**IO Review**

1. What is memory mapped I/O?

Memory mapped I/O is an interfacing technique in which memory related instructions are used for data transfer and the device is identified by a 16-bit address. In this type, the I/O devices are treated as memory locations. The control signals used are MEMR and MEMW. The interfacing between I/O and microprocessor will be same as single memory location. For data transfer between I/O device and microprocessor, microprocessor will send address, generate control signals MEMR and MEMW.

1. Why is DMA an improvement over CPU programmed I/O?

DMA (Direct Memory Access) controllers can typically move data from and to memory with either I/O or memory. Since it operates independently of the CPU, the CPU does not need to manage the process with each transfer.

If you are transferring multiple pieces of data, it can be faster using the DMA controller. For I/O, it might be a long time between each transfer.

Typically, the CPU sets up the transfer in the DMA controller, starts it, and then does other things while it waits for an interrupt from the DMA controller telling the CPU that it completed the task.If you were to use the CPU instead, you would probably have an interrupt (meaning you have to save the current program context, service the interrupt transferring the next piece of data, and then restore the program context). Using the DMA controller would be faster.

1. When would DMA transfer be a poor choice?

DMA is not useful when the amount of data to transferred between memory and the I/O device is very less. In this case the overhead of setting up the DMA transfer would outweigh the benefits of direct data transfer without the interference of the processor.

1. Disk Technology. Suppose we have a magnetic disk (resembling an IBM Microdrive) with the following parameters:

Average seek time 12 ms

Rotation rate 3600 RPM

Transfer rate 3.5 MB/second

Sectors per track 64

Sector size 512 bytes

Controller overhead 5.5 ms

Answer the following questions. (Note: you may leave any answer as a fraction.)

(a) What is the average time to read a single sector?

Disk Access Time = seek time + rotational delay + transfer time + controller overhead

rotational delay = rotation rate / 2

Rotation rate = 3600, thus time for one rotation

= 1min \* 60 seconds / 3600 = 0.016second = 16.66ms per rotation

Rotational delay for one disk = 1/ 2 \* 16.66ms = 8.33 = 30.000/RPM = (1 / (RPM / 60)) \* 0.5 \* 1000

Transfer time = (512 / 3.5\*2^20)\*1000 = x ms

= 12 + 8.33 + 0.14 + 5.5 = 25.97ms

b) What is the average time to read 8 KB in 16 consecutive sectors in the same cylinder?

Transfer time = (8 \* 1024 / 3.5\*2^20)\*1000 = x ms

= 12 + 8.33 + 2.24 + 5.5 = 28.07ms

c) Now suppose we have an array of 4 of these disks. They are all synchronized such that the arms on all the disks are always on the same sector within the track. The data is striped across the 4 disks so that 4 logically consecutive sectors can be read in parallel. What is the average time to read 32 consecutive KB from the disk array?

4 disks -> each disk can read a sector at a time, total memory can read at a time = 4 \* 512 (size of sector) = 2KB.

To read 32KB in 4 disks, need to read 8k in each disk 8k = 8 \* 1024 / 512 = 16 sectors

And all disks read at the same time -> answer is same as b

5. What is the average time to read or write a 512-byte sector for a typical disk rotating at 7200 RPM? The advertised average seek time is 8ms, the transfer rate is 20MB/sec, and the controller overhead is 2ms. Assume that the disk is idle so that there is no waiting time.

Access Time = seek time + rotational delay + transfer time + controller overhead

= 8 + (0.5\*60\*1000/7200) + (512/20\*220)\*1000 + 2 = 14.17 ms

6. A program repeatedly performs a three-step process: It reads in a 4-KB block of data from disk, does some processing on that data, and then writes out the result as another 4-KB block elsewhere on the disk. Each block is contiguous and randomly located on a single track on the disk. The disk drive rotates at 7200RPM, has an average seek time of 8ms, and has a transfer rate of 20MB/sec. The controller overhead is 2ms. No other program is using the disk or processor, and there is no overlapping of disk operation with processing. The processing step takes 20 million clock cycles, and the clock rate is 400MHz. What is the overall speed of the system in blocks processed per second assuming no other overhead?

Disk Read Time for a 4KB block

= seek time + rotational delay + tr

ansfer time + controller overhead

= 8 + (0.5\*60\*1000/7200) + (4\*1024/20\*)\*1000 + 2

= 14.17 ms

Processing Time = 20 \* \* (1/(400\* ))

= 1/20 = 0.05 s = 50 ms

Disk Write Time for 4 KB block = 14.17 ms

Total time to completely process a 4 KB block = 2\*14.17 + 50 = 78.34 ms

Number of blocks proce ssed per second = 1000/78.34 = 12.76

7. How much cylinder skew is needed for a 7200-rpm disk with a track-to-track seek time of 1 msec? The disk has 200 sectors of 512 bytes each on each track.

The disk rotates at 120 RPS, so 1 rotation takes 1000/120 msec. With 200 sectors

per rotation, the sector time is 1/200 of this number or 5/120 = 1/24 msec.

During the 1-msec seek, 24 sectors pass under the head. Thus the cylinder

skew should be 24.

8. Disk requests come in to the disk driver for cylinders 10, 22, 20, 2, 40, 6, and 38, in that order. A seek takes 6 msec per cylinder moved. How much seek time is needed for

1. (a) First-Come, first served.
2. (b) Closest cylinder next.
3. (c) Elevator algorithm (initially moving upward).

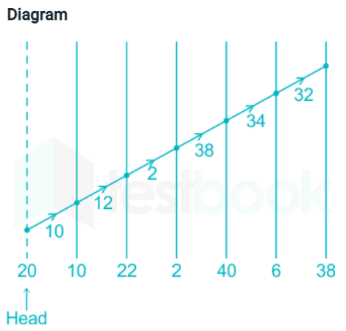
In all cases, the arm is initially at cylinder 20.

8. Disk requests come in to the disk driver for cylinders 10, 22, 20, 2, 40, 6, and 38, in that order. A seek takes 6 msec per cylinder moved. How much seek time is needed for

1. (a) First-Come, first served.

Since, it is a first-come-first-serve scheduling requests will be served in the given sequence :

10, 22, 20, 2, 40, 6 and 38

Total head movement = (20 - 10) + (22-10) + (22-2) + (40 - 2) + (40 - 6) + (38-6) = 10 + 12 +20 + 38 +34 +32 146

Total Seek time = 146 x 6 = 876 ms.

1. (b) Closest cylinder next.

20→20→22→10→6→2→38→40

Distance=0+2+12+4+4+36+2=60 cylinders

Total Seek time =60×6=360msec

1. (c) Elevator algorithm (initially moving upward).

The order for Elevator(SCAN):

20→20→22→38→40→10→6→2

Distance: =0+2+16+2+30+4+4=58cylinders

Total Seek time =58×6=348 msec