CS61C Spring 2026 — Virtual Memory Discussion 13

1 Page Table Walk

Assume we have 16-bit VPNs, 12-bit PPNs, 8-bit page offsets, and 32-bit page table entries (PTEs). The first six entries of the page table are shown below.

Page Table	Valid?	Dirty?	PPN
0xB61C 0483			
0xFB83 A61C			
0x8483 3F01			
0x7ABC 4103			
0xC012 F7CB			
0x15DA C203			
...			

...where each page table entry (PTE) is formatted as:

1 Valid Bit	1 Dirty Bit	18 Status Bits	12 PPN Bits

1.1 Of the first 6 entries in the TLB, fill out the above table for each entry. List whether the PTE is a valid mapping. If so, list translate its corresponding physical page and if the page is clean/dirty.

1.2 For each of the following virtual addresses, answer whether accessing will result in a 1) Page Table Hit or 2) Page Fault, and translate to its corresponding physical address. Each access occurs independently, not sequentially. The next available free page has PPN 0x42D.

(a) `0x000429`

(b) `0x00018D`

(c) `0x000345`

1.3 Recall that the Page Table Base Register (PTBR) stores the physical address of our page table. For this program, the PTBR = `0x10000`. What is the physical address of the page table entry `0xC012F7CB`?

1.4 We want to reserve the first 10 pages of physical memory to be read-only. How can we modify our page table to accomplish this?

2 Page Table with TLB

2.1 A processor has 16-bit addresses, 256 byte pages, and an 8-entry fully associative TLB with LRU replacement (the LRU field is 3 bits and encodes the order in which pages were accessed, 0 being the most recent). The TLB for the current process is the initial state given below, and we have three free physical pages. Assume that all current page table entries are in the initial TLB. Write out the physical addresses of each location accessed and fill in the final state of the TLB according to the following access pattern. **Free Physical Pages**: `0x17`, `0x18`, `0x19`

Initial TLB

VPN	PPN	Valid	Dirty	LRU
`0x01`	`0x11`	1	1	0
`0x00`	`0x00`	0	0	7
`0x10`	`0x13`	1	1	1
`0x20`	`0x12`	1	0	5
`0x00`	`0x00`	0	0	6
`0x11`	`0x14`	1	0	4
`0xac`	`0x15`	1	1	2
`0xff`	`0xff`	1	0	3

Final TLB

VPN	PPN	Valid	Dirty	LRU

Access Pattern:

1. `0x11f0` (Read)

2. `0x1301` (Write)

3. `0x20ae` (Write)

4. `0x2332` (Write))

5. `0x20ff` (Read)

6. `0x3415` (Write)

3 Past Exam Question (SU25 Final Q8)

3.1 For the following system:

- 64 GiB physical memory
- 128 GiB virtual memory
- 1 MiB pages
- 32-bit page table entries

Answer the following questions.

(a) How many physical addresses are there? Express as a power of 2.

(b) How many bits are in the VPN, PPN, and page offset?

VPN:

PPN:

Offset:

(c) How many entries are in the page table?

(d) How many pages are needed to store the page table?

3.2 For the remaining questions, assume:

- 24-bit VPN
- 20-bit PPN
- 12-bit offset
- 64-bit PTEs

The TLB and first few page table entries are shown below.

PTE Format (64 bits)

63	62–20	19–0
Valid	**Status Bits**	**PPN**

TLB

Valid	VPN	PPN
1	0x000000	0x0061C
0	0x000001	0x00700
0	0x000002	0x10502
0	0x000003	0x01620

Page Table

0x BE51 C1A5 5150 061C
0x C41B B100 0000 1620
0x 061C 051A FF04 A111
0x C41B B100 0001 5100
…

For each virtual address below:

- Translate it to a physical address
- Indicate whether it is a:
  - TLB Hit
  - TLB Miss & Page Table Hit
  - Page Fault

Assume each access is independent. You can also assume that there is an implicit leading 0 for virtual address.

(a) `0x00000C16`

Physical Address:

Result:

(b) `0x0000200F`

Physical Address:

Result:

(c) `0x00003505`

Physical Address:

Result: