Tamar Hariry

Operating Systems Final Assignment

① * It protects and separates processes from each other.
* Controls execution of programs to prevent errors and improper use of the computer.
Only 1 interrupt is generated per block, rather than the 1 interrupt per byte.

② - External mass storage
- input and output mechanisms
- memory that stores data and instructions
- A control unit that contains an instruction register and program counter
- A processing unit that contains an arithmetic logic unit and processor units
- (any stored-program computer in which an instruction fetch and a data operation cannot occur at the same time because they share a common bus.)

③

P_1 requesting <2,3,1>

	allocation	Max	Available	potential need
	A B C	A B C	A B C	A B C
P_0	1 1 0	5 5 5	3 3 3	4 4 5 ← 555 − 110
P_1	1 1 2 ← 231 + 343	4 4 4	333 − 231 = 102	1 0 1 ← 444 − 112
P_2	2 1 0	6 6 1		4 5 1 ← 661 − 210
P_3	3 2 1	5 2 2		2 0 1 ← 522 − 321
	Total: 10 8 6			

available + P_1 = 102 + 343 = 445

P_3 = + 321 = 766

P_0 = + 110 = 876

P_2 = + 210 = 10 8 6

$\boxed{\text{Answer: } <P_1, P_3, P_0, P_2>}$

state is safe, request can be granted

④ ⓐ size of page = 1mb (1 mil bit)

the formula for a page size 2^1n, we need 1 mg byte, so we use 2^120 to represent the amount of bytes within a page.

64 − 20 = 44, therefore, we need 2^144 pages.

(b) $2^{44} \cdot 1Mb = 2^{44} \cdot 2^{20} = 2^{64}$ bytes

⑤ TLB ~~[illegible]~~ is used nowadays with high speed memory
TLB reduces time taken to access user memory location

ⓒ split 44: 18 bits for inner page offset (2^{12} = 4-byte entries)
44-18 = 26 bits for outer page
2^{26} entries in the outer page table
every entry needs to be 8 bytes long
total size of outer page table = 2^{29} bytes ($2^{26} \cdot 8$)

ⓓ split 26: 18 bits for inner page offset (2^{12} = 4-byte entries)
8 bits outer page number
2^{8} entries in the outer page table
every entry needs to be 8 bytes long
total size of outer page table = 2^{11} bytes ($2^{8} \cdot 8$)

ⓔ advantages: makes the size of the outer page smaller from how it was two-level
disadvantages: there are more rows than hashed page tables, so it's not as efficient as an inverted page table

⑩

ⓕ benefits:
- no internal fragmentation
- could save memory if the segments are very small and ~~[illegible]~~ don't have to be combined into 1 page.

not solved:
- External fragmentation
- there are segments of unequal size and not suited so much for swapping.

⑦ ★ The main memory is more stable unlike TLB that changes more often
★ since TLB is already pretty fast and the main memory is slow, we would want to cache the main memory to access parts faster

⑧

	7	3	4	7	5	6	3	4	4	3	6	1	6	3	2	1	2	5	1	2	5	3	2	5	1
Frame 1:	7	7	7	7	7	7	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5
Frame 2:		3	3	3	5	5	5	4	4	4	4	1	1	1	2	2	2	2	2	2	2	2	2	2	2
Frame 3:			4	4	4	6	6	6	6	6	6	6	6	6	6	1	1	1	1	1	1	3	3	3	1
	F	F	F	H	F	F	F	F	H	H	H	F	H	H	F	F	H	F	H	H	H	F	H	H	F

F-Fault
H-Hits

⑨ Δ is 8, n=2 $\frac{8}{2}=4$, every 4 units there will be an interrupt

1 2 11 3 3 4 5 4 1 1 2 6 1 6 2 3 1 6 7

time 1 2 3 4 5 6 7 8 9 10 11 12 13

Page	time t1	t5	t9	t13	Answer 16
1	01	10	11	11	11
2	11	11	11	01	01
3	10	01	00	10	10
4	10	11	01	00	00
5	10	01	00	00	00
6	00	00	10	11	11
7	00	00	00	10	10
WS	(1,2,3,4,5)	(1,2,3,4,5)	(1,2,4,6)	(1,2,3,6,7)	WS = {1,2,3,6,7}
		added: 4,1,2 removed: 6,7	added: 6,1,2 removed: 3,5,7	added: 3,1,6,7 removed: 4,5	

same as 13 since it only changes at 17
13+4=17

⑩ FCFS:

56-55 = 1
55-37 = 18
37-76 = 39
76-88 = 12
88-185 = 97
185-30 = 155
30-126 = 96
126-12 = 114
12-100 = 88
100-69 = 31

1+18+43+12+97+185+96+114+88+31 = 651 distance

seek time ⇒ $\sqrt{(2 \cdot 651)} \sim 36.08$ time units

SSTF:

56-55=1
55-69=14
69-76=7
76-88=12
88-100=12
100-126=26
126-185=59
185-37=148
37-30=7
30-12=18

1+14+7+12+12+26+59+148+7+18 = 304 distance

seek time ⇒ $\sqrt{(2 \cdot 304)} \sim 24.65$ time units

SCAN: 56 - 69 = 13

69 - 76 = 7

76 - 88 = 12

88 - 100 = 12

100 - 126 = 26

126 - 185 = 59

185 - 199 = 14

199 - 55 = 144

55 - 37 = 18

37 - 30 = 7

30 - 12 = 18

13 + 7 + 12 + 12 + 26 + 59 + 14 + 144 + 18 + 7 + 18 = 330 distance

seek time ⇒ $\sqrt{(2 \cdot 330)}$ ~ 25.69

C-LOOK: 56 - 69 = 13

69 - 76 = 7

76 - 88 = 12

88 - 100 = 12

100 - 126 = 26

126 - 185 = 59

185 - 12 = 173

12 - 30 = 18

30 - 37 = 7

37 - 55 = 18

13 + 7 + 12 + 12 + 26 + 59 + 173 + 18 + 7 + 18 = 345

seek time ⇒ $\sqrt{(2 \cdot 345)}$ ~ 26.27

(11) **bad:**

Takes up alot of extra space because of all the extra pointers.

good!

you are able to go back to the original file since we created pointers back