

Assignment 2, Due Wed. 25th November 2009, 10.30am, room 2205. Worth 33.3% of CA. Note: demonstrations for Q.1 and Q.5 must be completed on or before Tues. 24th November 2009, 12.30pm.

Answers should be in your own words. For numerical answers, where appropriate, show your working; a numerical answer written down with no explanation of working is likely to get zero marks.

Late assignments. 10% of marks will be deducted for each day or part of day late; assignments seven days late will receive 0%.

Computing Department Rules: 1. Assignments must be neatly typed and printed; 2. A cover sheet and declaration must be included.

1. Give a quick demonstration to show that the CMD command-line practical has been completed.

[20 marks]

2. (a) Figure 1 shows a magnetic disk with four platters and and Figure 2 shows a possible flow of control and data through various layers of the operating system and hardware.

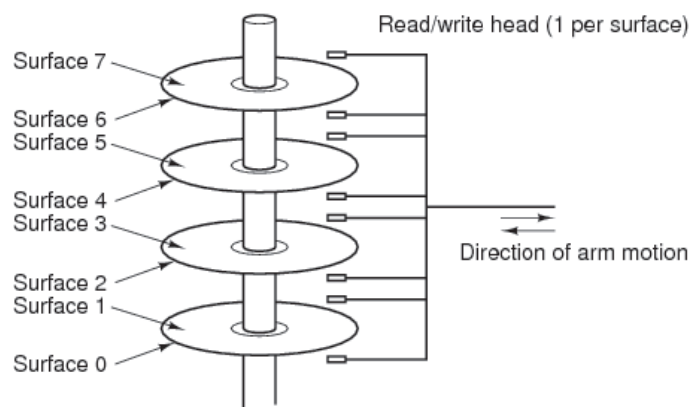


Figure 1: A disk with four platters

- (i) Explain the terms *head*, *cylinder*, *sector*. [15 marks]
- (ii) Explain *linear sector number*. [5 marks]
- (iii) Explain *allocation unit*. [5 marks]
- (v) Give one advantage and one disadvantage, with explanation, of choosing an allocation unit size of a large number of sectors, e.g. 64. [10 marks]

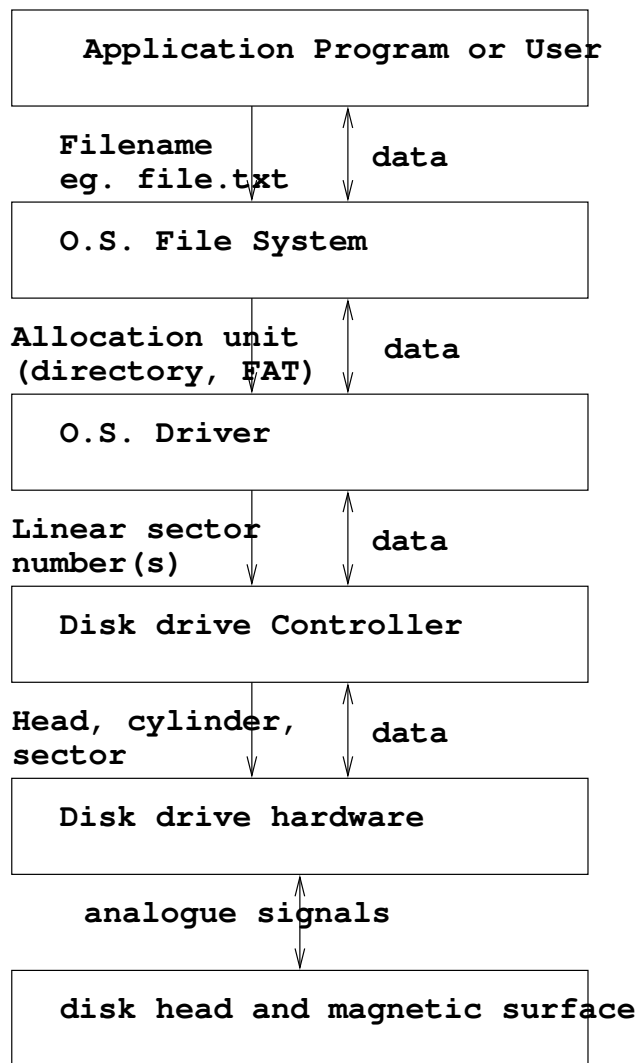


Figure 2: Possible flow of control and data following a file read/write request from an application program or user.

- (b) A hard disk has the following characteristics. Rotation rate: 10,000 revs per minute. Number of sectors per track (assume fixed): 1000. Number of platters **one**. Number of heads **two**. Number of cylinders 10,000. Average seek time — i.e. time for head servomechanism to wake up and to move to the desired cylinder: 8 milliseconds; time to move between adjacent cylinders: 1 millisecond. Sectors contain 512 bytes.
- (i) What is the capacity of the disk in bytes? [10 marks]
- (ii) Compute the maximum rotational latency. [10 marks]
- (iii) Compute the transfer latency for one sector. [10 marks]
- (iv) Compute the worst case time to read one sector. [10 marks]
- (v) Assume that the file system uses allocation units of **eight** sectors and that allocation units will always be stored on contiguous sectors and that eight contiguous sectors can be read at the cost of one seek time (latency) plus one rotational latency plus eight (sector) transfer latencies. Based on these assumptions compute the worst case time to read an allocation unit. [10 marks]
- (vi) Assume now that we have a file that occupies *three* allocation units (a total of 24 sectors). Assume that the allocation units are all on the same track, but they are not contiguous. As in (v) assume that an allocation unit (eight contiguous sectors) can be read at the cost of one seek time (latency) plus one rotational latency plus eight (sector) transfer latencies. Compute the worst case time to read the three allocation units. Hint: one seek time, three rotational latencies and three ($\times 8$) transfer latencies. [10 marks]
- (vii) Recompute the answer for (vi) assuming now that the three allocation units are on separate tracks/cylinders. Hint: three seek times [10 marks]
- (viii) Based on your answers to (vi) and (vii) explain why *fragmentation* is a *bad thing*. [10 marks]
- (ix) If you had a *read only* disk, e.g. in a web server, why would *contiguous allocation* be a good idea to speed up access. [10 marks]
3. (a) In the context of multitasking operating systems, explain the term *shell*. [5 marks]
- (b) How does typing a command in response to a shell prompt, e.g. `xyz.EXE` differ from clicking on an icon corresponding to a program `xyz.EXE`? [5 marks]
- (c) A process *state transition diagram* is shown in Figure 3. Give a brief explanation of each of the three states with a brief explanation of typical circumstances that would cause a process to make each of the transitions 1, 2, 3, and 4. [20 marks]
- (e) What do you think is happening at transitions a and b? [5 marks]
- (f) The *round-robin* scheduling algorithm is used on many interactive multitasking systems: the scheduler cycles around each *ready* process, giving the current process a quantum of time (or until it becomes *blocked* or terminates), then preempting it and giving the CPU to the next process in the queue. An operating system, which uses round-robin scheduling, is, for example, managing two processes (processes A, B). The round robin quantum (time slot size) is set such that the CPU cycles between processes as follows:
- process A gets time for 100 instructions (1 microsecond);

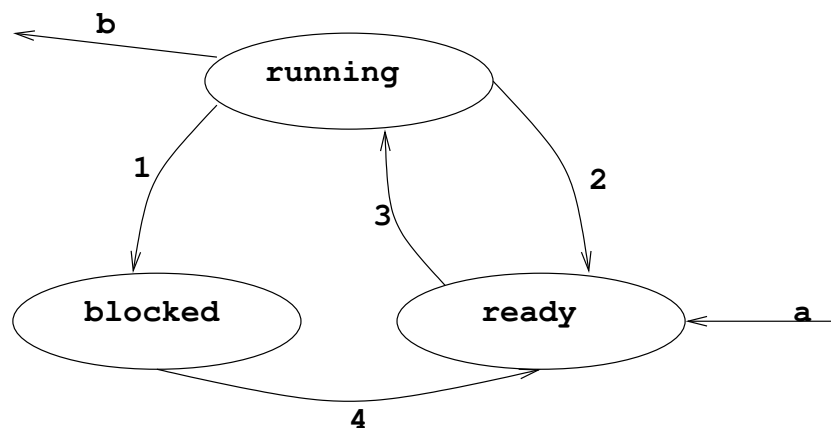


Figure 3: Process States

- interrupt handler and kernel (context switching) runs for 200 instructions (2 microseconds);
- process B gets time for 100 instructions (1 microsecond);
- interrupt handler and kernel (context switching) runs for 200 instructions (2 microseconds);
- cycle starts again;

Users of the system complain that it is slow. From the information given above explain why it may appear slow and criticise the choice of quantum; suggest, with explanation, an alternative size of quantum, taking into account that the system is interactive.

[20 marks]

4. (a) When multiple processes occupy memory, a big *challenge is to stop processes interfering with one another's memory area and with the OS's memory area*. Explain, using an example, how the OS, using some special hardware in the CPU, can *police* memory access.

[10 marks]

- (b) If the OS is unable to disallow application processes from accessing (any) memory cell at will, explain one easy method by which a process could *gain complete control* of the CPU and system.

[10 marks]

5. Interacting TCP client and server — Java programs from Chapter 16. Demonstration worth 100% of marks.

[20 marks]

- Form groups of two. One person is going to compile and execute the TCP client, the other the TCP server.
- Agree on a port number and modify the code in the two programs accordingly.
- The client program will need to hard-wire the IP address of the server.
- Compile and execute both programs. Show the client sending a message and receiving a reply from the server.