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 Операционые системы.
Тема Обработчики прерываний.

Студент Степанов А. О.
Группа ИУ7-63Б
Оценка (баллы)
Преподаватель Рязанова Н.Ю.

Москва, 2020 г.

ТАСКЛЕТЫ

Листинг 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  #define HANDLEDIRQ 1
8
9  MODULE_AUTHOR("Alexander Stepanov");
10 MODULE_LICENSE("GPL");
11
12 static int irq = HANDLEDIRQ;
13 static int irq_call_count = 0;
14 static int dev_id;
15 char tasklet_data[] = "tasklet_function was called";
16
17 void tasklet_function(unsigned long data);
18
19 DECLARE_TASKLET(tasklet, tasklet_function, (unsigned long)&tasklet_data);
20
21 void tasklet_function(unsigned long data)
22 {
23     struct timeval t;
24     struct tm brocken;
25     do_gettimeofday(&t);
26     time_to_tm(t.tv_sec, 0, &brocken);
27
28     printk(KERN_INFO
29         "[tasklet_module] Tasklet: { state: %ld, count: %d, data: %s },"
30         "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
35 static irqreturn_t interrupt_handler(int irq, void *dev_id)
36 {
37     if (irq == HANDLEDIRQ)
38     {
39         irq_call_count++;
```

```
40          printk(KERN_INFO
41              "[tasklet_module] irq call count = %d\n", irq_call_count);
42          tasklet_schedule(&tasklet);
43          return IRQ_HANDLED;
44      }
45      else
46      {
47          return IRQ_NONE;
48      }
49  }
50
51  static int __init tasklet_module_init(void)
52  {
53      int ret = request_irq(
54          irq, interrupt_handler, IRQF_SHARED,
55          "tasklet_interrupt_handler", &dev_id
56      );
57
58      if (ret)
59      {
60          printk(KERN_ERR "[tasklet_module] error while handle irq\n");
61          return -1;
62      }
63
64      printk(KERN_INFO "[tasklet_module] success load\n");
65      return 0;
66  }
67
68  static void __exit tasklet_module_exit(void)
69  {
70      tasklet_kill(&tasklet);
71      free_irq(irq, &dev_id);
72      printk(KERN_INFO "[tasklet_module] unload module\n");
73  }
74
75  module_init(tasklet_module_init);
76  module_exit(tasklet_module_exit);
```

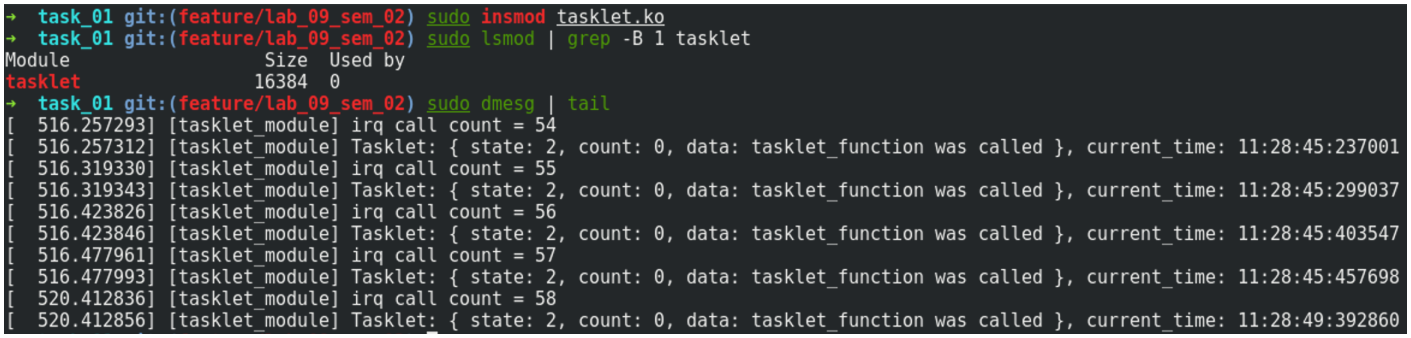

Рис. 1: Загрузка модуля

Рис. 2: Прерывания

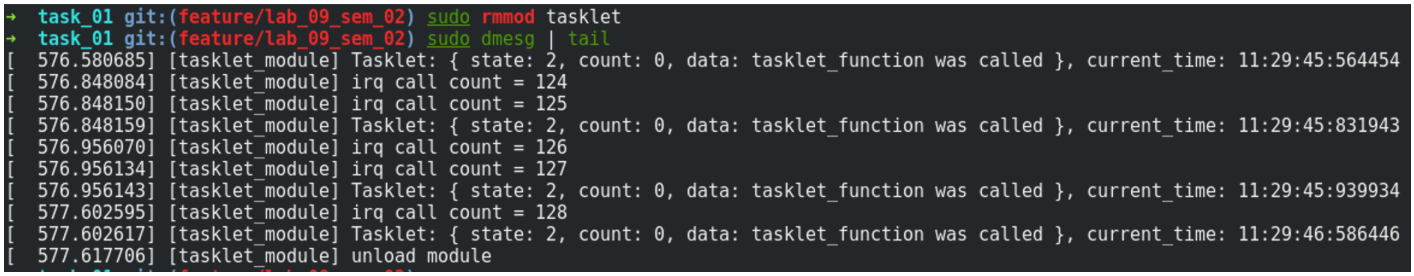

Рис. 3: Выгрузка модуля

ОЧЕРЕДИ РАБОТ

Листинг 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/workqueue.h>
6  #include <linux/time.h>
7
8  #define HANDLEDIRQ 1
9
10 MODULE_AUTHOR("Alexander␣Stepanov");
```

```
11 MODULE_LICENSE("GPL");
12
13 static int irq = HANDLEDIRQ;
14 static int irq_call_count = 0;
15 static int dev_id;
16 static struct workqueue_struct *workq = NULL;
17
18 void work_function(struct work_struct *work)
19 {
20     struct timeval t;
21     struct tm brocken;
22     do_gettimeofday(&t);
23     time_to_tm(t.tv_sec, 0, &brocken);
24
25     printk(KERN_INFO
26         "[workqueue_module] work: { data: %ld },"
27         "current_time: %d:%d:%d:%ld\n",
28         atomic_long_read(&work->data),
29         brocken.tm_hour + 3, brocken.tm_min, brocken.tm_sec, t.tv_usec);
30 }
31
32 DECLARE_WORK(work, work_function);
33
34 static irqreturn_t interrupt_handler(int irq, void *dev_id)
35 {
36     if (irq == HANDLEDIRQ)
37     {
38         irq_call_count++;
39         queue_work(workq, &work);
40         printk(KERN_INFO
41             "[workqueue_module] irq call count = %d\n", irq_call_count);
42         return IRQ_HANDLED;
43     }
44     else
45     {
46         return IRQ_NONE;
47     }
48
49 }
50
51 static int __init workqueue_module_init(void)
52 {
```

```
53      int ret = request_irq(
54          irq, interrupt_handler, IRQF_SHARED,
55          "workqueue_interrupt_handler", &dev_id
56      );
57
58      if (ret)
59      {
60          printk(KERN_ERR "[workqueue_module] error while handle irq\n");
61          return -1;
62      }
63
64      workq = create_workqueue("workqueue");
65
66      if (workq == NULL)
67      {
68          printk(KERN_ERR "[workqueue_module] error while create workqueue\n");
69          return -1;
70      }
71
72      printk(KERN_INFO "[workqueue_module] success load\n");
73      return 0;
74  }
75
76  static void __exit workqueue_module_exit(void)
77  {
78      flush_workqueue(workq);
79      destroy_workqueue(workq);
80      free_irq(irq, &dev_id);
81      printk(KERN_INFO "[workqueue_module] unload module\n");
82  }
83
84  module_init(workqueue_module_init);
85  module_exit(workqueue_module_exit);
```

```
→  task_02 git:(feature/lab_09_sem_02) sudo insmod workq.ko
→  task_02 git:(feature/lab_09_sem_02) sudo lsmod | grep -B 1 workq
Module                  Size  Used by
workq                  16384  0
→  task_02 git:(feature/lab_09_sem_02) sudo dmesg | tail
[  620.838050] [workqueue_module] irq call count = 82
[  620.838064] [workqueue_module] work: { data: 64 }, current_time: 11:30:29:824306
[  620.924572] [workqueue_module] irq call count = 83
[  620.924637] [workqueue_module] work: { data: 64 }, current_time: 11:30:29:910885
[  621.024974] [workqueue_module] irq call count = 84
[  621.025064] [workqueue_module] work: { data: 64 }, current_time: 11:30:30:11316
[  621.092699] [workqueue_module] irq call count = 85
[  621.092708] [workqueue_module] work: { data: 64 }, current_time: 11:30:30:78964
[  621.204109] [workqueue_module] irq call count = 86
[  621.204144] [workqueue_module] work: { data: 64 }, current_time: 11:30:30:190405
```

Рис. 4: Загрузка модуля

```
→  task_02 git:(feature/lab_09_sem_02) sudo cat /proc/interrupts
           CPU0       CPU1
  0:          8          0   IO-APIC   2-edge      timer
  1:         11       1964   IO-APIC   1-edge      i8042, workqueue_interrupt_handler
```

Рис. 5: Прерывания

```
→  task_02 git:(feature/lab_09_sem_02) sudo rmmod workq
→  task_02 git:(feature/lab_09_sem_02) sudo dmesg | tail
[  646.622481] [workqueue_module] work: { data: 64 }, current_time: 11:30:55:610022
[  646.697647] [workqueue_module] irq call count = 153
[  646.697669] [workqueue_module] work: { data: 64 }, current_time: 11:30:55:685214
[  646.749668] [workqueue_module] irq call count = 154
[  646.749694] [workqueue_module] work: { data: 64 }, current_time: 11:30:55:737241
[  646.830364] [workqueue_module] irq call count = 155
[  646.830398] [workqueue_module] work: { data: 64 }, current_time: 11:30:55:817949
[  647.588876] [workqueue_module] irq call count = 156
[  647.589030] [workqueue_module] work: { data: 64 }, current_time: 11:30:56:576619
[  647.600779] [workqueue_module] unload module
```

Рис. 6: Выгрузка модуля