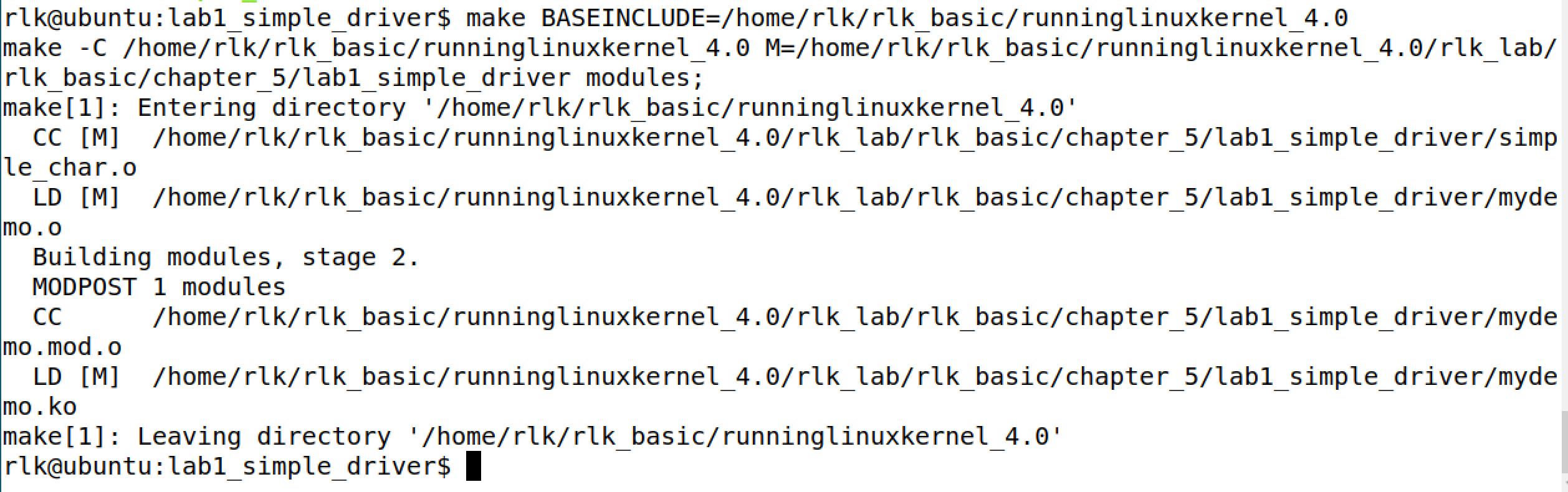
# Lab1\_从一个简单的字符设备开始

## 基本实验步骤

1. 进入对应的lab目录。

|  |
| --- |
| # export ARCH=arm  # export CROSS\_COMPILE=arm-linux-gnueabi-  # make BASEINCLUDE= 你的runninglinuxkernel\_4.0目录的绝对路径 |

这里BASEINCLUDE指定你当前runninglinuxkernel\_4.0的目录路径，每个人机器上路径可能不一样，请确保路径正确。



编译test测试程序。

|  |
| --- |
| # arm-linux-gnueabi-gcc test.c -o test --static |

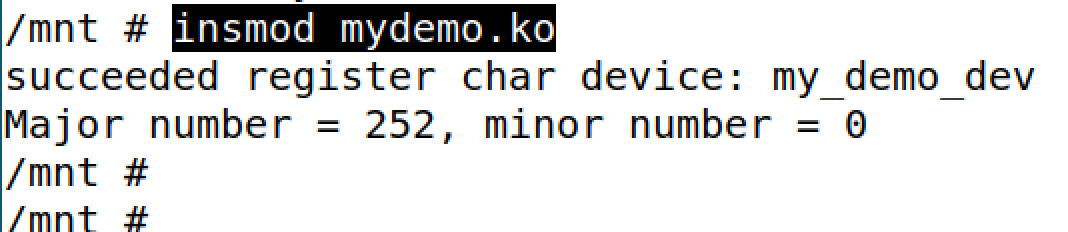
然后把ko内核模块和test程序拷贝到runninglinuxkernel\_4.0/kmodules目录下面。

运行如下脚本启动Qemu。

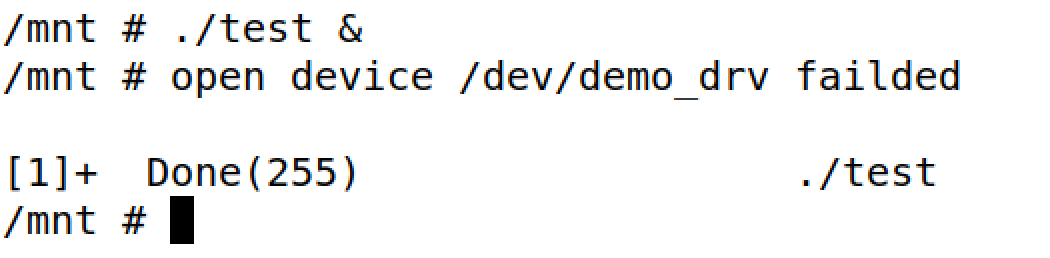
|  |
| --- |
| #cd runninglinuxkernel\_4.0  # sh run.sh arm32 #启动虚拟机 |

在Qemu虚拟机:

|  |
| --- |
| #cd /mnt  # insmod mydemo.ko |

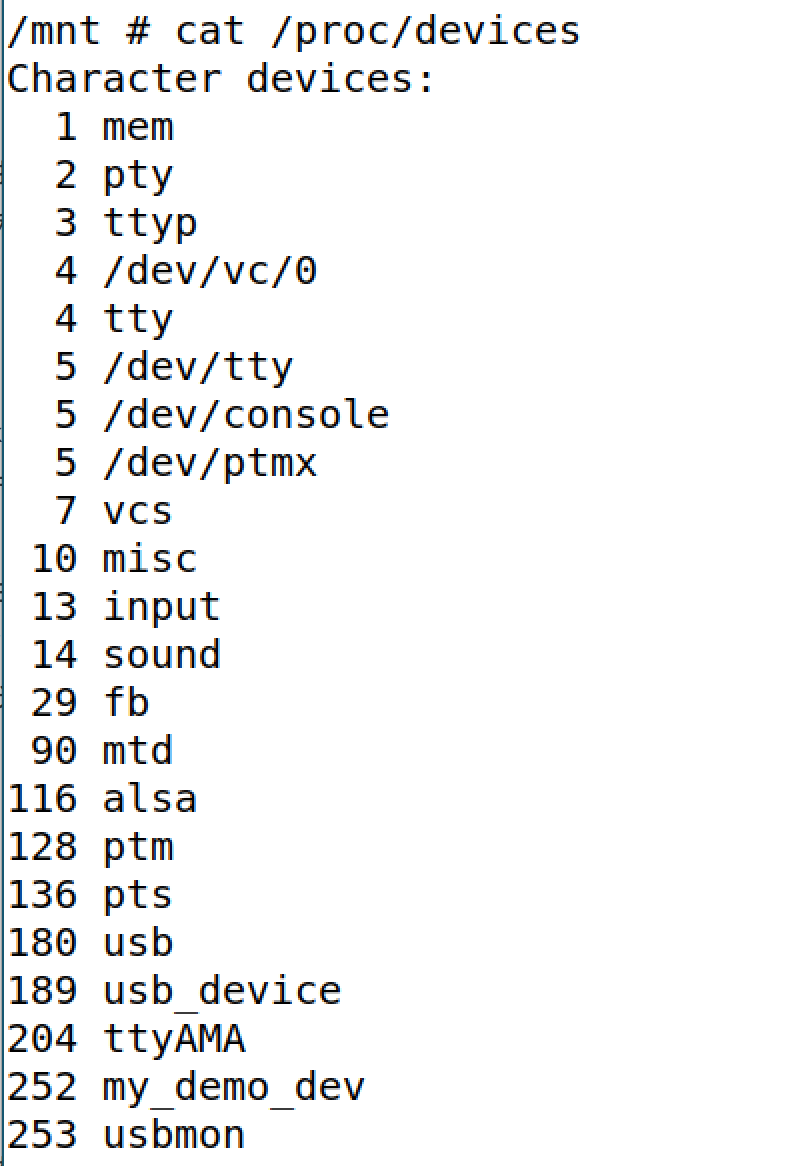


直接跑test程序看看是神马情况。

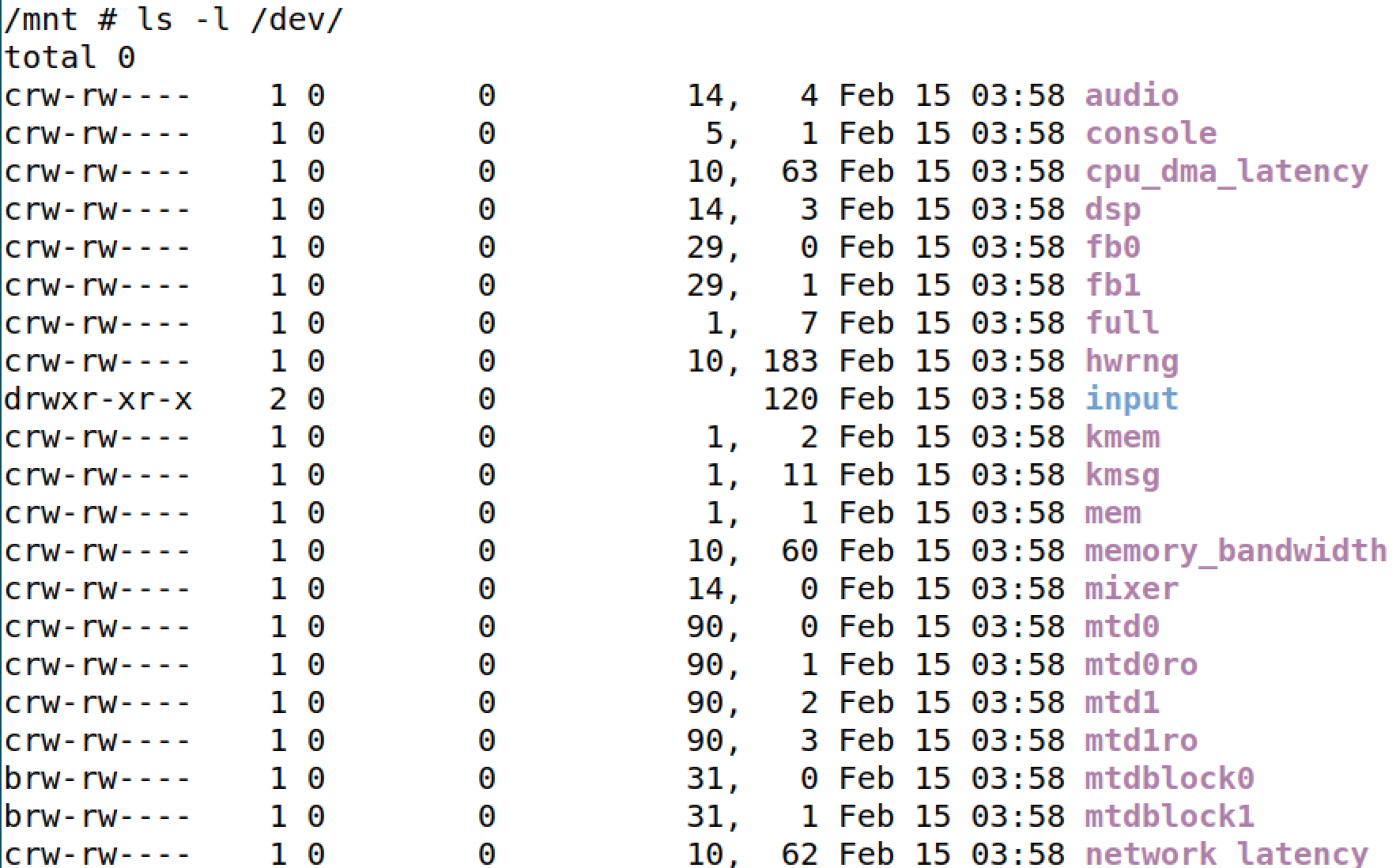


居然报错了，咋回事？

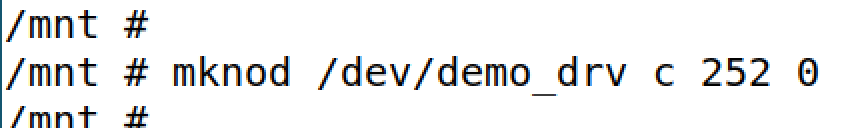
估计是设备节点还没创建。



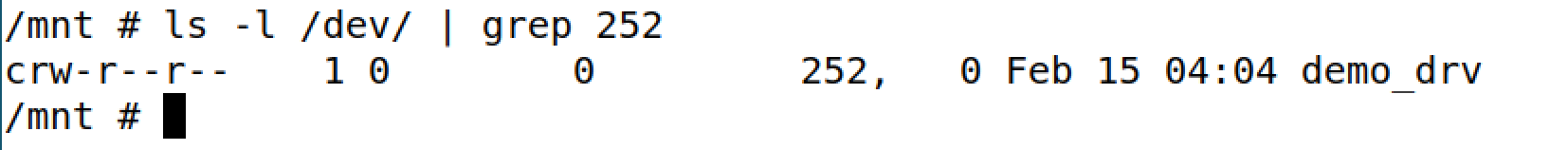
通过cat /proc/devices看到我们的设备的主设备号是252号，名称为my\_demo\_dev，但是到dev目录看看，的确是没有。



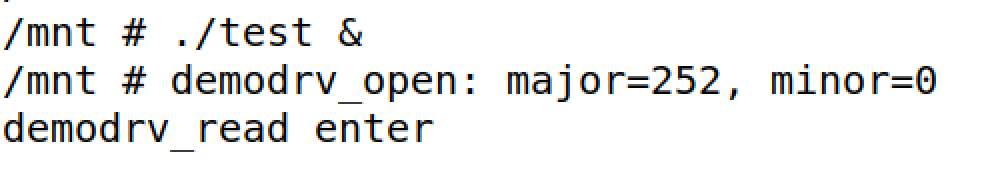
那我们就手动创建一个吧。



在查看一下dev目录。



这下有了。那我们在跑一下test程序。



这下打印了“demodrv\_read enter”，这是我们需要的，哈哈~~

测试成功了！

## 进阶思考

1. 大家可以思考一下，这里为什么要手工创建设备节点？
2. 在Linux系统中，比如Centos，优麒麟ubuntu等系统，他们是怎么自动创建新的设备节点的？

如果大家对这个问题感兴趣，可以关注笨叔的第一季旗舰篇视频，笨叔会在视频中和大家详细解答。

请关注奔跑吧Linux社区微信公众号，里面有更多更有料更好玩的Linux干货：



奔跑吧Linux社区的旗舰篇视频节目，帮您打通Linux和职场任督二脉！





笨叔最新力作，全程约5小时高清，140多页ppt，8大实验，基于x86\_64的Centos 7.6和arm64，提供全套实验素材和环境。全面介绍kdump+crash在死机黑屏方面的实战应用，全部案例源自线上云服务器和嵌入式产品开发实际案例！

不能为客户和老板解决死机黑屏问题的都不是合格的linux运维或者研发攻城狮！





订阅视频请到淘宝店：<https://shop115683645.taobao.com>

