Homework 13

Problem 1

Imagine a system with the following specifications:

- The system has 4GB of virtual memory
- The system has 256KB of physical memory
- The page size is 4KB
- The page table size is 4KB
- Length of each PTE is 4B
- The TLB is 4-way set associative with 16 total entries
- The L1 d-cache has 4 sets and 4 bytes in each line
- LRU replacement policy in TLB and d-cache

1. Please fill the blanks about the characteristics of the system:

   bits of page offset: 12
   bits of PPN: 6
   bits of VPN: 20
   number of PTEs in one of page table: 1024
   bits of VPN of each level: 10
   number of levels: 2
   bits of tag in TLB: 18
   bits of tag in cache: 14

2. Part of the content of TLB, page tables, and L1 d-cache are given below:

TLB												
Ind	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid
00	03	--	0	09	0D	1	00	--	0	07	02	1
01	03	2D	1	02	--	0	04	--	0	0A	--	0
10	02	--	0	08	--	0	06	--	0	03	--	0
11	07	--	0	04	0D	1	0A	34	1	02	--	0

L1 Page Table											
VPN	PPN	Valid	VPN	PPN	Valid	VPN	PPN	Valid	VPN	PPN	Valid
00	18	1	05	--	0	0A	0C	1	0F	--	0
01	--	0	06	--	0	0B	--	0	10	--	0
02	13	1	07	--	0	0C	--	0	11	1A	1
03	12	1	08	--	0	0D	1D	1	12	--	0
04	--	0	09	22	1	0E	--	0	13	0D	0

L2 Page Table @ 0x18000											
VPN	PPN	Valid	VPN	PPN	Valid	VPN	PPN	Valid	VPN	PPN	Valid
00	28	1	05	--	0	0A	1C	1	0F	--	0
01	--	0	06	--	0	0B	--	0	10	--	0
02	33	1	07	17	1	0C	--	0	11	3A	1
03	02	1	08	--	0	0D	2D	1	12	22	1
04	--	0	09	02	1	0E	--	0	13	0D	1

L1 d-cache						
Index	Tag	Valid	block0	block1	block2	block3
0	3A27	1	0x0A	0x0B	0x0C	0x0D
1	--	0	--	--	--	--
2	1C28	0	0x1A	0x1B	0x1C	0x1D
3	D27	1	0x00	0x01	0x02	0x03

Step through the following address translation and cache accessing. If there is a cache miss, enter "--" for "Cache Byte Returned". If there is a page fault, enter "--" for "PPN" & "Physical address" and leave the second table empty.

1) VA: 0x1327c

Parameter	**Value**
VPN	0x13
TLB index	0x3
TLB tag	0x4
TLB hit? (Y/N)	Y
Page fault? (Y/N)	N
PPN	0x0D
Physical Address	0xD27C

Parameter	**Value**
Cache Byte Offset	0x0
Cache index	0x3
Cache tag	0xD27
Cache hit? (Y/N)	Y
Cache Byte Returned	0x00

2) VA: 0x0A289

Parameter	**Value**
VPN	0xA
TLB index	0x2
TLB tag	0x2

TLB hit? (Y/N)	N
Page fault? (Y/N)	N
PPN	0x1C
Physical Address	0x1C289

Parameter	Value
Cache Byte Offset	0x1
Cache index	0x2
Cache tag	0x1C28
Cache hit? (Y/N)	N
Cache Byte Returned	--

3) VA: 0x411272

Parameter	Value
VPN	0x411
TLB index	0x1
TLB tag	0x104
TLB hit? (Y/N)	N
Page fault? (Y/N)	Y
PPN	--
Physical Address	--

Parameter	Value
Cache Byte Offset	--
Cache index	--
Cache tag	--
Cache hit? (Y/N)	--
Cache Byte Returned	--

3. Calculate the **total page table size** (sum of all page tables of all levels) of this system. For another system with the same virtual memory space, page size, and PTE size, but only uses a **single** page table, calculate the **total page table size** of this system. Compare these two sizes, and **explain the benefits of using multi-level page table**.
   (this system) total page table size: (1+1024) * 4K = 4K + 4M
   (the other system) total page table size: 4M
   Benefits of multi-level page table: reduce memory requirements: 1) If a PTE in the level 1 table is null, then the corresponding level 2 page table does not even have to exit. 2) Only the level 1 table needs to be in main memory at all times. The other levels of page tables can be created and paged in and out, which reduces pressure on main memory.