

به نام خدا

گروه 14

محمد بهرامی

نوید رئیسزاده

گزارش سوالات تمرین هشتم:

سوال اول:

**

**

**

$$A_{maT} = t_{cache} + MR_{cach} [t_{MM} + MR_{MM} (t_{VM})]$$

(سوال اول)

$$\bar{L} \rightarrow 1_{ns} + \underbrace{0.05}_{\text{miss rate}} \times \left[5 + \underbrace{0.15 \times 100}_{t_{main\ memory}} \right] = 2_{ns}$$

از لایه‌های بالای حافظه به ترتیب به سمت لایه‌های پایین‌تر پیش می‌رویم. ابتدا لایه ۱ و سپس در صورت نیاز

لایه ۲ نشود، لایه دوم حافظه cache یعنی L_۲ (نشود یعنی با miss مواجه کرد) و اگر لایه دوم نشود به مرحله

بعدی سلسله حافظه یعنی main memory پیش می‌رویم. (یعنی L_۲ با miss مواجه کرد)

سوال دوم:

Cache size: 32 bytes

Block size: 4 bytes

Associativity: 2-way set-associative

- **Capacity (C):**
 - number of data bytes in cache
- **Block size (b):**
 - bytes of data brought into cache at once
- **Number of blocks ($B = C/b$):**
 - number of blocks in cache: $B = C/b$
- **Degree of associativity (N):**
 - number of blocks in a set
- **Number of sets ($S = B/N$):**
 - each memory address maps to exactly one cache set

الف) تعداد کل بلوک های یک cache حاصل تقسیم ظرفیت بر اندازه هر بلوک است.

$$B = C/b \rightarrow B = 32 / 4 = 8.$$

ب) تعداد ست ها برابر با تعداد بلوک ها تقسیم بر درجه associativity که برای این cache برابر 2 است.

$$S = B/N \rightarrow S = 8 / 2 = 4.$$

پس 4 مجموعه (set) داریم که در هر کدام 2 بلوک است. در هر way یک بلوک.

ج) چون تعداد ست های ما 4 تا است پس به $\lg(4) = 2$ بیت index نیاز خواهیم داشت. دو بیت نیز offset هست که در نتیجه $32 - (2+2) = 28$ بیت برای tag باقی میماند.

د) در حلقه دوم که عملیات lw انجام میشود ما در هر 64 ایتريشن miss خواهیم داشت چون هر دفعه یک آدرس جدید آورده میشود و در cache قرار داده میشود و هیچوقت دوباره استفاده نمیشود و طبیعتاً miss داریم . و چون در هر way تنها یک بلوک جا دارد از spatial locality هم استفاده نمیشود .

100 % miss rate, 0% hit rate, 64 times.

Cache configuration:

Preset:

H

X

2^N Lines:

2

Repl. policy:

LRU

2^N Ways:

1

Wr. hit:

Write-back

2^N Words/Line:

0

Wr. miss:

Write allocate

Plot configuration:

Numerator

Hits

Denominator

Access count

☒ Ratio

☒ Moving avg.

50 cyc.

Statistics:

Size (bits):

504

?

Hit rate:

0

Writebacks:

64

Hits:

0

Misses:

128

```

1 addi s0, zero, 8 # s0 = SIZE = 8
2 addi t0, zero, 0 # t0 = i = 0
3 addi t1, zero, 0 # t1 = j = 0
4 addi t2, zero, 0 # t2 = sum = 0
5 add s1, zero, gp # s1 = head of array
6 for_i:
7     beq t0, s0, continue
8     addi t0, t0, 1 # i++
9     add t1, zero, zero # j = 0
10    for_j:
11        beq t1, s0, for_i|
12        add t3, t0, t1 # t3 = i+j
13        addi t4, t3, 1 # t4 = i+j+1
14        sw t4, 0(s1) # array[i][j] = i+j+1
15        addi s1, s1, 4 # به صورت خطی ذخیره میشه تو مموری در نهایت
16        addi t1, t1, 1 # j++
17        j for_j
18    j for_i
19 continue:
20    add s1, zero, gp # reset the pointer to head of array
21    addi t0, zero, 0 # reset i and j
22    addi t1, zero, 0
23    for_i_2:
24        beq t0, s0, end
25        addi t0, t0, 1 # i++
26        add t1, zero, zero # j = 0
27        for_j_2:
28            beq t1, s0, for_i_2
29            lw t5, 0(s1) # t5 = array[i][j]
30            addi s1, s1, 4 # مثل حلقه قبل
31            add t2, t2, t5 # sum = sum + t5
32            addi t1, t1, 1 # j++
33            j for_j_2
34        j for_i_2
35 end:
36    addi zero, zero, 0

```

سوال سوم :

(الف) طبق فرمول $B = C/b$ با ثابت موندن تعداد word ها و تغییر سایز بلوک ها تعداد بلوک ها تغییر میکنه در هر ست. در هر ست چهار word جا خواهد شد و تعداد بلاک ها دو خواهد بود و چون دایرکت مپ هست تعداد ست ها هم همین مقدار است.



در کد هیچ استفاده دوباره ای از یک دیتا نداریم و تنها spatial locality به ما کمک میکند. در هر ست هم یک ورد داریم که میس میخوره پس صد درصد میس ریت. تعداد رایت بک ها هم به اندازه نصف میس ها هست.

$$C = 8 \text{ word} = 32 \text{ bytes.}$$

$$\text{If } b = 4 \rightarrow B = 32/4 = 8 \text{ blocks.}$$

$$S = B/N = 8/1 = 8 \text{ Sets.}$$

Cache configuration:

Preset:  

2^N Lines:

3

Repl. policy:

LRU

2^N Ways:

0

Wr. hit:

Write-back

2^N Words/Line:

0

Wr. miss:

Write allocate

Plot configuration:

Numerator:

Hits

Denominator:

Access count

☒ Ratio


☒ Moving avg.

50 cyc.

Statistics:

Size (bits):

488



Hit rate:

0

Writebacks:

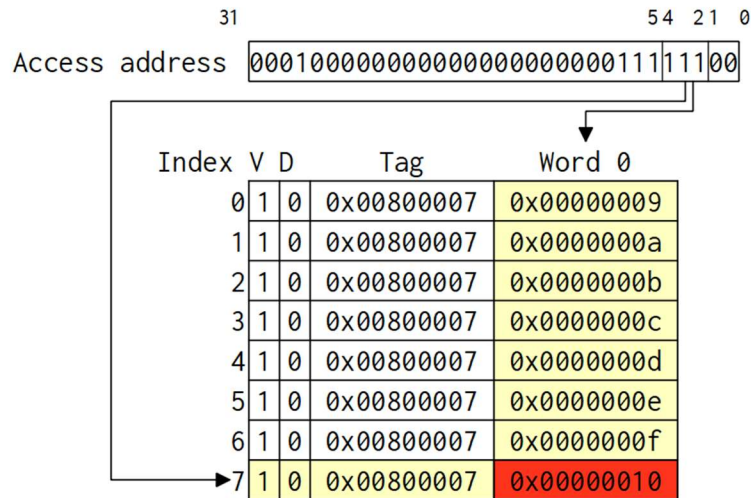
64

Hits:

0

Misses:



128







If $b = 8 \rightarrow B = 32/8 = 4$ blocks.



$S = B/N = 4/1 = 4$ Sets.

Cache configuration:

Preset:  

2^N Lines:   Repl. policy:

2^N Ways:   Wr. hit:



2^N Words/Line:   Wr. miss:

Plot configuration:


Numerator:

Denominator:

☒ Ratio

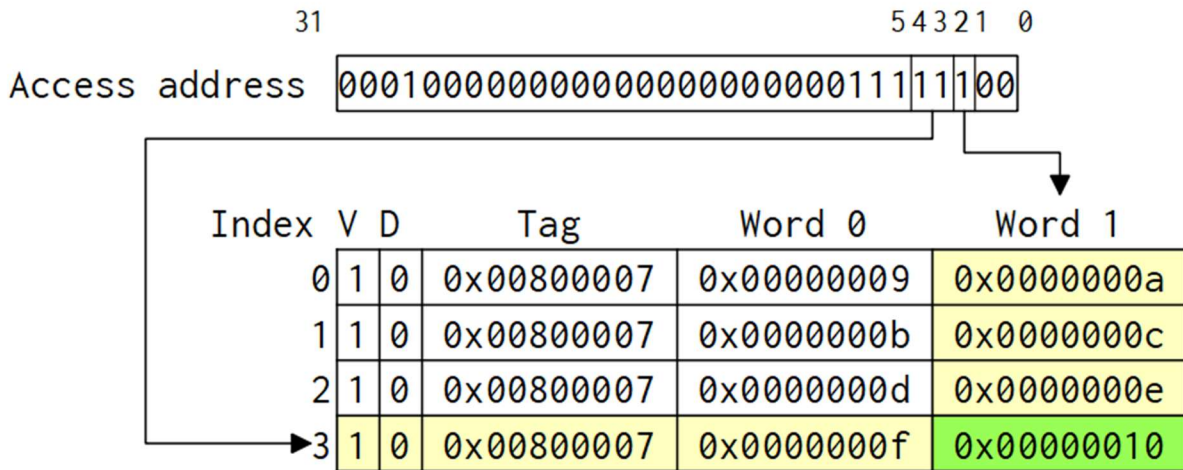
☒ Moving avg.  

Statistics:

Size (bits): 

Hit rate: Writebacks:

Hits: Misses:





با افزایش سایز بلوک در هر بار مقدار دیتای بیشتری آورده میشود و در اینجا دو word به یک ست اختصاص داده میشوند با یک تگ که به همین دلیل spatial locality داریم و دیتاهای نزدیک به هم در کش موجود خواهند بود و یک بار میس و یک بار هیت میخوریم . پس 50 درصد هیت داریم. و در نتیجه تعداد رایت بک ها هم نصف شده و به 32 می رسد.

If $b = 16 \rightarrow B = 32/16 = 2$ blocks.

$S = B/N = 2/1 = 2$ sets.

Cache configuration:

Preset:  

2^N Lines: 1 Repl. policy: LRU

2^N Ways: 0 Wr. hit: Write-back

2^N Words/Line: 2 Wr. miss: Write allocate

Plot configuration:


Numerator: Hits

Denominator: Access count

☒ Ratio

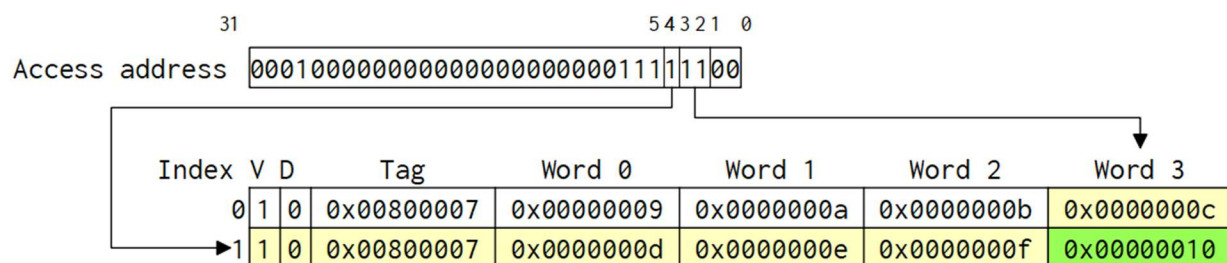
☒ Moving avg. 50 cyc.

Statistics:

Size (bits): 314 

Hit rate: 0.75 Writebacks: 16

Hits: 96 Misses: 32



در این حالت تعداد ست ها دو تا خواهد بود که در نتیجه چون $N=1$ است هر ست چهار ورد خواهد داشت. و به ازای هر میس سه هیت خواهیم داشت که هیت ریت 75 درصد رو به ما میده. باز هم تعداد رایت بک ها نصف میشود نسبت به دفعه قبل (متناسب با میس ریت هست که در حال کاهش) و به 16 میرسه.

(ب)

$C = 16 \text{ word} = 64 \text{ bytes}$. $b = 4$, $B = C/b = 64 / 4 = 16 \text{ blocks}$.

If $N = 2 \rightarrow S = B / N = 16 / 2 = 8$.

چون در هر way تنها یک ورد هست ، پس صد درصد میس ریت داریم. در نتیجه به میزان نصف میس ها رایت بک نیازه یعنی 64 تا.

Cache configuration:

Preset: H X

2^N Lines: 3 Repl. policy: LRU

2^N Ways: 1 Wr. hit: Write-back

2^N Words/Line: 0 Wr. miss: Write allocate

Plot configuration:

Numerator: Hits

Denominator: Access count

☒ Ratio

☒ Moving avg. 50 cyc.

Statistics:

Size (bits): 992 ?

Hit rate: 0 Writebacks: 64

Hits: 0 Misses: 128



Access address 000100000000000000000000000011111100

31 5 4 2 1 0

Index	V	D	LRU	Tag	Word 0
0	1	0	1	0x00800006	0x00000008
	1	0	0	0x00800007	0x00000009
1	1	0	1	0x00800006	0x00000009
	1	0	0	0x00800007	0x0000000a
2	1	0	1	0x00800006	0x0000000a
	1	0	0	0x00800007	0x0000000b
3	1	0	1	0x00800006	0x0000000b
	1	0	0	0x00800007	0x0000000c
4	1	0	1	0x00800006	0x0000000c
	1	0	0	0x00800007	0x0000000d
5	1	0	1	0x00800006	0x0000000d
	1	0	0	0x00800007	0x0000000e
6	1	0	1	0x00800006	0x0000000e
	1	0	0	0x00800007	0x0000000f
7	1	0	1	0x00800006	0x0000000f
	1	0	0	0x00800007	0x00000010

If $N = 4 \rightarrow S = B / N = 16 / 4 = 4$.

Cache configuration:

Preset:  

2^N Lines: 2 Repl. policy: LRU

2^N Ways: 2 Wr. hit: Write-back

2^N Words/Line: 0 Wr. miss: Write allocate

Plot configuration:


Numerator: Hits

Denominator: Access count

☒ Ratio

☒ Moving avg. 50 cyc.

Statistics:

Size (bits): 1024 

Hit rate: 0 Writebacks: 64

Hits: 0 Misses: 128

Access address 000100000000000000000000000011111100

Index	V	D	LRU	Tag	Word 0
0	1	0	3	0x0100000c	0x00000008
	1	0	2	0x0100000d	0x0000000c
	1	0	1	0x0100000e	0x00000009
	1	0	0	0x0100000f	0x0000000d
1	1	0	3	0x0100000c	0x00000009
	1	0	2	0x0100000d	0x0000000d
	1	0	1	0x0100000e	0x0000000a
	1	0	0	0x0100000f	0x0000000e
2	1	0	3	0x0100000c	0x0000000a
	1	0	2	0x0100000d	0x0000000e
	1	0	1	0x0100000e	0x0000000b
	1	0	0	0x0100000f	0x0000000f
3	1	0	3	0x0100000c	0x0000000b
	1	0	2	0x0100000d	0x0000000f
	1	0	1	0x0100000e	0x0000000c
	1	0	0	0x0100000f	0x00000010

باز هم چون تعداد بلاک ها و حجم ثابت افزایش درجه شرکت پذیری فرقی ایجاد نمیکنه و باز صد درصد میس

باز هم با زیاد کردن درجه شرکت پذیری و به 8way تبدیل شدن کش تغییری در تعداد ورد های یک بلاک نمیکند و صد درصد میس داریم.

از مقایسه نتایج دو بخش الف و ب به این نتیجه میرسیم که هنگامی که در کد از مقادیر دوباره استفاده نمیشود و تنها از مقادیر نزدیک داره استفاده میشه یعنی time locality نداریم و تنها spatial locality داریم اضافه کردن به درجه شرکت پذیری میس ریت رو تغییر نمیده و برای افزایش هیت ریت باید تعداد ورد های هر ست را زیاد کرد که با افزایش سایز بلوک این کار ممکن است.

سوال چهارم) بنام TLB با فرض اینکه آخرین خانه داده شده در جدول ذخیره می‌کند
fully associative cache \Rightarrow hit rate = 99% \Rightarrow data table

page \Rightarrow 4 kb main memory \Rightarrow 1 mb mis \Rightarrow page fault

$$a) AMAT = \frac{t_{cache}}{miss\ rate\ cache} + \left(\frac{t_{mm}}{miss\ rate\ main\ memory} + miss\ rate\ main\ memory \times (t_{vm}) \right)$$

$$\rightarrow 100 + \left(1 + \frac{0.01}{0.99} \right) = \left(100 + \frac{0.000001}{0.999999} \right) \left(\frac{1000000}{1000000} \right) = 101.01\ cycle$$

TLB بعد از TLB \Rightarrow همان عبارت بالا: $t_{TLB} + MR_{TLB} (t_{mm}) = 101.01 + 1 + 100 \times 0.000001 =$

b. TLB \Rightarrow 44 \Rightarrow 4, 11 cycle

\rightarrow valid bit + tag bit + physical page number

tag bit \Rightarrow vpn = 2. \rightarrow با صفر یا یک 11

هر بیت در جدول = 11 + 20 = 32 بیت

size TLB = 44 \times 32 = 1408 بیت در TLB

در ساختار TLB از آخرین خانه داده شده در جدول می‌گیریم.

این چیزی چهارم هست

valid tag 20 بیت (برای هر بیت) 11 است

SRam مورد نیاز = 1408 \times 1 = 1408 SRam