GEM5 Simulator in Full System Mode(2)

Yizi Gu

Tsinghua University

yizigu@gmail.com

October 27, 2014

Overview

1. Add Kernel Module
   - The problem met last week
   - A basic character device driver
2. Running benchmark
   - Important paths
   - Shortcut for running benchmarks

Inconsistent Version Magic

Problem:

```
# insmod hello.ko
insmod hello.ko
[  158.631323] hello: version magic '2.6.38-rc8 mod_unload ARMv6 ' should be '2.6.38.8-gem5
SMP mod_unload ARMv7 '
insmod: can't insert 'hello.ko': invalid module format
```

Figure : Inconsistent version magic

Solution: Modify the macro in the /include/linux/vermagic.h

```
#define MODULE_ARCH_VERMAGIC_MINE "ARMv7 "

#define VERMAGIC_STRING                                 \
        "2.6.38.8-gem5" " "                                     \
        MODULE_VERMAGIC_SMP MODULE_VERMAGIC_PREEMPT                     \
        MODULE_VERMAGIC_MODULE_UNLOAD MODULE_VERMAGIC_MODVERSIONS       \
        MODULE_ARCH_VERMAGIC_MINE
```

Figure : Modify the version magic

Cont'd

Result:

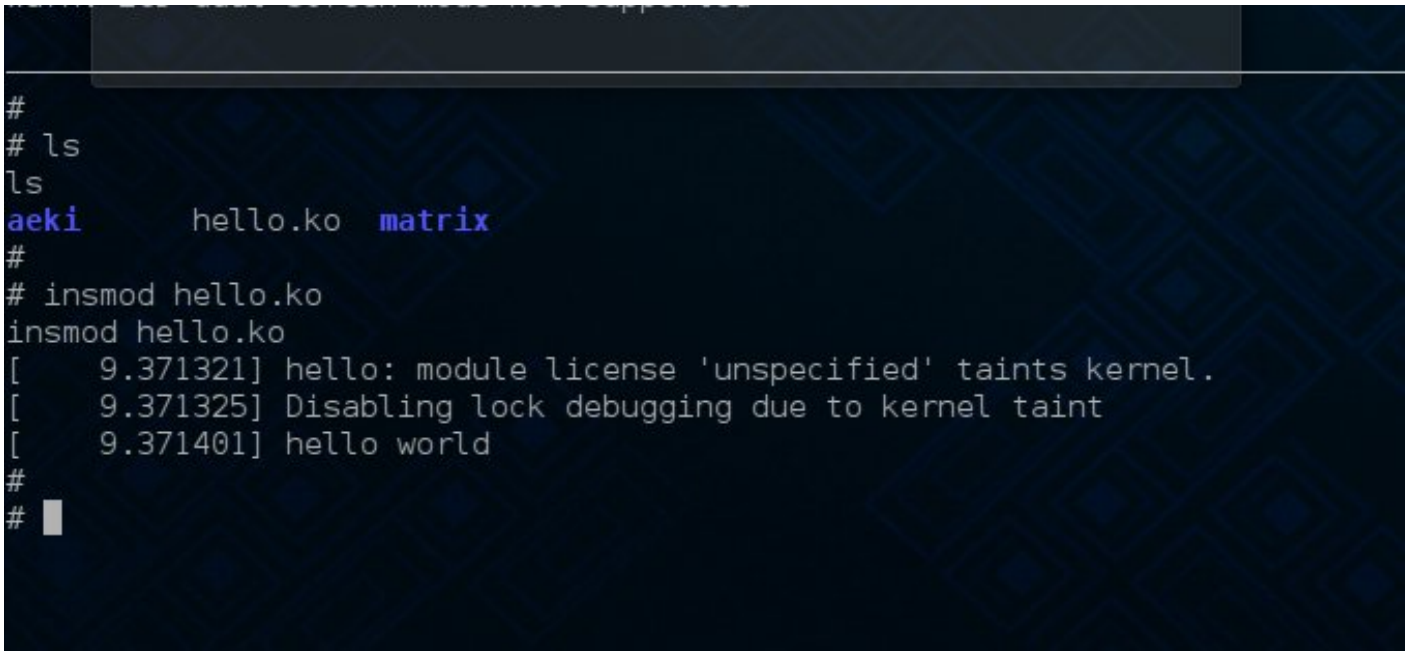

Figure : Successfully inserted the module

A basic character device driver

A simple device driver that manages read requests.

```
# ls
ls
a.out       aeki        chardev.ko  hello.ko    matrix
#
# insmod chardev.ko
insmod chardev.ko
[  187.041470] chardev: module license 'unspecified' taints kernel.
[  187.041474] Disabling lock debugging due to kernel taint
[  187.041634] I was assigned major number 254. To talk to
[  187.041637] the driver, create a dev file with
[  187.041641] 'mknod /dev/chardev c 254 0'.
[  187.041644] Try various minor numbers. Try to cat and echo to
[  187.041648] the device file.
[  187.041650] Remove the device file and module when done.
#
# mknod /dev/chardev c 254 0
mknod /dev/chardev c 254 0
#
# cat /dev/chardev
cat /dev/chardev
I already told you 0 times Hello world!
#
# cat /dev/chardev
cat /dev/chardev
I already told you 1 times Hello world!
#
```

Figure : A basic character device driver

Important paths

- configs/common/SysPaths.py
  Configures the path to kernel and disk image.
- configs/common/FSConfig.py
  Configures full-system parameters: memory size,kernel version,etc.
- configs/common/Benchmarks.py
  Configures benchmarks.

```
Benchmarks = {
    'PovrayBench':  [SysConfig('povray-bench.rcS', '512MB', 'povray.img')],
    'PovrayAutumn': [SysConfig('povray-autumn.rcS', '512MB', 'povray.img')],

    'NetperfStream':    [SysConfig('netperf-stream-client.rcS'),
                         SysConfig('netperf-server.rcS')],
    'NetperfStreamUdp': [SysConfig('netperf-stream-udp-client.rcS'),
```

Figure : Benchmarks.py

The .rcS file

- The .rcS files are bash scripts for running benchmarks automatically when system starts.
- The .rcS file resides in configs/boot/.

Example

A typical .rcS file

```
#!/bin/sh
cd /benchmarks/spec/gzip00/
/sbin/m5 checkpoint 0 0
/sbin/m5 checkpoint 100000000 200000000
/sbin/m5 loadsymbol
/sbin/m5 resetstats
./gzip lgred.log 1
/sbin/m5 exit
```

Register the .rcS file

Add an entry to the Benchmarks.py:

Example

'MibenchGSM': [SysConfig('mibench-gsm.rcS','256MB')]

Figure : Run the benchmark automatically

Simple Runing Script

- We could run the benchmarks without invoking the terminal.
- Suitable for batch processing.

Script Example

```
#!bin/bash
sed -e "s@/opt/[^']*@/opt/arm-system@" \
    -i ./configs/common/SysPaths.py

./build/ARM/gem5.opt configs/example/fs.py \
    --benchmark=MibenchGSM

cat m5out/system.terminal
```

This week's work

- Study GEM5 source code and kernel module.
- Call for ideas.

Thank you