Department of Electrical and Computer Engineering

Computer Engineering

Course ECSE-322B

Solutions to Problem Set 8

March 9, 2008

1.Consider a file occupying sequential sectors over four consecutive tracks of a floppy disk. The floppy disk has a track-to-track time of 3 ms and a head settling time of 17 ms. The diskette is rotated at 300 rpm. Determine how long it takes the drive to read the entire file after positioning and settling of the head over the very first sector.

Solution:
The rotational period is 60/300 = 0.2 seconds. Reading of the first track takes 200 milliseconds. Then, the disk moves the head to the next track. It takes 3 milliseconds to step to it plus another 17 milliseconds to settle -- that is, a total of 3+17 = 20 milliseconds. However, by now the first sector of the new track has passed under the head. Consequently, we have to wait for 200-20 = 180 milliseconds until we can continue reading of the file. In other words, (after settling) the head spends on the seconds track a total of 180 + 200 = 380 milliseconds. The same process is repeated for the third and fourth tracks. Hence, the total time amounts to

*4*200 + 3*20 + 3*180 = 1400 milliseconds, or 1.4 seconds.*

2. Consider a disk drive like the one discussed in question 1. The employed diskettes use a format based on 16 sectors. A particular file occupies one full track and two consecutive sectors of the adjacent track.

(a) Calculate how long it takes to read the entire file after positioning and settling of the head over the first sector of the file.
(b) How would you go about minimizing the file access time?

Solution:
*(a) The first track takes 200 milliseconds. Then, the head takes 20 milliseconds to step and settle on the adjacent track. There, it experiences a latency of 180 milliseconds. Since the disk has 16 sectors per track, it takes about (2/16)*200 = 25 milliseconds to read the two remaining sectors. Hence the total time is 200 + 20 + 180 + 25 = 425 milliseconds*

(b) Reading of a sector takes about 200/16 = 12.5 milliseconds. So, if (when we were writing the file) we had skipped two sectors and had continued recording on sectors 3 and 4 of the second track, we would eliminate the latency time. The total time would then be 425-180 = 245 milliseconds. This, of course, assumes that the system allows us to specify somehow the particular sector we want to write data in.

3. A two-sided diskette rotates at 360 rpm. Once the desired sector is located, data are provided/accepted at a rate of 500Kb/s. Determine its storage capacity in kilobytes given a total of 135 tracks per surface.

Solution:

The rotational period is 60/360 = 0.167 s. Then the capacity of a track is:

*0.167 sec/track * 500 Kb/sec = 83.5 Kb/track.*

The capacity of the disk is therefore:

*2 sides/diskette * 135 tracks/side * 83.5 Kb/track = 22.5 Megabits; that is about 2.8 Mbytes.*

4. Derive an expression for the data rate R of a magnetic hard disk given that r is the radius of the innermost track, d is the recording density on the innermost track and L is the time taken for a full rotation of the disk.

Solution:

*The length of the innermost track is 2*pi*r. At a density of d bits per unit of length, we can store (2*pi*r)*d bits. Since all this information can be read in a single rotation, the equivalent data rate is (2*pi*r*d) / L. Although the remaining tracks are longer, they have the same capacity (for convenience). In other words, recording densities are reduced as we move towards the outermost track. As a result, the data rate is the same for the rest of the tracks.*

5. Derive an expression for the recording density of track i in terms of the radius, r, and recording density, d, of the innermost track. Assume that the magnetic disk has a track density of D tracks per inch.

Solution:

*Assume for convenience that we designate the innermost track as track 0, the next one as track 1, and so on. Since there are D tracks per inch, the distance between two consecutive tracks is 1/D. Hence the distance of track i from the innermost one is i/D. This means that track i has a radius of r+i/D and thereby a capacity of 2*pi*(r+i/D)di, where di represents the recording density of track i. However, track i has a capacity equal to that of the innermost track -- that is:*

*2*pi*(r+i/D)*di = 2*pi*r*d.*

From the latter we obtain:

di =[r/(r+i/D)}d

6. A floppy disk rotates the diskette at 360 rpm and has a track-to-track time of 3ms. The employed diskettes have a total of 77 tracks per surface.

(a) Determine the minimum, maximum, and average values of the latency time

Solution:
The rotational period is 60/360 = 0.167 s. The minimum, maximum and average values of the latency time are therefore:

Minimum: 0 (found it right away)
Maximum: 167 ms (had to wait a full rotation)
Average: 167/2 = 83.5 ms (had to wait ½ a rotation).

(b) Repeat part (a) for the seek time.

Solution:
The minimum, maximum, and average values of the seek time are:
Minimum: 0,
*Maximum: (76*3) = 228 ms*
*Average: (76*3)/2 = 114 ms (assuming the head always starts from track 0 - if it starts from a random track, the seek time is (77*3)/3 = 77 ms. Problem set 8 will have a proof of this.).*

7. A disk drive has 300 tracks, formatted into 32 sectors containing 512 data bytes each. What is the total storage capacity of this disk?

During data transfer operations, the disk controller cannot access adjacent sectors on a given track. After reading one sector, it must skip the next sector, using that time for error checking and for data transfer to or from the host. Assuming that the disk is completely empty, show how a 63K-byte file may be stored to minimize transfer time.

Solution:
*Each track contains 32 sectors of 512 bytes. Hence each track contains 32*512 bytes = 16 KBytes. There are 300 tracks so the total amount of data is 300*16 KBytes = 4.8 Mbytes.*

A 63 Kbyte file will occupy 3 full tracks +15/16 tracks (i.e. 16Kbytes/track). The file will be stored in alternate sectors on the first track, i.e. it will be stored in sectors 0,2,4,... then, on the next revolution it will be stored in sectors 1,3,5.. of the same track. The next part of the file will be stored on the next track but staggered from the first track to allow for the amount of disk rotation during a head move. This is repeated for the whole file.

8. An I/O device is capable of accepting one byte of data every 100 microseconds. The data are transferred to the device in blocks of 256 bytes. Initially we consider designing the system such

that the device will cause an interrupt whenever it is ready to accept another byte of data. It is estimated that the servicing of each interrupt will take a total of about 40 microseconds.

(a) What will be the improvement ratio (in terms of CPU time) if we employ DMA in the cycle stealing mode and interrupt the CPU only after transferring the whole block? Assume a total of 2 microseconds to gain the bus, transfer a byte, and return bus control to the CPU. Also assume that the CPU does no productive work while its bus is tied up.

(b) Suppose that such 256-byte blocks are transferred to the device at an average rate of one every 2 seconds. What fraction of CPU time is consumed for device service when employing interrupt-driven I/O?

(c) Assume blocks are transferred at a rate of 20 per second. What fraction of CPU time will we economize if we employ DMA?

Solution:

(a) *With interrupt-driven I/O a block transfer takes:*

256 bytes x 0.04 ms/byte = 10.24 milliseconds of CPU time.

With DMA it takes only:

256 bytes x 0.002 ms/bytes + 0.04 ms = 0.552 milliseconds.

The improvement ratio is 10.24 / 0.552, i.e. about 18.

(b) *Transfer of a block takes:*

256 bytes x 40 x 10^{-6}*secs/byte = 0.01 seconds.*

In other words, they take only 0.01 / 2 = 0.5% of the CPU time.

(c) *This rate is 40 times higher than the one in part (b). So it takes:*

40 x 0.5% = 20% of the CPU's time.

Alternately, 20 block/sec x 10.24 ms/block = 204.8 ms/sec = 0.2sec/sec (approx) or 20%

With DMA, block transfers are 18 times faster. Therefore, it takes only:

20% / 18 = 1.1%.

Alternately, 20 block/sec x 0.552 ms/block = 11.04 ms/sec = 0.011sec/sec = 1.1%

Hence, we economize:

20 - 1.1 = 18.9% of the CPU's time.