Eric Paulz (epaulz)
CPSC 3300-002

Homework 7

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]
   (100 * 8 bytes) / 64 = 12.5

2. 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]
   0x2c = 44

3. 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]
   0x91a2 = 37282

4. 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];
  }
}
```

I think that array “b” is showing spatial locality because the convention of a 2D array is to read each row completely before moving to the next one. Array “a” skips to the next row on each iteration, so that data is not being accessed sequentially like array “b”s is.

5. 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 (8pts each)]

a) How many total blocks are there in cache?
   256
b) How many sets are there?
   128
c) Show how the main memory address is partitioned into fields for the cache access and give the bit lengths of these fields.

Tag (25 bits)	Index (3 bits)	Offset (4 bits)

Eric Paulz (epaulz)
CPSC 3300-002

6. Consider a direct-mapped data cache design in which a 32-bit address is divided into these three fields: 20-bit tag, 9-bit index, and 3-bit offset. [24pts (6pts each)]

a) How large is a line in number of bytes?
64

b) How many lines are in the cache?
8

c) How large is the cache in number of bytes?
256

d) 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];
}
```

.5 miss rate

7. 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 (10pts each)]

a) direct-mapped

0, 16, 1, 31, [✓ 2,] 32, 3, 17, 4, [✓ 18]

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

0, 16, [✓ 1,] 31, [✓ 2,] 32, [✓ 3,] [✓ 17,] 4, [✓ 18]