

ASSIGNMENT-3

REPORT

PLAGIARISM STATEMENT:-

We certify that this assignment/report is our own work, based on our personal study and/or research and that we have acknowledged all material and sources used in its preparation, whether they be books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. We also certify that this assignment/report has not previously been submitted for assessment in any other course, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that we have not copied in part or whole or otherwise plagiarised the work of other students and/or persons. We pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, We understand our responsibility to report honour violations by other students if we become aware of it.

Names: J.Deepak Reddy, A.Venkata Sai Mahesh

Date: 21/6/2020

Signatures: Abburi Venkata Sai Mahesh, Jillela Deepak Reddy

IMPORTANT NOTE:-

- This report only mentions 'TODO' parts of the assignment only.

UTILITY PROGRAM:-

1. For the demand mode I have set the mmap_flag to MAP_SHARED and for the prefetch mode I have set the mmap_flag to MAP_POPULATE | MAP_SHARED. Refer <https://man7.org/linux/man-pages/man2/mmap.2.html> for the reason.
2. While calling mmap I have also taken care of failing cases and returned EXIT_FAILURE as said in the document.
3. For the MAPREAD and MAPWRITE modes I have set the flag to PROT_READ and PROT_WRITE respectively.
4. For filling all the memory with the given short message, I have printed recursively the same message using the modulus operator.
5. I have Also taken care if the message len is zero(no message is given) then no.of page faults should be zero.

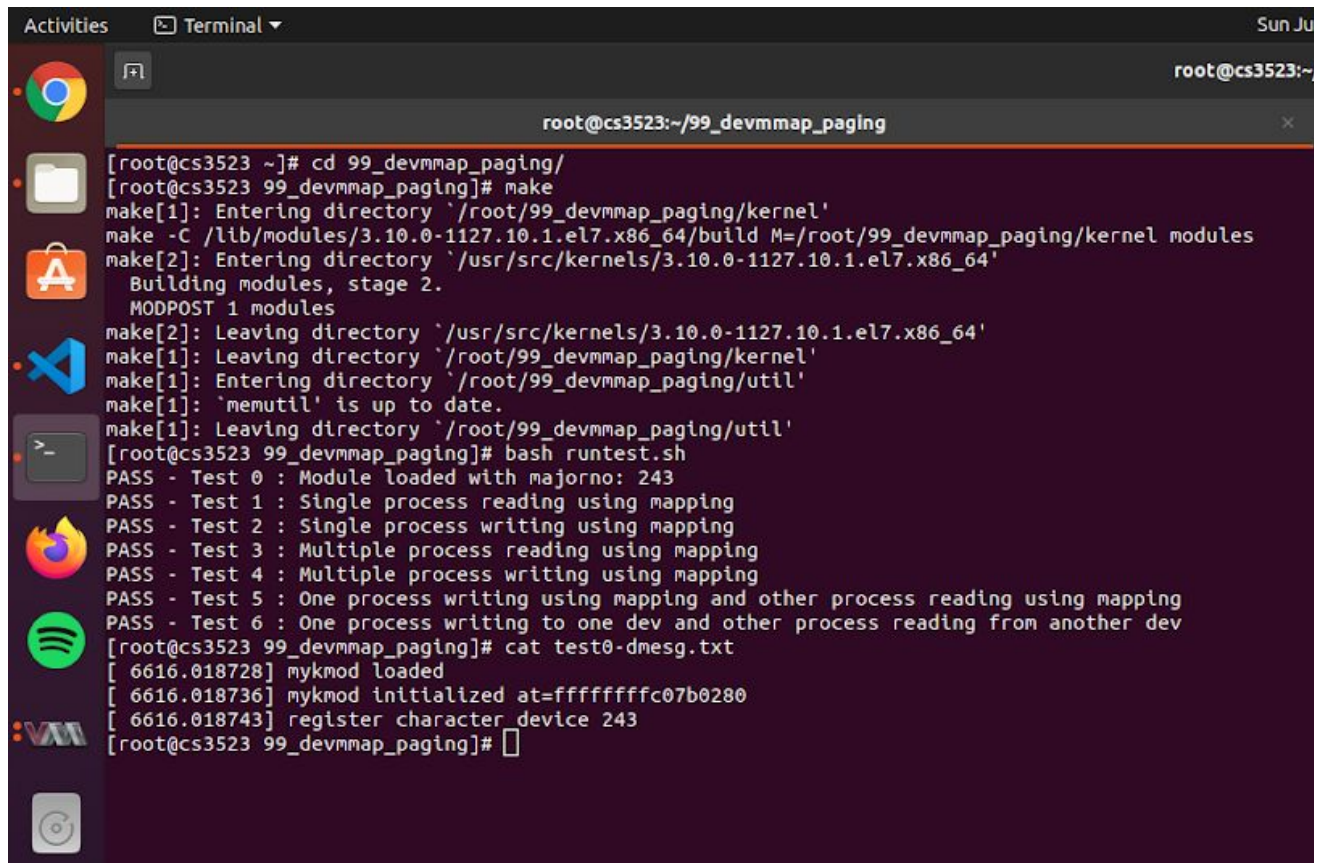
DEVICE DRIVER:-

1. 'struct mykmod_dev_info' is used for storing info about a device whose structure contains 'char* data' which stores the device name and 'size_t size' for storing the size of the driver.
2. As the maximum number of devices has been defined initially, I have just considered an array of mykmod_dev_info struct pointers to store all the devices information.
3. For the VMA info I have considered a structure with a variable to store device information and an unsigned long integer to store the number of page faults for that device. I have used a global pointer which points to the present running device.
4. In the init_module I have allocated space in kernel memory to store each device information.
5. In the cleanup_module I check whether the device is empty or not and delete the non-empty devices.
6. In the mykmod_open I have allocated memory for devinfo and stored it in the device table and i_private. Here I have stored using the MINOR value of the device to showcase the reality. We can even just allocate a random number, but I have thought that this would be better in avoiding the same indexes.
7. In the mykmod_close module I free the memory allocated to the current device using the globally declared pointer which points to the current device.
8. In the mykmod_mmap I have set up the vma's flags, saved the private data (device info, npagefaults) in vm_private_data.
9. In the mykmod_vm_open and mykmod_vm_close I considered a temp pointer to fetch the vma private information used in printing and setting the npagefaults value. As described in the document I have initialised npagefaults to 0 when VM segment is opened and re-initialised npagefaults to 0 after printing and before closing.
10. In the mykmod_vm_fault I fetched the required data from the vm_struct and checked the conditions for using virt_to_page() function(there should be at least some data associated with it and the page offset should be greater than virtual address). After all these conditions are satisfied, I have increased the no.of page faults and setted the vma->page to the page we got after paging. For reference read the link below.

Sources:

https://www.quora.com/What-is-the-difference-between-using-virt_to_page-and-doing-a-page-table-walk-for-a-kernel-virtual-address

Sample Output:-



```
Activities  Terminal  Sun Ju
root@cs3523:~/99_devmmap_paging

[root@cs3523 ~]# cd 99_devmmap_paging/
[root@cs3523 99_devmmap_paging]# make
make[1]: Entering directory '/root/99_devmmap_paging/kernel'
make -C /lib/modules/3.10.0-1127.10.1.el7.x86_64/build M=/root/99_devmmap_paging/kernel modules
make[2]: Entering directory '/usr/src/kernels/3.10.0-1127.10.1.el7.x86_64'
Building modules, stage 2.
MODPOST 1 modules
make[2]: Leaving directory '/usr/src/kernels/3.10.0-1127.10.1.el7.x86_64'
make[1]: Leaving directory '/root/99_devmmap_paging/kernel'
make[1]: Entering directory '/root/99_devmmap_paging/util'
make[1]: 'memutil' is up to date.
make[1]: Leaving directory '/root/99_devmmap_paging/util'
[root@cs3523 99_devmmap_paging]# bash runtest.sh
PASS - Test 0 : Module loaded with majorno: 243
PASS - Test 1 : Single process reading using mapping
PASS - Test 2 : Single process writing using mapping
PASS - Test 3 : Multiple process reading using mapping
PASS - Test 4 : Multiple process writing using mapping
PASS - Test 5 : One process writing using mapping and other process reading using mapping
PASS - Test 6 : One process writing to one dev and other process reading from another dev
[root@cs3523 99_devmmap_paging]# cat test0-dmesg.txt
[ 6616.018728] mykmod loaded
[ 6616.018736] mykmod initialized at=ffffffffc07b0280
[ 6616.018743] register character device 243
[root@cs3523 99_devmmap_paging]#
```