NAME: Soham Sarkar
SRN: PES2UG21CS532
Section: I

Project Title: Linux kernel module that lists all the currently running processes in the system and their corresponding states.

Github link for my Linux kernel module

This is a simple Linux kernel module written in C programming language that lists all the currently running processes in the system and their corresponding states.

How to run the kernel code?

0. Execute the following command to install all the header files and dependencies required for the kernel module.

sudo apt-get install build-essential linux-headers-$(uname -r)

1. Execute the `make` command to create the `.ko` file along with other files.
 [P.S: The makefile and the main.c file should be in the same directory(at the same directory level)]

make

```
sohoxic@pop-os:~/Music$ make
make -C /lib/modules/6.2.6-76060206-generic/build M=/home/sohoxic/Music modules
make[1]: Entering directory '/usr/src/linux-headers-6.2.6-76060206-generic'
warning: the compiler differs from the one used to build the kernel
  The kernel was built by: x86_64-linux-gnu-gcc-12 (Ubuntu 12.1.0-2ubuntu1~22.04
) 12.1.0
  You are using:           gcc-12 (Ubuntu 12.1.0-2ubuntu1~22.04) 12.1.0
  CC [M]  /home/sohoxic/Music/main.o
  MODPOST /home/sohoxic/Music/Module.symvers
  CC [M]  /home/sohoxic/Music/main.mod.o
  LD [M]  /home/sohoxic/Music/main.ko
  BTF [M] /home/sohoxic/Music/main.ko
Skipping BTF generation for /home/sohoxic/Music/main.ko due to unavailability of
 vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-6.2.6-76060206-generic'
```

2. Insert module into the kernel using `insmod`.

sudo insmod main.ko

```
sohoxic@pop-os:~/Music$ sudo insmod main.ko
[sudo] password for sohoxic:
```

3. To check if the kernel module is inserted into the kernel at runtime execute the `lsmod` command. This shows which loadable kernel modules are currently loaded(we can see the main module at the top.)

lsmod

```
sohoxic@pop-os:~/Music$ lsmod
Module                  Size  Used by
main                   16384  0
```

```
sohoxic@pop-os:~/Music$ lsmod
Module                  Size  Used by
main                   16384  0
tls                   147456  0
ccm                    20480  6
rfcomm                 98304  4
snd_seq_dummy          16384  0
snd_hrtimer            16384  1
cmac                   16384  3
algif_hash             20480  1
algif_skcipher         16384  1
af_alg                 32768  6 algif_hash,algif_skcipher
zstd                   16384  12
bnep                   32768  2
snd_ctl_led            24576  0
zram                   45056  2
snd_soc_dmic           16384  1
snd_acp3x_pdm_dma      16384  1
```

4. Execute `modinfo` command to display information about the kernel module.

modinfo main.ko

```
sohoxic@pop-os:~/Music$ modinfo main.ko
filename:       /home/sohoxic/Music/main.ko
author:         Soham Sarkar, PES2UG21CS532
description:    Listing tasks
license:        GPL
srcversion:     35A50C6AB09DD597613F070
depends:
retpoline:      Y
name:           main
vermagic:       6.2.6-76060206-generic SMP preempt mod_unload modversions
```

5. `dmesg` is a useful command-line tool that provides a convenient way to access and analyze kernel messages, making it an important tool for system administration and troubleshooting in Linux/Unix systems.

sudo dmesg

```
sohoxic@pop-os:~/Music$ sudo dmesg
[    0.000000] Linux version 6.2.6-76060206-generic (jenkins@warp.pop-os.org) (x86_64-linux-gn
0630~1681329778~22.04~d824cd4 SMP PREEMPT_DYNAMIC Wed A
[    0.000000] Command line: initrd=\EFI\Pop_OS-8d9537c4-78d4-4ecb-a8f8-31f812f23c51\initrd.im
plash
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   Hygon HygonGenuine
[    0.000000]   Centaur CentaurHauls
[    0.000000]   zhaoxin   Shanghai
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[    0.000000] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[    0.000000] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'compact
[    0.000000] signal: max sigframe size: 1776
[    0.000000] BIOS-provided physical RAM map:
[    0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009ffff] usable
[    0.000000] BIOS-e820: [mem 0x00000000000a0000-0x00000000000fffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000000100000-0x0000000009eeffff] usable
[    0.000000] BIOS-e820: [mem 0x0000000009ef0000-0x000000000a000fff] reserved
[    0.000000] BIOS-e820: [mem 0x000000000a001000-0x000000000a1fffff] usable
[    0.000000] BIOS-e820: [mem 0x000000000a200000-0x000000000a20bfff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x000000000a20c000-0x00000000c5f66fff] usable
[    0.000000] BIOS-e820: [mem 0x00000000c5f67000-0x00000000c60bcfff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000c60bd000-0x00000000c612ffff] ACPI data
[    0.000000] BIOS-e820: [mem 0x00000000c6130000-0x00000000c76e4fff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x00000000c76e5000-0x00000000ccffdfff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000ccffe000-0x00000000cdffffff] usable
[    0.000000] BIOS-e820: [mem 0x00000000ce000000-0x00000000cfffffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000f0000000-0x00000000f7ffffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fd000000-0x00000000fdffffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fe700000-0x00000000fe700fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000feb80000-0x00000000fec01fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fec10000-0x00000000fec10fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fed00000-0x00000000fed00fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fed40000-0x00000000fed44fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fed80000-0x00000000fed8ffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fedc4000-0x00000000fedc9fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fedcc000-0x00000000fedcefff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fedd5000-0x00000000fedd5fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000ff000000-0x00000000ffffffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000100000000-0x000000020f33ffff] usable
[    0.000000] BIOS-e820: [mem 0x000000020f340000-0x000000022fffffff] reserved
[    0.000000] NX (Execute Disable) protection: active
[    0.000000] e820: update [mem 0xc2077018-0xc2084857] usable ==> usable
[    0.000000] e820: update [mem 0xc2077018-0xc2084857] usable ==> usable
```

```
1413.957605] Process: kworker/3:0      PID:[6590]      State:Unknown Type:1026
1413.957606] Process: kworker/u24:6    PID:[6639]      State:Unknown Type:1026
1413.957608] Process: kworker/u24:7    PID:[6640]      State:Unknown Type:1026
1413.957609] Process: gnome-terminal-  PID:[6695]      State:TASK_INTERRUPTIBL
1413.957610] Process: bash     PID:[6719]      State:TASK_INTERRUPTIBLE
1413.957611] Process: kworker/u24:8    PID:[7002]      State:Unknown Type:1026
1413.957613] Process: kworker/u24:9    PID:[7003]      State:Unknown Type:1026
1413.957614] Process: sudo     PID:[7007]      State:TASK_INTERRUPTIBLE
1413.957615] Process: sudo     PID:[7008]      State:TASK_INTERRUPTIBLE
1413.957616] Process: insmod   PID:[7009]      State:TASK_RUNNING
1413.957617] Number of processes:369
```

6. Now remove the kernel module from the kernel using `rmmod` and then using `lsmod` to view that the module is no longer part of the kernel. [P.S - Only check the first line after executing lsmod to see whether the kernel module name `main` is there or not.]

sudo rmmod main

```
sohoxic@pop-os:~/Music$ sudo rmmod main
sohoxic@pop-os:~/Music$ lsmod
Module                  Size  Used by
tls                   147456  0
ccm                    20480  6
```

Code Explanation

- The module first includes the necessary header files, which contain definitions of various data structures and functions used in the module.
- The get_task_state function takes a process state as an argument and returns a string representation of that state. It does this by using a switch statement to match the state with one of the pre-defined constants, and returns a string representation of that constant. If the state is not recognized, it generates an "Unknown Type" message and returns it in the buffer.
- The test_tasks_init function is the entry point for the module, which is called when the module is loaded. It declares a pointer to the task_struct data structure, which represents a process in the Linux kernel. It then iterates over all processes in the system using the for_each_process macro, which iterates over a linked list of processes. For each process, it prints the process name, process ID, and process state by calling the pr_info function. Finally, it prints the total number of processes found.
- The test_tasks_exit function is the exit point for the module, which is called when the module is unloaded. It simply prints a message indicating that the module is being unloaded.
- The module is licensed under the GPL, has a description and author information, and specifies the entry and exit points of the module using the module_init and module_exit macros.