Assignment for Lab 12
Cache Memory and Processes
Computer Science 240

NOTE: the lecture notes and assigned reading should be helpful in understanding cache and process terminology

1. Cache Memory

Each of the 2 **direct-mapped cache** labeled A and B in the table below use a **32-bit address,** with the specified mapping for Tag, Index, and Offset bits:

	Tag	Index	Offset
A	31-10	9-4	3-0
B	31-12	11-5	4-0

What is the cache line/block size (number of bytes stored in each cache entry, encoded by the Offset bits) for:

Cache A?

Cache B?

How many lines/entries does each cache have (encoded by the Index bits) for:

Cache A?

Cache B?

Each line/entry in the cache contains the data for that line, the Tag bits, and a single Valid bit. What is the ratio for each line between the total bits stored to data bits only?

Cache A?

Cache B?

2. Explore Processes in a Running System

On a local lab machine in the Micro-focus, open a couple applications: use Firefox to view both the lab web and at least one additional more dynamic web page, such as Facebook. Also, open at least one other application.

Run the **top** command to visualize the currently executing processes and the resources they consume

```
$ top
```

Examine the output. Read the “Tasks” line, interpreting “Tasks” as “processes”.

How many processes are running? Sleeping? Stopped? Zombie?

Read the “CPU(s)” line, which shows the percentage of the time that the CPU is spending executing **us**er and operating **sy**stem kernel code, vs. being **id**le (and a few other categories we will ignore). These levels probably fluctuate at each sample that *top* displays.

Also displayed are a list of processes ranked by the percentage of CPU time they have used in the most recent time window. Which processes are using the most CPU time? About how much?

Run the **ps** command:

```
$ ps
```

By default, it lists only the processes run under your current login session. (Each terminal window you open actually creates a new login session and runs a shell in it.) For example:

```
23314 pts/1    00:00:00 bash
30086 pts/1    00:00:00 ps
```

Run **ps ux** to see the list of all processes belonging to you:

```
$ ps ux
```

How many have used at least 1 second of CPU time? (see the TIME column, in minutes:seconds form)

Run **ps aux** to see the list of all processes run by all users on this machine:

```
$ ps aux
```

List the contents of the **/proc** filesystem, which is provided by the Linux kernel as an interface to inspect information about process scheduling, individual processes, and other operating system status information.

```
$ ls /proc
```

You will see something like this, which is a list of subdirectories:

```
1     12327  171    217    248    29     315    3518  47    8512
10     12328  172    218    24801  290    3152   3519  48    86
100    124    173    219    249    29040  316    352   483   87
101    125    174    22     24946  291    31654  3520  49    88
```

The **/proc** filesystem has a subdirectory with information about each living process. Each directory is named with the associated PID (Proccess ID) of a process that is currently running.

9. Examine the interrupts file:

```
$ cat   /proc/interrupts
```

How many interrupts have occurred for scheduling (context-switching)? System calls (traps, labeled “Function Call”)?

Find the PID (Process ID) of *firefox*:

```
$ top
```

and find *firefox* and its listed PID.

Change into that directory (for example, if the PID of *firefox* is 1179):

```
$ cd /proc/1179
```

10. Inspect its status information by showing the contents of the *status* file:

```
$ cat status
```

How many context switches has Firefox experienced? (Look for “switches” or “ctxt switches”.)

How many child processes has Firefox created? (See the “task” subdirectory or run the **pstree -p** command to see the hierarchy of process ancestry.)