:~# ifconfig can0 up
root@imx8mmqiyang:~# █
```

2. 将一块开发板的 J20 的 1 脚、2 脚和 3 脚与另一块板子的 J20 的 1 脚、2 脚和 3 脚连接

3. 使用 CAN 测试程序进行测试：
这里将 can0 作为接收、can1 作为发送

```
# ./can_test
```

说明：①、can_test <device> 0 把 CAN 设置成接收数据。

②、can_test <device> 1 把 CAN 设置成发送数据。

```
root@imx8mmqiyang:/usr/test# ./can_test
Invalid arguments!
Usage: ./can_test <device> [01]
  device -- for example: can0
    0    -- test CAN recieve.
    1    -- test CAN send.

root@imx8mmqiyang:/usr/test# █
```

4. 这里把第一块板子的 CAN 当做接收端，在串口终端输入

```
# ./can_test can0 0
```

```
root@imx8mmqiyang:/usr/test#
root@imx8mmqiyang:/usr/test# ./can_test can0 0
CAN Start Testing ...
█
```

5. 把第二块板子的 CAN 当做发送端，在串口终端输入

有任何技术问题或需要帮助，请联系：supports@qiyangtech.com

第 10 页 共 41 页

购买产品，请联系销售：sales@qiyangtech.com

更多信息请访问：<http://www.qiytech.com>

©2020 Qiyangtech 版权所有

```
root@imx8mmqiyang:/usr/test# ./can_test can0 1
CAN Start Testing ...
send can datas: can_id = 0x123,data_len = 8
data[0] = 0x0
data[1] = 0x1
data[2] = 0x2
data[3] = 0x3
data[4] = 0x4
data[5] = 0x5
data[6] = 0x6
data[7] = 0x7
Test Success.
root@imx8mmqiyang:/usr/test#
```

此时第一块主板的串口调试终端收到，第二块主板发送的 CAN 数据

6. 然后对调测试、把第二块主板的 CAN 做为接收端、第一块主板的 CAN 做为发送端，测试方式一样。

测试源码:

/can_test/can_test.c

设备树文件:

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码:

imx_4.14.98_2.0.0_ga /drivers/net/can/spi/mcp251x.c

内核对应选项:

CONFIG_CAN_MCP251X=y

2.5、GPIO 测试

IAC-IMX8-MM 主板使用 I2C3 连接底板上的 CAT9555W 芯片作为扩展 GPIO。和 CPU 自带的 2 个 GPIO 做为测试

本次测试针对 J4 上面的 18 个 GPIO

GPIO6_0、GPIO6_1、GPIO6_2、GPIO6_3、GPIO6_4、GPIO6_5、GPIO6_6、GPIO6_7、GPIO5_21

GPIO6_8、GPIO6_9、GPIO6_10、GPIO6_11、GPIO6_12、GPIO6_13、GPIO6_14、GPIO6_15、GPIO3_09

测试原理：


gpio_test 0 测试 gpio 引脚无外部连接情况下，把所有引脚先设置成低电平，按回车后设置成高电平，通过外部测量 gpio 实际电平来确认 gpio 是否正常。

gpio_test 1 直接读取外接电平信号，用户可通过读取的电平值与连接的电平值做比较，确认 gpio 是否正常。

测试步骤和结果：

1. 运行 gpio 测试程序 gpio_test

```
# ./gpio_test
```



```
root@imx8mmqiyang:/usr/test# ./gpio_test
Invalid arguments!
Usage: ./gpio_test <device> <0|1>
<device> -- for example: /dev/qiyang_imx8_gpio
0         -- set gpio level.
1         -- get gpio level.
root@imx8mmqiyang:/usr/test# █
```

说明：
 gpio_test <device> 0 设置 gpio 的高电平和低电平
 gpio_test <device> 1 获取 gpio 的电平

2. J4 上无外接信号，测试输出低电平和高电平

```
# ./gpio_test /dev/qiyang_imx8_gpio 0
```

```

root@imx8mmqiyang:/usr/test# ./gpio_test /dev/qiyang_imx8_gpio 0
set gpio 'IMX_GPIO6_0' level '0'
set gpio 'IMX_GPIO6_8' level '0'
set gpio 'IMX_GPIO6_1' level '0'
set gpio 'IMX_GPIO6_9' level '0'
set gpio 'IMX_GPIO6_2' level '0'
set gpio 'IMX_GPIO6_10' level '0'
set gpio 'IMX_GPIO6_3' level '0'
set gpio 'IMX_GPIO6_11' level '0'
set gpio 'IMX_GPIO6_4' level '0'
set gpio 'IMX_GPIO6_12' level '0'
set gpio 'IMX_GPIO6_5' level '0'
set gpio 'IMX_GPIO6_13' level '0'
set gpio 'IMX_GPIO6_6' level '0'
set gpio 'IMX_GPIO6_14' level '0'
set gpio 'IMX_GPIO6_7' level '0'
set gpio 'IMX_GPIO6_15' level '0'
set gpio 'IMX_GPIO5_21' level '0'
set gpio 'IMX_GPIO3_9' level '0'
Gpios is output low level, now you can measure each pin!
Press the ENTER after measure each pins!
    
```

如上图所示，把每个 gpio 设置成低电平，用万用表测量对应的 gpio 的实际电平值来确定 gpio 是否正常，按回车之后把所有 gpio 设置成高电平，测量对应的 gpio 的实际电平值来确定 gpio 是否正常：

```

set gpio 'IMX_GPIO6_0' level '1'
set gpio 'IMX_GPIO6_8' level '1'
set gpio 'IMX_GPIO6_1' level '1'
set gpio 'IMX_GPIO6_9' level '1'
set gpio 'IMX_GPIO6_2' level '1'
set gpio 'IMX_GPIO6_10' level '1'
set gpio 'IMX_GPIO6_3' level '1'
set gpio 'IMX_GPIO6_11' level '1'
set gpio 'IMX_GPIO6_4' level '1'
set gpio 'IMX_GPIO6_12' level '1'
set gpio 'IMX_GPIO6_5' level '1'
set gpio 'IMX_GPIO6_13' level '1'
set gpio 'IMX_GPIO6_6' level '1'
set gpio 'IMX_GPIO6_14' level '1'
set gpio 'IMX_GPIO6_7' level '1'
set gpio 'IMX_GPIO6_15' level '1'
set gpio 'IMX_GPIO5_21' level '1'
set gpio 'IMX_GPIO3_9' level '1'
Gpios is output high level, now you can measure each pin!
Press the ENTER after measure each pins!
    
```

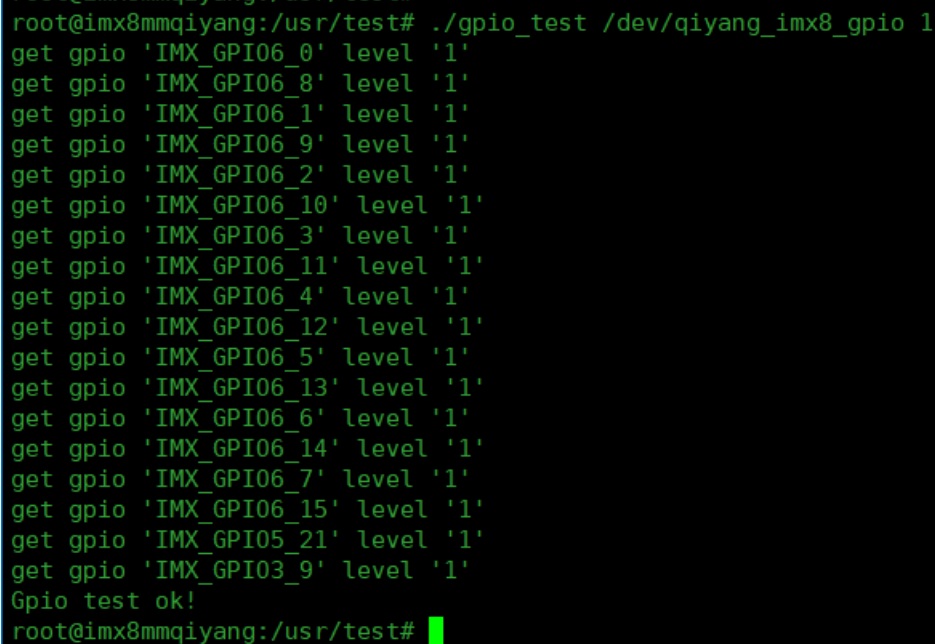
再按回车，提示测试 OK

```

Gpio test OK!
root@imx8mmqiyang:/usr/test# █
    
```

3. 将 gpio 设置为输入、外部接 3.3V 到管脚

```
# ./gpio_test /dev/qiyang_imx8_gpio 1
```



```
root@imx8mmqiyang:/usr/test# ./gpio_test /dev/qiyang_imx8_gpio 1
get gpio 'IMX_GPIO6_0' level '1'
get gpio 'IMX_GPIO6_8' level '1'
get gpio 'IMX_GPIO6_1' level '1'
get gpio 'IMX_GPIO6_9' level '1'
get gpio 'IMX_GPIO6_2' level '1'
get gpio 'IMX_GPIO6_10' level '1'
get gpio 'IMX_GPIO6_3' level '1'
get gpio 'IMX_GPIO6_11' level '1'
get gpio 'IMX_GPIO6_4' level '1'
get gpio 'IMX_GPIO6_12' level '1'
get gpio 'IMX_GPIO6_5' level '1'
get gpio 'IMX_GPIO6_13' level '1'
get gpio 'IMX_GPIO6_6' level '1'
get gpio 'IMX_GPIO6_14' level '1'
get gpio 'IMX_GPIO6_7' level '1'
get gpio 'IMX_GPIO6_15' level '1'
get gpio 'IMX_GPIO5_21' level '1'
get gpio 'IMX_GPIO3_9' level '1'
Gpio test ok!
root@imx8mmqiyang:/usr/test#
```

如上图所示，获取每一个 gpio 的电平状态，用户可以改变实际的接入 gpio 信号来确认 gpio 是否正常

设备节点:

/dev/qiyang_imx8_gpio

设备树文件:

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码:

imx_4.14.98_2.0.0_ga/drivers/misc/fsl_imx8_gpio.c

imx_4.14.98_2.0.0_ga/drivers/gpio/gpio-pca953x.c

内核对应选项:

CONFIG_FSL_IMX8_GPIO=y

CONFIG_GPIO_PCA953X=y

2.6、显示测试

IAC-IMX8-MB 主标配带有 qt5.10 的 qt 库，以及本司的 demo 程序，该程序将带你走进 QT 的世界。

2.6.1 hdmi 显示

测试步骤和结果:

1. 开发板默认启动为 HDMI 显示
运行 qt 测试程序
`# ./Imx6_qt_test`

设备树文件:

`imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts`

驱动代码:

`imx_4.14.98_2.0.0_ga/drivers/gpu/drm/bridge/lontium-lt9611.c`

内核对应选项:

`CONFIG_DRM_I2C_LT9611=y`

2.6.2 mipi-dsi 显示

2.6.2.1 hsd10p1 显示

测试步骤和结果:

1. 开发板启动时候在 u-boot 输入如下命令
u-boot=> **setenv fdt_file fsl-imx8mm-qiyang-hsd10p1.dtb**
u-boot=> **saveenv**
u-boot=> **boot**

```
Environment size: 2603/4092 bytes
u-boot=> setenv fdt_file fsl-imx8mm-qiyang-hsd10p1.dtb
u-boot=> saveenv
Saving Environment to MMC... Writing to MMC(1)... OK
u-boot=> boot
```

2. 运行 qt 测试程序
./Imx6_qt_test



设备树文件:

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang-hsd10p1.dts

驱动代码:

imx_4.14.98_2.0.0_ga/drivers/gpu/drm/panel/panel-jd9365-hsd10p1.c

内核对应选项:

CONFIG_DRM_PANEL_JD9365_HSD10P1=y

2.6.2.2 GN101IB27811433D 显示

测试步骤和结果:

1. 开发板启动时候在 uboot 输入如下命令
u-boot=> **setenv fdt_file fsl-imx8mm-qiyang-gnl.dtb**
u-boot=> **saveenv**
u-boot=> **boot**
2. 运行 qt 测试程序
./Imx6_qt_test

设备树文件:

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang-gnl.dts

驱动代码:

imx_4.14.98_2.0.0_ga/drivers/gpu/drm/panel/panel-gnl.c

内核对应选项:

CONFIG_DRM_PANEL_GNL=y

2.6.3 lvds 显示

1. 开发板启动时候在 uboot 输入如下命令
u-boot=> `setenv fdt_file fsl-imx8mm-qiyang-lt9211.dtb`
u-boot=> `saveenv`
u-boot=> `boot`
2. 运行 qt 测试程序
`./Imx8_qt_test`



2.7、触摸测试

测试原理：

打开 QT 测试程序，点击屏幕、屏幕显示正常，因为 HDMI 显示没有外接触摸屏，本测试针对 mipi-dsi 显示和 lt9211 显示

测试步骤和结果：

1. 运行 qt 测试程序
./Imx6_qt_test
触摸正常，无明显偏差

设备树文件：

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang-hsd10p1.dts

驱动代码：

imx_4.14.98_2.0.0_ga/drivers/input/touchscreen/goodix.c

内核对应选项：

CONFIG_TOUCHSCREEN_GOODIX=y

2.8、USB 测试

支持格式：fat32

在 IAC-IMX8-MB 主板上，共有 5 路 usb 口：

- a、1 路 J42 做为 device 口，用于 usb 方式下载固件
- b、1 路集成到 miniPCIE 接口上（J12）
- c、1 路 J6 做成接插件形式
- d、2 路 J5 作为 host 接口使用

本测试针对 J5 和 J6 测试（以下测试以 J5 做为例子）

测试原理：

开发板 usb 支持热插拔，将 U 盘插入后系统会自动识别并打印出 U 盘相关信息

识别后在/dev 目录下生成该设备节点/dev/sda 及分区节点/dev/sda1（若有多个分区，则数字部分依次增加）

最终系统会自动将所有分区挂载到/run/media/目录下，通过读写对应目录下的文件来判断该接口是否正常。

测试步骤和结果：

测试以只有一个分区的 U 盘为例

1. 将正常使用的 U 盘插入 J5 口，调试串口打印如下信息：

```

root@imx8mmqiyang:~# [ 1240.460050] usb 1-1.2: new high-speed USB device number 6 using ci_hdrc
[ 1240.589373] usb-storage 1-1.2:1.0: USB Mass Storage device detected
[ 1240.600210] scsi host0: usb-storage 1-1.2:1.0
[ 1241.640052] scsi 0:0:0:0: Direct-Access    Generic Flash Disk      8.07 PQ: 0 ANSI: 4
[ 1241.650292] sd 0:0:0:0: [sda] 15728640 512-byte logical blocks: (8.05 GB/7.50 GiB)
[ 1241.658911] sd 0:0:0:0: [sda] Write Protect is off
[ 1241.664653] sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[ 1241.679517]  sda: sda1
[ 1241.685524] sd 0:0:0:0: [sda] Attached SCSI removable disk
[ 1242.058393] FAT-fs (sda1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
    
```

如上图所示，显示 U 盘的一些基本信息，U 盘识别的设备节点为 sda，子节点为 sda1

2. 用 fdisk 命令来查看 sda 的信息

```
# fdisk -l /dev/sda
```

```

root@imx8mmqiyang:~# fdisk -l /dev/sda
Disk /dev/sda: 7.5 GiB, 8053063680 bytes, 15728640 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x7b4a8f76

Device      Boot Start      End  Sectors  Size Id Type
/dev/sda1   *          5120 15728639 15723520  7.5G  c W95 FAT32 (LBA)
root@imx8mmqiyang:~# █

```

- 查看 U 盘里的内容

```
# ls -l /run/media/sda1/
```

```

root@imx8mmqiyang:~# ls -l /run/media/sda1/
total 605588
drwxrwx--- 2 root disk      4096 Jan  1  1980 ??
drwxrwx--- 2 root disk      4096 Jan  1  1980 ???
-rwxrwx--- 1 root disk    238778 Feb 29 16:09 ??????????.pdf
drwxrwx--- 3 root disk      4096 Dec 23 18:01 Android
drwxrwx--- 2 root disk      4096 Jan  1  1980 CreateTestDir
drwxrwx--- 2 root disk      4096 Jan  1  1980 LOST.DIR
-rwxrwx--- 1 root disk    73015 Jun 23  2015 Serial.apk
drwxrwx--- 2 root disk      4096 Oct 16  2019 System Volume Information
drwxrwx--- 4 root disk      4096 Nov 28 11:02 UpdateAPK
-rwxrwx--- 1 root disk 12782495 Nov 28 11:02 UpdateAPK.zip

```

- 可以通过创建、拷贝、删除文件来测试 U 盘的读写
- 用相同的方法来测试 2 个 host usb 口，测试完毕后拔出 U 盘，打印信息如下

```

root@imx8mmqiyang:~# █
root@imx8mmqiyang:~# [ 1388.441506] usb 1-1.2: USB disconnect, device number 6
[ 1388.481497] FAT-fs (sda1): FAT read failed (blocknr 2120)

```

设备树文件:

```
imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts
```

驱动代码:

```
imx_4.14.78-1.0.0_ga/drivers/usb/chipidea/ci_hdrc_imx.c
```

内核对应选项:

```
CONFIG_USB_CHIPIDEA_OF=y
```

2.9、WIFI 测试

IAC-IMX8-MB 主板采用 AP6236wifi 芯片

测试原理:

使用 wpa_passphrase 与 wpa_supplicant 命令实现 wifi 的连接。

测试步骤和结果:

1、 加载驱动

```
# insmod /lib/modules/bcmdhd.ko
```

```
root@imx8mmqiyang:/usr/test# insmod /lib/modules/bcmdhd.ko
[ 2583.812776] dhd_module_init: in Dongle Host Driver, version 1.579.77.41.11 (r)
[ 2583.820091] ===== dhd_wlan_init_plat_data =====
[ 2583.825200] dhd_wlan_init_gpio: WL_HOST_WAKE=41, oob_irq=104, oob_irq_flags=0x414
[ 2583.832724] dhd_wlan_init_gpio: WL_REG_ON=42
[ 2583.837038] dhd_wifi_platform load: Enter
[ 2583.841097] Power-up adapter 'DHD generic adapter'
[ 2583.846065] wifi_platform_set_power = 1
[ 2583.849987] ===== PULL WL_REG_ON(42) HIGH! =====
[ 2584.360592] wifi_platform_bus_enumerate device present 1
[ 2584.365918] into dhd_wlan_set_carddetect
[ 2584.369857] ===== Card detection to detect SDIO card! =====
[ 2584.409114] mmc0: queuing unknown CIS tuple 0x80 (2 bytes)
[ 2584.416303] mmc0: queuing unknown CIS tuple 0x80 (3 bytes)
[ 2584.423482] mmc0: queuing unknown CIS tuple 0x80 (3 bytes)
[ 2584.431929] mmc0: queuing unknown CIS tuple 0x80 (7 bytes)
[ 2584.441058] mmc0: queuing unknown CIS tuple 0x81 (9 bytes)
[ 2584.542365] mmc0: new high speed SDIO card at address 0001
[ 2584.564488] bcmsdh_register: register client driver
```

2、 设置 wifi 的用户名: QYWIFI, 密码: QY@2019.com, 如果不一样, 请修改

```
# wpa_passphrase QYWIFI QY@2019.com >> /etc/wpa_supplicant.conf
# sync
```

3、 连接 wifi

```
# wpa_supplicant -Dnl80211 -i wlan0 -c /etc/wpa_supplicant.conf -B
```

4、 自动获取 IP

```
# udhcpc -i wlan0
```

```
root@imx8mmqiyang:~# udhcpc -i wlan0
udhcpc: started, v1.27.2
udhcpc: sending discover
udhcpc: sending select for 192.168.3.97
udhcpc: lease of 192.168.3.97 obtained, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 202.101.172.35
/etc/udhcpc.d/50default: Adding DNS 202.101.172.47
root@imx8mmqiyang:~# █
```

5、 静态 IP

有任何技术问题或需要帮助, 请联系: supports@qiyangtech.com

第 22 页 共 41 页

购买产品, 请联系销售: sales@qiyangtech.com

更多信息请访问: <http://www.qiytech.com>

©2020 Qiyangtech 版权所有

如果网段在 192.168.3.1，则设置 ip 的命令替换为

```
ifconfig wlan0 192.168.3.xxx
```

如果此时需要连接外网时，需要添加默认网关

```
route del default
```

```
route add default gw 192.168.3.1 dev wlan0
```

```
echo nameserver 114.114.114.114 > /etc/resolv.conf
```

6、Ping 百度

```
# ifconfig eth0 down
```

```
# ping -I wlan0 www.baidu.com
```

```
root@imx8mmqiyang:~# ping -I wlan0 www.baidu.com
PING www.a.shifen.com (180.101.49.12) from 192.168.3.97 wlan0: 56(84) bytes of data.
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=1 ttl=52 time=22.1 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=2 ttl=52 time=31.7 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=3 ttl=52 time=16.9 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=4 ttl=52 time=23.0 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=5 ttl=52 time=26.2 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=6 ttl=52 time=15.8 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=7 ttl=52 time=23.5 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=8 ttl=52 time=17.3 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=9 ttl=52 time=26.7 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=10 ttl=52 time=28.3 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=11 ttl=52 time=26.2 ms
^C
--- www.a.shifen.com ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 14052ms
rtt min/avg/max/mdev = 15.830/23.473/31.766/4.843 ms
root@imx8mmqiyang:~# █
```

设备树文件:

```
imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts
```

驱动代码:

```
imx_4.14.98_2.0.0_ga/drivers/net/wireless/bcmdhd/
```

```
imx_4.14.98_2.0.0_ga/drivers/mmc/host/sdhci-esdhc-imx.c
```

内核对应选项:

```
CONFIG_BCMDHD=m
```

```
CONFIG_MMC_SDHCI_ESDHC_IMX=y
```

2.10、蓝牙测试

IAC-IMX8-MB 主板采用 AP6236BT 芯片

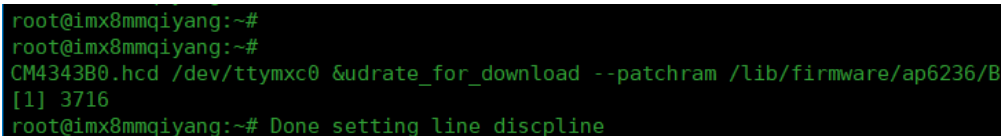
测试原理:

蓝牙扫描到外部蓝牙设备，发送数据给外部蓝牙设备，并且接收外部设备的数据

测试步骤和结果:

- 1、将固件通过串口写入蓝牙

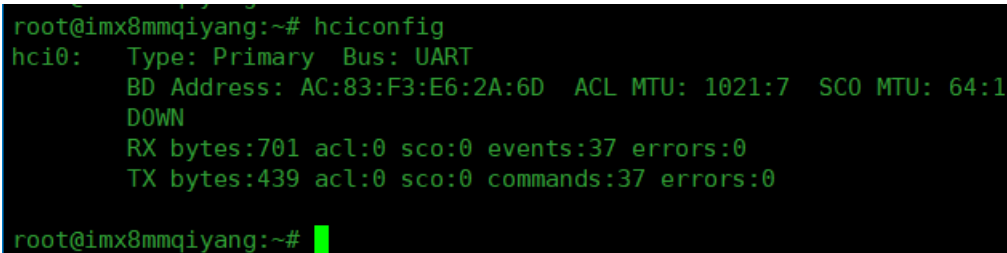
```
# brcm_patchram_plus --tosleep=50000 --no2bytes --enable_hci --baudrate 2000000
--use_baudrate_for_download --patchram /lib/firmware/ap6236/BCM4343B0.hcd
/dev/ttymx0 &
```



```
root@imx8mmqiyang:~#
root@imx8mmqiyang:~#
CM4343B0.hcd /dev/ttymx0 &udrate_for_download --patchram /lib/firmware/ap6236/B
[1] 3716
root@imx8mmqiyang:~# Done setting line discipline
```

- 2、输入 hciconfig 查看蓝牙设备

```
# hciconfig
```



```
root@imx8mmqiyang:~# hciconfig
hci0: Type: Primary Bus: UART
      BD Address: AC:83:F3:E6:2A:6D ACL MTU: 1021:7 SCO MTU: 64:1
      DOWN
      RX bytes:701 acl:0 sco:0 events:37 errors:0
      TX bytes:439 acl:0 sco:0 commands:37 errors:0

root@imx8mmqiyang:~#
```

- 3、配对蓝牙设备输入 bluetoothctl

```
# bluetoothctl (进入蓝牙配对模式)
```

```
[bluetooth]# power on
```

```
[bluetooth]# agent on
```

```
[bluetooth]# pairable on
```

```
[bluetooth]# devices (查看蓝牙是否有之前的设备配对)
```

```
[bluetooth]# remove 0C:D6:BD:4C:A7:55 (如果有将它移除)
```

```
[bluetooth]# devices (再次查看蓝牙是否有之前的设备配对)
```

```
[bluetooth]# scan on (输入该命令扫描蓝牙设备)
```

```
[bluetooth]# scan off (输入该命令结束蓝牙扫描)
```

```
[bluetooth]# trust 0C:D6:BD:4C:A7:55 (信任这个设备, 该设备为我测试手机)
```

```
[bluetooth]# pair 0C:D6:BD:4C:A7:55 (配对该设备)
```


[agent] Confirm passkey 817626 (yes/no): **yes** (选 yes 确定配对)
[bluetooth]# **quit** (退出)

```
root@imx8mmqiyang:~#
root@imx8mmqiyang:~# bluetoothctl
Agent registered
[bluetooth]# power on
[CHG] Controller AC:83:F3:E6:2A:6D Class: 0x00200000
Changing power on succeeded
[CHG] Controller AC:83:F3:E6:2A:6D Powered: yes
[bluetooth]# agent on
Agent is already registered
[bluetooth]# pairable on
Changing pairable on succeeded
[bluetooth]# devices
Device 0C:D6:BD:4C:A7:55 PLK-TL01
[bluetooth]# remove 0C:D6:BD:4C:A7:55
[DEL] Device 0C:D6:BD:4C:A7:55 PLK-TL01
Device has been removed
[bluetooth]# devices
[bluetooth]# scan on
Discovery started
[CHG] Controller AC:83:F3:E6:2A:6D Discovering: yes
[NEW] Device 4E:C8:32:0C:15:9C 4E-C8-32-0C-15-9C
[CHG] Device 4E:C8:32:0C:15:9C RSSI: -66
[NEW] Device 7F:D2:A3:AF:DE:BB 7F-D2-A3-AF-DE-BB
[NEW] Device 73:D3:8D:42:4F:24 73-D3-8D-42-4F-24
[NEW] Device 4A:00:C0:2C:14:A2 4A-00-C0-2C-14-A2
[NEW] Device 4C:49:E3:E0:BB:B2 小米手机
[NEW] Device 48:2C:A0:22:F2:05 小米手机
[NEW] Device 0C:D6:BD:4C:A7:55 PLK-TL01
[bluetooth]# scan off
Discovery stopped
[CHG] Controller AC:83:F3:E6:2A:6D Discovering: no
[CHG] Device 0C:D6:BD:4C:A7:55 RSSI is nil
[CHG] Device 48:2C:A0:22:F2:05 RSSI is nil
[CHG] Device 4C:49:E3:E0:BB:B2 RSSI is nil
[CHG] Device 4A:00:C0:2C:14:A2 TxPower is nil
[CHG] Device 4A:00:C0:2C:14:A2 RSSI is nil
[CHG] Device 73:D3:8D:42:4F:24 TxPower is nil
```

```

[CHG] Device 48:2C:A0:22:F2:05 RSSI is nil
[CHG] Device 4C:49:E3:E0:BB:B2 RSSI is nil
[CHG] Device 4A:00:C0:2C:14:A2 TxPower is nil
[CHG] Device 4A:00:C0:2C:14:A2 RSSI is nil
[CHG] Device 73:D3:8D:42:4F:24 TxPower is nil
[CHG] Device 73:D3:8D:42:4F:24 RSSI is nil
[CHG] Device 7F:D2:A3:AF:DE:BB RSSI is nil
[CHG] Device 4E:C8:32:0C:15:9C TxPower is nil
[CHG] Device 4E:C8:32:0C:15:9C RSSI is nil
[bluetooth]# trust 0C:D6:BD:4C:A7:55
[CHG] Device 0C:D6:BD:4C:A7:55 Trusted: yes
Changing 0C:D6:BD:4C:A7:55 trust succeeded
[bluetooth]# pair 0C:D6:BD:4C:A7:55
Attempting to pair with 0C:D6:BD:4C:A7:55
[CHG] Device 0C:D6:BD:4C:A7:55 Connected: yes
Request confirmation
[agent] Confirm passkey 817626 (yes/no): yes
[CHG] Device 0C:D6:BD:4C:A7:55 Modalias: bluetooth:v000Fp1200d1436
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001105-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 0000110a-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 0000110c-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001112-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001115-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001116-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 0000111f-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 0000112f-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001132-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001200-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001800-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 UUIDs: 00001801-0000-1000-8000-00805f9b34fb
[CHG] Device 0C:D6:BD:4C:A7:55 ServicesResolved: yes
[CHG] Device 0C:D6:BD:4C:A7:55 Paired: yes
Pairing successful
[CHG] Device 0C:D6:BD:4C:A7:55 ServicesResolved: no
[CHG] Device 0C:D6:BD:4C:A7:55 Connected: no
[bluetooth]# quit
root@imx8mmqiyang:~# █
    
```

4、发送和接收文件

```

# eval `dbus-launch --sh-syntax`
# /usr/libexec/bluetooth/obexd -r /usr/test -a & （其中/usr/test 是接收文件的目录）
    
```

```

root@imx8mmqiyang:~# eval `dbus-launch --sh-syntax`
root@imx8mmqiyang:~# /usr/libexec/bluetooth/obexd -r /usr/test -a &
[2] 3736
    
```

```

# echo 123456789 > /usr/test/test.txt （创建要发送的数据文件名）
# obexctl
[obex]# connect 0C:D6:BD:4C:A7:55 （连接手机）
[0C:D6:BD:4C:A7:55]# send /usr/test/test.txt 发送/usr/test/目录下的 test.txt 文件）
[0C:D6:BD:4C:A7:55]# quit （退出）
    
```

```

root@imx8mmqiyang:~# echo 123456789 > /usr/test/test.txt
root@imx8mmqiyang:~# obexctl
[NEW] Client /org/bluez/obex
[obex]# connect 0C:D6:BD:4C:A7:55
Attempting to connect to 0C:D6:BD:4C:A7:55
[NEW] Session /org/bluez/obex/client/session0 [default]
[NEW] ObjectPush /org/bluez/obex/client/session0
Connection successful
[0C:D6:BD:4C:A7:55]# send /usr/test/test.txt
Attempting to send /usr/test/test.txt to /org/bluez/obex/client/session0
[NEW] Transfer /org/bluez/obex/client/session0/transfer0
Transfer /org/bluez/obex/client/session0/transfer0
      Status: queued
      Name: test.txt
      Size: 10
      Filename: /usr/test/test.txt
      Session: /org/bluez/obex/client/session0
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Status: complete
[DEL] Transfer /org/bluez/obex/client/session0/transfer0
[0C:D6:BD:4C:A7:55]# quit
root@imx8mmqiyang:~# █
    
```

- 5、 # **hciconfig hci0 piscan** （让开发板能被其他蓝牙设备识别，自动接收文件，接收到的文件在/usr/test 目录下）

查看接收到的文件

```

root@imx8mmqiyang:~# hciconfig hci0 piscan
root@imx8mmqiyang:~# cat /usr/test/test-118.txt
1
root@imx8mmqiyang:~# █
    
```

设备树文件:

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码:

imx_4.14.98_2.0.0_ga/drivers/net/wireless/bcmdhd/

imx_4.14.98_2.0.0_ga/drivers/mmc/host/sdhci-esdhc-imx.c

内核对应选项:

CONFIG_BCMDHD=m

CONFIG_MMC_SDHCI_ESDHC_IMX=y

2.11、4G 测试

IAC-IMX8-MB 开发板引出 J12 miniPCI 接口，可以接 4G 模块及其可以接 4G 卡的 S1

支持全网通模块：EC20

支持移动 4G、联通 4G 模块、电信 4G

抽屉式卡槽，插上模块和模块支持的手机卡，上电，可以看到打印的信息

```
[ 15.542938] usb 2-1: new high-speed USB device number 2 using ci_hdrc
[ 15.716882] option 2-1:1.0: GSM modem (1-port) converter detected
[ 15.723463] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB0
[ 15.730765] option 2-1:1.1: GSM modem (1-port) converter detected
[ 15.737277] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB1
[ 15.744555] option 2-1:1.2: GSM modem (1-port) converter detected
[ 15.751217] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB2
[ 15.758468] option 2-1:1.3: GSM modem (1-port) converter detected
[ 15.765763] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB3
```

1、 关闭有线网络

```
# route del default
```

```
# ifconfig eth0 down
```

2、 在终端执行：（以下以移动卡为例）

```
联通：# pppd call wcdma &
```

```
移动：# pppd call tdsdma &
```

```
电信：# pppd call evdo &
```

可以打印出以下信息，说明联网成功

```
Script /usr/sbin/chat -s -v -f /etc/ppp/tdscdma-connect-chat finished (pid 3757), status = 0x0
Serial connection established.
using channel 1
Using interface ppp0
Connect: ppp0 <-> /dev/ttyUSB2
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x9d25a8db> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x0 <asyncmap 0x0> <auth pap> <magic 0xca0629f> <pcomp> <accomp>]
No auth is possible
sent [LCP ConfRej id=0x0 <auth pap>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x9d25a8db> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0xca0629f> <pcomp> <accomp>]
sent [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0xca0629f> <pcomp> <accomp>]
sent [CCP ConfReq id=0x1 <deflate 15> <deflate(old#) 15> <bsd v1 15>]
sent [IPCP ConfReq id=0x1 <compress VJ 0f 01> <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [LCP DiscReq id=0x2 magic=0xca0629f]
rcvd [LCP ProtRej id=0x3 80 fd 01 01 00 0f 1a 04 78 00 18 04 78 00 15 03 2f]
Protocol-Reject for 'Compression Control Protocol' (0x80fd) received
rcvd [IPCP ConfReq id=0x0]
sent [IPCP ConfNak id=0x0 <addr 0.0.0.0>]
rcvd [IPCP ConfRej id=0x1 <compress VJ 0f 01>]
sent [IPCP ConfReq id=0x2 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [IPCP ConfReq id=0x1]
sent [IPCP ConfAck id=0x1]
rcvd [IPCP ConfNak id=0x2 <addr 10.101.237.50> <ms-dns1 211.140.11.66> <ms-dns2 211.140.188.188>]
sent [IPCP ConfReq id=0x3 <addr 10.101.237.50> <ms-dns1 211.140.11.66> <ms-dns2 211.140.188.188>]
rcvd [IPCP ConfAck id=0x3 <addr 10.101.237.50> <ms-dns1 211.140.11.66> <ms-dns2 211.140.188.188>]
Could not determine remote IP address: defaulting to 10.64.64.64
Script /etc/ppp/ip-pre-up started (pid 3769)
Script /etc/ppp/ip-pre-up finished (pid 3769), status = 0x0
local IP address 10.101.237.50
remote IP address 10.64.64.64
primary DNS address 211.140.11.66
secondary DNS address 211.140.188.188
Script /etc/ppp/ip-up started (pid 3772)
Script /etc/ppp/ip-up finished (pid 3772), status = 0x0
```

3、设置 dns

```
# cp /run/ppp/resolv.conf /etc/resolv.conf
```

4、访问外网，按 ctrl+c 退出：

```
# ping -I ppp0 www.baidu.com
```

```
root@imx8mmevk:/# ping -I ppp0 www.baidu.com
PING www.a.shifen.com (36.152.44.96) from 10.101.237.50 ppp0: 56(84) bytes of data.
64 bytes from 36.152.44.96 (36.152.44.96): icmp_seq=1 ttl=54 time=75.1 ms
64 bytes from 36.152.44.96 (36.152.44.96): icmp_seq=2 ttl=54 time=49.2 ms
64 bytes from 36.152.44.96 (36.152.44.96): icmp_seq=3 ttl=54 time=44.8 ms
64 bytes from 36.152.44.96 (36.152.44.96): icmp_seq=4 ttl=54 time=47.4 ms
64 bytes from 36.152.44.96 (36.152.44.96): icmp_seq=5 ttl=54 time=33.8 ms
^C
--- www.a.shifen.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 13129ms
rtt min/avg/max/mdev = 33.859/50.099/75.122/13.605 ms
```

设备节点：

/dev/ttyUSB2

设备树文件：

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

2.12、串口测试

在 IAC-IMX8-MB 主板上，共有 5 路串口，其中 1 路被用来作为调试串口，3 路作为 232 串口、1 路做为 485 串口

| 串口 | 硬件位置 | 设备节点 |
|------------------|--------------------------|--------------|
| DEBUG (A53 调试串口) | J18 (收、发、GND 对应 1、2、3 脚) | /dev/ttymxc1 |
| DEBUG (M4 调试串口) | J19 (收、发、GND 对应 1、2、3 脚) | |
| COM1 (485 串口) | J23 (B、A、GND 对应 1、2、3 脚) | /dev/ttysWK0 |
| COM2 (232 串口) | J24 (收、发、GND 对应 1、2、3 脚) | /dev/ttysWK1 |
| COM3 (232 串口) | J22 (收、发、GND 对应 1、2、3 脚) | /dev/ttysWK2 |
| COM4 (232 串口) | J25 (收、发、GND 对应 1、2、3 脚) | /dev/ttysWK3 |

测试原理：

测试程序实现了一个串口每隔 1s 发送字符数据 `"/dev/ttyXXXX" test string!`，其中 x 为实际测试的设备节点，同时通过多线程方式，阻塞读取串口数据并打印

测试步骤和结果：

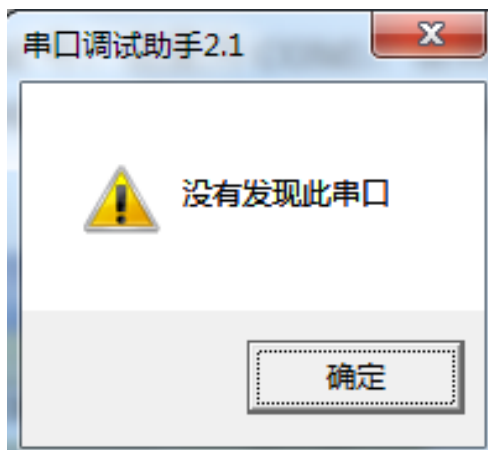
串口测试时 PC 机需要用到两个串口：

- ①、一个接调试串口，用于交互
- ②、一个接待测串口，用于测试串口收发数据

根据串口和硬件关系表，选择要测试的串口，通过提供的专用串口转接线，连接要测试的串口到 PC 机串口上。

PC 机打开光盘中的串口调试工具。

如果打开之后提示如下：



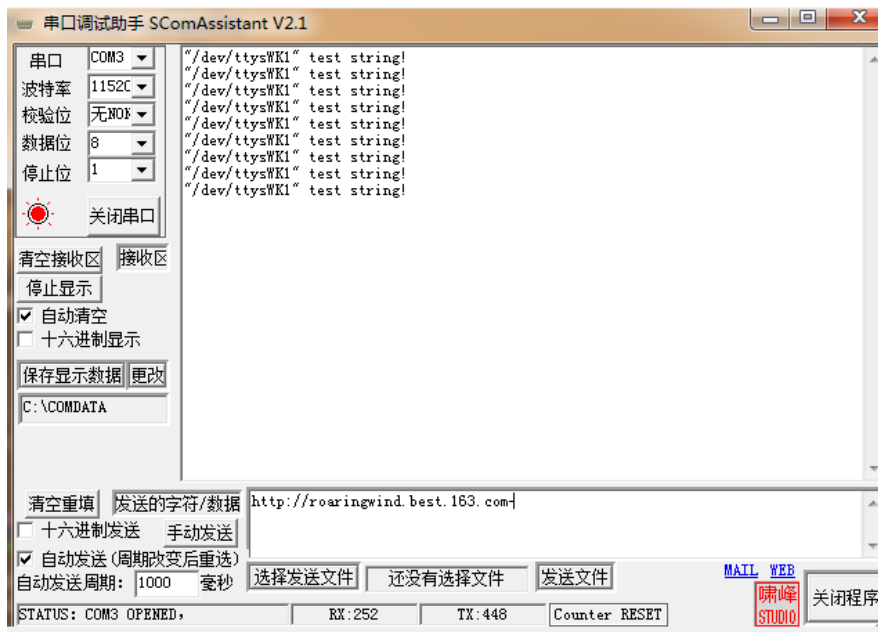
表示默认的 PC 机的 COM 口被终端占用了，关闭被占用的终端，然后再串口调试工具。

设置串口属性，串口对应 PC 机的 COM 号，这里为 COM3，波特率为 115200，数据位为 8 位，停止位为 1，奇偶校验为 NONE。



串口连接好并设置好之后就可以开始测试了
 分别测试 COM1、COM2、COM3、COM4 这 4 个串口
 这里以 COM2 为例作介绍，其他串口测试方法一样

```
# ./rs232_test /dev/ttyWK1 115200
```



调试串口收到数据

```

root@imx8mmqiyang:/usr/test# ./rs232_test /dev/ttySWK1 115200
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
receive 32 datas: http://roaringwind.best.163.com
    
```

因为 485 的流控脚由硬件来控制，使用这边 485 测试的方法和 232 是一样的

测试源码：

设备树文件：

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码：

imx_4.14.98_2.0.0_ga/drivers/spi/wk2xxx_spi.c

内核对应选项：

CONFIG_SPI_WK2124=y

2.13、摄像头测试

IAC-IMX8-MB 主板可以外接 ov5640 摄像头和 usb 摄像头

测试原理:

板子支持 OV5640 摄像头和 usb 摄像头，运行命令，测试摄像头是否正常。

主板上留出 camera 接口 J14，笔者测试的是 OV5640 摄像头，把 OV5640 与 J14 连接。

将 usb 摄像头与 J5 连接

测试步骤和结果:

1、 输入以下命令

```
# gst-launch-1.0 v4l2src device=/dev/video0 ! glimagesink
```

其中/dev/video0 表示摄像头的设备，修改/dev/video0 为/dev/video1 进行另外一个摄像头测试

2、 可以在屏上显示 camera 的图像

测试工具:

gst-launch-1.0

设备树文件:

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码:

imx_4.14.98_2.0.0_ga/drivers/media/platform/mxc/capture/ov5640_mipi_v2.c

imx_4.14.98_2.0.0_ga/drivers/media/usb/uvic/

内核对应选项:

CONFIG_MXC_CAMERA_OV5640_MIPI_V2=y

CONFIG_USB_VIDEO_CLASS=y

2.14、音频测试

IAC-IMX8-MB 主板外接 ES8388 音频芯片

测试原理：

可通过 `aplay` 命令播放音频文件，通过 `arecord` 录音，录音孔为 J8、播放音频为 J9（耳机）和 J10（喇叭）

测试步骤和结果：

1、录音测试

使用麦克风接好录音接口 J8，然后在终端中输入 `arecord -c 2 -r 44100 test.wav` 进行录音，录音文件名为 `test.wav`

```
# arecord -c 2 -r 44100 test.wav
```

2、播放音频测试

通过耳机（J9）和喇叭（J10）

通过 `aplay test.wav` 命令播放 `test.wav`

```
# aplay test.wav
```

测试工具：

`arecord`、`aplay`

设备树文件：

```
imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts
```

驱动代码：

```
imx_4.14.98_2.0.0_ga/sound/soc/codecs/es8328-i2c.c
```

```
imx_4.14.98_2.0.0_ga/sound/soc/codecs/es8328.c
```

```
imx_4.14.98_2.0.0_ga/sound/soc/fsl/imx-es8328.c
```

内核对应选项：

```
CONFIG_SND_SOC_ES8328=y
```

```
CONFIG_SND_SOC_ES8328_I2C=y
```

```
CONFIG_SND_SOC_IMX_ES8328=y
```

2.15、SD 卡测试

支持格式：fat32

IAC-IMX8-MB 主板提供了 1 路 SD 卡接口（J29）可供用户使用

测试原理：

板载 SD 卡接口支持热插拔，将 SD 卡插入后，系统会识别该 SD 卡，并打印出 SD 卡相关信息。

在/dev 目录下生成该设备节点及分区节点，之后系统会自动将所有分区挂载到 /run/media/目录下，通过读写对应目录下文件，可判断该接口是否正常。

测试步骤和结果：

以下测试步骤以只有一个分区的 SD 卡为例，若有多个分区，则测试方法类似。插入 SD 卡产生的设备节点为/dev/mmcblk1，分区 n 对应的分区设备节点为/dev/mmcblk1p1 在这里插入一张金斯顿 8G 的 SD 卡，打印信息如下：

```
root@imx8mmqiyang:/# [ 1479.217181] mmc1: host does not support reading read-only switch, assuming write-enable
[ 1479.233857] mmc1: new high speed SDHC card at address aaaa
[ 1479.239907] mmcblk1: mmc1:aaaa SC16G 14.8 GiB
[ 1479.249011] mmcblk1: p1
[ 1479.500045] FAT-fs (mmcblk1p1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
```

如上图所示，显示 SD 的卡一些基本信息，这里的设备节点为 mmcblk1，分区为 p1 也可以用 fdisk 命令来查看 SD 的信息

```
# fdisk -l /dev/mmcblk1
```

```
root@imx8mmqiyang:/# fdisk -l /dev/mmcblk1
Disk /dev/mmcblk1: 14.9 GiB, 15931539456 bytes, 31116288 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x00000000

Device            Boot Start      End  Sectors  Size Id Type
/dev/mmcblk1p1    0 31116287 31116288 14.9G  b W95 FAT32
root@imx8mmqiyang:/#
```

这里已经自动将 SD 卡挂载到/run/media/mmcblk1p1/目录下，可以直接查看 SD 卡里的内容

```
root@imx8mmqiyang:/# ls /run/media/mmcblk1p1/
ll.txt  Android  DCIM  LOST.DIR
root@imx8mmqiyang:/#
```

可以通过创建、拷贝、删除文件来测试 SD 卡的读写
拔出 SD 卡，打印信息如下

```
root@imx8mmqiyang:/#  
root@imx8mmqiyang:/# [ 1579.299337] mmc1: card aaaa removed  
[ 1579.380679] FAT-fs (mmcblk1p1): FAT read failed (blocknr 32)  
█
```

2.16、网口测试

开发板带 2 个网口，eth0 接口为 J11、eth1 接口为 J26（以下测试以 eth0 为例）

有任何技术问题或需要帮助，请联系：supports@qiyangtech.com

第 36 页 共 41 页

购买产品，请联系销售：sales@qiyangtech.com

更多信息请访问：<http://www.qiytech.com>

©2020 Qiyangtech 版权所有

测试原理:

设置板子网络，用 ping 工具检查网络是否连通

测试步骤和结果:

1. 用一根网线连接板子 J11 网口和路由器（交换机亦可），用另一根网线连接电脑和路由器（交换机亦可），确保网络能上网

```
root@imx8mmqiyang:~# [ 3292.964424] fec 30be0000.ethernet eth0: Link is Up - 1Gbps/Full - flow control rx/tx
[ 3292.972220] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
```

2. 设置板子网络
自动配置，输入
`# udhcpc -i eth0`

```
root@imx8mmqiyang:~#
root@imx8mmqiyang:~# udhcpc -i eth0
udhcpc: started, v1.27.2
udhcpc: sending discover
udhcpc: sending select for 192.168.1.226
udhcpc: lease of 192.168.1.226 obtained, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 202.101.172.35
/etc/udhcpc.d/50default: Adding DNS 202.101.172.47
root@imx8mmqiyang:~#
```

手动配置，输入

```
ifconfig eth0 192.168.1.71 （板子上电已经默认设置为这个）
echo nameserver 114.114.114.114 > /etc/resolv.conf
route add default gw 192.168.1.1 dev eth0
```

3. 测试内网，输入
`# ping -I eth0 192.168.1.1`

```
root@imx8mmqiyang:~# ping -I eth0 192.168.1.1
PING 192.168.1.1 (192.168.1.1) from 192.168.1.226 eth0: 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=254 time=1.51 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=254 time=2.09 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=254 time=2.10 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=254 time=2.10 ms
64 bytes from 192.168.1.1: icmp_seq=5 ttl=254 time=2.10 ms
```

4. 测试外网，输入
`# ping -I eth0 www.baidu.com`

```
root@imx8mmqiyang:/# ping -I eth0 www.baidu.com
PING www.a.shifen.com (180.101.49.11) from 192.168.1.226 eth0: 56(84) bytes of data.
64 bytes from 180.101.49.11 (180.101.49.11): icmp_seq=1 ttl=52 time=10.6 ms
64 bytes from 180.101.49.11 (180.101.49.11): icmp_seq=2 ttl=52 time=10.9 ms
64 bytes from 180.101.49.11 (180.101.49.11): icmp_seq=3 ttl=52 time=10.9 ms
64 bytes from 180.101.49.11 (180.101.49.11): icmp_seq=4 ttl=52 time=10.9 ms
64 bytes from 180.101.49.11 (180.101.49.11): icmp_seq=5 ttl=52 time=10.6 ms
64 bytes from 180.101.49.11 (180.101.49.11): icmp_seq=6 ttl=52 time=10.7 ms
64 bytes from 180.101.49.11 (180.101.49.11): icmp_seq=7 ttl=52 time=10.6 ms
```

2.17、屏背光测试

测试原理：

采用占空比(PWM)控制屏背光的亮度，一个有 8（0/1/2/3/4/5/6/7）个等级的亮度调节，本测试针对 mipi-dsi 显示

测试步骤和结果：

1. 将背光的亮度设置为最大
echo 0 >/sys/class/backlight/mipi_backlight/brightness
2. 将背光的亮度设置为最低
echo 7 >/sys/class/backlight/mipi_backlight/brightness

设备树文件：

imx_4.14.98_2.0.0_ga/arch/arm64/boot/dts/freescale/fsl-imx8mm-qiyang.dts

驱动代码：

imx_4.14.98_2.0.0_ga/ drivers/video/backlight/pwm_bl.c

内核对应选项：

CONFIG_BACKLIGHT_PWM=y

三、测试小结

开发板的基本功能到此测试完毕，对于测试过程中出现的问题，可根据提供的测试源码进行排查。编写中的笔者本着求真务实的精神对文字和程序进行斟酌和校验，但仍难免存在疏误，敬请读者批评指正和谅解。

浙江启扬智能科技有限公司

电话：0571-87858811 / 87858822

传真：0571-89935912

技术支持：0571-87858811 转 805

E-MAIL: supports@qiyangtech.com

网址： <http://www.qiytech.com>

地址：浙江省杭州市西湖科技园西园八路6号A幢3楼

邮编：310030