CSCI 332 Spring 2019
Programming Assignment 2

Objectives: Multi-Thread Programming and Synchronization in Kernel layer

1. Understand and Practice Loadable Kernel Module (LKM) programming.
2. Understand and Practice Kernel Threads and Synchronization operations.
3. (Could be related to developing I/O device drivers)

Rules:

This programming assignment will be graded on **structure** considering the constraints as well as **correctness**. There should be comments and indentation, variables and functions should have meaningful names, and the program should have a clear, simple structure.

The programming assignment must be individual work. Do not discuss it with your classmates except understanding of this problem and clarifying your idea/pseudo code. **Do not share your solution with anyone**, in person or electronically, until the end of the semester. Do not discuss the assignment online, or copy online solutions.

If you find some codes online, **you MUST cite both the website domain and the source codes,** which will be counted as deduction points; in other words it is strictly prohibited to copy some or most of the source codes online. (How to cite will be shown in the end of this file.)

Your program will be run through an automated plagiarism checker. If you are found to have cheated on any single programming assignment, **you will receive a 0 on the programming assignment**. Late work on this assignment will be accepted, with a penalty of 30% per day.

Write a LKM file that:

1. Create two kernel threads (one ***give*** and one ***take*** threads), explicitly starting the **take** thread firstly.
2. The ***give*** thread prints the "***lkm: give***" message with a busy waiting of 1 second; And then schedule the ***take*** thread using a process synchronization technique.
3. The ***take*** thread prints the "***lkm: take***" message with a busy waiting of 1 second; And then schedule the ***give*** thread using the process synchronization technique.
4. If there are critical sections, please protect the sections using a data synchronization technique.

Programming requirements:
Please submit only your ***lkm.c*** **file** on the moodle (one lkm.c file), which **MUST** be executable using the below commands for execution.

Tips for Implementation:
NOTE: After careful design, please implement. (Unlike POSIX programming, if there is a deadlock, you MUST reboot your system).

Commands for Execution:
$ make
$ sudo insmod lkm.ko
$ sudo rmmod lkm
$ dmesg | grep lkm

Sample Output:

[2842.604561] lkm: kthread init called
[2842.604765] lkm: the take thread was generated firstly
[2843.101362] lkm: the give thread was generated secondly
[2843.101365] lkm: give
[2844.105347] lkm: take

[2845.109362] lkm: give
[2846.113186] lkm: take
[2847.117962] lkm: give
[2848.121435] lkm: take
[2849.126019] lkm: give
[2850.129371] lkm: take
[2851.133685] lkm: give
([2852.133685] lkm: take) ← this print depends on kthread stop timing
[2852.137834] lkm: kthread exit called

How to check 1 second delay? The time difference (e.g., for the first take print, 2844.105347 - 2843.101365 (the values can be different))

References: in addition to the Loadable Kernel Moudle(LKM) on the moodle

// **Mutex Initialization and Operations**

struct mutex my_mutex;
mutex_init(&my_mutex);

void mutex_lock(&my_mutex);
void mutex_unlock(&my_mutex);

// **Semaphore Initialization and Operations**

#include <linux/semaphore.h>

struct semaphore my_sem;
sema_init(&my_sem, val);

void down((&sem);
void up(&sem);

How to cite both the website domain and the source codes:

Below the module_exit(), please add the comments and the full domain address with the source codes like the example **(** without any white spaces or indentation for http(s)). If you referred many sites, please add consecutively.

module_exit(xxx_exit);

/*START_CITE

http://www.domain1.com (or https://xxx)

Referred code 1 (just example)

```
void myMemCpy(void *dest, void *src, size_t n)
{
   // Typecast src and dest addresses to (char *)
   char *csrc = (char *)src;
   char *cdest = (char *)dest;

   // Copy contents of src[] to dest[]
   for (int i=0; i<n; i++)
       cdest[i] = csrc[i];
}
```

https://www.domain2.com

Referred code 2

END_CITE*/