Virtual Memory

1. Consider a processor that uses segmentation and paging (i.e., this is not a MIPS processor). Below is the segment table being used for the currently executing process and below that are pages tables for several processes in the system. Note that some processes may not use all of the available segments. Recall that VPN is the virtual page number, PFN is the physical frame number, V is the valid bit and D is the dirty bit (i.e., the page can be dirtied/modified).

Segment	PT base addr	Max VPN Value
4	70700000	3
3	70200000	3
2	70500000	3
1	70300000	3
0	70100000	3

VPN	PFN	V	D
3	5177	1	0
2	20	0	0
1	77	1	0
0	4251	0	0

Base addr: 70000000

VPN	PFN	V	D
3	1311	1	0
2	12	1	0
1	711	0	0
0	23	1	0

Base addr: 70700000

VPN	PFN	V	D
3	52	1	1
2	41	1	1
1	30	1	1
0	5177	0	1

Base addr: 70400000

VPN	PFN	V	D
3	65	1	1
2	77	1	1
1	567	1	1
0	672	1	1

Base addr: 70300000

VPN	PFN	V	D
3	641	0	1
2	753	1	1
1	2577	1	1
0	517	1	1

Base addr: 70200000

VPN	PFN	V	D
3	5532	0	1
2	5177	1	1
1	336	0	1
0	77	1	1

Base addr: 70600000

VPN	PFN	V	D
3	5177	1	1
2	34	1	1
1	563	1	1
0	1641	1	1

Base addr: 70500000

VPN	PFN	V	D
3	516	1	1
2	37	0	1
1	7731	1	1
0	6341	1	1

Base addr: 70100000

Assume that the processor is using 32-bits for virtual and physical addresses, that the page size is 64 KB, that all addresses (virtual and physical) and values shown in the segment table and page tables are expressed in hexadecimal, and that the system uses 4 bits for segments.

a. Explain how many bits of the virtual address will be used to represent the offset?
b. What is the maximum possible size of a segment in this system in bytes (expressed as an equation).
c. Convert the virtual address 0x20043751 into a 32-bit physical addresses (also expressed in hexadecimal). Show your work and if the address can not be translated, explain why.
d. Convert the virtual address 0x30022267 into a 32-bit physical addresses (also expressed in hexadecimal). Show your work and if the address can not be translated, explain why.
e. Convert the physical address 0x51773721 into a 32-bit virtual addresses (also expressed in hexadecimal). Show your work and if the address can not be translated, explain why.

Scheduling

1. Three threads are in the ready to run queue, in the order, T1, T2, and T3.
   - Thread T1 runs a loop that executes 4 times. During each loop it runs for 1 unit of time and then makes a system call that blocks for 4 units of time. After looping the program exits.
   - Threads T2, and T3 run a loop that executes 2 times. During each loop they run for 3 units of time and then call thread yield. After looping the program exits.

   On a timeline, show when each thread runs and when the processor is idle by shading the appropriate thread or idle line. If two events happen at the same time, assume that T3 has the highest priority (i.e., its event occurs first) and T1 has the lowest priority.
   a. Assume a non preemptive FIFO scheduler and the starting point is as described at the top of this question.
   b. Assume a preemptive round-robin scheduler with a quantum of 2 and that the running time for the thread is reset to zero every time the thread executes (i.e., unused quantums do not carry over to the next time the thread runs). Use the starting point as described at the top of the question.

2. There are 3 threads in the system: T1, T2, and T3.
   - T1 has a priority of 1 and has been running for 3 time units.
   - T2 has a priority of 2 and has been running for 7 time units.
   - T3 has a priority of 3 and has been running for 6 time units.

   Which thread is scheduled to run next by the Linux Completely Fair Scheduler?

I/O

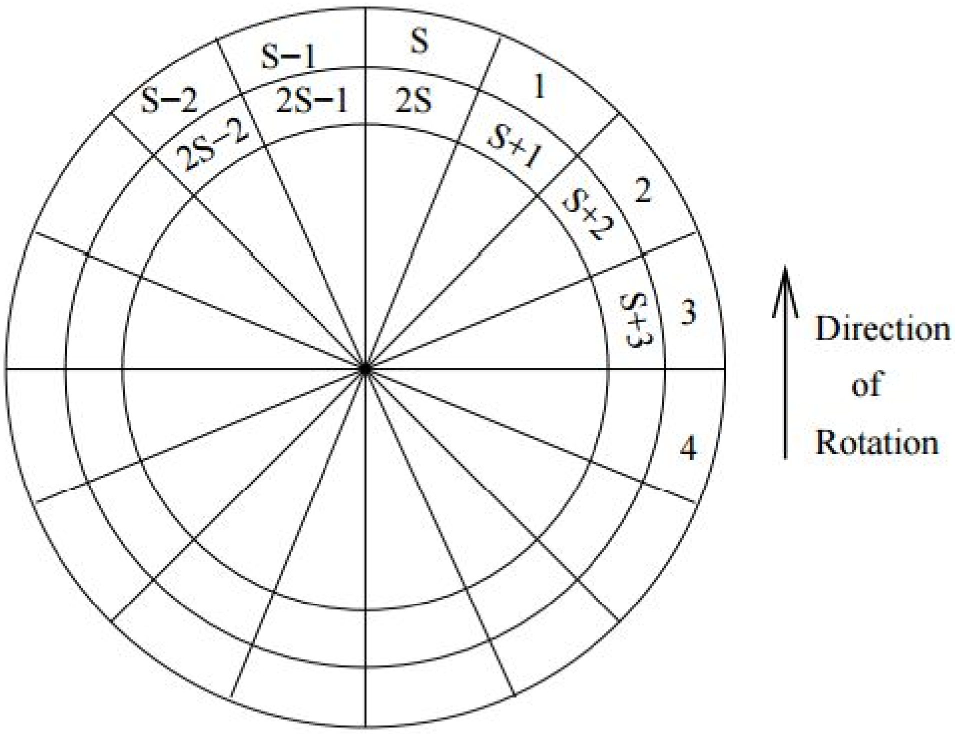
1. A disk drive has C cylinders, T tracks per cylinder, S sectors per track, and B bytes per sector. The rotational velocity of the platters is ω rotations per second.
   a. Consider s1 and s2, consecutive sectors on the same track of the disk. (Sector s2 will pass under the read/write head immediately after s1.) A read request for s1 arrives at the disk and is serviced. Exactly d seconds ($0 < d < 1/\omega$) after that request is completed by the disk, a read request for sector s2 arrives at the disk. (There are no intervening requests.) How long will it take the disk to service the request for sector s2?
   b. Suppose that s1 and s2 are not laid out consecutively on the disk. Instead, there are k sectors between s1and s2. For which value(s) of k will the time to service the read request for s2 be minimized? (Only consider $0 \le k \le S - 2$.)

2. A disk drive has S sectors per track and C cylinders. For simplicity, we will assume that the disk has only one, single-sided platter, i.e., the number of tracks per cylinder is one. The platter spins at ω rotations per millisecond. The following function gives

the relationship between seek distance d, in cylinders, and seek time, t_{seek}, in milliseconds:

$t_{seek} = 0$	$d = 0$
$t_{seek} = 5+0.05d$	$0 < d \le C$

The sectors are laid out and numbered sequentially, starting with the outer cylinder (cylinder 0), as shown in the diagram below.



a. Suppose the disk read/write head is located over cylinder 10. The disk receives a request to read sector S. What is the expected service time for this request?
b. Exactly d milliseconds after completing the request for S, the disk receives a request for sector S + 1. What is the expected service time for this request?

File Systems

1. Assume that a file F has been opened, and that F is 200,000 bytes long. Assume that the block size of the file system is 1,000 bytes, and that the size of a block pointer is 10 bytes. Assume that the file system uses UNIX-like index structures (i-nodes), each having ten direct data block pointers, one single indirect pointer, and one double

indirect pointer. Assume that the i-node for F is in memory but none of F's data blocks are.

a. How many data blocks does F occupy?
b. How many blocks in total (not including the i-node) does F occupy?
c. The process that opened F issues a read() request for bytes 9500-10300 from the file. How many blocks must the file system retrieve from the disk to satisfy this request?
d. The process that opened F issues a read() request for bytes 10500-11300 from the file. How many blocks must the file system retrieve from the disk to satisfy this request? Answer this question independently from part (c).

2. Assume that an i-node has 12 direct pointers, 1 single indirect pointer and 1 double indirect pointer. Assume that the block size of the file system is 4 KB and that the size of a block pointer is 32 bits.
   a. How many pointers fit in a single block?
   b. What is the largest number of data blocks that an i-node can reference?
   c. How many blocks must the file system retrieve from the disk (not including the i-node) to access the byte at offset 2^{23}?