

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический
университет имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 9

Дисциплина	Операционные системы.
Тема	Обработчики прерываний.
Студент	Куприй А. А.
Группа	ИУ7-63Б
Преподаватель	Рязанова Н.Ю.

Москва, 2020 г.

ВВЕДЕНИЕ

Задание:

Задание 1:

- Написать загружаемый модуль ядра, в котором зарегистрировать обработчик аппаратного прерывания с флагом `IRQF_SHARED`.
- Инициализировать тасклет.
- В обработчике прерывания запланировать тасклет на выполнение.
- Вывести информацию о таскете используя, или `printk()`, или `seq_file` interface - `<linux/seq_file.h`.

Задание 2:

- Написать загружаемый модуль ядра, в котором зарегистрировать обработчик аппаратного прерывания с флагом `IRQF_SHARED`.
- Инициализировать очередь работ.
- В обработчике прерывания запланировать очередь работ на выполнение.
- Вывести информацию об очереди работ используя, или `printk()`, или `seq_file` interface - `<linux/seq_file.h>`.

1 Практическая часть

Для первого и второго задания был написан обработчик прерывания для `irq = 1`.

1.1 Задание 1

На листинге ниже представлен текст программы первого задания.

Листинг 1.1 — Текст программы

```
1 #include <linux/module.h>
2 #include <linux/kernel.h>
3 #include <linux/init.h>
4 #include <linux/interrupt.h>
5 #include <linux/time.h>
6
7 MODULE_AUTHOR("Alexander Kupry");
8 MODULE_LICENSE("GPL");
9 MODULE_DESCRIPTION("lab_09_task_1");
10
11 #define HANDLEDIRQ 1
12
13 static int irq = HANDLEDIRQ;
14 static int irq_call_count = 0;
15 static int dev_id;
16
17 void tasklet_function(unsigned long data);
18
19 char tasklet_data[] = "tasklet_function was called";
20 DECLARE_TASKLET(tasklet, tasklet_function, (unsigned long)&tasklet_data);
21
22 void tasklet_function(unsigned long data)
23 {
24     struct timeval t;
25     struct tm brocken;
26     do_gettimeofday(&t);
27     time_to_tm(t.tv_sec, 0, &brocken);
28
29     printk(KERN_INFO "TASKLET MODULE\nTasklet: { state: %ld, count: %d,
30                data: %s }, current_time: %d:%d:%d:%ld\n",
31            tasklet.state, atomic_read(&tasklet.count), (char *)tasklet.data,
32            brocken.tm_hour + 3, brocken.tm_min, brocken.tm_sec, t.tv_usec);
33 }
```

```

34 static irqreturn_t interrupt_handler(int irq, void *dev_id)
35 {
36     if (irq == HANDLEDIRQ)
37     {
38         irq_call_count++;
39         printk(KERN_INFO "TASKLET MODULE\nirq call count = %d\n",
40                 irq_call_count);
41         tasklet_schedule(&tasklet);
42         return IRQ_HANDLED;
43     }
44     else
45     {
46         return IRQ_NONE;
47     }
48
49 static int __init tasklet_module_init(void)
50 {
51     int ret = request_irq(irq, interrupt_handler, IRQF_SHARED,
52                           "tasklet_interrupt_handler", &dev_id);
53
54     if (ret)
55     {
56         printk(KERN_ERR "TASKLET MODULE\nerror while handle irq\n");
57         return -1;
58     }
59
60     printk(KERN_INFO "TASKLET MODULE\nsuccess load\n");
61     return 0;
62
63 static void __exit tasklet_module_exit(void)
64 {
65     tasklet_kill(&tasklet);
66     free_irq(irq, &dev_id);
67     printk(KERN_INFO "TASKLET MODULE\nunload module\n");
68 }
69
70 module_init(tasklet_module_init);
71 module_exit(tasklet_module_exit);

```

Компиляция загружаемого модуля ядра при помощи Makefile:

```

san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ make
make -C /lib/modules/4.15.0-101-generic/build M=/home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_1 modules
make[1]: Entering directory '/usr/src/linux-headers-4.15.0-101-generic'
CC [M] /home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_1/tasklet.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_1/tasklet.mod.o
LD [M] /home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_1/tasklet.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-101-generic'
san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ ls
compile_commands.json  Makefile  modules.order  Module.symvers  tasklet.c  tasklet.ko  tasklet.mod.c  tasklet.mod.o  tasklet.o

```

Рисунок 1.1 — Скриншот результата работы make.

Загрузка модуля ядра с помощью команды insmod:

```

san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ sudo insmod tasklet.ko
san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ sudo lsmod | grep -B 1 tasklet
Module              Size  Used by
tasklet              16384  0

```

Рисунок 1.2 — Скриншот демонстрации успешной загрузки модуля.

Демонстрация успешного вызова обработчика прерываний tasklet_function:

```

san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ sudo dmesg | tail
[16295.519571] TASKLET MODULE
Tasklet: { state: 2, count: 0, data: tasklet_function was called }, current_time: 3:13:14:674391
[16295.559874] TASKLET MODULE
irq call count = 67
[16295.559895] TASKLET MODULE
Tasklet: { state: 2, count: 0, data: tasklet_function was called }, current_time: 3:13:14:714719
[16296.103607] TASKLET MODULE
irq call count = 68
[16296.103627] TASKLET MODULE
Tasklet: { state: 2, count: 0, data: tasklet_function was called }, current_time: 3:13:15:258518

```

Рисунок 1.3 — Скриншот команды sudo dmesg.

Демонстрация содержимого /proc/interrupts:

```

san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ sudo cat /proc/interrupts | head -4

```

	CPU0	CPU1	CPU2	CPU3	CPU4	CPU5	CPU6	CPU7			
0:	9	0	0	0	0	0	0	0	IR-I0-APIC	2-edge	timer
1:	0	0	25176	0	0	0	0	0	IR-I0-APIC	1-edge	i8042, tasklet_interrupt_handler
8:	0	0	0	1	0	0	0	0	IR-I0-APIC	8-edge	rtc0

Рисунок 1.4 — Скриншот команды cat /proc/interrupts.

Демонстрация успешной выгрузки модуля:

```

san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ sudo rmmod tasklet
san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ sudo dmesg | tail -2
[17901.758242] TASKLET MODULE
unload module
san_sanchez@LEX ~/workspace/OC/sem_2/lab_09/source/task_1$ sudo cat /proc/interrupts | head -4

```

	CPU0	CPU1	CPU2	CPU3	CPU4	CPU5	CPU6	CPU7			
0:	9	0	0	0	0	0	0	0	IR-I0-APIC	2-edge	timer
1:	0	0	27023	0	0	0	0	0	IR-I0-APIC	1-edge	i8042
8:	0	0	0	1	0	0	0	0	IR-I0-APIC	8-edge	rtc0

Рисунок 1.5 — Скриншот результата работы команды insmod.

1.2 Задание 2

На листинге ниже представлен текст программы второго задания.

Листинг 1.2 — Текст программы

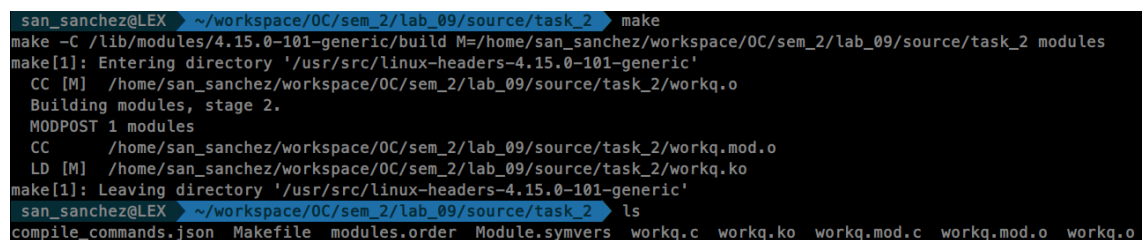
```
1  #include <linux/module.h>
2  #include <linux/kernel.h>
3  #include <linux/init.h>
4  #include <linux/interrupt.h>
5  #include <linux/time.h>
6
7  MODULE_AUTHOR("Alexander Kupry");
8  MODULE_LICENSE("GPL");
9  MODULE_DESCRIPTION("lab_09_task_2");
10
11 #define HANDLEDIRQ 1
12
13 static int irq = HANDLEDIRQ;
14 static int irq_call_count = 0;
15 static int dev_id;
16
17 void tasklet_function(unsigned long data);
18
19 char tasklet_data[] = "tasklet_function was called";
20 DECLARE_TASKLET(tasklet, tasklet_function, (unsigned long)&tasklet_data);
21
22 void tasklet_function(unsigned long data)
23 {
24     struct timeval t;
25     struct tm brocken;
26     do_gettimeofday(&t);
27     time_to_tm(t.tv_sec, 0, &brocken);
28
29     printk(KERN_INFO "TASKLET MODULE\nTasklet: { state: %ld, count: %d,
30         data: %s }, current_time: %d:%d:%d:%ld\n",
31         tasklet.state, atomic_read(&tasklet.count), (char *)tasklet.data,
32         brocken.tm_hour + 3, brocken.tm_min, brocken.tm_sec, t.tv_usec);
33 }
34 static irqreturn_t interrupt_handler(int irq, void *dev_id)
35 {
36     if (irq == HANDLEDIRQ)
37     {
38         irq_call_count++;
39         printk(KERN_INFO "TASKLET MODULE\nirq call count = %d\n",
40             irq_call_count);
```

```

40         tasklet_schedule(&tasklet);
41         return IRQ_HANDLED;
42     }
43     else
44     {
45         return IRQ_NONE;
46     }
47 }
48
49 static int __init tasklet_module_init(void)
50 {
51     int ret = request_irq(irq, interrupt_handler, IRQF_SHARED,
52         "tasklet_interrupt_handler", &dev_id);
53
54     if (ret)
55     {
56         printk(KERN_ERR "TASKLET MODULE\nerror while handle irq\n");
57         return -1;
58     }
59
60     printk(KERN_INFO "TASKLET MODULE\nsuccess load\n");
61     return 0;
62 }
63
64 static void __exit tasklet_module_exit(void)
65 {
66     tasklet_kill(&tasklet);
67     free_irq(irq, &dev_id);
68     printk(KERN_INFO "TASKLET MODULE\nunload module\n");
69 }
70
71 module_init(tasklet_module_init);
72 module_exit(tasklet_module_exit);

```

Компиляция загружаемого модуля ядра при помощи Makefile:



```

san_sanchez@LEX: ~/workspace/OC/sem_2/lab_09/source/task_2
make -C /lib/modules/4.15.0-101-generic/build M=/home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_2 modules
make[1]: Entering directory '/usr/src/linux-headers-4.15.0-101-generic'
CC [M] /home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_2/workq.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_2/workq.mod.o
LD [M] /home/san_sanchez/workspace/OC/sem_2/lab_09/source/task_2/workq.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-101-generic'
san_sanchez@LEX: ~/workspace/OC/sem_2/lab_09/source/task_2
ls
compile_commands.json Makefile modules.order Module.symvers workq.c workq.ko workq.mod.c workq.mod.o workq.o

```

Рисунок 1.6 — Скриншот результата работы make.

Загрузка модуля ядра с помощью команды insmod:

```

san_sanchez@LEX ~/workspace/0C/sem_2/lab_09/source/task_2 sudo insmod workq.ko
san_sanchez@LEX ~/workspace/0C/sem_2/lab_09/source/task_2 sudo lsmod | grep -B 1 workq
Module          Size  Used by
workq           16384  0

```

Рисунок 1.7 — Скришот демонстрации успешной загрузки модуля.

Демонстрация успешного вызова обработчика прерываний:

```

san_sanchez@LEX ~/workspace/0C/sem_2/lab_09/source/task_2 sudo dmesg | tail -8
[18957.481916] WORKQUEUE MODULE
                irq call count = 664
[18957.481941] WORKQUEUE MODULE
                { data: 128 }, current_time: 3:57:36:671350
[18957.734093] WORKQUEUE MODULE
                irq call count = 665
[18957.734117] WORKQUEUE MODULE
                { data: 128 }, current_time: 3:57:36:923529

```

Рисунок 1.8 — Скришот команды sudo dmesg.

Демонстрация содержимого /proc/interrupts:

```

san_sanchez@LEX ~/workspace/0C/sem_2/lab_09/source/task_2 sudo cat /proc/interrupts | head -4
CPU0      CPU1      CPU2      CPU3      CPU4      CPU5      CPU6      CPU7
0:         9         0         0         0         0         0         0      IR-I/O-APIC  2-edge  timer
1:         0         0      31062         0         0         0         0      IR-I/O-APIC  1-edge  i8042, workqueue_interrupt_handler
8:         0         0         0         1         0         0         0      IR-I/O-APIC  8-edge  rtc0

```

Рисунок 1.9 — Скришот команды cat /proc/interrupts.

Демонстрация успешной выгрузки модуля:

```

san_sanchez@LEX ~/workspace/0C/sem_2/lab_09/source/task_2 sudo rmmod workq.ko
san_sanchez@LEX ~/workspace/0C/sem_2/lab_09/source/task_2 sudo dmesg | tail -2
[18999.885558] WORKQUEUE MODULE
                unload module
san_sanchez@LEX ~/workspace/0C/sem_2/lab_09/source/task_2 sudo cat /proc/interrupts | head -4
CPU0      CPU1      CPU2      CPU3      CPU4      CPU5      CPU6      CPU7
0:         9         0         0         0         0         0         0      IR-I/O-APIC  2-edge  timer
1:         0         0      31168         0         0         0         0      IR-I/O-APIC  1-edge  i8042
8:         0         0         0         1         0         0         0      IR-I/O-APIC  8-edge  rtc0

```

Рисунок 1.10 — Скришот результата работы команды insmod.