

# 北京AEB 软件操作手册

---

## Contents

---

### ARM 主机linux 主机测试说明

#### 测试需求

##### UART 测试

操作方法

相关 Linux 指令

##### PWM 测试

操作方法

相关 Linux 指令

对应关系

##### 4G 测试

操作方法

相关 Linux 指令

##### USB 测试

操作方法

相关 Linux 指令

##### GPIO 测试

操作方法

相关 Linux 指令

对应关系

##### 网口（以太网）测试

操作方法

对应关系

相关 Linux 指令

##### CAN口（控制器局域网络）测试

操作方法

相关 Linux 指令

##### 视频设备测试

操作方法

相关 Linux 指令

安装必要工具

示例操作

高级命令示例

## 命令说明

### RTC 测试

#### 操作方法

检查 RTC 设备

设置 RTC 时间

同步系统时间到 RTC

## ARM 主机linux 主机测试说明

端口测试范围：

- **UART**（通用异步收发传输器 RS232/RS485）
- **PWM**（脉宽调制 PWM）
- **4G**（第四代移动通信技术 4G）
- **USB**（通用串行总线 USB2.0/3.0）
- **GPIO**（通用输入输出 GPIOs）
- **网口**（以太网接口 ethernet）
- **CAN口**（控制器局域网络接口 can, can-fd bus）
- **视频设备**（/dev/video0, video1, video2, video3）
- **RTC**（实时时钟）

注意：本次测试部分接口，我发未有足够外设，测试可能存在部分错误，后期可以更正

## 测试需求

### UART 测试

#### 操作方法

- 连接 UART 设备到主机的 UART 接口。
- 使用串口调试工具进行数据传输和接收测试。

#### 相关 Linux 指令

- 查看串口设备：

```
dmesg | grep tty  
或  
ls /dev/ttyS*  
/dev/ttyS1 /dev/ttyS7 /dev/ttyS8 /dev/ttyS9
```

```
/dev/ttyS1 对应RS232口  
/dev/ttyS7 对应RS485口
```

```
/dev/ttyS8 对应通讯MCU  
/dev/ttyS9 预留
```

- ssh登陆后，使用`minicom`进行串口通信：（TX/RX RS232短路测试）

```
sudo minicom -D /dev/ttyS1 -b 115200
```

- 使用`screen`进行串口通信：

```
screen /dev/ttyS1 115200
```

## PWM 测试

### 操作方法

- 通过 GPIO 引脚输出 PWM 信号。
- 使用示波器或逻辑分析仪观察 PWM 波形。

### 相关 Linux 指令

- 导出 PWM 控制接口：

```
echo <PWM_ID> > /sys/class/pwm/pwmchip0/export
```

- 设置 PWM 周期和占空比：

```
echo <周期> > /sys/class/pwm/pwm0/period  
echo <占空比> > /sys/class/pwm/pwm0/duty_cycle
```

- 启动 PWM 输出：

```
echo 1 > /sys/class/pwm/pwm0/enable
```

## 对应关系

```
PWM0: GPIO0_C2 48-PIN28 PWM0  
PWM1: GPIO3_C3 48-PIN32 PWM1
```

## 4G 测试

### 操作方法

- \* 安装SIM卡。
- \* 连接到移动网络，测试数据传输速度和稳定性。

### 相关 Linux 指令

- 查看 4G 模块设备：(EC20)

```
ls /dev/ttyUSB*    #显示 USB0~USB3
```

- 使用 `mmcli` 管理 4G 连接：

```
sudo mmcli -m 0 --simple-connect="apn=<APN>"
```

- 检查网络状态：

```
nmcli connection show
```

## USB 测试

### 操作方法

- 连接 USB 设备到主机的 USB 接口。
- 测试数据传输和设备识别情况。

### 相关 Linux 指令

- 查看 USB 设备列表：

```
lsusb
```

- 挂载 USB 存储设备：

```
sudo mount /dev/sdX1 /mnt/usb
```

## ■ 复制文件到 USB 设备：

```
cp /path/to/file /mnt/usb/
```

## GPIO 测试

### 操作方法

\* 通过编程或命令行控制 GPIO 状态。

### 相关 Linux 指令

```
sky@Ubuntu-Sky:/$ sudo find -name io_sys*
./sys/devices/platform/iooutput/io_sysfs1
./sys/devices/platform/iooutput/io_sysfs6
./sys/devices/platform/iooutput/io_sysfs4
./sys/devices/platform/iooutput/io_sysfs2
./sys/devices/platform/iooutput/io_sysfs0
./sys/devices/platform/iooutput/io_sysfs7
./sys/devices/platform/iooutput/io_sysfs5
./sys/devices/platform/iooutput/io_sysfs3
./sys/devices/platform/ioinput/io_sysfs1
./sys/devices/platform/ioinput/io_sysfs6
./sys/devices/platform/ioinput/io_sysfs4
./sys/devices/platform/ioinput/io_sysfs2
./sys/devices/platform/ioinput/io_sysfs0
./sys/devices/platform/ioinput/io_sysfs5
./sys/devices/platform/ioinput/io_sysfs3
```

### 对应关系

```
INPUT0: GPIO3_C6 48-PIN25 左转向
INPUT1: GPIO3_C7 48-PIN18 右转向
INPUT2: GPIO4_A0 48-PIN26 刹车灯
INPUT3: GPIO4_A1 48-PIN19 自定义H1
INPUT4: GPIO4_A6 48-PIN27 自定义H2
INPUT5: GPIO4_A7 48-PIN20 自定义L1
INPUT6: GPIO4_B0 48-PIN41 自定义L2

OUTPUT0: GPIO3_D0 INTERNAL 摄像头AHD1 电源
OUTPUT1: GPIO3_D1 INTERNAL 摄像头AHD2 电源
OUTPUT2: GPIO3_D2 INTERNAL 摄像头AHD3 电源
OUTPUT3: GPIO3_D3 INTERNAL 摄像头AHD4 电源
OUTPUT4: GPIO3_D4 INTERNAL 摄像头IPC1 电源
OUTPUT5: GPIO3_D5 INTERNAL I摄像头PC2 电源
OUTPUT6: GPIO0_C5 48-PIN45 正控输出
OUTPUT7: GPIO3_C6 48-PIN43 拉线继电器
```

读取INPUT0：

```
cat /sys/devices/platform/ioinput/io_sysfs0
```

写入OUTPUT0:

```
echo 1> /sys/devices/platform/iooutput/io_sysfs0
```

## 网口（以太网）测试

### 操作方法

- 连接以太网线到主机的网口。
- 配置网络参数，测试网络连通性和带宽。

### 对应关系

```
eth0 IPC1  
eth1 IPC2  
eth2 前面板RJ45
```

### 相关 Linux 指令

- 查看网络接口：

```
ip addr show
```

- 配置静态 IP 地址：

```
sudo ip addr add 192.168.1.100/24 dev eth2  
sudo ip link set eth2 up
```

- 使用 `ping` 测试连通性：

```
ping 1.1.1.1
```

- 测试带宽使用 `iperf3`：

```
iperf3 -c <服务器地址> -t 60
```

- 查看网络流量：

```
ifconfig eth2
```

- 重启网络接口：

```
sudo ip link set eth2 down
```

```
sudo ip link set eth2 up
```

## CAN口（控制器局域网）测试

### 操作方法

- 连接 CAN 设备到主机的 CAN 接口。
- 配置 CAN 接口参数，发送和接收 CAN 消息。

### 相关 Linux 指令

- 安装 CAN 工具包（如果尚未安装）：

```
sudo apt-get install can-utils
```

- 查看 CAN 设备：

```
ip link show can0
```

- 配置 CAN 接口速度并启用：

```
sudo ip link set can0 type can bitrate 500000  
sudo ip link set can0 up
```

- 发送 CAN 消息：

```
cansend can0 123#deadbeef
```

- 接收 CAN 消息：

```
candump can0
```

- 测试 CAN 网络连通性：

```
cansend can0 123#1122334455667788  
candump can0
```

## 视频设备测试

### 操作方法

- 连接摄像头到主机的视频接口。

- 使用相关工具进行视频捕获和测试。

## 相关 Linux 指令

- 查看视频设备列表：

```
ls /dev/video*
```

- 查看视频设备信息：

```
v4l2-ctl --list-devices
```

- 查看视频设备支持的格式：

```
v4l2-ctl --list-formats-ext -d /dev/video0
```

- 捕获单帧图像：

```
ffmpeg -f v4l2 -i /dev/video0 -vframes 1 output.jpg
```

- 录制视频：

```
ffmpeg -f v4l2 -i /dev/video0 -t 10 output.mp4
```

- 查看实时视频流：

```
mplayer tv:// -tv driver=v4l2:device=/dev/video0
```

- 调整视频设备参数（例如分辨率）：

```
v4l2-ctl --set-fmt-video=width=1280,height=720,pixelformat=YUYV -d /dev/video0
```

## 安装必要工具

- 安装 `v4l-utils`：

```
sudo apt-get install v4l-utils ffmpeg
```

- 安装 `ffmpeg`：

```
sudo apt-get install ffmpeg
```

- 安装 `mplayer`：



```
sudo apt-get install mplayer
```

## 示例操作

- 查看连接的视频设备：

```
ls /dev/video* #输出可能包括 `/dev/video0`, `/dev/video1`, `/dev/video2`, `/dev/video3`。
```

- 查看设备支持的格式：

```
v4l2-ctl --list-formats-ext -d /dev/video0
```

- 捕获一张图像：

```
ffmpeg -f v4l2 -i /dev/video0 -vframes 1 output.jpg
```

- 录制 10 秒的视频：

```
ffmpeg -f v4l2 -i /dev/video0 -t 10 output.mp4
```

- 查看实时视频流：

```
mplayer tv:// -tv driver=v4l2:device=/dev/video0
```

- 调整视频分辨率：

```
v4l2-ctl --set-fmt-video=width=1280,height=720,pixelformat=UYVY -d /dev/video0
```

## 高级命令示例

### 命令说明

```
v4l2-ctl --verbose -d /dev/video0 --set-fmt-video=width=1280,height=720,pixelformat=UYVY --stream-mmap=3 --stream-skip=10 --stream-to=./test.yuv --stream-count=1 --stream-poll
```

该命令用于配置并捕获视频设备 `/dev/video0` 的视频流，具体参数说明如下：

- `--verbose`：启用详细模式，输出更多的执行信息，便于调试和查看详细过程。
- `-d /dev/video0`：指定要操作的视频设备为 `/dev/video0`。
- `--set-fmt-video=width=1280,height=720,pixelformat=UYVY`：设置视频格式，分辨率为 1280x720，像素格式为 UYVY (YUV 4:2:2 格式的一种)。
- `--stream-mmap=3`：使用内存映射 (memory-mapped) 方式进行视频流捕获，并设置缓冲区数量为 3。
- `--stream-skip=10`：在开始捕获前跳过前 10 帧数据，通常用于清除缓冲区中的初始无效帧。
- `--stream-to=./test.yuv`：将捕获的视频数据保存到当前目录下的 `test.yuv` 文件中。

- `--stream-count=1`：指定捕获 1 帧视频数据。
- `--stream-poll`：使用轮询方式进行视频流捕获，这种方式在某些系统上可能比默认的触发方式更稳定。

该命令主要用于测试和验证视频设备的捕获功能。通过设置特定的分辨率和像素格式，使用内存映射方式进行高效的数据捕获，并将捕获的数据保存为 YUV 格式的文件，便于后续的图像处理和分析。详细的输出信息有助于调试和确保捕获过程的正确性。

## RTC 测试

### 操作方法

#### 检查 RTC 设备

- 确认 RTC 设备是否存在：

```
ls /dev/rtc* #通常，RTC 设备为 `/dev/rtc0`。
```

- 查看当前时间, 使用 `hwclock` 查看 RTC 当前时间：

```
sudo hwclock --show
```

- 使用 `timedatectl` 查看系统和 RTC 时间：

```
timedatectl
```

#### 设置 RTC 时间

- 设置 RTC 时间与系统时间同步：

```
sudo hwclock --systohc
```

- 手动设置 RTC 时间：

```
sudo hwclock --set --date="2024-04-27 12:34:56"  
sudo hwclock --hctosys
```

#### 同步系统时间到 RTC

- 将系统时间同步到 RTC：

```
sudo hwclock --systohc
```

- 同步 RTC 时间到系统时间, 将 RTC 时间同步到系统：

```
sudo hwclock --hctosys
```

- 调整系统时间, 设置系统时间:

```
sudo date -s "2024-04-27 12:34:56"
```

- 使系统时间生效并同步到 RTC:

```
sudo hwclock --systohc
```

---

Retrieved from "[https://w.g77k.com/index.php?title=北京AEB\\_软件操作手册&oldid=9371](https://w.g77k.com/index.php?title=北京AEB_软件操作手册&oldid=9371)"

---

**This page was last edited on 30 September 2024, at 07:56.**