《计算机组成原理》第九次作业

信息安全 胡博浩 2212998

5.12

1)

标准存储时间： 在 2GHz 机器上，每个周期需要 0.5 ps。那么我们主存访问需要 100/0.5 = 200 个周期。

1. 只有L1 cache：CPI = 1.5 + 0.07200 = 15.5
2. 一个直接映射的L2 cache：CPI = 1.5 + .07 × (12 + 0.035 × 200) = 2.83
3. 一个八路组相联的L2 cache：CPI = 1.5 + .07 × (28 + 0.015 × 200) = 3.67

如果主存访问时间加倍，即主存访问需要 400 个周期

1. 只有L1 cache：CPI = 1.5 + 0.07*400 = 29.5，增加 90%
2. 一个直接映射的L2 cache：CPI = 1.5 + .07 × (12 + 0.035 × 400) = 3.32，增加 17%
3. 一个八路组相联的L2 cache：CPI = 1.5 + .07 × (28 + 0.015 × 400) = 3.88，增加 5%

2)

原来是1.5，添加一个L3 cache 后变为0.07 × (12 + 0.035 × (50 + 0.13 × 200)) = 1.03。1.5>1.03，所以，增加 L3 cache 确实缩短了整体的主存访问时间，这也是有 L3 cache 的主要优势。缺点是 L3 cache 占用了其他类型资源（如功能单元）的空间。

3)

不管cache多大都无法实现性能目标。原因如下：

我们希望带有外部二级 cache 的 CPU的 CPI 最多为 2.83。设 x 为缺失率。

1.5 + 0.07*(50 + x*200) < 2.83

解得 x < - 0.155。这意味着即使二级缓存的缺失率为 0 ，50 毫秒的访问时间也会产生 1.5 + 0.07*(50 + 0*200) = 5 的 CPI，这大于片上二级缓存给出的 2.83。因此，任何大小的缓存都无法达到性能目标。

5.14

1)

9。对于 SEC，我们需要找出最小 p，使得 2^p >= p + d + 1，然后再加上一位。因此p = 8。然后，我们还需要为 SEC/DED 增加一位。故最少需要9位奇偶位。

2)

本章中描述的 (72,64) 代码需要 8/64 = 12.5% 额外的bits的开销，以容忍 72 bits内任何一个bit的丢失，提供 1.4% 保护率的性能。

而5.14.1部分的 (137,128) 代码需要 9/128 = 7.0% 的额外bits的开销，才能容许 137 bits内任何一个bit的丢失，提供 0.73% 保护率的性能。

两种代码的性价比如下：

1. (72,64) 代码：12.5/1.4 = 8.9
2. (136,128) 代码：7.0/0.73 = 9.6

则(72,64) 代码的性价比更高。

3)

使用第 5.5 节中的位编号，第 8 位是错误的，因此值将纠正为 0x365。

5.16

1)

Address	Virtual Page	TLB H/M	TLB		
			Valid	Tag	Physical Page
4669 0x123d	1	TLB miss PT hit PF	1	b	12
			1	7	4
			1	3	6
			1 (last access 0)	1	13
2227 0x08b	0	TLB miss PT hit	1 (last access 1)	0	5
			1	7	4
			1	3	6

			1 (last access 0)	1	13
13916 0x365c	3	TLB hit PT hit	1 (last access 1)	0	5
			1	7	4
			1 (last access 2)	3	6
			1 (last access 0)	1	13
34587 0x871b	8	TLB miss PT hit PF	1 (last access 1)	0	5
			1 (last access 3)	8	14
			1 (last access 2)	3	6
			1 (last access 0)	1	13
48870 0xbee6	b	TLB miss PT hit	1 (last access 1)	0	5
			1 (last access 3)	8	14
			1 (last access 2)	3	6
			1 (last access 4)	11	12
12608 0x3140	3	TLB hit PT hit	1 (last access 1)	0	5
			1 (last access 3)	8	14
			1 (last access 5)	3	6
			1 (last access 4)	b	12

49225 0xc040	c	TLB miss PT miss PF	1 (last access 6)	c	15
			1 (last access 3)	8	14
			1 (last access 5)	3	6
			1 (last access 4)	b	12

2)

Address	Virtual Page	TLBH/M	TLB		
			Valid	Tag	Physical Page
4669 0x123d	1	TLB miss PT hit	1	11	12
			1	7	4
			1	3	6
			1 (last access 0)	0	5
2227 0x08b3	0	TLB miss PT hit	1	11	12
			1	7	4
			1	3	6
			1 (last access 1)	0	5
13916 0x365c	3	TLB hit PT hit	1	11	12
			1	7	4
			1	3	6
			1 (last access 2)	0	5
34587 0x87lb	8	TLB miss PT hit	1 (last access 3)	2	13

		PF	1	7	4
			1	3	6
			2	0	5
48870 0xbee6	11	TLB miss PT hit	1 (last access 4)	2	13
			1	7	4
			1	3	6
			1 (last access 2)	0	5
12608 0x3140	3	TLB hit PT hit	1 (last access 4)	2	13
			1	7	4
			1	3	6
			5	0	5
49225 0xc040	12	TLB miss PT hit	1 (last access 4)	2	13
			1	7	4
			1 (last access 6)	3	6
			1 (last access 5)	0	5

页大小越大，TLB 缺失率越低，但会导致数据越分散即碎片率越高、物理内存利用率越低。

3)

使用4KiB的页和一个两路组相联的TLB

Address	Virtual Page	Tag	Index	TLB H/M	TLB			
					Valid	Tag	Physical Page	Index
	1	0	1		1	b	12	0

4669 0x123d				TLB miss PT hit PF	1	7	4	1
					1	3	6	0
					1 (last access 0)	0	13	1
2227 0x08b3	0	0	0	TLB miss PT hit	1 (last access 1)	0	5	0
					1	7	4	1
					1	3	6	0
					1 (last access 0)	0	13	1
13916 0x365c	3	1	1	TLB miss PT hit	1 (last access 1)	0	5	0
					1 (last access 2)	1	6	1
					1	3	6	0
					1 (last access 0)	1	13	1
34587 0x871b	8	4	0	TLB miss PT hit PF	1 (last access 1)	0	5	0
					1 (last access 2)	1	6	1
					1 (last access 3)	4	14	0
					1 (last access 0)	1	13	1
48870 0xbee6	b	5	1	TLB miss PT hit	1 (last access 1)	0	5	0
					1 (last access 2)	1	6	1
					1 (last access 3)	4	14	0

				1 (last access 4)	5	12	1
12608 0x3140	3	1	1	TLB hit PT hit	1 (last access 1)	0	5	0
					1 (last access 5)	1	6	1
					1 (last access 3)	4	14	0
					1 (last access 4)	5	12	1
49225 0xc049	c	6	0	TLB miss PT miss PF	1 (last access 6)	6	15	0
					1 (last access 5)	1	6	1
					1 (last access 3)	4	14	0
					1 (last access 4)	5	12	1

4)

使用4KiB的页和一个直接映射的TLB

Address	Virtual Page	Tag	Index	TLB H/M	TLB			
					Valid	Tag	Physical Page	Index
4669 0x123d	1	0	1	TLB miss PT hit PF	1	b	12	0
					1	0	13	1
					1	3	6	2
					0	4	9	3
2227 0x08b3	0	0	0	TLB miss PT hit	1	0	5	0
					1	0	13	1

					1	3	6	2
					0	4	9	3
13916 0x365c	3	0	3	TLB miss PT hit	1	0	5	0
					1	0	13	1
					1	3	6	2
					1	0	6	3
34587 0x871b	8	2	0	TLB miss PT hit PF	1	2	14	0
					1	0	13	1
					1	3	6	2
					1	0	6	3
48870 0xbee6	b	2	3	TLB miss PT hit	1	2	14	0
					1	0	13	1
					1	3	6	2
					1	2	6	3
12608 0x3140	3	0	3	TLB hit PT hit	1	2	14	0
					1	0	13	1
					1	3	6	2
					1	0	6	3
49225 0xc049	c	3	0	TLB miss PT miss PF	1	3	15	0
					1	0	13	1
					1	3	6	2
					1	0	6	3

5)

如果没有 TLB，几乎每次内存访问都需要对 RAM 进行两次访问： 首先访问页表，然后访问请求的数据。

5.17

1)

标记位大小为 32-log2(8192) = 32-13 = 19 位。总共有五个页表，需要 5 × (2^19 × 4) 字节 = 10 MB。

2)

在两级页表中，2^19 页表项被按需分配为 256 段。每个二级页表包含 2^(19 - 8) = 2048 项，每个项需要 2048 × 4 = 8 KB，则一共需要了 2048 × 8 KB = 16 MB (2^24) 的虚拟地址空间。

如果 "一半内存 "指的是 2^31 字节，那么二级表所需的最小内存量为 5 × (2^31/2^24)*8 KB = 5 MB。一级表需要额外的 5 × 128 × 6 字节 = 3840 字节。

如果所有第一级数据段都被使用，则需要在每个应用程序中使用全部 256 个数据段。这样，二级表需要 5 × 256 × 8KB = 10 MB，一级表需要 7680 字节。

3)

页面索引为 13 位（地址位 12 到 0）。

每个块有两个 64 位字即每个块有 16 字节，则一个 16 KB 的直接映射cache有 16 KB/16 字节 = 1024 个块。因此，它将有 10 个索引位和 4 个偏移位，超过了页面索引的位数。

设计者可以增加cache的关联性。这样可以减少索引位的数量，使cache的索引完全位于页面索引内。

5.20

1)

没有访问命中

2)

如图所示：

0	1	2	3	4	5	6	7	0	1	2	3	4	5
M	M	M	M	M	M	M	M	H	H	M	M	M	M

3)

每次掷硬币结果不一样，答案不固定

4)

MRU是最优策略，结果和第二问的相同

5)

驱逐的最佳区块是将来会造成最少丢失的区块。但是cache无法预知未来，我们只能做好预测。

6)

如果知道某个地址的时间位置有限并且会与缓存中的另一个块发生冲突，那么选择不缓存该地址就可以改善缺失率。反之，如果选择缓存的地址不当，则可能会使缺失率更糟。

5.21

1)

1. 没有I/O访问：CPI = 1.5 + 120/10000 × (15 + 175) = 3.78
2. 如果 VMM 开销加倍：CPI = 1.5 + 120/10000 × (15 + 350) = 5.88
3. 如果 VMM 开销减半：CPI = 1.5 + 120/10000 × (15 + 87.5) = 2.73
4. 在本地硬件上运行的机器 CPI 为 1.5 + 120/10000*15 = 1.68。要将性能下降控制在 10%，即满足 1.5 + 120/10000*(15 + x) < 1.1*1.68，解得陷入VMM的最大开销为14个周期。

2)

1. 非虚拟化：CPI = 1.5 + 120/10000 × 15 + 30/10000 × 1100 = 4.98
2. 虚拟化：CPI = 1.5 + 120/10000 × (15 + 175) + 30/10000 × (1100 + 175) = 7.60
3. I/O访问时间减半的非虚拟化：CPI = 1.5 + 120/10000 × 15 + 15/10000 × 1100 = 3.33
4. I/O访问时间减半的虚拟化：CPI = 1.5 + 120/10000 × (15 + 175) + 15/10000 × (1100 + 175) = 5.69.

5.29

1)

1. 影子页表：(1) 虚拟机创建页表，管理程序更新影子表；(2) 无；(3) 管理程序拦截缺页，创建新映射，并使 TLB 中的旧映射失效；(4) 虚拟机通知管理程序使进程的 TLB 条目失效。
2. 嵌套页表：(1) 虚拟机创建新页表，管理程序将 PA 中的新映射添加到 MA 表中。(2) 硬件同时读取两个页表，将 VA 转换为 MA；(3) 虚拟机和管理程序更新各自的页表，管理程序使过时的 TLB 条目失效；(4) 与影子页表相同，虚拟机通知管理程序使进程的 TLB 条目失效。

2)

1. 本地页表：4；NPT：24（管理员可更改页表的级别）
2. 嵌套页表： L；NPT： $L \times (L + 2)$。

3)

1. 影子页表：缺页率
2. 嵌套页表：TLB 缺失率

4)

1. 影子页表：$CPI = 1 + 0.001/1000 \times 30000 = 1.03$
2. 嵌套页表：$CPI = 1 + 0.2/1000 \times 200 = 1.04$

5)

合并多个页表更新

6)

NPT 缓存（类似于 TLB 缓存）