1 （20 分）. 假设你正在运行一个程序，该程序具有以下数据访问模式。这种模式只执行一次：

0x0 0x8 0x10 0x18 0x20 0x28

（1）如果使用直接映射缓存（Direct Mapped Cache），缓存大小为 1 KB，块大小（Block Size）为 8 字节（bytes），缓存中有多少组（sets）？

2^10/2^3=2^7=128 sets

（2）使用与问题（1） 中相同的缓存和块大小，对于给定的内存访问模式，直接映射缓存的缺失率（miss rate）是多少？

100%

（3）对于给定的内存访问模式，提出一种降低缺失率的缓存设计（保持缓存大小不变），并给出相应的解释。

Increasing the block size to 16 bytes, which decreases the miss rate by 50%.

（4）如果使用全相联数据缓存（Fully-associative Data Cache），缓存大小为 8 KB，块大小为 32 字节，假设地址位为 32 位（bits）。该缓存有多少组（sets）、有多少路（ways）、有多少索引位（index bits）、偏移位（offset bits）、标记位（tag bits）、标记数组（tag store）有多大？

Equations:

Sets = 1 due to fully associativity

As cache size = #sets x #ways x blocksize, we have Ways = 2^13/2^5= 2^8=256,

Index bits = log2 (#sets) = 0

Offset bits = log2 (blocksize) = 5 bits

We have Tag bits + index bits + offset bits = address width, thus Tag bits = 32-0-5=27 bits

Tag array size = #sets x #ways x tagsize, we have 27*1*256=6912 bits

2 （20 分）. 8 KB 的全相联数据缓存（fully-associative cache），Cacheline 大小为 64 字节，假设地址位为 40 位。

有多少个组（sets）？有多少路（ways）？

有多少索引位（index bits）、偏移位（offset bits）、标记位（tag bits）？

标记数组（tag store/array）有多大？

组(sets): 全相连 sets = 1

路(ways): 8KB/64B=2^13/2^6=2^7=128

索引位(index): 0

偏移位(offset): 64B => offset bit = 6b

标记位(tag bits): 40-6=34b

标记数组(tag store): 128*1*34b=544B

3 （20 分）. 考虑以下循环，在有 16 字节大小的指令缓存（I-cache）的系统上执行。数据缓存（D-cache）是全相联的，大小为 1 KB。两个缓存都使用 16 字节的块（Block）。指令长度和数据字大小为 4 B。寄存器 Register $1 的初始值为 40。寄存器 Register $0 的值为 0。（注意：假设循环的第一个指令与缓存块的开始对齐）。

```
Loop:   lw      $6,  X($1)
        addi    $6,  $6,  1
        sw   $6,  Y($1)
        subi    $1,  $1,  4
        beq     $1,  $0,  Exit
        j       Loop
Exit:   …
```

(a) 计算 I-cache 和 D-cache 的 miss rate，考虑以下情况：

X 和 Y 是不同的数组。

X 和 Y 是相同的数组。

First, we analyze a case with an infinite number of iterations (or a multiple of 4 iterations):

The I-cache can keep 4 instructions. Thus, in each iteration there will be 2 cache misses. The I-cache miss

rate will be 2/6 = 0.33.

• X and Y are different arrays.

If X and Y are different arrays, there will be 2 cache misses every 4 iterations, that is, one read miss

and one write miss every 8 accesses. In that case, the D-cache miss rate will be 2/8 = 0.25.

• X and Y are the same array.

If X and Y are the same array, the D-cache miss rate will be one half of the previous one (0.125).

Second, we consider 10 iterations of the loop:

In the last iteration, the jump is not executed. Thus, the I-cache miss rate is 20/59 = 0.34.

• X and Y are different arrays.

If X and Y are different arrays, there will be 2 cache misses every 4 iterations, but for the last two

iterations there will be 2 cache misses too. The D-cache miss rate will be 3/10 = 0.30.

• X and Y are the same array.

If X and Y are the same array, the D-cache miss rate will be one half of the previous one (0.15).

(b) 计算每条指令的平均周期数（CPI），使用基线理想 CPI（理想缓存）等于 2，并且 cache miss 延迟为 10 个时钟周期。

Since load and store instructions are one third of all the execute instructions, CPI is calculated as (for 10

iterations):

CPI = 2 + 0.34×10 + 0.33×0.30×10 = 6.390 cycles, if X and Y are different arrays.

CPI = 2 + 0.34×10 + 0.33×0.15×10 = 5.895 cycles, if X and Y are the same array.

(c) 如果使用一个 32 字节的 I-cache，(a)的结果会如何改变？

If the size of the I-cache is 32 B, the entire loop fits in it. There will be only two cold misses in the first

iteration. Thus, the I-cache miss rate will be 2/59 = 0.033.

4 （20 分）. 一个具有 16 位地址的字节可寻址系统（A byte-addressable system with 16-bit addresses），配备了一个两路组相联、写回策略的缓存，并且具有完美的 LRU 替换功能。标签存储（tag array）（包括标签和其他所有元数据）总共需要 4352 位的存储空间。缓存的块大小是多少？假设 LRU 信息是按 set 维护，占 1 位。

We formulate two basic equations:

4352 = 2^index ∗ (2 ∗ (tag + dirty + valid) + LRU)　　　　　Eq. (1)

tag + index + offset = 16　　　　　Eq. (2)

There is one dirty bit and one valid bit per block, and one LRU bit per set. So, now the Eq. 1 looks like:

4352 = 2^index ∗ (2 ∗ (tag + 2) + 1) = (2^index ∗ (2*tag + 4)) + 2^index

By using the hint (4352 = 2^12 + 2^8), we get 2^index = 2^8, so index = 8, and

from 2^index ∗ (2*tag + 4) = 2^12, we get (2*tag + 4) = 2^4, so tag = 6.

By solving the Eq. 2, we get offset = 2, so, the block size is 2^2 = 4 bytes

5 （20 分）. 如图所示，考虑一个具有以下参数的缓存：关联度（associativity）= 2，块（block）大小= 2 个字（word），字大小= 32 位，缓存大小= 32K 个字，地址大小= 32 位。你只需要考虑字地址。

(a) 显示地址的标签（tag）、组（set）、块偏移（block offset）和字节偏移位（byte offset）。说明每个字段需要多少位。

Tag:16 bits

set:13 bits

block offset:1bits

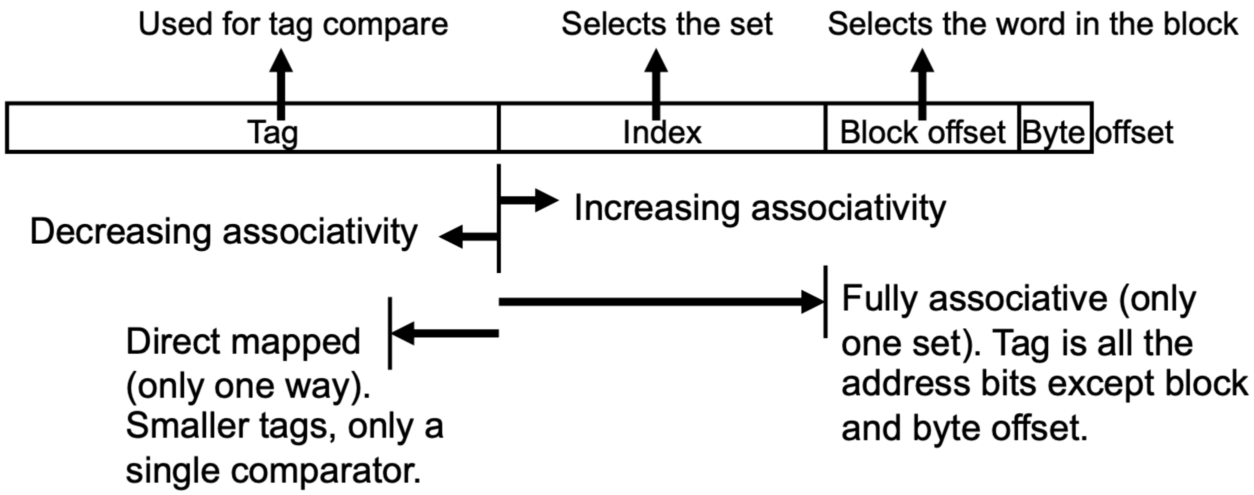
byte offset: 2bits

(b) 所有缓存标签（tag array）的大小是多少位？

256 kbits

(c) 假设每个缓存块还有一个有效位（V）和一个脏位（D）。包括数据、标签和状态位在内的每个缓存组的大小是多少？

164 bits


Used for tag compare
Selects the set
Selects the word in the block
Tag
Index
Block offset
Byte offset
Increasing associativity
Decreasing associativity
Direct mapped (only one way). Smaller tags, only a single comparator.
Fully associative (only one set). Tag is all the address bits except block and byte offset.