EI338 Lab Report Chapter2

Bao Xiaoyi 517030910306

October 16, 2019

1 Environment

- VirtualBox 5.2.18
- Ubuntu 14.04 (64 bit) running on the virtual machine

2 Assignment

1. Design a kernel module that creates a `/proc` file named `/proc/jiffies` that reports the current value of `jiffies` when the `proc/jiffies` file is read, such as the command

   ```
   cat /proc/jiffies
   ```

   Be sure to remove `/proc/jiffies` when the module is removed.

2. Design a kernel module that creates a `proc` file named `/proc/seconds` that reports the number of elapsed seconds since the kernel module was loaded. This ill involve using the value of `jiffies` as well as the `HZ` rate. When a user enters the command

   ```
   cat /proc/seconds
   ```

   your kernel module will report the number of seconds that elapsed since the kernel module was first loaded. Be sure to remove the `/proc/seconds` when the module is removed.

3 Decomposition and Analysis

Here, we try to decompose the original tasks into smaller parts, and solve the sub problems one by one.

1. Obtain the value of `HZ` and `jiffies` from certain module.

   Solution. *As noted in the textbook, the tick rate is the value `HZ` defined in `<asm/param.h>`. Also, the number of timer interrupts that have occured since the system was booted is stored as the global variable `jiffies` in `linux/jiffies.h`. Thus, we only need to import the two files, and make use of `HZ` and `jiffies` inside them.*

2. Consider the fact the variable `jiffies` changes its value every time we call it, we need to store its value when the module is first loaded.

   Solution. *we can store the original value of `jiffies` in a global variable. That is, it is at the same level as main. In this way, we can require it from any part of the code.*

3. Figure out the relationship between elapsed time, frequency, and the number of timer interrupt.

   Solution. *. Here, frequency means "ticks per second". We can represent the relationship among them with an equation:*

$$\begin{aligned} t &= \frac{\Delta interrupt}{tick\_per\_second} \\ &= \frac{jiffies_2 - jiffirs_1}{HZ} \end{aligned}$$

4. Find a way to pass certain information from kernel to user.

   Solution. *The example file, `hello.c` may give us some hint on the topic. As is shown in the textbook, a buffer is used to store the message, and an extra integer variable contains the length of the string.*

5. Use terminal in Linux to check our solution.

   Solution. *We just need to compile the file, add it to the kernel, then interact with the command line.*

4 Details

In this part, some codes are shown for better illustration. We mainly show the difference between our file and the example file, `hello.c`

- Two files that we include to deal with assignment1.

```
/* jiffies.c*/
#include <asm/uaccess.h>
#include <linux/jiffies.h>

/* in function proc_read */
rv = sprintf(buffer,
"The current value of jiffies is : %lu \n", jiffies);

/* the length, together with the length,
is passed to the user */
copy_to_user(usr_buf, buffer, rv);
```

- We made some further modification to the original file to satisfy assignment2.

```
/*seconds.c */

/* the global variable is defined in the beginning.
*/

unsigned long old_jiffies;

/* in function proc_init,
we output some basic information in the kernel */

old_jiffies = jiffies;

printk(KERN_INFO "frequency of the time interrupt: %d\n", HZ);

printk(KERN_INFO "jiffies when the module is loaded: %lu\n",
    old_jiffies);


/* the buffer, together with the length,
is passed to the user */

copy_to_user(usr_buf, buffer, rv);
```

```
/* in function proc_read */

int time_elasped = (jiffies - old_jiffies) / HZ;

/* in the kernel */

printk(KERN_INFO "Jiffies when the function is called: %lu\n", jiffies);

printk(KERN_INFO "Time elasped : %d \n", time_elasped);

/* to the user */

rv = sprintf(buffer, "Time elasped : %d \n", time_elasped);
copy_to_user(usr_buf, buffer, rv);
```

5 Result

Here are the results of our experiment:

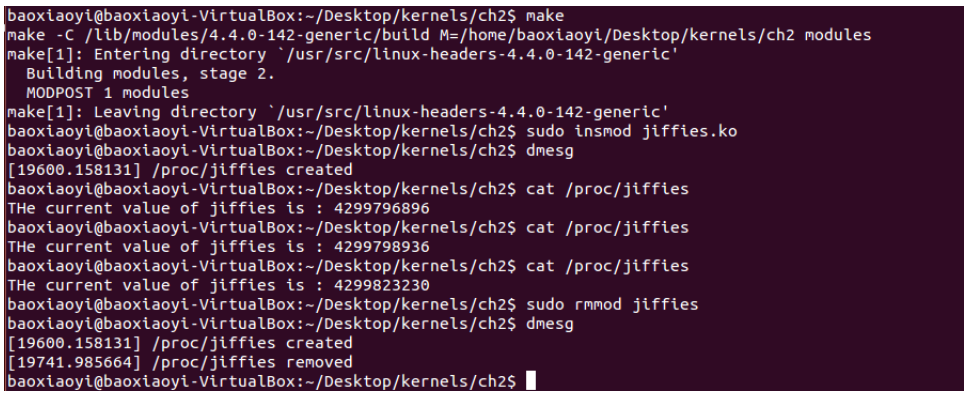

Figure 1: Result for assignment 1

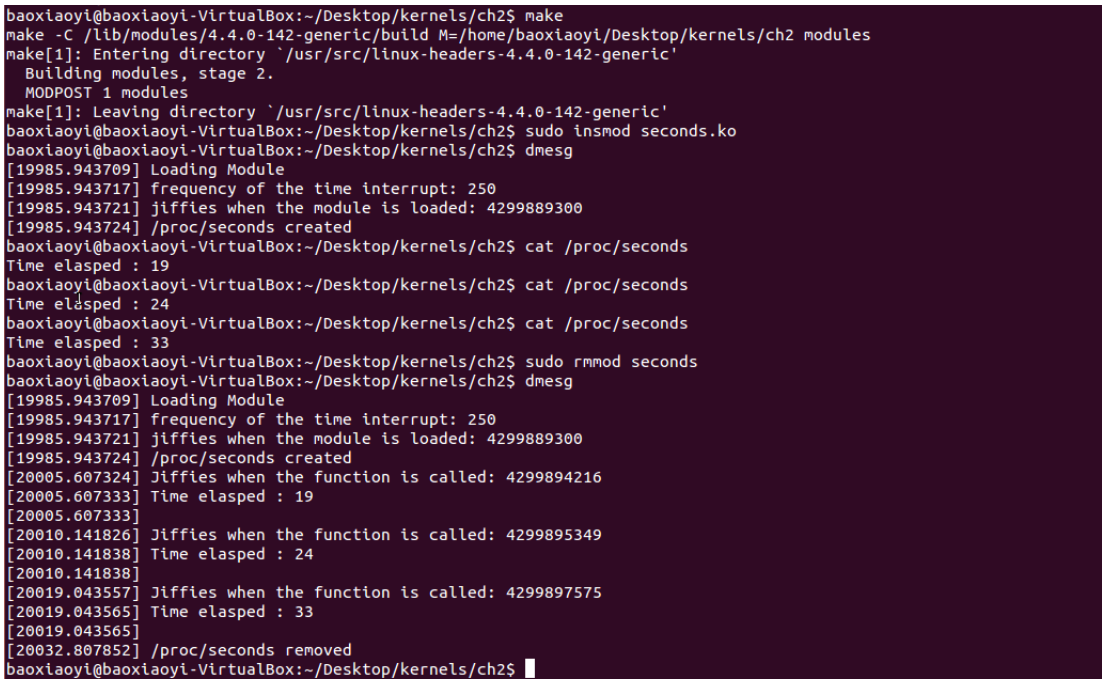

Figure 2: Result for assignment 2

6 Future work

It is just the start of our exploration into operating system, and of course there are a lot more to be done. To be more specific, there are two points that we may improve in the future:

- Elapsed time as a floating number. Thereby, the description of elapsed time can be more precise to users.
- A pointer for passing the value of jiffies when the module is loaded.