Operating System A7

Zhiyuan Zhong 12110517

Q1

1.

- Pros: Polling is simple and good fro I/O device that is fast. Interrupt-based I/O allows for overlap of computation and I/O, which can improve utilization(for both the CPU and the disk). It's suitable for slow I/O device.
- Cons: Polling is inefficient, it wastes a great deal of CPU time just waiting for the device to complete its activity, instead of switching to another ready process and thus better utilizing the CPU. Interrupt is not a good solution if the device performs its tasks very quickly, this will slow down the system because context-switch is expensive. And in network systems, when a huge stream of incoming packets each generate an interrupt, it is possible for the OS to livelock, that is, find itself only processing interrupts and never allowing a user-level process to run and actually service the requests.

2.

- PIO (Programmed I/O): When the main CPU is involved with the data movement. When using PIO to transfer a large chunk of data to a device, the CPU is once again over-burdened with a rather trivial task, and thus wastes a lot of time and effort that could better be spent running other processes. The CPU spends too much time moving data to and from devices by hand.
- DMA (Direct Memory Access): A DMA engine is essentially a very specific device within a system that can orchestrate transfers between devices and main memory without much CPU intervention. To transfer data to the device, the OS would program the DMA engine by telling it where the data lives in memory, how much data to copy, and which device to send it to. At that point, the OS is done with the transfer and can proceed with other work. When the DMA is complete, the DMA controller raises an interrupt, and the OS thus knows the transfer is complete. Now the copying of data is handled by the DMA controller.

3. For memory-mapped I/O, in order to access a particular register, the OS issues a load (to read) or store (to write) the address; the hardware then routes the load/store to the device instead of main memory. So that malicious process can't access certain I/O related memory. For explicit I/O instructions, the OS controls devices, and the OS thus is the only entity allowed to directly communicate with them.

Q2

1. READ/WRITE data time = Seek time + Rotational Latency + Transfer Time

2. (a)

Algorithm	**access sequence**
FIFO	(100), 70, 20, 90, 110, 60, 20

Algorithm	access sequence
SSTF	(100), 90, 70, 60, 20, 20, 110
SSTF(alternative)	(100), 110, 90, 70, 60, 20, 20
SCAN	(100), 110, (199), 90, 70, 60, 20, 20
CSCAN	(100), 110, (199), (0), 20, 20, 60, 70, 90

(b) 12000rev/min = 60000 / 12000 = 5ms/rev, average rotation delay = 5/2 = 2.5ms Assume transfer time is t, then:

- FIFO: |100-70| + |70-20| + |20-90| + |90-110| + |110-60| + |60-20| + 2.5*6 + t = 275 + t
- SSTF: |100-90| + |90-70| + |70-60| + |60-20| + |20-20| + |20-110| + 2.5*6 + t = 185 + t
- SSTF(alternative): |100-110| + |110-90| + |90-70| + |70-60| + |60-20| + |20-20| + 2.5*6 + t = 115 + t
- SCAN: |100-110| + |110-199| + |199-90| + |90-70| + |70-60| + |60-20| + |20-20| + 2.5*6 + t = 293 + t
- CSCAN: |100-110| + |110-199| + 199 + |0-20| + |20-20| + |20-60| + |60-70| + |70-90| + 2.5*6 + t = 403 + t