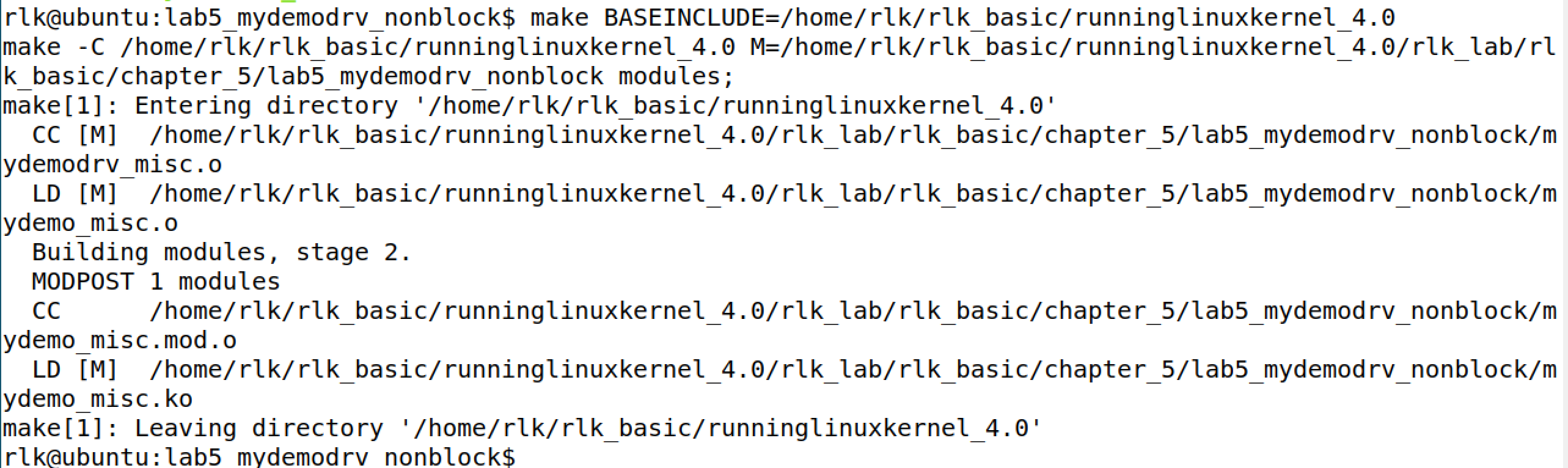
# Lab7：向虚拟设备中添加I/O多路复用支持

## 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 |

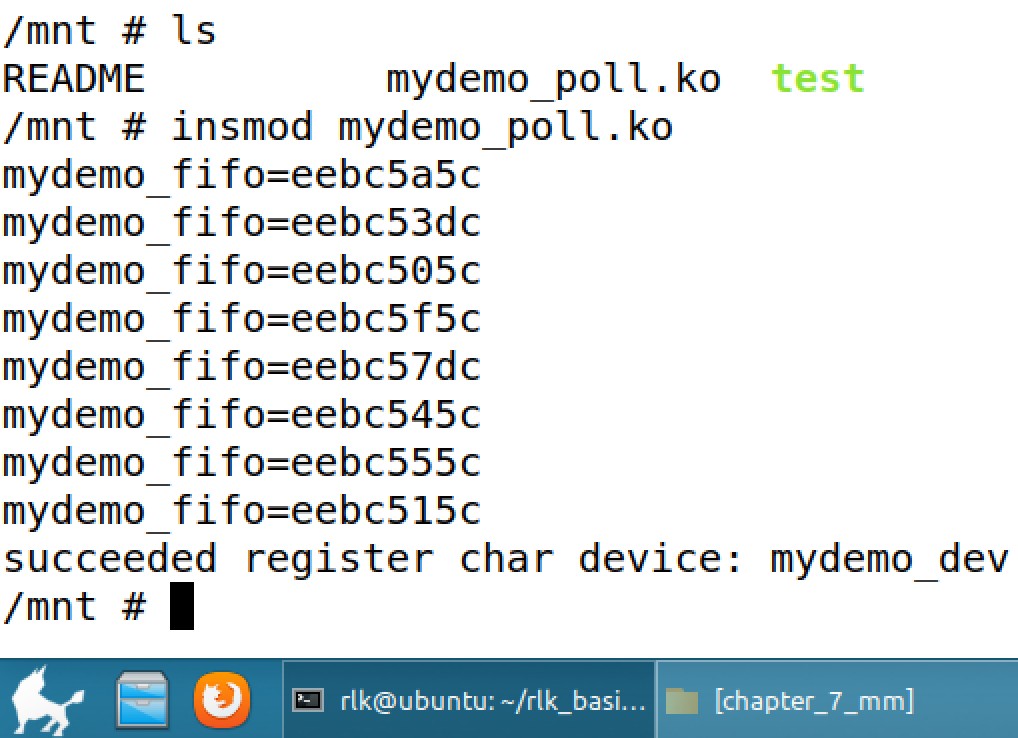
然后把ko内核模块和test程序拷贝到runninglinuxkernel\_4.0/kmodules目录下面。

运行如下脚本启动Qemu。

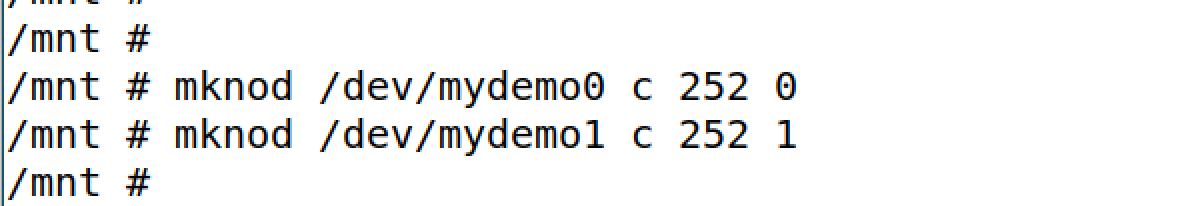
|  |
| --- |
| #cd runninglinuxkernel\_4.0  # sh run.sh arm32 #启动虚拟机，启动虚拟机之前，请先编译runninglinuxkernel\_4.0整个内核 |

在Qemu虚拟机:

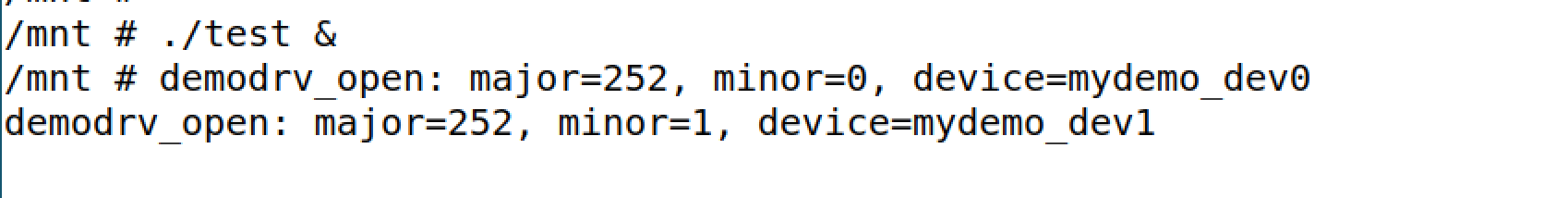
|  |
| --- |
| #cd /mnt  # insmod xxxx.ko //加载刚才编译的内核模块 |

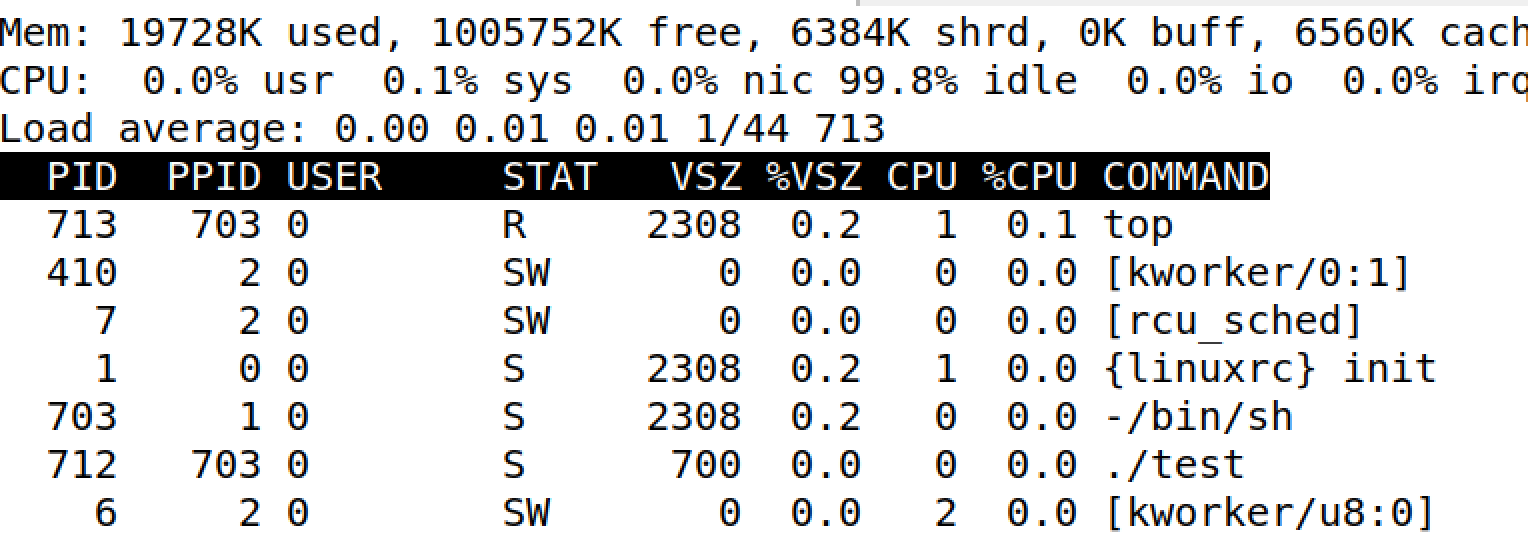


注意，这个实验，笨叔使用 传统的注册字符设备的方法，而没有使用misc机制，所以需要手工来创建设备节点。



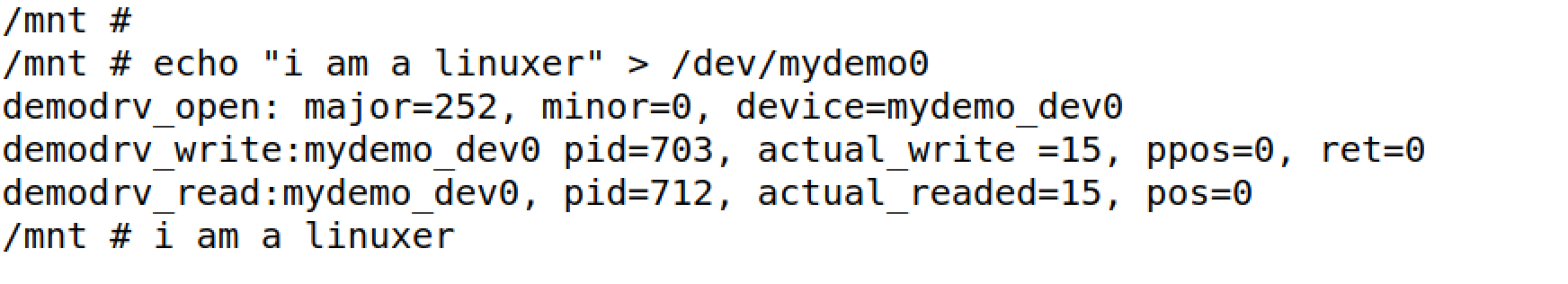
运行test程序，注意我们想让test程序在后台运行，所以添加了“&”，目的是让test进程来监听。





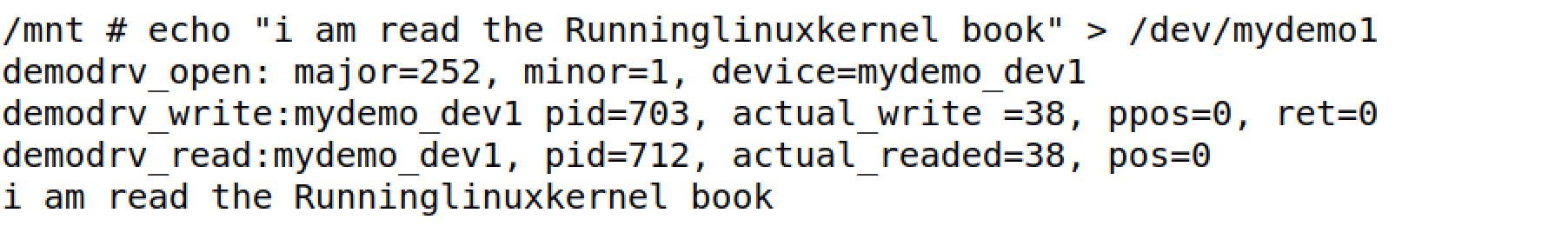
Test进程进入睡眠待命状态。

接下来使用echo来写，我们可以往设备0里写入字符串。



这时候可以看到test进程监听到了，并且打印出来。

下面我们试试往设备1写入数据。



同样，test进程可以监听到。

测试成功了！

## 进阶思考

字符设备的polling的方法是字符设备里高级的技巧和技能。

1. 大家可以去内核代码里看看，有哪些字符设备驱动使用了polling的方法？
2. 这个test程序，如果修改成select方法，如何修改？

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>

