Computer Science Department
Stanford University
Comprehensive Examination in Software Systems
Autumn 1994

October 20, 1994

READ THIS FIRST!

1. You should write your answers for this part of the Comprehensive Examination in a BLUE BOOK. Be sure to write your MAGIC NUMBER on the cover of every blue book that you use.
2) The number of POINTS for each problem indicates how elaborate an answer is expected. For example, an essay-type question worth 6 points or less doesn't deserve an extremely detailed answer, even though a person can expound at length on just about any topic in computer science.
3) The total number of points is 60, and the exam takes 60 minutes. This "coincidence" can help you plan your time.
4) This exam is CLOSED BOOK. You may NOT use notes, articles, books, computer, etc.
5) Show your work, since PARTIAL CREDIT will be given for incomplete answers. For example, you can get credit for making a reasonable start on a problem even if the idea doesn't work out; your can also get credit for realizing that certain approaches are incorrect.
6) If you are convinced you need to make an assumption to answer a question, state your assumption(s) as well as the answer.
7) Be sure to provide justification for your answers.

Comprehensive Exam: Software Systems (60 points)

1) (16 points) This question asks you to implement a barrier synchronization function using only semaphores and a small number of shared variables. A barrier synchronization function waits until the specified number of processes arrive at the "barrier" before allowing any of the processes to continue. For example assume N processes execute the following code fragment:

```
Func1();
barrier(N);
Func2();
barrier(N);
Func3();
```

The barrier should ensure that no process starts executing `Func2()` before all N of the processes have executed `Func1()`. Similarly, by the time that the first process calls `Func3()` all processes should have returned from `Func2()`. You function should take a single argument, N, the number of processes participating in the barrier. It should also work correctly on the above code fragment and contain no busy waiting.

2) (8 points) Some computer systems have been designed recently with a larger physical memory address space than they have virtual address space. In other words they have more fewer bits of virtual address space than physical memory address space. Explain why this is not a totally unreasonable design for a computer system. Be sure to indicate what limits the design imposes.

3) (10 points) As the price of DRAM memory has improved relative to that of magnetic disk space, the ratio of the amount of physical memory to the amount of backing store has been has been getting larger. Some system have as much or more physical memory than backing store (swap space). In response to this, some virtual memory systems have been modified to allocate backing store in a different way. Rather than allocating the backing store when a virtual page is first created, the backing store is allocated only when the page is first paged out.

   a) Describe the benefits of this change.

   b) Describe the problems introduced by it.

4) (8 points) In some computer systems the maximum size of a transfer to or from an I/O device is limited to a relatively small size (e.g. 32 kilobytes) due to historical artifacts. On these systems, would there be any advantages of using a file system with a block size larger than the maximum I/O transfer size?

5) (8 points) What is the difference between starvation and deadlock?

6) (10 points) Frequently when sending data over a network it is beneficial to both encrypt the data for security and compress the data to decrease the transfer time. Which order would you suggest these operations be performed? Be sure to justify your answer.