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 №9
По курсу «Операционные системы»
Тема: Обработчики прерываний

Студент: Кондрашова О.П.
Группа: ИУ7-65Б
Преподаватель: Рязанова Н.Ю.

Москва, 2020г.

Часть 1. Тасклет

Листинг 1. Код программы

```
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/interrupt.h>
#include <linux/time.h>

MODULE_LICENSE("GPL");
MODULE_AUTHOR("Kondrashova");
MODULE_DESCRIPTION("lab9");

#define SHARED_IRQ 1

static int my_dev_id;
char tasklet_data[] = "tasklet was called";

void tasklet_function(unsigned long data);

DECLARE_TASKLET(my_tasklet, tasklet_function, (unsigned long) &tasklet_data);

void tasklet_function(unsigned long data)
{
    printk(KERN_INFO "Tasklet: state - %ld, count - %d, data - %s\n",
        my_tasklet.state,
        my_tasklet.count,
        my_tasklet.data);
}

// Обработчик прерывания
static irqreturn_t my_interrupt(int irq, void *dev_id)
{
    // Проверка, что произошло нужное прерывание
    if (irq = SHARED_IRQ)
    {
        printk(KERN_INFO "Tasklet scheduled\n");
        // Постановка тасклета в очередь на исполнение
        tasklet_schedule(&my_tasklet);
    return IRQ_HANDLED;     // Прерывание обработано
    }

    else
        return IRQ_NONE;    // Прерывание не обработано
}

// Инициализация модуля
static int __init my_tasklet_init(void)
{
    // Разделение линии IRQ с другими устройствами
    int ret = request_irq(SHARED_IRQ, my_interrupt, IRQF_SHARED, "my_interrupt", &my_dev_id);

    if (ret)
    {
    printk(KERN_INFO "Error on request_irq\n");
    return -1;
    }
    printk(KERN_INFO "Module loaded\n");
    return 0;
}

// Выход загружаемого модуля
static void __exit my_tasklet_exit(void)
{
    // Удаление тасклета
    tasklet_kill(&my_tasklet);
```

```
63.     // Освобождение линии прерывания
64.     free_irq(SHARED_IRQ, &my_dev_id);
65.     printk(KERN_INFO "Module unloaded\n");
66. }
67. 
68. module_init(my_tasklet_init);
69. module_exit(my_tasklet_exit);
```

Загрузим модуль ядра и проверим список загруженных модулей:

```
olga@olga-VirtualBox:~/Documents/lab9/part1$ sudo insmod tasklet.ko
olga@olga-VirtualBox:~/Documents/lab9/part1$ lsmod | grep tasklet
tasklet                16384  0
```

Вывод буфера сообщений ядра в стандартный поток вывода:

```
olga@olga-VirtualBox:~/Documents/lab9/part1$ sudo dmesg | tail -4
[ 2309.503343] Tasklet scheduled
[ 2309.503360] Tasklet: state - 2, count - 0, data - tasklet was called
[ 2309.780713] Tasklet scheduled
[ 2309.780733] Tasklet: state - 2, count - 0, data - tasklet was called
```

Файл /proc/interrupts предоставляет таблицу о количестве прерываний на каждом из процессоров:

```
olga@olga-VirtualBox:~/Documents/lab9/part1$ cat /proc/interrupts
           CPU0
  0:         30   IO-APIC   2-edge      timer
  1:        458   IO-APIC   1-edge      i8042, my_interrupt
  8:          0   IO-APIC   8-edge      rtc0
  9:          0   IO-APIC   9-fasteoi   acpi
```

Выгружаем модуль ядра и выводим буфер сообщений ядра:

```
olga@olga-VirtualBox:~/Documents/lab9/part1$ sudo rmmod tasklet
olga@olga-VirtualBox:~/Documents/lab9/part1$ sudo dmesg | tail -8
[ 2448.092848] Tasklet: state - 2, count - 0, data - tasklet was called
[ 2448.249630] Tasklet scheduled
[ 2448.249650] Tasklet: state - 2, count - 0, data - tasklet was called
[ 2448.307037] Tasklet scheduled
[ 2448.307058] Tasklet: state - 2, count - 0, data - tasklet was called
[ 2448.626697] Tasklet scheduled
[ 2448.626717] Tasklet: state - 2, count - 0, data - tasklet was called
[ 2448.635277] Module unloaded
```

Часть 2. Очередь работ

Листинг 2. Код программы

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/interrupt.h>
#include <linux/workqueue.h>

MODULE_LICENSE("GPL");
MODULE_AUTHOR("Kondrashova");
MODULE_DESCRIPTION("lab9");

#define SHARED_IRQ 1

static int my_dev_id;
static int irq_call_counter = 0;

static struct workqueue_struct *wq;

void my_workqueue_function(struct work_struct *work);

DECLARE_WORK(workqueue_name, my_workqueue_function);

void my_workqueue_function(struct work_struct *work)
{
    printk(KERN_INFO "Workqueue: working, counter: %d\n", ++irq_call_counter);
}

static irqreturn_t my_interrupt(int irq, void *dev)
{
    // Проверка, что произошло нужное прерывание
    if (irq = SHARED_IRQ)
    {
    queue_work(wq, &workqueue_name);
        printk(KERN_INFO "Workqueue: starting\n");
    return IRQ_HANDLED;     // Прерывание обработано
    }
    else
        return IRQ_NONE;    // Прерывание не обработано
}

// Инициализация модуля
static int __init my_workqueue_init(void)
{
    // Разделение линии IRQ с другими устройствами
    int ret = request_irq(SHARED_IRQ, my_interrupt, IRQF_SHARED, "my_interrupt", &my_dev_id);

    if (ret)
    {
        printk(KERN_ERR "Module error: couldn't register handler\n");
        return ret;
    }

    // Создание очереди работ
    wq = create_workqueue("my_workqueue");
    if (!wq)
    {
        free_irq(SHARED_IRQ, &my_dev_id);
        printk(KERN_INFO "Module error: couldn't create workqueue\n");
        return -ENOMEM;
    }
```

```
58.
59.     printk(KERN_INFO "Module loaded\n");
60.     return 0;
61. }
62.
63. // Выход загружаемого модуля
64. static void __exit my_workqueue_exit(void)
65. {
66.     // Удаление очереди работ
67.     flush_workqueue(wq);
68.     destroy_workqueue(wq);
69.     // Освобождение линии прерывания
70.     free_irq(SHARED_IRQ, &my_dev_id);
71.     printk(KERN_INFO "Module unloaded\n");
72. }
73.
74. module_init(my_workqueue_init)
75. module_exit(my_workqueue_exit)
```

Загрузим модуль ядра и проверим список загруженных модулей:

```
olga@olga-VirtualBox:~/Documents/lab9/part2$ sudo insmod work.ko
olga@olga-VirtualBox:~/Documents/lab9/part2$ lsmod | grep work
work                   16384  0
```

Вывод буфера сообщений ядра в стандартный поток вывода:

```
olga@olga-VirtualBox:~/Documents/lab9/part2$ dmesg | tail -8
[ 3912.285036] Workqueue: starting
[ 3912.285062] Workqueue: working, counter: 181
[ 3912.528748] Workqueue: starting
[ 3912.529027] Workqueue: working, counter: 182
[ 3912.588316] Workqueue: starting
[ 3912.588340] Workqueue: working, counter: 183
[ 3912.779765] Workqueue: starting
[ 3912.780052] Workqueue: working, counter: 184
```

Файл /proc/interrupts предоставляет таблицу о количестве прерываний на каждом из процессоров:

```
olga@olga-VirtualBox:~/Documents/lab9/part2$ cat /proc/interrupts
           CPU0
  0:         30   IO-APIC   2-edge      timer
  1:        836   IO-APIC   1-edge      i8042, my_interrupt
  8:          0   IO-APIC   8-edge      rtc0
  9:          0   IO-APIC   9-fasteoi   acpi
```

Выгружаем модуль ядра и выводим буфер сообщений ядра:

```
olga@olga-VirtualBox:~/Documents/lab9/part2$ sudo rmmod work
olga@olga-VirtualBox:~/Documents/lab9/part2$ dmesg | tail -8
[ 3943.940559] Workqueue: working, counter: 223
[ 3944.152821] Workqueue: starting
[ 3944.153176] Workqueue: working, counter: 224
[ 3944.218034] Workqueue: starting
[ 3944.218067] Workqueue: working, counter: 225
[ 3945.092895] Workqueue: starting
[ 3945.092931] Workqueue: working, counter: 226
[ 3945.103758] Module unloaded
```