用户手册

[SBC-EC8800]

历史版本

Rev.	Note	Author
20160307	Initial	Baijy
20160315	修改 dtb 文件	Baijy
20160321	修改错误的命令	Sandy
20160323	1. 增加 wifi 和蓝牙测试 2. 增加 PWRON RESETn 按键测试	Rongdong
20160331	增加从 SPI Flash 启动	Baijy
20160505	1. 4.3 inch 10 分钟后白屏问题 2. 7inch LCD 没关背光问题 3. [ADC] voltage5 查看采样值失败 4. [Uboot]板连电脑，uboot 下 ping 不通电脑 5. [Uboot]板连路由器，uboot 下 ping 不通路由器 6. [QSPI flash] 启动 reset 和 reboot 7. [蓝牙音频]配对失败	Rongdong
20160707	Rev01 版本	Sandy

目录

Release Note

1. 镜像版本

SBC-EC8800-Release-SDcard-EMMC-REV01.img

2. 功能列表

Feature List	SBC-EC8800			
	Schematic Page#	On-Chip Peripherals	On-Board Peripherals	Detail Functions(existing)
u-boot version	2015.09			Supports kernel boot
kernel version	4.1.6			Supports all below functionality
Filesystem				Default root file system used by debian
CPU	EC8800-U17	AM437X_ZDN		Null
DDRAM	EC8800-p7-u20 /u15	DDR	MT41K256M16HA-125	Can access read write and run code
PMIC	EC8800-p3-u16	I2C0	TPS65218	Null
MicroSD_(TF)	EC8800-p8-J2	MMC0	uSD-SCHA5B	Can access read write and boot
Integrited-RTC	EC8800-p5	RTC	Null	can read write and keep time off power
LEDs	EC8800-p12-D8 /D9	gpio	Null	System can control LED to light or not
ADC	EC8800-P12-J5	ADC	Null	Can read the ad value from pin
LCD	EC8800-P10-J1	RGB	Null	Can show picture on the screen
Backlight	EC8800-P10-J1	PWM	Null	System can control the LCD backlight
TouchScreen	EC8800-P10-J1	ADC-TSC	Null	System use touchscreen
eMMC	EC8800-p8-u22	MMC1	MTFC4GACAAAM-4M IT	Can access read write
EEPROM	EC8800-p8-u12	I2C0	CAT24C256W	Can access read write
SPI-FLASH	EC8800-p8-u3	QSPI	N25Q256A13EF840	1. Boot from SPI-Flash 2. SPI-Flash access in kernel
SPI	EC8800-P12-J11	SPI1	Null	System can send and receive data in loopback mode

CAN-1	EC8800-P12-J5	CAN1	Null	System can send and receive data between two board
CAN-2	EC8800-P12-J5	CAN0	Null	System can send and receive data between two board
UART-0	EC8800-P12-J13	UART0	Null	System can send and receive data in loopback mode
UART-1	EC8800-P12-J11	UART1	Null	System can send and receive data in loopback mode
UART-5	EC8800-P12-J5	UART5	Null	System can send and receive data in loopback mode
RS485	EC8800-P12-J5	UART3	Null	System can send and receive data between two board
USB-Host	EC8800-P5-J9	USB1	USB2514	Can recognize U disk by USB host
USB-OTG	EC8800-P5-J10	USB0	Null	Can recognize U disk in host mode, and can work as usb ethernet in device mode
WIFI	PH1800-P13-J24/J25	UART1&MMC2&MCAPS0&I2C1	EXP-WFB00(Jorjin WG7801-D0)	1. Can ping the server using 2.4Ghz

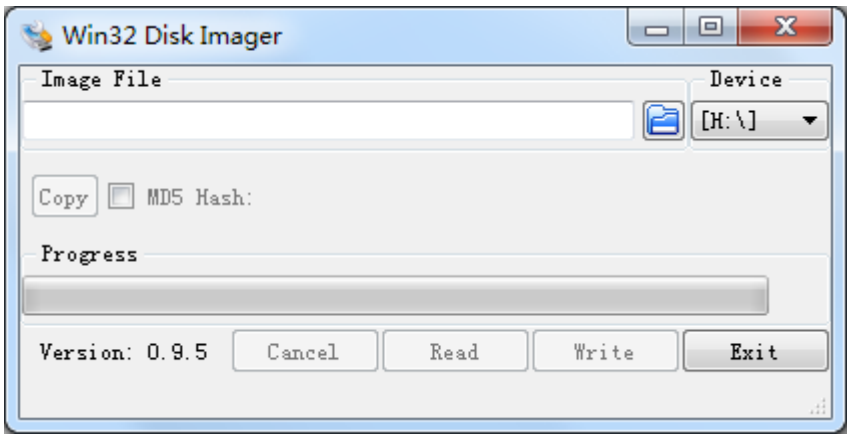
3. 已知问题

Known issue List	Detail
WIFI&Bluetooth	1. gstreamer
CAMERA	Could Preivew, take picture and record video
SDcard	1. Use 16G high speed SD card to burn the image, power on start up. 2. Short connect pin 39 and 40 in J5, execute the serial transceiving instructions, check the serial print info, “open dev/ttyOMAP0 error frequently occurs.
eth	Board to board ping, offline and connect again.

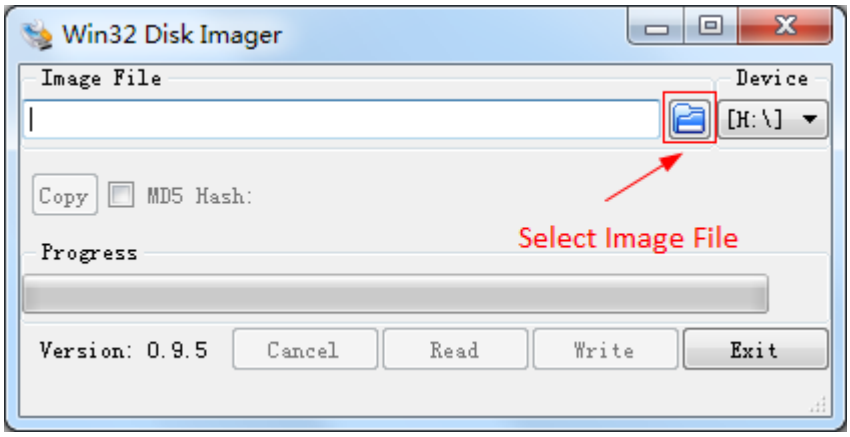
第1章 快速启动

1.1 烧写镜像到 SD 卡

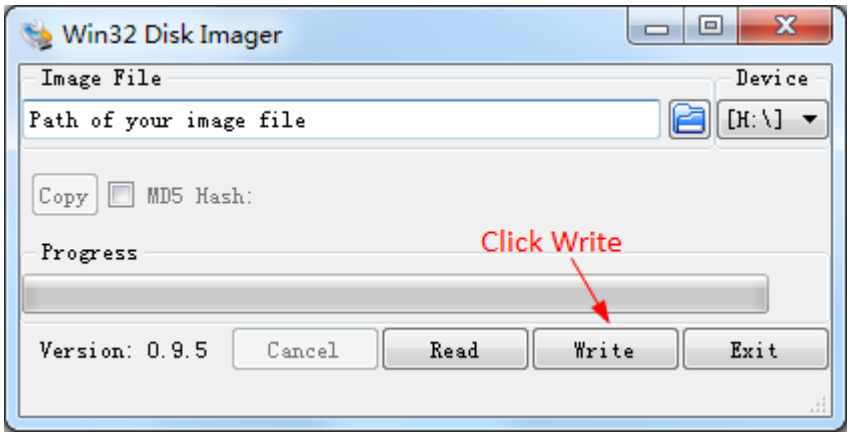
- 首先，你需要准备一张不小于 2G 的 SD 卡
- 然后，你需要从 https://sourceforge.net/projects/win32diskimager/ 下载并安装 Win32 Disk Imager



- 选择需要烧写的镜像，SBC-EC8800-Release-REV01\image\SBC-EC8800-Release-SDcard-EMMC-REV01.img:



- 点击 Write 烧写镜像：



1.2 从 SD 卡启动系统

- 在 PC 上安装串口软件（例如 SecureCRT），选择正确的端口号，波特率 115200，8 位数据位，1 位停止位，无奇偶校验
- 用 USB 转 TTL 模块连接板上的 DEBUG 接口(J13)和 PC
- 把 SD 卡插入板子的插槽(J2)
- 用 5V，2A 的电源，给板子供电
- 系统启动完毕之后，串口显示如下:

```
[  OK  ] Started Login Service.
         Starting Getty on tty1...
[  OK  ] Started Getty on tty1.
         Starting Serial Getty on ttyS0...
[  OK  ] Started Serial Getty on ttyS0.
[  OK  ] Reached target Login Prompts.
[   13.965466] wlcore: firmware booted (Rev 8.9.0.1.55)
[   14.155041] FAT-fs (mmcblk0p1): Volume was not properly unmounted. Some data
may be corrupt. Please run fsck.
[  OK  ] Started Embest AutoExec Service.

Debian GNU/Linux 8 embest ttyS0

embest login:
```

输入用户名和密码 root 登录；

```
Debian GNU/Linux 8 embest ttyS0

embest login: root
Password:
Linux embest 4.1.6 #1 PREEMPT Mon Jun 20 17:42:57 CST 2016 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@embest:~#
```

1.3 从 SPI Flash 启动

参考 1.2，先从 SD 卡启动，终端中打印如下信息时，按“回车键”进入 uboot:

```
U-Boot SPL 2015.07 (Jun 20 2016 - 17:15:48)
SPL: Please implement spl_start_uboot() for your board
SPL: Direct Linux boot not active!
reading u-boot.img
reading u-boot.img


U-Boot 2015.07 (Jun 20 2016 - 17:15:48 +0800)


I2C:   ready
DRAM:  1 GiB
```

PMIC: TPS65218
MMC: OMAP SD/MMC: 0, OMAP SD/MMC: 1
reading uboot.env

** Unable to read "uboot.env" from mmc0:1 **
Using default environment

Net: <ethaddr> not set. Validating first E-fuse MAC
cpsw, usb_ether
Hit any key to stop autoboot: 0
U-Boot#
（按下 Enter 键）
在终端中执行以下命令:
U-Boot# run update_qspi_flash
switch to partitions #0, OK
mmc0 is current device
SD/MMC found on device
reading u-boot-spl.bin
56904 bytes read in 9 ms (6 MiB/s)
SF: Detected N25Q256 with page size 256 Bytes, erase size 4 KiB, total 32 MiB, mapped at 30000000
SF: 589824 bytes @ 0x0 Erased: OK
device 0 offset 0x0, size 0xde48
SF: 56904 bytes @ 0x0 Written: OK
输入下列命令从 SD 卡启动系统：
U-Boot# boot

将 SBC-EC8800-Release-SDcard-EMMC-REV01.img 拷贝到 U 盘，将 U 盘插入 USB 接口（J9）：
root@embest:~# ls /dev/sd*
/dev/sda /dev/sda1
root@embest:~# mount /dev/sda1 /mnt/
root@embest:~# dd if=/mnt/SBC-EC8800-Release-SDcard-EMMC-REV01.img of=/dev/mmcblk1

注意：烧写时间较长，请耐心等待...

烧写结束后，上电复位并启动系统

第2章 功能测试

首先，请参考第一章 1.1，把系统启动起来. 然后跟随下面的指引测试各项功能.

2.1 LED 测试

用户能够控制 SBC-EC8800 上的 LED（D8,D9）指示灯。在终端中执行以下命令来进行测试；（其中 D8 对应 user_leds_d8，D4 对应 user_leds_d9）

熄灭 LED:

root@embest:~# echo 0 > /sys/class/leds/user_leds_d8/brightness

root@embest:~# echo 0 > /sys/class/leds/user_leds_d9/brightness

点亮 LED:

root@embest:~# echo 1 > /sys/class/leds/user_leds_d8/brightness

root@embest:~# echo 1 > /sys/class/leds/user_leds_d9/brightness

2.2 RTC 测试

在串口终端输入：

查看当前时间：

root@embest:~# date

Sat Jan 1 00:02:07 UTC 2000

设置时间 2016 年 3 月 9 日 10 时 46 分：

root@embest: # date 030910462016

Wed Mar 9 10:46:00 UTC 2016

把系统时钟写入 RTC:

root@embest: # hwclock –w

读取 RTC:

root@embest: # hwclock

Wed 09 Mar 2016 10:46:23 AM UTC -0.432561 seconds

可以看到，硬件时钟 RTC 被设置成 2016 年 3 月 9 日，系统时钟被保存到硬件时钟里。

重启系统并查看时间：

root@embest:~# date

Wed Mar 9 10:46:45 UTC 2016

2.3 EEPROM 测试

在串口终端输入一下命令：

root@embest:~# ./eeprom_test

data will write to EEPROM at 0x400

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f
30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f
50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f
60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f
70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f
80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f
90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e 9f
a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae af
b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd be bf
c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd ce cf
d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de df
e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee ef
f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ff

data read from EEPROM at 0x400

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f
30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f
50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f
60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f
70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f
80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f
90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e 9f
a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae af
b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd be bf
c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd ce cf
d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de df

e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee ef

f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ff

写数据与读到的数据相同，测试通过；

2.4 EMMC 测试

在串口终端执行：

root@embest:~# touch emmc_read emmc_write

编辑 emmc_write:

root@embest:~# vi emmc_write

例如写入“emmc write test”

写 emmc 命令:

root@embest:~# dd if=emmc_write of=/dev/mmcblk1

[929.393325] mmcblk1: p1 p2

0+1 records in

0+1 records out

17 bytes (17 B) copied, 0.135215 s, 0.1 kB/s

读 emmc 命令：

root@embest:~# dd if=/dev/mmcblk1 of=emmc_read bs=1K count=10

0+1 records in

0+1 records out

17 bytes (17 B) copied, 0.00152725 s, 11.1 kB/s

查看 emmc_read:

root@embest:~# cat emmc_read

emmc write test

测试成功；

2.5 ADC 测试

在串口终端输入一下命令，采样值返回：

root@embest:~# cat /sys/bus/platform/devices/TI-am335x-adc/iio\:device0/in_voltage4_raw

1054

root@embest:~# cat /sys/bus/platform/devices/TI-am335x-adc/iio\:device0/in_voltage5_raw

530

root@embest:~# cat /sys/bus/platform/devices/TI-am335x-adc/iio\:device0/in_voltage6_raw

586

root@embest:~# cat /sys/bus/platform/devices/TI-am335x-adc/iio\:device0/in_voltage7_raw

594

2.6 LCD 测试

连接显示屏到 J1

4.3 寸屏：

打开 SD 卡中 uEnv.txt 文件，修改 fdtfile= embest-SBC-EC8800-4.3inch_LCD.dtb，重新启动系统

7 寸屏：

打开 SD 卡中 uEnv.txt 文件，修改 fdtfile= embest-SBC-EC8800-7inch_LCD.dtb，重新启动系统

2.7 背光测试

背光的亮度设置范围为（1—8），1 表示亮度最低，8 表示亮度最高，在串口终端下输入如下命令进行背光测试：

最暗：

```
root@embest:~# echo   1   > /sys/class/backlight/backlight/brightness
```

最亮：

```
root@embest:~# echo   8   > /sys/class/backlight/backlight/brightness
```

2.8 触摸屏测试

连接显示屏到 J1,在串口终端输入以下命令执行触摸屏校准程序：

```
root@embest:~# ts_calibrate
```

按照屏幕上提示，点击 “+” 图标 5 次完成校准。

2.9 串口测试

开发板上有 3 个串口，其中 UART-0(J13)为 debug 接口

2.9.1 UART1

短接 J11 第 8，10 号接口：

```
root@embest:~# ./uart_test -d /dev/ttyS1 -b 115200
/dev/ttyS1 SEND: 1234567890
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 1
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 2
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 3
/dev/ttyS1 RECV 1 total
```

/dev/ttyS1 RECV: 4
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 5
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 6
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 7
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 8
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 9
/dev/ttyS1 RECV 1 total
/dev/ttyS1 RECV: 0

注意：Ctrl+C 中断串口测试

2.9.2 UART5

短接 J5 第 20，22 号接口：

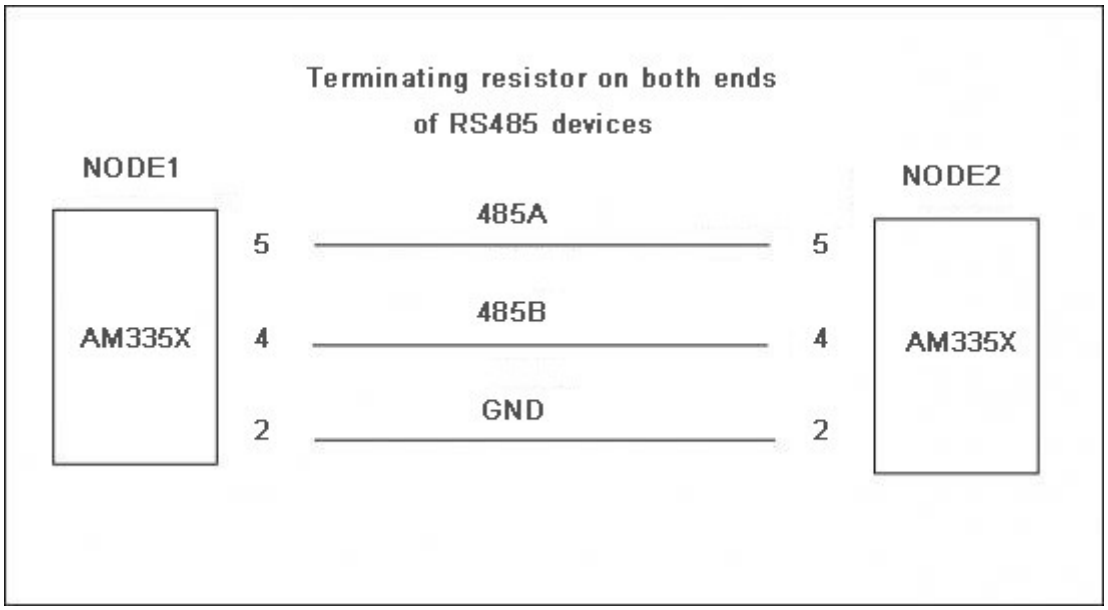
root@embest:~# ./uart_test -d /dev/ttyS5 -b 115200

/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 1
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 2
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 3
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 4
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 5
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 6
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 7
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 8
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 9
/dev/ttyS5 RECV 1 total
/dev/ttyS5 RECV: 0

注意：Ctrl+C 中断串口测试

2.10 RS485 测试

SBC-EC8800 可以作为一个 RS485 使用，按照下图所示连接原理，并参考原理图找到对应的引脚，用连接线连接 SBC-EC8800 的 RS485 接口和另一个 RS485 设备接口。



一个设备输入：

root@embest:~# ./uart_test2 /dev/ttyS3 9600 1

send_buf size =50

/dev/ttyO3 input bandrate value = 9600 flag = 1 set bandrate is 9600

[databits = 8, stopbits= 1, parity= 78]

[SET CRTSCTS]

uart setup done!!!

uart fd = 3

temp = 1

start receive.....

处于等待接收状态

另一个设备输入：

root@embest:~# ./uart_test2 /dev/ttyS3 9600 0 10

send_buf size =50

/dev/ttyO3 input bandrate value = 9600 flag = 0 set bandrate is 9600

[databits = 8, stopbits= 1, parity= 78]

[SET CRTSCTS]

uart setup done!!!

uart fd = 3

temp = 0

send buf times 10

1 send data successful

2 send data successful

3 send data successful

4 send data successful

5 send data successful

6 send data successful

7 send data successful

8 send data successful

9 send data successful

10 send data successful

发 10 个数据

观察接收设备端有无数据接收。

2.11 CAN 测试

SBC-EC8800 可以作为一个 CAN 设备使用。由于 SBC-EC8800 有两个 can，可以用自身的 can0 can1 进行测试,can0 与 can1 用连接线连接。J5 的 33，34 号引脚连接，35，36 号引脚连接，37，38 号引脚连接

测试方法如下：

1. 打开 can0 can1

root@embest:~# ip link set can0 type can bitrate 50000 triple-sampling on

root@embest:~# ip link set can1 type can bitrate 50000 triple-sampling on

root@embest:~# ip link set can0 up

root@embest:~# ip link set can1 up

2. can1 接收，can0 往 can1 发数据

root@embest:~# candump can1&

root@embest:~# cansend can0 123#11223344556677

3. 测试完毕关闭设备

root@embest:~# ip link set can0 down

root@embest:~# ip link set can1 down

用户可以根据以上命令进行相互收发测试，还可以设置不同的波特率进行通信，在设置不同波特率之前必须先关闭设备，可设置的波特率有：

- 25KBPS（250000）
- 50KBPS（50000）
- 125KBPS（125000）
- 500KBPS（500000）
- 650KBPS（650000）

- 1MKBPS（1000000）

以上的波特率均能正常通信，还有其它波特率可以设置，用户可以自己尝试，看能否通信。另外也可以外接其他板的 can 接口测试。

2.12 网络测试

连接网线到 J8，在串口终端中输入以下命令来设置 IP 地址：

root@embest:~# ifconfig eth0 192.168.2.64

网络测试：

root@embest:~# ping 192.168.2.1

2.13 USB 测试

2.13.1 Host 测试

将 U 盘插入 USB host 接口（J9）,串口显示磁盘信息:

```
[    69.262552] usb 1-1: new high-speed USB device number 2 using xhci-hcd
[    69.409547] usb 1-1: New USB device found, idVendor=058f, idProduct=6387
[    69.416660] usb 1-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[    69.425401] usb 1-1: Product: Mass Storage
[    69.429814] usb 1-1: Manufacturer: Generic
[    69.435235] usb 1-1: SerialNumber: 7CF60344B2B3
[    69.444585] usb-storage 1-1:1.0: USB Mass Storage device detected
[    69.454790] scsi host0: usb-storage 1-1:1.0
[    70.454501] scsi 0:0:0:0: Direct-Access     Generic  Flash Disk       8.07 PQ: 0 ANSI: 4
[    70.476791] sd 0:0:0:0: [sda] 7598080 512-byte logical blocks: (3.89 GB/3.62 GiB)
[    70.489773] sd 0:0:0:0: [sda] Write Protect is off
[    70.497971] sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[    70.516678]  sda: sda1
[    70.529248] sd 0:0:0:0: [sda] Attached SCSI removable disk
```

串口终端输入如下命令：

root@embest:~# ls /dev/sd*

/dev/sda /dev/sda1

/dev 下存在设备节点；

2.13.2 OTG 测试

1 主设备

通过转接线连接 U 盘到 J10:

[386.862557] usb 3-1: new high-speed USB device number 3 using xhci-hcd

[387.009497] usb 3-1: New USB device found, idVendor=058f, idProduct=6387

[387.016691] usb 3-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3

[387.025426] usb 3-1: Product: Mass Storage

[387.029848] usb 3-1: Manufacturer: Generic

[387.035314] usb 3-1: SerialNumber: 7CF60344B2B3

[387.044159] usb-storage 3-1:1.0: USB Mass Storage device detected

[387.054108] scsi host2: usb-storage 3-1:1.0

[388.054519] scsi 2:0:0:0: Direct-Access Generic Flash Disk 8.07 PQ: 0 ANSI: 4

[388.076888] sd 2:0:0:0: [sda] 7598080 512-byte logical blocks: (3.89 GB/3.62 GiB)

[388.089891] sd 2:0:0:0: [sda] Write Protect is off

[388.098064] sd 2:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA

[388.116711] sda: sda1

[388.129424] sd 2:0:0:0: [sda] Attached SCSI removable disk

串口终端输入如下命令：

root@embest:~# ls /dev/sd*

/dev/sda /dev/sda1

/dev 下存在设备节点；

2. 从设备

连接 J10 到 PC 端，打开设备管理器，识别到如下设备：

2.14 WIFI 测试

2.14.1配置 WIFI 频段

开机进入系统后，默认为 2.4G 频段，如果要使用 5G 频段，请先配置 WIFI，提供两种配置方法，配置方法如下：

1. 进入目录 /usr/sbin/wlconf，输入命令./configure-device.sh

root@embest:~# cd /usr/sbin/wlconf/

root@embest:/usr/sbin/wlconf# ./configure-device.sh

根据提示 输入： y 1837 y 2 2,就可以了

Please provide the following information.

Are you using a TI module? [y/n] : **y**

What is the chip flavor? [1801/1805/1807/1831/1835/1837 or 0 for unknown] : **1837**

Should Japanese standards be applied? [y/n] : **y**

How many 2.4GHz antennas are fitted? [1/2] : **2**

How many 5GHz antennas are fitted? [0/1/2] : **2**

[1461.083174] wlcore: down

The device has been successfully configured.

TI Module: y

Chip Flavor: 1837

Number of 2.4GHz Antennas Fitted: 2

Number of 5GHz Antennas Fitted: 2

Diversity Support: y

SISO40 Support: y

Japanese Standards Applied: y

Class 2 Permissive Change (C2PC) Applied: n

root@embest:/usr/sbin/wlconf# [1461.954230] wlcore: wl18xx HW: 183x or 180x, PG 2.2 (ROM 0x11)

[1462.005515] wlcore: loaded

[1462.008412] wlcore: driver version: R8.6_SP1

[1462.362905] wlcore: PHY firmware version: Rev 8.2.0.0.233

[1462.595072] wlcore: firmware booted (Rev 8.9.0.1.55)

2. 进入目录 /usr/sbin/wlconf，输入命令：

root@embest:~# cd /usr/sbin/wlconf

root@embest:/usr/sbin/wlconf# ./wlconf -o /lib/firmware/ti-connectivity/wl18xx-conf.bin -I /usr/sbin/wlconf/official_inis/WG7833-B0A_INI_rev1.ini

两个方法任选其一就可以，配置之后就可以使用 5GWIFI 了，同时这个配置也是兼容 2.4G 的。使用 5G WIFI 只需要在第一次使用前配置一下，再次使用无需配置。

2.14.2连接 WIFI

在串口终端输入：

root@embest:~# cd /usr/share/wl18xx/

root@embest:/usr/share/wl18xx# ./sta_start.sh

root@embest:/usr/share/wl18xx# Successfully initialized wpa_supplicant

[94.422934] cfg80211: Calling CRDA for country: US

Could not read interface p2p-dev-wlan0 flags: No such device

[94.599340] cfg80211: Regulatory domain changed to country: US

[94.605627] cfg80211: DFS Master region: FCC

[94.610029] cfg80211: (start_freq - end_freq @ bandwidth), (max_antenna_gain, max_eirp), (dfs_cac_time)

[94.621813] cfg80211: (2402000 KHz - 2472000 KHz @ 40000 KHz), (N/A, 3000 mBm), (N/A)

[94.631326] cfg80211: (5170000 KHz - 5250000 KHz @ 80000 KHz, 160000 KHz AUTO), (N/A, 1700 mBm), (N/A)

[94.642261] cfg80211: (5250000 KHz - 5330000 KHz @ 80000 KHz, 160000 KHz AUTO), (N/A, 2300 mBm), (0 s)

[94.654119] cfg80211: (5490000 KHz - 5730000 KHz @ 160000 KHz), (N/A, 2300 mBm), (0 s)

[94.662666] cfg80211: (5735000 KHz - 5835000 KHz @ 80000 KHz), (N/A, 3000 mBm), (N/A)

[94.672235] cfg80211: (57240000 KHz - 63720000 KHz @ 2160000 KHz), (N/A, 4000 mBm), (N/A)

p2p-dev-wlan0: CTRL-EVENT-REGDOM-CHANGE init=USER type=COUNTRY alpha2=US

root@embest:/usr/share/wl18xx# ./sta_connect-ex.sh embest-test WPA-PSK 12345678

其中 embest-test 是 SSID， 12345678 是 Wi-Fi 密码

netid=0

==========

OK

OK

OK

OK

root@embest:/usr/share/wl18xx# wlan0: SME: Trying to authenticate with b0:48:7a[1017.520349] wlan0: authenticate with b0:48:7a:4b:0b:2a

:4b:0b:2a (SSID='embest-test' freq=2437 MHz)

[1017.531999] wlan0: send auth to b0:48:7a:4b:0b:2a (try 1/3)

[1017.571449] wlan0: authenticated

wlan0: Trying to associate with b0:48:7a:4b:0b:2a (SSID='embest-test' freq=2437 MHz)

[1017.583246] wlan0: associate with b0:48:7a:4b:0b:2a (try 1/3)

[1017.721188] wlan0: RX AssocResp from b0:48:7a:4b:0b:2a (capab=0x431 status=0 aid=2)

[1017.735614] wlan0: associated

wlan0: Associated with b0:48:7a:4b:0b:2a[1017.739377] cfg80211: Calling CRDA for country: US

[1017.764361] cfg80211: Regulatory domain changed to country: US

[1017.770526] cfg80211: DFS Master region: FCC

[1017.775904] cfg80211: (start_freq - end_freq @ bandwidth), (max_antenna_gain, max_eirp), (dfs_cac_time)

[1017.786369] cfg80211: (2402000 KHz - 2472000 KHz @ 40000 KHz), (N/A, 3000 mBm), (N/A)

[1017.795875] cfg80211: (5170000 KHz - 5250000 KHz @ 80000 KHz, 160000 KHz AUTO), (N/A, 1700 mBm), (N/A)

[1017.807298] cfg80211: (5250000 KHz - 5330000 KHz @ 80000 KHz, 160000 KHz AUTO), (N/A, 2300 mBm), (0 s)

[1017.818171] cfg80211: (5490000 KHz - 5730000 KHz @ 160000 KHz), (N/A, 2300 mBm), (0 s)

[1017.827331] cfg80211: (5735000 KHz - 5835000 KHz @ 80000 KHz), (N/A, 3000 mBm), (N/A)

[1017.836317] cfg80211: (57240000 KHz - 63720000 KHz @ 2160000 KHz), (N/A, 4000 mBm), (N/A)

p2p-dev-wlan0: CTRL-EVENT-REGDOM-CHANGE init=COUNTRY_IE type=COUNTRY alpha2=US

wlan0: WPA: Key negotiation completed with b0:48:7a:4b:0b:2a [PTK=CCMP GTK=TKIP]

wlan0: CTRL-EV[1017.906052] wlcore: Association completed.

ENT-CONNECTED - Connection to b0:48:7a:4b:0b:2a completed [id=3 id_str=]

用 ping 命令测试 wifi 连接

root@embest:/usr/share/wl18xx# ping www.baidu.com

PING www.a.shifen.com (103.235.46.39) 56(84) bytes of data.

64 bytes from 103.235.46.39: icmp_seq=1 ttl=50 time=122 ms

2.15 Bluetooth 测试

2.15.1复位蓝牙模块

执行以下 4 条命令复位:

root@embest:~# echo 0 > /sys/class/leds/ec8800\:bt_en/brightness

root@embest:~# echo 1 > /sys/class/leds/ec8800\:bt_en/brightness

root@embest:~# echo 0 > /sys/class/leds/ec8800\:bt_en/brightness

root@embest:~# echo 1 > /sys/class/leds/ec8800\:bt_en/brightness

2.15.2初始化蓝牙模块

root@embest:~# hciattach /dev/ttyS5 texas 115200

如果初始化成功，串口将打印如下信息：

Found a Texas Instruments' chip!

Firmware file : /lib/firmware/TIInit_11.8.32.bts

Loaded BTS script version 1

texas: changing baud rate to 3000000, flow control to 1

Device setup complete

2.15.3测试蓝牙功能

打开蓝牙模块：

root@embest:~# hciconfig hci0 up

开始扫描：

root@embest:~# hcitool scan

超级终端窗口显示信息如下：

Scanning ...

00:23:01:28:BD:5C Q8S

关闭蓝牙模块：

root@embest:~# hciconfig hci0 down

2.15.4测试蓝牙音频

该方法目前只支持 wav 的 44.1k 格式的音频，镜像提供了一个 文件名为 1K.wav 的文件，按照如下步骤播放这个文件后，可以听到“滋滋”的声音。

cd /root/BluetopiaPM/bin/

root@embest:~/BluetopiaPM/bin#./SS1BTPM &

root@embest:~/BluetopiaPM/bin#./LinuxAUDM

AUDM>1 1

AUDM>9 1

AUDM>27

AUDM>35 1

AUDM>33 0

AUDM>16 0

Notice: You can find your BT address in this step.

Eg.

AUDM>

Remote Device Found.

BD_ADDR: 00230128BD5C

COD: 0x040424

Device Name: Q8S

Device Flags: 0x80000605

RSSI: -51

Friendly Name:

App. Info: : 00000000

Paired State : TRUE

Connect State: FALSE

Encrypt State: FALSE

Sniff State : FALSE

Serv. Known : FALSE

AUDM>17

Notice: You can enter 17 after you find your BT address.

AUDM>37 0 [BD address]

此时会提示如下信息：

AUDM>37 0 00230128BD5C

AUDM_Connect_Audio_Stream() Success: 0.

AUDM>

Remote Device Properties Changed.

BD_ADDR: 00230128BD5C

Device Flags: 0x80000649

Connect State: TRUE

AUDM>

Authentication Request received for 00230128BD5C.

I/O Capability Request.

DEVM_AuthenticationResponse() Success.

AUDM>

Authentication Request received for 00230128BD5C.

I/O Capability Response.

Remote I/O Capabilities: No Input/Output, MITM Protection: FALSE.

AUDM>

Authentication Request received for 00230128BD5C.

User Confirmation Request.

User Confirmation: 802495

Respond with the command: UserConfirmationResponse

此时输入以下命令完成配对（命令后跟的参数由上述信息红色部分给出）

AUDM>UserConfirmationResponse User Confirmation

(eg. 802495)

用这个命令播放音频文件

AUDM>AUDPlayWAV [BD address] /boot/firmware/1k.wav

第3章 系统编译

3.1 配置编译环境

将 SBC-EC8800-Release-REV01 文件夹拷贝到 Linux 环境下的$HOME 目录下，编译工具 gcc-linaro-4.9-2015.05-x86_64_arm-linux-gnueabihf 在$HOME/SBC-EC8800-Release-REV01/tool 目录下，用如下命令解压：

$xz -d gcc-linaro-4.9-2015.05-x86_64_arm-linux-gnueabihf.tar.xz

$tar –xvf gcc-linaro-4.9-2015.05-x86_64_arm-linux-gnueabihf.tar

导入环境变量：

$export CROSS_COMPILE=$HOME/SBC-EC8800-Release-REV01/tool/gcc-linaro-4.9-2015.05-x86_64_arm-linux-gnueabihf/bin/arm-linux-gnueabihf-

$export ARCH=arm

3.2 编译 UBOOT

3.2.1 获取 uboot 源码

Uboot 源码在$HOME/SBC-EC8800-Release-REV01/sourcecode/目录下，解压 u-boot*.tar.gz :

$ cd $HOME/SBC-EC8800-Release-REV01/sourcecode/

$ tar -zxvf u-boot*.tar.gz

3.2.2 编译并烧写镜像到 SD 卡

$ cd $HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot*

$ make distclean

$make sbc_ec8800_defconfig

$make

编译完成后在$HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot*目录下生成 MLO, u-boot.img，将两个文件拷贝到 SD 卡中；

3.2.3 编译并烧写镜像 SPI Flash

$ cd $HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot*

$ make distclean

$make sbc_ec8800_qspiboot_defconfig

$make

编译完成后在$HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot*目录下生成 u-boot.bin，

$HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot*/spl 目录下生成 u-boot-spl.bin,将两个文件拷贝到 SD 卡中;

从 SD 卡启动，在 uboot 阶段执行：

U-Boot# run update_qspi_flash

等待执行结束，这两个文件就烧写到 SPI flash 中。

（参考 1.3 从 SPI Flash 启动系统）

3.3 Kernel

3.3.1 获取内核源码

内核源码存在$HOME/SBC-EC8800-Release-REV01/sourcecode/目录下,解压 linux*.tar.gz

$ cd $HOME/SBC-EC8800-Release-REV01/sourcecode/

$ tar -zxvf linux*.tar.gz

3.3.2 编译并烧写镜像到 SD 卡

$ cd $HOME/SBC-EC8800-Release-REV01/sourcecode/linux*

$ make distclean

$ make embest_ti_8800_defconfig

$ make

编译完成后：

- 在$HOME/SBC-EC8800-Release-REV01/sourcecode/linux*/arch/arm/boot 目录下生成 zImage
- 在$HOME/SBC-EC8800-Release-REV01/sourcecode/linux*/arch/arm/boot/dts 目录下生成：
  1. embest-SBC-EC8800-4.3inch_LCD.dtb
  2. embest-SBC-EC8800-7inch_LCD.dtb

dtb 文件分别对应 4.3 寸屏，7 寸屏（配置方法参考 LCD 测试;）

将文件拷贝到 SD 卡中。