User Manual

[SBC-EC9100]

Revision History

Rev.	Note	Author
20160907	Initial	Sandy

Catalog

Release Note

1. Images Version

91200003_SBC-EC9100-SDcard_Shipment_Image_REV00.img

91200003_SBC-EC9100-EMMC_Shipment_Image_REV00.img

2. Feature List

Feature List	SBC-EC9100			
	Schematic Page#	On-Chip Peripherals	On-Board Peripherals	Detail Functions(Needed)
u-boot version	2015.04			Can only boot the kernel
kernel version	3.14.52			Kernel will support all the function below
Filesystem	buildroot			buildroot of embest own
CPU	EC9100-U2	imx6ul		
DDRAM	EC9100-U7	DDR3	MT41K256M16HA-125	Can access read write and run code
MicroSD_(TF)	EC9100-J3	MMC0	Null	Can access read write and boot
eMMC	EC9100-U6	MMC1		Can access read write
UART-0		UART1		System can send and receive data in loopback mode
UART-1		RS485		System can send and receive data in loopback mode
UART-2		UART3		Debug Serial
PMIC	EC9100-U3	I2C0	PZUFE3001	
Ethernet-1	EC9100-U8	RGMII1	KSZ8081RNXIA	
USB-Host	EC9100-J7	USB1		Can recognize U disk by USB host
USB-OTG	EC9100-J8	USB0	Null	Can recognize U disk by USB host
LCD	EC9100-J5	RGB	Null	Can show picture on the screen
Backlight	EC9100-U24	PWM	Null	System can control the LCD backlight
Touchscreen	EC9100-J5	ADC-TSC	Null	
EEPROM	EC9100-U25	I2C0	AT24C256W	Can access read write

CAN-1	EC9100-u14	CAN1	mc33901wef	System can send and receive data in loopback mode
CAN-2	EC9100-U17	CAN0	mc33901wef	System can send and receive data between two board
RS485-1	EC9100-MN1	UART1	ADM3485	System can send and receive data between two board
debian filesystem				
ADC		Null	Null	
CAMERA		CSI&I2C1	Null	
WIFI	EC9100-U21	SDIO2	Null	
SPI		SPI3	Null	

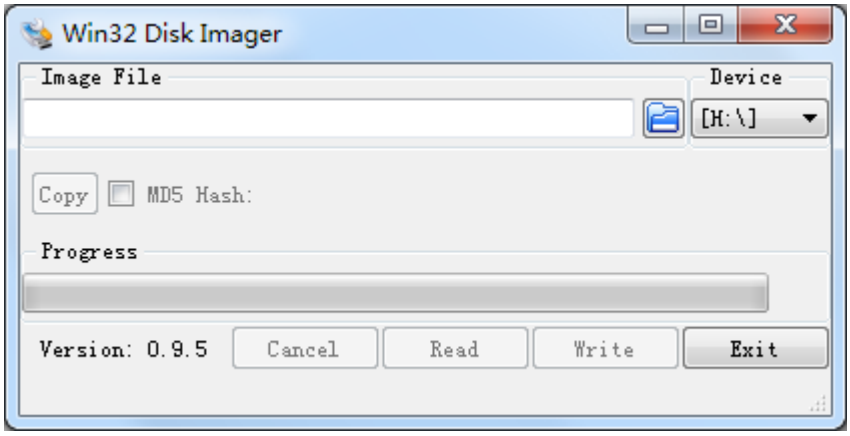
3. Known Issues

1. The highest resolution of Camera is 720*576 resolution.
2. Wifi module is selectable ,but now cannot work;

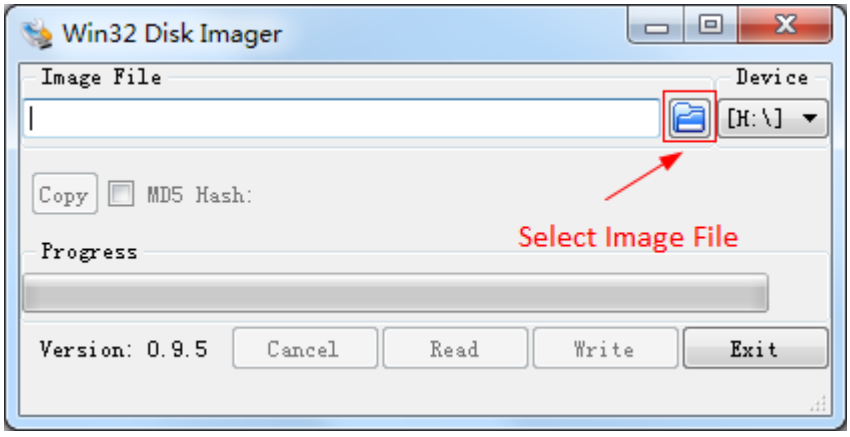
Chapter 1 Quick Start

1.1 Burn the System Images to the SD Card

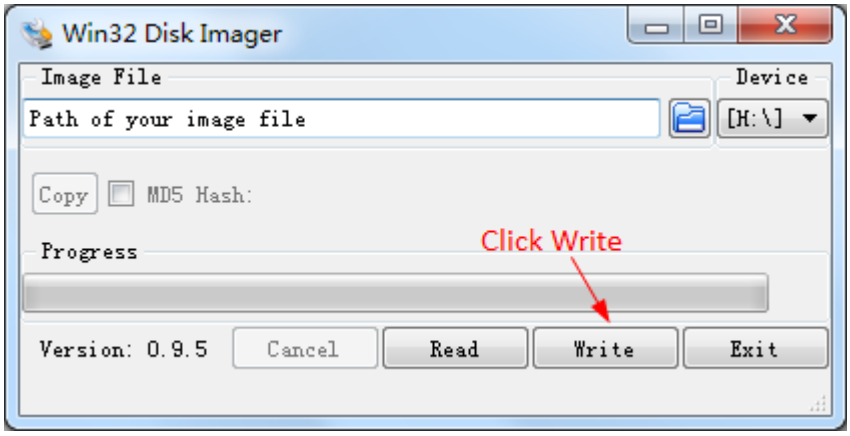
- Firstly, you should prepare a SD card, which is no less than 1GB.
- Then, download and install “Win32 Disk Imager” from https://sourceforge.net/projects/win32diskimager/.



- Select the system images file: 91200003_SBC-EC9100-SDcard_Shipment_Image_REV00.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 Uart Tx (Pin 8 in J10) and Rx (Pin 10 in J10) to PC with USB to TTL convertor.
- Insert the SD card into the card slot J3.
- Powered the board with a 5V, 2A power (J1).
- Change the dial switch SW3 to 00, SW4 to 0010.
- Wait for the system boot up, then the serial output will show the following information:

```
[  OK  ] Started System Logging Service.
         Starting Getty on tty1...
[  OK  ] Started Getty on tty1.
         Starting Serial Getty on ttymxc2...
[  OK  ] Started Serial Getty on ttymxc2.
[  OK  ] Reached target Login Prompts.
[  OK  ] Started Embest AutoExec Service.
[  OK  ] Started Login Service.

Debian GNU/Linux 8 embest ttymxc2

embest login:
```

Enter username and password as “root” to login;

```
Debian GNU/Linux 8 embest ttymxc2

embest login: root
Password:
Linux embest 3.14.52 #1 SMP PREEMPT Wed Aug 31 12:08:45 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 EMMC

Copy the 91200003_SBC-EC9100-EMMC_Shipment_Image_REV00.img to a U-disk;

Refer to 1.2, boot the system from SD Card, then plug the U disk to USB interface J8:

Execute the following instructions on the serial terminal:

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

/dev/sda /dev/sda1

root@embest:~# mount /dev/sda1 /mnt/

root@embest:~# dd if=/mnt/91200003_SBC-EC9100-EMMC_Shipment_Image_REV00.img of=/dev/mmcblk1

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

If it is the first time to burn the EMMC, you need to Enable boot from EMMC user partition, see chapter 3.2.3. Then you don't need to do this operation anymore.

Power off the board after the burning finished, change the dial switch SW3 to 10, SW4 to 0110, plug out the SD card, power on 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 Button

Insert the micro SD card to SBC-EC9100, power on the board, boot the system.

Press the RESET after system boot, it will reboot the system.

Push ON/OFF for more than 8 seconds, system will halt, then push 8 seconds again, system will reboot.

Users can also test it using the following instructions:

root@embest:~# evtest /dev/input/event0

Input driver version is 1.0.1

Input device ID: bus 0x19 vendor 0x0 product 0x0 version 0x0

Input device name: "20cc000.snvs-pwrkey"

Supported events:

  Event type 0 (EV_SYN)

  Event type 1 (EV_KEY)

    Event code 116 (KEY_POWER)

Properties:

Testing ... (interrupt to exit)

Press the button:

Event: time 1469459707.798194, type 1 (EV_KEY), code 116 (KEY_POWER), value 1

Event: time 1469459707.798194, ---------- EV_SYN ----------

Event: time 1469459708.038234, type 1 (EV_KEY), code 116 (KEY_POWER), value 0

Event: time 1469459708.038234, ---------- EV_SYN ----------

Event: time 1469459710.058188, type 1 (EV_KEY), code 116 (KEY_POWER), value 1

Event: time 1469459710.058188, ---------- EV_SYN ----------

Event: time 1469459710.238220, type 1 (EV_KEY), code 116 (KEY_POWER), value 0

Event: time 1469459710.238220, ---------- EV_SYN ----------

2.2 RTC

Connect the battery module to the board, then 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

mmcblk1: p1 p2

0+1 records in

0+1 records out

16 bytes (16 B) copied, 0.0396547 s, 0.4 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, 0.0179817 s, 569 kB/s

Check emmc_read value:

root@embest:~# cat emmc_read

emmc write test

Test passes;

2.5 GPIO

SBC-EC9100 have 4 pins which can be used as GPIO controlled output, they are:

- GPIO4 (Pin 8 in J9)
- GPIO5 (Pin 7 in J9)
- GPIO9 (Pin 23 in J9)
- GPIO132 (Pin 7 in J10)

Users can execute the following instruction to test GPIO function:

Example: Test GPIO4

1. Enable the GPIO and initialize it:

root@embest:~# echo 4 > /sys/class/gpio/export

root@embest:~# echo out > /sys/class/gpio/gpio4/direction

2. Set the level of pin and test the voltage of the pin:

root@embest:~# echo 1 > /sys/class/gpio/gpio4/value

Now the pin output should be high level

root@embest:~# echo 0 > /sys/class/gpio/gpio4/value

Now the pin output should be low level

Change the “4” to 5, 9, 132 in above instructions to test GPIO5 (Pin 7 in J9), GPIO9 (Pin 23 in J9) and GPIO132 (Pin 7 in J10).

2.6 LCD

Connect the screen module to J5,

4.3” LCD:

Open the uEnv.txt file from SD card, modify fdtfile=embest_fsl_sbc_ec9100_4.3inch.dtb

Then reboot the system;

7” LCD:

Open the uEnv.txt file from SD card, modify fdtfile=embest_fsl_sbc_ec9100_7inch.dtb

Then reboot the system;

Backlight test:

Execute the following instruction in the serial terminal

root@embest:~# echo 0 > /sys/class/backlight/backlight.9/brightness

echo from 0 to 7 to find the screen backlight change.

2.7 Touchscreen

Connect the screen module to J5, make sure fuse is written. Read the fuse value by the following instruction. (You just need to set bit 20 and bit21 to 1, eg. 0x003000xx)

root@embest:~# cat /sys/fsl_otp/HW_OCOTP_CFG2

To set fuse value, execute the following instruction (xx change to the value you cat):

root@embest:~# echo 0x003000xx > /sys/fsl_otp/HW_OCOTP_CFG2

Note: The value of different board may not be same, but it has no influence on the test.

Enter the following instruction to make the touch calibrate function work:

root@embest:~# export TSLIB_TSDEVICE=/dev/input/event1

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.

2.8 Serial

The board has 2 serial interfaces, while the UART-3 is the debug interface. Short connect J203 to configure the iomux IC working at UART & CAN mode.

2.8.1 Loopback

UART2 test, short connect Pin 20 and 22 in J9:

Execute the following instructions on the serial terminal:

root@embest:~# ./uart_test -d /dev/ttymxc1 -b 115200

/dev/ttymxc1 SEND: 1234567890

/dev/ttymxc1 RECV 10 total

/dev/ttymxc1 RECV: 1234567890

2.8.2 Board to Board

Use two SBC-EC9100 boards, connect the Rx pin (22 in J9) with the Tx pin of the other board (Pin 20 in J9), Tx pin to the Rx pin of the other board.

Execute the following instructions on the serial terminal for each board:

root@embest:~# ./uart_test -d /dev/ttymxc1 -b 115200

Two boards will both send and receive data.

The serial terminal will print the following info:

/dev/ttymxc1 SEND: 1234567890

/dev/ttymxc1 RECV 10 total

/dev/ttymxc1 RECV: 1234567890

……

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

2.9 RS485

Use two SBC-EC9100 boards to make the test, connect the pin 39 and 40 on board to the other board, then:

Execute the following instructions on board A:

root@embest:~# ./uart_test2 /dev/ttymxc0 9600 0 100

Execute the following instructions on board B:

root@embest:~# ./uart_test2 /dev/ttymxc0 9600 1

If test passed, board B will receive info like following:

*****data length = 31 ******

41 54 31 32 33 34 35 36 37 38 39 58 59 5a 61 62 63 64 65 64 66 68 69 6a 6b 6c 6d 6e 0d 00 fb

******* receive data 135667 pkts...0 bytes.....

Then switch the execute instruction.

2.10 CAN

Connect CAN0 with CAN1 by connect Pin 33 to Pin 34, Pin 35 to Pin 36 in J9.

Test method as below:

1. Open 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

flexcan 2090000.can can0: writing ctrl=0x27292085

root@embest:~# ip link set can1 up

flexcan 2094000.can can1: writing ctrl=0x27292085

2. Send and Receive

CAN1 receive data, CAN0 send data to CAN1, if the following info printed, the test passes:

root@embest:~# candump can1&

[4] 589

root@embest:~# cansend can0 123#01020304050607

 can1 123 [7] 01 02 03 04 05 06 07

Use show command to check the status of CAN.

CAN0 Tx added 1 packet, 7 bytes. CAN1 Rx added 1 packet, 7 bytes.

root@embest:~# ip -d -s link show can0

2: can0: <NOARP,UP,LOWER_UP,ECHO> mtu 576 qdisc pfifo_fast state UNKNOWN mode DEFAULT group default qlen 10

    link/can promiscuity 0

    can <TRIPLE-SAMPLING> state ERROR-ACTIVE (berr-counter tx 0 rx 0) restart-ms 0

          bitrate 50000 sample-point 0.866

          tq 1333 prop-seg 6 phase-seg1 6 phase-seg2 2 sjw 1

          flexcan: tseg1 4..16 tseg2 2..8 sjw 1..4 brp 1..256 brp-inc 1

          clock 30000000

          re-started bus-errors arbit-lost error-warn error-pass bus-off

```
          0          0          0          0          0          0
    RX: bytes  packets  errors  dropped overrun mcast
    0          0          0          0          0          0
    TX: bytes  packets  errors  dropped carrier collsns
    7          1          0          0          0          0
```

Note: Two CAN modules master be set at the same baudrate.

2.11 Network

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
```

2.12 USB

2.12.1 Host

2.12.1.1 U-disk

Insert the U-disk to the USB Host interface (J8), serial terminal will display the disk information:

```
usb 2-1: USB disconnect, device number 3
usb 2-1: new high-speed USB device number 4 using ci_hdrc
usb-storage 2-1:1.0: USB Mass Storage device detected
scsi0 : usb-storage 2-1:1.0
scsi 0:0:0:0: Direct-Access     Generic  Flash Disk       8.07 PQ: 0 ANSI: 4
sd 0:0:0:0: [sda] 7823360 512-byte logical blocks: (4.00 GB/3.73 GiB)
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
 sda: sda1
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;

2.12.1.2 Keyboard

Connect USB Keyboard to the USB Host interface (J8), serial terminal will print as follows:

usb 2-1: new low-speed USB device number 3 using ci_hdrc

input: SIGMACH1P USB Keykoard as /devices/soc0/soc.0/2100000.aips-bus/2184200.usb/ci_hdrc.1/usb2/2-1/2-1:1.0/0003:1C4F:0002.0003/input/input4

hid-generic 0003:1C4F:0002.0003: input: USB HID v1.10 Keyboard [SIGMACH1P USB Keykoard] on usb-ci_hdrc.1-1/input0

input: SIGMACH1P USB Keykoard as /devices/soc0/soc.0/2100000.aips-bus/2184200.usb/ci_hdrc.1/usb2/2-1/2-1:1.1/0003:1C4F:0002.0004/input/input5

hid-generic 0003:1C4F:0002.0004: input: USB HID v1.10 Device [SIGMACH1P USB Keykoard] on usb-ci_hdrc.1-1/input1

Execute evtest command to test /dev/input/event2:

root@embest:~# evtest /dev/input/event2

Input driver version is 1.0.1

Input device ID: bus 0x3 vendor 0x1c4f product 0x2 version 0x110

Input device name: "SIGMACH1P USB Keykoard"

Supported events:

Event type 0 (EV_SYN)

Event type 1 (EV_KEY)

Event code 1 (KEY_ESC)

Event code 2 (KEY_1)

Event code 3 (KEY_2)

Event code 192 (KEY_F22)

Event code 193 (KEY_F23)

Event code 194 (KEY_F24)

Event code 240 (KEY_UNKNOWN)

Event type 4 (EV_MSC)

Event code 4 (MSC_SCAN)

Event type 17 (EV_LED)

Event code 0 (LED_NUML)

Event code 1 (LED_CAPSL)

Event code 2 (LED_SCROLLL)

Key repeat handling:

Repeat type 20 (EV_REP)

Repeat code 0 (REP_DELAY)

Value 250

Repeat code 1 (REP_PERIOD)

Value 33

Properties:

Testing ... (interrupt to exit)

Press the key of the key board, the corresponding data for the key will be printed:

Event: time 73542.642111, type 4 (EV_MSC), code 4 (MSC_SCAN), value 70017

Event: time 73542.642111, type 1 (EV_KEY), code 20 (KEY_T), value 1

Event: time 73542.642111, ---------- EV_SYN ----------

Event: time 73542.762091, type 4 (EV_MSC), code 4 (MSC_SCAN), value 70017

Event: time 73542.762091, type 1 (EV_KEY), code 20 (KEY_T), value 0

Event: time 73542.762091, ---------- EV_SYN ----------

Event: time 73544.202085, type 4 (EV_MSC), code 4 (MSC_SCAN), value 7000a

Event: time 73544.202085, type 1 (EV_KEY), code 34 (KEY_G), value 1

Event: time 73544.202085, ---------- EV_SYN ----------

Event: time 73544.290084, type 4 (EV_MSC), code 4 (MSC_SCAN), value 7000a

Event: time 73544.290084, type 1 (EV_KEY), code 34 (KEY_G), value 0

Event: time 73544.290084, ---------- EV_SYN ----------

2.12.2 OTG

Connect U-disk or USB keyboard to J7 with an OTG cable, test command and phenomenon is similar with Host Test, while the USB id is different:

2.12.2.1 U-Disk

ci_hdrc ci_hdrc.0: timeout waiting for 00000800 in 12

ci_hdrc ci_hdrc.0: EHCI Host Controller

ci_hdrc ci_hdrc.0: new USB bus registered, assigned bus number 1

ci_hdrc ci_hdrc.0: USB 2.0 started, EHCI 1.00

hub 1-0:1.0: USB hub found

hub 1-0:1.0: 1 port detected

usb 1-1: new high-speed USB device number 2 using ci_hdrc

usb-storage 1-1:1.0: USB Mass Storage device detected

scsi1 : usb-storage 1-1:1.0

scsi 1:0:0:0: Direct-Access Generic Flash Disk 8.07 PQ: 0 ANSI: 4

sd 1:0:0:0: [sda] 7823360 512-byte logical blocks: (4.00 GB/3.73 GiB)

sd 1:0:0:0: [sda] Write Protect is off

sd 1:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA

sda: sda1

sd 1:0:0:0: [sda] Attached SCSI removable disk

2.12.2.2 Keyboard

USB Keyboard:

usb 1-1: new low-speed USB device number 4 using ci_hdrc

input: SIGMACH1P USB Keykoard as /devices/soc0/soc.0/2100000.aips-bus/2184000.usb/ci_hdrc.0/usb1/1-1/1-1:1.0/0003:1C4F:0002.0003/input/input4

hid-generic 0003:1C4F:0002.0003: input: USB HID v1.10 Keyboard [SIGMACH1P USB Keykoard] on usb-ci_hdrc.0-1/input0

input: SIGMACH1P USB Keykoard as /devices/soc0/soc.0/2100000.aips-bus/2184000.usb/ci_hdrc.0/usb1/1-1/1-1:1.1/0003:1C4F:0002.0004/input/input5

hid-generic 0003:1C4F:0002.0004: input: USB HID v1.10 Device [SIGMACH1P USB Keykoard] on usb-ci_hdrc.0-1/input1

2.12.2.3 Slave Device

Connect J7 to PC, open the device manager, and check if the following device is recognized:

2.13 Camera

2.13.1 Record Video

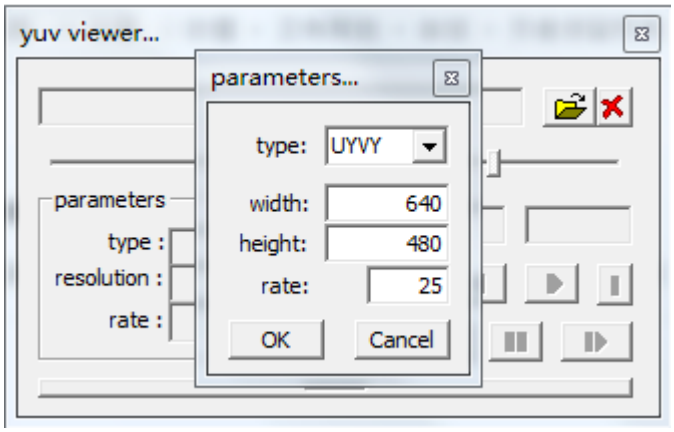
Connect Camera module to J6, execute the following instructions on the serial terminal:

```
root@embest:~# ./mxc_v4l2_capture -iw 640 -ih 480 -ow 640 -oh 480 -c 25 -f UYVY /boot/firmware/test.yuv
root@embest:~# sync
```

Camera will record a video with 640*480 resolution, rate 25, generate the video file test.yuv in SD card folder.

Connect SD card to PC, open it with Pyuv.exe.

Parameters of Pyuv.exe should be set as follows:



Note: Pyuv.exe is provided from tool folder.

2.13.2 Record Photo

Connect screen and camera, then execute the following instructions on the serial terminal:

```
root@embest:~# ./v4l2_capture_jpeg img1.jpg
```

SBC-EC9100 will display the photo on screen.

Chapter 3 System Compilation

3.1 Building Development Environment

Copy the SBC-EC9100-Release-REV01 folder to Linux's $HOME directory, while the compilation tool fsl-linaro-toolchain-master.tar.gz locate under path $HOME/S5_Tool. Use the following instructions to extract it:

$tar –xzvf fsl-linaro-toolchain-master.tar.gz

Import the environment variable:

$export CROSS_COMPILE=$HOME/S5_Tool/fsl-linaro-toolchain-master/bin/arm-fsl-linux-gnueabi-

$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/S4_Sourcecode, extract the u-boot*.tar.gz:

$ cd $HOME/S4_Sourcecode

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

3.2.2 Compile and Burn the Images to SD Card

$ cd $HOME/S4_Sourcecode/u-boot

$ make distclean

$make embest_fsl_ec9100_sdcard_defconfig

$make

When the compilation finished, it will generate a **u-boot.imx** under path $HOME/S4_Sourcecode/u-boot. Burn this file to SD Card use dd command:

$ sudo dd if=u-boot.imx of=/dev/sdx bs=512 seek=2 conv=fsync

Note: You need to change sdx to the actual SD card name on your Linux system like sda, sdb, etc.

3.2.3 Compile and Burn the Images to EMMC

$ cd $HOME/S4_Sourcecode/u-boot

$ make distclean

$make embest_fsl_ec9100_emmc_defconfig

$make

When the compilation finished, it will generate a **u-boot.imx** under path $HOME/S4_Sourcecode/u-boot. Copy the file to SD Card:

Refer to 1.2, first, you need to boot up the system from SD card. Then, type the following command on the system which already running on SD card to format and burn EMMC.

1. Prepare the EMMC partition

$ sudo dd if=/dev/zero of=/dev/mmcblk1 bs=1K count=1

$ echo -e " o\nn\np\n1\n20480\n+64M\na\nt\nc\nn\np\n2\n151552\n\nw\n " | fdisk /dev/mmcblk1

$ sudo mkfs.vfat /dev/mmcblk1p1

$ sudo mkfs.ext4 /dev/mmcblk1p2

$ sudo fdisk /dev/mmcblk1 –l

2. Burn the u-boot.imx (for emmc) to EMMC use dd command

$ sudo dd if=u-boot.imx of=/dev/mmcblk1 bs=512 seek=2 conv=fsync

3. Enable boot from EMMC user partition

$ sudo echo 56 > /sys/block/mmcblk1/device/boot_config

4. Check whether the enable operation work

$ sudo cat /sys/block/mmcblk1/device/boot_info

If the operation works, it will print the following info:

boot_partition:0x78;

BOOT_ACK:1 - Boot acknowledge sent during boot operation

BOOT_PARTITION-ENABLE: 7 - User area enabled for boot

Now the U-boot is burned to EMMC

3.3 Kernel

3.3.1 Get Kernel Source Code

The source code of the kernel locate under $HOME/S4_Sourcecode/, extract the linux*.tar.gz

$ tar -zxvf linux*.tar.gz

3.3.2 Compile and Burn the Images to SD Card

$ cd $HOME/S4_Sourcecode/linux

$ make distclean

$ make embest_fsl_sbc_ec9100_defconfig

$ make

When the compilation finished, it will generate

- zImage under $HOME/S4_Sourcecode/linux/arch/arm/boot;
- following dtb files under $HOME/ sourcecode/linux/arch/arm/boot/dts
  1. embest_fsl_sbc_ec9100_4.3inch.dtb
  2. embest_fsl_sbc_ec9100_7inch.dtb

  The dtb file is corresponding for 4.3” LCD and 7” LCD. (Refer to LCD test)

  Copy the files to SD Card.