Example cache (w/ DM) → Fast MATH processor (Intrinsity)

- embedded processor
- MIPS architecture
- 32 bit address word (data & instr. word)
- 12-stage pipline
- separate instruction and data caches
- Each cache is 16KiB or 4096 words
- 1 block = 16 words = 2^4 words

$MM\ size = 2^{32}B = 2^2 \cdot 2^{30}B = 4GiB$

32 bits = 4B = 2^2B → Byte offset (2)

Cache size 16KiB

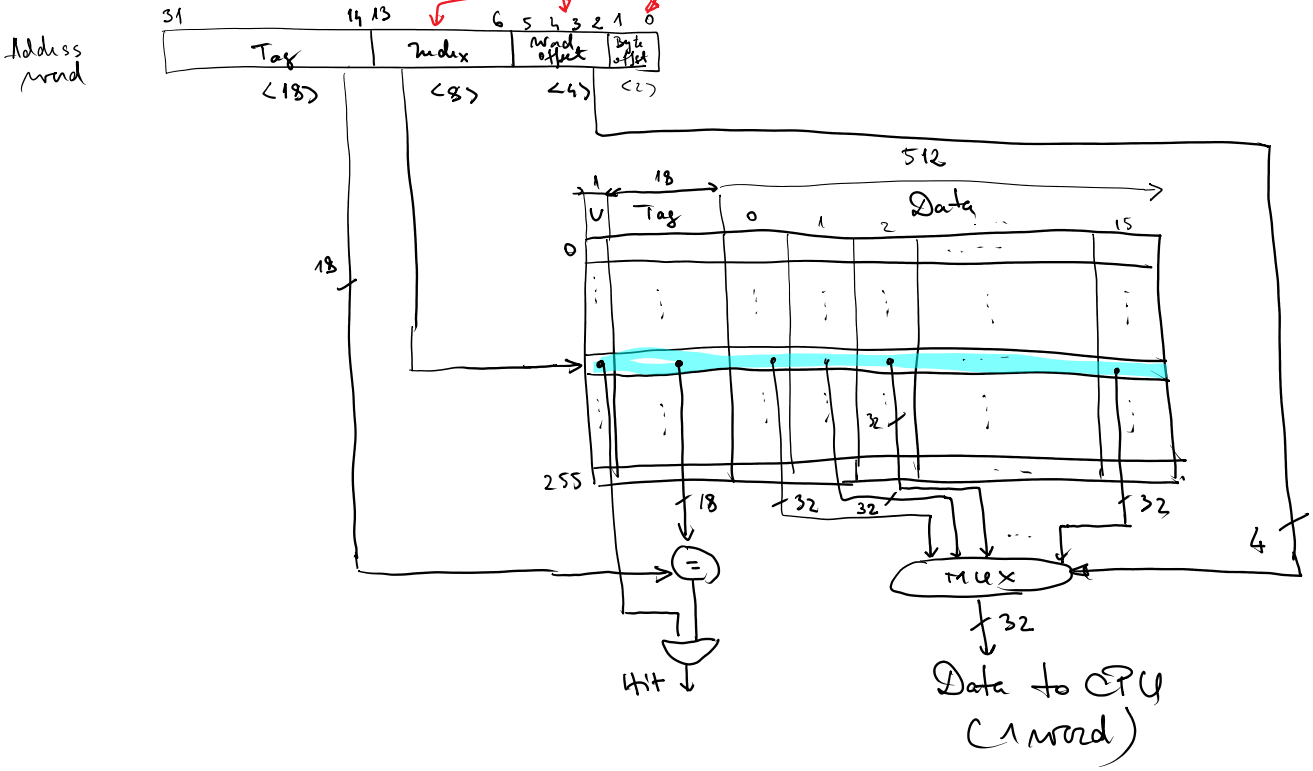
Block size = 2^4 words = $= 2^4 \cdot 2^2 B = 2^6 B$

$\frac{\text{Cache size}}{\text{Block size}} = \frac{2^4 \cdot 2^{10} B}{2^6} =$

$= 2^8$ lines → Index (8)

2^4 words/block = $2^4 \cdot 2^2 B = 2^6 B = 2^6 \cdot 2^3 = 2^9$ bits = 512 bits

Word offset (4)



1. Send address to the appropriate cache. Address comes either from PC (for an instruction) or from the ALU (data)
2. If the cache signals a hit, the requested word is available on the data lines. Since there are 16 words in a block, we need a MUX
3. If the cache signals a miss, the CPU sends the address to MM, and when MM returns the data, the data is written into the cache, then read by the CPU to fulfill the request.

Cache coherency → WT (Write Through)

→ WB (Write Back) ← both offered by Fast MATH

The OS decides