软件应用指南
T536 评估板

成都亿佰特电子科技有限公司
Chengdu Ebyte Electronic Technology Co.,Ltd.

目录

免责申明和版权公告

本文中的信息，如有变更，恕不另行通知。 文档“按现状”提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档不负任何责任，包括使用本文档内信息产生的侵犯任何专利权行为的责任。本文档在此未以禁止反 言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。

文中所得测试数据均为亿佰特实验室测试所得，实际结果可能略有差异。

文中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

最终解释权归成都亿佰特电子科技有限公司所有。

注 意 ：

由于产品版本升级或其他原因，本手册内容有可能变更。亿佰特电子科技有限公司保留在没有任何通知或者提示的情况下对本手册的内容进行修改的权利。本手册仅作为使用指导，成都亿佰特电子科技有限公司尽全力在本手册中提供准确的信息，但是成都亿佰特电子科技有限公司并不确保手册内容完全没有错误，本手册中的所有陈述、信息和建议也不构成任何明示或暗示的担保。

1. 概述

Linux 软件评估指南用于介绍在亿佰特的评估板上运行开源 Linux 系统下的核心资源与外设资源的测试步骤与评估方法。本文可作为前期评估指导使用，也可以作为通用系统开发的测试指导书使用。

1.1.软件资源

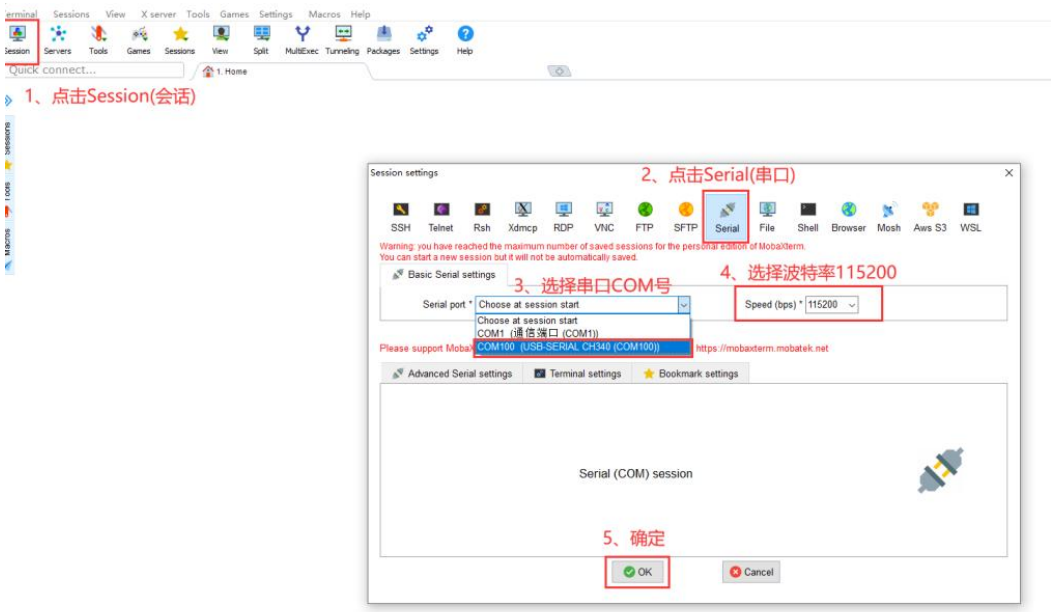
ECB33 搭载基于 Linux 5.10 版本内核的操作系统，评估板出厂附带嵌入式 Linux 系统开发所需要的交叉编译工具链， U-boot 源代码，Linux 内核和各驱动模块的源代码，以及适用于 Windows 桌面环境和 Linux 桌面环境的各种开发调试工具，应用开发样例等。

2. 使用前准备

2.1.1. 串口软件安装

使用串口前主要需要安装 CH340 串口驱动和 MobaXterm 串口终端。具体安装方法可见核心板开发指南。

连接好串口之后，安装 MobaXterm 后进行配置，按如下方式打开串口。



2.2.快速启动

亿佰特出厂的评估板默认不带启动固件，需要用户自行烧录一个固件。用户可以根据评估板型号来选择烧录固件，这里我们主要帮助用户快速启动评估板，只讲解使用官方工具烧录固件到 emmc 的方法， nand 也是一样的。更多烧录方法和烧录中的问题见

Development_Guide。

2.2.1. 烧录评估板 Flash

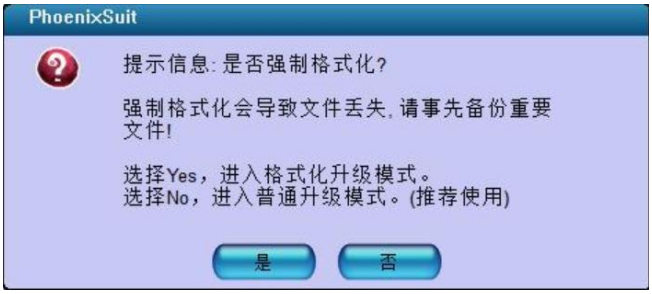
使用全志官方工具 PhoenixSuit 实现 USB 快速烧录。

点击上方的“一键刷机” 项， 进入到镜像选择。ECB33 选择后不需要点击立即升级按钮。

ECB33 选择 ebyte_t536_uart0_linux-5.10-origin.img

接下来我们连接两根 type-a 和 type-c 数据线，一根用来观察烧写日志，一根用来烧写板上 flash。连接好之后，我们需要进入烧写模式。 **type-a 评估板这边必须要接入上面的口子，上面才是 OTG**。

ECB33 引出了 FEL 按键，用户可以按下 FEL 然后保持，再按下复位按键或者给板子上电，即可进入烧写模式，弹出烧写询问框。



用户按需选择是否格式化升级，等待烧录完成即可。

需要注意 T536 优先从 SD 卡进行启动，烧录到板上 flash 之后，拔掉 sd 卡才能从板上 flash 上启动。

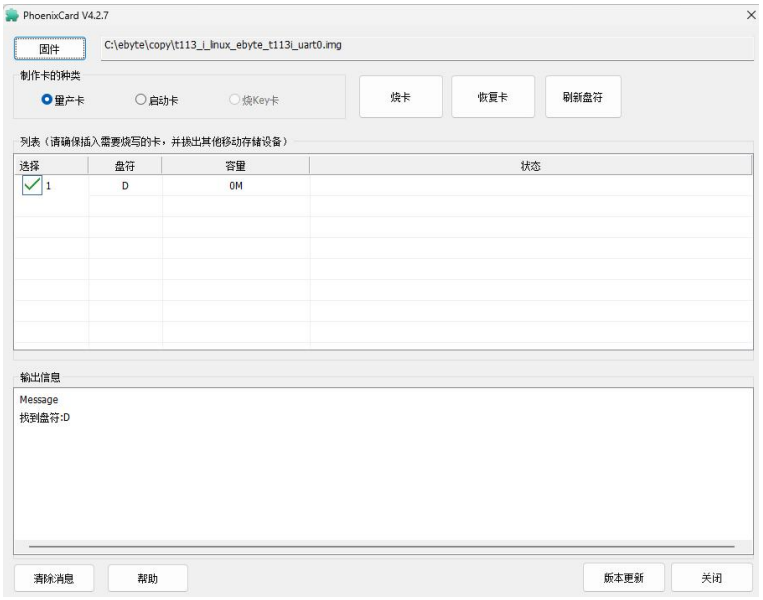
2.2.2. 使用串口交互评估板

在烧录完成之后，我们可以接入串口来查看评估板的启动了。需要使用 usb 数据线连接评估板的调试串口。然后再使用上面的串口软件 mobaXterm 选择正确的串口号和波特率 115200 即可。接下来我们就可以进入第三章的功能测试内容了。

ECB33 登录评估板的用户名和密码都是 root。

2.2.3. 烧录 SD 卡

这一节是对烧录功能的补充，用户可以烧录到 SD 卡来启动评估板，这里我们需要使用 PhoenixCard 来进行 SD 卡的烧录。



用户请依次选择待烧写固件，选择和烧录到 emmc 同样的固件。选择卡的种类为“启动卡”， 而后选择待烧写的 SD 卡设备， 该软件会自动扫描可用的移动存储设备， 请注意不要选择了错误的设备。 最后点击“烧卡” 按钮， 烧卡过程中会实时显示当前进度。

3. 功能测试

3.1.核心资源

在 Linux 系统中，提供了 proc 虚拟文件系统来查询各项核心资源的参数以及一些通用工具来评估资源的性能。下面将具体对 CPU，memory，eMMC，RTC 等核心资源的参数进行读取与测试。

3.1.1. CPU

3.1.1.1. 查看 CPU 信息

使用 cat /proc/cpuinfo 查看 CPU 信息。ECB33 配备 Arm® Cortex®-A55*4 处理器， 最高主频可达 1.5GHz。

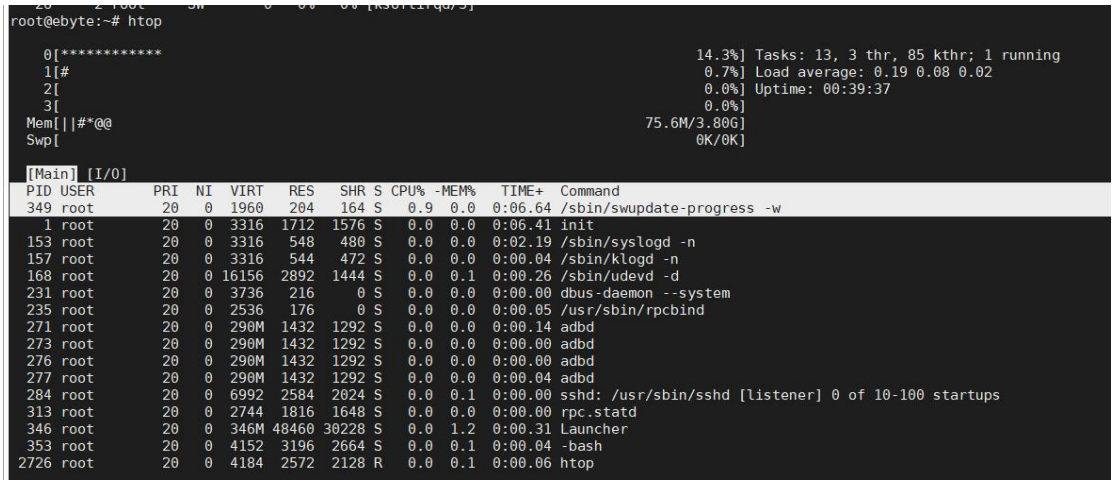
```
root@ebyte:~# cat /proc/cpuinfo
processor       : 0
BogoMIPS        : 48.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp cpuid
asimdrdm lrcpc dcpop asimddp
CPU implementer : 0x41
CPU architecture: 8
CPU variant     : 0x2
CPU part        : 0xd05
CPU revision    : 0

processor       : 1
BogoMIPS        : 48.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp cpuid
asimdrdm lrcpc dcpop asimddp
CPU implementer : 0x41
CPU architecture: 8
CPU variant     : 0x2
CPU part        : 0xd05
CPU revision    : 0

processor       : 2
BogoMIPS        : 48.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp cpuid
asimdrdm lrcpc dcpop asimddp
CPU implementer : 0x41
CPU architecture: 8
CPU variant     : 0x2
CPU part        : 0xd05
CPU revision    : 0

processor       : 3
BogoMIPS        : 48.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp cpuid
asimdrdm lrcpc dcpop asimddp
CPU implementer : 0x41
CPU architecture: 8
CPU variant     : 0x2
CPU part        : 0xd05
CPU revision    : 0
```

processor：系统中逻辑处理核的编号，对于多核处理器则可以是物理核、或者使用超线程技术虚拟的逻辑核

model name：CPU 属于的名字及其编号

BogoMIPS：在系统内核启动时粗略测算的 CPU 每秒运行百万条指令数（MillionInstructions Per Second）

3.1.1.2. 查看 CPU 使用率



3.1.1.3. 查看 CPU 温度信息

通过 cat /sys/class/thermal/thermal_zone0/temp 来查看 CPU 的内部温度。

```
#cat /sys/class/thermal/thermal_zone0/temp
39061
```

上面的数值除以 1000 就是正确的温度值，单位为摄氏度。

3.1.2. 内存

3.1.2.1. 查看内存信息

通过 cat /proc/meminfo 可以查看内存信息。

```
root@ebyte:~# cat /proc/meminfo
MemTotal:        3988940 kB
MemFree:         3843356 kB
MemAvailable:    3865152 kB
Buffers:            2928 kB
Cached:            55312 kB
SwapCached:            0 kB
Active:            13084 kB
```

MemTotal: 总内存量。这里显示的值为 449616 kB，表示系统总共有大约 449 MB 的物理内存可用。

MemFree: 空闲内存量。这里显示的值为 370472 kB，表示当前系统中有大约 370 MB

的内存是空闲的，未被使用。

MemAvailable: 可用内存量。这里显示的值为 407640 kB，表示当前可供系统使用的内存总量，包括已缓存的内存和可用的内存。

Buffers: 缓冲区使用量。这里显示的值为 0 kB，表示系统当前没有使用任何内存作为缓冲区。

Cached: 缓存的内存量。这里显示的值为 39208 kB，表示系统当前用于缓存的内存量。

SwapCached: 交换缓存的内存量。这里显示的值为 0 kB，表示当前没有被缓存到交换空间中的内存。

Active: 活跃的内存量。这里显示的值为 13408 kB，表示当前正在使用的内存量。

3.1.2.2. 内存压力测试

通过给定测试内存的大小和次数, 可以对系统现有的内存进行压力上的测试。可使用系统工具 memtester 进行测试，如指定内存大小 10MB，测试次数为 1，测试命令为“memtester 10M 10"。

```
#memtester 10M 1
memtester version 4.3.0 (32-bit)
Copyright (C) 2001-2012 Charles Cazabon.
Licensed under the GNU General Public License version 2 (only).

pagesize is 4096
pagesizemask is 0xfffff000
want 10MB (10485760 bytes)
got  10MB (10485760 bytes), trying mlock ...locked.
Loop 1/1:
  Stuck Address       : ok
  Random Value        : ok
  Compare XOR         : ok
  Compare SUB         : ok
  Compare MUL         : ok
  Compare DIV         : ok
  Compare OR          : ok
  Compare AND         : ok
  Sequential Increment: ok
  Solid Bits          : ok
  Block Sequential    : ok
  Checkerboard        : ok
  Bit Spread          : ok
  Bit Flip            : ok
  Walking Ones        : ok
  Walking Zeroes      : ok
  8-bit Writes        : ok
  16-bit Writes       : ok

Done.
```

3.1.3. eMMC 测试

查看 emmc 容量

通过 fdisk-l 命令可以查询到 eMMC 分区信息及容量。

```
root@ebyte:~# fdisk -l
Found valid GPT with protective MBR; using GPT

Disk /dev/mmcblk0: 60620800 sectors,  928M
Logical sector size: 512
Disk identifier (GUID): ab6f3888-569a-4926-9668-80941dcb40bc
Partition table holds up to 6 entries
First usable sector is 73728, last usable sector is 60620766

Number  Start (sector)    End (sector)  Size Name
     1           73728          139263 32.0M boot-resource
     2          139264          172031 16.0M env
     3          172032          368639 96.0M boot
     4          368640          401407 16.0M private
     5          401408         2498559 1024M rootfs
     6         2498560        60620766 27.7G UDISK
```

查看 emmc 分区

通过 df 命令可以查询到 eMMC 分区信息，使用情况，挂载目录等信息。

```
root@ebyte:~# df -h
Filesystem                Size  Used Avail Use% Mounted on
/dev/root                 991M  297M  679M  31% /
devtmpfs                  1.9G     0  1.9G   0% /dev
tmpfs                     2.0G  244K  2.0G   1% /tmp
tmpfs                     2.0G  292K  2.0G   1% /run
/dev/by-name/UDISK         28G   16K   28G   1% /mnt/UDISK
```

3.1.4. NPU

ECB33 部分型号内置了全志自行研发的 NPU， 其为轻量级的端侧 AI 计算提供了解决方案。 该 NPU 采用了出色的 VPU（视觉处理单元） 架构， 以便提供高效能的计算速

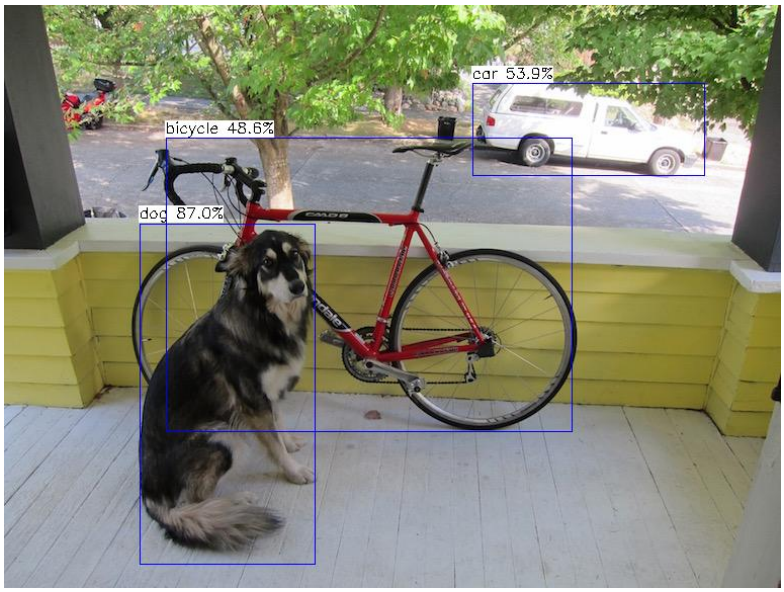
度，可以满足图像识别、 语音识别等任务的需求。

这里我们运行一个 npu 的模型 demo，用于目标检测识别。

```
root@ebyte:/etc/npu/yolov5# pwd
/etc/npu/yolov5
root@ebyte:/etc/npu/yolov5# ./yolov5 model/yolov5.nb input_data/dog.jpg
```

```
  tensor to fp: 95.64 ms.
awnn_run total: 125.69 ms.
yolov5_postprocess.cpp run.
detection num: 3
16:  87%, [ 134,  216,  308,  552], dog
 2:  54%, [ 463,   77,  693,  170], car
 1:  49%, [ 160,  131,  562,  423], bicycle
awnn_destroy total: 8.04 ms.
awnn_uninit total: 8.30 ms.
root@ebyte:/etc/npu/yolov5# ls
input_data  model  result.png  yolov5
```

运行结束之后生成一个 result.png，将其拷贝到虚拟机查看，可以看到识别到了 dog,bicycle 和 car。



3.1.5. RTC

Linux 系统中分系统时钟（软件时钟）和 RTC 时钟（硬件时钟），系统时钟掉电即会消失， RTC 时钟在安装电池的情况下会长期运行。如需使用外部 RTC 时钟， 请将 ML2032（3V 可充）或 CR2032（3V 不可充） 电池安装至 RTC 纽扣电池座。

设置系统时间

将系统时间设置为 2024.07.12 16:35:15

```
# date 071216352024.15
Fri Jul 12 16:35:15 UTC 2024
```

将系统时间写入 RTC

```
# hwclock -w
```

```
# hwclock -r
2000-01-01 07:07:15.965766+0000
```

掉电保持 RTC 时间

将评估板关机断开电源，一段时间后重新上电开机。查看 RTC 时间和系统时间：

```
# hwclock -r
2024-07-12 16:52:42.648561+0000
```

3.1.6. WatchDog

使用测试例程测试看门狗，文件已经被编译好放在 tool 目录下，拷贝到评估板后。运行看门狗应用,超时时间为 4s，每间隔 1s 喂一次狗：

```
# ./watchdog 4 1 0
Starting wdt_driver (timeout: 4, sleep: 1, test: ioctl)
Trying to set timeout value=4 seconds
The actual timeout was set to 4 seconds
Now reading back -- The timeout is 4 seconds
```

如果将上面的 1s 改到大于 4s, 则超过了要求的 4s 喂狗时间，评估板会重启。

3.1.7. 电源管理

本章节演示 Linux 电源管理的 Suspend 功能，让评估板睡眠 ，通过外部事件唤醒。Linux 内核一般提供了三种 Suspend: Freeze 和 STR(Suspend to RAM)，在用户空间向”/sys/power/state”文件分别写入”freeze”“和”mem”，即可触发它们。

查看当前支持的模式

```
# cat /sys/power/state
freeze mem
```

休眠到内存

```
root@ebyte-ubuntu:~# echo "mem" > /sys/power/state
```

3.2.外设接口

3.2.1. GPIO

查看所有 GPIO，列出所有 registered 的 pins。

```
root@ebyte:~# mount -t debugfs none /sys/kernel/debug
root@ebyte:~# cat /sys/kernel/debug/pinctrl/3604000.pinctrl/pinmux-pins
Pinmux settings per pin
Format: pin (name): mux_owner|gpio_owner (strict) hog?
pin 0 (PA0): UNCLAIMED
pin 1 (PA1): UNCLAIMED
pin 2 (PA2): UNCLAIMED
pin 3 (PA3): UNCLAIMED
pin 4 (PA4): UNCLAIMED
pin 5 (PA5): UNCLAIMED
pin 6 (PA6): UNCLAIMED
pin 7 (PA7): UNCLAIMED
pin 8 (PA8): UNCLAIMED
pin 9 (PA9): GPIO 3604000.pinctrl:9
pin 32 (PB0): device 2602000.uart function uart2 group PB0
pin 33 (PB1): device 2602000.uart function uart2 group PB1
pin 34 (PB2): UNCLAIMED
pin 35 (PB3): UNCLAIMED
pin 36 (PB4): device 2608000.uart function uart8 group PB4
pin 37 (PB5): device 2608000.uart function uart8 group PB5
pin 38 (PB6): UNCLAIMED
pin 39 (PB7): device 4541800.can function can1 group PB7
pin 40 (PB8): device 4541800.can function can1 group PB8
pin 41 (PB9): device 2600000.uart function uart0 group PB9
pin 42 (PB10): device 2600000.uart function uart0 group PB10
******
```

导出 GPIO

这里以 ECB33 为例，原理图查看底板的连接插针 J16 第 13 号脚是 PB2。再通过上面的 pins 可以知道 GPIO 序号是 34。我们通过 gpiofs 导出 gpio34，如下所示操作。

```
#cd /sys/class/gpio/
#ls
export        gpiochip0    unexport
#echo 34 > export
#ls
export        gpio34        gpiochip0    unexport
```

设置 GPIO 方向

输入

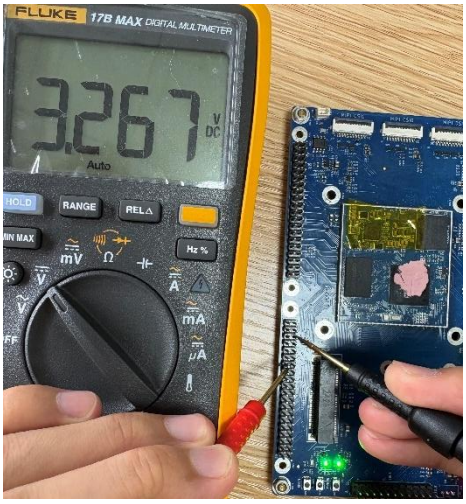
```
#cd gpio34
#ls
active_low                edge                      uevent
device                    power                     value
direction                 subsystem                 waiting_for_supplier
#echo in > direction
```

输出

```
#echo out > direction
```

设置 GPIO 值为 1 之后可以使用万用表进行测量，可以测量到 J16 的 13 号脚为 3.3V。实际测量误差范围±5%。

```
#echo 1 > value
```



注销 GPIO

```
#echo 34 > unexport
]#ls
export          gpiochip0    unexport
```

3.2.2. LED 灯

Linux 系统提供了一个独立的子系统以方便从用户空间操作 LED 设备，该子系统以文件的形式为 LED 设备提供操作接口。这些接口位于/sys/class/leds 目录下。在硬件资源列表中，我们已经列出了评估板上所有的 LED。下面通过命令读写 sysfs 的方式对 LED 进行测试。下述命令均为通用命令，也是操控 LED 的通用方法。

```
root@ebyte:/sys/class/leds# ls
led-Run    led-User
```

点亮和熄灭 LED

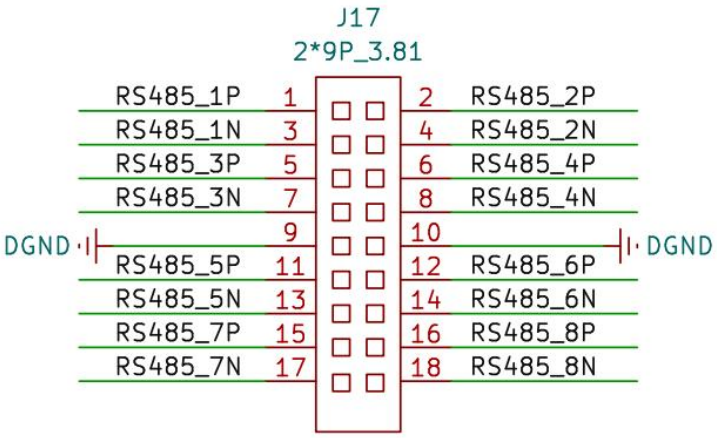
```
# echo 0 > brightness
# echo 1 > brightness
```

查看 LED 触发模式

```
root@ebyte-ubuntu:/sys/class/leds/green:heartbeat# cat trigger
[none] rfkill-any rfkill-none kbd-scrolllock kbd-numlock kbd-capslock kbd-kanalock kbd-shiftlock kbd-altgrlock kbd-ctrllock kbd-altlock kbd-shiftllock kbd-shiftrlock kbd-ctrllock kbd-ctrlrlock timer oneshot heartbeat backlight gpio cpu cpu0 default-on transient flash torch mmc0 mmc1
```

3.2.3. 串口

评估板引出了八路 485 串口供使用，原理图如下。



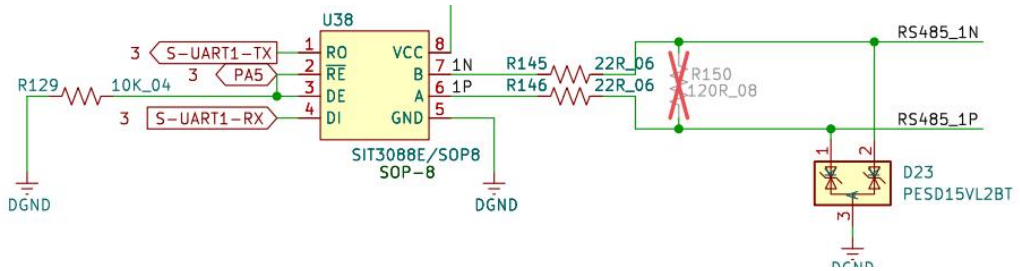
映射到 linux 的设备名称分别如下表所示。

RS485_1	/dev/ttyAS16
RS485_2	/dev/ttyAS15
RS485_3	/dev/ttyAS2
RS485_4	/dev/ttyAS1
RS485_5	/dev/ttyAS4
RS485_6	/dev/ttyAS9
RS485_7	/dev/ttyAS11
RS485_8	/dev/ttyAS12

这里我们测试 RS485_1，先使用一个 USB 的 RS485 自动收发器连接 PC。A+连接 RS485_1P，B-连接 RS485_1N。

收信息

可以看到原理图的 485 控制器 RS485_1P 对应的方向使能引脚是 PA5，并且有拉低，默认为收信息。



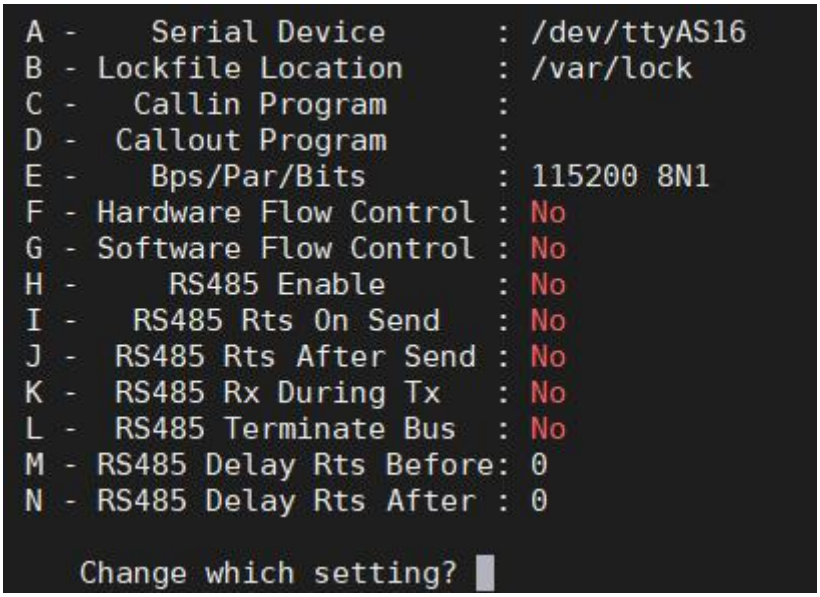
使用 minicom -s 进入串口配置 seria port setup。按 A 输入串口设备/dev/ttyAS16,按 F 关闭硬件流控，按 enter 确认。最后按 esc 选择 exit 到串口收发界面。

minicom -s

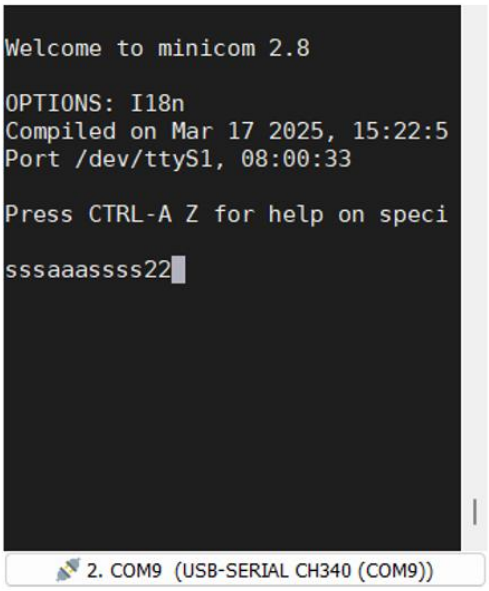
按 A 设置串口号，按回车结束

按 F 设置关闭硬件流控，按回车结束

选择 exit 进入通讯界面。



使用 PC 打开我们的 USB to RS485 串口控制器，发送信息，我们的评估板可以收到信息。

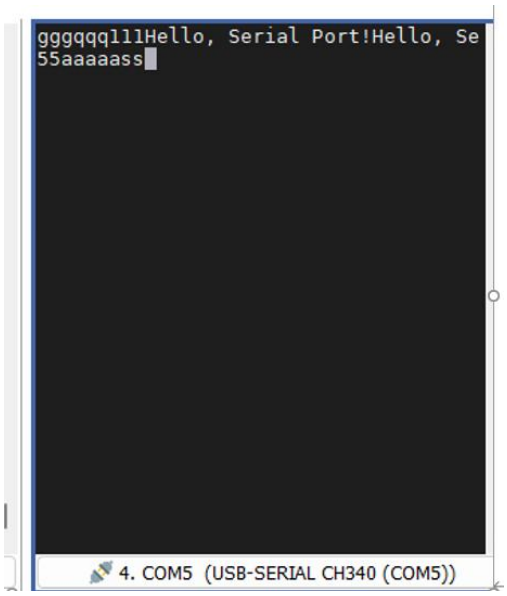


发信息

我们按下 ctrl + A，再按下 X 退出 minicom，将上述的方向使能 PA5 引脚拉高。

```
root@ebyte:/# cd /sys/class/gpio/
root@ebyte:/sys/class/gpio# echo 5 > export
root@ebyte:/sys/class/gpio# cd gpio5
root@ebyte:/sys/class/gpio/gpio5# echo out > direction
root@ebyte:/sys/class/gpio/gpio5# echo 1 > value
root@ebyte:/sys/class/gpio/gpio5# cat value
1
```

然后重复上面打开 minicom 的操作，我们在评估板的终端上输入信息，可以看到 PC 的 485 控制器收到了信息



3.2.4. USB

本节通过相关命令或热插拔、USB HUB 验证 USB Host 驱动的可行性，实现读写 U 盘的功能、usb 枚举功能。

3.2.4.1. 1.插入 U 盘

根据丝印，下面的 type-a 才是 USB，将 U 盘插入下面的口子。

挂载/dev/sda 到/mnt/usb，然后使用 df-h 查找存储设备。

```
root@ebyte:/# mkdir /mnt/usb/ -p
root@ebyte:/# mount /dev/sda1 /mnt/usb/
[   322.468359] FAT-fs (sda1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
root@ebyte:/# df -h
Filesystem                Size  Used Avail Use% Mounted on
/dev/root                 991M  297M  679M  31% /
devtmpfs                  973M     0  973M   0% /dev
tmpfs                     976M  176K  976M   1% /tmp
tmpfs                     976M  312K  976M   1% /run
/dev/by-name/UDISK         14G  248M   14G   2% /mnt/UDISK
/dev/sda1                  29G  4.8G   24G  17% /mnt/usb
```

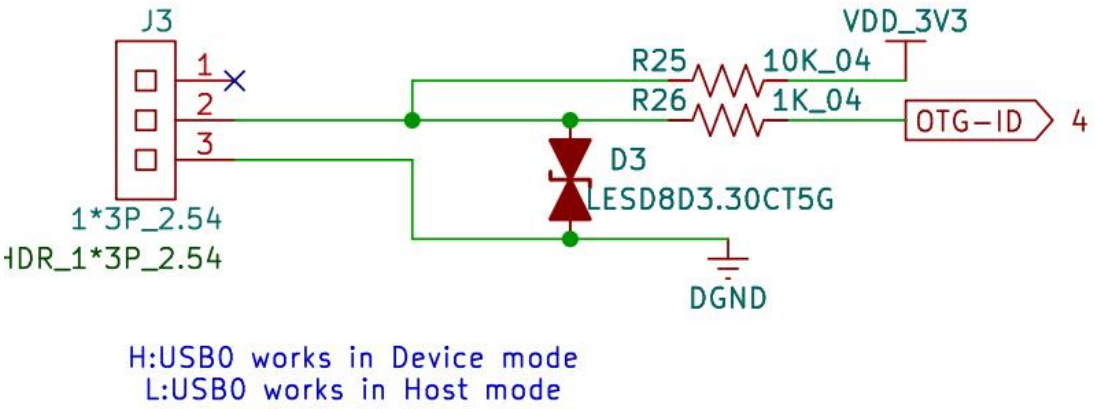
3.2.4.2. U 盘读写测试

经测试，可以正常读写文件。

```
root@ebyte:/# dd if=/dev/random of=/mnt/usb/testfile count=1 bs=500M
1+0 records in
1+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 17.2115 s, 30.5 MB/s
root@ebyte:/# echo 3 > /proc/sys/vm/drop_caches
[   563.506938] bash (353): drop_caches: 3
root@ebyte:/# dd if=/mnt/usb/testfile of=/dev/null count=10 bs=50M
10+0 records in
10+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 30.0336 s, 17.5 MB/s
```

3.2.5. OTG

3.2.5.1. USB HOST 模式测试



跳线短接 2，3 脚，然后在 otg 口插入 U 盘

执行如下命令，可查看到当前为 HOST 模式。

```
#cat /sys/bus/platform/drivers/otg\ manager/soc\@3000000\:usbc0\@0/otg_role
usb_host
```

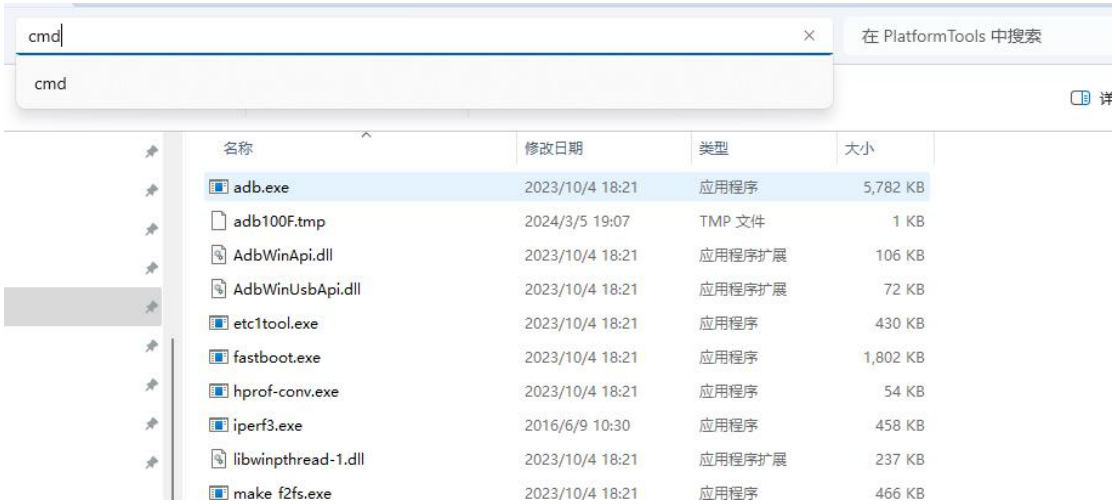
3.2.5.2. USB DEVICE 模式测试

跳线短接 1，2 脚，配置为 device 模式。系统上电启动， OTG 接口默认配置为 DEVICE 模式。

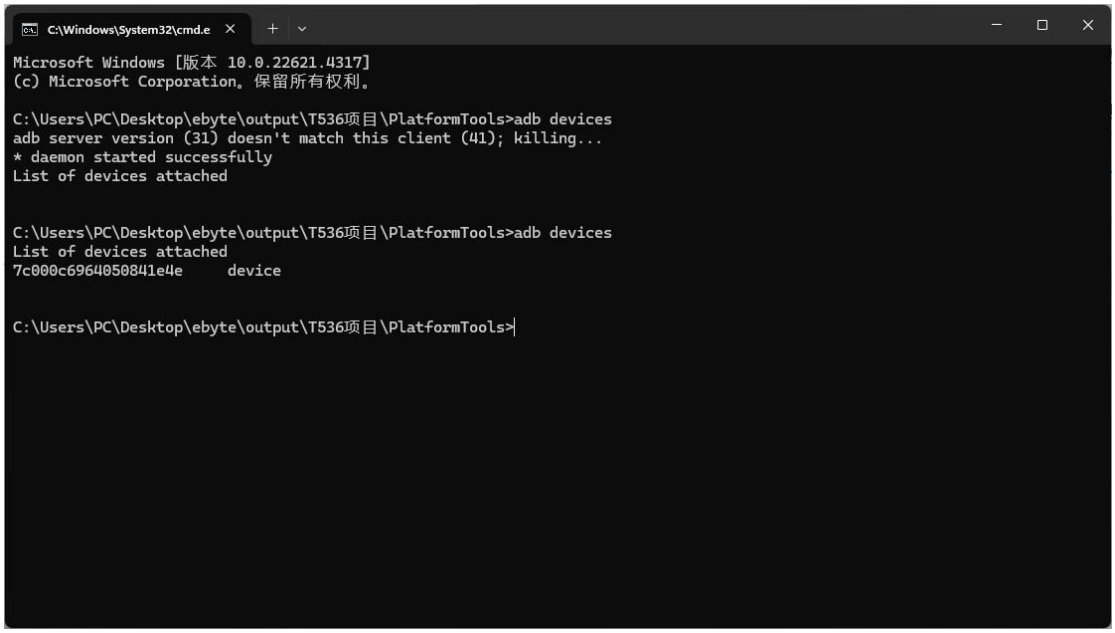
```
#cat /sys/bus/platform/drivers/otg\ manager/soc\@3000000\:usbc0\@0/otg_role
usb_device
```

使用 type-A to typa-A 连接 PC 之后，使用 adb 工具登录评估板。

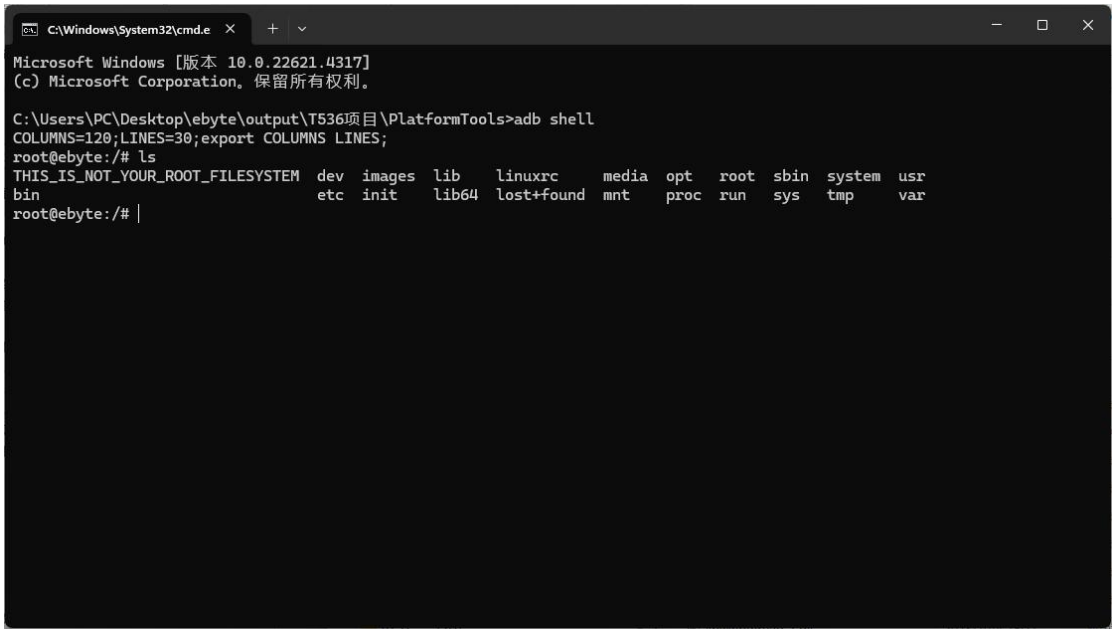
登录之前先关闭烧录工具 phoneix suit。打开 tools 目录下的 PlatformTools 文件夹，然后在上方输入 cmd，再按下回车。



在终端输入 adb devices 来查看列出的设备

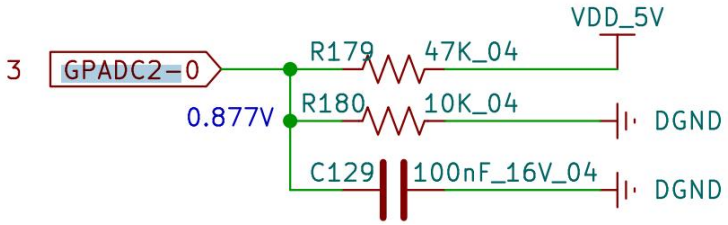


再使用 adb shell 登录到评估板



3.2.6. GPADC

GPADC 是 12bit 采样精度的模数转换模块，我们对 gpadc2 的通道 0 进行测试，这里 5V 经过分压，gpadc2 测量值应该为 0.877。我们使用软件实测读取一下数值。




```
root@ebyte:~# cd /sys/class/gpadc/gpadc_chip2
root@ebyte:/sys/class/gpadc/gpadc_chip2# echo 0 > data
root@ebyte:/sys/class/gpadc/gpadc_chip2# cat data
gpadc2-channel0 voltage data is 877
```

经过实际读取，GPADC2 通道 0 的电压确实是 877mA。

3.2.7. LocalBus

ECB33 提供 1 路 16 位 LocalBus 接口，lbc 提供了测试、配置节点，节点所在路径如下：

```
/sys/devices/platform/soc@3000000/2810000.lbc-controller/lbc/lbc
```

clk_freq：查看/设置当前 lbc 频率

transfer_mode: 查看/设置传输模式：

0：直接访问模式

1.：中断模式

2.：DMA 模式

transfer_width：查看/设置传输宽度

0：8bit 位宽

1：16bit 位宽

2：32bit 位宽

message: 查看当前 lbc 信息

```
cd /sys/devices/platform/soc\@3000000/2810000.lbc-controller/lbc/lbc/
echo 10000000 > clk_freq
echo 2 > transfer_mode
echo 0 > transfer_width
echo 7 > burst_mode
lbc_dma_memtester
```

这里我们使用 10M，DMA 模式，位宽为 8bit，设置为地址自增模式。信号线连接 FPGA 进行测试。

```
lbc_dma_memtester s/platform/soc@3000000/2810000.lbc-controller/lbc/lbc]#
local dma loop 0 :  ok
local dma loop 1 :  ok
local dma loop 2 :  ok
local dma loop 3 :  ok
local dma loop 4 :  ok
local dma loop 5 :  ok
local dma loop 6 :  ok
local dma loop 7 :  ok
local dma loop 8 :  ok
local dma loop 9 :  ok
local dma loop 10 :  ok
local dma loop 11 :  ok
local dma loop 12 :  ok
local dma loop 13 :  ok
local dma loop 14 :  ok
local dma loop 15 :  ok
local dma loop 16 :  ok
```

3.2.8. Mini PCIE

ECB33 有 1 路 MiniPCIe 插座，支持 PCIex1 GEN2，我们插入一张 SSD 来进行测试。

插入后系统自动识别到/dev/nvme0n1，我们需要做的就是为这个盘创建分区，然后格式化一下，最后挂载到目录就可以使用了。

首先，打开 fdisk 工具，选择设备进行分区操作

```
#fdisk /dev/nvme0n1
```

1.查看现有分区：

输入 p 显示现有分区表。

2.删除现有分区（如果需要）：

如果 /dev/nvme0n1p1 已经有分区，而你想删除并重新创建它，可以输入 d 删除现有分区。

3.创建新分区：

输入 n 创建新分区。然后选择：

分区类型：一般选择 primary（通常是默认选项）。

分区编号：选择分区编号（如果是第一个分区，通常选择 1）。

起始扇区：按回车使用默认值（通常是第一个可用扇区）。

结束扇区：按回车使用默认值，或者设置大小（例如 +20G 创建一个 20GB 的分区）。

4.保存并退出：

输入 w 保存分区表并退出 fdisk。

5.格式化分区，创建挂载点并挂载

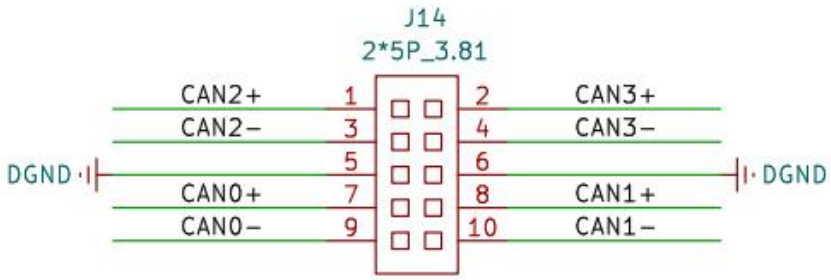
```
# mkfs.ext4 /dev/nvme0n1p1

# mkdir -p /mnt/nvme

# mount /dev/nvme0n1p1 /mnt/nvme
```

最后查看下分区，可以看到挂载了一个 117G 的 nvme 硬盘了。

```
root@ebyte:/# df -h
Filesystem          Size  Used Avail Use% Mounted on
/dev/root           991M  291M  685M  30% /
devtmpfs            973M     0  973M   0% /dev
tmpfs               976M  248K  976M   1% /tmp
tmpfs               976M  312K  976M   1% /run
/dev/by-name/UDISK   14G  8.0K   14G   1% /mnt/UDISK
/dev/nvme0n1p1      117G   24K  111G   1% /mnt/nvme
```

3.2.9. CAN

ECB33 有四路 CAN。我们可以使用 CAN 盒连接 PC 进行收发测试，也可以将两路 CAN 的 CAN-H 和 CAN-L 分别短接来进行测试。本次测试使用亿佰特的 ECAN-U01S 测试 CAN0。can H 连接 J14 的 7 脚，canL 连接 J14 的 9 脚。



我们使用 ifconfig -a 可以看到有 can0，can1，can2 和 can3。

```
root@ebyte:/# ifconfig -a
can0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          NOARP  MTU:16  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:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:56

can1      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          NOARP  MTU:16  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:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:59

can2      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          NOARP  MTU:16  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:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:62

can3      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          NOARP  MTU:16  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:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:65
```

打开 can0，并设置 can0 的 can 设备通信波特率为 1000000， 也就是通信波特率 1MBit/s。

```
ip link set can0 up type can bitrate 1000000
```

PC 打开 ECAN-U01S 配套的 CAN 收发软件 ECAN-Tools，如下所示。

评估板发送 can 数据：

```
cansend can0 123#01.02.03.04.05.06.07.08
```

评估板接收 can 数据：

```
candump -ta can0 &
```

3.2.10. SD 卡

本小节使用 SanDisk 公司、 32GB 容量的 Micro SD 卡来测试评估板 Micro SD 接口性能。请参考《Linux 系统启动卡制作及系统固化》文档将其制作成 Linux 系统启动卡，再进行测试。不同的 Micro SD 卡以及不同的测试方法，对 Micro SD 接口测试结果将造成一定差异。

请将 Linux 系统启动卡插至评估板 Micro SD 卡槽，评估板上电，进入评估板文件系统执行如下命令查看 Linux 系统启动卡信息。

3.2.10.1. 查看 TF 卡容量

通过 fdisk -l 命令可以查询到 TF 卡分区信息及容量：

```
#fdisk -l
Found valid GPT with protective MBR; using GPT
Disk /dev/mmcblk0: 62333952 sectors, 1764M
Logical sector size: 512
Disk identifier (GUID): ab6f3888-569a-4926-9668-80941dcb40bc
Partition table holds up to 12 entries
First usable sector is 73728, last usable sector is 62333947
Number  Start (sector)    End (sector)  Size Name
     1           73728          108165 16.8M boot-resource
     2          108166          110213 1024K env
     3          110214          112261 1024K env-redund
     4          112262          147461 17.1M boot
     5          147462         2244613 1024M rootfs
     6         2244614         2246661 1024K dsp0
     7         2246662         2248709 1024K riscv0
     8         2248710         2281477 16.0M private
     9         2281478        62331903 28.6G UDISK
```

3.2.10.2. TF 卡的性能测试

性能测试主要测试 eMMC 在 linux 系统下对文件的读写速度，一般结合 time 与 dd 双命令进行测试。挂载需要测试的 TF 卡分区，这里以最后一个分区/dev/mmcblk0p9 为例，挂载目录为/run/meida/mmcblk0p9。

执行如下命令，创建目录并对分区进行挂载。

```
mkdir -p /run/media/mmcblk0p9
mount /dev/mmcblk0p9 /run/media/mmcblk0p9
```

Micro SD 接口写速度测试

进入评估板系统， 执行如下命令测试 Micro SD 接口写速度

```
echo 3 > /proc/sys/vm/drop_caches
time dd if=/dev/zero of=/run/media/mmcblk0p9/test bs=1024K count=100 conv=fsync


100+0 records in
100+0 records out


real    0m6.473s
user    0m0.000s
sys     0m1.633s
```

time 命令有计时作用， dd 用于复制，从 if(input file)文件读出，写到 of(output file)指定的文件， bs 是每次写块的大小， count 是读写块的数量。

"if=/dev/zero"不产生 IO，即不断输出数据，可用来测试纯写速度。

此处一共写 100MByte 测试数据至 Linux 系统启动卡的 test 文件，可看到本次测试的 Micro SD 接口写速度约为： 100MByte / 6.47s ≈ 15.45MB/s。

Micro SD 接口读速度测试

执行如下命令测试 Micro SD 接口读速度。

```
echo 3 > /proc/sys/vm/drop_caches
time dd if=/run/media/mmcblk0p9/test of=/dev/null bs=1024K count=100
```

"of=/dev/null"不产生 IO，即不断接收数据，可用来测试纯读速度。

此处从 test 文件一共读出 100MByte 的数据，可看到本次测试的 Micro SD 接口读速度约为： 100MByte / 4.47s ≈ 22.40MB/s。

测试完成后， 执行如下命令卸载挂载分区。

```
umount /run/media/mmcblk0p9/
rm -r /run/media/mmcblk0p9
```

3.2.11. 显示

本模块由显示引擎（DE）和各类型控制器（tcon）组成。输入图层（layers）在 DE 中进行显示相关处理后，通过一种或多种接口输出到显示设备上显示，以达到将众多应用渲染的 图层合成后在显示器呈现给用户观看的作用。 DE 有 2 个独立单元（可以简称 de0、de1），可以 分别接受用户输入的图层进行合成，输出到不同的显示器，以实现双显。 DE 的每个独立的单元有 1-4 个通道（典型地， de0 有 4 上， de1 有 2 个），每个通道可以同时处理接收 4 个格式相同的 图层。 sunxi 平台有视频通道和 UI 通道之分。视频通道功能强大，可以支持 YUV 格式和 RGB 图层。 UI 通道只支持 RGB 图层。

3.2.11.1.5.5 英寸 MIPI 显示

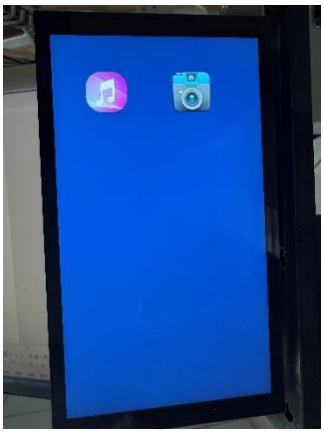
请将 5.5 英寸 MIPI 显示屏（型号： 亿佰特 ECA11-5P5LCDMIPI1019CT-C，分辨率：1080x1920）与评估板的 DSI（显示）接口正确连接。

注意： 请务必使用我司配套的 7 英寸 MIPI 显示屏、 FFC 软排线，并按照如下方法进行硬件连接。 若采用第三方配件，需仔细核对评估板接口、 FFC 软排线、 MIPI 显示屏三者线序， 否则可能烧毁 MIPI LCD 屏。

请将 26pin FFC 软排线连接 MIPI LCD 屏至评估板 MIPI LCD 接口。

评估板重启后， MIPI 显示屏可观察到启动 logo。

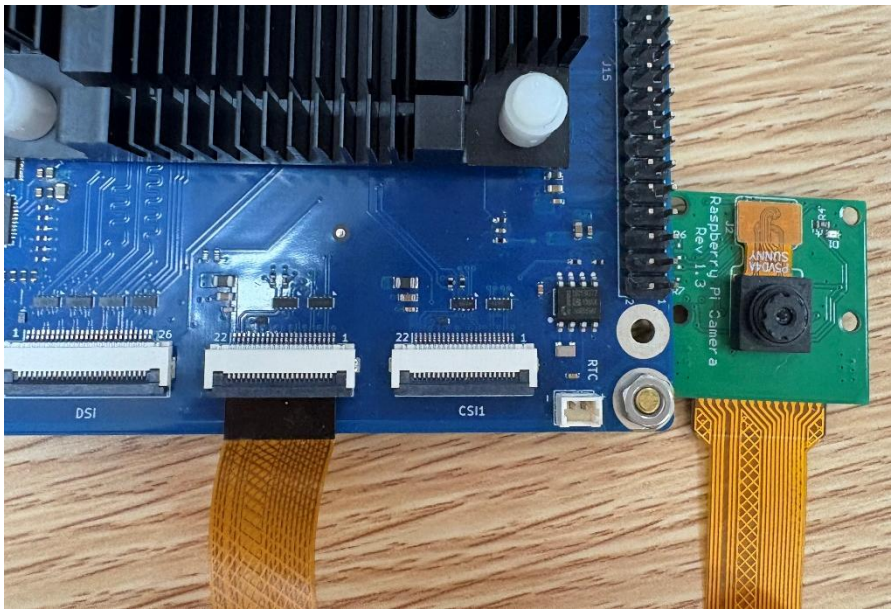
进入系统之后，屏幕将会出现 QT 界面，单击两个图标分别会出现不同的打印信息。



```
root@ebyte:~# menu:on_camera_home_btn_clicked
sh: line 1: /sys/class/disp/disp/attr/fb2ui: No such file or directory
sh: line 1: /sys/inputFocus/write: No such file or directory
menu:on_camera_home_btn_clicked
sh: line 1: /sys/class/disp/disp/attr/fb2ui: No such file or directory
sh: line 1: /sys/inputFocus/write: No such file or directory
menu:on_media_btn_clicked
sh: line 1: /sys/class/disp/disp/attr/fb2ui: No such file or directory
sh: line 1: /sys/inputFocus/write: No such file or directory
menu:on_media_btn_clicked
sh: line 1: /sys/class/disp/disp/attr/fb2ui: No such file or directory
sh: line 1: /sys/inputFocus/write: No such file or directory
```

3.2.12. 摄像头

ECB33 支持 MIPI-CSI 接口，本司以树莓派 P5V04A 摄像头为例，该摄像头使用的芯片为 ov5647。我们使用 CSI0 接口连接摄像头进行测试。



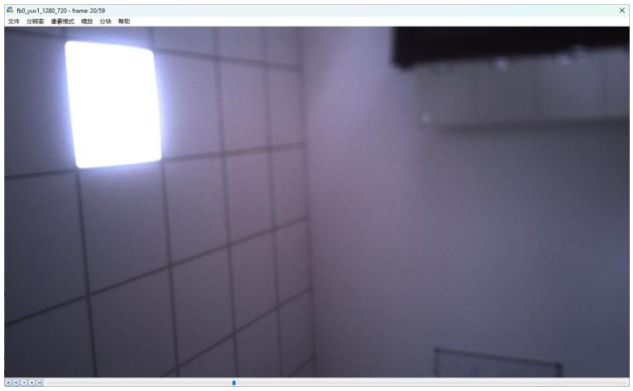
进入系统后，执行如下命令挂载驱动，然后使用官方测试程序测试图像采集。

```
root@ebyte:~# modprobe vin_v4l2
root@ebyte:~# ls /dev/video*
/dev/video0    /dev/video4
root@ebyte:~# ./csi_test_mplane 0 0 1280 720 /root/ 1 60 20
open /dev/video0 fd = 3
resolution got from sensor = 1280*720 num_planes = 3
VIDIOC_STREAMON ok
VIDIOC_STREAMOFF ok
mode 1 test done at the 0 time!!
time cost 2.094031(s)
```

上面命令存放在`bsp/drivers/vin/vin_test/mplane_image`目录中，需要手动拷贝到评估板中，上面命令的含义如下：

1. 测试程序
2. /dev/video0
3. set_input 的 index 参数，默认 0 即可
4. 配置的图像宽度，此处为 1280
5. 配置的图像高度，此处为 720
6. 采集后图像存储路径
7. 采集的图像格式，1 表示 YUV420M，更多格式查看命令所在路径的源码。
8. 采集的帧数
9. 采集的帧率

采集完成后会在对应路径生成一个 bin 文件，其本质为一个 yuv420m 的视频图像，直接修改后缀名字为 yuv 后可以使用随附的工具 yuvplayer.exe 播放查看，注意需要配置 size 和 color 对应图像格式。



3.3.网络接口

3.3.1. Ethernet

使用 net-tools 工具包中的 ifconfig 对网络进行手动配置，首先通过通过 ifconfig 命令查看网络设备信息如下。ECB33 有两个网口，分别是 eth0 和 eth1。我们先连接 eth0 进行测试。

```
eth0      Link encap:Ethernet  HWaddr 66:7D:3B:66:D2:26
          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 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:142

eth1      Link encap:Ethernet  HWaddr 1E:33:99:53:81:CB
          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 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:143 Base address:0x8000
```

配置好以太网连接之后就可以使用 PING 对网络连接进行简单的测试，ping 同一网段的虚拟机测试。

```
root@ebyte:~# ifconfig eth0 192.168.20.76 up
root@ebyte:~# ifconfig eth1 192.168.20.77 up
root@ebyte:~# ping 192.168.20.66 -I eth0
PING 192.168.20.66 (192.168.20.66): 56 data bytes
64 bytes from 192.168.20.66: seq=0 ttl=64 time=11.756 ms
64 bytes from 192.168.20.66: seq=1 ttl=64 time=2.860 ms
^C
--- 192.168.20.66 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 2.860/7.308/11.756 ms
```

将网线连接到 eth1 之后

```
root@ebyte:~# ping 192.168.20.66 -I eth1
PING 192.168.20.66 (192.168.20.66): 56 data bytes
64 bytes from 192.168.20.66: seq=0 ttl=64 time=11.756 ms
64 bytes from 192.168.20.66: seq=1 ttl=64 time=2.860 ms
^C
--- 192.168.20.66 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 2.860/7.308/11.756 ms
```

3.3.1.1. Iperf3

iperf3 是在 IP 网络上主动测量最大可实现带宽的工具。 它支持调节测试时间、缓冲区大小和协议(IPV4 和 IPV6 下的 TCP、 UDP、 SCTP)等各种参数。 iperf3 按角色可以分 为服务端模式或客户端模式，我们可以用它来测试和查看 TCP 模式下的网络带宽，TCP 窗口值，重传的概率等，也可以测试指定 UDP 带宽下丢包率，延迟和抖动情况。

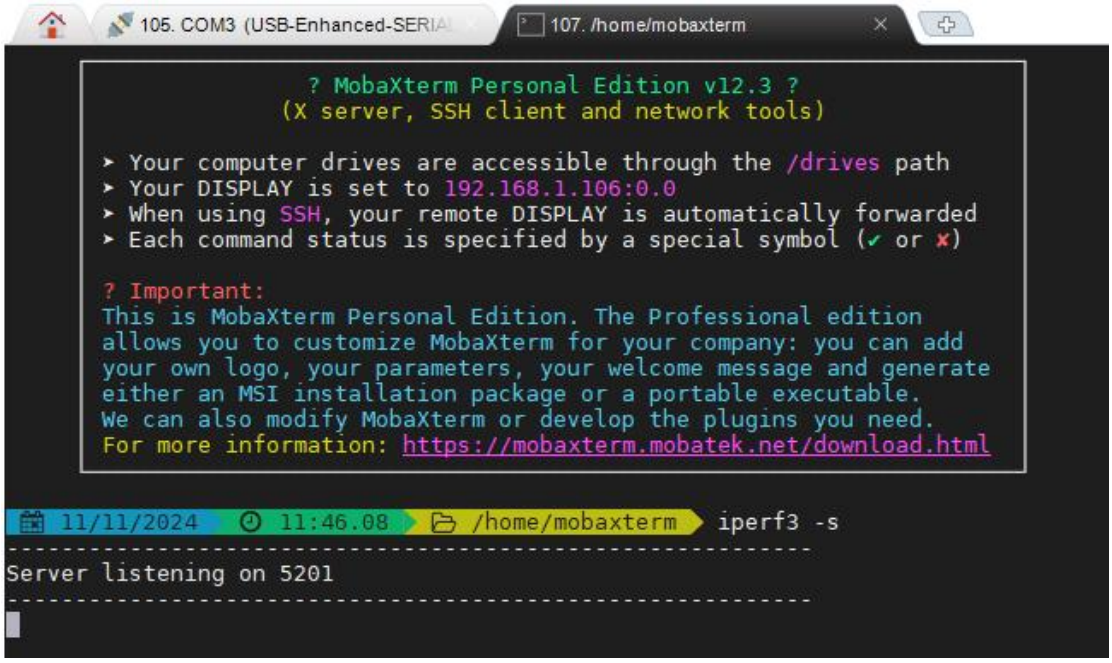
我们在开发主机上打开 MobaXterm，带千兆网卡的主机作为 iperf3 的服务端，被测试的评估板作为客户端分别测试评估板网卡 TCP 和 UDP 的性能。

将服务器和评估板通过 CAT6 网线直连，并配置好各自的 IP 地址。例如我们设置服务器 ip 为 192.168.1.106，设评估板 IP 为 192.168.1.250, 并使用 ping 命令测试确保它们之间是连通的。

注意：尽量不要连接路由器或交换机，以免测试结果受到中间设备传输转发的影响。

测试 TCP 性能

服务器上 iperf3 使用-s 参数表示工作在服务端模式。



评估板上运行的 iperf3 程序工作在客户端， TCP 模式，其中参数说明如下：

➢ -c 192.168.0.102：工作在客户端，连接服务端 192.168.1.106

➢ -i 2：测试结果报告时间间隔为 2 秒

➢ -t 10：总测试时长为 10 秒

```
#iperf3 -c 192.168.1.106 -i 2 -t 10
Connecting to host 192.168.1.106, port 5201
[  5] local 192.168.1.250 port 58936 connected to 192.168.1.106 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-2.00   sec   215 MBytes   900 Mbits/sec    0    754 KBytes
[  5]   2.00-4.00   sec   216 MBytes   906 Mbits/sec    0    754 KBytes
[  5]   4.00-6.00   sec   202 MBytes   851 Mbits/sec    0    847 KBytes
[  5]   6.00-8.00   sec   198 MBytes   828 Mbits/sec    0    936 KBytes
[  5]   8.00-10.00  sec   216 MBytes   907 Mbits/sec    0    936 KBytes
- - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-10.00  sec  1.02 GBytes   878 Mbits/sec    0             sender
[  5]   0.00-10.00  sec  1.02 GBytes   876 Mbits/sec                  receiver

iperf Done.
```

客户端经过 10 秒之后测试结束并显示上面的测试结果，表明 TCP 带宽为 900 Mbits 左右。

测试 UDP 性能

服务器上继续运行 iperf3 使用-s 参数表示工作在服务端模式。

设备上 iperf3 工作在客户端， UDP 模式，其中参数说明如下：

- -u：工作在 UDP 模式
- -c 192.168.1.106：工作在客户端，连接服务端 192.168.0.106
- -i 2：测试结果报告时间间隔为 2 秒
- -t 10：总测试时长为 10 秒
- -b 100M：设定 UDP 传输带宽为 100Mbps.

```
#iperf3 -c 192.168.1.106 -u -i 2 -t 10 -b 100M
Connecting to host 192.168.1.106, port 5201
[  5] local 192.168.1.250 port 58311 connected to 192.168.1.106 port 5201
[ ID] Interval           Transfer     Bitrate         Total Datagrams
[  5]   0.00-2.00   sec  1.50 MBytes  6.28 Mbits/sec  1085
[  5]   2.00-4.00   sec  80.6 KBytes  330 Kbits/sec   57
[  5]   4.00-6.00   sec   322 KBytes  1.32 Mbits/sec  228
[  5]   6.00-8.00   sec   161 KBytes   660 Kbits/sec  114
[  5]   8.00-10.00  sec   242 KBytes   990 Kbits/sec  171
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Jitter    Lost/Total Datagrams
[  5]   0.00-10.00  sec  2.29 MBytes  1.92 Mbits/sec  0.000 ms  0/1655 (0%)  sender
[  5]   0.00-10.00  sec  2.29 MBytes  1.92 Mbits/sec  0.099 ms  0/1655 (0%)
receiver
```

客户端经过 10 秒之后测试结束并显示上面的测试结果，表明 UDP 在指定带宽为 100Mbps 时没有丢包。

3.3.1.2. 访问外网

在接好网线后，手动配置 DNS 和网关即可访问外网。或者通过 dhcp 自动配置网络来访问外网。

我们的路由器分配的网段是 192.168.20.*，所以我们手动配置网络如下。

```
#ifconfig eth0 192.168.20.7 up
#echo "nameserver 114.114.114.114" > /etc/resolv.conf #添加 dns
#ip route add default via 192.168.20.1     #添加网关
```

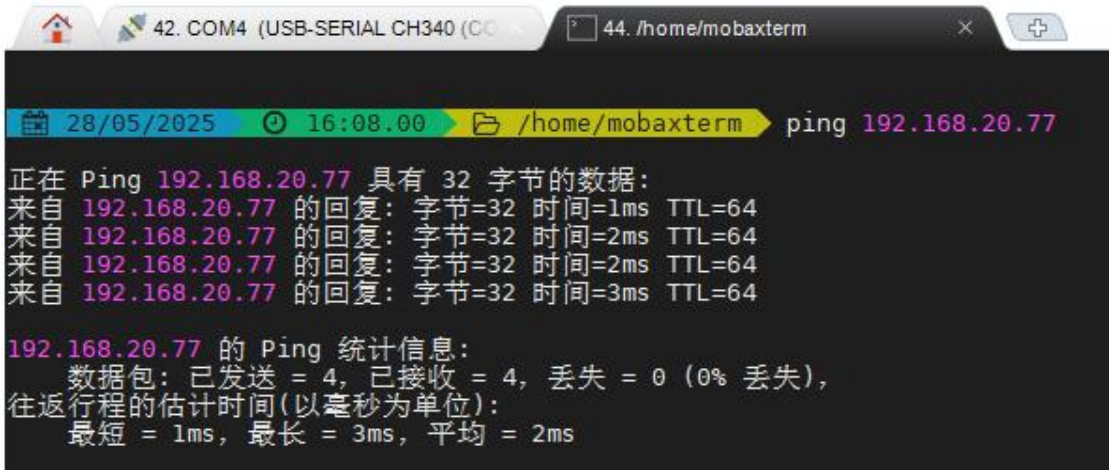
自动配置网络只需要 udhcpc 命令。

```
# udhcpc -i eth0
```

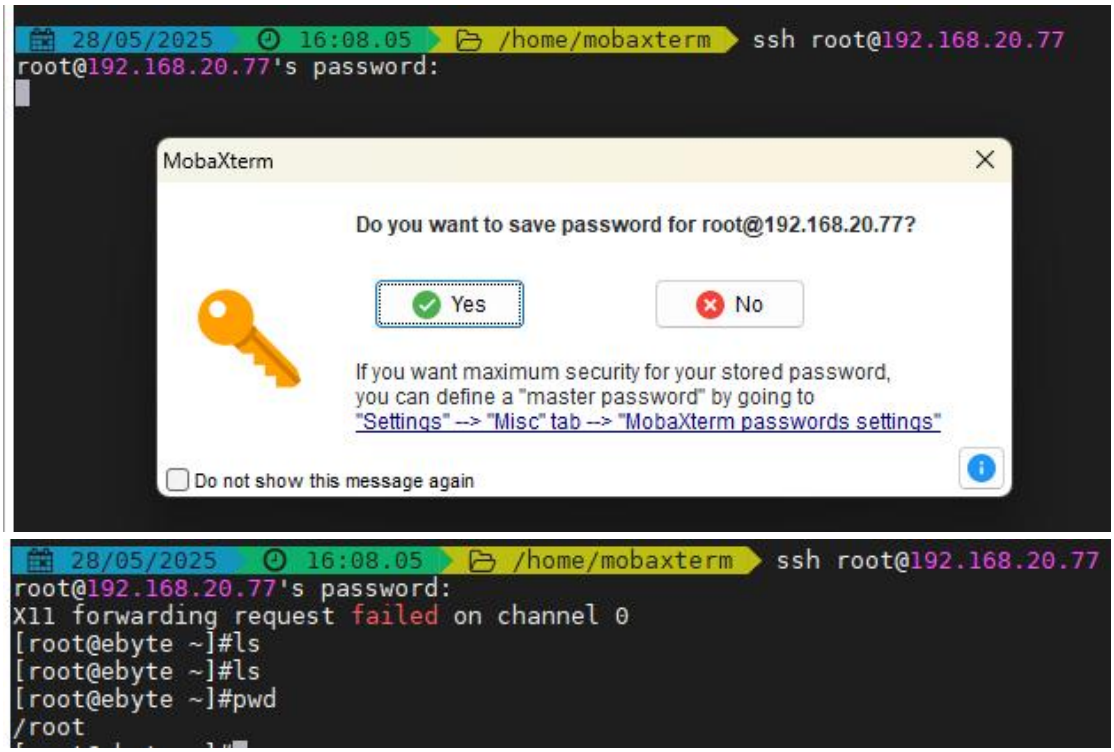
3.3.1.3. SSH 登录评估板

可以使用 SSH 登录评估板，前提条件是需要登录的机器和评估板在同一网段，可以互相 PING 通。可以使用 mobaExterm 提供的 SSH 服务，也可以使用 cmd 命令行的自带的 SSH。mobaExterm 更方便传输文件，我们以 mobaExterm 为例。

首先保证主机可以 ping 通评估板。192.168.20.77 是评估板设置的 IP 地址。



SSH 登录，评估板默认用户名和密码都是 root。



如果需要传输文件，直接使用拖动想要传输的文件到左侧的 sftp 服务栏即可。

3.3.1.4. SCP 传输文件

如果想使用虚拟机和评估版互传文件，使用 scp 拷贝文件即可。

前提是虚拟机和评估板都开启 ssh 服务，并且可以互相 ping 通。

虚拟机拷贝文件到评估板

```
hu@hu:~/linux/temp$ scp dog.jpg root@192.168.20.33:/root
The authenticity of host '192.168.20.33 (192.168.20.33)' can't be established.
ED25519 key fingerprint is SHA256:c2iSpbtR9F/KN7t/bPNZeIxgw4cFoV03nR4+/TA+9gk.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.20.33' (ED25519) to the list of known hosts.
root@192.168.20.33's password:
dog.jpg                                       100%  160KB   2.9MB/s   00:00
```

可以在评估板的 root 目录看到文件

```
[root@ebyte /root]#ls
dog.jpg
```

如果需要拷贝文件夹，在 scp 后面加上-r 参数即可，例如:

```
#scp -r test-dir root@192.168.20.33:/root
```

评估板拷贝文件到虚拟机

```
[root@ebyte /root]#echo helloworld > a.file
[root@ebyte /root]#scp a.file hu@192.168.20.140:/home/hu/linux/temp
The authenticity of host '192.168.20.140 (192.168.20.140)' can't be established.
ECDSA key fingerprint is SHA256:FcbovPE9fo9FD1LVzqRHVpNAtfDeLaSK/a6NmQwyIOQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.20.140' (ECDSA) to the list of known hosts.
hu@192.168.20.140's password:
```

可以在虚拟机的对应目录看到文件

```
hu@hu:~/linux/temp$ ls a.file
a.file
```

3.3.1.5. NFS 共享目录

如果想让评估版直接访问虚拟机的目录，就可以使用 NFS 共享目录了，这样可以在使用大文件时更方便让评估版使用，或者开发时更加的方便。

首先需要搭建虚拟机的 NFS 服务器。安装 NFS 服务后配置文件夹位置，最后重启 NFS 服务即可。

```
#sudo apt-get install nfs-kernel-server rpcbind
#sudo vi /etc/exports     #文件最后一行添加配置内容


/home/hu/linux/nfs *(rw,sync,no_root_squash)
```

最后重启 NFS 服务。

```
#sudo /etc/init.d/nfs-kernel-server restart
```

需要保证评估版可以 ping 通虚拟机。然后创建一个目录，用来挂载 NFS 服务器即可。

```
#mkdir /mnt/nfs
#mount -t nfs 192.168.20.140:/home/hu/linux/nfs /mnt/nfs
```

3.4.音频接口

准备一个耳机插头，左右声道分别和 LINEOUTLP 和 LINEOUTLN 连接，再连接一个 GND。

准备一个耳机，连接耳机插头。

播放

```
# aplay /etc/44100.wav
```

4. 参考资料

- Linux kernel 开源社区：

  https://www.kernel.org/

- 全志 SDK 模块开发指南

5. 修订说明

修订说明表

版本	修改内容	修改时间	编制	校对	审批
V1.0	初稿	25-06-25	HSL	WYQ	WFX

6.关于我们

销售热线：4000-330-990
技术支持：support@cdebyte.com 官方网站：https://www.ebyte.com
公司地址：四川省成都市高新西区西区大道 199 号 B5 栋

成都亿佰特电子科技有限公司
Chengdu Ebyte Electronic Technology Co.,Ltd.