

Міністерство освіти та науки України
Національний технічний університет України
“Київський політехнічний інститут ім. Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки

Лабораторна робота №5

з дисципліни “Технології програмування для комп'ютерних систем – 3”

Виконав:
студент групи ІВ-91мн
Гончаренко Олександр

Київ 2020 р.

Завдання

Завдання розраховане на вже виконане завдання #4, і полягає у модифікації реалізації того завдання.

Зауваження:

I. Для відповідності Linux kernel coding style ознайомтеся зі скриптом `checkpatch.pl`.

A. Приклад використання `$KDIR/scripts/checkpatch.pl -f файл.c`

B. Позначеного `ERROR` не повинно бути взагалі. З позначеного `WARNING` частина вас ще не стосується, проте простих речей на зразок “не треба на початку рядка ставити пробіл, а за ним табуляцію” також не повинно залишатися.

II. При виконанні роботи, не забувайте виконувати необхідні перевірки.

Завдання Basic:

I. Оголосити структуру даних для розміщення у списку, яка крім елемента `struct list_head` містить поле типу `ktime_t` (`include/linux/ktime.h` у вашому репозиторії `linux-stable`).

II. Створити статичну змінну голови списку.

III. Перед кожним друком привітання виділити пам'ять для екземпляра оголошеної структури, занести в неї поточний час ядра, отриманий функцією `ktime_get()`.

IV. У функції `hello_exit()` пройти по списку і надрукувати час кожної події в наносекундах, вилучити елемент списку і звільнити виділену пам'ять. Приклад проходження по списку з вилученням елемента є у *appendix*.

На даному етапі досить виділяти пам'ять викликом

```
ptr = kmalloc(sizeof(*ptr), GFP_KERNEL);
```

і звільнити її викликом

```
kfree(ptr);
```

Докладніше у додатку *Allocating Memory*.

Завдання Advanced:

У цьому завданні вся робота зі списком виконується в модулі `hello1`. Також додати ще одне поле типу `ktime_t` і засікати час до та після виклику функції друку, а на вивантаженні модуля надрукувати час, який пішов на кожен друк.

Послідовність виконання роботи

1. **Створюємо модуль.** Для цього у файловій системі створюємо директорію `lab5`, у яку додаємо усі необхідні файли для роботи з модулем. Виконуємо команди `export` для

збирання ядра, а також `export KDIR=/home/alex/lab3/linux-stable/`. Виконуємо команду `make` для збирання модулю. Створимо архів `CPIO` для `rootfs` та заархівуємо його за допомогою `GZip`.

```
alex@alex-VirtualBox: ~/lab3/busybox
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ ll
total 20
drwxrwxr-x 2 alex alex 4096 tpa 27 21:55 ./
drwxrwxr-x 15 alex alex 4096 tpa 27 21:44 ../
-rw-rw-r-- 1 alex alex 46 tpa 27 21:46 Kbuild
-rw-rw-r-- 1 alex alex 135 tpa 27 21:47 Makefile
-rw-rw-r-- 1 alex alex 1472 tpa 27 21:55 module4.c
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ export PATH=/opt/gcc-arm-8.3-2019.03-x86_64-arm-eabi/bin:$PATH
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ export CROSS_COMPILE='ccache arm-eabi-'
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ export ARCH=arm
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ export KDIR=/home/alex/lab3/linux-stable/
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ make
make -C /home/alex/lab3/linux-stable/ M=$PWD
make[1]: Entering directory '/home/alex/lab3/linux-stable'
CC [M] /home/alex/lab3/busybox/_install/lab5/module4.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/alex/lab3/busybox/_install/lab5/module4.mod.o
LD [M] /home/alex/lab3/busybox/_install/lab5/module4.ko
make[1]: Leaving directory '/home/alex/lab3/linux-stable'
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ cd ..
alex@alex-VirtualBox:~/lab3/busybox/_install$ find . | cpio -o -H newc | gzip > ../rootfs.cpio.gz
89988 blocks
alex@alex-VirtualBox:~/lab3/busybox/_install$ cd ..
alex@alex-VirtualBox:~/lab3/busybox$ qemu-system-arm -kernel _install/boot/zImage -initrd rootfs.cpio.gz -machine virt -nographic -m 512 --append "root=/dev/ram0 rw console=ttyAMA0,115200 mem=512M"
```

2. Перевірка Linux kernel coding style:

```
alex@alex-VirtualBox:~/lab3/busybox/_install/lab5$ ~/lab3/linux-stable/scripts/checkpatch.pl -f module4.c
WARNING: Missing or malformed SPDX-License-Identifier tag in line 1
#1: FILE: module4.c:1:
+#include <linux/init.h>

WARNING: line over 80 characters
#59: FILE: module4.c:59:
+         pr_info("Time needed for printing is: %lld(ns).\n", curr->time_before);

total: 0 errors, 2 warnings, 70 lines checked

NOTE: For some of the reported defects, checkpatch may be able to
      mechanically convert to the typical style using --fix or --fix-inplace.

module4.c has style problems, please review.

NOTE: If any of the errors are false positives, please report
      them to the maintainer, see CHECKPATCH in MAINTAINERS.
```

3. Виконання завдання Basic:

Оголосити структуру даних для розміщення у списку, яка крім елемента `struct list_head` містить поле типу `ctime_t` (include/linux/ctime.h у вашому репозиторії linux-stable).

Створити статичну змінну голови списку.

Перед кожним друком привітання виділити пам'ять для екземпляра оголошеної структури, занести в неї поточний час ядра, отриманий функцією `ctime_get()`.

У функції `hello_exit()` пройти по списку і надрукувати час кожної події в наносекундах, вилучити елемент списку і звільнити виділену пам'ять. Приклад проходження по списку з вилученням елемента є у *appendix*.



```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/printk.h>
#include <linux/ctime.h>
#include <linux/slab.h>

static LIST_HEAD(lab5_list_head);

struct time_keeper {
    ctime_t time_before;
    struct list_head time_list;
};

static void print_text(unsigned int repeats)
{
    unsigned int repeat;
    struct time_keeper *ptr;

    for (repeat = 0; repeat < repeats; repeat++) {
        ptr = kmalloc(sizeof(*ptr), GFP_KERNEL);
        ptr->time_before = ctime_get();
        pr_info("Hello, world!\n");
        list_add(&ptr->time_list, &lab5_list_head);
    }
}

static unsigned int repeats = 1;

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "Hello print:");

static int __init module4_init(void)
{
    if (repeats > 10) {
        pr_err("Cannot repeat more than 10 times\n");
        return -EINVAL;
    }

    if (repeats >= 5 && repeats <= 10)
        pr_warn("Repeatition from 5 to 10 times\n");

    if (repeats == 0)
        pr_warn("No repeatition\n");

    print_text(repeats);
}
```

```
Activities Terminal tpa 27 21:54
alex@alex-VirtualBox: ~/lab3/busybox/_i

static unsigned int repeats = 1;
module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "Hello print:");

static int __init module4_init(void)
{
    if (repeats > 10) {
        pr_err("Cannot repeat more than 10 times\n");
        return -EINVAL;
    }

    if (repeats >= 5 && repeats <= 10)
        pr_warn("Repeation from 5 to 10 times\n");

    if (repeats == 0)
        pr_warn("No repeation\n");

    print_text(repeats);
    return 0;
}

static void __exit module4_exit(void)
{
    struct list_head *p;
    struct list_head *n;
    struct time_keeper *curr;

    pr_info("Bye-Bye!\n");

    list_for_each_safe(p, n, &lab5_list_head) {
        curr = list_entry(p, struct time_keeper, time_list);
        pr_info("Time needed for printing is: %lld(ns).\n", curr->time_before);
        list_del(p);
        kfree(curr);
    }
}

module_init(module4_init);
module_exit(module4_exit);

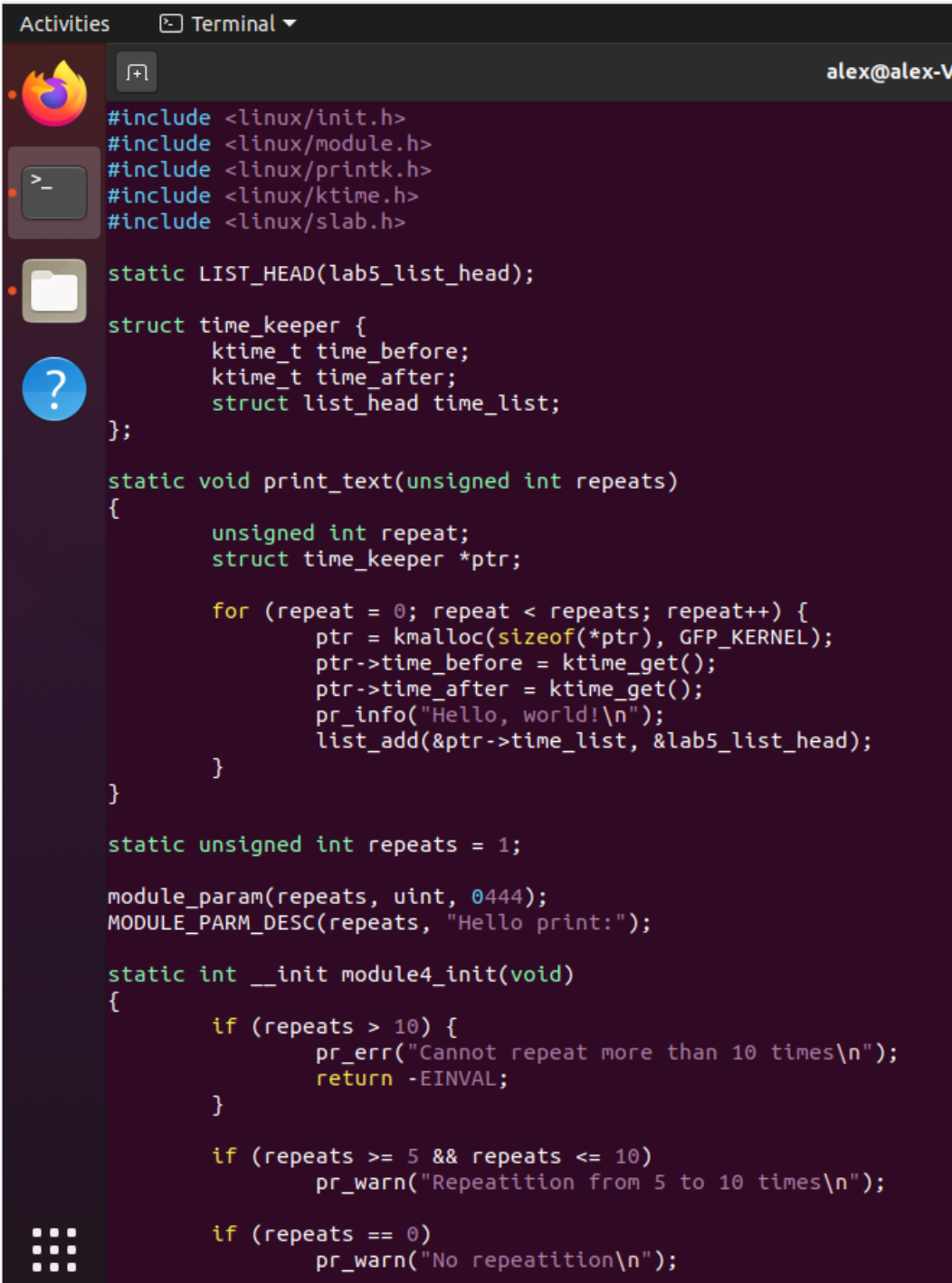
MODULE_AUTHOR("AlexHoncharenko");
MODULE_DESCRIPTION("Training to work with list");
MODULE_LICENSE("Dual BSD/GPL");
```

4. Тест роботи модуля:

```
Please press Enter to activate this console.
/ # cd lab5/
/lab5 # insmod module4.ko repeats=4
[ 48.869455] module4: loading out-of-tree module taints kernel.
[ 48.919711] Hello, world!
[ 48.925135] Hello, world!
[ 48.926314] Hello, world!
[ 48.928028] Hello, world!
/lab5 # rmmod module4.ko
[ 59.521174] Bye-Bye!
[ 59.522806] Time needed for printing is: 48632520960(ns).
[ 59.525441] Time needed for printing is: 48630818896(ns).
[ 59.528329] Time needed for printing is: 48629635952(ns).
[ 59.529236] Time needed for printing is: 48624168480(ns).
```

5. Виконання завдання Advanced:

У цьому завданні вся робота зі списком виконується в модулі hello1. Також додати ще одне поле типу `ktime_t` і засікати час до та після виклику функції друку, а на вивантаженні модуля надрукувати час, який пішов на кожен друк.



```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/printk.h>
#include <linux/ktime.h>
#include <linux/slab.h>

static LIST_HEAD(lab5_list_head);

struct time_keeper {
    ktime_t time_before;
    ktime_t time_after;
    struct list_head time_list;
};

static void print_text(unsigned int repeats)
{
    unsigned int repeat;
    struct time_keeper *ptr;

    for (repeat = 0; repeat < repeats; repeat++) {
        ptr = kmalloc(sizeof(*ptr), GFP_KERNEL);
        ptr->time_before = ktime_get();
        ptr->time_after = ktime_get();
        pr_info("Hello, world!\n");
        list_add(&ptr->time_list, &lab5_list_head);
    }
}

static unsigned int repeats = 1;

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "Hello print:");

static int __init module4_init(void)
{
    if (repeats > 10) {
        pr_err("Cannot repeat more than 10 times\n");
        return -EINVAL;
    }

    if (repeats >= 5 && repeats <= 10)
        pr_warn("Repeatition from 5 to 10 times\n");

    if (repeats == 0)
        pr_warn("No repeatition\n");
}
```

```
Activities Terminal alex@alex-VirtualBox

static unsigned int repeats = 1;

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "Hello print:");

static int __init module4_init(void)
{
    if (repeats > 10) {
        pr_err("Cannot repeat more than 10 times\n");
        return -EINVAL;
    }

    if (repeats >= 5 && repeats <= 10)
        pr_warn("Repeation from 5 to 10 times\n");

    if (repeats == 0)
        pr_warn("No repeation\n");

    print_text(repeats);
    return 0;
}

static void __exit module4_exit(void)
{
    struct list_head *p;
    struct list_head *n;
    struct time_keeper *curr;

    pr_info("Bye-Bye!\n");

    list_for_each_safe(p, n, &lab5_list_head) {
        curr = list_entry(p, struct time_keeper, time_list);
        pr_info("Time needed for printing is: %lld(ns).\n",
                curr->time_after - curr->time_before);
        list_del(p);
        kfree(curr);
    }

    module_init(module4_init);
    module_exit(module4_exit);

    MODULE_AUTHOR("AlexHoncharenko");
    MODULE_DESCRIPTION("Training to work with list");
    MODULE_LICENSE("Dual BSD/GPL");
}
```

6. Тест роботи модулів:

```
/ # cd lab5/
/lab5 # insmod module
module4.c      module4.mod.c  module4.o
module4.ko     module4.mod.o  modules.order
/lab5 # insmod module4.ko repeats=3
[ 57.741280] module4: loading out-of-tree module taints kernel.
[ 57.784991] Hello, world!
[ 57.789214] Hello, world!
[ 57.790531] Hello, world!
/lab5 # rmmod module4.ko
[ 67.535431] Bye-Bye!
[ 67.536871] Time needed for printing is: 800(ns).
[ 67.539254] Time needed for printing is: 976(ns).
[ 67.541425] Time needed for printing is: 16704(ns).
```