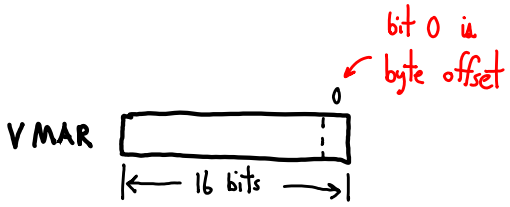
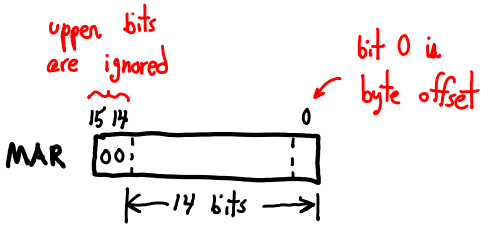
Consider the following 16-bit machine (datapath and register width is 2B).

-- Virtual memory:
  -- byte addressable.
  -- 16-bit virtual addresses.
  -- the VMAR is 16-bits.



-- Physical memory:
  -- byte addressable.
  -- 14-bit physical addresses.
  -- the MAR is 16-bits, but the upper
    2 bits are read-only and always zero.



-- Cache
  -- 8B cache blocks;
  -- 2-way Set-Associative (SA),
    that is, two Direct-Mapped (DM) caches running in parallel;
  -- 4 entries per DM;
  -- physically tagged, physically indexed

Q. Give the sizes of the following in bytes, words, cache blocks, and pages:

-- physical memory

14-bit address, byte addressable: $2^{14}\,B = 2^4 \cdot 2^{10}\,B = 16\,kB$

$16\,kB\left(\frac{word}{2B}\right) = 8\,k\ words$, $8\,k\ words\left(\frac{1\ block}{4\ words}\right) = 2\,k\ blocks$

$2^{14}\,B\left(\frac{page}{2^8\,B}\right) = 2^6\ pages = 64\ pages$

-- virtual memory

16-bit address byte addressable: $2^{16}\,B = 2^6 \cdot 2^{10} = 64\,kB$

$64\,kB\left(\frac{word}{2B}\right) = 32\,k\ word$ $\quad 32\,k\ word\left(\frac{1\ block}{4\ words}\right) = 8\,k\ blocks$

$2^{16}\,B\left(page/2^8\,B\right) = 2^8\ pages = 256\ pages$

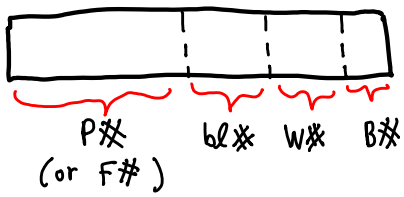
Q. How many words per page?

$(256\ B/page)(word/2B) = 128\ words/page$

Q. How many cache blocks per page?

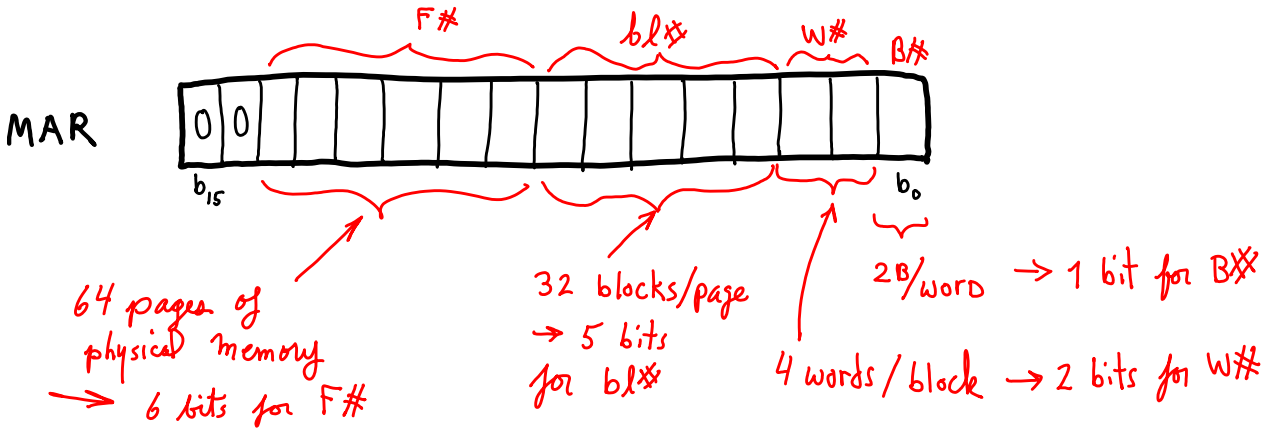
$(256\ B/page)(block/8B) = 32\ blocks/page$

A memory address can be thought of in this format:



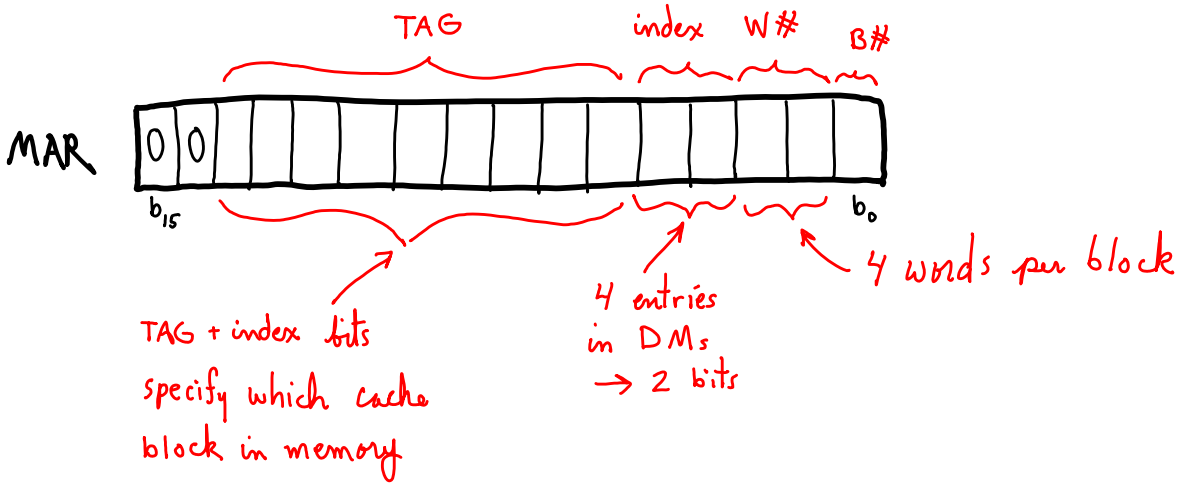
P# --- this number designates which page of virtual memory (or F#, which physical memory frame)
bl# --- which block within the page
W# --- which word within the block
B# --- which byte within the word

Q. Given the physical memory address register (MAR) shown below, indicate which bits correspond to each of the bit fields defined above (F#, bl#, W#, B#). Explain the size of each field.

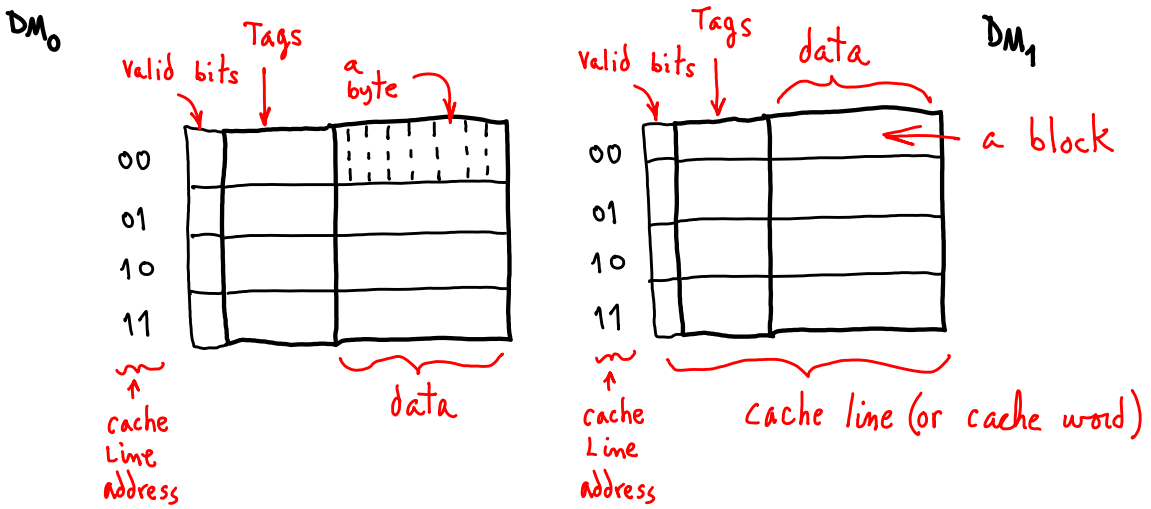


The same address bits can be thought of in this format: [TAG, index, W#, B#]:
TAG --- Address bits to identify the cache entry
index --- bits to specify which block or set of blocks in the cache
W# --- which word within the block
B# --- which byte within the word
Note that the [TAG, index] bits taken together could be thought of as specifying which block of memory: [bl#, W#, B#], where bl# is now a global block number, rather than which block in a page.

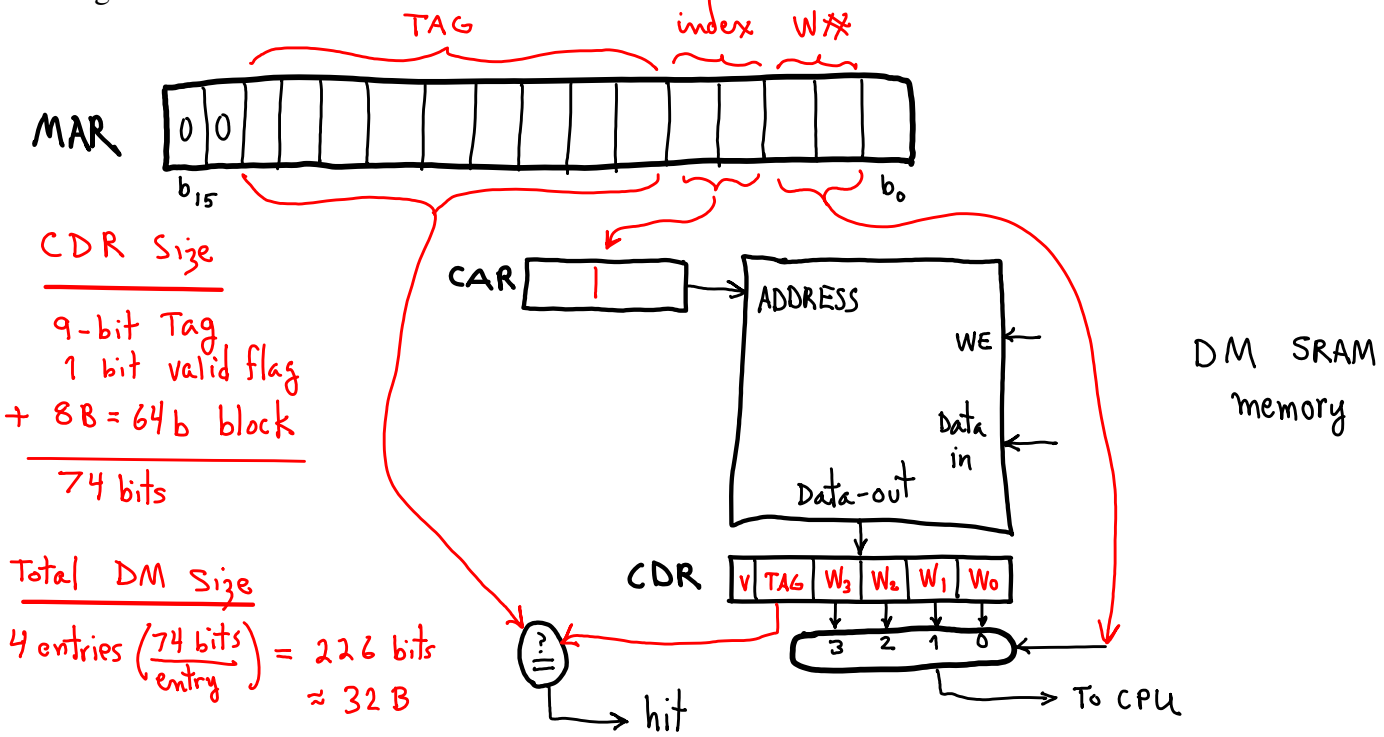
Q. Again given the MAR, indicate which bits correspond to each category above. Explain.



Q. The figure below shows the cache's two DMs. Indicate which blocks form each set and the index for each set.



Note that each DM is a small memory, with an address port, a data-out port, a data-in port, and a write-enable, as shown below. We can think of this memory as having an address register equivalent to an MAR, the Cache Address Register (CAR), and an equivalent to the MDR, the Cache Data Register. The diagram shows one of the two DMs.



Q. Draw the wire connections from the MAR to the CAR, to the MUX select, and to the comparator. Indicate the total number of bits in the CDR and total DM bits.

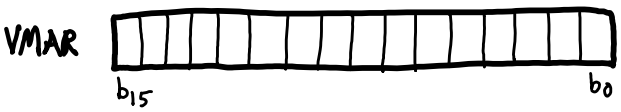
Q. Complete the diagrams below showing the virtual-to-physical address mapping, set indexing, tag comparators, hit signal logic, and datapath (including MUXing) to the CPU for a read operation (instruction or data) using:

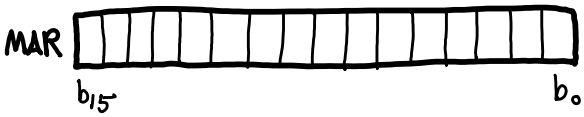
-- (1) physically indexed cache

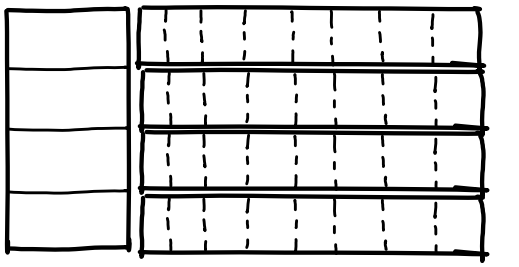
-- (2) virtually indexed cache

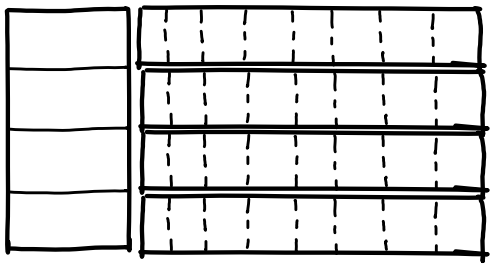
Assume that data sent to the CPU is a single word, and address translation is done in hardware. Indicate which are bits are routed where and data connectivity by byte.

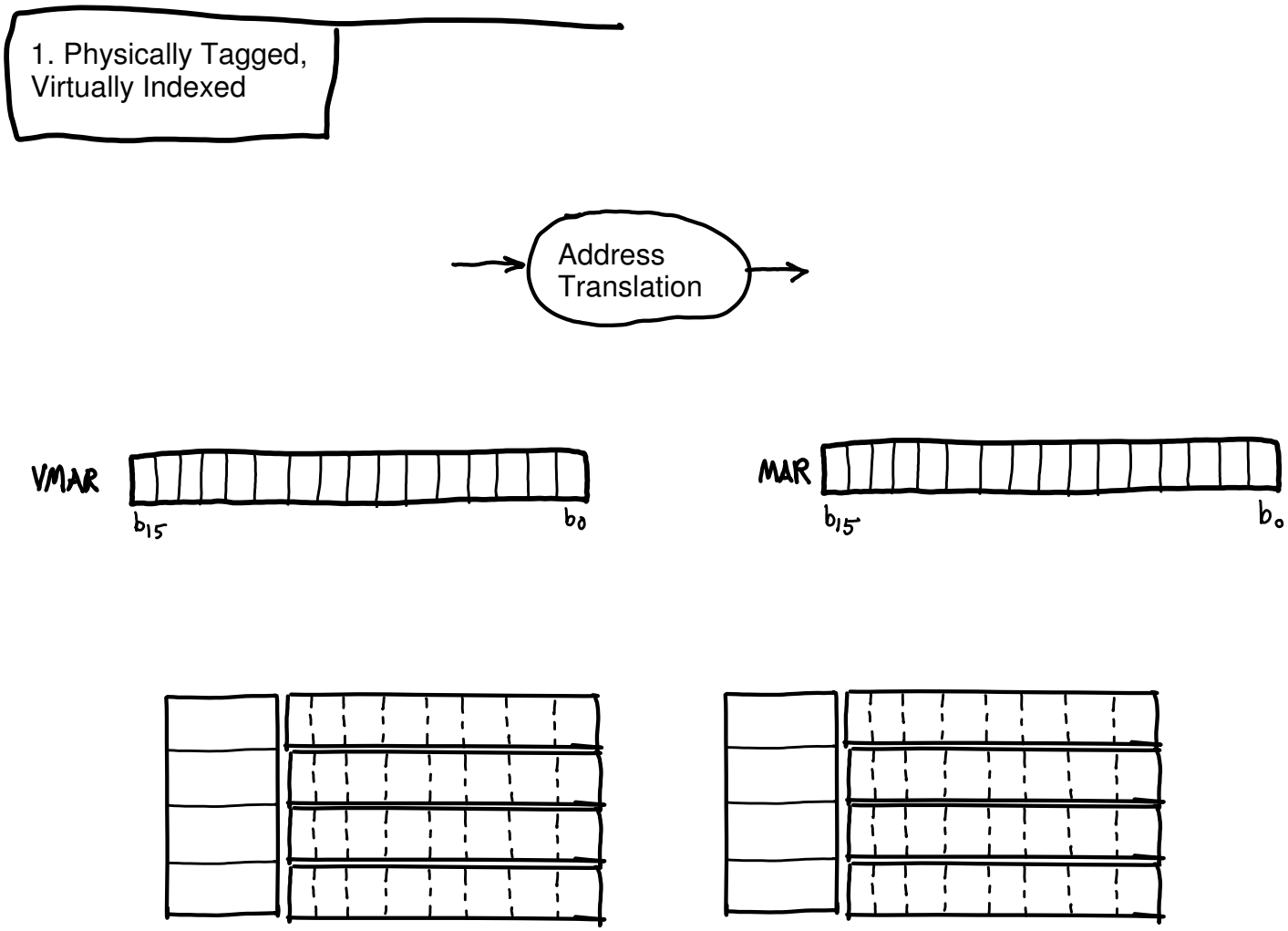
1. Physically Tagged, Physically Indexed











Q. For each design, (1) and (2), show the critical path that limits the CPU's clock cycle.

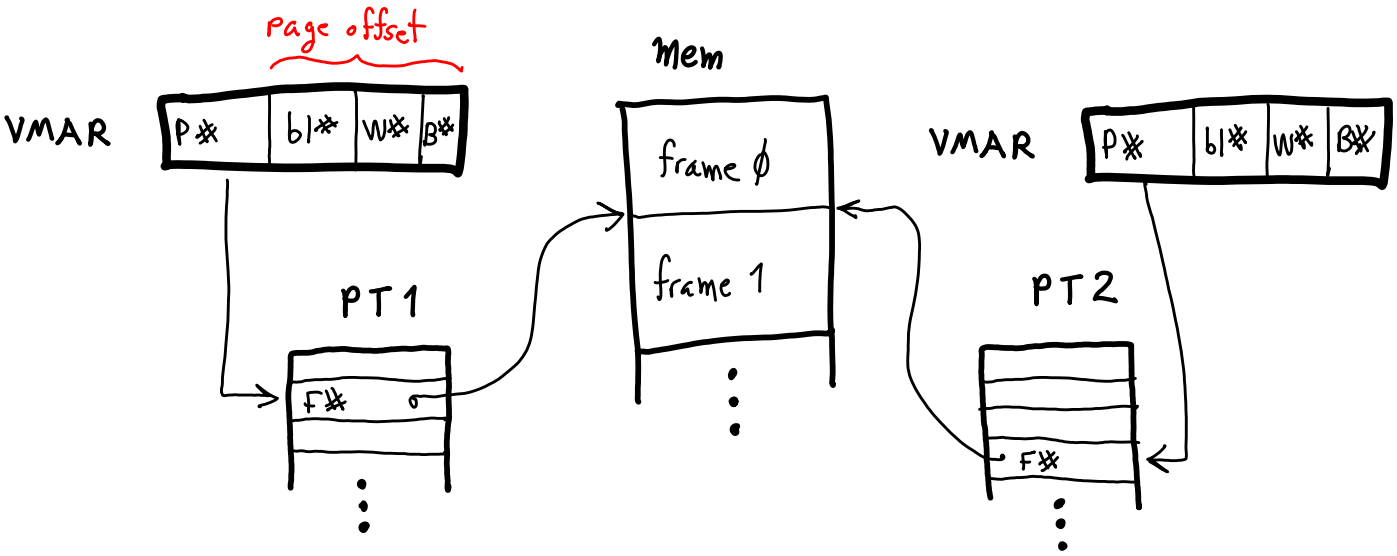
Q. Does one of your designs allow for a faster clock cycle than the other? Why or why not?

Two virtual memory accesses for the memory system above is shown below: on the left using page table PT1 and on the right PT2. (They could be the same page table.) The virtual page number portion of a virtual address is shown as P#, and the physical frame number as F#. Memory addresses and page table indices start from 0 at the top.

Page offset bits can be separated into three offsets: a cache block number within the page (bl#), a word number within a block (W#), and a byte number within a word (B#).

Q. For the 16-bit virtual address, how many bits designate a page number, P#? How many entries in a page table?

Q. How big (in bytes) is a page table? Aside from the address translation information, assume a page table entry includes 4 protection-bits, a valid-bit, a dirty-bit, and an accessed-bit. Assume the size of a page table entry is rounded up to an integer number of Bytes.



The above scenario shows two different virtual mappings for the same physical memory frame. This is called a synonym.

Q. Fill in P#s and F#s to match the indicated mappings.