软件应用指南

ECB10-PGD

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

本文主要介绍基于亿佰特核心板定制一个完整的嵌入式 Linux 系统的完整流程，其中包括开发环境的准备，代码的获取，以及如何进行 Bootloader, Kernel 的移植，定制适合自身应用需求的 Rootfs 等。我们首先介绍如何基于我们提供的源代码构建适用于 ECB10 开发板的系统镜像，如何将构建好的镜像烧录到开发板。针对那些基于 ECK10 核心板进行项目开发的用户，我们重点介绍了将这一套系统移植到用户的硬件平台上的方法和一些要点，并通过一些实际的移植案例和 Rootfs 定制的案例，使用户能够迅速定制适合自己硬件的系统镜像。

1.1. 软件资源

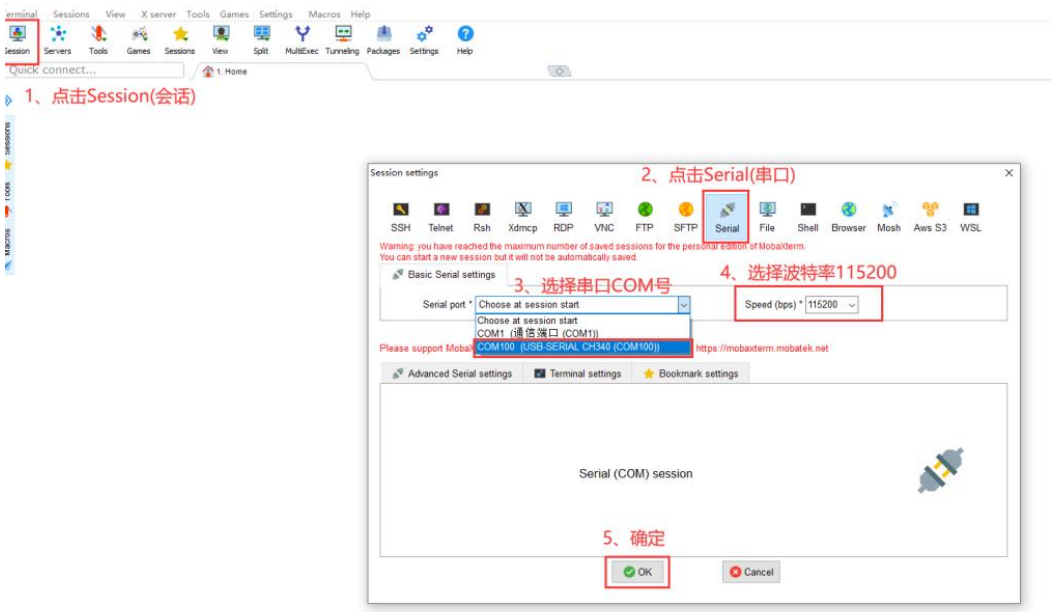
ECB10-TB13X 搭载基于 Linux 5.15 版本内核的操作系统，开发板出厂附带嵌入式 Linux 系统开发所需要的交叉编译工具链，TF-A 源代码，Optee-os 源码，U-boot 源代码，Linux 内核和各驱动模块的源代码，以及适用于 Windows 桌面环境和 Linux 桌面环境的各种开发调试工具，应用开发样例等。

2. 使用前准备

2.1. 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. 设置拨码开关

单板机支持三种启动模式，分别是 emmc，SD 卡和 USB（Download）启动模式。

B2 B1 B0	Initial boot mode	
0 0 0	UART and USB	
0 1 0	eMMC	
0 1 1	Nand Flash	Don't use
1 0 1	SD Card	
1 1 0	UART and USB	

单板机设置从 emmc 模式启动后，串口终端打印 TF-A、 OP-TEE、 U-Boot 和内核的运行信息，如下所示。注意单板机默认没有镜像，参考第三章快速开始来烧录镜像。

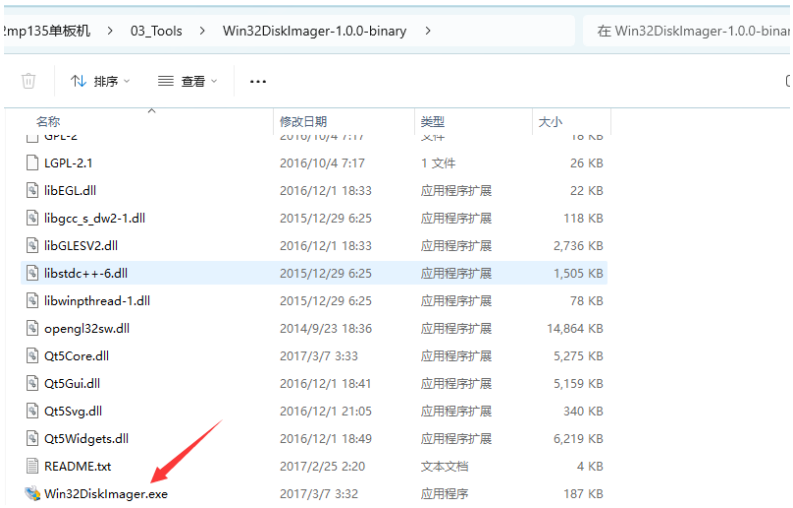
3. 快速开始

3.1. 镜像配置

出厂开发板没有烧写镜像，配置拨码为从 SD 卡启动，安装好串口终端软件。ECB10 可以从 SD 卡或者板上 emmc 启动。制作 SD 卡启动器的速度更快，所以这里没有介绍如何烧录 emmc，更多详细内容请参考《Software_Development_Guide》。

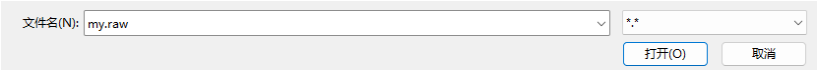
下面描述如何制作 SD 卡快速启动板子：

使用第三方的镜像烧录工具将位于 02_Images/sdmmc 中的 img 镜像文件，根据核心板型号选择镜像，135-D5E8 选择 FlashLayout_sdcard_ebyte-stm32mp135-D5E8-optee.raw，135-D2 选择 FlashLayout_sdcard_ebyte-stm32mp135-D2-optee.raw。在 03_Tools/中已经包含了一个烧写工具：Win32DiskImager-1.0.0-binary。双击打开即可。



选择镜像文件

文件类型要选择*.*才能找到烧录文件



使用读卡器找到设备之后，单击写入，然后等待烧写完成即可。

3.2. 上电启动

插入 SD 卡，在板子上找到 DEBUG 丝印，正确连接之后，使用 USB-TTL 串口连接上电脑，配置拨码为从 SD 卡启动。

单板机启动后，串口终端打印 U-Boot 和内核的运行信息。

```
NOTICE:  Model: STMicroelectronics STM32MP135F-DK Discovery Board
INFO:    Reset reason (0x34):
INFO:      Pad Reset from NRST
INFO:    FCONF: Reading TB_FW firmware configuration file from: 0x2ffe0000
INFO:    FCONF: Reading firmware configuration information for: stm32mp_io
INFO:    Using SDMMC
INFO:      Instance 1
INFO:    Boot used partition fsbl1
NOTICE:  BL2: v2.6-stm32mp1-r2.0(debug):5cd2082-dirty(5cd20824)
NOTICE:  BL2: Built : 02:36:15, Sep 12 2025
INFO:    BL2: Doing platform setup
INFO:    RAM: DDR3-1066 bin F 1x4Gb 533MHz v1.53
INFO:    Memory size = 0x20000000 (512 MB)
INFO:    BL2: Loading image id 1
INFO:    Loading image id=1 at address 0x30006000
INFO:    Image id=1 loaded: 0x30006000 - 0x30006246
INFO:    FCONF: Reading FW_CONFIG firmware configuration file from: 0x30006000
INFO:    FCONF: Reading firmware configuration information for: mce_config
INFO:    FCONF: Reading firmware configuration information for: dyn_cfg
INFO:    FCONF: Reading firmware configuration information for: stm32mp1_firewall
INFO:    BL2: Loading image id 4
INFO:    Loading image id=4 at address 0xde200000
INFO:    Image id=4 loaded: 0xde200000 - 0xde20001c
INFO:    OPTEE ep=0xde200000
INFO:    OPTEE header info:
INFO:          magic=0x4554504f
INFO:          version=0x2
INFO:          arch=0x0
INFO:          flags=0x0
INFO:          nb_images=0x1
INFO:    BL2: Loading image id 8
INFO:    Loading image id=8 at address 0xde200000
INFO:    Image id=8 loaded: 0xde200000 - 0xde2782e8
INFO:    BL2: Loading image id 2
```

启动后在 uboot 阶段请根据核心板的型号来选择加载的设备树文件。这里默认选择 135-D5E8 的型号，如果超过选择时间则自动加载默认设备树。

3.3. 修改默认配置

因为不同核心板支持的硬件功能有差异，而且内存容量也不同，所以需要按照核心板的型号来选择加载的设备树文件。如果需要修改默认配置，需要改动 bootfs.ext4 里面的配置文件。可以在烧录之前修改 bootfs.ext4（推荐，更安全）或者启动开发板之后进行修改。

```
Hit any key to stop autoboot:  0
Boot over mmc0!
switch to partitions #0, OK
mmc0 is current device
Scanning mmc 0:8...
Found /mmc0_extlinux/extlinux.conf
Retrieving file: /mmc0_extlinux/extlinux.conf
983 bytes read in 11 ms (86.9 KiB/s)
Retrieving file: /splash_landscape.bmp
7802 bytes read in 12 ms (634.8 KiB/s)
Select the boot mode
1:      ebyte-stm32mp135-D5E8
2:      ebyte-stm32mp135-D2
3:      ebyte-stm32mp131-D2E8
4:      ebyte-stm32mp131-D2
Enter choice: 1:        ebyte-stm32mp135-D5E8
```

mmc0 对应 SD 卡，mmc1 对应 emmc。

如果在启动开发板之后进行修改，我们需要直接访问挂载好的/media/mmcblk0p8/。然后进行修改。

```
#cd /media/mmcblk0p8/
#ls
boot.scr.uimg                  ebyte-stm32mp135-D5E8.dtb   splash_landscape.bmp
ebyte-stm32mp131-D2.dtb        lost+found                  splash_portrait.bmp
ebyte-stm32mp131-D2E8.dtb      mmc0_extlinux               st-image-resize-initrd
ebyte-stm32mp135-D2.dtb        mmc1_extlinux               uImage
```

这里的 mmc0_extlinux 和 mmc1_extlinux 分别代表 SD 卡和 EMMC。我们是从 SD 卡启动的，所以这里修改 mmc0_extlinux 下的配置文件 extlinux.conf。

```
# vi mmc0_extlinux/extlinux.conf
```

将其中的 DEFALT 改为下面的任意一个 LABEL 即可。

如果需要在烧录之前就提前修改好，则按照上述的方法挂载提供的 Images 里面的 bootfs.ext4，然后修改对应位置。最后使用 cubeprogrammer 烧录到 emmc，或者制作 raw 文件烧录到 sd 卡即可。

4. 功能测试

4.1. 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。

4.1.1. CPU

ECB10 核心芯片是 STM32MP135DAF7，基于高性能单核 Arm®Cortex®-A7 32 位 RISC 核心，工作频率为 1Ghz。Cortex-A7 处理器的 CPU 核心包括一个 32 kbyte L1 指令缓存，一个 32 kbyte L1 数据缓存，一个 128 kbyte 二级缓存。Cortex-A7 处理器是一款非常节能的应用处理器，旨在为高端可穿戴设备以及其他低功耗嵌入式和消费应用提供丰富的性能。它提供了比 Cortex-A5 多 20%的单线程性能，并且提供了与 Cortex-A9 相似的性能。

4.1.1.1. 查看 CPU 信息

使用 cat /proc/cpuinfo 查看 CPU 信息。

```
root@ebyte-ubuntu:~# cat /proc/cpuinfo
processor       : 0
model name      : ARMv7 Processor rev 5 (v7l)
BogoMIPS        : 48.00
Features        : half thumb fastmult vfp edsp thumbee neon vfpv3 tls vfpv4 idiva idivt
vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x0
CPU part        : 0xc07
CPU revision    : 5

Hardware        : STM32 (Device Tree Support)
Revision        : 0000
Serial          : 000A0010343251163537334
```

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）

4.1.1.2. 查看 CPU 使用率

```
top - 13:56:36 up 58 min,  0 users,  load average: 0.00, 0.00, 0.11
Tasks:  64 total,   1 running,  33 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  1.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   449616 total,   370472 free,    30388 used,    48756 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   407640 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  255 root      20   0    4908   2284   1868 R   0.7   0.5   0:00.16 top
    1 root      20   0   28520   5696   4472 S   0.0   1.3   0:03.30 systemd
    2 root      20   0       0      0      0 S   0.0   0.0   0:00.01 kthreadd
    3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
    5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
    6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
    8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0+
    9 root      20   0       0      0      0 I   0.0   0.0   0:03.06 kworker/u2:+
   10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
   11 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_k+
   12 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_t+
   13 root      20   0       0      0      0 S   0.0   0.0   0:00.25 ksoftirqd/0
   14 root      20   0       0      0      0 I   0.0   0.0   0:00.34 rcu_preempt
   15 root      20   0       0      0      0 S   0.0   0.0   0:00.02 kdevtmpfs
   16 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 inet_frag_wq
   20 root      20   0       0      0      0 I   0.0   0.0   0:00.77 kworker/0:4+
```

%usr：表示用户空间程序的 cpu 使用率

%sys：表示系统空间的 cpu 使用率

%idle：空闲 cpu

%irq：cpu 处理硬中断的数量

%sirq：cpu 处理软中断的数量

4.1.1.3. 查看 CPU 温度信息

通过 cat /sys/class/hwmon/hwmon0/temp1_input 来查看 CPU 的内部温度。

```
root@ebyte-ubuntu:~# cat /sys/class/hwmon/hwmon0/temp1_input
54988
```

上面的数值除以 1000 就是正确的温度值，单位为摄氏度。

4.1.2. 内存

4.1.2.1. 查看内存信息

通过 cat /proc/meminfo 可以查看内存信息。

```
root@ebyte-ubuntu:~# cat /proc/meminfo
MemTotal:         449616 kB
MemFree:          370472 kB
MemAvailable:     407640 kB
Buffers:               0 kB
Cached:           39208 kB
SwapCached:            0 kB
Active:           13408 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。

4.1.2.2. 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 10，测试命令为“memtester 10M 10"。

```
root@ebyte-ubuntu:~# memtester 10M 10
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 1/10:
  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
*****
```

4.1.3. NandFlash 测试

4.1.3.1. 查看 Nand 容量及大小

```
root@ebyte-ubuntu:~# cat /proc/mtd
dev:    size   erasesize  name
mtd0: 00080000 00020000 "fsbl1"
mtd1: 00080000 00020000 "fsbl2"
mtd2: 00080000 00020000 "metadata1"
mtd3: 00080000 00020000 "metadata2"
mtd4: 00400000 00020000 "fip-a1"
mtd5: 00400000 00020000 "fip-a2"
mtd6: 00400000 00020000 "fip-b1"
mtd7: 00400000 00020000 "fip-b2"
mtd8: 0ee00000 00020000 "UBI"
```

4.1.4. RTC

RTC（Real-time clock）本身是一个时钟，用来记录真实时间，当软件系统关机后保留系统时间并继续进行计时，系统重新开启后在将时间同步进软件系统。

STM32MP135 芯片内部包含 RTC 时钟，如果实际产品对 RTC 功耗要求不是很高，对断电时间保持要求在一个月以内，可以直接使用芯片内部 RTC，否则就需要采用专用外部 RTC 芯片了。RTC 的测试通常采用 Linux 系统常用的 hwclock 和 date 命令配合进行，下面测试将系统时间写入 RTC，读取 RTC 时间并设置为系统时间并进行时间掉电保持的测试。

设置系统时间

将系统时间设置为 2024.07.12 16:35:15

```
root@ebyte-ubuntu:~# date 071216352024.15
Fri Jul 12 16:35:15 UTC 2024
```

将系统时间写入 RTC

```
root@ebyte-ubuntu:~# hwclock -w
```

```
root@ebyte-ubuntu:~# hwclock -r
2000-01-01 07:07:15.965766+0000
```

掉电保持 RTC 时间

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

```
root@ebyte-ubuntu:~# hwclock -r
2024-07-12 16:52:42.648561+0000
```

4.1.5. WatchDog

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

```
root@ebyte-ubuntu:/home# ./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，开发板会重启。

4.1.6. 电源管理

本章节演示 Linux 电源管理的 Suspend 功能，让开发板睡眠 ，通过外部事件唤醒。Linux 内核一般提供了三种 Suspend: Freeze、Standby 和 STR(Suspend to RAM)，在用户空间向”/sys/power/state”文件分别写入”freeze”、”standby”和”mem”，即可触发它们。目前只支持休眠到内存的方式，即“mem”方式。

查看当前支持的模式

```
root@ebyte-ubuntu:~# cat /sys/power/state
mem
```

设置唤醒源

```
root@ebyte-ubuntu:/home#echoenabled >/sys/devices/platform/soc/40010000.serial/tty/ttySTM0/power/wakeup
```

休眠到内存

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

4.2. 外设接口

4.2.1. GPIO

GPIO 的测试是通过文件系统 sysfs 接口来实现的。pin 脚的编号定义为 #define PIN_NO(port, line) (((port) - 'A') * 0x10 + (line))，其中 port 为 gpio 端口，line 为该 gpio 对应引脚，((port)-'A') 代表 ASCII 码相减。如 PA8 对应的 pin 引脚编号为:PIN_NO('A',8)=(0x41-0x41)*0x10+8=(65-65)*16+8=8。

导出 GPIO

```
root@ebyte-ubuntu:~# cd /sys/class/gpio/
root@ebyte-ubuntu:/sys/class/gpio# ls
export       gpiochip112  gpiochip16  gpiochip48  gpiochip80  unexport
gpiochip0    gpiochip128  gpiochip32  gpiochip64  gpiochip96
root@ebyte-ubuntu:/sys/class/gpio# echo 8 > export
```

设置 GPIO 方向

输入

```
root@ebyte-ubuntu:/sys/class/gpio# echo "in" > PA8/direction
```

输出

```
root@ebyte-ubuntu:/sys/class/gpio# echo "out" > PA8/direction
```

设置 GPIO 值

```
root@ebyte-ubuntu:/sys/class/gpio# echo 1 > PA8/value
```

注销 GPIO

```
root@ebyte-ubuntu:/sys/class/gpio# echo 8 > unexport
root@ebyte-ubuntu:/sys/class/gpio# ls
export       gpiochip112  gpiochip16  gpiochip48  gpiochip80  unexport
gpiochip0    gpiochip128  gpiochip32  gpiochip64  gpiochip96
```

4.2.2. 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，我们已经列出了开发板上所有的 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-ubuntu:/sys/class/leds# ls
green:heartbeat
```

点亮和熄灭 LED

```
root@ebyte-ubuntu:/sys/class/leds/green:heartbeat# echo 0 > brightness
root@ebyte-ubuntu:/sys/class/leds/green:heartbeat# echo 1 > brightness
```

查看 LED 触发模式

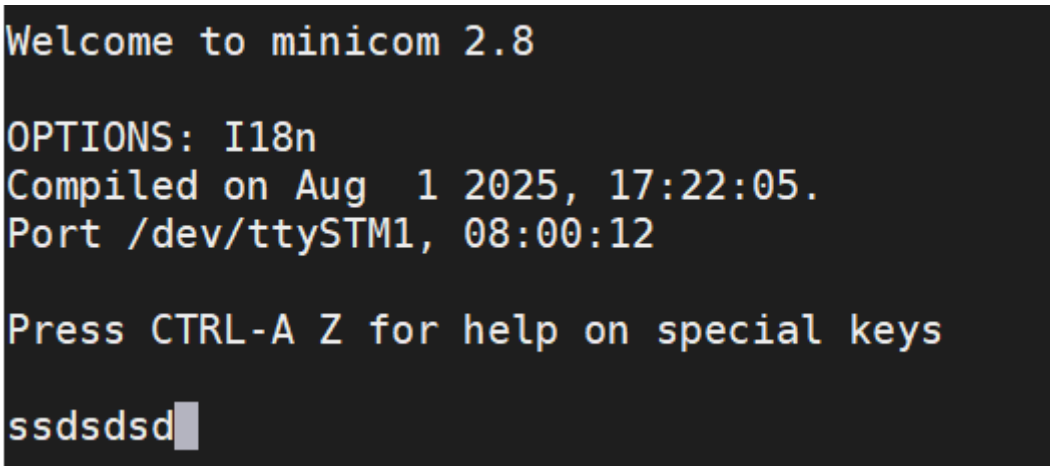
```
root@ebyte-ubuntu:/sys/class/leds/green:heartbeat# cat trigger
[none] rfkill-any rfkill-none kbd-scrolllock kbd-numlock kbd-capslock kbd-kanalock kbd-shiftlock kbd-altgrlock kbd-ctrllock kbd-altlock kbd-shiftllock kbd-shiftrlock kbd-ctrllock kbd-ctrlrlock timer oneshot heartbeat backlight gpio cpu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

4.2.3. 串口

本单板机引出了两路串口供使用，分别是 UART5 和 USART3,其中 UART5 是双线制，只有 RX 和 TX，USART3 是四线制，有 RX,TX,RTS 和 CTS。

4.2.3.1. UART7

此串口可以直接使用 usb-ttl 转接线连接 PC 进行测试。在单板机中被注册为 ttySTM1，使用 minicom 打开 ttySTM1 即可通讯。



4.2.4. 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。

4.2.4.1. 1.插入 U 盘

插入 U 盘和使用 lsblk 查找存储设备。

```
root@ebyte-ubuntu:~# lsblk
NAME         MAJ:MIN RM    SIZE RO TYPE MOUNTPOINT
sda            8:0    1 58.6G    0 disk
|-sda1         8:1    1 58.6G    0 part
`-sda2         8:2    1       1M    0 part
mtdblock0    31:0    0    512K    0 disk
mtdblock1    31:1    0    512K    0 disk
mtdblock2    31:2    0    512K    0 disk
mtdblock3    31:3    0    512K    0 disk
mtdblock4    31:4    0      4M    0 disk
mtdblock5    31:5    0      4M    0 disk
mtdblock6    31:6    0      4M    0 disk
mtdblock7    31:7    0      4M    0 disk
mtdblock8    31:8    0    238M    0 disk
```

4.2.4.2. U 盘挂载读写

挂载 U 盘

```
root@ebyte-ubuntu:~# mount /dev/sda1 /mnt/sda/
```

读文件

提前在 u 盘中创建一些文件

```
root@ebyte-ubuntu:~# mount /dev/sda1 /mnt/sda/
root@ebyte-ubuntu:~# ls /mnt/sda/
MobaXterm_Installer_v12.3.zip    conf
'System Volume Information'      setup.exe
 autorun.inf                      sources
 boot                             support
```

写文件

```
root@ebyte-ubuntu:~# mount /dev/sda1 /mnt/sda/
root@ebyte-ubuntu:~# ls /mnt/sda/
MobaXterm_Installer_v12.3.zip    conf
'System Volume Information'      setup.exe
 autorun.inf                      sources
 boot                             support
```

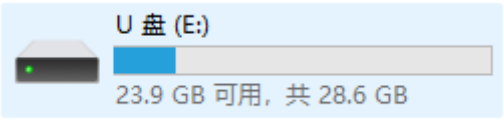
4.2.5. OTG

我们可以将插入单板机的 U 盘挂载到 PC，或者 dd 一个区域供 PC 访问。

我们先插入一个 U 盘

```
# ls /dev/sda1
/dev/sda1
# modprobe libcomposite
# modprobe usb_f_mass_storage
# modprobe g_mass_storage file=/dev/sda1 removable=1
```

此时 windows 应该会弹出提示，如果开着虚拟机的话，选择 u 盘给 windows 访问。就可以看到一个 U 盘。



或者创建一个 100M 的分区供 PC 访问

```
# modprobe g_mass_storage -r
# dd if=/dev/zero of=/usb_otg_disk.img bs=100M count=1
# mkfs.vfat /usb_otg_disk.img
# modprobe g_mass_storage file=/usb_otg_disk.img removable=1
```

配置为主模式之后，使用 usb 转 typec 母连接 usb 设备，将会自动挂载，比如鼠标，u 盘。

插入 U 盘之后，将自动切换为 host 模式。

```
# cat /sys/devices/platform/soc/49000000.usb-otg/usb_role/49000
000.usb-otg-role-switch/role
# ls /dev/sda1
/dev/sda1
```

4.2.6. SD 卡

ECB10-135A5M5M-I 使用 SDMMC1 连接 Micro SD。

4.2.6.1. 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：

```
lsblk
NAME            MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
|-mmcblk0p1 179:1    0   256K   0 part
|-mmcblk0p2 179:2    0   256K   0 part
|-mmcblk0p3 179:3    0   256K   0 part
|-mmcblk0p4 179:4    0   256K   0 part
|-mmcblk0p5 179:5    0     4M   0 part
|-mmcblk0p6 179:6    0     4M   0 part
|-mmcblk0p7 179:7    0   512K   0 part
|-mmcblk0p8 179:8    0    64M   0 part
`-mmcblk0p9 179:9    0 29.7G   0 part
```

4.2.6.2. 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/mnt/rootfs。

写文件测试

```
time dd if=/dev/zero of=/mnt/rootfs/home/tempfile bs=1M count=100 conv=fdatasync
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 5.95716 s, 17.6 MB/s

real    0m6.203s
user    0m0.000s
sys     0m2.125s
```

读文件测试

读文件时忽略 cache 的影响。这是可以指定参数 iflag=direct，nonblock。

```
time dd if=/mnt/rootfs/home/tempfile of=/dev/null bs=1M count=100 iflag=direct,nonblock
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 4.46744 s, 23.5 MB/s


real    0m4.495s
user    0m0.000s
sys     0m0.059s
```

4.2.7. 显示

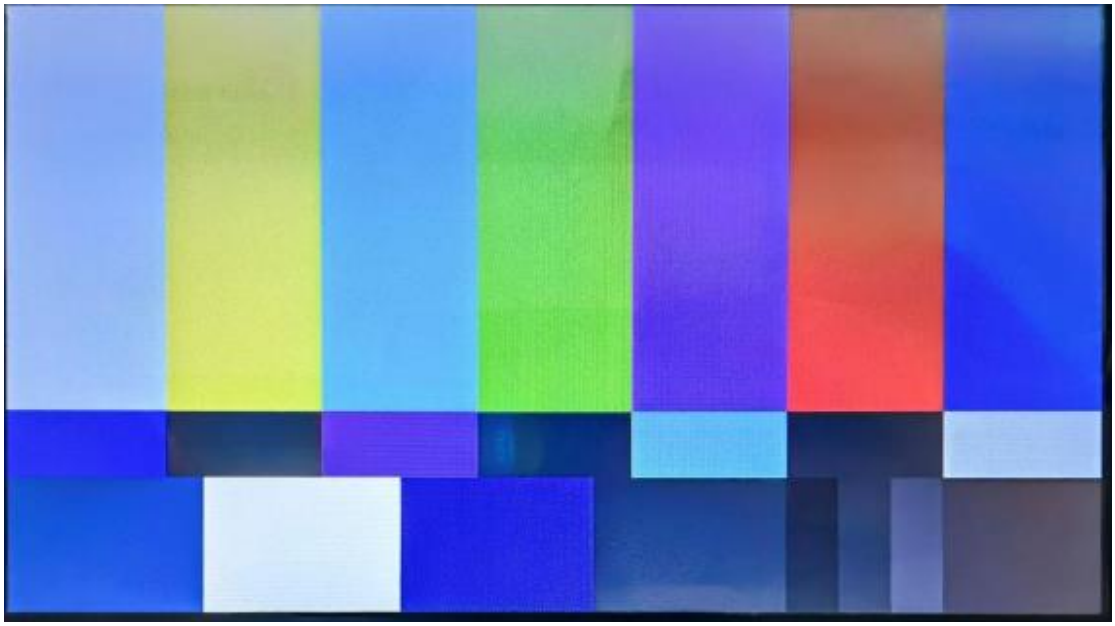
ECB10-PGD 支持 LCD 显示方案：

LCD 显示：LCD 测试部分将演示对 Linux 的 drm 设备操作，实现液晶输出显示 RGB 颜色和颜色合成测试。LCD 采用 RGB888 的显示模式。

4.2.7.1. LCD 显示

如果用户使用过程需要 LCD 显示功能，启动开发板，会显示开机 logo。 使用 modetest 来测试屏幕的显示。

```
# modetest -M stm -s 32@41:1024x600
```



4.2.8. 触摸

evtest 命令测试

终端执行“evtest”进入测试界面。选择测试外设为触摸屏，这里默认为输入中断 0，测试界面选择“0”按下回车即可开始测试。

```
evtest
No device specified, trying to scan all of /dev/input/event*
Available devices:
/dev/input/event0:          Goodix Capacitive TouchScreen
/dev/input/event1:          wake_up
Select the device event number [0-1]: 0
Input driver version is 1.0.1
Input device ID: bus 0x18 vendor 0x416 product 0x38f version 0x1060
Input device name: "Goodix Capacitive TouchScreen"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 59 (KEY_F1)
    Event code 60 (KEY_F2)
    Event code 61 (KEY_F3)
    Event code 62 (KEY_F4)
    Event code 63 (KEY_F5)
    Event code 64 (KEY_F6)
    Event code 125 (KEY_LEFTMETA)
    Event code 330 (BTN_TOUCH)
  Event type 3 (EV_ABS)
    Event code 0 (ABS_X)
      Value      0
      Min        0
      Max     1023
    Event code 1 (ABS_Y)
      Value      0
      Min        0
      Max      599
    Event code 47 (ABS_MT_SLOT)
      Value      0
      Min        0
```

```
        Min             0
        Max          1023
    Event code 54 (ABS_MT_POSITION_Y)
        Value           0
        Min             0
        Max           599
    Event code 57 (ABS_MT_TRACKING_ID)
        Value           0
        Min             0
        Max         65535
Properties:
  Property type 1 (INPUT_PROP_DIRECT)
Testing ... (interrupt to exit)
Event: time 946684926.419206, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value 0
Event: time 946684926.419206, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value 248
Event: time 946684926.419206, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 313
Event: time 946684926.419206, type 3 (EV_ABS), code 48 (ABS_MT_TOUCH_MAJOR), value 50
Event: time 946684926.419206, type 3 (EV_ABS), code 50 (ABS_MT_WIDTH_MAJOR), value 50
Event: time 946684926.419206, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 1
Event: time 946684926.419206, type 3 (EV_ABS), code 0 (ABS_X), value 248
Event: time 946684926.419206, type 3 (EV_ABS), code 1 (ABS_Y), value 313
Event: time 946684926.419206, ---------- SYN_REPORT ------------
Event: time 946684926.472114, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value 314
Event: time 946684926.472114, type 3 (EV_ABS), code 1 (ABS_Y), value 314
```

由上面可知，主要显示坐标值、键值，具体信息如下：

EV_SYN：同步事件

EV_KEY：按键事件，如 BTN_TOUCH 表示是触摸按键

EV_ABS：绝对坐标，如触摸屏上报的坐标

BTN_TOUCH：触摸按键

ABS_MT_TRACKING_ID 表示采集信息开始，后面一个 ABS_MT_TRACKING_ID 表示采集信息结束

单点触摸信息是以 ABS 承载并按一定顺序发送,如：

ABS_X：是相对于屏幕绝对坐标 X

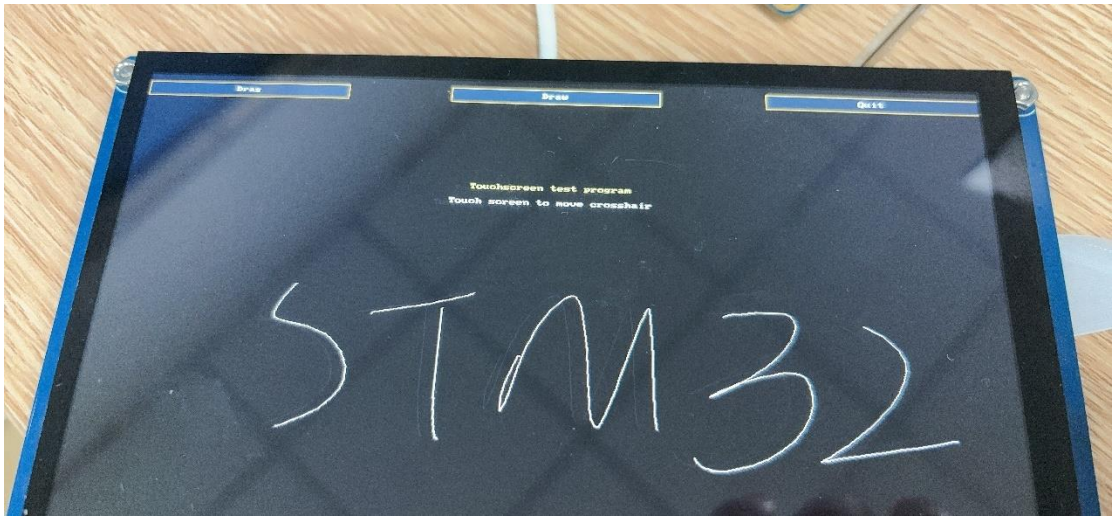
ABS_Y:是相对于屏幕绝对坐标 Y 而多点触摸信息则是以 ABS_MT 承载并按一定顺序发送，如：

ABS_MT_POSITION_X：表示屏幕接触面的中心点 x 坐标位置.

ABS_MT_POSITION_Y：表示屏幕接触面的中心点 Y 坐标位置

Ts_test 也可以测试

```
# ts_test
```



4.3. 音频

请准备一个带麦克风的耳机，连接 HP OUT/MIC IN 接口。

播放

```
#aplay /usr/share/misc/test.wav
```

录音

```
# arecord test.wav
# aplay test.wav
```

4.4. 网络接口

4.4.1. 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。

```
ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.250  netmask 255.255.255.0  broadcast 192.168.0.255
        ether 00:04:9f:04:d2:35  txqueuelen 1000  (Ethernet)
        RX packets 3524  bytes 3681632 (3.6 MB)
        RX errors 0  dropped 73  overruns 0  frame 0
        TX packets 1262  bytes 218100 (218.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 44  base 0xc000
```

下面介绍给 eth0 手动配置 IP 地址 192.168.0.250 的方法，命令如下：

```
ifconfig eth0 192.168.0.250 netmask 255.255.255.0 up
```

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。

```
ping 192.168.0.130
PING 192.168.0.130 (192.168.0.130) 56(84) bytes of data.
64 bytes from 192.168.0.130: icmp_seq=1 ttl=64 time=1.53 ms
64 bytes from 192.168.0.130: icmp_seq=2 ttl=64 time=1.34 ms
64 bytes from 192.168.0.130: icmp_seq=3 ttl=64 time=1.37 ms
64 bytes from 192.168.0.130: icmp_seq=4 ttl=64 time=1.30 ms
```

4.4.1.1. 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
```

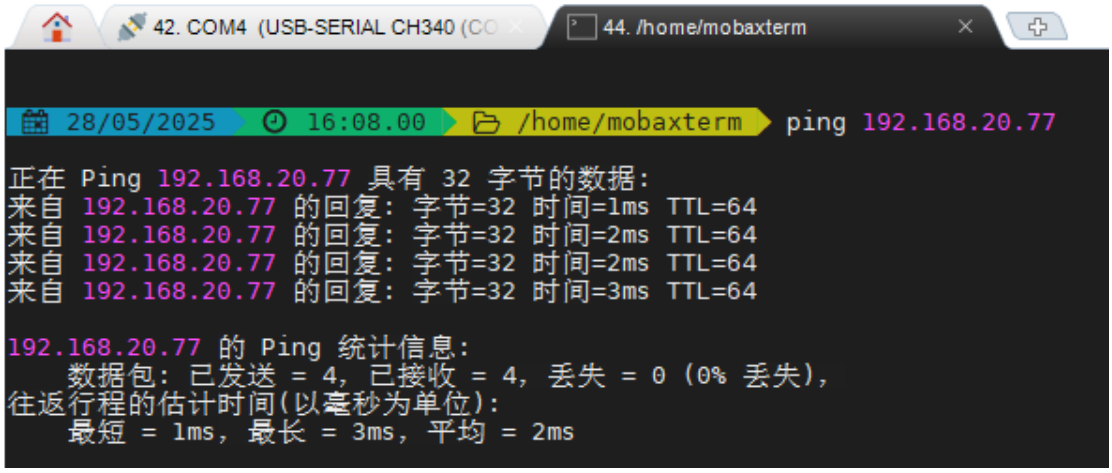
自动配置网络只需要 udhcpc 命令。

```
# udhcpc -i eth0
```

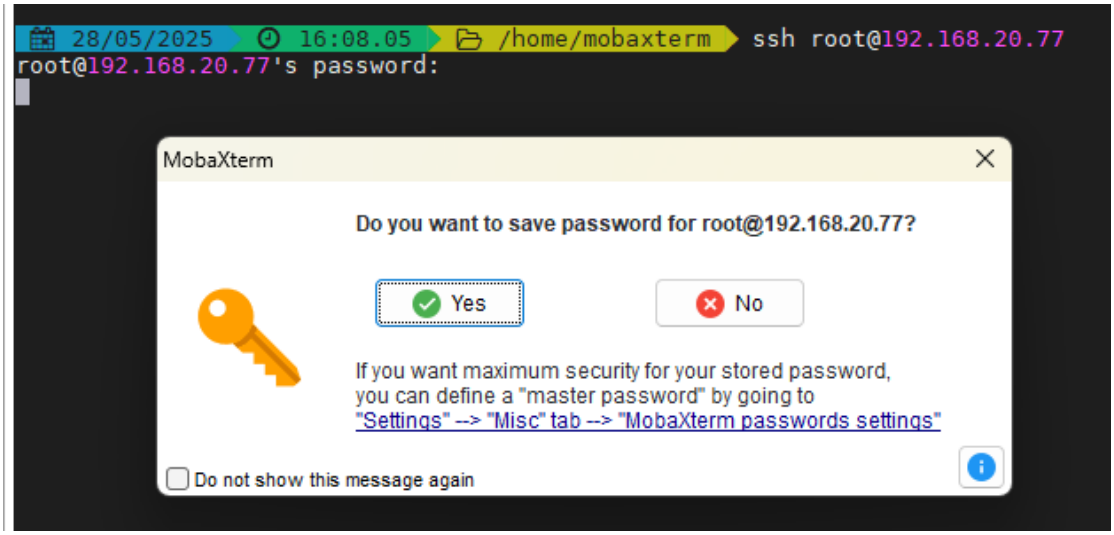
4.4.1.2. SSH 登录开发板

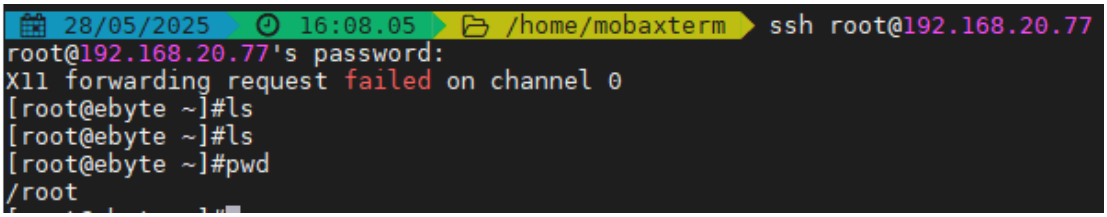
可以使用 SSH 登录开发板，前提条件是需要登陆的机器和开发板在同一网段，可以互相 PING 通。可以使用 mobaExterm 提供的 SSH 服务，也可以使用 cmd 命令行的自带的 SSH。mobaExterm 更方便传输文件，我们以 mobaExterm 为例。

首先保证主机可以 ping 通开发板。192.168.20.77 是开发板设置的 IP 地址。



SSH 登录，开发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。

4.4.1.3. 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。

虚拟机拷贝文件到开发板

```
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
```

4.4.2. WIFI

本节主要介绍 Linux 下 Wi-Fi 的配置和使用，通常 Wi-Fi 模块可以支持两种工作模式，分别是 STA 模式和 AP 模式，有些外设模块还支持 STA 和 AP 模式同时工作。STA 模式允许开发板连接外部 Wi-Fi 热点，AP 模式将开发板变成 Wi-Fi 热点，供其它外部设备连接。

ECB10-PGB 核心板板载 wifi 是 rtl8723du，使用前加载驱动。

```
# depmod
# modprobe 8723du
# rfkill unblock 1
# ifconfig wlan0 up
```

驱动加载的过程中会将位于/lib/firmware/brcm 的 Wi-Fi 固件加载到模块内部。WiFi 模块驱动加载成功之后生成 Wi-Fi 的网络节点 wlan0，如下所示

```
# ifconfig wlan0
wlan0        Link encap:Ethernet    HWaddr E0:C1:1A:20:53:6D
             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 (0.0 B)    TX bytes:0 (0.0 B)
# iw dev wlan0 scan | grep SSID #扫描附近的 wifi 热点
```

4.4.2.1. STA 模式连接 WIFI 热点

下面尝试手动连接附近的 Wi-Fi 热点“TST-2.4G”，这是一个采用 WPA2 加密方式的 Wi-Fi 热点，密码为 12345678。

```
wpa_passphrase    TST-2.4G TST12345678 >> /etc/wpa_supplicant.conf
```

连接 wifi 热点

```
killall wpa_supplicant
wpa_supplicant -B -c /etc/wpa_supplicant.conf -i wlan0
```

使用-B 参数可以在后台运行连接 WiFi，如果去掉可以查看连接过程。

连接成功后需要分配 IP 地址，使用 udpchc 分配 IP 地址

```
udhcpc -i wlan0
udhcpc: started, v1.30.1
udhcpc: sending discover
udhcpc: sending select for 172.20.10.5
udhcpc: lease of 172.20.10.5 obtained, lease time 86400
ip: RTNETLINK answers: File exists
```

查看连接状态

```
iw wlan0 link
Connected to 4a:fe:e2:56:e0:36 (on wlan0)
        SSID: iPhone
        freq: 2437
        RX: 2652 bytes (17 packets)
        TX: 1806 bytes (13 packets)
        signal: -54 dBm
        tx bitrate: 24.0 MBit/s

        bss flags:      short-slot-time
        dtim period:    1
        beacon int:     100
```

扫描 WIFI 热点

```
iw dev wlan0 scan | grep SSID
        SSID: iPhone
        SSID: zhendu
        SSID: TP-LINK_\xef\xbc\x88\xe7\xa1\xac\xe4\xbb\xb6\xe7\xbb\x84)
        SSID: \x00\x00\x00\x00\x00\x00
        SSID: TP-LINK_6C35
        SSID: B05_2.4G_\xe6\x89\x93\xe5\x8d\xb0\xe6\x9c\xba
        SSID: MERCURY_2.4G_D02D
        SSID: KingOfSoftware
        SSID: \xe5\xa4\xa7\xe5\x8e\x85\xe7\xbd\x91\xe7\xbb\x9cA-5G
```

4.4.2.2. AP 模式开启 WIFI 热点

Hostapd 是一个带加密功能的无线接入点程序，是 Linux 操作系统上构件无线接入点的一个比较方便的工具，支持 IEEE 802.11 协议和 IEEE 802.1X/WPA/WPA2/EAP/RADIUS 加密。板子作为 WiFi 热点，需要为每一个接入该热点的终端（例如手机）分配 IP，路由等网络参数。如创建 SSID 为 EBYTE，PASSWD 为 12345678 的无线 wifi 热点。下面先

以手动方式进行配置：

当使用 AP 模式时，需要为激活 wlan0 并配置一个静态 IP 地址，这里配置一个默认的 IP 地址：192.168.10.1。

```
ifconfig wlan0 192.168.10.1 up
```

在前面我们有说明当前 Wi-Fi 模块不支持 STA 和 AP 模式同时工作，所以这里需要清除 STA 模式的配置，如下：

```
killall udhcpc
killall wpa_supplicant
```

wlan0 工作在 AP 模式，当其它外部设备连接这个 AP 热点的时候，它需要通过 wlan0 为其它外部设备动态分配 IP 地址，所以需要使用 wlan0 来运行 DHCP 服务程序 udhcpd。udhcpd 对应的配置文件为/etc/udhcpd.conf，内容如下：

```
# File: /etc/udhcpd.conf
# the start and end of the IP lease block
start 192.168.10.10
end 192.168.10.254
# the interface that udhcpd will use
interface wlan0
opt         dns         8.8.8.8
option    subnet    255.255.255.0
opt         router    192.168.10.1
option    domain    local
option    lease     864000
```

配置 AP 模式最关键的一步当然是启动 hostapd 服务。启动服务之前需要通过/etc/hostapd.conf 配置 AP 模式的 ssid, password, 加密算法，驱动类型，工作模式等，完整的参数配置说明参见：http://w1.fi/cgit/hostap/plain/hostapd/hostapd.conf,下面是针对当前硬件的/etc/hostapd.conf 配置，内容如下：

```
# File: /etc/hostapd.conf
interface=wlan0
driver=nl80211
# mode Wi-Fi (a = IEEE 802.11a, b = IEEE 802.11b, g = IEEE 802.11g)
hw_mode=g
ssid=EBYTE
channel=7
wmm_enabled=0
macaddr_acl=0
# Wi-Fi closed, need an authentication
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=12345678
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

配置文件准备好之后，执行下面的命令启动 hostapd 服务：

```
#hostapd -B /etc/hostapd.conf
#udhcpd -f /etc/udhcpd.conf &
Configuration file: /etc/hostapd.conf
wlan0: Could not connect to kernel driver
Using interface wlan0 with hwaddr 50:41:1c:b6:9c:32 and ssid "EBYTE"
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

如上面 log 显示，即可以正常使用热点服务了。如果以太网卡 eth0 已经连接 Internet，那么通过下面的配置进行 IP 转发，那连接到 EBYTE 的外部设备也可以连接 Internet 了。

```
echo "1" > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -s 192.168.10.1/24 -o eth0 -j MASQUERADE
```

可以用手机连接上述热点进行上网测试。

4.4.3. 蓝牙

ECB10-PGB 核心板板载的 wifi 模块 rtl8723du 带蓝牙。

```
# hciconfig hci0 up
# hcitool scan
Scanning ...
        E8:FF:F4:98:79:11       iPhone
        D4:D2:52:83:D5:1E       n/a
```

5. 参考资料

- Linux kernel 开源社区：
  https://www.kernel.org/
- STM32MPU 开发社区：
  https://wiki.st.com/stm32mpu/wiki/Development_zone
- STM32MP131 数据手册
- STM32MP135 数据手册

6. 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	25-9-10	HSL	WYQ	WFX

7.

8.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.