Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Лабораторна робота № 3

з дисципліни «Архітектура комп’ютерів 2. Процесори»

Виконав:

Перевірив:

викладач

Гайдай А. Р.

Київ 2023

**Завдання:**

I. Продивитися всі три приклади (драйвер там однаковий, різні лише

системи збирання).

A. Обрати один з них для подальшої роботи, зібрати і виконати

insmod та rmmod на платі BBB (або емуляторі QEMU).

II. Модифікувати модуль, додавши до нього параметр\* типу uint, який

визначає, скільки разів має бути надрукований рядок “Hello, world!”

A. Значення параметра за умовчанням 1.

B. Якщо значення параметра 0 або знаходиться між 5 і 10,

надрукувати попередження і продовжити роботу.

C. Якщо значення параметра більше 10, то функція ініціалізації

повинна надрукувати повідомлення про помилку і повернути

значення -EINVAL (модуль не має завантажити взагалі).

D. Додати опис параметра (для його виводу командою modinfo).

III. 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).

IV. Створити статичну змінну голови списку.

V. Перед кожним друком привітання виділити пам&#39;ять\*\* для

екземпляра оголошеної структури, занести в неї поточний час ядра,

отриманий функцією ktime\_get().

VI. У функції hello\_exit() пройти по списку і надрукувати час

кожної події в наносекундах, вилучити елемент списку і звільнити

виділену пам&#39;ять. Приклад проходу по списку з вилученням

елемента є у appendix.

VII. Виконати insmod/rmmod модуля на платі BBB без параметра у

командному рядку, зі значеннями параметра 0, довільним між 1 і 10,

довільним більше 10.

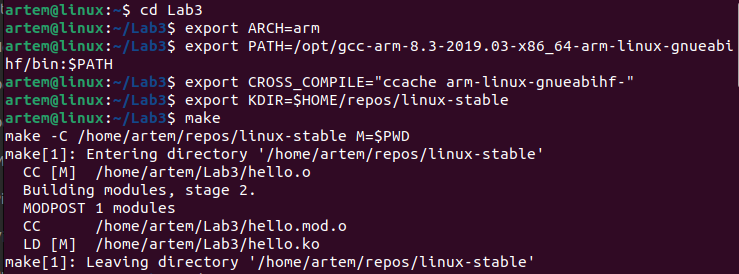
A. Після котрогось із insmod подивитися значення встановленого

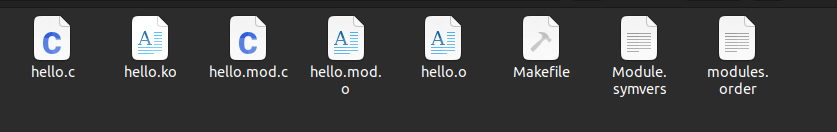
параметра (каталог /sys/module/hello/parameters)

**GitHub:**

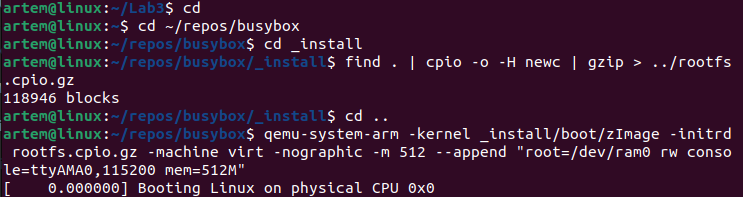
**Хід роботи**

Створення необхідних файлі для busybox

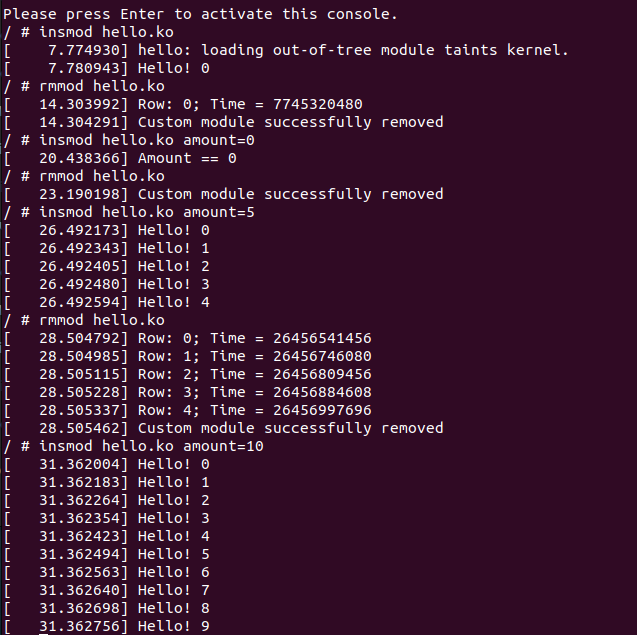
****

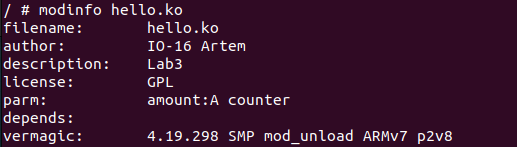
****

Оновлення busybox

****

Тест QEMU

****

****

**Hello.c**

#include <linux/init.h>

#include <linux/module.h>

#include <linux/printk.h>

#include <linux/moduleparam.h>

#include <linux/ktime.h>

#include <linux/slab.h>

MODULE\_AUTHOR("IO-16 Artem");

MODULE\_DESCRIPTION("Lab3");

MODULE\_LICENSE("GPL");

static uint amount = 1;

module\_param(amount, uint, 0444);

MODULE\_PARM\_DESC(amount, "A counter");

struct my\_custom\_struct {

struct list\_head list\_item;

ktime\_t time\_stamp;

};

static LIST\_HEAD(my\_custom\_list\_head);

static int \_\_init my\_custom\_init(void)

{

int i;

if (amount == 0) {

printk(KERN\_WARNING "Amount == 0\n");

} else if (amount > 10) {

printk(KERN\_WARNING "Amount > 10, because WARN!\n");

return -EINVAL;

}

for (i = 0; i < amount; i++) {

struct my\_custom\_struct \*item = kmalloc(sizeof(struct my\_custom\_struct), GFP\_KERNEL);

item->time\_stamp = ktime\_get();

list\_add\_tail(&item->list\_item, &my\_custom\_list\_head);

printk(KERN\_INFO "Hello! %d\n", i);

}

return 0;

}

static void \_\_exit my\_custom\_exit(void)

{

struct my\_custom\_struct \*entry, \*temp;

int idx = 0;

list\_for\_each\_entry\_safe(entry, temp, &my\_custom\_list\_head, list\_item) {

pr\_info("Row: %d; Time = %lld\n", idx++, ktime\_to\_ns(entry->time\_stamp));

list\_del(&entry->list\_item);

kfree(entry);

}

printk(KERN\_INFO "Custom module successfully removed\n");

}

module\_init(my\_custom\_init);

module\_exit(my\_custom\_exit);

**Висновок:**

Драйвер був завантажений на обладнання BeagleBone Black або віртуальне середовище QEMU, використовуючи команди insmod та rmmod для встановлення та видалення. Додатково була реалізована обробка специфічних значень параметрів драйвера. Ці зміни забезпечили стабільну та надійну роботу системи без перебоїв.