Operating Systems II

Programming Assignment 3 - README

Sandeep Kumar : CS18BTECH11041 | Vedant Singh: CS18BTECH11047

List of files submitted:

1. mykmod_main.c
2. mem_util.cpp
3. README.pdf
4. Report.pdf

How to Compile:

1. Enter the CentOS VM after copying the files into it and unzipping it.
2. Cd into the `99_devmmap_paging` folder
3. Type `make`
4. Type `insmod kernel/mykmod.ko` to load module.
5. Type `grep mykmod /proc/devices` to find out the major number. (Assume it is 243 here)
6. Run `mknod/tmp/<devname> c 243 <minorno>`
7. Run `./util/memutil [options] <devname>`
   Options:
   a. `--op <optype>` : Where optype can be mapread, mapwrite
   b. `--mes <message>` : Message to be written/read-compare to/from the device memory
   c. `--pt <ptype>` : Where ptype can be prefetch or demand
   d. `--help` : Show this help
8. Run `dmesg | grep -e mykmod_vm_open -e mykmod_vm_close` to display page fault information

Sample Outputs:

The memutil file being used here loops the reading/writing of the message repeatedly, thereby filling up the entire mmap-ed memory region. Below are some sample cases to depict the working of the mmap() call we have created.

Case: Demand Read with an empty message

mknod /tmp/mydev_JZI c 243 11
./util/memutil /tmp/mydev_JZI --pt demand --op mapread

```
# dmesg | grep -e mykmod_vm_open -e mykmod_vm_close
[36395.118615] mykmod_vm_open: vma=ffff890bb5b7c000 npagefaults:0
[36395.118632] mykmod_vm_close: vma=ffff890bb5b7c000 npagefaults:0
```

Case: Prefetch Read with an empty message

```
# mknod /tmp/mydev_JZl c 243 11
# ./util/memutil /tmp/mydev_JZl --pt prefetch --op mapread
# dmesg | grep -e mykmod_vm_open -e mykmod_vm_close
[  930.432198] mykmod_vm_open: vma=ffff8dac372d4ca8 npagefaults:0
[  930.433898] mykmod_vm_close: vma=ffff8dac372d4ca8 npagefaults:256
```

Case : Prefetch Write and Read from separate processes into the same device file

```
# mknod /tmp/mydev_fBc c 243 20
# ./util/memutil /tmp/mydev_fBc --pt prefetch --op mapwrite --mes test
# dmesg | grep -e mykmod_vm_open -e mykmod_vm_close
[35839.250221] mykmod_vm_open: vma=ffff890bb5bb3360 npagefaults:0
[35839.256442] mykmod_vm_close: vma=ffff890bb5bb3360 npagefaults:256
# ./util/memutil /tmp/mydev_fBc --pt prefetch --op mapread --mes test
# dmesg | grep -e mykmod_vm_open -e mykmod_vm_close
[36091.949897] mykmod_vm_open: vma=ffff890bb3c26bd0 npagefaults:0
[36091.953223] mykmod_vm_close: vma=ffff890bb3c26bd0 npagefaults:256
```

Case : Demand Read and Write in same process from same device file

```
# mknod /tmp/mydev_JZl c 243 11
# ./util/memutil /tmp/mydev_JZl --pt demand --op mapwrite --op mapread --mes test
# dmesg | grep -e mykmod_vm_open -e mykmod_vm_close
[34246.939579] mykmod_vm_open: vma=ffff890bb3c70d80 npagefaults:0
[34246.952410] mykmod_vm_close: vma=ffff890bb3c70d80 npagefaults:256
[34246.952432] mykmod_vm_open: vma=ffff890bb3c70d80 npagefaults:0
[34246.963559] mykmod_vm_close: vma=ffff890bb3c70d80 npagefaults:256
```

Case : Demand Read and Write from separate processes into different device special files

```
# mknod /tmp/mydev_JZl c 243 11
# mknod /tmp/mydev_pR6 c 243 10
# ./util/memutil /tmp/mydev_JZl --pt demand --op mapwrite --mes test
# ./util/memutil /tmp/mydev_pR6 --pt demand --op mapread
# dmesg | grep -e mykmod_vm_open -e mykmod_vm_close
[ 3361.086845] mykmod_vm_open: vma=ffff8b1a38db7ca8 npagefaults:0
[ 3361.089435] mykmod_vm_close: vma=ffff8b1a38db7ca8 npagefaults:256
[ 3366.804937] mykmod_vm_open: vma=ffff8b1a38db7870 npagefaults:0
[ 3366.804957] mykmod_vm_close: vma=ffff8b1a38db7870 npagefaults:0
```