1. Consider an array declared in C as "double a[100];". How many 64-byte cache lines are required to hold the complete array? [8pts]

total # lines in cache = size / line size

#lines = size / 64 bytes

#lines = sizeof(a) / 64 bytes = 100\*8 bytes/ 64 bytes

#lines = 100 / 8

#lines = 12.5 rounded up is 13

#lines = 13 64-byte cache lines

1. Consider the byte address 0x002468ac. What is the value modulo 64? (That is, what is the offset of this address within a 64-byte block?) [8pts]

The last 6 digits of 0x002468ac: 0x0000002c

1. Consider the byte address 0x002468ac. What is the value shifted to the right by 6 bits? (That is, what is the block address corresponding to this byte address when using 64-byte blocks?) [8pts]

0x002468ac = 0b 0000 0000 0010 0100 0110 1000 1010 1100

shift right 6 bits 🡪0b 0000 0000 0000 0000 1001 0001 1010 0010 = 0x000091a2

1. Consider matrix transpose written in C. Which array is exhibiting spatial locality: array "a", "b", or both? (Note that NROWS and NCOLS could each be relatively large compared to the size of the cache.) [8pts]

for(i=0;i<NROWS;i++){

for(j=0;j<NCOLS;j++){

b[i][j] = a[j][i];

}

}

matrix “b” exhibits spatial locality but matrix “a” DOES NOT

1. Consider a 4 GB byte-addressable main memory (32-bit address) with a level-1 data cache that is eight-way set-associative, 32 KB in size, with 64-byte block size. [24pts]
2. How many total blocks are there in cache?

1 KB = 1024 bytes

32 KB = 32768 bytes so 32768 / 64 = 512 blocks

1. How many sets are there?

512/8 = 64 sets

1. Show how the main memory address is partitioned into fields for the cache access and give the bit lengths of these fields.

Block size = 64 = 2^6 🡪 Offset size = 6 bits

# of sets = 64 = 2^6 🡪 Index size = 6 bits

Tag size = 32 bit – 6 bit – 6 bit = 20 bit

31 12 6 0

+------------------------------------+------+------+

| 20-bit tag | 6-bit | 6-bit |

+------------------------------------+------+------+

Tag Index offset

1. Consider a direct-mapped data cache design in which a 32-bit address is divided into these three fields: 18-bit tag, 10-bit index, and 4-bit offset. [24pts]
2. How large is a line in number of bytes?

2^offset = 2^4 = 16

1. How many lines are in the cache?

2^index = 2^10 = 1024

1. How large is the cache in number of bytes?

16 bytes \* 1024 lines = 16384 bytes

1. For the following segment of code written in C, where "sum" and the array "a" are typed as 4-byte integers, what is the miss rate?

(Assume the variable "sum" and the loop index "i" are register-allocated by the compiler within the body of the loop and thus do not cause data cache accesses within the loop.)

for(i=0;i<4096;i++){

sum = sum + a[i];

}

Miss rate is 25%. Each line can only hold 4 integers for a line size of 16 bytes and the array is being processed sequentially. Each cache miss fetches two integers from memory to cache. For example, a[0] is a miss. When a[0] is accessed, a[0], a[1], a[2], and a[3] are fetched from memory to cache. Thus when sum+a[1] through sum+a[3] ars calculated, a[1], a[2], and a[3] are hits. Similarly, a[4] is a miss but a[5], a[6], and a[7] are hits, and so far and so forth.

1. Assume a 256-byte main memory and a four-line cache with four bytes Per line. The cache is initially empty. For the byte address reference stream (reads) given below circle which of the references are hits for the different cache placement schemes. Also, show the final contents of the cache. (The byte addresses are in decimal.) [20pts]

a) direct-mapped

0, 16, 8, 1, 10, 30, 18, 29, 2, 25

Fields: 2-bit tag | 2-bit index | 2-bit offset

e.g. 0 = 0b 00 00 00, 16 = 0b 01 00 00, 8 = 0b 00 10 00, 1 = 0b 00 00 01, 10 = 0b 00 10 10

Hits on 10 and 29

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 16 | 8 | 1 | 10 | 30 | 18 | 29 | 2 | 25 | FINAL |
| Line 0b00 | 0-3 | 16-19 |  | 0-3 |  |  | 16-19 |  | 0-3 |  | 0-3 |
| Line 0b01 |  |  |  |  |  |  |  |  |  |  | empty |
| Line  0b10 |  |  | 8-11 |  | 8-11 |  |  |  |  | 24-27 | 24-27 |
| Line 0b11 |  |  |  |  |  | 28-31 |  | 28-31 |  |  | 28-31 |

b) fully-associative with first-in-first-out replacement

0, 16, 8, 1, 10, 30, 18, 29, 2, 25

Hits on 1, 10, 18, 29, and 2

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 16 | 8 | 1 | 10 | 30 | 18 | 29 | 2 | 25 | FINAL |
| Line ob00 | 0-3 |  |  | 0-3 |  |  |  |  | 0-3 | 24-27 | 24-27 |
| Line 0b01 |  | 16-19 |  |  |  |  | 16-19 |  |  |  | 16-19 |
| Line 0b10 |  |  | 8-11 |  | 8-11 |  |  |  |  |  | 8-11 |
| Line 0b11 |  |  |  |  |  | 28-31 |  | 28-31 |  |  | 28-31 |