ECE 344: Operating Systems Final Exam
Spring 2001

C. Gibson
S. Anastasiadis

Answer all questions in the space provided in **this** booklet. Please write your name and student number on the top of each page. This is a **closed book** test: no additional reference material or aids are permitted, other than a non-programmable calculator. If you require more space, please continue your answer on the back of the following page and indicate this **clearly** next to the question. You will be given **2.5 hours** to complete all 10 questions (100 marks).

Name: _____

Student Number: _____

Q1 _____ Q2 _____ Q3 _____ Q4 _____ Q5 _____

Q6 _____ Q7 _____ Q8 _____ Q9 _____ Q10 _____

Total _____ / 100

Deadlock [*This question is worth a total of 10 marks.*]

1. **a)** The textbook proposes a deadlock detection algorithm for determining if deadlocks exist. Using that algorithm, you are to determine which of processes P1, P2, P3 and P4 are deadlocked (if any).

   The system has the following Resource Vector (R) and Available Vector (V):

   ```
   R = [ 4, 1, 1, 2, 3 ]   V = [ 0, 0, 0, 0, 3 ]
   ```

   The Request Matrix (Q) and Allocation Matrix (A) for the four processes are:

   ```
   Q = [ 0, 1, 0, 0, 1 ]   A = [ 2, 0, 1, 1, 0 ]
       | 0, 0, 0, 0, 2 |       | 0, 0, 0, 1, 0 |
       [ 0, 0, 1, 0, 1 |       | 2, 1, 0, 0, 0 |
       [ 3, 0, 1, 0, 1 ]       [ 0, 0, 0, 0, 0 ]
   ```

   b) The Banker's Algorithm for resource allocation denial uses a Claim Matrix instead of a Request Matrix. Why does this make the Banker's Algorithm harder, in practice, to implement?

Lab Assignments [*This question is worth a total of 10 marks.*]

2. In Lab 4 you were given a kernel.c file which contained the following definitions for SAVE_CONTEXT and RESTORE_CONTEXT:

```
#define SAVE_CONTEXT() \
  asm("st %%fp,%0" : "=g"(Active->Registers.sp));    /* save sp */ \
  asm("st %%i7,%0" : "=g"(Active->Registers.pc));    /* save pc */

#define RESTORE_CONTEXT() \
  asm("ld %0,%%fp"::"g"(Active->Registers.sp));      /* restore sp */ \
  asm("ld %0,%%i7"::"g"(Active->Registers.pc));      /* restore pc */
```

SAVE_CONTEXT was then used in the following way:

```
void trap (MSGptr request)
{
        /* save context of calling process */
        SAVE_CONTEXT ()

        switch (request->opcode)
        {
                case CREATEPROCESS:
                        ...
        }
        /* schedule new Active process if context switch is required */
        if (Active == NULL)
                if (DispatchProcess () < 0)
        ...
        ...
}
```

As is noted in the code, SAVE_CONTEXT saves the PC (program counter) at the point immediately after trap(...) is called. That value is then later restored in RESTORE_CONTEXT.

Given this method of saving and restoring the PC, it might appear that the trap code is executed forever in a loop. Is this the case? (i.e., Does RESTORE_CONTEXT set the PC to point back to the begining of trap(...) each time?) Justify your answer by describing the values that the various registers take on.

Memory Management [*This question is worth a total of 10 marks.*]

3. Consider a system that provides virtual memory support to its applications. Each process is permitted to use at most three (3) frames of physical memory. We observe that a particular application references the following virtual pages (in order):

   2, 3, 4, 3, 2, 4, 5, 6, 7, 4, 5, 6, 7, 2, 1

   Assuming that primary memory is initially unloaded, how many page faults will the given reference stream incur under the following paging schemes?

   a. Optimal algorithm

   b. LRU

   c. FIFO

   d. Simple Clock Algorithm

More Memory Management [*This question is worth a total of 10 marks.*]

4. Given memory partitions of 100K, 500K, 200K, 300K, and 600K (in order), how would each of the first-fit, best-fit, and next-fit algorithms place processes of 221K, 417K, 112K, and 426K (in order)? Which algorithm makes the most efficient use of memory ?

Virtual Memory [*This question is worth a total of 10 marks.*]

5. Consider a system whose page tables are organized as a three-level tree (i.e., a root table and two levels of sub-tables) and 16kB pages. Each node in the tree contains sixty-four entries (i.e., 64 pointers or 64 page descriptors).

   a. How many address bits are required to encode the page offset?

   b. How many bits are required for each of the three table offsets?

   c. Given a virtual address of 0x00042F35, what are the values for the page offset and the three table offsets?

   d. What is the total amount of virtual memory that can be addressed?

   e. How would this approach differ, both in terms of performance and memory usage, from a system with one large page table?

Scheduling [*This question is worth a total of 10 marks.*]

6. The Shortest Job First processor scheduling algorithm minimizes the average response time. Prove this for a batch of n jobs which arrive at the same time with known service times:

   ```
   t1 <= t2 <= t3 <= ... < tn
   ```

   Assume that no other jobs are subsequently queued, and that the system has no preemption (e.g., jobs run to completion once started).

Scheduling [*This question is worth a total of 10 marks.*]

7. Describe briefly the scheduling algorithm used in traditional Unix systems. Explain how interactive jobs are treated relatively to background jobs. Justify your answer based on the mechanism used for determining process priorities.

I/O Management [*This question is worth a total of 10 marks.*]

8. Consider a disk drive with 5000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 143, and the previous request was at cylinder 125. The queue of pending requests, in FIFO order, is:

   ```
   86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130
   ```

   Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests, for each of the following disk-scheduling algorithms:

   a. FCFS

   b. SSTF

   c. SCAN

   d. C-SCAN

File Systems [*This question is worth a total of 10 marks.*]

9. Consider the organization of a Unix file as represented by the inode. Assume that there are 12 direct block pointers and a singly, doubly, and triply indirect pointer in each inode. Further, assume that the system block size and the disk sector size are both 8kB. If the disk block pointer is 32 bits, with 8 bits to identify the physical disk and 24 bits to identify the physical block, then:

   a. What is the maximum file size supported by this system?

   b. What is the maximum file system partition supported by this system?

   c. Assuming that only the file's inode is already in main memory, how many disk accesses are required to read byte number 13,423,956 from the file?

   To simplify the calcuations, assume that 1kB = 2^10 bytes, 1MB = 2^20 bytes, etc..

Device Drivers [*This question is worth a total of 10 marks.*]

10. **a)** Consider a device that generates large quantities of data. Its existing driver, written using Programmed I/O, is replaced with a new device driver that uses DMA. Will this increase the load on the CPU? Justify your answer.

b) Why does double-buffering provide benefit over single buffering?

c) Explain the concept of circular I/O buffers. How does this improve on double-buffering?