Kai Wen Tsai

Ktsai017

861261944

CS153

HW3

I. Consider the following segment table:

Segment	Base	Limit
0	800	600
1	1200	140
2	70	100
3	1350	880
4	2200	96

A. Is the information in the segment table consistent? If there are possible errors that you identify, what will their implication be? (2 points)
   a. The information in the table is consistent.
B. What will be the result of translating the following virtual addresses?
   a. 4,90
      i. **The physical address would be location 2200 up to 96. 2200+90 = 2290**
   b. 2,200
      i. **The physical address would be location 70 up to 100. But 200 > 100 which is the limit, so this is illegal.**

II. (a) An OS is using two-level paging to implement a 28-bit virtual address space per process. The page size is 256-bytes, and the machine does not have a TLB. Explain the steps involved in looking up the virtual address 0x03bf04d, when all pages are present in memory. (2 points)

First, we need to know the bits of page size for the offset. 256 is equal to 2^8, we need 8 bits for it. For a two level paging, the virtual page number would be split into two part, one for index to the second level, another part for the second level of pages. Since there are 28 bits of VA, 28-8 = 20, 20 bits is used as VPN. Assuming that the entry size is 4. Each page is 256 bytes, which means that it can have 256/4=64 = 2^6 entries. 20-6 = 14. 14 bits are used as the VPN for outer pages. The VPN is 0x03bf04d which is

0000 0011 1011 1111 0000 0100 1101

Last eight bits is for the offset. 0000 0011 1011 1111 would be split into two part. Last 6 bits are used for the outer page offset. From that, we can do the translation from virtual address to physical address.

For the system above, what is the maximum number of page faults that could be generated in response to a memory access? (2 points)

i. **The page of the secondary page table might not be in the memory, that lead to a 2 page faults**

III. Consider a virtual address system with the following parameters.

- The memory is byte addressable.
- Virtual addresses are 16 bits wide.
- Physical addresses are 15 bits wide.
- The page size is 512 bytes.
- The TLB is fully associative with 16 total entries.

Recall that a fully associative cache has just one set of entries—The tag field is simply the VPN and we need to search the full TLB to check if the VPN we are seeking is in the TLB. In the following tables, all numbers are given in hexadecimal.

The contents of the TLB and the page table for the first 16 virtual pages are as follows. If a VPN is not listed in the page table, assume it generates a page fault.

TLB

Tag	PPN	Valid
03	1B	1
06	06	0
28	23	1
01	18	0
31	01	1
12	00	0
07	3D	1
0B	11	1
2A	2C	0
11	1C	0
1F	03	1
08	14	1
09	2A	1
3F	30	0
10	0D	0
32	11	0

Page Table

VPN	PPN	Valid
00	27	1
01	0F	1
02	19	1
03	1B	1
04	06	0
05	03	0
06	06	0
07	3D	0
08	14	1
09	2A	1
0A	21	1
0B	11	1
0C	1C	1
0D	2D	0
0E	0E	0
0F	04	1

Which bits of an address hold the VPO and which hold the VPN? (1 point)

i. **Page size is 2^9 which means Last 9 bits to represent VPO, VPN is first 16-9 = 7 bits**

(b) For the virtual address 0x0422, indicate the physical address. Indicate whether or not there is a TLB miss. If there is a page fault, enter "—" for the PPN and Physical Address. All answers should be given in hexadecimal. (1 points)

i. **0000 0100 0010 0010 is the virtual address. Last 9 bits are offset. So VPN is 0000 010 which is 0x02 in hex. In the TLB, there is no 02 tag, so it is a TLB miss. Next we check page table and find it in the entry and valid. So it is a page hit. PPN is 19 (0001 1001). We then replace VPN with PPN, so the physical address is 0011 0010 0010 0010 which is 0x3222**

(c) Repeat for address 0x02F1 (1 points)

i. **0000 0010 1111 0001 is VA. VPN is 0000 001 which is 0x01 in hex. In the TLB, there is no 01 tag, so it is a TLB miss. Next we check 01 in the page table and find a hit. PPN is 0F (0000 1111). Next we replace VPN with PPN, so the physical address is 0001 1110 1111 0001 which is 0x1EF1**

(d) Repeat for address 0x0c20 (1 points)

i. **0000 1100 0010 0000 is VA. VPN is 0000 110 which is 0x06 in hex. In the TLB, there is 06 in the tag, so it is a hit. PPN is 06 (0000 0110). Next we replace VPN with PPN, so the physical address is 110 000100000 which is 0x0C20**

IV. Consider a process that has been allocated 5 pages of memory: P1, P2, P3, P4, and P5. The process accesses these pages in the following order:

P1 P2 P3 P4 P1 P2 P5 P1 P2 P3 P4 P5

Illustrate Belady's anomaly by precisely describing the execution of the FIFO page eviction algorithm in two cases: a) where the machine has 3 pages of physical memory, and b) where the machine has 4 pages of physical memory, and by comparing the number of page faults incurred in these two cases. (When the process begins executing, none of its pages are present in memory.) (2 points)

1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

i.3 pages (row: page, column: order)

1	2	3	4	1	2	5	1	2	3	4	5
		1	2	3	4	1	1	1	2	5	5
	1	2	3	4	1	2	2	2	5	3	3
1	2	3	4	1	2	5	5	5	3	4	4
PF	PF	PF	PF	PF	PF	PF	X	X	PF	PF	X

Using 3 pages, there are 9 page faults

ii. 4 pages

1	2	3	4	1	2	5	1	2	3	4	5
			1	1	1	2	3	4	5	1	2
		1	2	2	2	3	4	5	1	2	3
	1	2	3	3	3	4	5	1	2	3	4
1	2	3	4	4	4	5	1	2	3	4	5
PF	PF	PF	PF	X	X	PF	PF	PF	PF	PF	PF

Using 4 pages, there are 10 page faults

Belady's algorithm state that it is possible to have more page fault when number of pages increases.

b. Show how the LRU page eviction algorithm would work in the same scenarios a) and b) described above.

1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

3 pages

1	2	3	4	1	2	5	1	2	3	4	5
		1	2	3	4	1	2	5	1	2	3
	1	2	3	4	1	2	5	1	2	3	4
1	2	3	4	1	2	5	1	2	3	4	5
PF	PF	PF	PF	PF	PF	PF	X	X	PF	PF	PF

Using 3 pages, there are 10 pages faults

ii. 4 pages

1	2	3	4	1	2	5	1	2	3	4	5
			1	2	3	4	4	4	5	1	2
		1	2	3	4	1	2	5	1	2	3
	1	2	3	4	1	2	5	1	2	3	4
1	2	3	4	1	2	5	1	2	3	4	5
PF	PF	PF	PF	X	X	PF	X	X	PF	PF	PF

Using 4 pages, there are 8 pages faults