

IDO-EVB3020 Ubuntu系统使用手册

调试

[串口调试](#)

[ADB调试](#)

[SSH调试](#)

[串口测试](#)

[CAN测试](#)

[WIFI使用](#)

[蓝牙使用](#)

[以太网使用](#)

[静态IP设置](#)

[4G使用](#)

[摄像头使用](#)

[10.1 测试](#)

[10.1.1 测试摄像头是否存在](#)

[10.1.2 抓取视频流](#)

[U盘](#)

[USB OTG](#)

[USB HOST](#)

[SD卡](#)

[按键](#)

[ADC](#)

[ADC值读取](#)

[ADC电压转换关系](#)

[时间设置](#)

[RTC时间读取和同步](#)

[NTP时间同步](#)

[时区](#)

[查看时区](#)

[设置时区](#)

音频

[Lineout](#)

[耳机](#)

[录音](#)

[打开mic通道](#)

[录音](#)

[播放录音](#)

显示屏

[显示屏接口说明](#)

[显示设置](#)

[屏幕背光亮度设置](#)



IDO-EVB3020

Ubuntu 系统使用手册

深圳触觉智能科技有限公司

www.industio.cn

文档修订历史

版本	修订内容	修订	审核	日期
V1.0	1、创建文档	刘崇凯		2023/3/24

调试

IDO-EVB3020开发板支持串口调试、ADB调试和远程SSH调试。

串口调试

串口调试接口位于J4端口，见下图。请使用配套的usb串口调试工具。



为TTL电平，通信参数为1500000 8 N 1。

ADB调试



上图红色框内的USB接口为支持OTG模式切换，使用双公头 USB 数据线连接开发板和 PC 端的 USB接口，在PC终端识别到 ADB 设备，即可使用 adb shell 调试。

SSH调试

SSH登录账号密码为： `ido @ 123456`。

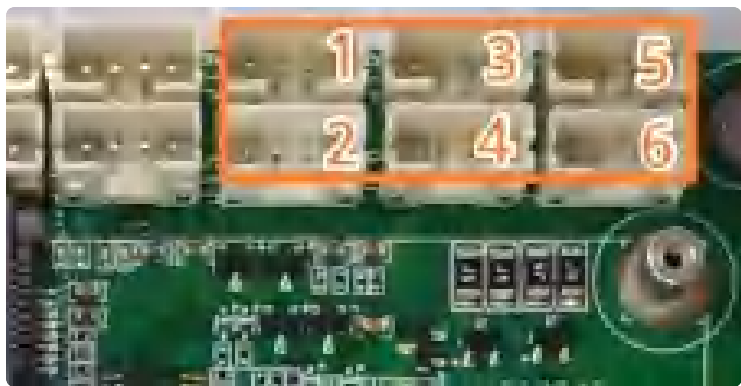
```
ido@ido: ~
login as: ido
ido@192.168.0.58's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 4.4.194 aarch64)

System information as of Fri Mar 24 06:20:34 UTC 2023

System load:  0.16 0.10 0.09   Up time:           17 min           Local users:  2
Memory usage: 28 % of 957MB   IP:             192.168.0.58
Usage of /:   52% of 6.7G

Last login: Fri Mar 24 06:18:41 2023 from 192.168.0.31
ido@ido:~$
```

串口测试



串口接口位置及引脚定义如上图所示，设备节点列表如下：

序号	接口位置	电平	串口设备节点
1	J19	RS232	/dev/ttyS0
2	J15	RS232	/dev/ttyS1

3	J18	RS232	/dev/ttyS4
4	J16	RS232	/dev/ttyS5
5	J14	RS485	/dev/ttyS3
6	J13	RS485	/dev/ttyS2 (默认此接口不开启，配置为调试串口)

其中RS485接口可以通过软件控制VCC供电，供电接口如下所示：

序号	接口位置	控电节点
1	J13	/sys/class/leds/rs485_vout1/brightness
2	J14	/sys/class/leds/rs485_vout2/brightness

以J13端口的VCC供电控制为例，控制方法如下

```

▼ Bash |
1  #断电
2  echo 0 > /sys/class/leds/rs485_vout1/brightness
3  #开启
4  echo 1 > /sys/class/leds/rs485_vout1/brightness

```

使用工具microcom，可以进行发送和接收测试。

需要先安装microcom工具

```

▼ Bash |
1  sudo apt-get update
2  sudo apt-get install microcom

```

```

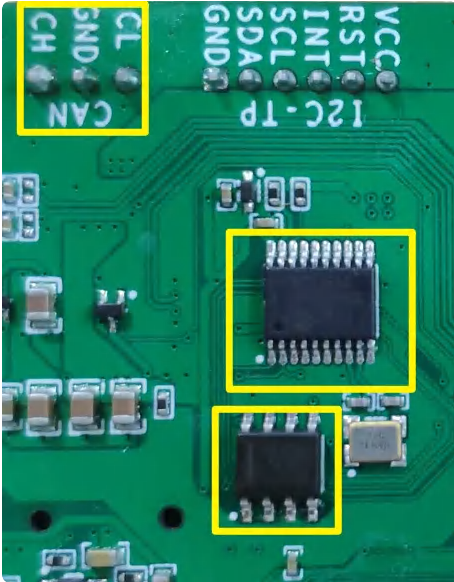
▼ Plain Text |
1  root@ido:~# microcom -s 115200 -P /dev/ttyS0
2  [ 44.730195] of_dma_request_slave_channel: dma-names property of node '/s
   erial@fdd50000' missing or empty
3  connected to /dev/ttyS0
4  Escape character: Ctrl-\
5  Type the escape character to get to the prompt.
6  fjskdfjsdfjsdklfjdsfdfsdfdfdsfsdfd

```

短接TX-RX或者接上USB转串口工具。

按下键盘，将发送对应的字符；而接收的内容，会显示在终端。

CAN测试



开发板使用 MCP2515-I IC实现 SPI转CAN，在使用之前先确认开发板硬件电路是否贴有此MCP2515和CAN收发器芯片，芯片位置如上图所示。

默认 can0 接口状态为 down，需要设置 can 参数并执行 up 操作后才能执行收发。测试时可 将开发板的 can 接口与另外一块开发板 can 接口互连，或者连接 USB 转 can 工具，收发双方 设置相同的波特率等参数来测试接口功能。

CAN接口测试方法如下：

```
1  #关闭can0设备
2  ip link set can0 down
3
4  #设置仲裁段1M波特率，数据段3M波特率
5  ip link set can0 type can bitrate 1000000 dbitrate 3000000 fd on
6
7  #打印can0信息
8  ip -details link show can0
9
10 #启动can0
11 ip link set can0 up
12
13 #执行candump，阻塞等待can0接收
14 candump can0
15
16 #canfd格式发送
17 cansend can0 123##1DEADBEEF
18
19 #can格式发送
20 cansend can0 123#1122334455667788
```

WIFI使用

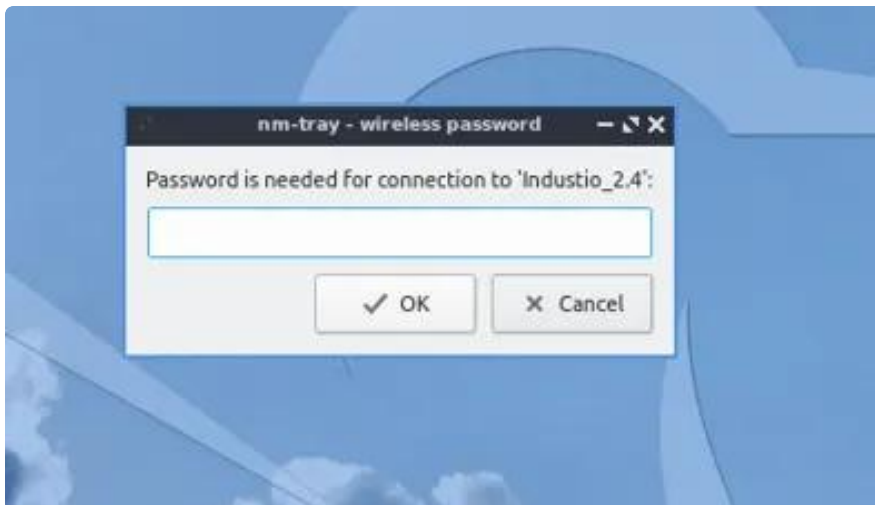
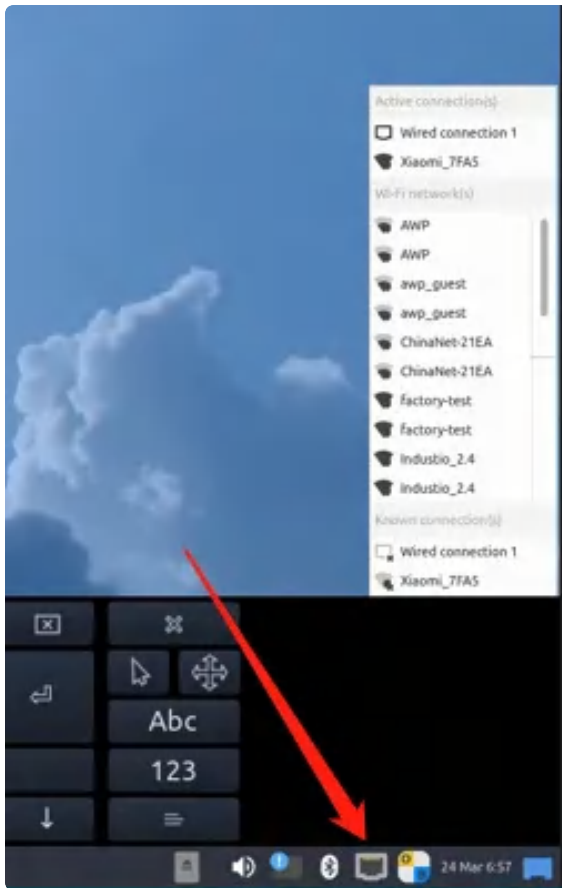
开发板板载USB WiFi 模块为RTL8723DU。

网络设备节点：wlan0

联网方法：Ubuntu–desktop系统可在界面上配置SSID和密码连接附近的WiFi路由。

在Ubuntu系统桌面连接wifi热点：

鼠标右键右下角网口图标，弹出wifi框，点击想要连接的热点。



这里我们可以通过虚拟键盘，进行输入密码。



点击上图图标会弹出虚拟键盘。

用户命令行连接wifi

修改wifi配置文件：

```
Bash |
1 root@ido:~#vi /etc/wpa_supplicant/wpa_supplicant.conf
```

连接wifi：

```
Bash |
1 root@ido:~#vi wpa_supplicant -D nl80211 -i wlan0 -c /etc/wpa_supplicant/wpa
 _supplicant.conf -B &
```

蓝牙使用

开发板板载USB WiFi+蓝牙二合一模块RTL8723DU

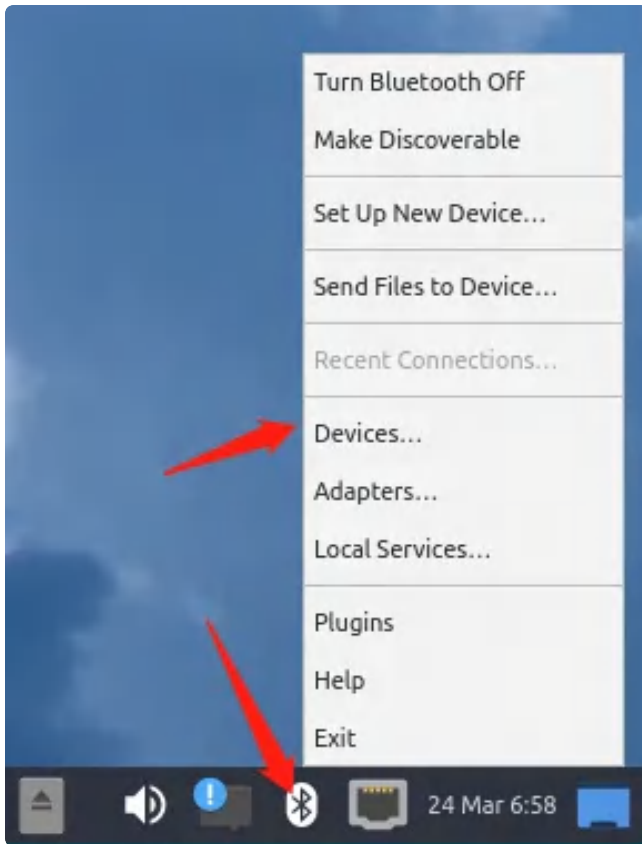
设备节点：hci0

蓝牙标准：蓝牙4.2双模

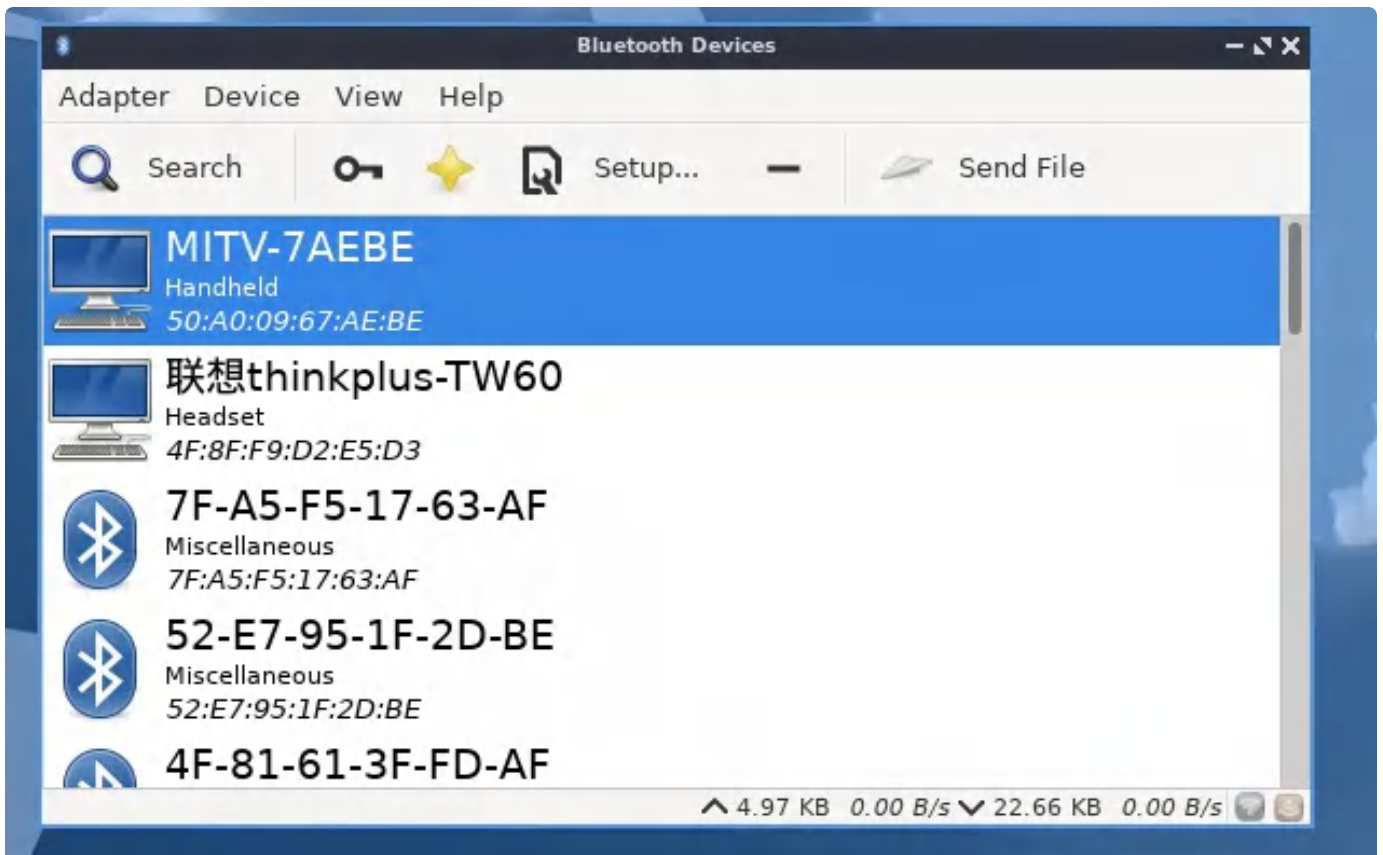
连接方法：Ubuntu系统可在界面上可以搜索附近的蓝牙并连接，

在Ubuntu系统桌面连接蓝牙：

鼠标右键点击右下角蓝牙图标，弹出选项框后，点击“Devices”



弹出蓝牙搜索界面。



右键点击相应的蓝牙设备，连接即可。

以太网使用



开发板具有一路自适应 10/100Mbps 以太网接口。

网络设备节点：eth0

IP设置方式：默认IP地址分配方式为 dhcp，只需要将以太网接口连接路由器即可为开发板动态分配 IP 地址。

静态IP设置

以eth0设置静态IP地址为例，修改/etc/network/interfaces，在文件中添加如下内容

```
1  auto lo
2  iface lo inet loopback
3
4  auto eth0
5  iface eth0 inet static
6  address 192.168.0.234
7  netmask 255.255.255.0
8  gateway 192.168.0.1
9  dns-nameservers 114.114.114.114
```

其中，dns-nameservers一项为默认dns。

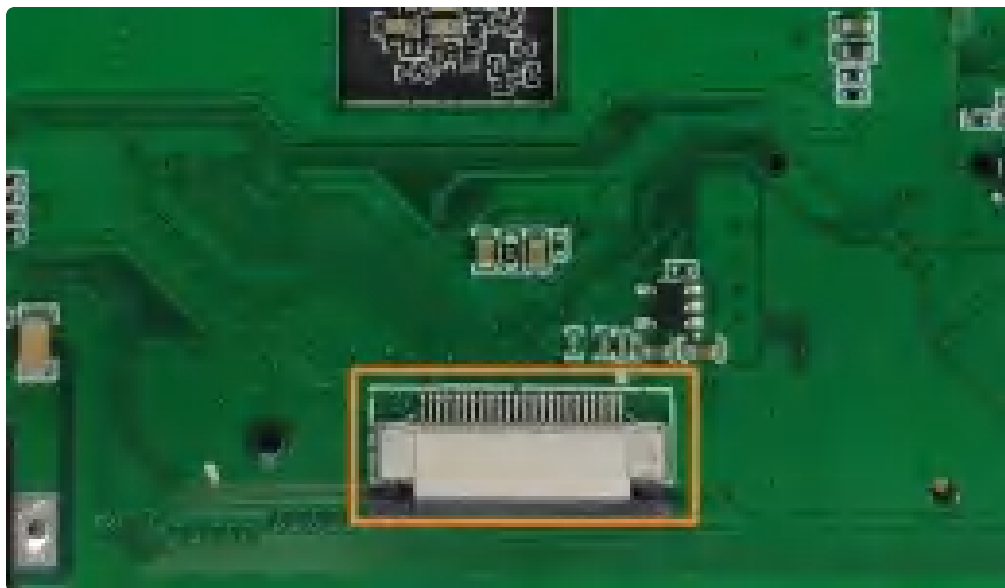
4G使用

IDO-EVB3568-V1 默认适配EC20模块，系统中已经安装了对应的拨号上网服务。

```
1 /lib/systemd/system/ec20.service
```

正常拨号成功后，wwan0将会分配到ip地址，此时可以测试是否能够正常ping通外网。

摄像头使用



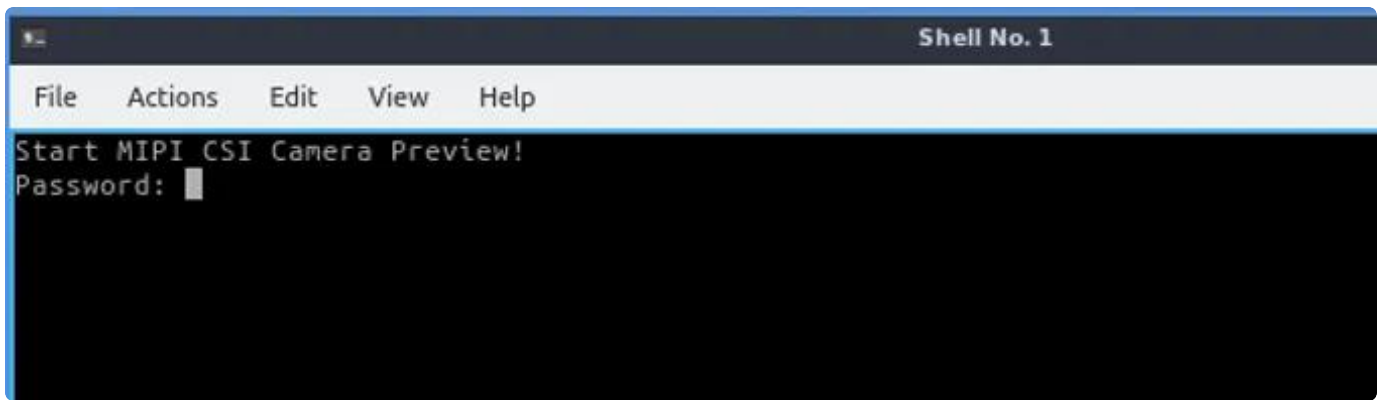
系统默认支持OV5648 MIPI摄像头模组，接口位置如上图所示。

摄像头节点为：/dev/video0

摄像头可以使用系统自带的软件Test camera打开摄像头测试。软件的位置如下图所示：



点开软件输入password：123456



10.1 测试

10.1.1 测试摄像头是否存在

```

1  root@ido:~# media-ctl -p -d /dev/media0
2  Media controller API version 0.1.0
3
4  Media device information
5  -----
6  driver          rkisp1
7  model           rkisp1
8  serial
9  bus info
10 hw revision     0x0
11 driver version  0.0.0
12
13 Device topology
14 - entity 1: rkisp1-isp-subdev (4 pads, 7 links)
15     type V4L2 subdev subtype Unknown flags 0
16     device node name /dev/v4l-subdev0
17     pad0: Sink
18         [fmt:SBGGR10_1X10/2592x1944 field:none
19         crop.bounds:(0,0)/2592x1944
20         crop:(0,0)/2592x1944]
21         <- "rkisp1_dmapath":0 []
22         <- "rockchip-mipi-dphy-rx":1 [ENABLED]
23     pad1: Sink
24         <- "rkisp1_input_params":0 [ENABLED]
25     pad2: Source
26         [fmt:YUYV8_2X8/2592x1944 field:none
27         crop.bounds:(0,0)/2592x1944
28         crop:(0,0)/2592x1944]
29         -> "rkisp1_selfpath":0 [ENABLED]
30         -> "rkisp1_mainpath":0 [ENABLED]
31         -> "rkisp1_rawpath":0 [ENABLED]
32     pad3: Source
33         -> "rkisp1_statistics":0 [ENABLED]
34
35 - entity 2: rkisp1_mainpath (1 pad, 1 link)
36     type Node subtype V4L flags 0
37     device node name /dev/video0
38     pad0: Sink
39         <- "rkisp1-isp-subdev":2 [ENABLED]
40
41 - entity 3: rkisp1_selfpath (1 pad, 1 link)
42     type Node subtype V4L flags 0

```

结果显示m00_b_ov5648，说明摄像头存在，最高分辨率支持2592x1944。

10.1.2 抓取视频流

使用v4l2-ctl工具可以抓取摄像头的视频数据流。

```

1  root@ido:~# v4l2-ctl --verbose -d /dev/video0 --set-fmt-video=width=1920,height=1080,pixelformat='NV12' --stream-mmap=4 --set-selection=target=crop,flags=0,top=0,left=0,width=1920,height=1080 --stream-to=./out.yuv
2  VIDIOC_QUERYCAP: ok
3  VIDIOC_G_FMT: ok
4  VIDIOC_S_FMT: ok
5  Format Video Capture Multiplanar:
6      Width/Height      : 1920/1080
7      Pixel Format       : 'NV12' (Y/CbCr 4:2:0)
8      Field              : None
9      Number of planes   : 1
10     Flags              :
11     Colorspace          : Default
12     Transfer Function   : Default
13     YCbCr/HSV Encoding  : Default
14     Quantization        : Full Range
15     Plane 0             :
16         Bytes per Line  : 1920
17         Size Image      : 3110400
18 VIDIOC_G_SELECTION: ok
19 VIDIOC_S_SELECTION: ok
20     VIDIOC_REQBUFS returned 0 (Success)
21     VIDIOC_QUERYBUF returned 0 (Success)
22     VIDIOC_QUERYBUF returned 0 (Success)
23     VIDIOC_QUERYBUF returned 0 (Success)
24     VIDIOC_QUERYBUF returned 0 (Success)
25     VIDIOC_QBUF returned 0 (Success)
26     VIDIOC_QBUF returned 0 (Success)
27     VIDIOC_QBUF returned 0 (Success)
28     VIDIOC_QBUF returned 0 (Success)
29     VIDIOC_STREAMON returned 0 (Success)
30 cap dqbuf: 0 seq:      1 bytesused: 3110400 ts: 1384.549991 (ts-monotonic, ts-src-eof)
31 cap dqbuf: 1 seq:      2 bytesused: 3110400 ts: 1384.616490 delta: 66.499 ms (ts-monotonic, ts-src-eof)
32 cap dqbuf: 2 seq:      3 bytesused: 3110400 ts: 1384.682975 delta: 66.485 ms (ts-monotonic, ts-src-eof)
33 cap dqbuf: 3 seq:      4 bytesused: 3110400 ts: 1384.749486 delta: 66.511 ms (ts-monotonic, ts-src-eof)
34 cap dqbuf: 0 seq:      5 bytesused: 3110400 ts: 1384.816022 delta: 66.536 ms fps: 15.04 (ts-monotonic, ts-src-eof)
35 cap dqbuf: 1 seq:      6 bytesused: 3110400 ts: 1384.882509 delta: 66.487 ms fps: 15.04 (ts-monotonic, ts-src-eof)
36 cap dqbuf: 2 seq:      7 bytesused: 3110400 ts: 1384.949025 delta: 66.516 ms fps: 15.04 (ts-monotonic, ts-src-eof)

```

```

37 cap dqbuf: 3 seq:      8 bytesused: 3110400 ts: 1385.015545 delta: 66.520
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
38 cap dqbuf: 0 seq:      9 bytesused: 3110400 ts: 1385.082051 delta: 66.506
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
39 cap dqbuf: 1 seq:     10 bytesused: 3110400 ts: 1385.148567 delta: 66.516
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
40 cap dqbuf: 2 seq:     11 bytesused: 3110400 ts: 1385.215079 delta: 66.512
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
41 cap dqbuf: 3 seq:     12 bytesused: 3110400 ts: 1385.281594 delta: 66.515
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
42 cap dqbuf: 0 seq:     13 bytesused: 3110400 ts: 1385.348115 delta: 66.521
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
43 cap dqbuf: 1 seq:     14 bytesused: 3110400 ts: 1385.414669 delta: 66.554
   ms fps: 15.03 (ts-monotonic, ts-src-eof)
44 cap dqbuf: 2 seq:     15 bytesused: 3110400 ts: 1385.481133 delta: 66.464
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
45 cap dqbuf: 3 seq:     16 bytesused: 3110400 ts: 1385.547656 delta: 66.523
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
46 cap dqbuf: 0 seq:     17 bytesused: 3110400 ts: 1385.614172 delta: 66.516
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
47 cap dqbuf: 1 seq:     18 bytesused: 3110400 ts: 1385.680680 delta: 66.508
   ms fps: 15.04 (ts-monotonic, ts-src-eof)
48 cap dqbuf: 2 seq:     19 bytesused: 3110400 ts: 1385.747241 delta: 66.561
   ms fps: 15.03 (ts-monotonic, ts-src-eof)
49 cap dqbuf: 3 seq:     20 bytesused: 3110400 ts: 1385.813714 delta: 66.473
   ms fps: 15.03 (ts-monotonic, ts-src-eof)
50 ^C

```

按Ctrl-C停止抓取，视频流保存到文件out.yuv。

使用ffplay工具播放抓取的视频流：

```

1 root@ido:~# ffplay -f rawvideo -video_size 1920x1080 -pix_fmt nv12 ./out.yuv
2 ffplay version 4.2.4-1ubuntu1.0firefly1 Copyright (c) 2003-2020 the FFmpeg
  developers
3   built with gcc 9 (Ubuntu 9.3.0-17ubuntu1~20.04)
4   configuration: --prefix=/usr --extra-version=1ubuntu1.0firefly1 --toolch
ain=hardened --libdir=/usr/lib/aarch64-linux-gnu --incdir=/usr/include/aar
ch64-linux-gnu --arch=arm64 --enable-gpl --disable-stripping --enable-avre
sample --disable-filter=resample --enable-avisynth --enable-gnutls --enabl
e-ladspa --enable-libaom --enable-libass --enable-libbluray --enable-libbs
2b --enable-libcaca --enable-libcdio --enable-libcodec2 --enable-libflite
--enable-libfontconfig --enable-libfreetype --enable-libfribidi --enable-l
ibgme --enable-libgsm --enable-libjack --enable-libmp3lame --enable-libmys
ofa --enable-libopenjpeg --enable-libopenmpt --enable-libopus --enable-lib
pulse --enable-librsvg --enable-librubberband --enable-libshine --enable-l
ibsnappy --enable-libsoxr --enable-lbspeex --enable-libssh --enable-libth
eora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-l
ibvpx --enable-libwavpack --enable-libwebp --enable-libx265 --enable-libxm
l2 --enable-libxvid --enable-libzmq --enable-libzvbi --enable-lv2 --enable
-omx --enable-openal --enable-openc1 --enable-opengl --enable-sdl2 --enabl
e-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chromaprint --enab
le-frei0r --enable-libx264 --enable-libdrm --enable-librga --enable-rkmp
p --enable-version3 --disable-libopenh264 --disable-vaapi --disable-udpau
--disable-decoder=h264_v4l2m2m --disable-decoder=vp8_v4l2m2m --disable-dec
oder=mpeg2_v4l2m2m --disable-decoder=mpeg4_v4l2m2m --disable-muxer='ac3,ea
c3,mlp,truehd' --disable-encoder='ac3_fixed,ac3,mlp,spdif,truehd' --disabl
e-demuxer='ac3,eac3,mlp,truehd,dts,dtsd' --disable-parser='aac,ac3,mlp' --
-disable-decoder='ac3,eac3,mlp,dolby_e' --enable-shared --disable-doc
5   libavutil      56. 31.100 / 56. 31.100
6   libavcodec     58. 54.100 / 58. 54.100
7   libavformat    58. 29.100 / 58. 29.100
8   libavdevice    58.  8.100 / 58.  8.100
9   libavfilter     7. 57.100 / 7. 57.100
10  libavresample   4.  0.  0 / 4.  0.  0
11  libswscale      5.  5.100 / 5.  5.100
12  libswresample   3.  5.100 / 3.  5.100
13  libpostproc    55.  5.100 / 55.  5.100
14 Option -pix_fmt is deprecated, use -pixel_format.
15 libGL error: failed to create dri screen
16 libGL error: failed to load driver: rockchip
17 libGL error: failed to create dri screen
18 libGL error: failed to load driver: rockchip
19 [rawvideo @ 0x7f3c000ba0] Estimating duration from bitrate, this may be in
  accurate
20 Input #0, rawvideo, from './out.yuv':

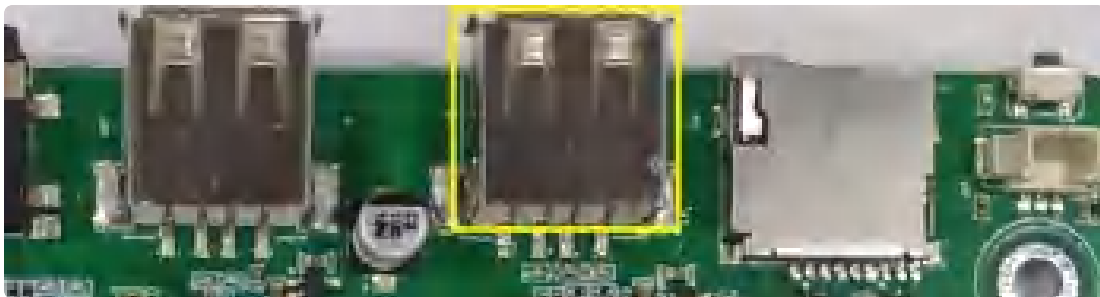
```

```
21 Duration: 00:00:04.00, start: 0.000000, bitrate: 622075 kb/s
22 Stream #0:0: Video: rawvideo (NV12 / 0x3231564E), nv12, 1920x1080, 622
23 080 kb/s, 25 tbr, 25 tbn, 25 tbc
```



U盘

USB OTG



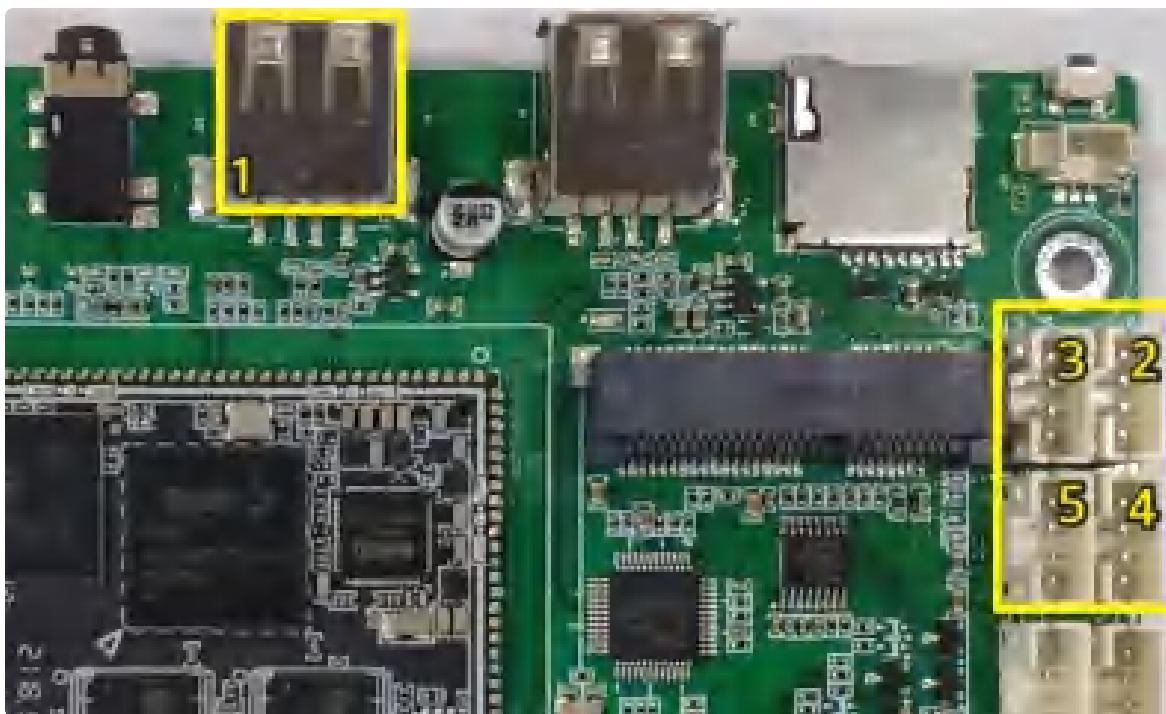
当 OTG 接口在上电时已使用 USB 数据线连接 PC 端的 USB 口，此接口会自动切换为 Device 模式，在此模式下可通过 ADB 的方式来调试开发板。当 USB OTG 接口没有链接 USB 数据线或连接 U 盘设备时，开始自动切换为 Host 模式。USB OTG 执行应用层调用写设备节点的方式控制接口的 Device 和 Host 模式，设置方法如下：

```

1  #设置为Host模式
2  echo HOST > /dev/otg_mode
3  #设置为Device模式
4  echo DEVICE > /dev/otg_mode

```

USB HOST



开发板共有5路USB HOST 2.0接口，启动一路为Type A 接口，另外四路为 PH2.0-4 端子。支持挂载U盘，USB摄像头、USB鼠标等标准USB设备。

开发板每一路USB HOST均可通过软件控制接口的供电，接口对应列表如下：

序号	位置	接口
1	TYPE-A	/sys/class/leds/usb1_pwr/brightness
2	J5	/sys/class/leds/usb2_pwr/brightness
3	J3	/sys/class/leds/usb3_pwr/brightness
4	J4	/sys/class/leds/usb4_pwr/brightness
5	J6	/sys/class/leds/usb5_pwr/brightness

以Type A接口USB HOST 供电控制为例，控制方法如下：

```
▼ Bash |
1  #开电
2  echo 1 > /sys/class/leds/usb1_pwr/brightness
3  #断电
4  echo 0 > /sys/class/leds/usb1_pwr/brightness
```

当接入U盘设备时，默认挂载到/media/ido/目录下。

```
▼ Bash |
1  root@ido:~# mount
2  ...
3  /dev/sda1 on /media/ido/KINGSTON type vfat (rw,nosuid,nodev,relatime,uid=1001,gid=1001,mask=0022,dmask=0022,codepage=936,iocharset=utf8,shortname=mixed,showexec=utf8,flush,errors=remount-ro,uhelper=udisks2)
4  ...
```

SD卡

将SD卡插入到SD卡槽中，将自动挂载到/media/ido/目录下。

```
▼ Bash |
1  root@ido:~# mount
2  ...
3  /dev/mmcblk0p1 on /media/ido/E2C4-11A5 type vfat (rw,nosuid,nodev,relatime,uid=1001,gid=1001,mask=0022,dmask=0022,codepage=936,iocharset=utf8,shortname=mixed,showexec=utf8,flush,errors=remount-ro,uhelper=udisks2)
4  ...
```

按键

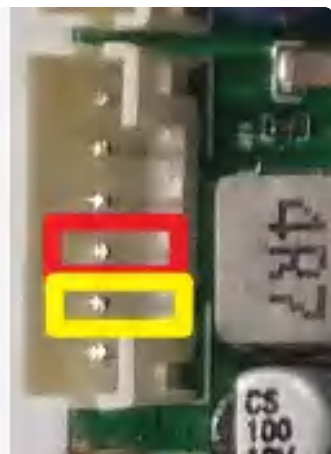
IDO-EVB3020 配置了一个Recovery按键，在设备断电的情况下，该按键用于烧录固件。在系统正常启动后，则可作为普通按键使用。对应的设备节点为/dev/input/event2，键值为KEY_VOLUMEUP。

使用evtest进行测试：

```
▼ Bash |
1 root@ido:~# evtest
2 root@ido:~# evtest
3 No device specified, trying to scan all of /dev/input/event*
4 Available devices:
5 /dev/input/event0:      rk8xx_pwrkey
6 /dev/input/event1:      PixArt Dell MS116 USB Optical Mouse
7 /dev/input/event2:      adc-keys
8 Select the device event number [0-2]: 2
9 Input driver version is 1.0.1
10 Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
11 Input device name: "adc-keys"
12 Supported events:
13   Event type 0 (EV_SYN)
14   Event type 1 (EV_KEY)
15     Event code 1 (KEY_ESC)
16     Event code 102 (KEY_HOME)
17     Event code 114 (KEY_VOLUMEDOWN)
18     Event code 115 (KEY_VOLUMEUP)
19     Event code 139 (KEY_MENU)
20 Properties:
21 Event: time 1679646794.524543, ----- SYN_REPORT -----P), v
    alue 0
22
23
```

在选择event number为2后，按下RECOVERY按键，即可看到按下和松开打印的信息。

ADC



开发板引出两路10bit有效位的数模转化器，参考电源为1.8V，读取接口如下

序号	位置	接口
1	红色	cat /sys/bus/iio/devices/iio\:device0/in_voltage0_raw
2	黄色	cat /sys/bus/iio/devices/iio\:device0/in_voltage1_raw

ADC值读取

▼

Bash

```
1 cat /sys/bus/iio/devices/iio\:device0/in_voltage0_raw
```

ADC电压转换关系

▼

Bash

```
1 V=(in_voltage0_raw/1024)*1.8v
```

假设in_voltage2_raw的值为500，则对应的ADC电压为 $V=(500/1024)*1.8v=0.879v$

时间设置

RTC时间读取和同步

系统时间读取和设置

▼

Bash

```
1 # date
2 Fri Mar 18 12:00:22 CST 2022
3 # date -s "2022-03-18 12:01:00"
```

rtc时间设置

```
▼ Bash |
1 # hwclock -r
2 2022-03-18 12:01:06.991425+08:00
3 # hwclock -w
```

NTP时间同步

系统默认开启了NTP服务，连接网络后，将自动同步网络时间。

时区

查看时区

```
▼ Bash |
1 root@ido:~# date -R
2 Wed, 26 Oct 2022 03:26:46 +0000
```

+0000表示在0时区。

设置时区

```
▼ Bash |
1 root@ido:~#export TZ='Asia/Shanghai'
2
3 root@ido:~# date -R
4 Wed, 26 Oct 2022 11:30:02 +0800
5
```

音频

使用[aplay](#)工具查看声卡设备

```

1 root@ido:~# aplay -l
2 **** List of PLAYBACK Hardware Devices ****
3 card 0: rockchiprk809co [rockchip,rk809-codec], device 0: ff070000.i2s-rk81
  7-hifi rk817-hifi-0 []
4   Subdevices: 1/1
5   Subdevice #0: subdevice #0

```

Lineout

不插入耳机，使用aplay播放wav音频测试

```

1 root@ido:/# aplay usr/share/sounds/alsa/Rear_Center.wav
2 Playing WAVE 'usr/share/sounds/alsa/Rear_Center.wav' : Signed 16 bit Little
  Endian, Rate 48000 Hz, Mono
3

```

耳机

插入耳机，使用aplay播放wav音频测试

```

1 root@ido:/# aplay usr/share/sounds/alsa/Rear_Center.wav
2 Playing WAVE 'usr/share/sounds/alsa/Rear_Center.wav' : Signed 16 bit Little
  Endian, Rate 48000 Hz, Mono
3

```

录音

打开mic通道

```

1 alsamixer

```

Capture MIC Path选择Main Mic


```
1 root@ido:~# aplay test.wav
```

显示屏

显示屏接口说明



MIPI屏接口位于开发板背面。接线方式为上接。



TP触摸接口唯一J22，接口接线方式为上接。

注意：不要接错屏线和TP，会存在烧屏和烧TP的风险。

显示设置

屏幕背光亮度设置

- MIPI屏背光控制

设备节点：/sys/class/backlight/backlight/brightness

设置方法：（支持调节范围 0-255）

```
▼ Bash |
1  #关闭
2  echo 0 > /sys/class/backlight/backlight/brightness
3  #最亮
4  echo 255 > /sys/class/backlight/backlight/brightness
```

-