Lincoln Sand

1. Consider a program that can execute with no stalls and a CPI of 1 if the underlying processor can somehow magically service every load instruction with a 1-cycle L1 cache hit. In practice, 9% of all load instructions suffer from an L1 cache miss, 6% of all load instructions suffer from an L2 cache miss, and 3% of all load instructions suffer from an L3 cache miss (and are serviced by the memory system). An L1 cache miss stalls the processor for 8 cycles while the L2 is looked up. An L2 cache miss stalls the processor for 25 cycles while the L3 is looked up. An L3 cache miss stalls the processor for an additional 200 cycles while data is fetched from memory. What is the CPI for this program if 35% of the program's instructions are load instructions? (40 points)

Total instructions that are L1 misses: $0.35 \cdot 0.09 = 0.0315$

Total instructions that are L2 misses: $0.35 \cdot 0.06 = 0.021$

Total instructions that are L3 misses: $0.35 \cdot 0.03 = 0.0105$

Compute the additional cycles due to cache misses:

Extra cycles for L1 misses: $0.0315 \cdot 8 = 0.252$ cycles

Extra cycles for L2 misses: $0.021 \cdot 25 = 0.525$ cycles

Extra cycles for L3 misses: $0.0105 \cdot 200 = 2.1$ cycles

Calculate the total CPI including the base CPI of 1 (since the processor executes other instructions with no stalls):

Total CPI: $1 + 0.252 + 0.525 + 2.1 = 3.877$

2. Consider an L1 cache that has 8 sets, is direct-mapped (1-way), and supports a block size of 64 bytes. How many bits of the address are used to calculate the offset, index, and tag (assume that the CPU generates 32-bit addresses)? For the following memory access pattern (shown as byte addresses), show which accesses are hits and misses. For each hit, indicate the set that yields the hit. (30 points)

40, 96, 112, 24, 400, 500, 560, 32, 600, 48, 80.

Step 1: Calculate the sizes of the offset, index, and tag

Block size: $\log_2(64) = 6$ bits.

Number of sets (Index): $\log_2(8) = 3$ bits.

Tag: $32 - 6 - 3 = 23$ bits.

Step 2: Show the hits and misses for the list of memory addresses

For the given list, we get:

40: Miss

96: Miss

112: Hit, Set 1

24: Hit, Set 0

400: Miss

500: Miss

560: Miss

32: Miss

600: Miss

48: Hit, Set 0

80: Miss

3. A 128 KB L1 cache has a 128 byte block size and is 2-way set-associative. How many sets does the cache have? How many bits are used for the offset, index, and tag, assuming that the CPU provides 32-bit addresses? How large is the tag array? Please show your equations and steps. (30 points)

Step 1: Calculate the number of sets

Cache Size (C) = 128 KB (which is $128 \cdot 1024 = 131072$ bytes)

Block Size (B) = 128 bytes

Associativity (A) = 2-way

The number of sets (S) is given by:

$$S = \frac{C}{B \cdot A}$$

$$S = \frac{131072}{128 \cdot 2} = \frac{131072}{256} = 512$$

Step 2: Determine the number of bits for offset, index, and tag

Offset: $\log_2(128) = 7$ bits

Index: $\log_2(512) = 9$ bits

Tag: $32 - 7 - 9 = 16$ bits

Step 3: Calculate the size of the tag array

$$\text{Total tag array size} = S \cdot A \cdot \text{Tag Entry Size}$$

$$\text{Total tag array size} = 512 \cdot 2 \cdot 16 \text{ bits}$$

$$\text{Total tag array size} = 16384 \text{ bits} = 2048 \text{ bytes} = 2 \text{ KB}$$