Worked out Examples

5.1 Polling for an I/O completion can waste a large number of CPU cycles if the processor iterates a busy-waiting loop many times before the I/O completes. But if the I/O device is ready for service, polling can be much more efficient than is catching and dispatching an interrupt. Describe a hybrid strategy that combines polling, sleeping, and interrupts for I/O device service. For each of these three strategies (pure polling, pure interrupts, hybrid), describe a computing environment in which that strategy is more efficient than is either of the others.

Ans: A hybrid approach could switch between polling and interrupts depending on the length of the I/O operation wait. For example, we could poll and loop N times, and if the device is still busy at N+1, we could set an interrupt and sleep. This approach would avoid long busy-waiting cycles. This method would be best for very long or very short busy times. It would be inefficient it the I/O completes at N+T (Where T is a small number of cycles) due to the overhead of polling plus setting up and catching interrupts. Pure polling is best with very short wait times. Interrupts are best with known long wait times.

5.2 Disk with geometrics exceeding the following maximums could not be handled by early Dos systems :

Cylinders	**1024**
Heads	**16**
Sectors per track	**63**

What is the maximum size disk these systems could use?

Ans: 1024 * 16 * 63 * 512 = 528,482,304 = 582 Mb

5.3 State three advantages of placing functionality in a device controller, rather than in the kernel. State three disadvantages.

Ans:

Three advantages:-

a. Bugs are less likely to cause an operating system crash.
b. Performance can be improved by utilizing dedicated hardware and hard-coded algorithms.

c. The kernel is simplified by moving algorithms out of it.

Three disadvantages:

a. Bugs are harder to fix - a new firmware version or new hardware is needed
b. Improving algorithms likewise require a hardware update rather than just kernel or device driver update
c. Embedded algorithms could conflict with application's use of the device, causing decreased performance.

5.4 For what types of operations is DMA useful?

a. For large & fast data transfers between memory & I/O devices.
b. For large & slow data transfers between memory & I/O devices.
c. For slow & small data transfers between memory & I/O devices.
d. For small data transfers between memory & cache.

Ans: a