МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

з дисципліни «Дослідження та проектування комп’ютерних систем»

Виконав:

студент гр. ІО-91мн

Русінов В.В.

Перевірив:

Каплунов А.В.

Київ 2019

**Завдання**

Завдання Basic:

Продивитися всі три приклади (драйвер там однаковий, різні лише системи збирання).

Обрати один з них для подальшої роботи, зібрати і виконати insmod та rmmod на платі BBB (або емуляторі QEMU).

Модифікувати модуль, додавши до нього параметр типу uint, який визначає, скільки разів має бути надрукований рядок “Hello, world!”

Значення параметра за умовчанням 1.

Якщо значення параметра 0 або знаходиться між 5 і 10, надрукувати попередження і продовжити роботу.

Якщо значення параметра більше 10, то функція ініціалізації повинна надрукувати повідомлення про помилку і повернути значення -EINVAL (модуль не має завантажити взагалі).

Додати опис параметра. Подивитися його командою modinfo.

Виконати insmod/rmmod модуля на платі BBB без параметра у командному рядку, зі значеннями параметра 0, довільним між 1 і 10, довільним більше 10.

Після котрогось із insmod подивитися значення встановленого параметра (каталог /sys/module/hello/parameters)

Див. $KDIR/include/linux/moduleparam.h, опис макросів module\_param(), MODULE\_PARM\_DESC().

Завдання Advanced:

Розділити проект на два модулі, hello1 та hello2.

Модуль hello1 повинен експортувати функцію print\_hello(), яку використовуватиме модуль hello2 (параметр кількості викликів функції перенести у модуль hello2).

Заголовковий файл hello1.h, який використовуватимуть обидва модулі, винести у підкаталог inc, який додати у систему збирання так, щоб файли \*.c могли використовувати директиву #include "hello1.h" лише з іменем файлу, без шляху (див. ccflags-y).

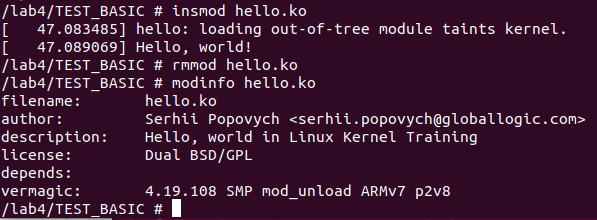
Замінити printk на відповідні ситуації pr\_err, pr\_warn, pr\_info (для друку привітання використати pr\_info)

Виконати insmod hello1.ko, потім insmod hello2.ko з такими значеннями параметра, щоб отримати всі можливі повідомлення і знайти їх (може знадобитися dmesg, grep).

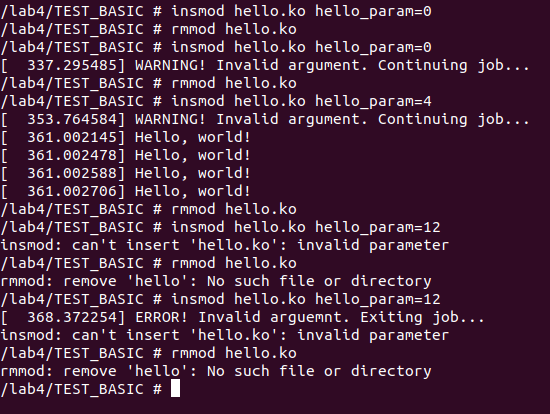
Спробувати завантажити hello2.ko, не завантажуючи hello1.ko, пояснити результат.

**Виконання завдання**

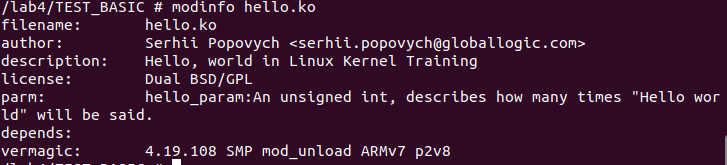
Виконання прикладу 1:



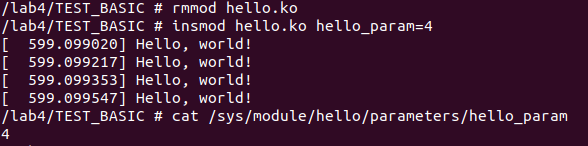
Виконання прикладу 2:



Виконання прикладу 3:

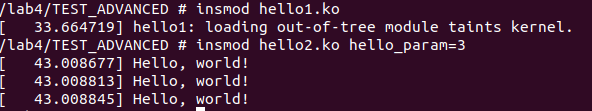


Виконання прикладу 4:



Advanced:

Виконання прикладу 4:



Виконання прикладу 5:



**Програмний код**

**hello.c**

#include <linux/init.h>

#include <linux/module.h>

#include <linux/printk.h>

MODULE\_AUTHOR("Serhii Popovych <serhii.popovych@globallogic.com>");

MODULE\_DESCRIPTION("Hello, world in Linux Kernel Training");

MODULE\_LICENSE("Dual BSD/GPL");

static uint hello\_param = 1;

module\_param(hello\_param, uint, S\_IRUGO); // visibility: world, immutable

MODULE\_PARM\_DESC(hello\_param, "An unsigned int, describes how many times \"Hello world\" will be said.");

static int \_\_init hello\_init(void)

{

int i = 0;

if(hello\_param == 0 || (hello\_param >= 5 && hello\_param <= 10))

printk(KERN\_EMERG "WARNING! Invalid argument. Continuing job...");

else if(hello\_param > 10) {

printk(KERN\_EMERG "ERROR! Invalid arguemnt. Exiting job...");

return -EINVAL;

}

else

for(i = 0; i < hello\_param; i++)

printk(KERN\_EMERG "Hello, world!\n");

return 0;

}

static void \_\_exit hello\_exit(void)

{

/\* Do nothing here right now \*/

}

module\_init(hello\_init);

module\_exit(hello\_exit);

**hello1.h**

int print\_hello(uint n);

static int \_\_init hello\_init(void);

static void \_\_exit hello\_exit(void);

**hello1.c**

#include <linux/init.h>

#include <linux/module.h>

#include <linux/printk.h>

#include "hello1.h"

MODULE\_AUTHOR("Serhii Popovych <serhii.popovych@globallogic.com>");

MODULE\_DESCRIPTION("Hello, world in Linux Kernel Training");

MODULE\_LICENSE("Dual BSD/GPL");

EXPORT\_SYMBOL(print\_hello);

static int \_\_init hello\_init(void)

{

return 0;

}

static void \_\_exit hello\_exit(void)

{

/\* Do nothing here right now \*/

}

int print\_hello(uint n){

int i = 0;

if(n == 0 || (n >= 5 && n <= 10))

pr\_warn("WARNING! Invalid argument. Continuing job...");

else if(n > 10) {

pr\_err("ERROR! Invalid arguemnt. Exiting job...");

return -EINVAL;

}

else

for(i = 0; i < n; i++)

pr\_info("Hello, world!\n");

return 0;

}

module\_init(hello\_init);

module\_exit(hello\_exit);

**hello2.c**

#include <linux/init.h>

#include <linux/module.h>

#include <linux/printk.h>

#include "hello1.h"

extern int print\_hello(uint n);

MODULE\_AUTHOR("Serhii Popovych <serhii.popovych@globallogic.com>");

MODULE\_DESCRIPTION("Hello, world in Linux Kernel Training");

MODULE\_LICENSE("Dual BSD/GPL");

static uint hello\_param = 1;

module\_param(hello\_param, uint, S\_IRUGO); // visibility: world, immutable

MODULE\_PARM\_DESC(hello\_param, "An unsigned int, describes how many times \"Hello world\" will be said.");

static int \_\_init hello\_init(void)

{

return print\_hello(hello\_param);

}

static void \_\_exit hello\_exit(void)

{

/\* Do nothing here right now \*/

}

module\_init(hello\_init);

module\_exit(hello\_exit);

**MakeFile**

ifneq ($(KERNELRELEASE),)

# kbuild part of makefile

obj-m := hello1.o

obj-m += hello2.o

ccflags-y := -I$(obj)/inc

else

# normal makefile

KDIR ?= /lib/modules/`uname -r`/build

default:

$(MAKE) -C $(KDIR) M=$$PWD

clean:

$(MAKE) -C $(KDIR) M=$$PWD clean

endif

**Висновки**

Під час виконання лабораторної роботи, було розроблено модулі для виконання на ядрі Linux. Було досліджено мовні засоби C для розробки модулів, поведінку модуля, застосовуючи різні вхідні параметри модуля.