User Manual

[SBC-EC8800]

Revision History

Rev.	Note	Author
20160307	Initial	Baijy
20160314	Translate	Sandy
20160315	Modify dtb files	Baijy
20160321	Modify some wrong instructions	Sandy
20160323	1. Add WIFI and Bluetooth test 2. Add PWRON RESETn Keypad test	Rongdong
20160331	Add Boot from SPI Flash	Baijy
20160622	Rev01 Release	Sandy

Catalog

Release Note

1. Images Version

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

2. Feature List

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

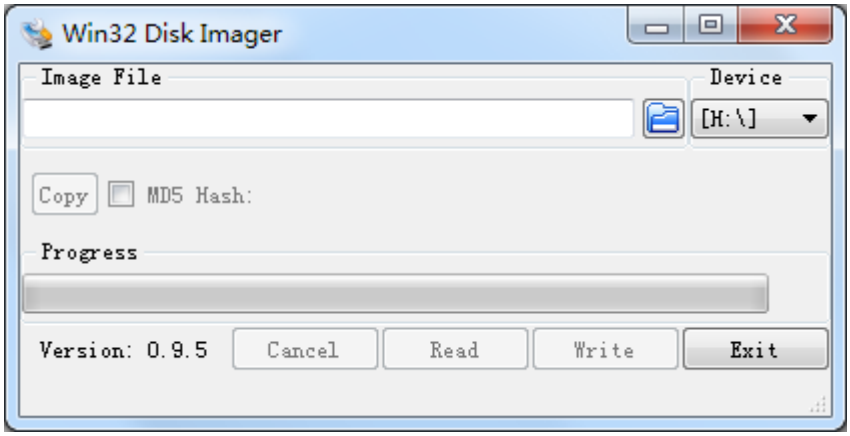
3. Known Issues

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.

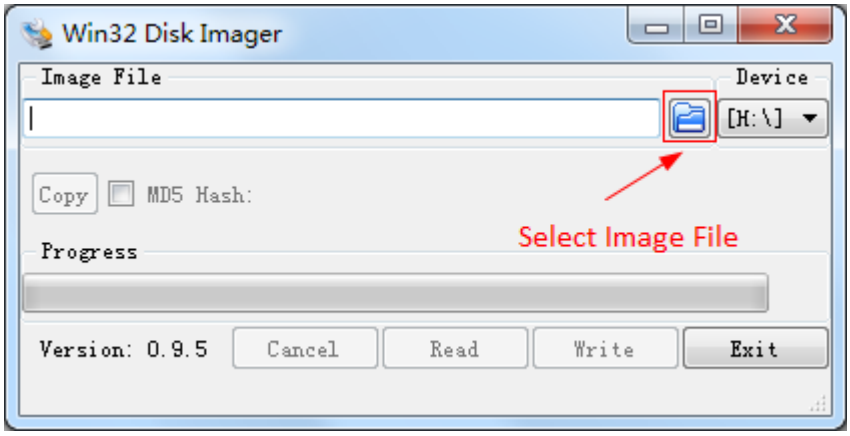
Chapter 1 Quick Start

1.1 Burn the System Images to the SD Card

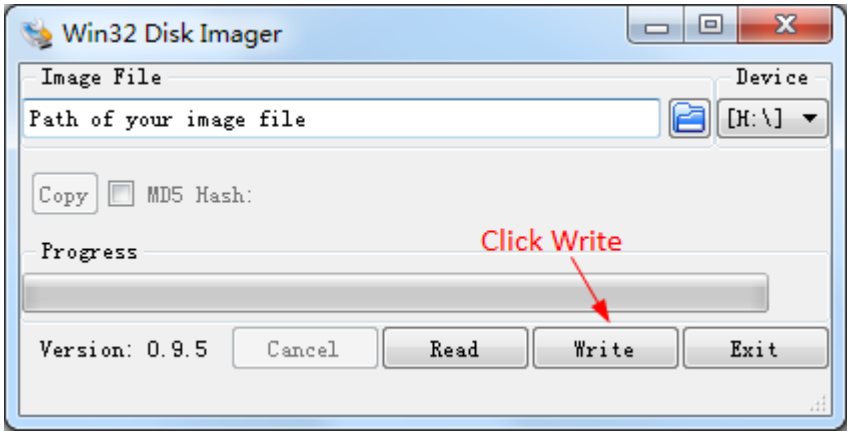
- Firstly, you should prepare a SD card, which is no less than 2GB.
- Then, download and install “Win32 Disk Imager” from https://sourceforge.net/projects/win32diskimager/.



- Select the system image: SBC-EC8800-Release-REV01\image\SBC-EC8800-Release-SDcard-EMMC-REV01.img



- Click “Write” button to burn the images:



1.2 System Boot from SD Card

- Install the Serial Communication software (e.g. SecureCRT), select the corresponding port number, baudrate as 115200, data bits as 8, stop bits as 1, parity as none.
- Connect the DEBUG interface (J13) to the serial interface of PC with a USB to TTL module.
- Insert the SD card into the card slot (J2).
- Power the board with a 5V, 2A power.
- Wait for the system boot up, then the serial output will show the following information:

```
[  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:
```

Enter username and password as “root” to login;

```
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 System Boot from SPI Flash

Refer to 1.2, boot the system from SD Card, press “Enter” when the serial terminal prints the following info:

```
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# (Press Enter now.)

Execute the following instructions on the serial terminal:

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

Enter following instruction to boot from SD Card first:

U-Boot# boot

Copy the PH8800-Release-SDcard-EMMC-REV01.img to a U-disk, then plug the U-disk to J9;

Execute the following instructions on the serial terminal:

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

/dev/sda /dev/sda1

root@embest:~# mount /dev/sda /mnt/

root@embest:~# dd if=/mnt/SBC-EC8800-Release-SDcard-EMMC-REV01.img of=/dev/mmcblk1

Note: Burn the EMMC takes a long time, please wait patiently.

Then power reset the board to boot from EMMC.

Chapter 2 Function test

First of all, please refer to Chapter 1.1 and boot up the system. Then test the functions according to the following guidance.

2.1 LED

User can control LED (D8, D9) indicators on SBC-EC8800 Board. After the system boot up, please execute the following instructions in serial terminal to implement the test; (D8 is attached to user_leds_d8, D9to user_leds_d9)

Light out LED:

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

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

Light up 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

Execute the following instructions on the serial terminal:

Check the current system time:

root@embest:~# date

Sat Jan 1 00:02:07 UTC 2000

Set current time as 10:46, March 9, 2016

root@embest: # date 030910462016

Wed Mar 9 10:46:00 UTC 2016

Write system clock into RTC:

root@embest: # hwclock –w

Read RTC value:

root@embest: # hwclock

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

The above information indicates that the hardware clock-RTC-has been set to March 9, 2016, so the system clock is saved in the hardware clock. Reboot the system and check the current system time:

root@embest:~# date

Wed Mar 9 10:46:45 UTC 2016

2.3 EEPROM

Execute the following instructions on the serial terminal:

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

If write and read data are the same, the test passes.

2.4 EMMC

Execute the following instructions on the serial terminal:

root@embest:~# touch emmc_read emmc_write

Modify emmc_write value:

root@embest:~# vi emmc_write

E.g. Write “emmc write test” into the system

Write emmc instructions:

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

Read emmc instructions:

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

10+0 records in

10+0 records out

10240 bytes (10 kB) copied, 0.00446492 s, 2.3 MB/s

Check emmc_read value:

root@embest:~# cat emmc_read

emmc write test

Test passes;

2.5 ADC

Execute the following instructions on the serial terminal to get the sampling values returned:

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

4.3” LCD:

Open the uEnv.txt file from SD card, modify fdtfile= embest-SBC-EC8800-4.3inch_LCD.dtb

Connect the screen module to J1, then reboot the system.

7” LCD:

Open the uEnv.txt file from SD card, modify fdtfile= embest-SBC-EC8800-7inch_LCD.dtb

Connect the screen module to J1, then reboot the system.

2.7 Backlight

The backlight brightness has a range from 1 to 8, in which 8 means highest brightness, 1 means lowest.

Execute the following instructions on the serial terminal to implement the backlight test:

The darkest:

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

The brightest:

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

2.8 Touchscreen

Connect the screen module to J1, execute the following instructions on the serial terminal to implement the touch screen calibration program:

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

Following the notes on LCD, click the “+” icon for five times to complete the calibration.

2.9 Serial

The board has 3 serial interfaces, while the UART0 (J13) is the debug interface. Execute the following instructions on the serial terminal to test UART 1and UART5:

2.9.1 UART1

Short Pin 8 and 10 in J11:

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

Note: Press “CTRL+C” to exit the serial test.

2.9.2 UART5

Short Pin 20 and 22 in J5:

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

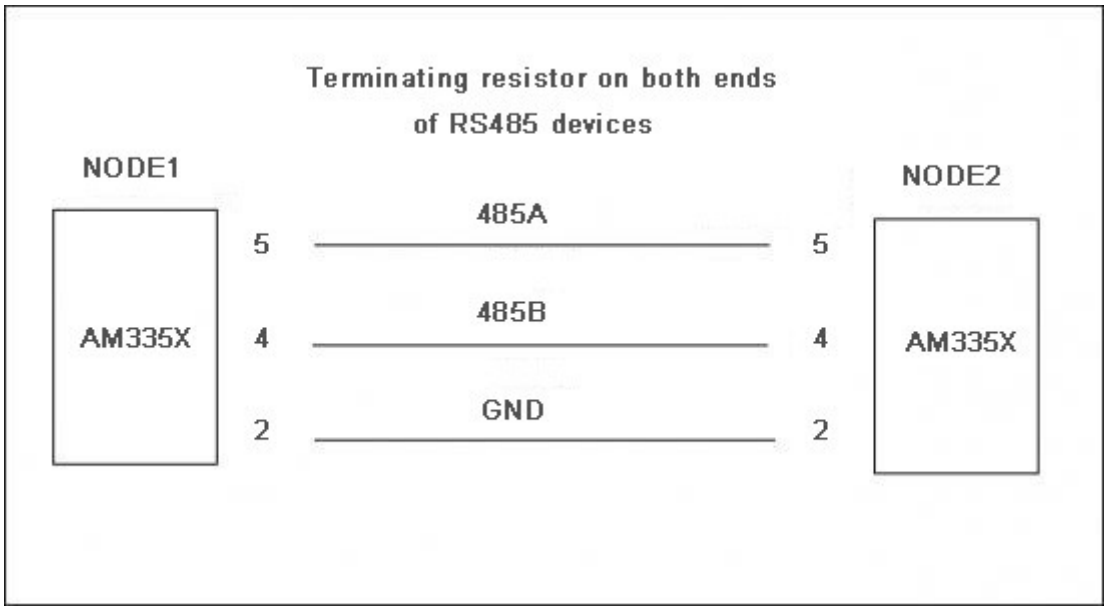
/dev/ttyS5 RECV 1 total

/dev/ttyS5 RECV: 0

Note: Press “CTRL+C” to exit the serial test.

2.10 RS485

SBC-EC8800 can be used as an RS485 device, the connect principle is shown in following diagram. Users need to find the corresponding pins according to the schematics, connect the RS485 interface on SBC-EC8800 to another RS485 device with cables.



Enter the following instructions from one device:

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.....

It will be working at receive status.

Enter the following instructions from the other device:

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
```

Send 10 data.

Check if the receive device receive data correctly.

2.11 CAN

SBC-EC8800 can be used as a CAN device. As SBC-EC8800 has 2 CAN modules, we can make an internal communication test use both CAN0 and CAN1 by connect them together. Connect Pin 33 and 34, Pin 35 and 36, Pin 37 and 38 in J5.

Test method as follows:

1. Open CAN0 and 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 receive data, CAN0 send data to CAN1

```
root@embest:~#    candump can1&
root@embest:~#    cansend can0 123#11223344556677
```

3. Shut off the device after test finished.

```
root@embest:~#    ip link set can0 down
root@embest:~#    ip link set can1 down
```

Users can do the transceiving test according to the above instructions, and set different baudrate to communicate.

Note you must shut off the device before set a different baudrate. Effective baudrate contains:

- 25KBPS (250000)
- 50KBPS (50000)
- 125KBPS (125000)
- 500KBPS (500000)
- 650KBPS (650000)
- 1MKBPS (1000000)

The board can communicate at the above baudrate. Users can also test with other baudrate to see if the device works too. The CAN module can also connect a CAN module from other board to test.

2.12 Network

Connect net cable to J, execute the following instructions on the serial terminal:

Configure the IP address:

root@embest:~# ifconfig eth0 192.168.2.64

Testing network interface:

root@embest:~# ping 192.168.2.1

2.13 USB

2.13.1 USB Host

Insert the U disk to the USB Host interface (J9), serial terminal will display the disk information:

```
[    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
```

Execute the following instructions on the serial terminal:

```
root@embest:~# ls /dev/sd*
/dev/sda      /dev/sda1
```

Storage nodes locate under /dev;

2.13.2 USB OTG

1. Master Device

Connect U disk to J10 with an OTG cable:

```
[    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
```

Execute the following instructions on the serial terminal:

```
root@embest:~# ls /dev/sd*
/dev/sda      /dev/sda1
```

Storage nodes locate under /dev;

2. Slave Device

Connect J10 to PC, open the device manager, and check if the following device is recognized:

2.14 WIFI

2.14.1 Configure WIFI Antennas

After the first boot, the wifi is working at 2.4GHz frequency in default, if you need to use the 5GHz frequency, please configure the WIFI module at first. Here we provide two methods:

1. Enter the path /usr/sbin/wlconf, enter command ./ configure-device.sh

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

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

Then enter “y 1837 y 2 2” according to the prompt:

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. Enter path /usr/sbin/wlconf, enter the command:

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

You just need to choose one method, then you can use 5G WIFI. This configuration support 2.4G, too. The operation only need to be executed before the first use of WIFI. You don’t need to execute again when you open WIFI or boot system again.

2.14.2 Connect WIFI

Execute the following instructions on the serial terminal:

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

Note: In above instructions, “embest-test” is the SSID of the WIFI, “12345678” is the password.

Then the serial terminal will show:

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=]

Test the wifi connection with ping command:

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

2.15.1 Reset Bluetooth Module

Execute the following 4 instructions to reset the module

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 Initialize the Bluetooth Module

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

If the initialization success, serial terminal will print the following information:

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 Bluetooth Scan Test

Open Bluetooth:

root@embest:~# hciconfig hci0 up

Start Bluetooth scan:

root@embest:~# hcitool scan

Serial terminal will print the following information:

Scanning ...

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

Shut off Bluetooth:

root@embest:~# hciconfig hci0 down

2.15.4 Bluetooth Audio Test

Currently, this method only supports audio format of wav, 44.1K. The image provide file “1K.wav” to test the function. Play the file follow the below operation, you can hear sounds like “zizi” from the Bluetooth audio play device.

c 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]

Then the Bluetooth module will print following info:

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

Then enter following command to finish the match (the parameter is provided by the red string in above info)

AUDM>UserConfirmationResponse User Confirmation

(eg. 802495)

Play the Audio file with the following command:

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

Chapter 3 System Compilation

3.1 Building Development Environment

Copy the release folder to Linux's $HOME directory, while the compilation tool gcc-linaro-4.9-2015.05-x86_64_arm-linux-gnueabihf under path $HOME/SBC-EC8800-Release-REV01/tool. Use the following instructions to extract it:

$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

Import the environment variable:

$export CROSS_COMPILE=$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 Compiling U-Boot

3.2.1 Get the U-Boot Source Code

U-boot source code locates under path $HOME/SBC-EC8800-Release-REV01/sourcecode/, extract the u-boot*.tar.gz:

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

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

3.2.2 Compile and Burn the Images to SD Card

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

$ make distclean

$make sbc_ec8800_defconfig

$make

When the compilation finished, it will generate a MLO and u-boot.img under path $HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot, copy the two files to SD card;

3.2.3 Compile and Burn the Images to SPI Flash

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

$ make distclean

$make sbc_ec8800_qspiboot_defconfig

$make

When the compilation finished, it will generate:

1. **u-boot.bin** under path $HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot*
2. **u-boot-spl.bin** under path $HOME/SBC-EC8800-Release-REV01/sourcecode/u-boot*/spl

Copy the two files to SD card;

Boot from SD card, execute the following instructions in U-Boot phase:

U-Boot# run update_qspi_flash

Wait for the execute finished, the two files are burn into SPI flash.

(Refer to 1.3 System Boot from SPI Flash)

3.3 Compiling Kernel

3.3.1 Get Kernel Source Code

The source code of the kernel locate under $HOME/SBC-EC8800-Release-REV01/sourcecode/, extract the linux*.tar.gz

$ tar -zxvf linux*.tar.gz

3.3.2 Compile and Burn the Images to SD Card

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

$ make distclean

$ make embest_ti_8800_defconfig

$ make

When the compilation finished, it will generate

- zImage under $HOME/SBC-EC8800-Release-REV01/sourcecode/linux*/arch/arm/boot;
- the following 2 files under $HOME/SBC-EC8800-Release-REV01/sourcecode/linux*/arch/arm/boot/dts
  1. embest-SBC-EC8800-4.3inch_LCD.dtb
  2. embest-SBC-EC8800-7inch_LCD.dtb

The dtb files are corresponding for 4.3” LCD, 7” LCD. (Refer to and 2.6 LCD)

Copy the files to SD Card.