Danny Tan

1i.

Address length = 6\*4=24

Set is log2(4096) = 12

Offset is log2(16) = 4

Tag = 24-12-4 = 8 bit.

So the stored will be 0xAB = (10101011)2

ii.

Set is log2(4096/16) = log2(256) = 8

24 – 8 - 4 = 12 bit

So the stored will be So 0xABC (101010111100)2

iii.

Set is 0 for fully associative, 24 – 0 – 4 = 20 bits

So the stored will be So 0xABCDE = (10101011110011011110)2

2.a. The address is split up into the offset bit, set bit and the tag bit.

Address length is 20

Offset is log2(256) = 8 bit

Set is log2(32/2) = 4 bit

Tag is 20 – 8 – 4 = 8 bit

Therefore the address is split up into tag: 8 bit, set: 4 bit, offset: 8 bit.

b.

So the set size is 4 bit.

The sets will be 7 and c each with 2 size.

0x55c88, 0x55774, 0x5479c, 0x54c00 will be compulsory misses.

0x55784 will be a hit

0x56c80 will be a compulsory miss

0x56718 will be a compulsory miss

0x54738 will be a conflict miss

c. The hit rate is 1/8 = 12.5 percent

d. Since the hit rate is 12.5 percent and is so low, the cache is a bad cache and it will not help the system.

3.a 2

b. Gcc terminal or in the code as an include

c.

Advantage for gcc terminal: Using library statically – will produce a reduction in overhead from no longer having to call functions from a library and this will give a faster load time

Disadvantage for gcc terminal: Will produce a larger executable file since it has to compile all of the library code

Advantage for code as an include: Using library dynamically - create smaller executable file and reduce in memory space since it reduces total resource consumption

Disadvantage for code as an include: Will give a slower load time since it keeps on having to function from a library

4.

copy:

movl %rdi, %ecx // exc = i

movl %rsi, %ebx // ebx = j

leal (%ecx,%ecx,8), %edx //edx = 9\*i

sall $2, %edx // edx = (9 \* i) \* 4

movl %ebx, %eax // eax = j

sall $4, %eax //eax = 16 \* j

subl %ebx, %eax //eax = 16 \* j – j = 15 \* j

sall $2, %eax //eax = (15\*j) \* 4

movl array2(%eax,%ecx,4), %eax // array2 + eax + ecx \* 4

//array2 + 15\* j \* 4 + i \* 4

// array2 + M \* j \* sizeof(type) + I \* sizeof(type)

movl %eax, array1(%edx,%ebx,4) //array1 + edx +ebx \* 4

// array1 + 9 \* i \* 4 + j \* 4

//array1 + N \* I \* 4 + j \* 4

ret

So M is 15 and N is 9

5.

movl %rdi, %edx // edx = a

movl %rsi, %eax // eax = b

cmpl %eax, %edx if a >= b return eax

jg2 .L9

movl %edx, %eax if a < b return edx

The first one

int fun2(int a, int b) {

if (a < b ) return a;

else

return b;

}

6.

|  |  |  |  |
| --- | --- | --- | --- |
| Hexadecimal | Binary | Decimal signed | Decimal unsigned |
| 0x8A | 1000 1010 | -118 | 138 |
| 0x21 | 0010 0001 | 33 | 33 |