

lab2_workqueue 实验说明

在第 5 章的虚拟 FIFO 设备中，我们并没有考虑多个进程同时访问设备驱动的情况，请使用互斥锁对虚拟 FIFO 设备驱动程序进行并发保护

基本实验步骤

1. 进入 `rlk_lab/rlk_basic/chapter_10/lab1` 目录。

```
# export ARCH=arm
# export CROSS_COMPILE=arm-linux-gnueabi-
# make BASEINCLUDE=/home/figo/work/runninglinuxkernel/runninglinuxkernel_4.0
```

这里 `BASEINCLUDE` 指定你当前 `runninglinuxkernel_4.0` 的目录路径。

编译 `test` 测试 app。

```
# arm-linux-gnueabi-gcc test.c -o test
```

然后把 `*.ko` 和 `test` 拷贝到 `runninglinuxkernel_4.0/kmodules` 目录下。

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_work.ko
```

```

/mnt # insmod mydemo_work.ko
my_class mydemo:252:0: create device: 252:0
mydemo_fifo=ee0b1f5c
my_class mydemo:252:1: create device: 252:1
mydemo_fifo=ee0b1e9c
my_class mydemo:252:2: create device: 252:2
mydemo_fifo=ee0b195c
my_class mydemo:252:3: create device: 252:3
mydemo_fifo=ee0b1c5c
my_class mydemo:252:4: create device: 252:4
mydemo_fifo=ee0b1b9c
my_class mydemo:252:5: create device: 252:5
mydemo_fifo=ee0bladc
my_class mydemo:252:6: create device: 252:6
mydemo_fifo=ee0b1a1c
my_class mydemo:252:7: create device: 252:7
mydemo_fifo=ee0b17dc
succeeded register char device: mydemo_dev

```

你会看到创建了 8 个设备。你可以到 `/sys/class/my_class/` 目录下面看到这些设备。

```

/mnt # cd /sys/class/my_class/
/sys/class/my_class # ls
mydemo:252:0  mydemo:252:2  mydemo:252:4  mydemo:252:6
mydemo:252:1  mydemo:252:3  mydemo:252:5  mydemo:252:7
/sys/class/my_class #

```

我们可以看到创建了主设备号为 252 的设备。我们再来看一下 `/dev/` 目录。

```

/sys/class/my_class # ls -l /dev
total 0
crw-rw----  1 0      0          14,   4 Feb  1 09:46 audio
crw-rw----  1 0      0           5,   1 Feb  1 09:46 console
crw-rw----  1 0      0          10,  63 Feb  1 09:46 cpu_dma_latency
crw-rw----  1 0      0          14,   3 Feb  1 09:46 dsp
crw-rw----  1 0      0          29,   0 Feb  1 09:46 fb0
crw-rw----  1 0      0          29,   1 Feb  1 09:46 fb1
crw-rw----  1 0      0           1,   7 Feb  1 09:46 full
crw-rw----  1 0      0          10, 183 Feb  1 09:46 hwrng
drwxr-xr-x  2 0      0          120 Feb  1 09:46 input
crw-rw----  1 0      0           1,   2 Feb  1 09:46 kmem
crw-rw----  1 0      0           1,  11 Feb  1 09:46 kmsg
crw-rw----  1 0      0           1,   1 Feb  1 09:46 mem
crw-rw----  1 0      0          10,  60 Feb  1 09:46 memory_bandwidth
crw-rw----  1 0      0          14,   0 Feb  1 09:46 mixer
crw-rw----  1 0      0          90,   0 Feb  1 09:46 mtd0

```

发现并没有主设备为 252 的设备。

所以我们需要手工创建一个设备用来 test app。

```
#mknod /dev/mydemo0 c 252 1
```

接下来跑我们的 test 程序：

```
# ./test & #这里让 test 程序在后台跑
```

```
/mnt # ./test &  
/mnt # my_class mydemo:252:1: demodrv_open: major=252, minor=1, device=mydemo_dev1  
my_class mydemo:252:1: demodrv_fasync send SIGIO
```

然后使用 echo 命令来往/dev/mydemo0 这个设备写入字符串。

```
/mnt #  
/mnt # echo "i am learning runninglinuxkernel" > /dev/mydemo0  
my_class mydemo:252:1: demodrv_open: major=252, minor=1, device=mydemo_dev1  
demodrv_write kill fasync  
my_class mydemo:252:1: demodrv_write:mydemo_dev1 pid=703, actual_write =33, ppos=0, ret=0  
my_class mydemo:252:1: do_work: trigger a work  
FIFO is not empty  
my_class mydemo:252:1: demodrv_read:mydemo_dev1, pid=741, actual_readed=33, pos=0  
i am learning runninglinuxkernel  
  
FIFO is not full
```

可以看到从 workqueue 的回调函数打印的一句话 “do_work: trigger a work”。

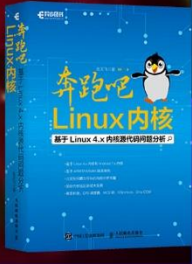
进阶思考

如果大家对这些问题感兴趣，可以关注笨叔的旗舰篇视频，笨叔会在视频中和大家详细解答。

shop115683645.taobao.com

配套视频 **旗舰篇**

第**1**季
内存管理



奔跑吧
LINUX 社区

旗舰篇一次订阅，持续更新

规划中	第二季	进程管理和调度 / 中断 / 锁（已出）
	第三季	虚拟化
	第四季	Linux 内核和应用开发调试必杀技
	第五季	红帽系列

shop115683645.taobao.com

Linux视频课程



微信公众号：奔跑吧 linux 社区

1. > 一键订阅，持续更新
2. > 最有深度和广度的 Linux 视频
3. > 手把手解读 Linux 内核代码
4. > 紧跟 Linux 开源社区技术热点
5. > 笨叔叔的 VIP 私密群答疑
6. > 图书 + 视频，全新学习模式

微店：

微店



奔跑吧Linux内核



扫码识别

淘宝店：<https://shop115683645.taobao.com/>

微信公众号：

