

Full Name: _____

CS 450 Fall 2009

Final Exam

December 9, 2009

Instructions:

- This exam is closed-book, closed-notes.
- Write your full name on the front, and make sure that your exam is not missing any sheets.
- Good luck!

Problem 1	(/16) :
Problem 2	(/12) :
Problem 3	(/12) :
Problem 4	(/8) :
Problem 5	(/12) :
TOTAL	(/60) :

Problem 1. (16 points):

Multiple choice. For each of the following multiple choice problems, choose the *single best* answer by circling its corresponding letter.

1. Which of the following bits of code carries out a context switch to the kernel's scheduling/swapping process, a.k.a. `proc[0]`?
 - (a) `p = &proc[0];`
 - (b) `retu(proc[0].p_addr);`
 - (c) `u.u_procp = rpp;`
 - (d) `proc[0].p_flag |= SLOAD|SSYS;`
2. Which of the following data structures associated with active processes can be swapped out when the process is not running?
 - (a) the `proc` struct
 - (b) the `callout` array
 - (c) the interrupt vector
 - (d) the kernel stack
3. It is frequently the case that v6 code needs to be run *atomically*, i.e., without the possibility of being interrupted by code that will modify crucial data structures concurrently. Which of the following can be used to *begin* an atomic chunk of code?
 - (a) `spl0()`
 - (b) `rp->p_stat = SWAIT`
 - (c) `mov $1, SSR0`
 - (d) `bis $340, PS`
4. The current user struct can be accessed in the kernel via the pointer `_u`. Which of the following correctly initializes `_u`?
 - (a) `_u = 140000`
 - (b) `_u = *ka6`
 - (c) `mov $USIZE-1\<8|6, _u`
 - (d) `UISA->r[7] = ka6[1]`

Problem 2. (12 points):

This problem is based on the following bit of code taken from the `newproc` function:

```
1859  rip = u.u_procp;
1860  up = rip;
1861  rpp->p_stat = SRUN;
1862  rpp->p_flag = SLOAD;
1863  rpp->p_uid = rip->p_uid;
1864  rpp->p_ttyp = rip->p_ttyp;
1865  rpp->p_nice = rip->p_nice;
1866  rpp->p_textp = rip->p_textp;
1867  rpp->p_pid = mpid;
1868  rpp->p_ppid = rip->p_pid;
1869  rpp->p_time = 0;
```

- Which of the pointers `rpp` and `rip` points to the newly created process? Justify your answer.
- Explain the purpose of lines 1868 and 1869.
- Does line 1861 make the process identified by `rpp` the currently running process? Why or why not?

Problem 3. (12 points):

This problem is based on the following section of code:

```
n = -1;
for(rp = &proc[0]; rp < &proc[NPROC]; rp++)
    if((rp->p_flag&(SSYS|SLOCK|SLOAD))==SLOAD &&
        (rp->p_stat==SRUN || rp->p_stat==SSLEEP) &&
        rp->p_time > n) {
        p1 = rp;
        n = rp->p_time;
    }
```

- What is the purpose of this `for` loop?
- Upon exiting the loop, what does the variable `n` refer to?
- Where and when would you expect to find this code executed in the kernel?

Problem 4. (8 points):

Of the value returned by `swtch`, Lions has this to say:

... none of the procedures which call “swtch” directly examines the value returned here.

Only the procedures which call “newproc” are interested in this value, because of the way the child process is first activated!

What is Lions referring to? Justify the above quote.

Problem 5. (12 points):

This problem is based on the following section of code:

```
1743     a = u.u_procp->p_addr;
1744     up = &u.u_uisa[16];
1745     rp = &UISA->r[16];
1746     if(cputype == 40) {
1747         up -= 8;
1748         rp -= 8;
1749     }
1750     while(rp > &UISA->r[0])
1751         *--rp = *--up + a;
```

- What does the variable `a`, defined at line 1743, refer to?
- What is the purpose of the `if` statement starting at line 1746?
- What does the loop at 1750 accomplish? More specifically, what are we populating, and why?