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 предмету: «Операционные системы»

Тема: Обработчики прерываний

Студент: Лаврова А. А.,
Группа: ИУ7-65Б
Преподаватель: Рязанова Н. Ю.

Москва, 2020 г.

Тасклет

Листинг 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("Lavrova");
MODULE_DESCRIPTION("lab_9");

#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);
    // Прерывание обработано
      return IRQ_HANDLED;
    }
    else
        // Прерывание не обработано
        return IRQ_NONE;
}

// 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);
    // Освобождение линии прерывания
    free_irq(SHARED_IRQ, &my_dev_id);
    printk(KERN_INFO "Module unloaded\n");
}

module_init(my_tasklet_init);
module_exit(my_tasklet_exit);
```

Результаты работы программы:

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ sudo insmod tasklet.ko
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ lsmod | grep tasklet
tasklet                16384  0
```

Рис.1: Загрузка модуля ядра и проверка списка загруженных модулей

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ sudo dmesg | tail -4
[ 8922.051604] Tasklet scheduled
[ 8922.051607] Tasklet: state - 2, count - 0, data - tasklet was called
[ 8922.623147] Tasklet scheduled
[ 8922.623152] Tasklet: state - 2, count - 0, data - tasklet was called
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$
```

Рис.2: Вывод буфера сообщений ядра в стандартный поток вывода

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ cat /proc/interrupts
           CPU0       CPU1
  0:          2          0   IO-APIC   2-edge      timer
  1:          0       4351   IO-APIC   1-edge      i8042, my_interrupt
  8:          1          0   IO-APIC   8-edge      rtc0
  9:          0      13364   IO-APIC   9-fasteoi   acpi
```

Рис.3: Файл /proc/interrupts предоставляет таблицу о количестве прерываний на каждом из процессоров

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ sudo rmmod tasklet
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ sudo dmesg | tail -8
[ 8992.169171] Tasklet: state - 2, count - 0, data - tasklet was called
[ 8992.359201] Tasklet scheduled
[ 8992.359221] Tasklet: state - 2, count - 0, data - tasklet was called
[ 8992.428486] Tasklet scheduled
[ 8992.428506] Tasklet: state - 2, count - 0, data - tasklet was called
[ 8992.936258] Tasklet scheduled
[ 8992.936278] Tasklet: state - 2, count - 0, data - tasklet was called
[ 8992.958915] Module unloaded
```

Рис.4: Загрузка модуля ядра и вывод буфера сообщений ядра

Очередь работ

Листинг программы:

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

MODULE_LICENSE("GPL");
MODULE_AUTHOR("Lavrova");
MODULE_DESCRIPTION("lab_9");

#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");
    // Прерывание обработано
      return IRQ_HANDLED;
    }
    else
        // Прерывание не обработано
        return IRQ_NONE;
}

// 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;
    }

    printk(KERN_INFO "Module loaded\n");
    return 0;
}

// Выход загружаемого модуля
static void __exit my_workqueue_exit(void)
{
    // Удаление очереди работ
    flush_workqueue(wq);
    destroy_workqueue(wq);
    // Освобождение линии прерывания
    free_irq(SHARED_IRQ, &my_dev_id);
    printk(KERN_INFO "Module unloaded\n");
}

module_init(my_workqueue_init)
module_exit(my_workqueue_exit)
```

Результаты работы программы:

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ sudo insmod work.ko
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ lsmod | grep work
work                   16384  0
```

Рис.5: Загрузка модуля ядра и проверка списка загруженных модулей

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ dmesg | tail -8
[ 9191.650829] Workqueue: starting
[ 9191.650858] Workqueue: working, counter: 131
[ 9192.020892] Workqueue: starting
[ 9192.020917] Workqueue: working, counter: 132
[ 9192.106949] Workqueue: starting
[ 9192.106959] Workqueue: working, counter: 133
[ 9192.303929] Workqueue: starting
[ 9192.303950] Workqueue: working, counter: 134
```

Рис.6: Вывод буфера сообщений ядра в стандартный поток вывода

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ cat /proc/interrupts
           CPU0       CPU1
  0:          2          0   IO-APIC   2-edge      timer
  1:          0       4859   IO-APIC   1-edge      i8042, my_interrupt
  8:          1          0   IO-APIC   8-edge      rtc0
  9:          0      13791   IO-APIC   9-fasteoi   acpi
```

Рис.7: Файл /proc/interrupts предоставляет таблицу о количестве прерываний на каждом из процессоров

```
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ sudo rmmod work
parallels@parallels-Parallels-Virtual-Platform:~/lab_9$ dmesg | tail -8
[ 9309.963460] Workqueue: working, counter: 295
[ 9310.076464] Workqueue: starting
[ 9310.076492] Workqueue: working, counter: 296
[ 9310.138356] Workqueue: starting
[ 9310.138377] Workqueue: working, counter: 297
[ 9310.599495] Workqueue: starting
[ 9310.599523] Workqueue: working, counter: 298
[ 9310.618479] Module unloaded
```

Рис. 8: Загрузка модуля ядра и вывод буфера сообщений ядра