## 7.11 Exercises

**7.1** [10] <§7.2> Describe the general characteristics of a program that would exhibit very little temporal and spatial locality with regard to data accesses. Provide an example program (pseudocode is fine).

**7.2** [10] <§7.2> Describe the general characteristics of a program that would exhibit very high amounts of temporal locality but very little spatial locality with regard to data accesses. Provide an example program (pseudocode is fine).

**7.3** [10] <§7.2> Describe the general characteristics of a program that would exhibit very little temporal locality but very high amounts of spatial locality with regard to data accesses. Provide an example program (pseudocode is fine).

**7.4** [10] <§7.2> Describe the general characteristics of a program that would exhibit very little temporal and spatial locality with regard to instruction fetches. Provide an example program (pseudocode is fine).

**7.5** [10] <§7.2> Describe the general characteristics of a program that would exhibit very high amounts of temporal locality but very little spatial locality with regard to instruction fetches. Provide an example program (pseudocode is fine).

**7.6** [10] <§7.2> Describe the general characteristics of a program that would exhibit very little temporal locality but very high amounts of spatial locality with regard to instruction fetches. Provide an example program (pseudocode is fine).

**7.7** [10] <§7.2> Here is a series of address references given as word addresses: 1, 4, 8, 5, 20, 17, 19, 56, 9, 11, 4, 43, 5, 6, 9, 17. Assuming a direct-mapped cache with 16 one-word blocks that is initially empty, label each reference in the list as a hit or a miss and show the final contents of the cache.

**7.8** [10] <§7.2> Using the series of references given in Exercise 7.7, show the hits and misses and final cache contents for a direct-mapped cache with four-word blocks and a *total size* of 16 words.

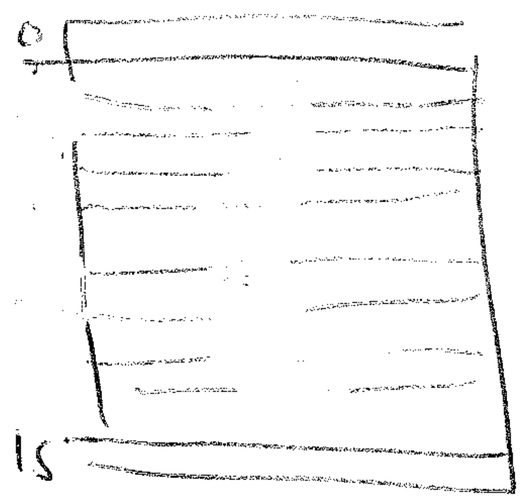
**7.9** [10] <§7.2> Compute the total number of bits required to implement the cache in Figure 7.10 on page 557. This number is different from the size of the cache, which usually refers to the number of bytes of data stored in the cache. The number of bits needed to implement the cache represents the total amount of memory needed for storing all of the data, tags, and valid bits.

7-7 set of references:

1, 4, 8, 5, 20, 17, 19, 56, 9, 11, 4, 43, 5, 6, 9, 17

Given no word addresses.

Assume direct mapped cache with 16 ~~word~~ blocks (each 1 word). Find hit, miss




mapping = Address module # of blocks

| Address | Map | hit/miss |
|---|---|---|
| 1 | 1 | miss |
| 4 | 4 | miss |
| 8 | 8 | miss |
| 5 | 5 | miss |
| 20 | 4 | miss |
| 17 | 1 | miss |
| 19 | 3 | miss |
| 56 | 8 | miss |
| 9 | 9 | miss |
| 11 | 11 | miss |
| 4 | 4 | miss (20) |
| 43 | 11 | miss (11) |
| 5 | 5 | hit |
| 6 | 6 | miss |
| 9 | 9 | hit |
| 17 | 1 | hit |

3 hit / 16

7-8 same reference if block size = 4 words and cache size = 16 words.

# blocks = 4

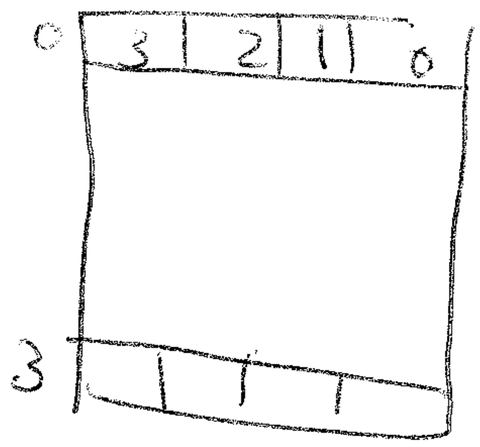
mapping

memory block number = Address / 4

mapped memory block # modulo 4

| Address | Memory block # | Cache block | hit/miss |
|---|---|---|---|
| 1 | 0 | 0 | miss |
| 4 | 1 | 1 | miss |
| 8 | 2 | 2 | miss |
| 5 | 1 | 1 | hit — |
| 20 | 5 | 0 | miss |
| 17 | 4 | 1 | miss (5) |
| 19 | 4 | 1 | hit (4) — |
| 56 | 14 | 2 | miss (2) |
| 9 | 2 | 2 | miss (14) |
| 11 | 2 | 2 | hit (2) — |
| 4 | 1 | 1 | miss (20) |
| 43 | 10 | 2 | miss (2) |
| 5 | 1 | 1 | hit — |
| 6 | 1 | 1 | hit — |
| 9 | 2 | 2 | miss (10) |
| 17 | 4 | 1 | hit (4) — |

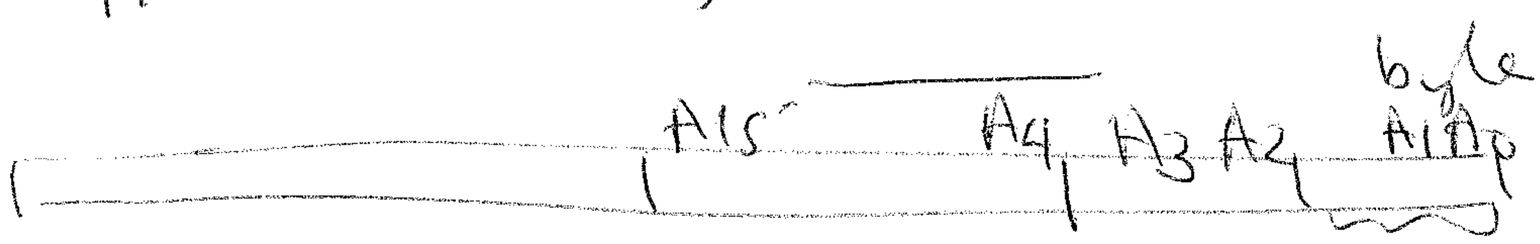
blk # Cache




6 hits

7-9 Find cache size (data, tag, valid bit)

has 4K entries, block = four words = 16 byte.




byte select = $\log_2$ 4 byte = 2 bits

Word select in block = $\log_2$ 4 = 2 bits

index = $\log_2$ 4K = 12 bits

Tag = 32 − 12 − 2 − 2 = 16 bits + 1 bit valid

= 17 bits X 4K = 2 byte X 4K + 4K bit

Data = 16B X 4K = 64KB

Total = 72 KB + 4K bit valid

**7.10** [10] <§7.2> Find a method to eliminate the AND gate on the valid bit in Figure 7.7 on page 549. (Hint: You need to change the comparison.)

**7.11** [10] <§7.2> Consider a memory hierarchy using one of the three organizations for main memory shown in Figure 7.13 on page 561. Assume that the cache block size is 16 words, that the width of organization b of the figure is four words, and that the number of banks in organization c is four. If the main memory latency for a new access is 10 cycles and the transfer time is 1 cycle, what are the miss penalties for each of these organizations?

**7.12** [10] <§7.2> {Ex. 7.11} Suppose a processor with a 16-word block size has an effective miss rate per instruction of 0.5%. Assume that the CPI without cache misses is 1.2. Using the memories described in Figure 7.13 on page 561 and Exercise 7.11, how much faster is this processor when using the wide memory than when using narrow or interleaved memories?

**7.13** [15] <§7.2> Cache C1 is direct-mapped with 16 one-word blocks. Cache C2 is direct-mapped with 4 four-word blocks. Assume that the miss penalty for C1 is 8 clock cycles and the miss penalty for C2 is 11 clock cycles. Assuming that the caches are initially empty, find a reference string for which C2 has a lower miss rate but spends more cycles on cache misses than C1. Use word addresses.

**7.14** [15] <§7.2> For the caches in Exercise 7.13, find a series of references for which C2 has more misses than C1. Use word addresses.

In More Depth

## Average Memory Access Time

To capture the fact that the time to access data for both hits and misses affects performance, designers often use average memory access time (AMAT) as a way to examine alternative cache designs. Average memory access time is the average time to access memory considering both hits and misses and the frequency of different accesses; it is equal to the following:

$$\text{AMAT} = \text{Time for a hit} + \text{Miss rate} \times \text{Miss penalty}$$

AMAT is useful as a figure of merit for different cache systems.

**7.15** [5] <§7.2> Find the AMAT for a machine with a 2-ns clock, a miss penalty of 20 clock cycles, a miss rate of 0.05 misses per instruction, and a cache access time (including hit detection) of 1 clock cycle. Assume that the read and write miss penalties are the same and ignore other write stalls.