Question 1: Consider a virtual memory system with a 40-bit virtual byte address, 8KB pages and 512 MB of physical memory.

5

a) What is the minimum number of bytes needed to store each entry of the page table assuming that the Valid, Protection, Dirty and LRU/Use bits take a total of one byte per entry?

2^{16} physical pages →Each page table entry needs 3 bytes

b) What is the total size of the page table (in bytes) for each process on this machine?

2^{27} virtual pages →Total # of byte in the page table = 3* 2^{27} =384MB

c) Express the size of the page table in terms of number of memory pages it will occupy

3* 2^{27} / 2^{13} = 3* 2^{14} = 48 Kilo pages.

d) Assume that a two level page table is to be implemented, where the first level page table is always stored in a fixed location in the physical memory and contains the page table entries for the virtual pages storing the page table (the second level of the page table). How many entries will the first level page table have and how many pages will it occupy?

- A first level PT will keep track of the virtual-to-physical mapping of the 3* 2^{14} pages of the PT
- The first level PT needs 3* 2^{14} PT entries, each occupying 3 bytes for a total of 9* 2^{14} bytes.
- The first level PT occupies 9* 2^{14} / 2^{13} = 18 pages.
- A 3-level PT can be used with a root PT tracking the virtual-to-physical mapping of the 18 pages (which will now become a level-2 PT rather than a first level PT)

3

Question 2: Consider a virtual memory system with a 40-bit virtual byte address, 512 MB of physical memory and 2MB superpages.

a) What is the minimum number of bytes needed to store each entry of the page table assuming that the Valid, Protection, Dirty and LRU/Use bits take a total of one byte per entry?

2^{8} physical pages →Each page table entry needs 2 bytes

b) What is the total size of the page table (in bytes) for each process on this machine?

2^{19} virtual pages → Total # of byte in the page table = 2* 2^{19} = 1MB.

c) Express the size of the page table in terms of number of memory pages it will occupy

The entire page table can fit in one superpage.

5

Question 3: Given the following page table and TLB for a system with 64-word pages and 16 physical pages:

8-entries TLB (2-way set associative)

valid	tag	Physical page #	valid	tag	Physical page #
1	0001	0011	1	0010	1110
0			0		
1	0000	1100	1	0001	0001
0			0		

Page Table

	valid	Physical page #
0	0	
1	0	
2	1	1100
3	0	
4	1	0011
5	0	
6	1	0001
7	0	
8	1	1110
9	1	1010
10	1	0101
11		
.	.	.
.	.	.
63	.	.

a) Assuming that the virtual word address is 12 bits, how many entries does the page table contain?

The virtual space consists of 2^{12} words.
Hence, it contains $2^{12} / 64 = 2^6 = 64$ virtual pages.
Hence, the page table contains 64 entries

b) For each of the following memory references (virtual word addresses) determine if it results in a TLB hit or miss, and whether it causes a page fault or not. Also determine the physical address for each reference that does not result in a page fault:

1) 001000101000

virtual page number 001000 and page offset 101000,
PT entry 0010 00 is mapped to the TLB at index 00 with tag 0010
No tag at TLB index 00 equals 0001 – hence TLB miss
Entry 001000 (decimal 8) of the PT is valid – hence no page fault
Entry 001000 of the PT indicates that the page is mapped to physical page 1110
Hence the physical address is 1110 101000 (where 101000 is the page offset)
Entry 001000 (decimal 8) of the PT will be cached in the TLB using index 00 and tag 0010

7

000110010101
virtual page number 000110 and page offset 010101,
PT entry 000110 →TLB at index 10 with tag 0001
The second tag in index 10 of the TLB matches 0001
Hence TLB hit, no page fault
Physical page number is 0001
Hence the physical address is 0001 010101

TLB (2-way set associative)

valid	tag	Physical page #	valid	tag	Physical page #
1	0001	0011	1	0010	1110
0			0		
1	0000	1100	1	0001	0001
0			0		

Page Table

	valid	Physical page #
0	0	
1	0	
2	1	1100
3	0	
4	1	0011
5	0	
6	1	0001
7	0	
8	1	1110
9	1	1010
10	1	0101
11		
12	.	.
13	.	.
	.	.

000010000111
virtual page number 000010 and page offset 000111,
PT entry 000010 →TLB at index 10 with tag 0000
The first tag in index 10 of the TLB matches 0000 – hence TLB hit, no page fault
Physical page number is 1100 → physical address is 1100 000111

000101100111
virtual page number 000101 and page offset 100111,
PT entry 000101 →TLB at index 01 with tag 0001
No TLB entry at index 01 is valid – hence TLB miss
From PT entry 000101 (decimal 5) , the page is not in memory (Page faults)

c) Show the content of the TLB and PT after the above 4 references assuming that any page brought to the physical memory will be placed in page 0111.

TLB (2-way set associative)

valid	tag	Physical page #	valid	tag	Physical page #
1	0001	0011	1	0010	1110
1	0001	0111	0		
1	0000	1100	1	0001	0001
0			0		

Page Table

	valid	Physical page #
0	0	
1	0	
2	1	1100
3	0	
4	1	0011
5	1	0111
6	1	0001
7	0	
8	1	1110
9	1	1010
10	1	0101
11		
12	.	.
13	.	.
	.	.

000101100111

virtual page number 000101 and page offset 100111,

PT entry 000101 → TLB at index 01 with tag 0001

No TLB entry at index 01 is valid – hence TLB miss

From PT entry 000101 (decimal 5) , the page is not in memory (Page faults)

The page will be brought to memory into physical page 0111

Entry 000101 (5) of the PT will be updated to reflect the mapping to physical page 0111

Entry 000101 of the PT will be cached in the TLB at index 01 with tag 0001