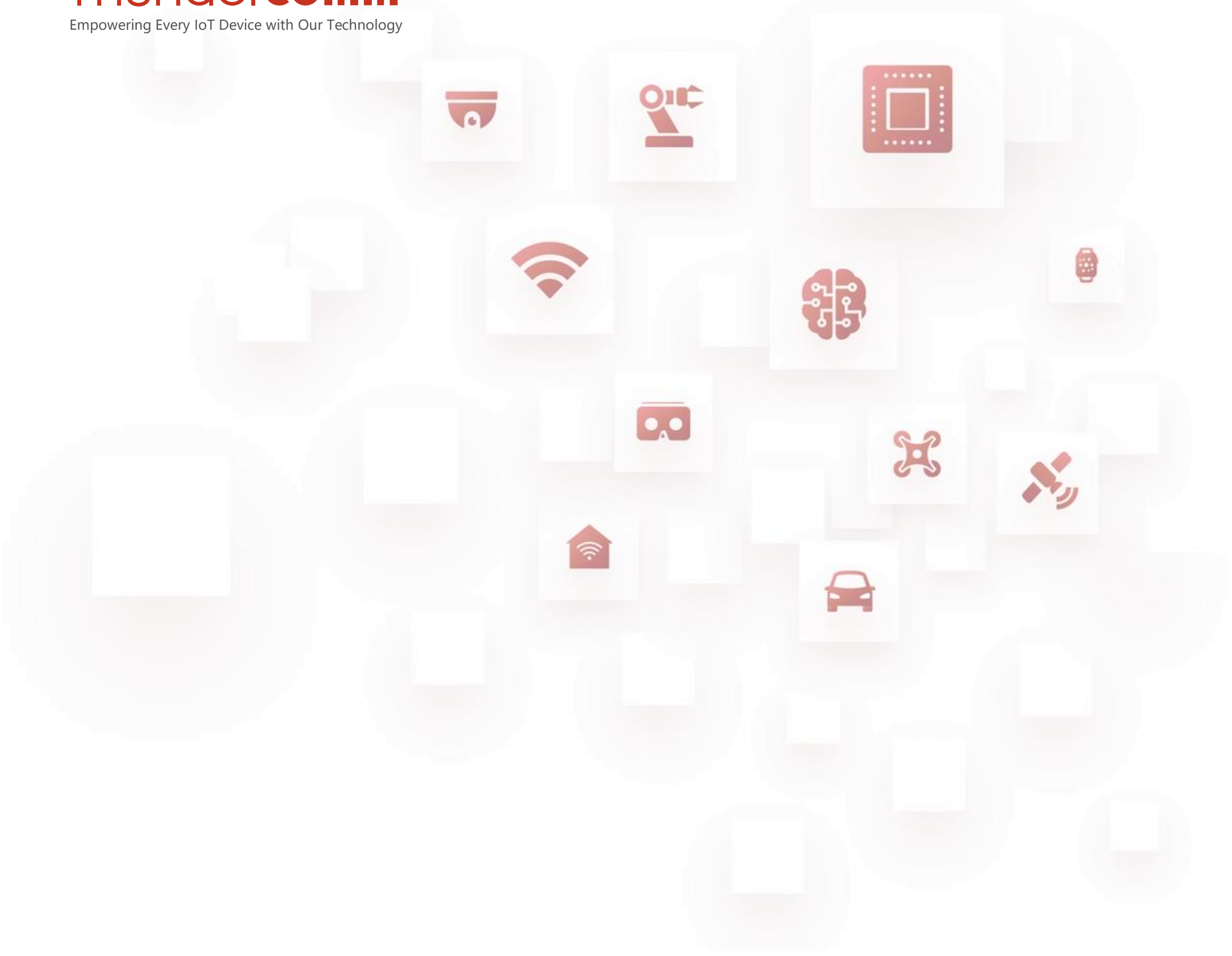
Thundercomm
Empowering Every IoT Device with Our Technology



Thundercomm Turbox™ C410/C610 Open Kit Linux Enablement

Software User Manual

Rev. V2.0
Apr. 01, 2021

DN: TC_C610LE_2320

Revision History

Revision	Date	Description
1.0	Aug. 15, 2020	Initial release.
2.0	Apr. 01, 2021	Update Display port. add Digital zoom. add Lens distortion correction. add Video. add Face detection. add Sleep mode.

Table List

Table 1-1: Product overview

Table 1-2: Features and specifications

About this document

- Illustrations in this documentation might look different from your product.
- Depending on the model, some optional accessories, features, and software programs might not be available on your device.
- Depending on the version of operating systems and programs, some user interface instructions might not be applicable to your device.
- Documentation content is subject to change without notice. Thundercomm makes constant improvements on the documentation of your computer, including this guidebook.
- Function declarations, function names, type declarations, attributes, and code samples appear in a different format, for example, `cp armcc armcpp`.
- Code variables appear in angle brackets, for example, `<number>`.
- Button, tool, and key names appear in bold font, for example, click **Save** or press **Enter**.
- Commands to be entered appear in a different font, for example:
```
$ adb devices
```
- On the host computer use $ as shell prompt, for example:
```
$ adb shell
```
- On the target device use # as shell prompt, for example:
```
# logcat
```

Table of Contents

Chapter 1. Product Overview - 2 -
1.1. Features and specifications - 3 -
1.2. Product diagram - 4 -
Chapter 2. Function Test - 5 -
2.1. Debug port - 5 -
2.2. SD card - 5 -
2.3. DSI LCD panel - 6 -
2.4. Display port - 7 -
2.5. Touch panel - 9 -
2.6. Type-A USB connector - 9 -
2.7. LED control - 9 -
2.8. Button event - 10 -
2.9. Wi-Fi antenna slot and Bluetooth - 10 -
2.10. Ethernet - 14 -
2.11. Camera - 17 -
2.12. Video - 21 -
2.13. Audio - 24 -
2.14. TensorFlow lite - 28 -
2.15. SNPE - 28 -
2.16. Face detection - 30 -
2.17. Sleep mode (to be updated) - 30 -
Appendix 1. Notices - 32 -
Appendix 2. Trademarks - 34 -

Chapter 1. Product Overview

The Turbox™ C410/C610 Open Kit is a cost-effective solution to evaluate the performance and build the prototype of the smart cameras and intelligent IoT devices quickly. The Open Kit features rich interfaces, including USB Type A, HDMI in, Ethernet, and camera mezzanine-board expansions enabling the Kit 'open' for building camera-based devices, including panorama, smart camera, and video conference camera. The Open Kit supports Linux(Q2,2020) and Android(Q4,2020) operating system, combining powerful artificial intelligence and machine learning to address increasing demand for smart cities, commercial and enterprise, homes, and vehicles applications.

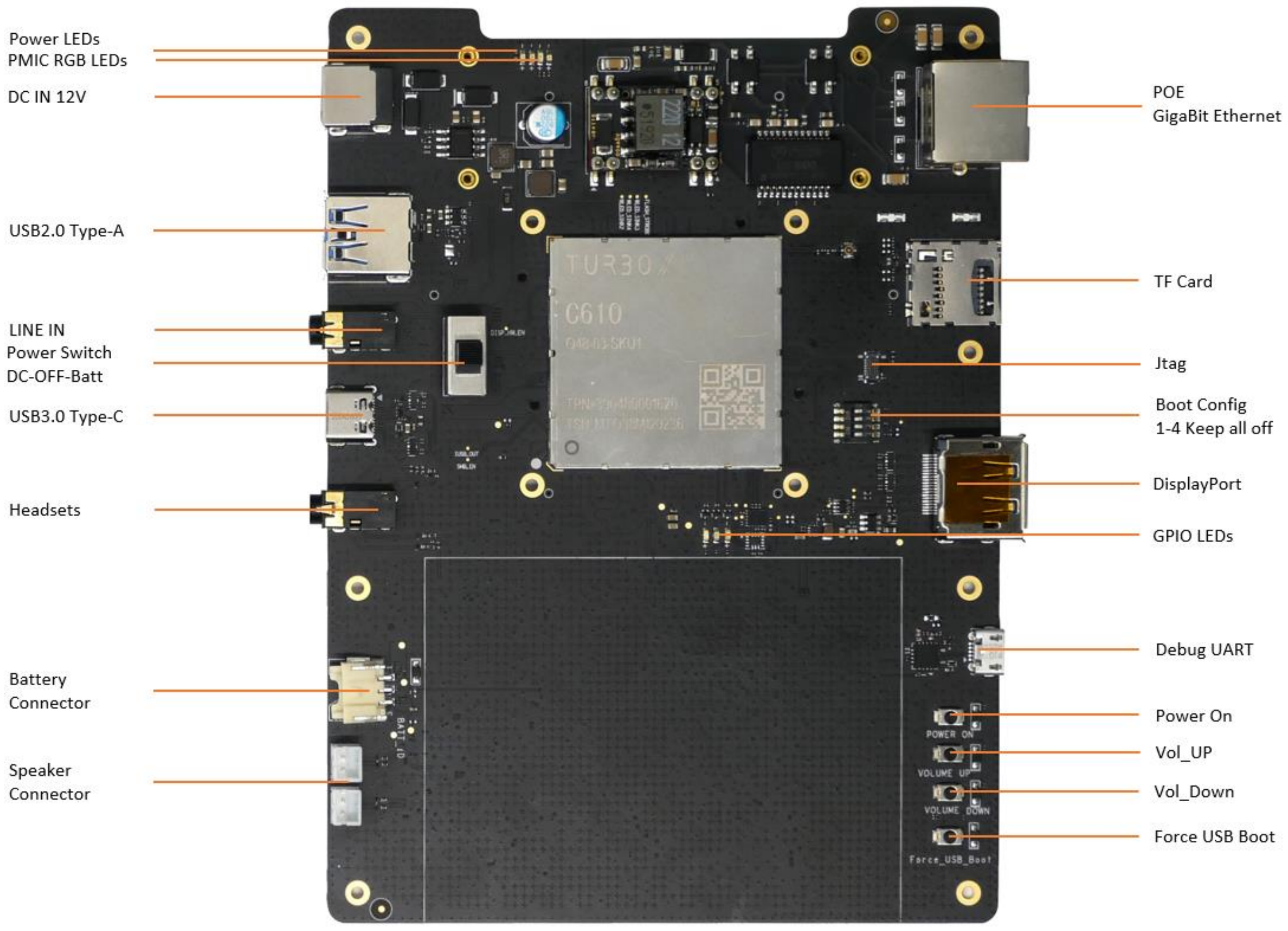


Figure 1-1. Top View

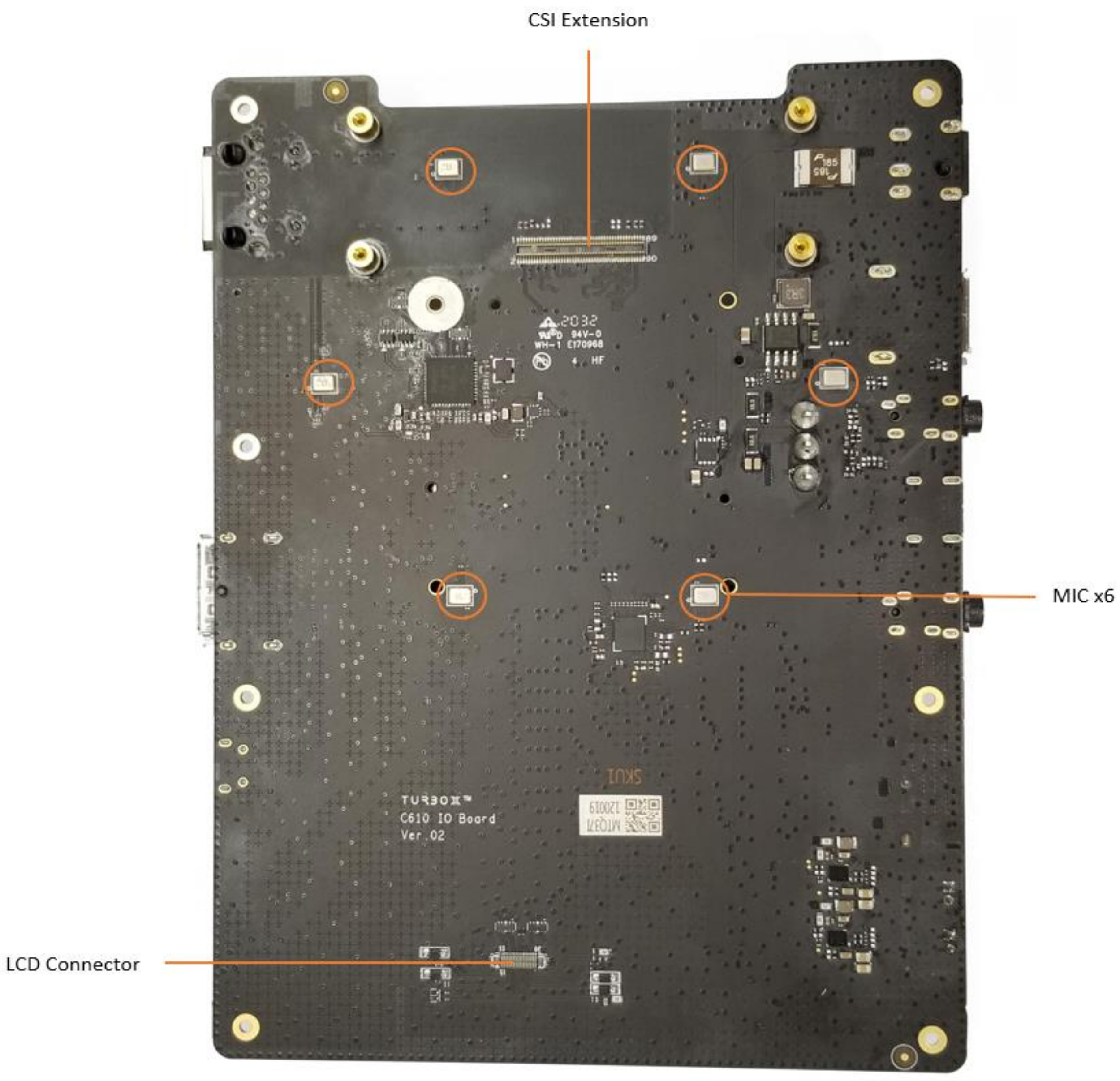


Figure 1-2. Bottom View

NOTE: Camera Mezzanine Board, Camera Module and TFT-LCD are accessories to the Open Kit.

1.1. Features and specifications

Table 1-2: Features and specifications

Category	Description
SOM on board	Turbox™ C410/C610 SOM
Vertical Mezzanine Board (Camera board)	• Vertical A: Panorama - 2 x IMX577 • Vertical B: Smart Security - 1 x IMX347 / 1 x IMX334 (3x Optical Zooming) • Vertical C: Conference: - 1 x IMX415 + 1 x HDMI IN
Display Interfaces	1 x DP; 1 x 5.0“ TFT-LCD;

Category	Description
Audio Interfaces	• 1 x 3.5mm headset • 6 x Digital Mics (From External Codec) • 2 x Speaker out (From Codec)
General Interfaces on Mother Board	• 1 x USB 3.2 Type C • 1 x USB 2.0 Type A • 1 x Micro USB (For Debug) • 1 x SD card; • 1 x GbE Ethernet • 1 x ACCELEROMETER+GYRO Sensor • 1 x Wifi/Bt Antenna Connector • 6 x LEDs • 4 x Keys (Vol Up/Down, Force USB Boot, Power)
Power Supply	DC IN 12V, POE 15W, Battery Connector
Operating Environment	• Operation Temperature: -20℃ ~ 70℃ • Operation Humidity: 5%~95%, non-condensing
Dimension	120 x 250 mm
OS Support	Linux, Android

1.2. Product diagram

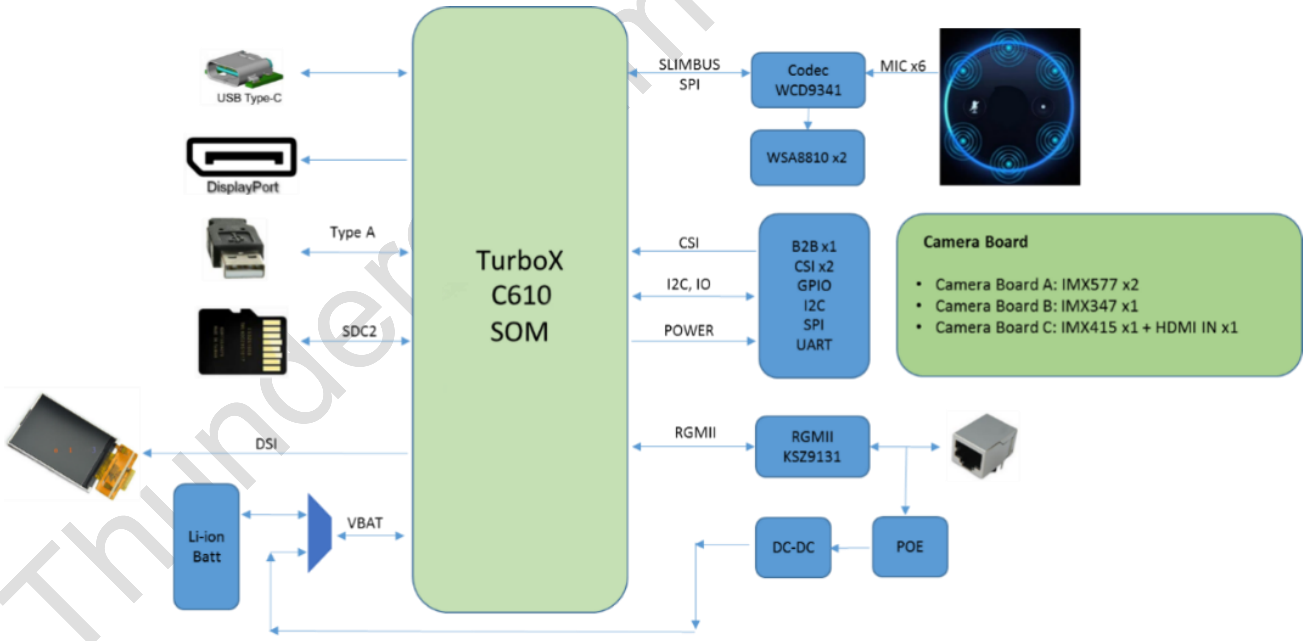


Figure 1-3. Block Diagram

Chapter 2. Function Test

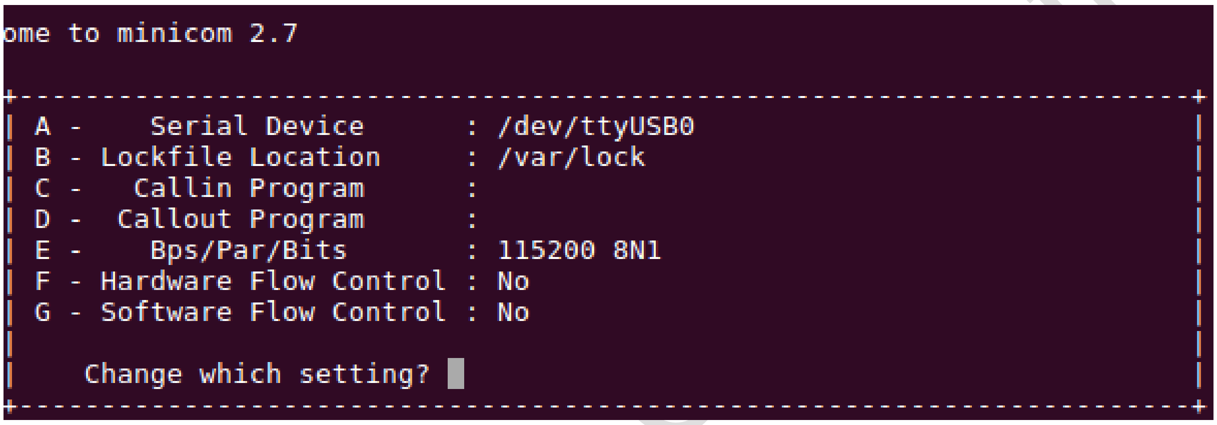
This chapter introduces the detailed testing steps on board function.

2.1. Debug port

Step 1.Connect the C410/C610 board to the computer via Micro USB (the UART debug port).

Step 2.Use a UART tool you prefer, such as **Minicom**.

A new serial port will be added to your computer after the device been connected. Select the new serial port, and configure it to 115200 8N1. Refer to the configuration of **Minicom 2.7** as following:



```
ome to minicom 2.7

+-----------------------------------------------------------------------+
| A -    Serial Device      : /dev/ttyUSB0                              |
| B - Lockfile Location     : /var/lock                                 |
| C -   Callin Program      :                                           |
| D -  Callout Program      :                                           |
| E -    Bps/Par/Bits       : 115200 8N1                                |
| F - Hardware Flow Control : No                                        |
| G - Software Flow Control : No                                        |
|                                                                       |
|    Change which setting?                                              |
+-----------------------------------------------------------------------+
```

Figure 2-1. Minicon 2.7 Configuration Screen Capture

Step 3.Log in the board system with the following account credentials.

```
… …
[  OK  ] Started start dsp variants.
[  OK  ] Reached target Multi-User System.
         Starting Update UTMP about System Runlevel Changes...
[  OK  ] Started Update UTMP about System Runlevel Changes.

qsap 202001010057 qcs610-odk ttyMSM0
qcs610-odk login: root
Password: oelinux123
```

➲ **NOTE:** UART debug cannot be used in perf/user version.

2.2. SD card

SD card is hot-swappable. When inserted to the board, SD card shall automatically mount to */sdcard/* to access contents.

Run the following commands to test read and write speed of the SD card:

```
$ adb root && adb shell
# cd sdcard
# dd if=/dev/urandom of=test.txt bs=30M count=2 conv=fsync
dd if=/dev/urandom of=test.txt bs=30M count=2 conv=fsync
2+0 records in
2+0 records out
62914560 bytes (60.0MB) copied, 6.366916 seconds, 9.4MB/s
# echo 3 >/proc/sys/vm/drop_caches
# dd if=/sdcard/test.txt of=/dev/null
```

```
dd if=/sdcard/test.txt of=/dev/null
122880+0 records in
122880+0 records out
62914560 bytes (60.0MB) copied, 0.092965 seconds, 645.4MB/s
```

2.3. DSI LCD panel

Step 1.Connect the LCD panel to the board. Refer to Figure 2-2.

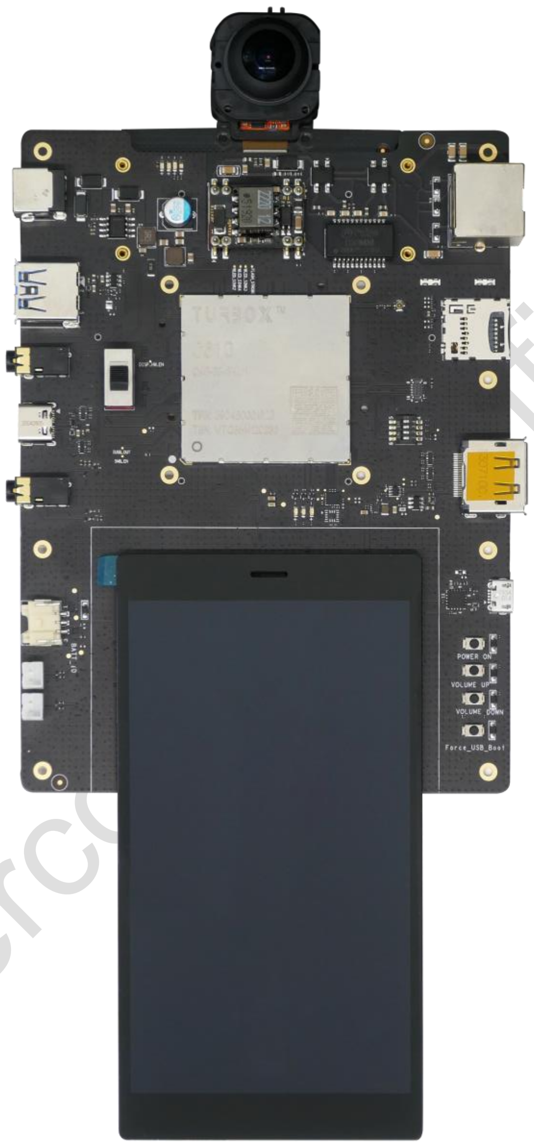


Figure 2-2. Camera Connection

Step 2.Testing LCD panel with **Weston**. The LCD panel is successfully connected with a lot of flowers displaying on the screen.

1. Disable */dev/dm-0 write-protected*.

```
$ adb root
$ adb disable-verity
$ adb reboot
```

2. Input the following command to output many flowers covered the LCD.

```
# adb root && adb remount && adb shell mount -o remount,rw /
# adb shell
# export XDG_RUNTIME_DIR=/dev/socket/weston
# /usr/bin//weston --tty=1 --idle-time=0 &
```

3. Camera stream preview on the LCD.

```
# export XDG_RUNTIME_DIR=/dev/socket/weston
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf ! "video/x-
raw(memory:GBM),format=NV12,camera=0,width=1920,height=1080,framerate=30/1" !
waylandsink sync=false fullscreen=true
```

4. Exit **Weston**.

```
# killall weston
```

Step 3.Testing LCD panel with **Weston**.

Please refer to: Video file playback.

2.4. Display port

If there is no code compilation environment, please refer to Multiplex Decoding for more information on display port settings.

You can also follow the steps below to compile the **LCD** tool and test display port.

Step 1.Set up the **modetest** on the Yocto project.

1. Export machine distro.

```
$ cd <project_root>/apps_proc/
$ MACHINE=qcs610-odk DISTRO=qti-distro-fullstack-virtualization-debug
VARIANT=debug
$ source poky/qti-conf/set_bb_env.sh
```

2. Modify `local.conf`.

```
$ vim conf/local.conf
  - INHERIT += "rm_work"
  + #INHERIT += "rm_work"
```

3. Build `libdrm`.

```
bitbake -fc cleanall libdrm ; bitbake libdrm
```

4. Set up the **modetest** on the device

```
$ cd  tmp-glibc/work/armv7ahf-neon-oe-linux-gnueabi/libdrm/2.4.83-r0/package/
$ tar -zcvf modetest.tar usr/
$ adb root && adb remount && adb shell mount -o remount,rw /
$ adb push modetest /usr/bin/
$ adb shell
# chmod 777 /usr/bin/modetest
```

Step 2.Connect the LCD panel to the board.

Step 3.Test external display monitor with **modetest**.

- Require the panel information.

```
# modetest -M msm_drm | more
Encoders:
id      crtc    type    possible crtcs  possible clones
27      0       DSI     0x00000007      0x00000000
43      0       Virtual 0x00000007      0x00000000
52      0       TMDS    0x00000007      0x00000000
57      0       DPMST   0x00000007      0x00000000
58      0       DPMST   0x00000007      0x00000000
```

```
....

53      0     connected DP-1           510x290         12    52
  modes:
  name refresh (Hz) hdisp hss hse htot vdisp vss vse vtot)
  1920x1080 60 1920 2008 2052 2200 1080 1084 1089 1125 flags: phsync, pvsync;
type: preferred, driver
  1600x900 60 1600 1624 1704 1800 900 901 904 1000 flags: phsync, pvsync; type:
driver
  1280x1024 75 1280 1296 1440 1688 1024 1025 1028 1066 flags: phsync, pvsync;
type: driver
  1280x1024 60 1280 1328 1440 1688 1024 1025 1028 1066 flags: phsync, pvsync;
type: driver
  1152x864 75 1152 1216 1344 1600 864 865 868 900 flags: phsync, pvsync; type:
driver
  1024x768 75 1024 1040 1136 1312 768 769 772 800 flags: phsync, pvsync; type:
driver
  1024x768 60 1024 1048 1184 1344 768 771 777 806 flags: nhsync, nvsync; type:
driver
  800x600 75 800 816 896 1056 600 601 604 625 flags: phsync, pvsync; type: driver
  800x600 60 800 840 968 1056 600 601 605 628 flags: phsync, pvsync; type: driver
  640x480 75 640 656 720 840 480 481 484 500 flags: nhsync, nvsync; type: driver
  640x480 60 640 656 752 800 480 490 492 525 flags: nhsync, nvsync; type: driver
  720x400 70 720 738 846 900 400 412 414 449 flags: nhsync, pvsync; type: driver
  props:
  1 EDID:
        flags: immutable blob
        blobs:

        value:
              00ffffffffffff0010ac6bf053414543
              1f190104a5331d783add45a3554fa027
              125054a54b00714f8180a9c0d1c00101
              010101010101023a801871382d40582c
              4500fd1e1100001e000000ff004d5231
              594b353752434541530a000000fc0044
              454c4c204532333136480a20000000fd
              00384c1e5311000a2020202020200075
  2 DPMS:
        flags: enum
        enums: On=0 Standby=1 Suspend=2 Off=3
        value: 0
  5 link-status:
        flags: enum
        enums: Good=0 Bad=1
        value: 0
  18 CRTC_ID:
        flags: object
        value: 0
  29 capabilities:
        flags: immutable blob
        blobs:

        value:
              646973706c617920747970653d756e6b
              6e6f776e0a00
....
```

- Test display function with **modetest**.

```
# modetest -M msm_drm -s 53:1920x1080
setting mode 1920x1080-60Hz@XR24 on connectors 53, crtc 107
```

2.5. Touch panel

It is not supported yet.

2.6. Type-A USB connector

Step 1.Connect a mouse to the open kit via Type-A USB interface and check the information of the input device.

```
$ adb shell
# dmesg  -c
[  469.069224] msm-dwc3 a800000.hsusb: DWC3 exited from low power mode
[  469.330682] usb 1-1: new low-speed USB device number 2 using xhci-hcd
[  469.497240] usb 1-1: New USB device found, idVendor=04ca, idProduct=0061
[  469.504821] usb 1-1: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[  469.522175] usb 1-1: Product: USB Optical Mouse
[  469.527070] usb 1-1: Manufacturer: PixArt
[  469.579764] input: PixArt USB Optical Mouse as
/devices/platform/soc/a800000.hsusb/a800000.dwc3/xhci-hcd.0.auto/usb1/1-1/1-
1:1.0/0003:04CA:0061.0001/input/input2
[  469.650707] hid-generic 0003:04CA:0061.0001: input: USB HID v1.11 Mouse [PixArt
USB Optical Mouse] on usb-xhci-hcd.0.auto-1/input0

/*According to the log ,it's  create a input device at '/dev/input/event2'*/
# cat /dev/input/event2 | hexdump
0000000 8b0c 5f1f 4a16 000c 0002 0000 ffff ffff
0000010 8b0c 5f1f 4a16 000c 0000 0000 0000 0000
0000020 8b0c 5f1f 68fc 000c 0002 0000 fff9 ffff
0000030 8b0c 5f1f 68fc 000c 0000 0000 0000 0000
0000040 8b0c 5f1f 883d 000c 0002 0000 fff5 ffff
0000050 8b0c 5f1f 883d 000c 0002 0001 0002 0000
0000060 8b0c 5f1f 883d 000c 0000 0000 0000 0000
0000070 8b0c 5f1f a781 000c 0002 0000 fffd ffff
f1f a781 000c 0002 0001 0001 0000
```

Step 2.Connect a USB storage device and check the device information.

```
# dmesg | grep usb
[  597.589443] msm-dwc3 a800000.hsusb: DWC3 exited from low power mode
[  598.920681] usb 1-1: new high-speed USB device number 3 using xhci-hcd
[  599.078146] usb 1-1: New USB device found, idVendor=0781, idProduct=5588
[  599.086489] usb 1-1: New USB device strings: Mfr=2, Product=3, SerialNumber=1
[  599.094176] usb 1-1: Product: USB Extreme Pro
[  599.098700] usb 1-1: Manufacturer: SanDisk
[  599.103199] usb 1-1: SerialNumber: 12345778C5FD
[  599.129923] usb-storage 1-1:1.0: USB Mass Storage device detected
[  599.148015] scsi host0: usb-storage 1-1:1.0
[  600.168565] scsi 0:0:0:0: Direct-Access     SanDisk  Extreme Pro      0    PQ: 0
ANSI: 6
[  600.202613] sd 0:0:0:0: [sda] Write Protect is off
[  600.207624] sd 0:0:0:0: [sda] Mode Sense: 43 00 00 00
[  600.276102]  sda: sda1
```

2.7. LED control

The LED is controlled by the following command.

```
$ adb shell
# echo 0 > /sys/devices/platform/soc/soc:leds-gpios/leds/led_bt/brightness     //Turn off
# echo 1 > /sys/devices/platform/soc/soc:leds-gpios/leds/led_bt/brightness     //Turn on
# echo 0 > /sys/devices/platform/soc/soc:leds-gpios/leds/led_wifi/brightness   //Turn off
# echo 1 > /sys/devices/platform/soc/soc:leds-gpios/leds/led_wifi/brightness   //Turn on
# echo 0 > /sys/devices/platform/soc/soc:leds-gpios/leds/led_59/brightness     //Turn off
# echo 1 > /sys/devices/platform/soc/soc:leds-gpios/leds/led_59/brightness     //Turn on
# echo 255 > /sys/class/leds/red/brightness //Turn on
# echo 0 > /sys/class/leds/red/brightness //Turn off
# echo 255 > /sys/class/leds/green/brightness //Turn on
# echo 0 > /sys/class/leds/green/brightness //Turn off
# echo 255 > /sys/class/leds/blue/brightness //Turn on
# echo 0 > /sys/class/leds/blue/brightness //Turn off
```

2.8. Button event

Refer to the following figure to locate the buttons.

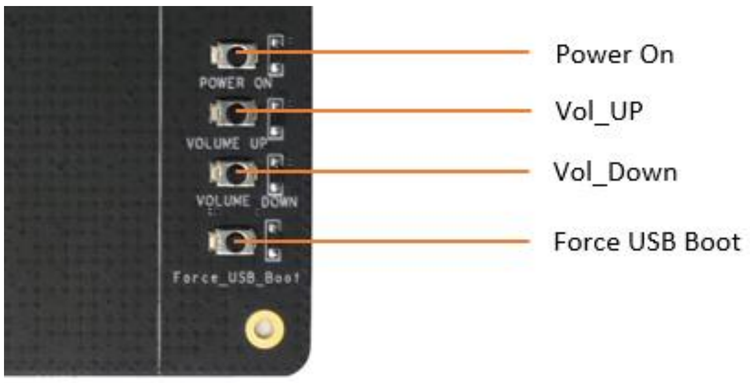


Figure 2-3. Button Location

Key	Device node
VOL UP	/dev/input/event1
VOL DOWN	/dev/input/event0
POWER ON	/dev/input/event0
USBBOOT	-

NOTE: **Force USB BOOT** is used to force the system enter emergency download mode.

To Get interrupt event, run the following commands:

```
$ adb root
$ adb shell
# cat dev/input/event0 | hexdump
# cat dev/input/event1 | hexdump
```

2.9. Wi-Fi antenna slot and Bluetooth

Step 1.Set up Wi-Fi STA mode

1. Connect the antenna to the board.
2. Run the following command to open `mobileap_cfg.xml`.

```
$ adb root && adb shell
# vi /etc/data/mobileap_cfg.xml
```

3. Input i to enter edit mode and set `<WLANEnableAtBootup>` value to `1` in `mobileap_cfg.xml` to automatically enable Wi-Fi connection function at system bootup.

```
<WLANEnableAtBootup>1</WLANEnableAtBootup>
```

4. Press **ESC** key to exit edit mode.
5. Input `:wq!` to save and exit the modified configuration file.
6. Run the following command to open `wlan_cfg.xml`.

```
# vi /etc/data/wlan_cfg.xml
```

7. Input `i` to enter edit mode and modify `<WLANMode>` value to `STA` in `wlan_cfg.xml` to enable STA mode.

```
<WLANMode>STA</WLANMode>
```

8. Press **ESC** key to exit edit mode.
9. Input `:wq!` to save and exit the modified configuration file.
10. Execute the following commands to open `wpa_supplicant.conf` and verify the connection while the device is in **Station Mode**.

```
# vi /etc/misc/wifi/wpa_supplicant.conf
```

11. Input `i` to enter edit mode and add information on ssid and psk of Wi-Fi as following.

```
# Only WPA-PSK is used. Any valid cipher combination is
ctrl_interface=/var/run/wpa_supplicant
p2p_disabled=1
network={
#Open
#       ssid="example open network"
#       key_mgmt=NONE
#WPA-PSK
  ssid="zcm"                     //Input your AP's name
# proto=WPA
# key_mgmt=WPA-PSK
# pairwise=TKIP CCMP
# group=TKIP CCMP
  psk="11111111"                //Input your AP's password
#WEP
# ssid="example wep network"
# key_mgmt=NONE
# wep_key0="abcde"
# wep_key1=0102030405
# wep_tx_keyidx=0
}
```

12. Execute the following commands to enable **WiFi Station Mode** and connect to AP network.

```
# wpa_supplicant -Dnl80211 -iwlan0 -c /etc/misc/wifi/wpa_supplicant.conf -ddddt &
# dhcpcd wlan0
```

Run the following command to check network connection. The device is successfully connected to the network with log of IP address permission displayed on your screen.

```
# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr 00:0A:F5:83:66:EF
         inet addr:192.168.43.92  Bcast:192.168.43.255  Mask:255.255.255.0
         inet6 addr: fe80::20a:f5ff:fe83:66ef%1736140884/64 Scope:Link
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:15 errors:0 dropped:0 overruns:0 frame:0
         TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:3000
         RX bytes:1864 (1.8 KiB)  TX bytes:1828 (1.7 KiB)
```

13. Perform a pin test.

```
# ping www.baidu.com
```

```
PING www.baidu.com (23.59.93.51): 56 data bytes
64 bytes from 23.59.93.51: seq=0 ttl=48 time=171.722 ms
64 bytes from 23.59.93.51: seq=1 ttl=48 time=163.070 ms
64 bytes from 23.59.93.51: seq=2 ttl=48 time=244.932 ms
64 ytes from 23.59.93.51: seq=3 ttl=48 time=166.135 ms
```

NOTE: If the device is failed to perform a ping test, check the **Firewall** settings, and perform ping test outside the **Firewall**.

Step 2.Set up Wi-Fi AP mode

1. Connect the antenna to the board.
2. Run the following command to check the configuration.

```
$ adb shell
# vi /etc/data/mobileap_cfg.xml
                     <Profile>1</Profile>
                     <AutoConnect>1</AutoConnect>
                     <Roaming>0</Roaming>
...
                    <MobileAPEnableAtBootup>1</MobileAPEnableAtBootup>
```

NOTE: Ensure that the `<AutoConnect>` values `1` and `<MobileAPEnableAtBootup>` values 1, or you have to modify the value to `1` in `mobileap_cfg.xml`.

3. Run the following command to open `mobileap_cfg.xml`.

```
# vi /etc/data/mobileap_cfg.xml
```

4. Input i to enter edit mode and modify the `<WLANEnableAtBootup>` value to `1` in `mobileap_cfg.xml` to automatically enable Wi-Fi connection function at system bootup.

```
<WLANEnableAtBootup>1</WLANEnableAtBootup>
```

5. Press **ESC** key to exit edit mode.
6. Input `:wq!` to save and exit the modified configuration file.
7. Run the following command to open `wlan_cfg.xml`.

```
# vi /etc/data/wlan_cfg.xml
```

8. Input `i` to enter edit mode and modify `<WLANMode>` value to `AP` in `wlan_cfg.xml` to enable AP mode.

```
<WLANMode>AP</WLANMode>
```

9. Press **ESC** key to exit edit mode.
10. Input `:wq!` to save and exit the modified configuration file.
11. Reset the device and execute the following commands to check hostapd.config:

```
$ adb root && adb shell
# ps | grep hostapd
2345      /etc/misc/wifi/hostapd.conf
```

12. Connect your phone to Wi-Fi.

    Default Wi-Fi Name: AndroidAP-sdmsteppe-WPA2

    Default Password: 123456789.

Step 3.Before running **btapp**, run the btproperty in the background (run only once at the beginning).

```
$adb root && adb shell
# sed -i "s/BtA2dpSinkSplitEnable=false/BtA2dpSinkSplitEnable=true/g"
/etc/bluetooth/bt_app.conf
# btproperty &
# btapp
```

Step 4.After running **btapp**, input **gap_menu** and press **Enter**.

```
gap_menu
***************** Menu *******************
        enable
        disable
        inquiry
        cancel_inquiry
        pair<space><bt_address>      eg. pair 00:11:22:33:44:55
        unpair<space><bt_address>    eg. unpair 00:11:22:33:44:55
        inquiry_list
        bonded_list
        get_state
        get_bt_name
        get_bt_address
        set_bt_name<space><bt name>    eg. set_bt_name MDM_Fluoride
        set_le_bt_name<space><bt name>  eg. set_le_bt_name MDM_LE_Fluoride
        main_menu
       ******************************************
```

Step 5.Input **enable** and press **Enter** to enable Bluetooth.

```
enable
killall: wcnssfilter: no process killed
killall: btsnoop: no process killed
killall: qcbtdaemon: no process killed
/bin/sh: qcbtdaemon: not found
...
Load Audio HAL competed
BT State is ON
```

Step 6.Input **inquiry** and press **Enter** to start inquiry.

```
inquiry
Inquiry Started
Device Found details:
Found device Addr: 28:11:a5:01:00:a2
Found device Name: LE-Bose SoundSport
Device class is: 7936
Device Found details:
Found device Addr: e4:ba:d9:10:00:c9
Found device Name: 360FLY4K_00C8
Device class is: 7936
Device Found details:
Found device Addr: 28:11:a5:24:01:05
Found device Name: LE-reserved_N
Device class is: 7936
Inquiry Stopped automatically
```

NOTE: To cancel inquiry, issue the following command while the inquiry in progress: `cancel_inquiry`.

Step 7.input **inquiry_list** and press **Enter** to check the inquiry list.

```
inquiry_list
**************************** Inquiry List*******************************
LE-Bose SoundSport                            28:11:a5:01:00:a2
360FLY4K_00C8                                e4:ba:d9:10:00:c9
LE-reserved_N                                28:11:a5:24:01:05
**************************** End of List *******************************
```

Step 8.Run the following command to pair outgoing SSP.

```
pair<mac address>0/1/2
```

To accept or reject the outgoing pairing for the following example (pair e4:ba:d9:10:00:c9), type Yes or No and press Enter.

```
pair e4:ba:d9:10:00:c9
*************************************************
 BT pairing request::Device iPhone::Pairing Code:: 281155
*************************************************
 ** Please enter yes / no **
yes
*************************************************
 Pairing state for 360FLY4K_00C8 is BONDED
*************************************************
```

Step 9.Check the bonded list.

1. Input **bonded_list** and press **Enter** to check the bonded list:

```
bonded_list
************************** Bonded Device List *************************
360FLY4K_00C8                                a4:f1:e8:c6:2f:b4
**************************** End of List *****************************
```

2. To disconnect bonded device, type **disable** and press **Enter**.

```
disable
killall: qcbtdaemon: no process killed
 BtHfpAgMsgHandler event = 1029
killall: wcnssfilter: no process killed
 BT State is OFF
```

3. To exit from **btapp**, navigate to the main menu and enter the following command:

```
exit
```

2.10. Ethernet

Refer to the figure below to connect the antenna.

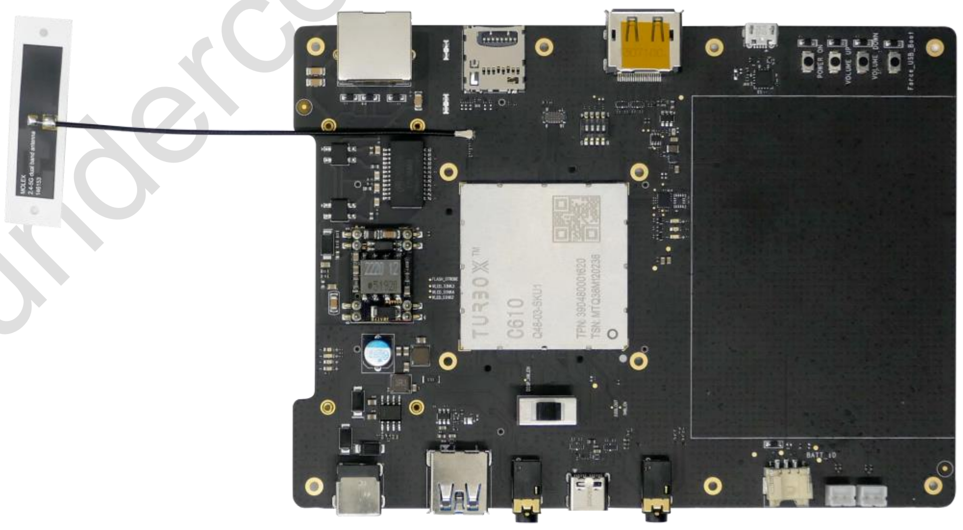


Figure 2-4. Antenna Connection

2.10.1. WAN Mode

Step 1.eth0 interface will be automatically created after device normal bootup.

```
$ adb root && adb shell
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:55:7B:B5:7D:F7
         UP BROADCAST MULTICAST  MTU:1500  Metric:1
         RX packets:121 errors:0 dropped:0 overruns:0 frame:0
         TX packets:106 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:12422 (12.1 KiB)  TX bytes:7990 (7.8 KiB)
         Interrupt:86
```

Step 2.Set up the eth0 interface to function under WAN mode.

```
# vi /etc/data/mobileap_cfg.xml

<EthBackhaul>
       <EthBackhaulMode>1</EthBackhaulMode>
</EthBackhaul>
```

Step 3.Reset device and Ethernet port to connect to network.

```
$ adb root && adb shell
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:55:7B:B5:7D:F7
         inet addr:192.168.1.101  Bcast:192.168.1.255  Mask:255.255.255.0
         inet6 addr: fe80::255:7bff:feb5:7df7/64 Scope:Link
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:124 errors:0 dropped:0 overruns:0 frame:0
         TX packets:124 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:13662 (13.3 KiB)  TX bytes:9924 (9.6 KiB)
         Interrupt:86
```

Step 4.Perform a ping test.

```
# su && ping www.baidu.com
PING www.baidu.com (220.181.38.149): 56 data bytes
64 bytes from 220.181.38.149: seq=0 ttl=48 time=8.708 ms
64 bytes from 220.181.38.149: seq=1 ttl=48 time=9.975 ms
64 bytes from 220.181.38.149: seq=2 ttl=48 time=9.404 ms
64 bytes from 220.181.38.149: seq=3 ttl=48 time=8.329 ms
^C
--- www.baidu.com ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 8.329/9.104/9.975 ms
```

NOTE: If the device is failed to perform a ping test, check the **Firewall** settings and perform ping test outside the **Firewall**.

2.10.2. LAN Mode

Step 1.eth0 interface will be automatically created after device normal bootup.

```
$ adb root && adb shell
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:55:7B:B5:7D:F7
         UP BROADCAST MULTICAST  MTU:1500  Metric:1
         RX packets:121 errors:0 dropped:0 overruns:0 frame:0
         TX packets:106 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:12422 (12.1 KiB)  TX bytes:7990 (7.8 KiB)
         Interrupt:86
```

Step 2.Set up the eth0 interface to function under LAN mode:

```
# vi /etc/data/mobileap_cfg.xml

<EthBackhaul>
       <EthBackhaulMode>0</EthBackhaulMode>
</EthBackhaul>
```

Step 3.Reset device and connect the device to PC via Ethernet port, and the device will get an IP address automatically.

```
$ adb root && adb shell
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:55:7B:B5:7D:F7
         inet addr:169.254.4.1  Bcast:169.254.4.255  Mask:255.255.255.0
         inet6 addr: fe80::255:7bff:feb5:7df7/64 Scope:Link
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:158 errors:0 dropped:0 overruns:0 frame:0
         TX packets:32 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:19196 (18.7 KiB)  TX bytes:2844 (2.7 KiB)
         Interrupt:86
```

Step 4.Perform ping test between the device and PC, or between PC and the device.

NOTE: If the device is failed to perform a ping test, check the **Firewall** settings and perform ping test outside the **Firewall**.

2.10.3. Ethernet MAC address

Step 1.Create a file to store the MAC address.

```
$ adb root && adb shell
# mkdir -p /persist/factory/ethernet/
# touch /persist/factory/ethernet/emac_config.ini
```

Step 2.Add the MAC address to emac_config.ini.

```
11:22:33:44:55:66
```

Step 3.Verify that the MAC address was written successfully

```
$ adb reboot
$ adb root && adb shell
# Ifconfig eth0
         Link encap:Ethernet  HWaddr 11:22:33:44:55:66  Driver qcom-emac-dwc-eqos
         UP BROADCAST MULTICAST  MTU:1500  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:0 TX bytes:0
         Interrupt:93
```

2.11. Camera

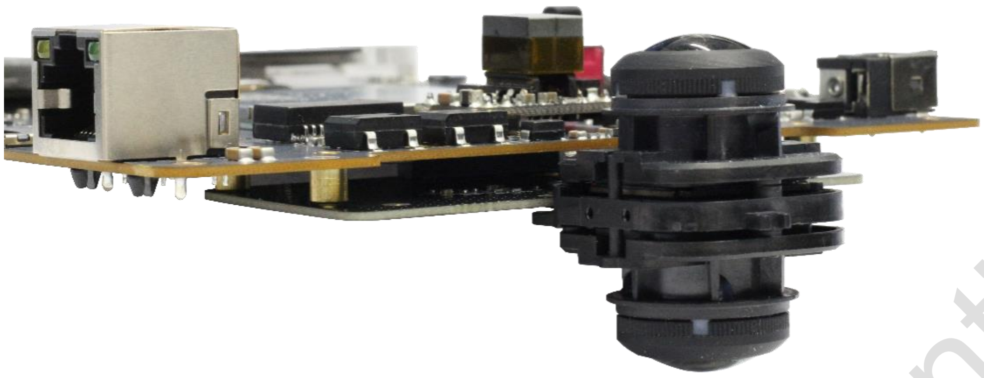


Figure 2-5. Camera Module Installation

This section introduces the **GStreamer** commands for setting up multimedia functions supported on LEPDK. The **gst-launch** is a command used for building and running a GStreamer pipeline. The pipeline to be built and run is specified as a collection of elements and the properties separated by a '!' mark.

For more information on the **gst-launch**, refer to: https://gstreamer.freedesktop.org/documentation/tools/gst-launch.html?gi-language=c.

2.11.1. Environment configuration

Follow these steps to configure environment. The following steps can only be performed for one time after device flash.

Step 1.Disable */dev/dm-0 write-protected*.

```
$ adb root
$ adb disable-verity
$ adb reboot
```

Step 2.Set up the camera module.

```
$ adb root
$ adb shell mkdir -p /etc/camera
$ adb shell touch /etc/camera/camxoverridesettings.txt
$ adb shell "echo IFEDualClockThreshold=600000000 >>
/etc/camera/camxoverridesettings.txt"
$ adb shell "echo maxHalRequests=8 >> /etc/camera/camxoverridesettings.txt"
```

Step 3.Set the display log level.

```
$ adb shell mkdir -p /data/misc/display
$ adb shell "echo 0 > /data/misc/display/gbm_dbg_cfg.txt"
$ adb shell sync
```

Step 4.Reboot the device.

```
$ adb reboot
```

Step 5.Configure the **GStreamer** environment.

```
$ adb shell
$ source /etc/gstreamer1.0/set_gst_env.sh
```

2.11.2. Push video stream to PC via adb tool

Step 1.Start to run **gst-launch-1.0** on device.

```
$ adb root && adb remount && adb shell mount -o remount,rw /
$ adb forward tcp:8900 tcp:8900
$ adb shell
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf ! video/x-
h264,format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse config-
interval=1 ! mpegtsmux name=muxer ! queue ! tcpserversink port=8900 host=127.0.0.1
```

NOTE: After a few seconds, the following log will display on your screen, indicating that the camera is in operation.

```
Setting pipeline to PAUSED …
Pipeline is live and does not need PREROLL …
Setting pipeline to PLAYING …
New clock:GstSystemClock
```

Step 2.Start to run **VLC-Media-Player** on PC (of Windows/Ubuntu system).

For Windows:

- Open **VLC** media play and turn off **Windows Firewall**.
- Go to **Media** → **Open Network Stream**.
- Enter *tcp://127.0.0.1:8900* to get network URL.

For Ubuntu:

```
$ vlc -vvv tcp://127.0.0.1:8900
```

2.11.3. Push video stream to PC via network

Make sure the device (board) and host (PC) are in the same network segment! For more information, refer to Ethernet.

Step 1.Run a RTSP server and **gst-launch-1.0** on device.

```
$ adb root && adb remount && adb shell mount -o remount,rw /
$ adb shell
# gst-rtsp-server -a IP -p 8900 -m /live "( udpsrc name=pay0 port=8554
caps=\"application/x-rtp,media=video,clock-rate=90000,encoding-
name=H264,payload=96\" )" &
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf ! video/x-
h264,format=NV12,width=3840,height=2160,framerate=30/1 ! h264parse config-interval=-
1 ! rtph264pay pt=96 ! udpsink host=IP port=8554
```

NOTE: Please replace `IP` in the above command lines with the IP address of the target board!

Step 2.Start to run **VLC-Media-Player** on PC (of Windows/Ubuntu system).

For Windows:

- Open **VLC** media play and turn off **Windows Firewall**.
- Go to **Media** → **Open Network Stream**.
- Enter *rtsp://**IP**:8900/live* to get network URL.

For Ubuntu:

```
$ vlc -vvv rtsp://IP:8900/live
```

NOTE:

Please replace `IP` in the above command lines with the IP address of the target board!

VLC media player configuration: **Tools** -> **Preferences** -> **Input/Codecs** -> **Live555 stream transport** -> **RTP over RTSP(TCP)**

2.11.4. Live snapshot

Snapshot can be taken along with video capture. The following command is used to take a 4K snapshot while 1080p streaming is running.

```
# gst-pipeline-app -e qtiqmmfsrc name=qmmf ! video/x-
h264,format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse !mp4mux ! queue !
filesink location="/data/mux2.mp4" qmmf. !
"image/jpeg,width=3840,height=2160,framerate=30/1" ! multifilesink
location=/data/frame%d.jpg sync=true async=false
```

After running the preceding command, a menu is printed out. Select the **capture-image** option by entering **qmmf** to capture images. The captured images and the MP4 record file are saved at: */data/*.

2.11.5. Dual camera concurrency recorder 1080p

Use the **camera=** parameter to specify the camera to be used. In addition, dual 4K coding is not supported.

Step 1.Open a new terminal for rear camera.

```
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf camera=1 qmmf.video_0 ! video/x-
h264,format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse ! mp4mux !
queue ! filesink location="/data/cam0_avc.mp4"
```

Step 2.Open another terminal for front camera.

```
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf camera=2 qmmf.video_0 ! video/x-
h264,format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse ! mp4mux !
queue ! filesink location="/data/cam1_avc.mp4"
```

2.11.6. Digital zoom

The **QTI gst-pipeline-app** is a helper tool that exposes the same capability as **GStreamer's gst-launch-1.0** tool. It also provides an option to set runtime properties for GST elements.

Refer to Environment configuration and Push video stream to PC via adb tool to push streaming to the VLC of PC end via adb.

Step 1.Run the following command:

```
# gst-pipeline-app -e qtiqmmfsrc name=qmmf qmmf.video_0 ! video/x-h264,
format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse config-interval=1 !
mpegtsmux name=muxer ! queue ! tcpserversink port=8900 host=127.0.0.1
```

Step 2.Input **qmmf** (the name of the plugin to be controlled), select number **18**, and finally enter value **<960,540,960,540>**.

Take sensor mode 1920x1080 as an example,

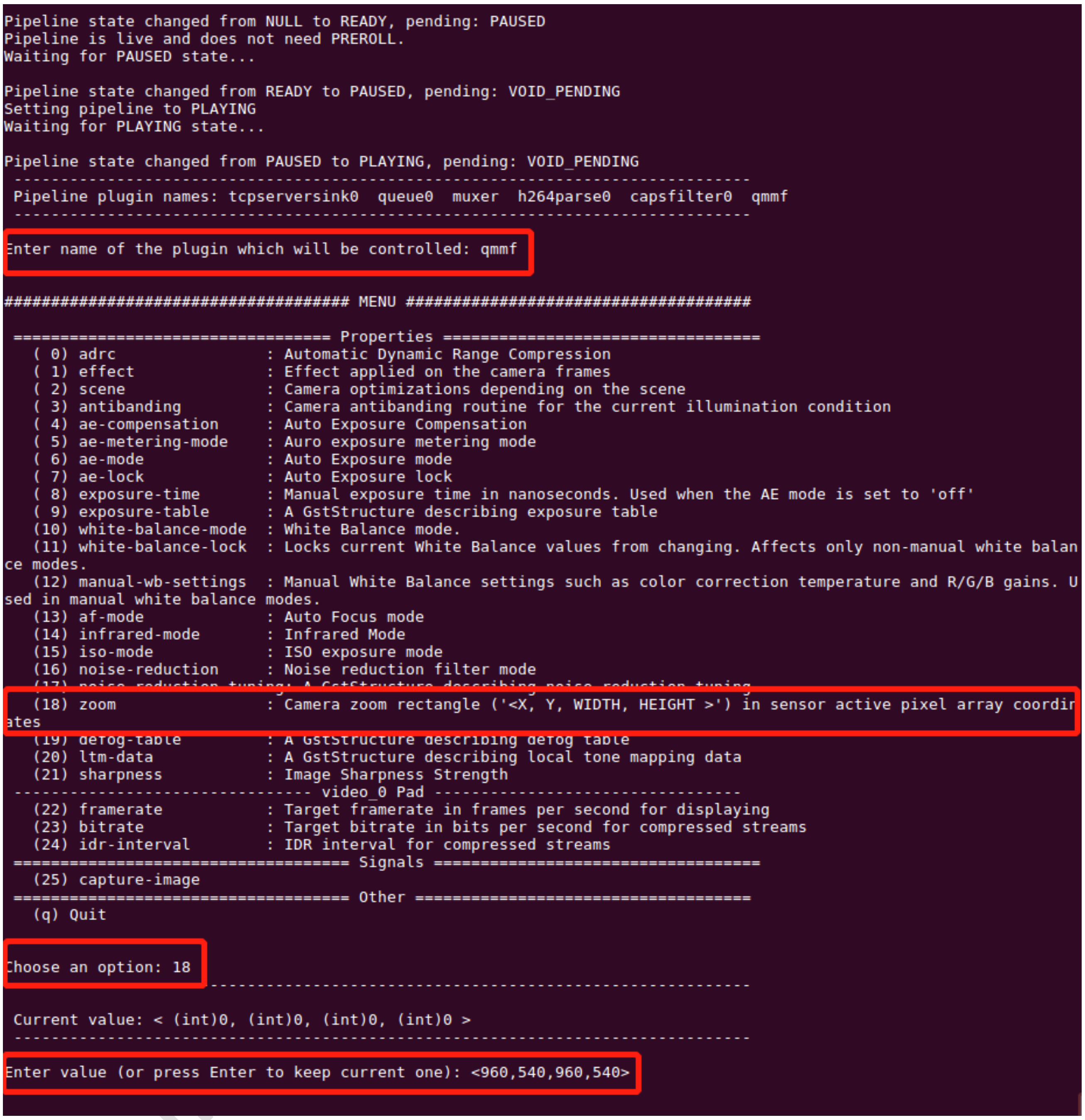
Zoom in:

- For 1x zoom, input value should be <0,0,1920,1080>
- For 2x zoom, sensor mode width and height should be dvided by 2. So the input value would be <0,0,960,540>.

Zoom out:

For 2x zoom, specify the location(x,y) as the first 2 arguments. So the input would be <480,270,960,540>. This (x,y) value is obtained by dividing the final resolution by 2, since this is a 2x zoom, the final resolution would be 960x540 and center of the frame would be (480,270)

The following options are displayed:



```
Pipeline state changed from NULL to READY, pending: PAUSED
Pipeline is live and does not need PREROLL.
Waiting for PAUSED state...

Pipeline state changed from READY to PAUSED, pending: VOID_PENDING
Setting pipeline to PLAYING
Waiting for PLAYING state...

Pipeline state changed from PAUSED to PLAYING, pending: VOID_PENDING
 --------------------------------------------------------------------------------
 Pipeline plugin names: tcpserversink0  queue0  muxer  h264parse0  capsfilter0  qmmf
 --------------------------------------------------------------------------------

Enter name of the plugin which will be controlled: qmmf

##################################### MENU #####################################

 ================================== Properties ==================================
   ( 0) adrc                    : Automatic Dynamic Range Compression
   ( 1) effect                  : Effect applied on the camera frames
   ( 2) scene                   : Camera optimizations depending on the scene
   ( 3) antibanding             : Camera antibanding routine for the current illumination condition
   ( 4) ae-compensation         : Auto Exposure Compensation
   ( 5) ae-metering-mode        : Auro exposure metering mode
   ( 6) ae-mode                 : Auto Exposure mode
   ( 7) ae-lock                 : Auto Exposure lock
   ( 8) exposure-time           : Manual exposure time in nanoseconds. Used when the AE mode is set to 'off'
   ( 9) exposure-table          : A GstStructure describing exposure table
   (10) white-balance-mode      : White Balance mode.
   (11) white-balance-lock      : Locks current White Balance values from changing. Affects only non-manual white balan
ce modes.
   (12) manual-wb-settings      : Manual White Balance settings such as color correction temperature and R/G/B gains. U
sed in manual white balance modes.
   (13) af-mode                 : Auto Focus mode
   (14) infrared-mode           : Infrared Mode
   (15) iso-mode                : ISO exposure mode
   (16) noise-reduction         : Noise reduction filter mode
   (17) noise-reduction-tuning: A GstStructure describing noise-reduction-tuning
   (18) zoom                    : Camera zoom rectangle ('<X, Y, WIDTH, HEIGHT >') in sensor active pixel array coordin
ates
   (19) defog-table             : A GstStructure describing defog table
   (20) ltm-data                : A GstStructure describing local tone mapping data
   (21) sharpness               : Image Sharpness Strength
 -------------------------------- video_0 Pad ---------------------------------
   (22) framerate               : Target framerate in frames per second for displaying
   (23) bitrate                 : Target bitrate in bits per second for compressed streams
   (24) idr-interval            : IDR interval for compressed streams
 ==================================== Signals ===================================
   (25) capture-image
 ==================================== Other ====================================
   (q) Quit

Choose an option: 18
 --------------------------------------------------------------------------------

 Current value: < (int)0, (int)0, (int)0, (int)0 >
 --------------------------------------------------------------------------------

Enter value (or press Enter to keep current one): <960,540,960,540>
```

Figure 2-6. Screen Capture

2.11.7. Lens distortion correction

The LDC is the process to correct the distortion that is introduced due to fish eye lens, which makes a straight line in a scene to be captured as a curved line due to the distortion of lens.

LDC is part of the use case.xml as a separate use case with the output of IPE node passed to the image warp CHI node. The iWarp CHI node has the static mesh hardcoded. The CHI node calls the image warping (iWarp) library APIs internally.

The iWarp module is a software feature that runs on the Adreno GPU. It applies warping to the given input image based on the input static mesh. The iWarp library supports OpenGL:

In one terminal run weston server:

```
$ adb shell
# . /etc/profile
# export XDG_RUNTIME_DIR=/dev/socket/weston
# chmod 0700 $XDG_RUNTIME_DIR
# cd /usr/bin/
# ./weston --"tty=1"
```

In another terminal run the **gstreamer** with LDC use case:

```
$ adb shell
# . /etc/profile
# chown -R qmmfsvr:qmmfsvr /dev/socket/weston
```

Apply the following gst use cases and enter **ldc=TRUE** in gstreamer.

```
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf ldc=TRUE qmmf.video_0 ! video/x-h264,
format=NV12, width=1920,height=1080,framerate=30/1 ! h264parse config-interval=1 !
mpegtsmux name=muxer ! queue ! tcpserversink port=8900 host=127.0.0.1
```

Refer to Push video stream to PC via adb tool to play the stream via **adb**.

2.12. Video

2.12.1. Video recorder

Support single H264/H265 stream encoding with MP4 file dump

To record 4K hardware encoded H264 video, run the following command.

```
$ adb root && adb remount && adb shell mount -o remount,rw /
$ adb shell
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf ! video/x-
h264,format=NV12,width=3840,height=2160,framerate=30/1 ! h264parse ! mp4mux ! queue !
filesink location="/data/mux_h264_4k_avc.mp4"
```

To record 1080p hardware encoded H265 video, run the following command.

```
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf ! video/x-
h265,format=NV12,width=1920,height=1080,framerate=30/1 ! h265parse ! mp4mux ! queue !
filesink location="/data/mux_h265_avc.mp4"
```

To stop the recording, click **CTRL+C**.

2.12.2. Multiplex decoding

Multi-decoder demo configures the decoder by reading the content of the configuration file, and supports RTSP URL and local file decoding.

The demo display screen is through the DP interface monitor.

Step 1.Configure **Weston** environment

To Download *Turbox-C610-aarch64_AI_Demo_Firmware.tgz*, click here.

Push the *firmware package* to target device's */data/* directory.

```
$ adb root
$ adb disable-verity && adb reboot
$ adb root && adb shell mount -o remount,rw /
$ adb push Turbox-C610-aarch64_AI_Demo_Firmware.tgz /data/
$ adb shell
# tar -zxvf /data/Turbox-C610-aarch64_AI_Demo_Firmware.tgz -C /data/
```

Unpack *Turbox-C610-aarch64_Weston_DP_Firmware.tgz* under root directory to enable **weston** output to DP.

```
# tar -zxvf /data/Turbox-C610-aarch64_AI_Demo_Firmware/aarch64-weston-dp.tgz -C /
```

Step 2.Run the **multi-decoder**, and the demo video will play on the DP screen.

```
# ./data/Turbox-C610-aarch64_AI_Demo_Firmware/weston_dp_cmd decode
```

Step 3.Exit **Weston**.

```
# killall weston
```

2.12.3. Multiplex streaming

The following gst-launch pipeline provides the capture of two 1080p resolution H264/AVC encoded video streams. One stream is stored to file while the other is streamed over adb network (TCP).

```
$ adb root && adb remount && adb shell mount -o remount,rw /
$ adb shell
# gst-launch-1.0 -e qtiqmmfsrc name=qmmf ! video/x-
h264,format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse ! mp4mux ! queue !
filesink location="/data/mux4k.mp4" qmmf. ! video/x-
h264,format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse config-interval=1 !
mpegtsmux name=muxer ! queue ! tcpserversink port=8900 host=127.0.0.1
```

- The qtiqmmfsrc element is used to capture and encode both video streams.
- The h264parse and mp4mux elements process the buffers from the first stream and prepare them for storage.
- The filesink elements stores the buffers from first stream in the file.
- The h264parse and mpegtsmux elements process the buffers from the second stream and prepare them for streaming.

The tcpserversink element then sends the buffers on the network. The queue element makes sure each path/track runs independently of the other.Refer to Push video stream to PC via adb tool to play the stream via **adb**.

To stop the use case, press **CTRL** + **C**, pull the recorded content from device using the following **adb** pull command, and then play content on host PC.

```
$adb pull /data/mux4k.mp4
```

2.12.4. Video file playback on LCD

Prerequisite: Weston must be running.

Run the following command to verify:

```
# ps | grep weston
xxxx root      1h07 weston --idle-time=4294967
```

NOTE: If the processing output is not displayed, refer to DSI LCD panel.

Following command uses the filesrc plug-in to read the file, which is then demuxed by the qtdemux plug-in, parsed by the h264parse plug-in, and decoded by the qtivdec plug-in. Then, the decoded YUV buffers are rendered to LCD display by the waylandsink plug-in, which is client to Weston server. The /data/demo.mp4 format supports 720p, 1080p,and 4K(3840x2160) 30FPS, users can record by themselves.

Run the following command to start video playback:

```
# export XDG_RUNTIME_DIR=/dev/socket/weston
# /usr/bin//weston --tty=1 --idle-time=0 &
# export XDG_RUNTIME_DIR=/dev/socket/weston && gst-launch-1.0 filesrc
location=/data/demo.mp4 ! qtdemux name=demux demux. ! queue ! h264parse ! qtivdec !
video/x-raw\(memory:GBM\),compression=ubwc ! waylandsink fullscreen=true
```

2.12.5. HDMI IN

Refer to the figure below to locate HDMI IN interface on IMX415 camera module.

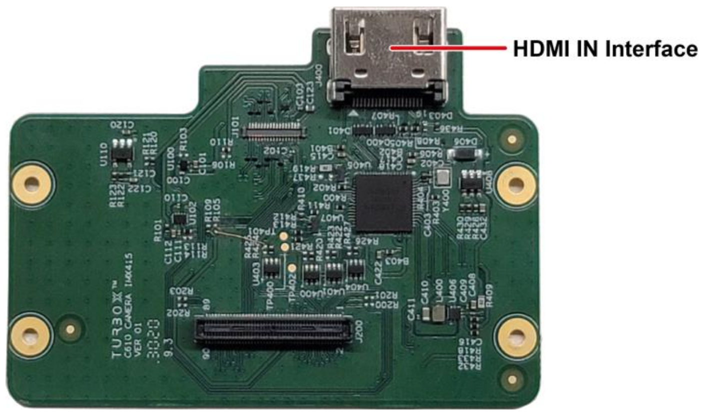


Figure 2-8. IMX415 Camera Module

Resolution supported by HDMI IN:

```
(1)1920×1080, Hz:60
(2)3840×2160, Hz:30
```

Step 1.Connect HDMI to test host computer or laptop and connect a USB device to your computer.

Step 2.After system bootup, check the firmware file. Firmware name and path: *lib/firmware/lt6911uxc.bin*.

```
$ adb root && adb remount && adb shell mount -o remount,rw /
$ adb shell
# mkdir -p /data/misc/display
# echo 0 >/data/misc/display/gbm_dbg_cfg.txt
# ls lib/firmware/lt6911uxc.bin
# reboot
```

Step 3.Open another terminal. Execute the following commands to detect HDMI IN interface and upgrade firmware for the first time.

```
$ adb root && adb remount && adb shell mount -o remount,rw /$ adb forward tcp:8900
tcp:8900
$ adb shell
# source /etc/gstreamer1.0/set_gst_env.sh
# gst-launch-1.0 -e  qtiqmmfsrc name=qmmf shdr=TRUE af-mode=3 qmmf.video_0 !
video/x-h264,format=NV12,width=1920,height=1080,framerate=30/1 ! h264parse config-
interval=1 ! mpegtsmux name=muxer ! queue ! tcpserversink port=8900 host=127.0.0.1
```

Step 4.Wait for fimware upgrading. It takes about 45s. Check the log to make sure whether the upgrade is successful or not.

```
# adb wait-for-device logcat -b kernel | tee 1234.txt | grep -iE
"lt6911uxc|cam_qup_i2c_read"
02-07 09:50:07.437     0     0 E CAM_ERR : CAM-SENSOR: lt6911uxc_firmware_upgrade:
1256 lt6911uxc: firmware upgrade success
```

Step 5.Wait a few seconds, then open another terminal and run the following command to display the video stream.

```
# vlc -vvv tcp://127.0.0.1:8900
```

NOTE: You can switch resolution from the source end by synchronizing the resolution parameters.

2.13. Audio

2.13.1. Playback WAV

Connect the speaker to board.

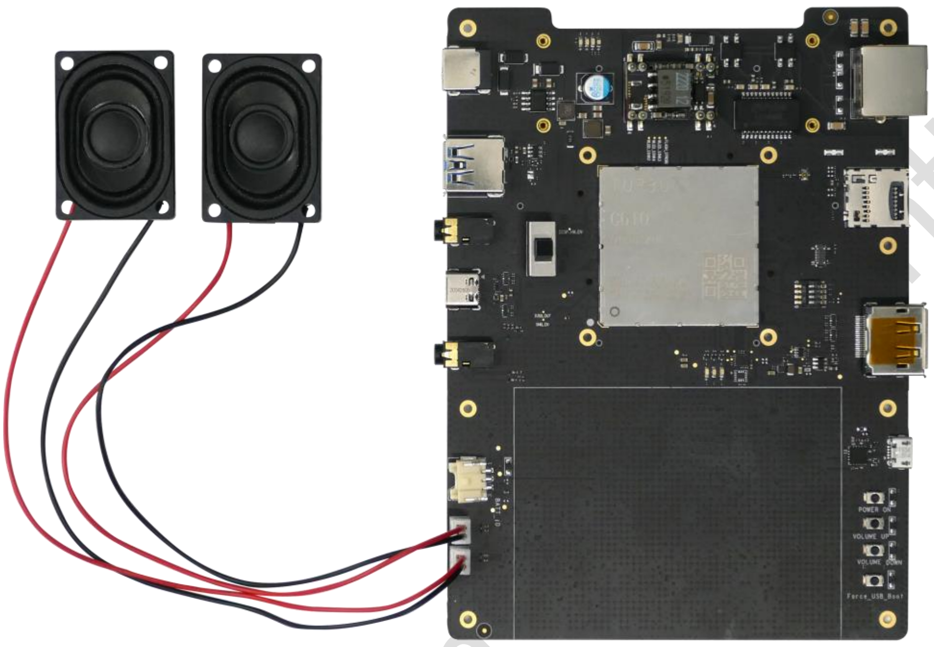


Figure 2-9. Speaker Connection

To download the test file, go to:

https://thundercomm.s3.ap-northeast-1.amazonaws.com/shop/doc/1582852506026401/audiotest.wav.

- Playback under tinyalsa via speaker

  1. Push contents

```
$ adb push audiotest.wav /data/
```

  2. Playback

```
$ adb shell
# systemctl stop pulseaudio
# tinymix set "SLIM RX0 MUX" "AIF1_PB"
# tinymix set "CDC_IF RX0 MUX" "SLIM RX0"
# tinymix set "SLIM RX1 MUX" "AIF1_PB"
# tinymix set "CDC_IF RX1 MUX" "SLIM RX1"
# tinymix set "SLIM_0_RX Channels" "Two"
# tinymix set "RX INT7_1 MIX1 INP0" "RX0"
# tinymix set "RX INT8_1 MIX1 INP0" "RX1"
# tinymix set "COMP7 Switch" "1"
# tinymix set "COMP8 Switch" "1"
# tinymix set "SpkrLeft COMP Switch" "1"
# tinymix set "SpkrLeft BOOST Switch" "1"
# tinymix set "SpkrLeft VISENSE Switch" "1"
# tinymix set "SpkrLeft SWR DAC_Port Switch" "1"
# tinymix set "SpkrRight COMP Switch" "1"
# tinymix set "SpkrRight BOOST Switch" "1"
# tinymix set "SpkrRight VISENSE Switch" "1"
# tinymix set "SpkrRight SWR DAC_Port Switch" "1"
# tinymix set "SLIMBUS_0_RX Audio Mixer MultiMedia1" "1"
# tinyplay /data/audiotest.wav
```

3. Disable audio path

```
$ adb shell
# tinymix set "SLIM RX0 MUX" "ZERO"
# tinymix set "CDC_IF RX0 MUX" "SLIM RX0"
# tinymix set "SLIM RX1 MUX" "ZERO"
# tinymix set "CDC_IF RX1 MUX" "SLIM RX1"
# tinymix set "SLIM_0_RX Channels" "One"
# tinymix set "RX INT7_1 MIX1 INP0" "ZERO"
# tinymix set "RX INT8_1 MIX1 INP0" "ZERO"
# tinymix set "COMP7 Switch" "0"
# tinymix set "COMP8 Switch" "0"
# tinymix set "SpkrLeft COMP Switch" "0"
# tinymix set "SpkrLeft BOOST Switch" "0"
# tinymix set "SpkrLeft VISENSE Switch" "0"
# tinymix set "SpkrLeft SWR DAC_Port Switch" "0"
# tinymix set "SpkrRight COMP Switch" "0"
# tinymix set "SpkrRight BOOST Switch" "0"
# tinymix set "SpkrRight VISENSE Switch" "0"
# tinymix set "SpkrRight SWR DAC_Port Switch" "0"
# tinymix set "SLIMBUS_0_RX Audio Mixer MultiMedia1" "0"
```

- Playback under hal_play_test_64bit via speaker

1. Push contents

```
$ adb push audiotest.wav /data/
```

2. Playback

```
$ adb shell
# setenforce 0
# systemctl stop pulseaudio
# hal_play_test_64bit -f /data/audiotest.wav -t 1 -d 2 -v 0.3 -r 48000 -c 2
```

- Playback under tinyalsa via headphones

1. Push contents

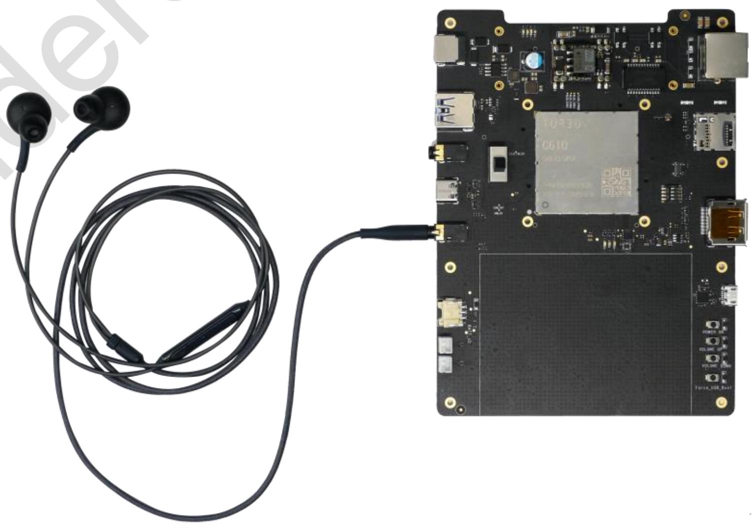


Figure 2-10. Headphone Connection

```
$ adb push audiotest.wav /data/
```

2. Playback

```
$ adb shell
# systemctl stop pulseaudio
# tinymix set "SLIM RX2 MUX" "AIF4_PB"
# tinymix set "SLIM RX3 MUX" "AIF4_PB"
# tinymix set "SLIM_6_RX Channels" "Two"
# tinymix set "RX INT1_2 MUX" "RX2"
# tinymix set "RX INT2_2 MUX" "RX3"
# tinymix set "SLIMBUS_6_RX Audio Mixer MultiMedia1" "1"
# tinyplay /data/audiotest.wav
```

3. Disable audio path

```
$ adb shell
# systemctl stop pulseaudio
# tinymix set "SLIM RX2 MUX" "ZERO"
# tinymix set "SLIM RX3 MUX" "ZERO"
# tinymix set "SLIM_6_RX Channels" "Two"
# tinymix set "RX INT1_2 MUX" "ZERO"
# tinymix set "RX INT2_2 MUX" "ZERO"
# tinymix set "SLIMBUS_6_RX Audio Mixer MultiMedia1" "0"
```

2.13.2. Capture with Dmics

- Capture via single DMIC

1. Capture with DMIC0

```
$ adb shell
# systemctl stop pulseaudio
# tinymix set 'AIF1_CAP Mixer SLIM TX0' 1
# tinymix set 'CDC_IF TX0 MUX' DEC0
# tinymix set 'ADC MUX0' DMIC
# tinymix set 'DMIC MUX0' DMIC0
# tinymix set 'SLIM_0_TX Channels' One
# tinymix set 'MultiMedia1 Mixer SLIM_0_TX' 1
# tinycap /data/dmic0.wav -D 0 -d 0 -t 10 -r 48000 -c 1
```

2. Capture with DMIC1

```
$ adb shell
# systemctl stop pulseaudio
# tinymix set 'AIF1_CAP Mixer SLIM TX0' 1
# tinymix set 'CDC_IF TX0 MUX' DEC0
# tinymix set 'ADC MUX0' DMIC
# tinymix set 'DMIC MUX0' DMIC1
# tinymix set 'SLIM_0_TX Channels' One
# tinymix set 'MultiMedia1 Mixer SLIM_0_TX' 1
# tinycap /data/dmic1.wav -D 0 -d 0 -t 10 -r 48000 -c 1
```

3. Disable audio path

```
$ adb shell
# tinymix set 'AIF1_CAP Mixer SLIM TX0' 0
# tinymix set 'CDC_IF TX0 MUX' ZERO
# tinymix set 'ADC MUX0' AMIC
# tinymix set 'DMIC MUX0' ZERO
# tinymix set 'SLIM_0_TX Channels' One
```

- Capture via 6 dmics

1. Capture

```
$ adb shell
# systemctl stop pulseaudio
# tinymix set 'AIF1_CAP Mixer SLIM TX0' 1
# tinymix set 'CDC_IF TX0 MUX' DEC0
# tinymix set 'ADC MUX0' DMIC
# tinymix set 'DMIC MUX0' DMIC0
# tinymix set 'AIF1_CAP Mixer SLIM TX1' 1
# tinymix set 'CDC_IF TX1 MUX' DEC1
# tinymix set 'ADC MUX1' DMIC
# tinymix set 'DMIC MUX1' DMIC1
# tinymix set 'AIF1_CAP Mixer SLIM TX2' 1
# tinymix set 'CDC_IF TX2 MUX' DEC2
# tinymix set 'ADC MUX2' DMIC
# tinymix set 'DMIC MUX2' DMIC2
# tinymix set 'AIF1_CAP Mixer SLIM TX6' 1
# tinymix set 'CDC_IF TX6 MUX' DEC6
# tinymix set 'ADC MUX6' DMIC
# tinymix set 'DMIC MUX6' DMIC3
# tinymix set 'AIF1_CAP Mixer SLIM TX4' 1
# tinymix set 'CDC_IF TX4 MUX' DEC4
# tinymix set 'ADC MUX4' DMIC
# tinymix set 'DMIC MUX4' DMIC4
# tinymix set 'AIF1_CAP Mixer SLIM TX5' 1
# tinymix set 'CDC_IF TX5 MUX' DEC5
# tinymix set 'ADC MUX5' DMIC
# tinymix set 'DMIC MUX5' DMIC5
# tinymix set 'MultiMedia1 Mixer SLIM_0_TX' 1
# tinymix set 'SLIM_0_TX Channels' Six
# tinycap /data/6mics.wav -D 0 -d 0 -t 10 -r 48000 -c 6
```

2. Disable audio path

```
# tinymix set 'AIF1_CAP Mixer SLIM TX0' 0
# tinymix set 'CDC_IF TX0 MUX' ZERO
# tinymix set 'ADC MUX0' AMIC
# tinymix set 'DMIC MUX0' ZERO
# tinymix set 'AIF1_CAP Mixer SLIM TX1' 0
# tinymix set 'CDC_IF TX1 MUX' ZERO
# tinymix set 'ADC MUX1' AMIC
# tinymix set 'DMIC MUX1' ZERO
# tinymix set 'AIF1_CAP Mixer SLIM TX2' 0
# tinymix set 'CDC_IF TX2 MUX' ZERO
# tinymix set 'ADC MUX2' AMIC
# tinymix set 'DMIC MUX2' ZERO
# tinymix set 'AIF1_CAP Mixer SLIM TX6' 0
# tinymix set 'CDC_IF TX6 MUX' ZERO
# tinymix set 'ADC MUX6' AMIC
# tinymix set 'DMIC MUX6' ZERO
# tinymix set 'AIF1_CAP Mixer SLIM TX4' 0
# tinymix set 'CDC_IF TX4 MUX' ZERO
# tinymix set 'ADC MUX4' AMIC
# tinymix set 'DMIC MUX4' ZERO
# tinymix set 'AIF1_CAP Mixer SLIM TX5' 0
# tinymix set 'CDC_IF TX5 MUX' ZERO
# tinymix set 'ADC MUX5' AMIC
# tinymix set 'DMIC MUX5' ZERO
# tinymix set 'MultiMedia1 Mixer SLIM_0_TX' 0
# tinymix set 'SLIM_0_TX Channels' One
```

- Play the wav file

```
$ adb pull /data/dmic0.wav .
$ adb pull /data/dmic1.wav .
$ adb pull /data/6mics.wav .
```

Play the **.wav*, and you can hear what you have recorded.

2.14. TensorFlow lite

Step 1.Test TensorFlow lite with label image.

```
$ adb shell mkdir /data/tf
```

Step 2.Acquire model and label file.

```
$ wget
https://storage.googleapis.com/download.tensorflow.org/models/tflite/mobilenet_v1_22
4_android_quant_2017_11_08.zip
$ unzip mobilenet_v1_224_android_quant_2017_11_08.zip
$ adb push mobilenet_quant_v1_224.tflite /data/tf
$ adb push labels.txt /data/tf
```

Step 3.Acquire sample image.

```
$ wget
https://upload.wikimedia.org/wikipedia/commons/thumb/a/ad/Commodore_Grace_M._Hopper%
2C_USN_%28covered%29.jpg/800px-Commodore_Grace_M._Hopper%2C_USN_%28covered%29.jpg
$ convert 800px-Commodore_Grace_M._Hopper\,_USN_\(covered\).jpg grace_hopper.bmp
$ adb push grace_hopper.bmp /data/tf
$ adb shell
# cd /data/tf/
# label_image -a 1 -m mobilenet_quant_v1_224.tflite -c 10
```

Step 4.The test is successful with the following information displayed on your screen.

```
/data/tf # label_image -a 1 -m mobilenet_quant_v1_224.tflite -c 10
Loaded model mobilenet_quant_v1_224.tflite
resolved reporter
INFO: Initialized TensorFlow Lite runtime.
Input file: ./grace_hopper.bmp
Input file size: 2400138
Input file width, height, channels: 800, 1000, 3
INFO: Created TensorFlow Lite delegate for NNAPI.

[/home/turbox/lxf/project/qcs610_2/apps_proc/build-qti-distro-fullstack-debug/tmp-
glibc/work/armv7ahf-neon-oe-linux-gnueabi/nn-framework/1.0-
r0/nn/runtime/Manager.cpp(468):(2)]DeviceManager::DeviceManager
[/local/mnt/workspace/lnxbuild/project/trees_in_use/free_tree_dir/checkout/build-
qti-distro-fullstack-perf/tmp-glibc/work/armv7ahf-neon-oe-linux-gnueabi/nnhal/1.0-
r0/nnhal/unifiedhal/driver/src/NnHalService.cpp(381):(2)]nnhal main 1.2-17
INFO: Replacing 31 node(s) with delegate (TfLiteNnapiDelegate) node, yielding 1
partitions.
Applied NNAPI delegate.invoked
average time: 12.6158 ms
0.121569: 620 lampshade
0.0745098: 495 chime
0.0745098: 410 analog clock
0.0627451: 550 envelope
0.0627451: 521 crib
```

2.15. SNPE

Step 1.Preparation before SNPE validation:

- To Download Qualcomm's sdk, go to: https://developer.qualcomm.com/software/qualcomm-neural-processing-sdk.
- Refer to SNPE documentation from: https://developer.qualcomm.com/docs/snpe/overview.html.
- Push DSP libs to *lib/dsp to /usr/lib/rfsa/adsp/*.
- Push all binaries to *bin/armv7-oe-linux-gcc8.2hf to /usr/bin/*.

- Run `chmod 755` command to set permission for required binaries.

Step 2.Validate SNPE runtime.

1. Run the following command.

```
$ adb shell
# snpe-platform-validator --runtime all
```

2. With the expected output displaying on your screen, SNPE DSP runtime is ready for use.

```
# PF_VALIDATOR: DEBUG: starting calculator test
# PF_VALIDATOR: DEBUG: Successfully loaded DSP library -
#'libcalculator_domains.so'. Setting up pointers.
# PF_VALIDATOR: DEBUG: Success in executing the sum function
# Runtime DSP Prerequisites: Present.
# PF_VALIDATOR: DEBUG: Calling PlatformValidator->setRuntime
# PF_VALIDATOR: DEBUG: CPU side validation passed.
```

Step 3.Run models on SNPE.

```
snpe-net-run --container <full path to model > --input_list <target_raw_list.txt> --
perf_profile balanced --use_dsp
```

For more information, refer to documentation of converting your model to SNPE DLC format and generating input images.

Example:

1. Preparation before SNPE validation.

```
$ adb root
$ adb remount
$ adb disable-verity
$ adb reboot
$ adb root
$ adb shell
# mount -o remount,rw /
# exit
```

2. Download the example: https://thundercomm.s3.ap-northeast-1.amazonaws.com/shop/doc/1582852506026401/snpe.zip.

```
$ wget https://thundercomm.s3.ap-northeast-
1.amazonaws.com/shop/doc/1582852506026401/snpe.zip
$ unzip snpe.zip
```

3. Run the script for SNPE.

```
$ cd snpe
$ chmod 777 snpe.sh
$ ./snpe.sh
```

4. Perform the testing process.

```
$ adb shell
# cd data
# snpe-net-run --container bvlc_alexnet.dlc  --input_list target_raw_list.txt --
perf_profile balanced --use_dsp
```

5. If the following information displayed on your screen, the test is successful.

```
# snpe-net-run --container bvlc_alexnet.dlc  --input_list target_raw_list.
txt --perf_profile balanced --use_dsp
-------------------------------------------------------------------------------
Model String: N/A
SNPE v1.37.0.788
-------------------------------------------------------------------------------
Processing DNN input(s):
```

```
trash_bin.raw
Processing DNN input(s):
notice_sign.raw
Processing DNN input(s):
chairs.raw
Processing DNN input(s):
plastic_cup.raw
Processing DNN input(s):
handicap_sign.raw
```

2.16. Face detection

The C410/C610 open kit has strong AI computing power and can support different types of detection algorithms. The face detection algorithm sample supports the face detection function of decoded images, and draws a block diagram in the face position, showing the detection effect intuitively.

Face-detector demo configures the decoder by reading the content of the configuration file, and supports RTSP URL and local file decoding. face-detector demo supports up to 1 channel.

If you want to decode the local file, change the path to the absolute path of the file and change rtsp to 0, for example path=/data/video.mp4, rtsp=0. If you want to decode the RTSP stream, change the path to URL and change rtsp ro 1.

The configuration file format is as follows:

```
[decoder_0]
cameraid=0
cameraname=cameraOne
decode=h264
enable=on
height=1080
path=rtsp://admin:1234@10.0.20.23:554
width=1920
rtsp=1
```

The demo display screen is through the DisplayPort interface monitor.

Step 1.Configure Weston environment.

To Download *Turbox-C610-aarch64_AI_Demo_Firmware.tgz*, click here.

Push the *firmwarw package* to target device's */data/* directory.

```
$ adb root && adb shell mount -o remount,rw /
$ adb push demo-ai.tgz /data/
$ adb shell
# tar -zxvf /data/Turbox-C610-aarch64_AI_Demo_Firmware.tgz -C /data/
```

Unpack *weston.tgz* under root directory.

```
# tar -zxvf /data/Turbox-C610-aarch64_AI_Demo_Firmware/aarch64-weston-dp.tgz -C /
```

Step 2.Run the **multi-decoder** app, and the demo video will play on the DP screen.

```
# ./data/Turbox-C610-aarch64_AI_Demo_Firmware/weston_dp_cmd face
```

Step 3.Exit **Weston**.

```
# killall weston
```

2.17. Sleep mode (to be updated)

Step 1.Connect the board to PC via serial port only.

```
# cd /sys/power/
# echo mem > autosleep
[ 1040.139909] PM: suspend entry (deep)
[ 1040.149367] PM: Syncing filesystems ... done.
[ 1040.217636] Freezing user space processes ... (elapsed 0.005 seconds) done.
[ 1040.230908] OOM killer disabled.
[ 1040.234297] Freezing remaining freezable tasks ... (elapsed 0.003 seconds) done.
[ 1040.251446] [kworker/u16:17][0x4a9de0784][02:43:49.187919] wlan: [2744:E:OSIF]
wlan_abort_scan: 1645: Failed get vdev
[ 1040.262554] [kworker/u16:17][0x4a9e148bf][02:43:49.199029] wlan: [2744:E:OSIF]
wlan_abort_scan: 1645: Failed get vdev
[ 1040.275014] [kworker/u16:17][0x4a9e4eb99][02:43:49.211440] wlan: [2744:I:HDD]
hdd_suspend_wlan: 1059: WLAN being suspended by OS
[ 1040.288109] [kworker/u16:17][0x4a9e8c5a2][02:43:49.224587] wlan: [2744:I:PMO]
pmo_core_is_wow_applicable: 323: lpass enabled, enabling wow
[ 1040.301933] [kworker/u16:1][0x4a9ecd213][02:43:49.238407] wlan: [86:I:HDD]
__wlan_hdd_bus_suspend: 789: starting bus suspend
[ 1040.313584] [kworker/u16:1][0x4a9f03c49][02:43:49.250062] wlan: [86:I:WMI]
suspend type: WOW_IFACE_PAUSE_DISABLED
[ 1040.330533] [kworker/u16:1][0x4a9f5335d][02:43:49.267010] wlan: [86:I:HDD]
__wlan_hdd_bus_suspend: 844: bus suspend succeeded
[ 1040.370598] [kworker/u16:1][0x4aa00efde][02:43:49.307071] wlan: [86:I:HDD]
__wlan_hdd_bus_suspend_noirq: 905: start bus_suspend_noirq
[ 1040.383016] [kworker/u16:1][0x4aa0493bf][02:43:49.319495] wlan: [86:I:HDD]
__wlan_hdd_bus_suspend_noirq: 946: bus_suspend_noirq done
[ 1040.396756] Disabling non-boot CPUs ...
[ 1040.428472] CPU1: shutdown
[ 1040.442634] CPU2: shutdown
[ 1040.463774] IRQ 6: no longer affine to CPU3
[ 1040.464403] CPU3: shutdown
[ 1040.484129] CPU4: shutdown
[ 1040.495777] CPU5: shutdown
[ 1040.503303] CPU6: shutdown
[ 1040.510702] CPU7: shutdown
[ 1040.517194] suspend ns:   1040517185727   suspend cycles:     20034872001
```

Step 2.Press the power button to wake up board system.

```
[ 1040.529404] GICv3: CPU1: found redistributor 100 region 0:0x0000000017a80000
[ 1040.529505] CPU1: Booted secondary processor [51df805e]
[ 1040.534238] CPU1 is up
[ 1040.550239] GICv3: CPU2: found redistributor 200 region 0:0x0000000017aa0000
[ 1040.550332] CPU2: Booted secondary processor [51df805e]
[ 1040.556035] CPU2 is up
[ 1040.571983] GICv3: CPU3: found redistributor 300 region 0:0x0000000017ac0000
[ 1040.572074] CPU3: Booted secondary processor [51df805e]
[ 1040.578942] CPU3 is up
[ 1040.594862] GICv3: CPU4: found redistributor 400 region 0:0x0000000017ae0000
[ 1040.594955] CPU4: Booted secondary processor [51df805e]
[ 1040.603352] CPU4 is up
[ 1040.619365] GICv3: CPU5: found redistributor 500 region 0:0x0000000017b00000
[ 1040.619457] CPU5: Booted secondary processor [51df805e]
[ 1040.629642] CPU5 is up
[ 1040.646483] GICv3: CPU6: found redistributor 600 region 0:0x0000000017b20000
[ 1040.646568] CPU6: Booted secondary processor [51df804e]
[ 1040.653245] CPU6 is up
[ 1040.669501] GICv3: CPU7: found redistributor 700 region 0:0x0000000017b40000
[ 1040.669582] CPU7: Booted secondary processor [51df804e]
[ 1040.677476] CPU7 is up
```

Appendix 1. Notices

Thundercomm may have patents or pending patent programs covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries to service@thundercomm.com.

THUNDERCOMM PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

Changes are made periodically to the information herein; these changes will be incorporated in new editions of the publication. To provide better service, Thundercomm reserves the right to improve and/or modify the products and software programs described in the manuals, and the content of the manual, at any time without additional notice.

The software interface and function and hardware configuration described in the manuals included with your development board or system on module might not match exactly the actual configuration of that you have purchased. For the configuration of the product, refer to the related contract (if any) or product packing list, or consult the distributor for the product sales. Thundercomm may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

The products described in this document are not intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Thundercomm product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Thundercomm or third parties. All information contained in this document was obtained in specific environments and is presented as an illustration. The result obtained in other operating environments may vary.

The information of this document should not be as any invitation for offer or any advice to the visitors. Please consult the professional comments from the sales consultant prior to do any actions of invest or purchase.

Thundercomm may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any references in this publication to non-Thundercomm Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this Thundercomm product, and use of those Web sites is at your own risk.Thundercomm shall not be responsible for the content of the third party.

Any performance data contained herein was determined in a controlled environment. Therefore, the result obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

This document is copyrighted by Thundercomm and the property right of the date mentioned in this document, including but not limited trademarks, patents, copyrights, trade name etc. is are not covered by any open-source license. Thundercomm may update this document at any time without notice.

Anyone doesn't have the right to amend, reprint, republication, reproduce, transmit, distribute or any other way to use this document in business or public purpose without the prior written consent by Thundercomm.

E-mail messages sent to Thundercomm via the Internet are not guaranteed to be completely secure.Thundercomm shall not be liable for any loss incurred by the surfer when transmitting any information over the Internet or for any loss incurred by Thundercomm when sending any information over the Internet at your request.

Thundercomm has all rights under other relevant exemptions provided by laws and regulations, and Thundercomm's failure to claim or delay in claiming such rights shall not be deemed to be a waiver of such rights by Thundercomm.

Thundercomm reserves the right of final interpretation of this document.

Appendix 2. Trademarks

Thundercomm, Thundercomm Turbox, TURBOX, Thundersoft turbox are trademarks of Thundercomm Corporation or its associate companies in China and/or other countries. Intel, Intel SpeedStep, Optane, and Thunderbolt are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Microsoft, Windows, Direct3D, BitLocker, and Cortana are trademarks of the Microsoft group of companies. Mini DisplayPort (mDP), DisplayPort, and VESA are trademarks of the Video Electronics Standards Association. The terms HDMI and HDMI High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC in the United States and other countries. Wi-Fi, Wi-Fi Alliance, WiGig, and Miracast are registered trademarks of Wi-Fi Alliance. USB-C is a registered trademark of USB Implementers Forum. All other trademarks are the property of their respective owners.