Caches and Virtual Memory

8.32 A 4-way set associative cache has a size of 32 KB and a block size of 64 bytes. The number of bits in an address is 32. Determine the number of bits for the tag, index, and block offset.

number of sets in cache:

$32768 \div 64 = 512$ blocks (32 KB = 32×1024 = 32768 Bytes. Cache size divided by block size)

$512 \div 4 = 128$ sets (number of blocks divided by set associativity)

number of bits for cache index:

$\log_2 128 = 7$ bits (log base 2 of number of sets)

number of bits for block offset:

$\log_2 64 = 6$ bits (log base 2 of cache block size)

number of bits for cache tag:

$32 - 7 - 6 = 19$ bits

8.33 Consider a 2-way set associative cache with a 14-bit tag, 8-bit index, and 4-byte block size. Suppose that all of the valid cache entries are as shown in the table.

		Data			
Tag	Index	Byte 00_2	Byte 01_2	Byte 10_2	Byte 11_2
01010101011110_2	00000000_2	DE_{16}	AD_{16}	BE_{16}	EF_{16}
11110110110111_2	00000000_2	ED_{16}	DA_{16}	EB_{16}	FE_{16}
01010101011110_2	00000001_2	$A0_{16}$	$A1_{16}$	$A2_{16}$	$A3_{16}$
11111111000000_2	00000001_2	$0A_{16}$	$1A_{16}$	$2A_{16}$	$3A_{16}$
01010101011110_2	01111110_2	90_{16}	91_{16}	92_{16}	93_{16}
11111001011110_2	01111110_2	09_{16}	19_{16}	29_{16}	39_{16}
01010101011110_2	01111111_2	$B0_{16}$	$B1_{16}$	$B2_{16}$	$B3_{16}$
11110000111100_2	01111111_2	$0B_{16}$	$1B_{16}$	$2B_{16}$	$3B_{16}$
01010101011110_2	10000000_2	$C0_{16}$	$C1_{16}$	$C2_{16}$	$C3_{16}$
01100110001011_2	10000000_2	$0C_{16}$	$1C_{16}$	$2C_{16}$	$3C_{16}$
01010101011110_2	10000001_2	$D0_{16}$	$D1_{16}$	$D2_{16}$	$D3_{16}$
11010100010110_2	10000001_2	$0D_{16}$	$1D_{16}$	$2D_{16}$	$3D_{16}$
01010101011110_2	11111111_2	$E0_{16}$	$E1_{16}$	$E2_{16}$	$E3_{16}$
11110110101111_2	11111111_2	$0E_{16}$	$1E_{16}$	$2E_{16}$	$3E_{16}$

(a) Determine the number of blocks in the cache.

(b) Determine the number of blocks in memory.

(c) Determine the cached value for the byte at address $557A02_{16}$ if present.

(d) Determine the cached value for the byte at address $FFFFFF_{16}$ if present.

(a)

$2^8 = 256$ sets

$2 \times 256 = 512$ blocks

(b)

$2^{14+8+2} = 2^{24} = 16777216$ bytes

$2^{24} \div 2^2 = 2^{22} = 4194304$ blocks

(c)

14 bits tag | 8 bits index | 2 bits block offset

$557A02_{16} = (01010101011110 \mid 10000000 \mid 10)_2$

data: $C2_{16}$

(d)

14 bits tag	8 bits index	2 bits block offset

$FFFFFF_{16} = (11111111111111 \mid 11111111 \mid 11)_2$

Cached value for address $FFFFFF_{16}$ is not present in the table, cache miss.

8.34 Consider the code fragments A and B, as given in Listings 8.1 and 8.2, where the variable `a` is declared as

```
double a[1024][1024];
```

Determine the number of cache misses that occurs during the execution of each of the code fragments A and B, subject to the following assumptions:

- the system has a 8 KB direct-mapped cache with a block size of 64 bytes;
- the cache is initially empty;
- the variables `sum`, `i`, and `j` are kept in registers for the duration of the code fragment execution, so that accesses to these variables do not impact caching;
- an object of type **`double`** requires 8 bytes of storage (i.e., **`sizeof`**`(`**`double`**`)` is 8);
- the array `a` is aligned on a 64-byte boundary; and
- while the code fragment is running, no other code executes.

Listing 8.1: Code fragment A

```
1 sum = 0.0;
2 for (int i = 0; i < 1024; ++i) {
3     for (int j = 0; j < 1024; ++j) {
4         sum += a[i][j];
5     }
6 }
```

Listing 8.2: Code fragment B

```
1 sum = 0.0;
2 for (int i = 0; i < 1024; ++i) {
3     for (int j = 0; j < 1024; ++j) {
4         sum += a[j][i];
5     }
6 }
```

$8KB = 8 \times 1024 = 2^3 \times 2^{10} = 2^{13} = 8192$ bytes

Code fragment A:

number of cache block

$8192 \div 64 = 2^{13} \div 2^6 = 2^7 = 128$ blocks

number of objects for each cache block

$64 \div 8 = 8$ objects

number of cache misses for each $a[i]$

$1024 \div 8 = 2^{10} \div 2^3 = 2^7 = 128$ cache misses

total number of cache misses for 1024 objects

$1024 \times 128 = 2^{10} \times 2^7 = 2^{17} = 131072$ cache misses

Code fragment B:

total number of cache misses for 1024×1024 objects

$2^{17} \times 2^{10} = 2^{27} = 134217728$ cache misses

8.36 Consider a system with 24-bit virtual addresses, 16-bit physical addresses, and a 1 KB page size. Determine the number of virtual and physical pages, and the number of bits in a virtual page number, physical page number, and page offset.

page size: 1 KB = 2^{10} = 1024 bytes

number of virtual pages:

$2^{24} \div 2^{10} = 2^{14} = 16384$

(number of virtual addresses divided by page size)

number of bits in a page offset:

$\log_2 1024 = 10$ bits

(log base 2 of page size)

number of bits in a virtual page number:

$24 - 10 = 14$ bits

number of phyical pages:

$2^{16} \div 2^{10} = 2^6 = 64$

(number of physical addresses divided by page size)

number of bits in a physical page number:

$16 - 10 = 6$ bits

8.37 Consider a system where the sizes of a virtual page number (VPN), physical page number (PPN), and page offset (PO) are 14, 6, and 10 bits, respectively. Suppose that the page table contains the following information for address translation and protection:

VPN	PPN	Flags
00000000000001_2	001000_2	present, readable, not writable, executable
00000000000010_2	001001_2	present, readable, not writable, not executable
00000000000011_2	001010_2	present, readable, writable, not executable
11111111111100_2	001111_2	present, readable, writable, not executable
11111111111101_2	001110_2	present, readable, writable, not executable
11111111111110_2	001101_2	present, readable, writable, not executable
11111111111111_2	001100_2	present, readable, writable, not executable

Determine the result of the address translation and protection check for each of the following accesses to memory:

(a) a data read (1 byte) at address $000000000000000000000000_2$;
(b) a data write (2 bytes) at address $000000000000110011001100_2$; and
(c) an instruction fetch (2 bytes) at address $111111111111110000000000_2$.

(a)

14 bits VPN | 10 bits PO

$(00000000000000 \quad | \quad 0000000000 \quad)_2$

access violation(no address mapping)

(b)

14 bits VPN | 10 bits PO

$(00000000000011 \quad | \quad 0011001100 \quad)_2$

physical adress: 001010_2

(c)

14 bits VPN | 10 bits PO

$(11111111111111 \quad | \quad 0000000000 \quad)_2$

access violation(physical adress: 001100_2 not executable)