


5.12



南开大学
Nankai University

5.12

5.12.1

1) 先计算缺失代价 = $\frac{10/115}{\frac{1}{26412}} = 200$ (时钟周期)

所以只有 L1 下的 CPI 为 = $\frac{1.51 + 0.07 \times 200}{1} = 15.5$

2) 直接映射 L2 的 CPI = $\frac{1.51 + 0.07 \times 12 \times 1 + 0.07 \times 0.035 \times 200}{1} = 2.83$

3) 8 路组相联的二级 cache = $\frac{1.51 + 0.07 \times 28 \times 1 + 0.07 \times 0.015 \times 200}{1} = 3.67$

代价加倍时:

1) 只有 L1 CPI = $\frac{1.51 + 0.07 \times 400}{1} = 29.5$ $\left(\frac{29.5}{15.5} - 1 \right) \times 100\% = 90.3\%$

2) 直接映射 L2 = $\frac{1.51 + 0.07 \times 12 \times 1 + 0.07 \times 0.035 \times 400}{1} = 3.32$ $\left(\frac{3.32}{2.83} - 1 \right) \times 100\% = 17.3\%$

3) 8 路组相联 = $\frac{1.51 + 0.07 \times 28 \times 1 + 0.07 \times 0.015 \times 400}{1} = 3.88$ $\left(\frac{3.88}{3.67} - 1 \right) \times 100\% = 5.7\%$

5.12.2

计算 CPI

$$CPI = \frac{1.51 + 0.07 \times 121 + 0.07 \times 0.035 \times 50 + 0.07 \times 0.035 \times 0.13 \times 200}{1}$$

$$= 1.03$$

我们可以看出增加 L2 之后确实降低了 CPI, 也就是说它会让程序运行得更快。但多加一个 cache, 也会增加很多代价和成本。

5.12.3

要使 CPI 低于 2.83, 有

$$1.51 + 0.07 \times 50 + 0.07 \times (0.04 - 0.007x) \times 200 < 2.83$$

我们可以看出, 要想满足不等式, $(0.04 - 0.007x)$ 必须是负数, 这显然是不对的, 所以不论 cache 的容量为多大, 它都不能匹配直接相联的。

5.14

我们知道,校验位和数据位有如下关系:

$$2^p \geq p + d + 1 \Rightarrow p \geq \log(p + d + 1)$$

所以当 $d=128$ 时 $2^p \geq p + 128 + 1$

因为 $2^7 = 128$ $2^8 = 256$ 所以 $p=8$

5.14.2.

64位用8位校验码: 开销 = $\frac{8}{64} \times 100\% = 12.5\%$
性能 = $\frac{1}{12} \times 100\% = 8.3\%$ } 性价比 = $\frac{12.5}{8.3} = 1.5$

128位用9位校验码: 开销 = $\frac{9}{128} \times 100\% = 7\%$
性能 = $\frac{1}{137} \times 100\% = 0.73\%$ } 性价比 = $\frac{7}{0.73} = 9.6$

5.14.3.

十六进制: 0x375

= 进制: 0011 0110 101

P_1 校验的序列为: 0 1 0 1 0 0 (没错)

P_2 校验的序列为: 0 1 1 1 1 0 (没错)

P_4 校验的序列为: 1 0 1 1 1 (没错)

P_8 校验的序列为: 1 0 1 0 1 (有错)

我们可以看到第八位错了, 修改后 0x365.

5.16

5.16

9.16-1. 因为页大小是 4096 字节, 所以需用 12 位表示偏移, 所以十六进制第一位就是标记位

① 0x123d.

标记位: 1 TLB 缺失, 页表命中, 产生页面故障

有效位	标记	物理页号	最近一次访问
1	6	12	5
1	7	4	2
1	3	6	4
1	1	13	1

② 0x08b3

标记位: 0 TLB 缺失, 页表命中, 产生页面故障

有效位	标记	物理页号	最近一次访问
1	0	5	1
1	7	4	3
1	3	6	5
1	1	13	2

③ 0x305c

标记位: 3 TLB 命中

有效位	标记	物理页号	最近一次访问
1	0	5	2
1	7	4	4
1	3	6	1
1	1	13	3



南开大学
Nankai University

④ 0x871b

标记位=8 TLB 命中, 页表命中, 产页面故障

有效位	标记	物理页号	最近一次访问
1	0	5	2
1	8	14	1
1	3	6	2
1	1	13	3

⑤ 0xbec6

标记位=b TLB 命中, 页表命中

有效位	标记	物理页号	最近一次访问
1	0	5	3
1	8	14	2
1	3	6	3
1	b	12	1

⑥ 0x3140

标记位=3 TLB 命中

有效位	标记	物理页号	最近一次访问
1	0	5	3
1	8	14	2
1	3	6	1
1	b	12	2

⑦ 0xc049

标记位=c TLB 命中, 页表命中, 产页面故障

有效位	标记	物理页号	最近一次访问
1	c	15	1
1	8	14	3
1	3	6	2
1	b	12	3

5.11-2

页大小变成 $1KB = 2^{10}$ 个字节.

① $0x123d$ 标记位: 0. TLB 缺失. 页表命中

有效位	标记	物理页号	最近访问
1	11	12	5
1	7	4	2
1	3	6	4
1	0	5	1

② $0x08b3$ 标记位: 0. TLB 命中

有效位	标记	物理页号	最近访问
1	11	12	6
1	7	4	3
1	3	6	5
1	0	5	2

③ $0x3f5c$ 标记位: 0 TLB 命中

有效位	标记	物理页号	最近访问
1	11	12	7
1	7	4	4
1	3	6	6
1	0	5	3

④ $0x871b$ 标记位: 2 TLB 缺失. 页表中产生新故障

有效位	标记	物理页号	最近访问
1	2	13	1
1	7	4	5
1	3	6	7
1	0	5	4

⑤ $0xb0e6$ 标记位: 2. TLB 命中

有效位	标记	物理页号	最近访问
1	2	13	2
1	7	4	6
1	3	6	8
1	0	5	5

⑥ $0x3190$ 标记位: 0 TLB 命中

有效位	标记	物理页号	最近访问
1	2	13	3
1	7	4	7
1	3	6	9
1	0	5	1

⑦ $0xc049$ 标记位: 3 TLB 命中

有效位	标记	物理页号	最近访问
1	2	13	4
1	7	4	8
1	3	6	10
1	0	5	7



南开大学

Nankai University

5. 16-3

最开始的 TLB.

有效位	标记	物理页号	索引	最近访问
1	11	12	0	4
1	7	4	1	1
1	3	6	0	3
0	4	9	1	7

标记	索引	字节偏移
3	1	12

虚拟页

① 0x23d 标记位=0 索引=1 TLB缺失, 页表命中

有效位	标记	物理页号	索引	最近访问
1	11	12	0	5
1	7	4	1	2
1	3	6	0	4
1	0	13	1	1

② 0x08b3 标记位=0 索引=0 TLB缺失, 页表命中

有效位	标记	物理页号	索引	最近访问
1	0	5	0	6
1	7	4	1	3
1	3	6	0	5
1	0	13	1	2

③ 0x365c 标记位=1 索引=1 TLB缺失, 页表命中, 产生页面故障

有效位	标记	物理页号	索引	最近访问
1	0	5	0	2
1	1	6	1	1
1	3	6	0	6
1	0	13	1	3

④ 0x871b 标记位=4 索引=0 TLB缺失, 页表命中, 产生页面故障

有效位	标记	物理页号	索引	最近访问
1	0	5	0	3
1	1	6	1	2
1	4	14	0	1
1	0	13	1	4

⑥ 0xbce6 标记=5 索引=1 TLB 缺失, 页命中

有效位	标记	物理页	索引	最近访问
1	0	5	0	4
1	1	6	1	3
1	4	14	0	2
1	5	12	1	1

⑦ 0x3140 标记=1 索引=1 TLB 命中

有效位	标记	物理页	索引	最近访问
1	0	5	0	5
1	1	6	1	1
1	4	14	0	3
1	5	12	1	2

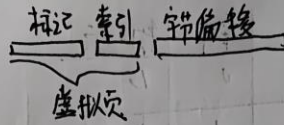
⑧ 0xc049 标记=6 索引=0 TLB 命中, 页缺失, 产生页故障

有效位	标记	物理页	索引	最近访问
1	6	15	0	1
1	1	6	1	1
1	4	14	0	3
1	5	12	1	2



5.16.4

直接映射 \Rightarrow 索引变2位



① 0x123d 标记: 0 索引: 1 TLB缺失, 页表命中, 产生页码故障

有效位	标记	索引	物理页
1	11	0	12
01	0	1	13
1	3	2	6
0	4	3	9

③ 0x871b 标记: 2 索引: 0 TLB缺失, 页表命中, 产生页码故障

有效位	标记	索引	物理页
1	2	0	14
1	0	1	13
1	3	2	6
1	0	3	6

② 0x08b3 标记: 0 索引: 0 TLB缺失, 页表命中

有效位	标记	索引	物理页
11	0	0	5
1	0	1	13
1	3	2	6
0	4	3	9

⑤ 0xbec6 标记: 2 索引: 3 TLB缺失, 页表命中

有效位	标记	索引	物理页
1	2	0	14
1	0	1	13
1	3	2	6
1	2	3	12

④ 0x365c 标记: 0 索引: 3 TLB缺失, 页表命中

有效位	标记	索引	物理页
1	0	0	5
1	0	1	13
1	3	2	6
1	0	3	6

⑥ 0x3140 标记: 0 索引: 3 TLB缺失, 页表命中

有效位	标记	索引	物理页
1	2	0	14
1	0	1	13
1	3	2	6
1	0	3	6

①

0xc049 标记:3 索引:0 TLB缺失页表缺失,产生页面故障

4.6.2

有效位	标记	索引	物理页
1	3	0	15
1	0	1	13
1	3	2	6
1	3	3	6

5.16.5

如果没有TLB的话, CPU每次处理一个虚拟地址的时候都会两次访问存储器
但有了TLB, 因为TLB在CPU里面, 所以就只需要访问一次存储器就可以。
如果没有TLB, 每次访问一个地址, 需要从页表中找到相应的物理地址, 然
后再去访问存储器。



5.17

5.17.1

每页表项数为 $= \frac{2^{32}}{2^{13}} = 2^{19}$

每个表项大小为 $= 2^{19} \times 4$

所以总大小为 $= 5 \times 4 \times 2^{19} = 10MB$

5.17.2



要把 2^{19} 个表项分成 256 份, 所以每份有 $\frac{2^{19}}{2^8} = 2^{11}$ 个表项, 所以每份大小为 $2048 \times 4 = 8KB$, 因为页大小为 8KB, 所以每份能覆盖的空间为 $2048 \times 8KB = 16MB = 2^{24}B$.
① 页表所需最小内存为 刚好能满足要求.

计算一共有多少个二级页表: $\frac{2^{24}B}{2^{20}B} = 2^4$ 个
所以主表就有 128 个表项, 所以总容量为 =
 $5 \times 128 \times 6 + 5 \times 128 \times 8KB = (5 \times 2^{20} + 3840) bytes$

② 最大内存就是一级页表全使用的时候:

总大小为 $= 5 \times 256 \times 6 + 5 \times 256 \times 8KB = (10 \times 2^{20} + 7680) bytes$

5.17.3

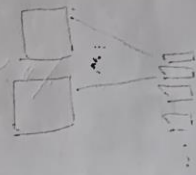
由 5.17.1 可知, 该页的偏移位有 13 位, 而对于 cache 来说, 一共有 $\frac{16KB}{16B} = 1024$ 块, 所以有 10 个索引位和 4 个偏移位, 这大于页偏移, 设计者可以使用组相联, 这会使索引位减少, 从而跟页偏移相同.

5.20

0组 = =

5.20.1

0(缺失) 1(缺失) 2(缺失) 3(缺失) 4(缺失)
 5(缺失) 6(缺失) 7(缺失) 0(缺失) 1(缺失) 2(缺失) 3(缺失)
 4(缺失) 5(缺失) 6(缺失) 7(缺失) 0(缺失)



5.20.2

0(缺失) 1(缺失) 2(缺失) 3(缺失) 4(缺失) 5(缺失)
 6(缺失) 7(缺失) 0(命中) 1(命中) 2(缺失) 3(缺失)
 4(缺失) 5(缺失) 6(命中) 7(命中) 0(缺失)

5.20.3

0(缺失) 1(缺失) 2(缺失) 3(缺失) ①掷硬币: 假设是正面 4→0
 4(缺失) ②掷硬币: 假设是反面 5→3 5(缺失) ③掷硬币: 假设是正面
 6→4 6(缺失) ④掷硬币: 假设是反面 7→5 7(缺失) ⑤掷硬币: 假
 设是正面 0→6 0(缺失) 1(命中) 2(命中) ⑥掷硬币: 假设是反面 3→1
 3(缺失) ⑦掷硬币: 假设是正面 4→0 4(缺失) ⑧掷硬币: 假设是反面 5→3
 5(缺失) ⑨掷硬币: 假设是正面 6→2 6(缺失) 7(命中) 0(缺失) 反面 0→6



5.20.4

我们可以知道, 该序列最优最多能命中四个块, 所以由 5.20.2 可知 MRU 是最优的一种策略。

0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0
缺失 缺失 缺失 缺失 缺失 缺失 缺失 命中 命中 缺失 缺失 缺失 缺失 命中 命中 缺失

5.20.5

在我看来, 最大的难点就是我们不能像做题一样站在上帝视角知道下一个要访问的地址序列是多少, 对于不同的地址序列都有它独特的最优策略, 但我们并不能找出对所有地址序列最优的一种策略。

5.20.6

如果可以选择该地址是否缓存, 那可以在发生冲突时选择不缓存或缓存特定的块来于是高命中率或降低命中率。

5.21

5.21.1

① 因为没有 I/O 所以 $CP1 = \frac{1.51 + \frac{I}{10000} \times 120 \times (15 + 175) \cdot I}{I}$

$$= 3.78$$

$$② CP1 = \frac{1.51 + \frac{I}{10000} \times 120 \times (15 + 350) \cdot I}{I} = 5.9$$

$$③ CP1 = \frac{1.51 + \frac{I}{10000} \times 120 \times (15 + \frac{175}{2}) \cdot I}{I} = 2.7$$

设最大开销为 X ，由题意知 $(\frac{w_{cp1}}{1.5} - 1) \times 100\% < 10\%$
 $\therefore w_{cp1} < 1.65$

$$w_{cp1} = 1.5 + \frac{.120}{10000} \times (15 + X) < 1.65$$

计算得 $X < 14$ 个时钟周期

5.21.2

$$① CP1 = \frac{1.51 + \frac{I}{10000} \times 120 \times 15 + \frac{I}{10000} \times 30 \times (1100 + 175) \cdot I}{I} = 4.98$$

$$② CP1 = \frac{1.51 + \frac{I}{10000} \times 120 \times (15 + 175) \cdot I + \frac{I}{10000} \times 30 \times (1100 + 175) \cdot I}{I} = 7.60$$

$$③ CP1 = \frac{1.51 + \frac{I}{10000} \times 120 \times 15 + \frac{I}{10000} \times 30 \times (1100)}{I} = 3.3$$

$$④ CP1 = \frac{1.51 + \frac{I}{10000} \times (20) \times (15 + 175) + \frac{I}{10000} \times 15 \times (120)}{I} = 5.7$$



5.29

5.29.1

影子页表：在创建进程的时候，虚拟机创建相关页表，并生成影子页表，在TLB缺失时不做任何改变，在发生缺页时，创建新的映射，更新影子页表。在上下文切换时，使进程的TLB失效。

嵌套页表：在刚创建进程时，虚拟机创建页表，并生成从物理地址到机器地址的映射，在TLB缺失时，虚拟机将虚拟地址转换成机器地址。产生缺页时虚拟机则增加新的映射，使过时的TLB失效。在上下文切换时，使进程的TLB失效。

5.29.2

本地页表 = 4

嵌套页表 = 24

5.29.3

从表中可知影子页面故障代价是最大的，所以对影子页面来说，缺页率更为重要。而对于嵌套页面来说它发生TLB缺失的代价是最大的，因为要遍历两张表，所以TLB缺失率更为重要。

5.29.4

$$CPI_{\text{影子}} = \frac{1 + \frac{7}{1000} \times 0.101 \times 30000}{1} \approx 1.03$$

$$CPI_{\text{嵌套}} = \frac{1 + \frac{7}{1000} \times 0.2 \times 200}{2} = 1.04$$

0.29.5

从上面题可以知道,我们可以通过将低缺页率来减少影子页表带来的开销,具体来讲,可以通过合并多页表,从而使缺页率降低。

5.29.6

从上面题可以知道嵌套页表的开销主要在TLB缺失上,所以我们可以适当增加TLB的容量或者引入像多级cache这样的多级TLB结构,从而使TLB命中率增大。