|  |  |
| --- | --- |
| 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»*

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** |  |

**Дисциплина:  *Операционные системы***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ***ИУ7И-66Б*** |  |  | **Нгуен Ф. С.** |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | **Рязанова Н. Ю.** |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

*Москва, 2021*

**I. Tasklet**

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

#include <linux/interrupt.h>

#define **SHARED\_IRQ** 1

MODULE\_LICENSE("Dual BSD/GPL");

MODULE\_AUTHOR("Nguyensanghso@gmail.com");

static int my\_dev\_id;

char tasklet\_data[] = "This is tasklet data";

**void tasklet\_handler(unsigned long data);**

**DECLARE\_TASKLET**(my\_tasklet, tasklet\_handler, (unsigned long) &tasklet\_data);

**void tasklet\_handler(unsigned long data)**

{

printk(KERN\_INFO "[TL] Tasklet: state - %ld, count - %d, data - %s\n",

my\_tasklet.state, my\_tasklet.count, my\_tasklet.data);

}

**static irqreturn\_t my\_interrupt(int irq, void \*dev\_id)**

{

if (irq == SHARED\_IRQ)

{

printk(KERN\_INFO "[TL] Tasklet scheduled\n");

tasklet\_schedule(&my\_tasklet);

return IRQ\_HANDLED;

}

else

return IRQ\_NONE;

}

**static int \_\_init my\_tasklet\_init(void)**

{

if (request\_irq(SHARED\_IRQ, my\_interrupt, IRQF\_SHARED, "my\_interrupt", &my\_dev\_id))

{

printk(KERN\_ERR "[TL] Error on request\_irq\n");

return -1;

}

printk(KERN\_INFO "[TL] Module loaded!\n");

return 0;

}

**static void \_\_exit my\_tasklet\_exit(void)**

{

tasklet\_kill(&my\_tasklet);

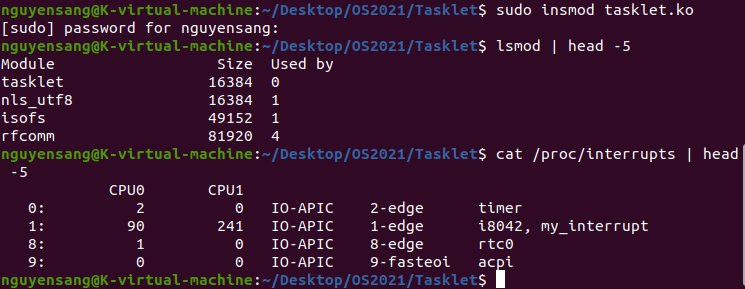
free\_irq(SHARED\_IRQ, &my\_dev\_id);

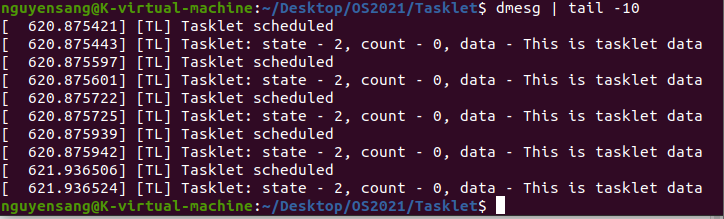
printk(KERN\_INFO "[TL] Module unloaded!\n");

}

module\_init(my\_tasklet\_init);

module\_exit(my\_tasklet\_exit);





**II. Work Queue**

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

#include <linux/interrupt.h>

#include <linux/workqueue.h>

#define IRQ 1

MODULE\_LICENSE("Dual BSD/GPL");

MODULE\_AUTHOR("Nguyensanghso@gmail.com");

**static int my\_dev\_id;**

**static int irq\_call\_n = 0;**

**struct workqueue\_struct \*wq;**

**void my\_workqueue\_function(struct work\_struct \*work)**

{

printk(KERN\_INFO "[WQ] Workqueue: counter %d\n", ++irq\_call\_n);

}

**DECLARE\_WORK(my\_workqueue, my\_workqueue\_function);**

**static irqreturn\_t my\_interrupt(int irq, void \*dev\_id)**

{

if (irq == IRQ)

{

queue\_work(wq, &my\_workqueue);

printk(KERN\_INFO "[WQ] Workqueue in my\_interrupt\n");

return IRQ\_HANDLED;

}

else

return IRQ\_NONE;

}

**static int \_\_init my\_workqueue\_init(void)**

{

if (request\_irq(IRQ, my\_interrupt, IRQF\_SHARED, "my\_interrupt", &my\_dev\_id))

{

printk(KERN\_ERR "[WQ] Error: can't register handler\n");

return -1;

}

wq = create\_workqueue("workqueue");

if (wq)

printk(KERN\_INFO "[WQ] Workqueue created!\n");

else

{

free\_irq(IRQ, &my\_dev\_id);

printk(KERN\_ERR "[WQ] Error: can't create workqueue\n");

return -ENOMEM;

}

printk(KERN\_INFO "[WQ] Module loaded!\n");

return 0;

}

**static void \_\_exit my\_workqueue\_exit(void)**

{

flush\_workqueue(wq);

destroy\_workqueue(wq);

free\_irq(IRQ, &my\_dev\_id);

printk(KERN\_INFO "[WQ] Module unloaded\n");

}

module\_init(my\_workqueue\_init)

module\_exit(my\_workqueue\_exit)

