Innopolis University, Academic Year 2017/2018
Second Retake of Operating Systems – 19th January 2018
Max 20 points + 6 points (~30%) extra credit – Total time: 90 minutes

1. (4 points) A 255-GB disk has 65,536 cylinders with 255 sectors per track and 512 bytes per sector. How many platters and heads does this disk have? Assuming an average cylinder seek time of 11 ms, average rotational delay of 7 msec and reading rate of 100 MB/sec, calculate the average time it will take to read 400 KB from one sector.
*Number of heads = 255 GB / (65536*255*512) = 16*
Number of platters = 16/2 = 8
The time for a read operation to complete is seek time + rotational latency +transfer time. The seek time is 11 ms, the rotational latency is 7 ms and the transfer time is 4 ms, so the average transfer takes 22 msec.
2. (3 points) When a user program makes a system call to read or write a disk file, it provides an indication of which file it wants, a pointer to the data buffer, and the count. Control is then transferred to the operating system, which calls the appropriate driver. Suppose that the driver starts the disk and terminates until an interrupt occurs. In the case of reading from the disk, obviously the caller will have to be blocked (because there are no data for it). What about the case of writing to the disk? Need the caller be blocked awaiting completion of the disk transfer?
Maybe. If the caller gets control back and immediately overwrites the data, when the write finally occurs, the wrong data will be written. However, if the driver first copies the data to a private buffer before returning, then the caller can be allowed to continue immediately. Another possibility is to allow the caller to continue and give it a signal when the buffer may be reused, but this is tricky and error prone.
3. (4 points) Consider a system in which threads are implemented entirely in user space, with the run-time system getting a clock interrupt once a second. Suppose that a clock interrupt occurs while some thread is executing in the run-time system. What problem might occur? Can you suggest a way to solve it?
It could happen that the runtime system is precisely at the point of blocking or unblocking a thread, and is busy manipulating the scheduling queues. This would be a very inopportune moment for the clock interrupt handler to begin inspecting those queues to see if it was time to do thread switching, since they might be in an inconsistent state. One solution is to set a flag when the runtime system is entered. The clock handler would see this and set its own flag, then return. When the runtime system finished, it would check the clock flag, see that a clock interrupt occurred, and now run the clock handler.
4. (3 points) Synchronization within monitors uses condition variables and two special operations, wait and signal . A more general form of synchronization would be to have a single primitive, waituntil , that had an arbitrary Boolean predicate as parameter. Thus, one could say, for example,

```
waituntil x  < 0 or y  + z  < n
```

The signal primitive would no longer be needed. This scheme is clearly more general than that of Hoare or Brinch Hansen, but it is not used. Why not? (Hint : Think about the implementation.)
It is very expensive to implement. Each time any variable that appears in a predicate on which some process is waiting changes, the run-time system must re-evaluate the predicate to see if the process can be unblocked. With the Hoare and Brinch Hansen monitors, processes can only be awakened on a signal primitive.

5. (3 points) You are given the following data about a virtual memory system:
   1. The TLB can hold 1024 entries and can be accessed in 1 clock cycle (1 nsec).
   2. A page table entry can be found in 100 clock cycles or 100 nsec.
   3. The average page replacement time is 6 msec.

If page references are handled by the TLB 99% of the time, and only 0.01% lead to a page fault, what is the effective address-translation time?

The chance of a hit is 0.99 for the TLB, 0.0099 for the page table, and 0.0001 for a page fault (i.e., only 1 in 10,000 references will cause a page fault). The effective address translation time in nsec is then:

$$0.99 \times 1 + 0.0099 \times 100 + 0.0001 \times 6 \times 106 \approx 602 \text{ clock cycles.}$$

Note that the effective address translation time is quite high because it is dominated by the page replacement time even when page faults only occur once in 10,000 references..

6. (3 points) Free disk space can be kept track of using a free list or a bitmap. Disk addresses require D bits. For a disk with B blocks, F of which are free, state the condition under which the free list uses less space than the bitmap. For D having the value 16 bits, express your answer as a percentage of the disk space that must be free.

The bitmap requires B bits. The free list requires DF bits. The free list requires fewer bits if DF < B. Alternatively, the free list is shorter if F/B < 1/D, where F/B is the fraction of blocks free. For 16-bit disk addresses, the free list is shorter if 6% or less of the disk is free.

7. (30% extra credit) Consider a magnetic disk consisting of 16 heads and 400 cylinders. This disk has four 100-cylinder zones with the cylinders in different zones containing 160, 200, 240. And 280 sectors, respectively. Assume that each sector contains 512 bytes, average seek time between adjacent cylinders is 1 msec, and the disk rotates at 7200 RPM. Calculate the (a) disk capacity, (b) optimal track skew, and (c) maximum data transfer rate.

Consider,

(a) The capacity of a zone is tracks × cylinders × sectors/cylinder × bytes/sect.

Capacity of zone 1: 16 × 100 × 160 × 512 = 131072000 bytes
Capacity of zone 2: 16 × 100 × 200 × 512 = 163840000 bytes
Capacity of zone 3: 16 × 100 × 240 × 512 = 196608000 bytes
Capacity of zone 4: 16 × 100 × 280 × 512 = 229376000 bytes
Sum = 131072000 + 163840000 + 196608000 + 229376000 = 720896000

(b) A rotation rate of 7200 means there are 120 rotations/sec. In the 1 msec track-to-track seek time, 0.120 of the sectors are covered. In zone 1, the disk head will pass over 0.120 × 160 sectors in 1 msec, so, optimal track skew for zone 1 is 19.2 sectors. In zone 2, the disk head will pass over 0.120 × 200 sectors in 1 msec, so, optimal track skew for zone 2 is 24 sectors. In zone 3, the disk head will pass over 0.120 × 240 sectors in 1 msec, so, optimal track skew for zone 3 is 28.8 sectors. In zone 4, the disk head will pass over 0.120 × 280 sectors in 1 msec, so, optimal track skew for zone 3 is 33.6 sectors.

(c) The maximum data transfer rate will be when the cylinders in the outermost zone (zone 4) are being read/written. In that zone, in one second, 280 sectors are read 120 times. Thus the data rate is 280 × 120 × 512 = 17,203,200 bytes/sec.

This is the end of the questions and solutions of the exam.