|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Лабораторная работа №3*

*По предмету: «Операционные системы»*

**Тема: Загружаемые модули ядра**

Преподаватель: Рязанова Н.Ю.

Студент: Гасанзаде М.А.,

Группа: ИУ7-66Б

Москва, 2020 г.

ОГЛАВЛЕНИЕ

[ЗАДАНИЕ 1 3](#_Toc41999777)

[Листинг 1 - Код модуля md 3](#_Toc41999778)

[Результат работы программы 3](#_Toc41999779)

[ЗАДАНИЕ 2 7](#_Toc41999780)

[Модуль md1 7](#_Toc41999781)

[Листинг 2 – Код модуля md1 7](#_Toc41999782)

[Результат работы программы 9](#_Toc41999783)

[Модуль md2 9](#_Toc41999784)

[Листинг 3 – Код модуля md2 9](#_Toc41999785)

[Результат работы программы 10](#_Toc41999786)

[Модуль md3 11](#_Toc41999787)

[Листинг 4 – Код модуля md3 11](#_Toc41999788)

[Результат работы программы 12](#_Toc41999789)

# ЗАДАНИЕ 1

Реализовать загружаемый модуль ядра, который при загрузке записывает в

системный журнал информацию о запущенных процессах. Модуль должен

собираться при помощи make-файла.

## Листинг 1 - Код модуля md

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

#include <linux/sched.h>

#include <linux/init\_task.h>

MODULE\_LICENSE**(**"GPL"**);**

MODULE\_AUTHOR**(**"Hasanzade M.A."**);**

static int \_\_init md\_init**(**void**)** **{**

printk**(**KERN\_INFO "Module is loaded.\n"**);**

struct task\_struct **\***task **=** **&**init\_task**;**

**do** **{**

printk**(**KERN\_INFO " |comm: %15s|pid: %8d|ppid: %15d|\n"**,** task**->**comm**,** task**->**pid**,** task**->**parent**->**comm**,** task**->**parent**->**pid**);**

**}** **while** **((**task **=** next\_task**(**task**))** **!=** **&**init\_task**);**

printk**(**KERN\_INFO " [current]|comm: %15s|pid: %8d|ppid: %15d|\n"**,** current**->**comm**,** current**->**pid**,** current**->**parent**->**comm**,** current**->**parent**->**pid**);**

**return** 0**;**

**}**

static void \_\_exit md\_exit**(**void**)** **{**

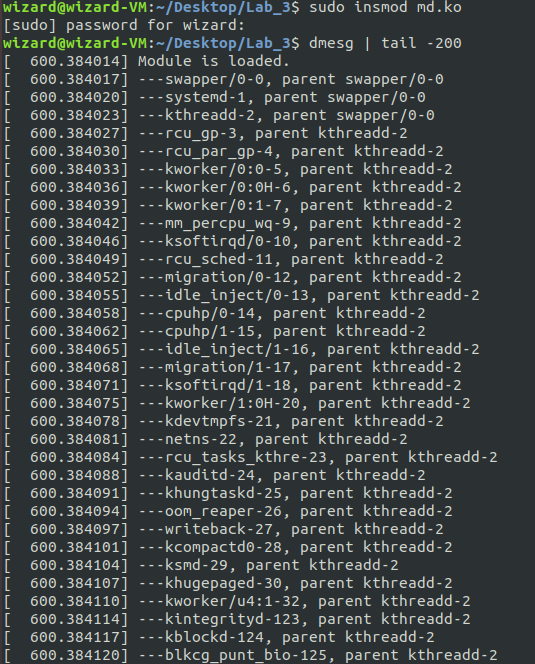
printk**(**KERN\_INFO " Module is unloaded.\n"**);**

**}**

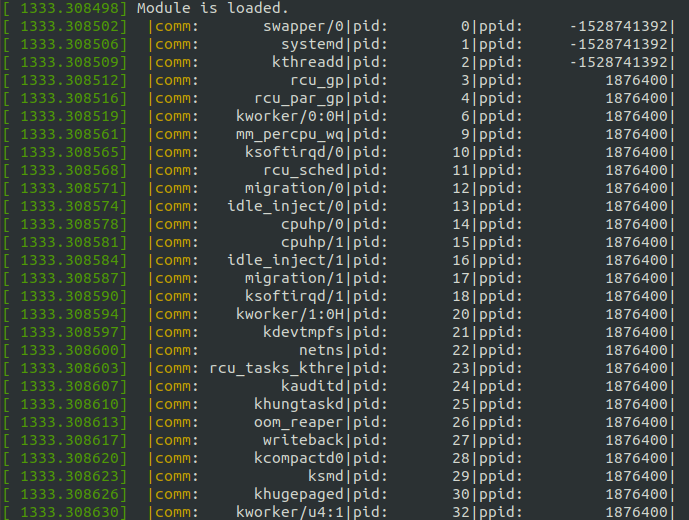
module\_init**(**md\_init**);**

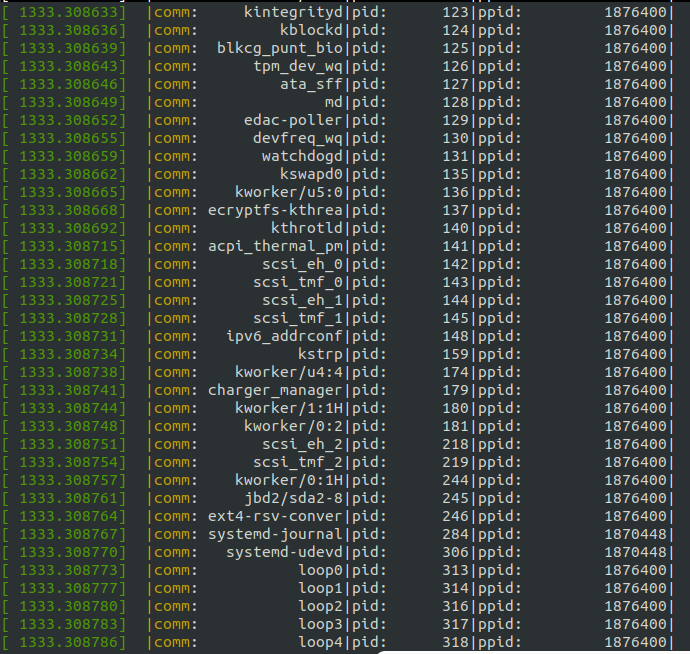
module\_exit**(**md\_exit**);**

## Результат работы программы



……………………………………………………………………………………………………………





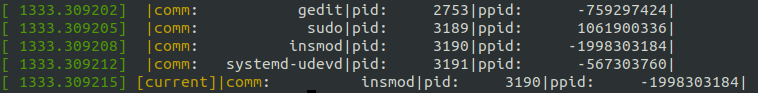


Рис. 1 – сборка и информация модуля ядра

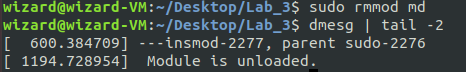


Рис. 2 – выгрузка модуля ядра

# ЗАДАНИЕ 2

Реализовать три загружаемых модуля ядра:

* Вызываемый модуль md1
* Вызывающий модуль md2
* «Отладочный» модуль md3

Загружаемые модули должны собираться при помощи Make-файла (сборка командой make). Вызов каждой функции модуля должен сопровождаться записью в системный журнал информации, какая функция, какого модуля была вызвана.

## Модуль md1

Модуль md1 демонстрирует возможность создания экспортируемых данных и функций. Данный модуль ядра должен содержать:

* Экспортируемые строковые (char \*) и численные (int) данные.
* Экспортируемые функции, возвращающие строковые и числовые

значения.

Например:

* Функция, возвращающая в зависимости от переданного

целочисленного параметра различные строки (на усмотрение студента);

* Функция, производящая подсчет факториала переданного

целочисленного параметра;

* Функция, возвращающая 0.

### Листинг 2 – Код модуля md1

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

MODULE\_LICENSE**(** "GPL" **);**

MODULE\_AUTHOR**(** "Hasanzade M.A" **);**

char**\*** md1\_str\_data **=** "Hello world!"**;**

int md1\_int\_data **=** 42**;**

extern char**\*** md1\_get\_str**(**int n**)** **{**

printk**(** "+ md1: md1\_get\_str() called!\n" **);**

**switch** **(**n**){**

**case** 1**:**

**return** "Hello world!\n"**;**

**break;**

**case** 2**:**

**return** "Привет Мир!\n"**;**

**break;**

**default:**

**return** "Передайте 1 для получения английского сообщения или 2 для получения русского.\n"**;**

**break;**

**}**

**}**

extern int md1\_factorial**(**int n**)** **{**

int i**,** ans**;**

ans **=** 1**;**

printk**(** "+ md1: md1\_factorial() called!\n" **);**

**for** **(**i **=** 2**;** i **<=** n**;** i**++)** ans **\*=** i**;**

**return** ans**;**

**}**

extern int md1\_return\_zero**(**void**){**

printk**(** "+ md1: md1\_return\_zero() called!\n" **);**

**return** 0**;**

**}**

EXPORT\_SYMBOL**(**md1\_str\_data**);**

EXPORT\_SYMBOL**(**md1\_int\_data**);**

EXPORT\_SYMBOL**(**md1\_get\_str**);**

EXPORT\_SYMBOL(md1\_factorial);

EXPORT\_SYMBOL(md1\_return\_zero);

static int \_\_init md1\_init( void ) {

printk( "+ md1: Module md1 is loaded!\n" );

return 0;

}

static void \_\_exit md1\_exit( void ) {

printk( "+ md1: Module md1 is unloaded!\n" );

}

module\_init( md1\_init );

module\_exit( md1\_exit );

### **Результат работы программы**



Рис. 3 – загрузка и информация от модуля ядра



Рис. 4 – выгрузка модуля ядра

## Модуль md2

Модуль md2 демонстрирует использование данных и функций

экспортируемых первым модулем (md1). Данный модуль должен при загрузке:

* Вызывать все экспортированные модулем md1 процедуры и вывести в системный журнал возвращаемые ими значения с указанием имени вызванной процедуры.
* Вывести в системный журнал все экспортированные модулем md1 данные.

### Листинг 3 – Код модуля md2

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

MODULE\_LICENSE**(**"GPL"**);**

MODULE\_AUTHOR**(**"Hasanzade M.A"**);**

extern char**\*** md1\_str\_data**;**

extern int md1\_int\_data**;**

extern char**\*** md1\_get\_str**(**int num**);**

extern int md1\_factorial**(**int num**);**

extern int md1\_return\_zero**(**void**);**

static int \_\_init md2\_init**(**void**)** **{**

printk**(**KERN\_INFO "+md2: Module 2 is loaded.\n"**);**

printk**(**KERN\_INFO "+ md2: Export char\* from MD1 - %s.\n"**,** md1\_str\_data**);**

printk**(**KERN\_INFO "+ md2: Export int from MD1 - %d.\n"**,** md1\_int\_data**);**

printk**(**KERN\_INFO "+ md2: Function 'md1\_get\_str(1)' from MD1 return %s.\n"**,** md1\_get\_str**(**1**));**

printk**(**KERN\_INFO "+ md2: Function 'md1\_factorial(5)' from MD1 return %d.\n"**,** md1\_factorial**(**5**));**

printk**(**KERN\_INFO "+ md2: Function 'md1\_return\_zero' from MD1 return %d.\n"**,** md1\_return\_zero**());**

**return** 0**;**

**}**

static void \_\_exit md2\_exit**(**void**)** **{**

printk**(**KERN\_INFO "+ md2: Module 2 is unloaded.\n"**);**

**}**

module\_init**(**md2\_init**);**

module\_exit**(**md2\_exit**);**

### **Результат работы программы**

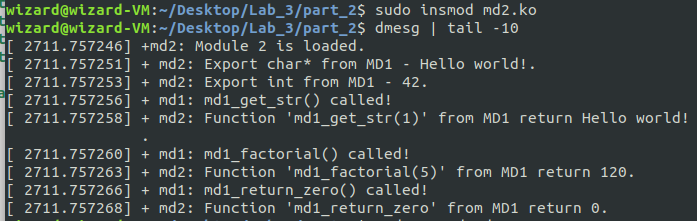


Рис. 5 – загрузка и информация от модуля ядра

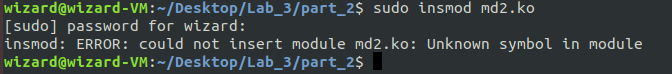


Рис. 6 – ситуация, когда данные, к которым обращается модуль не доступны.



Рис. 7 – выгрузка модуля ядра

## Модуль md3

Модуль md3 демонстрирует сценарий некорректного завершения установки модуля, и возможность использования загружаемого модуля в качестве функции, выполняемой в пространстве ядре.

Процедура инициализации этого загружаемого модуля должна возвращать ненулевое значение и выводить в системный журнал данные и возвращаемые значения экспортированных модулем md1 процедур (аналогично md2). Данный модуль включен в работу для проработки вопросов, связанных с отладкой модулей ядра.

### Листинг 4 – Код модуля md3

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

MODULE\_LICENSE**(**"GPL"**);**

MODULE\_AUTHOR**(**"Hasanzade M.A"**);**

extern char**\*** md1\_str\_data**;**

extern int md1\_int\_data**;**

extern char**\*** md1\_get\_str**(**int num**);**

extern int md1\_factorial**(**int num**);**

extern int md1\_return\_zero**(**void**);**

static int \_\_init md3\_init**(**void**)** **{**

printk**(**KERN\_INFO "+md3: Module 3 is loaded.\n"**);**

printk**(**KERN\_INFO "+ md3: Export char\* from MD1 - %s.\n"**,** md1\_str\_data**);**

printk**(**KERN\_INFO "+ md3: Export int from MD1 - %d.\n"**,** md1\_int\_data**);**

printk**(**KERN\_INFO "+ md3: Function 'md1\_get\_str(1)' from MD1 return %s.\n"**,** md1\_get\_str**(**1**));**

printk**(**KERN\_INFO "+ md3: Function 'md1\_factorial(5)' from MD1 return %d.\n"**,** md1\_factorial**(**5**));**

printk**(**KERN\_INFO "+ md3: Function 'md1\_return\_zero' from MD1 return %d.\n"**,** md1\_return\_zero**());**

**return** 0**;**

**}**

static void \_\_exit md3\_exit**(**void**)** **{**

printk**(**KERN\_INFO "+ md3: Module 3 is unloaded.\n"**);**

**}**

module\_init**(**md3\_init**);**

module\_exit**(**md3\_exit**);**

### **Результат работы программы**



Рис. 8 – ситуация, когда данные, к которым обращается модуль не доступны



Рис. 9 – Ошибка при значении **return** -1**;**(Листинг 4). Нужно поменять на значение 0.

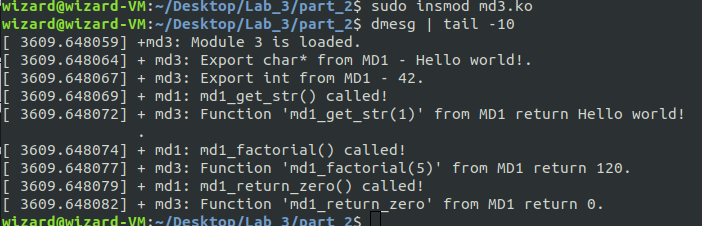


Рис. 10 – Успешная загрузка модуля и информация, записанная им.



Рис. 11 – выгрузка модуля ядра