НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. І. СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ І ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №6

з курсу «ДПКС»

Виконав:

Студент 5-го курсу ФІОТ

Групи ІВ-91мп

Захарчук Д. В.

Київ – 2020

Завдання:

Ознайомитися з можливостями дебагу модуля.

Хід виконання Basic 1:

Модифікуємо файли, експортуємо необхідні параметри та виконуємо збирання модулю. Створюємо архів CPIO для roofs. Запускаємо емулятор.

```
dmitriy@dmitriy:~/repos/busybox/_install$ mkdir lab6
dmitriy@dmitriy:~/repos/busybox/_install$ cd lab6
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ ls -l
total 12
-rw-rw-rw- 1 dmitriy dmitriy 2200 чер  3 01:29 hello.c
-rw-rw-rw- 1 dmitriy dmitriy   44 чер  3 01:29 Kbuild
-rw-rw-rw- 1 dmitriy dmitriy  136 чер  3 01:29 Makefile
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ mv hello.c module6.c
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ nano module6.c
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ export PATH=/opt/gcc-arm-8.3-2019.03-x86_64-arm-ea
bi/bin:$PATH
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ export CROSS_COMPILE='ccache arm-eabi-'
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ export ARCH=arm
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ export KDIR=/home/dmitriy/repos/linux-stable/
```

```
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ make
make -C /home/dmitriy/repos/linux-stable/ M=$PWD
make[1]: Entering directory '/home/dmitriy/repos/linux-stable'
  CC [M]  /home/dmitriy/repos/busybox/_install/lab6/module6.o
/home/dmitriy/repos/busybox/_install/lab6/module6.c: In function 'module6_init':
/home/dmitriy/repos/busybox/_install/lab6/module6.c:44:2: error: expected ';' before 'if'
  if (repeats >= 5 && repeats <= 10)
  ^~
scripts/Makefile.build:309: recipe for target '/home/dmitriy/repos/busybox/_install/lab6/module6.
o' failed
make[2]: *** [/home/dmitriy/repos/busybox/_install/lab6/module6.o] Error 1
Makefile:1522: recipe for target '_module_/home/dmitriy/repos/busybox/_install/lab6' failed
make[1]: *** [_module_/home/dmitriy/repos/busybox/_install/lab6] Error 2
make[1]: Leaving directory '/home/dmitriy/repos/linux-stable'
Makefile:6: recipe for target 'default' failed
make: *** [default] Error 2
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ nano module6.c
Use "fg" to return to nano.

[7]+  Stopped                 nano module6.c
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ nano module6.c
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ make
make -C /home/dmitriy/repos/linux-stable/ M=$PWD
make[1]: Entering directory '/home/dmitriy/repos/linux-stable'
  CC [M]  /home/dmitriy/repos/busybox/_install/lab6/module6.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/dmitriy/repos/busybox/_install/lab6/module6.mod.o
  LD [M]  /home/dmitriy/repos/busybox/_install/lab6/module6.ko
make[1]: Leaving directory '/home/dmitriy/repos/linux-stable'
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ cd ..
dmitriy@dmitriy:~/repos/busybox/_install$ find . | cpio -o -H newc | gzip > ../rootfs.cpio.gz
119205 blocks
dmitriy@dmitriy:~/repos/busybox/_install$ cd ..
dmitriy@dmitriy:~/repos/busybox$ qemu-system-arm -kernel _install/boot/zImage -initrd rootfs.cpio
.gz -machine virt -nographic -m 512 --append "root=/dev/ram0 rw console=ttyAMA0,115200 mem=512M"
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.19.116 (dmitriy@dmitriy) (gcc version 8.3.0 (GNU Toolchain for the
 A-profile Architecture 8.3-2019.03 (arm-rel-8.36))) #1 SMP Wed Apr 22 00:03:26 EEST 2020
[    0.000000] CPU: ARMv7 Processor [412fc0f1] revision 1 (ARMv7), cr=10c5387d
[    0.000000] CPU: div instructions available: patching division code
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, PIPT instruction cache
```

Модифікуємо Kbuild файл

```
dmitriy@dmitriy:~/repos/busybox/_install$ cd lab6/
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ cat Kbuild

# kbuild part of makefile
ccflags-y += -g
obj-m := module6.o
dmitriy@dmitriy:~/repos/busybox/_install/lab6$
```

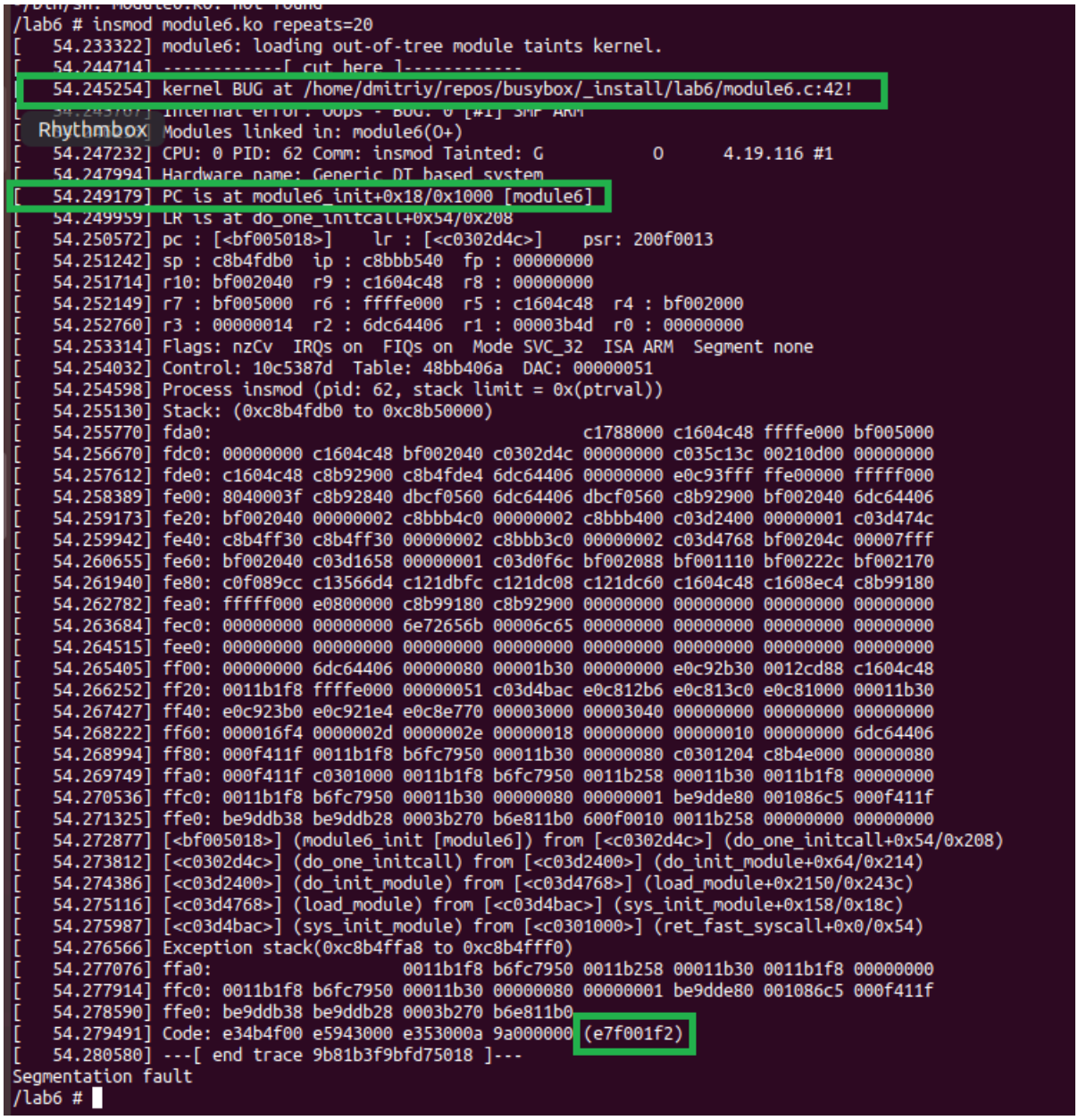
Лістинг коду модуля.

```
#define DEBUG

#include <linux/init.h>
#include <linux/module.h>
#include <linux/printk.h>

#include <linux/ktime.h>
#include <linux/slab.h>

static LIST_HEAD(lab_list_head);

struct time_keeper {
	ktime_t time_before;
	ktime_t time_after;
	struct list_head time_list;
};

static void print_text(unsigned int repeats)
{
	unsigned int repeat;
	struct time_keeper *ptr;

	for (repeat = 0; repeat < repeats; repeat++) {
		if (repeat == 2)
			ptr = 0; <- Навмисно створюємо помилку
		else
			ptr = kmalloc(sizeof(*ptr), GFP_KERNEL);
		ptr->time_before = ktime_get();
		pr_info("Hello there!\n");
		ptr->time_after = ktime_get();
		list_add(&ptr->time_list, &lab_list_head);
	}
}

static unsigned int repeats = 1;

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "How many times to print hello");

static int __init module6_init(void)
```

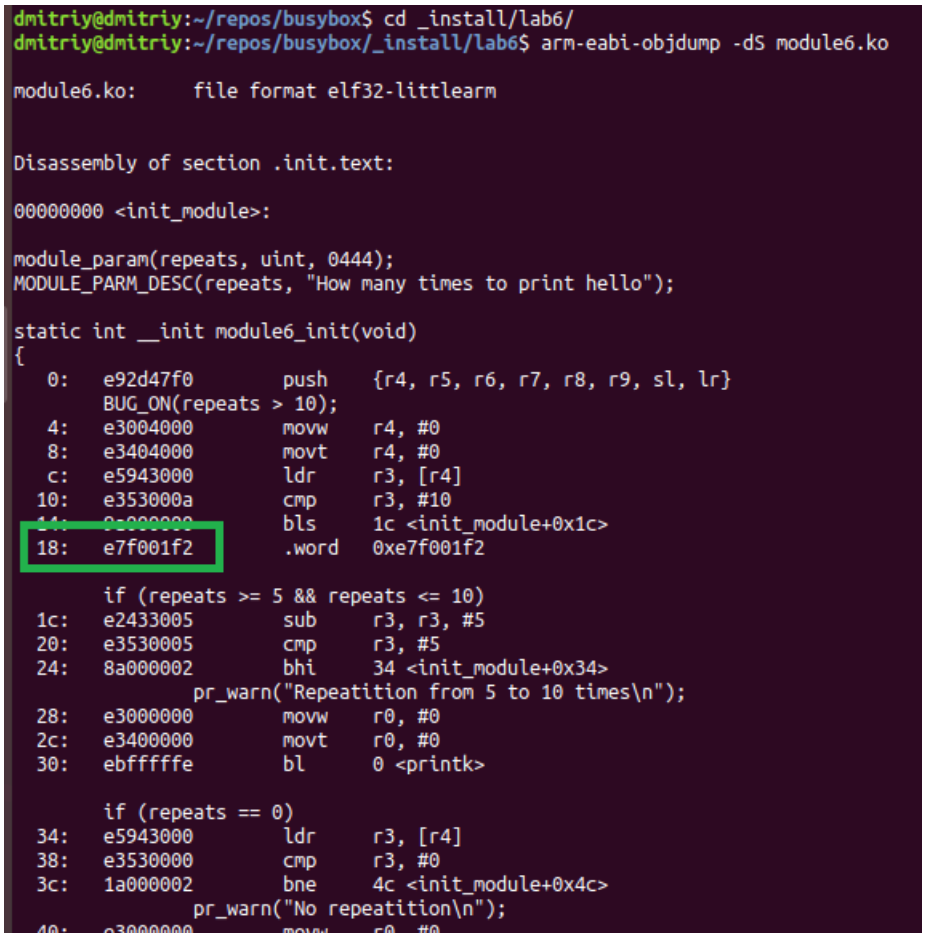
```
{
	BUG_ON(repeats > 10); <- Виклик BUG_ON

	if (repeats >= 5 && repeats <= 10)
		pr_warn("Repeatition from 5 to 10 times\n");

	if (repeats == 0)
		pr_warn("No repeatition\n");

	print_text(repeats);
	return 0;
}

static void __exit module6_exit(void)
{
	struct list_head *p;
	struct list_head *n;
	struct time_keeper *curr;

	pr_info("Module 5 exit\n");

	list_for_each_safe(p, n, &lab_list_head) {
		curr = list_entry(p, struct time_keeper, time_list);
		pr_info("Time needed for printing is: %lld(ns).\n",
				curr->time_after - curr->time_before);
		list_del(p);
		kfree(curr);
	}
}

module_init(module6_init);
module_exit(module6_exit);

MODULE_AUTHOR("Dmytro Zakharchuk");
MODULE_DESCRIPTION("Test work with debug");
MODULE_LICENSE("Dual BSD/GPL");
```

Результат виконання:

Введемо параметер який буде більше 10

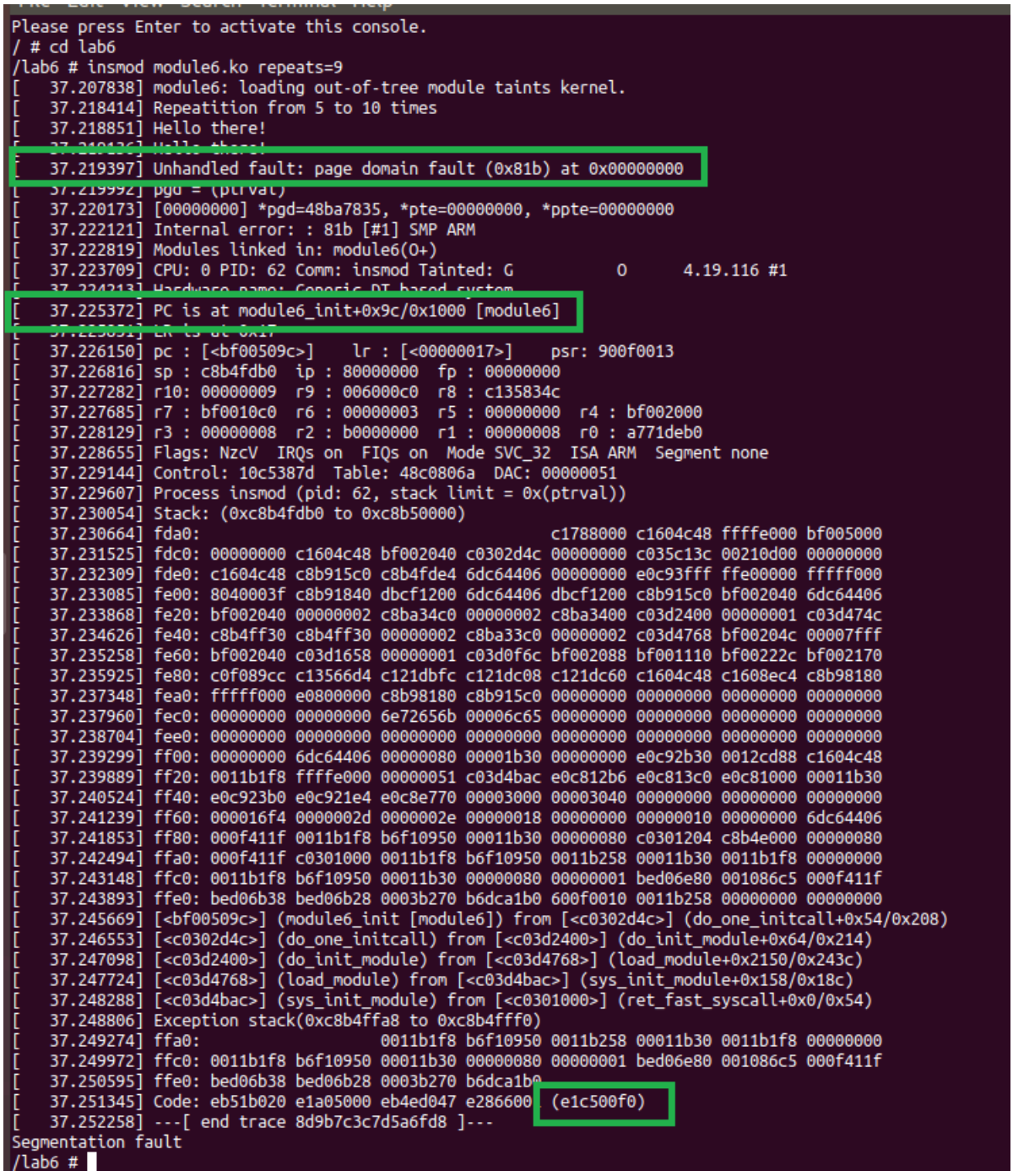


Запустимо objdump та побачимо що значення PC та рядку ідентичні.

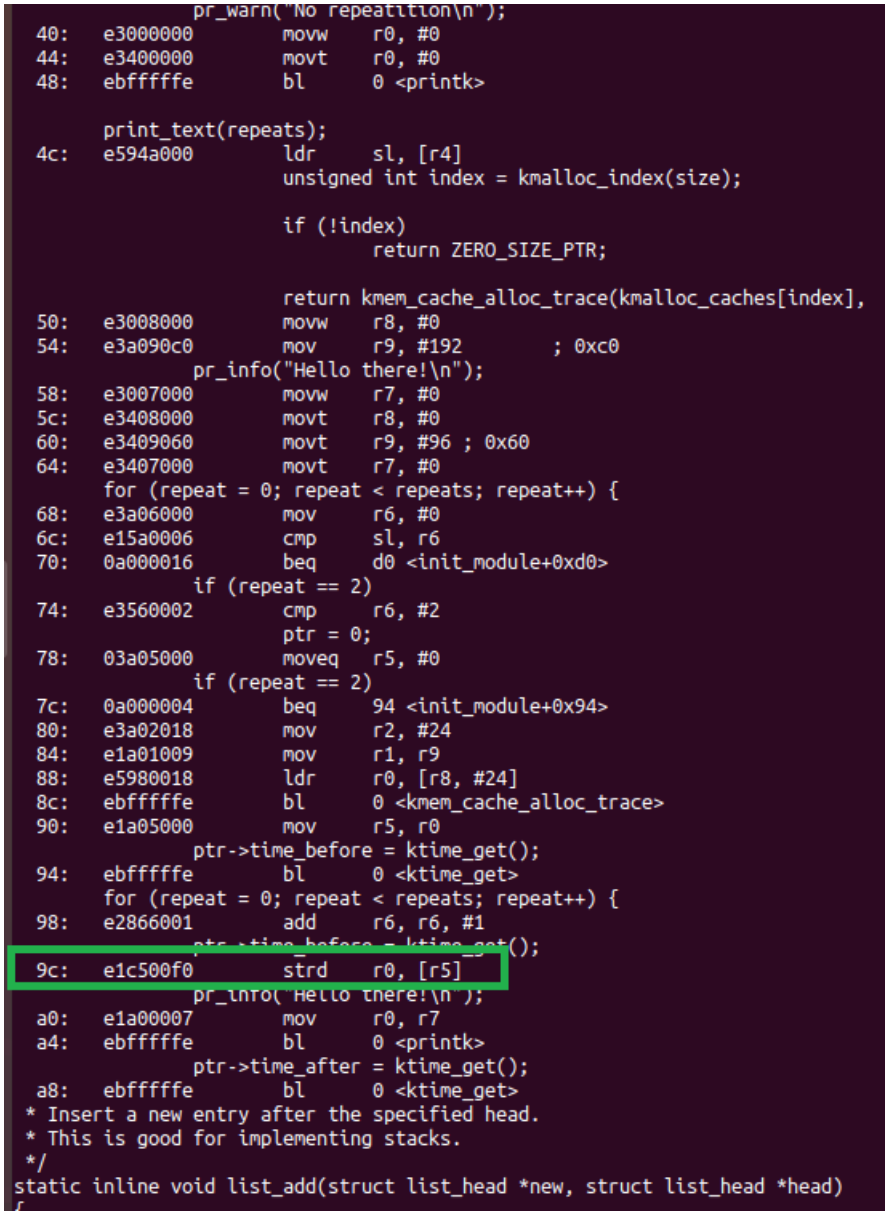


```
dmitriy@dmitriy:~/repos/busybox$ cd _install/lab6/
dmitriy@dmitriy:~/repos/busybox/_install/lab6$ arm-eabi-objdump -dS module6.ko

module6.ko:     file format elf32-littlearm


Disassembly of section .init.text:

00000000 <init_module>:

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "How many times to print hello");

static int __init module6_init(void)
{
   0:   e92d47f0        push    {r4, r5, r6, r7, r8, r9, sl, lr}
        BUG_ON(repeats > 10);
   4:   e3004000        movw    r4, #0
   8:   e3404000        movt    r4, #0
   c:   e5943000        ldr     r3, [r4]
  10:   e353000a        cmp     r3, #10
  14:   9a000000        bls     1c <init_module+0x1c>
  18:   e7f001f2        .word   0xe7f001f2

        if (repeats >= 5 && repeats <= 10)
  1c:   e2433005        sub     r3, r3, #5
  20:   e3530005        cmp     r3, #5
  24:   8a000002        bhi     34 <init_module+0x34>
                pr_warn("Repeatition from 5 to 10 times\n");
  28:   e3000000        movw    r0, #0
  2c:   e3400000        movt    r0, #0
  30:   ebfffffe        bl      0 <printk>

        if (repeats == 0)
  34:   e5943000        ldr     r3, [r4]
  38:   e3530000        cmp     r3, #0
  3c:   1a000002        bne     4c <init_module+0x4c>
                pr_warn("No repeatition\n");
  40:   e3000000        movw    r0, #0
```

Введемо значення від 5 до 10.



Запустимо objdump.


pr_warn("No repeattition\n");
40: e3000000 movw r0, #0
44: e3400000 movt r0, #0
48: ebffffe bl 0 <printk>
print_text(repeats);
4c: e594a000 ldr sl, [r4]
unsigned int index = kmalloc_index(size);
if (!index)
return ZERO_SIZE_PTR;
return kmem_cache_alloc_trace(kmalloc_caches[index],
50: e3008000 movw r8, #0
54: e3a090c0 mov r9, #192 ; 0xc0
pr_info("Hello there!\n");
58: e3007000 movw r7, #0
5c: e3408000 movt r8, #0
60: e3409060 movt r9, #96 ; 0x60
64: e3407000 movt r7, #0
for (repeat = 0; repeat < repeats; repeat++) {
68: e3a06000 mov r6, #0
6c: e15a0006 cmp sl, r6
70: 0a000016 beq d0 <init_module+0xd0>
if (repeat == 2)
74: e3560002 cmp r6, #2
ptr = 0;
78: 03a05000 moveq r5, #0
if (repeat == 2)
7c: 0a000004 beq 94 <init_module+0x94>
80: e3a02018 mov r2, #24
84: e1a01009 mov r1, r9
88: e5980018 ldr r0, [r8, #24]
8c: ebffffe bl 0 <kmem_cache_alloc_trace>
90: e1a05000 mov r5, r0
ptr->time_before = ktime_get();
94: ebffffe bl 0 <ktime_get>
for (repeat = 0; repeat < repeats; repeat++) {
98: e2866001 add r6, r6, #1
9c: e1c500f0 strd r0, [r5]
pr_info("Hello there!\n");
a0: e1a00007 mov r0, r7
a4: ebffffe bl 0 <printk>
ptr->time_after = ktime_get();
a8: ebffffe bl 0 <ktime_get>
* Insert a new entry after the specified head.
* This is good for implementing stacks.
*/
static inline void list_add(struct list_head *new, struct list_head *head)