Initials: Manit Gandhi

### 3. Virtual Memory [40 points]

Assume a 32-bit byte-addressable machine that utilizes 4MB page size with a one-level page table.

What is the number of bits required for the page offset?

$\log_2(4)$
$= 22$ bits

Assuming that the index bits for each level in the page table are divided equally, what is the maximum number of page table entries per page table?

$32 - 22 = 10$

With all the above assumption, what is the maximum number of physical memory the system can support?

$2^{10} = 1024$

With all the above assumption, what is the maximum capacity is needed for the entire page table structure?

Let's create a much simpler **BIG endian** machine that still utilize 4MB page size, and 32-bit address. Assuming the following data in the memory and the page table root is at `0x0`, and the page table entries is 32-bit long, where the 10 most significant bits are used for the physical page number.

| Address | Values (in hexadecimal) [Lowest bit – Highest bit] |
|---|---|
| 0x00 | 00 10 20 30 40 50 60 70 80 90 a0 b0 c0 d0 e0 f0 |
| 0x10 | 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f |
| 0x20 | 00 10 20 30 40 50 60 70 80 90 a0 b0 c0 d0 e0 f0 |
| 0x30 | 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f |
| 0x40 | 00 10 20 30 40 50 60 70 80 90 a0 b0 c0 d0 e0 f0 |
| 0x50 | 19 15 12 0a 6b 3a 4b 12 91 ac ff fe 3c 3d 3e 4f |
| 0x60 | 12 50 62 8a 5e 5f df ea 99 ac 74 6b 91 44 33 ef |
| 0x70 | 70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f |
| 0x80 | 91 40 8a 00 8c 14 fe ff 74 13 02 ba 6b 12 4b 31 |
| 0x90 | 90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e 9f |
| 0xa0 | 80 00 8a 00 8c 14 fe ff 72 14 0a 6b 10 02 e1 ba |
| 0xb0 | 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f |
| 0xc0 | 80 00 8a 00 8c 14 fe ff 72 14 0a 6b 10 01 e1 ba |
| 0xd0 | 91 40 8a 00 8c 14 fe ff 74 13 02 ba 6b 12 4b 31 |
| 0xe0 | 70 00 8a 00 8c 14 fe ff 72 14 0a 6b 10 03 e1 ba |
| 0xf0 | 91 40 8a 00 8c 14 fe ff 74 13 02 ba 6b 12 4b 31 |

$\frac{32}{8} = 4$ bytes

$4 \times 16 = 64$

What is the physical address for a virtual address `0xdeadbeef`? Put in **Not enough information** if the table does not provide enough information to get the physical address.

d e a d b e e f

1101 1110 1101 1101 1011 1110 1110 1111

11 01 1110 10 → pg no. (virtual)

decimal = 890

890 > 64, not enough info

What is the physical address for a virtual address `0x0fffffff`? Put in **Not enough information** if the table does not provide enough information to get the physical address.

0 f f f f f f f

0000 1111 1111 1111 1111 1111 1111 1111

11 1111 1111 1111 1111 1111 → offset

0000 1111 11 → virtual pg no.

decimal = 63 → 6 3

6 b 1 2 4 6 3 1

0110 1010 0001 0010 0100 1011 0011 0001

physical address

0110 1010 00 11 1111 1111 1111 1111 1111

## 4. Translation Lookaside Buffer [20 points]

(a) What is the benefit of a translation lookaside buffer (TLB)?

It stores recent translation. Saves time because no need to access DRAM during translation. This reduces number of cycles.

(b) Assuming that the memory access takes 100 cycles to access DRAM, the system has 2-level page table, an TLB access takes 1 cycle, and a L1 cache access takes 1 cycle. How long does it takes to load a data that has a TLB miss and a L1 cache hit? Feels free to explain your answer.

look up = 1 cycle
2 level = 2x100 = 200 cycles
L1 = 1 cycle
200+1+1 = 202 cycles

(c) Assuming that the memory access takes 100 cycles to access DRAM, the system has 2-level page table, an TLB access takes 1 cycle, and a L1 cache access takes 1 cycle. How long does it takes to load a data that has a TLB hit and a L1 cache hit using a PIPT cache? Feels free to explain your answer.

TLB look up = 1 cycle
Cache look up = 1 cycle
1+1 = 2 cycles

(d) Given the above setup, how much faster is an access to an L1 cache if VIPT is used. Feels free to explain your answer

(e) Considering that the page offset is a piece of information shared by both the VA and the PA. How much data can a 256-entry TLB structure cover given the page size is 4kB. Why?

256 x 4 = 1024kB = 1 MB