Assignment of Memory Hierarchy

余宏忠
5140309470

November 8, 2016

1 Caching

分析如下:

- 通过第一行的数据可以得到 Block size。因为恰好 hit 两次，说明只能是较为接近的 2 和 4hit 了，也就说明 Block size 为 8 bytes。
- 通过第二行的数据可以得到的是关联度，因为每次地址访问的地址长度过大，无法区分，所以只有关联度影响 hit 率 (其实替换方式也会影响，但在这里不会)，而 hit 了三次根据数据，只能是第二次 1536，1024 和 512 有 hit，故关联度为 4。
- 第三行也有九次也是 hit 了三次，和第二次一模一样，说明每一路的 size 不足以区分 64 和 128，因此总 cache size 为 256bytes。
- 第四行很明显是为了测试替换的方式，而注意到只有两次 hit 说明第二次 512 没有 hit 到，也就是被移除了，而 0 并没有被移除过，因此只能是 LRU 模式。

综上, 该 Cache 为一个总大小为 256bytes,Block size 为 8bytes 的 4 路 LRU 模式 Cache。

2 Memory Address Translation

1. 虚拟地址为: 0x03a9

   - 二进制表达: 00 0011 1010 1001
   - 表格 3 如下:

Parameter	Value
VPN	0000 1110 (0E)
TLB index	10 (2)
TLB tag	00 0011 (03)
TLB hit?	N
Page fault?	N
PPN	01 0001 (11)

   - 物理地址为 0100 0110 1001
   - 表格 4 如下:

Parameter	Value
Byte offset	01 (1)
Cache index	1010 (A)
Cache tag	01 0001 (11)
Cache hit	N
Cache byte returned	-

2. 虚拟地址为: 0x0040

   - 二进制表达: 00 0000 0100 0000
   - 表格 3 如下:

Parameter	Value
VPN	0000 0001 (01)
TLB index	00 (0)
TLB tag	00 0000 (00)
TLB hit?	N
Page fault?	Y
PPN	-

   - 没有对应物理地址和 Cache 的问题。

3. 虚拟地址为: 0x03d7

   - 二进制表达: 00 0011 1101 0111
   - 表格 3 如下:

Parameter	Value
VPN	0000 1111 (0F)
TLB index	11 (3)
TLB tag	00 0011 (03)
TLB hit?	Y
Page fault?	N
PPN	00 1101 (0D)

   - 物理地址为 0011 0101 0111
   - 表格 4 如下:

Parameter	Value
Byte offset	11 (3)
Cache index	0101 (5)
Cache tag	00 1101 (0D)
Cache hit	Y
Cache byte returned	1D

3 Cache Parameters

1. 正确，因为容量不变导致 set 数减半，tag 数也减半。
2. 错误，因为容量和 line size 都不变，tag 数不变。
3. 正确，因为容量增大了，tag 长度变大，冲突导致的 miss 变少。
4. 错误，因为只有 line size 不变，原本的 compulsory misses 的仍然会发生。
5. 正确，因为每次可以有更多数据进入 Cache 中，更高概率能减少 compulsory miss。

4 Cache and Virtual Memory

1. 总的 set 数为 2^{11}，一共需要 11bits。
2. 每个 set 有 64 bytes，一共 2^{17}bytes = 128KB
3. 总共有 8 个可能的位置，因而有 3 位 overlap。
4. Index: 16-6 Offset:5-0
5. Frame number: 31-14 Offset:13-0
6. 总大小:2^{14}bytes = 16KB
7. 总大小:$2^{10+10+14}$bytes = 16GB
8. 总大小:$2^{3}2$bytes = 4GB

5 Caches

	1	2	3	4
1	miss	miss	miss	miss
4	miss	miss	miss	miss
8	miss	miss	miss	miss
5	miss	hit	miss	miss
20	miss	miss	miss	miss
17	miss	miss	miss	miss
19	miss	hit	miss	miss
56	miss	miss	miss	miss
9	miss	hit	miss	miss
11	miss	hit	miss	miss
4	miss	miss	hit	hit
43	miss	miss	miss	miss
5	hit	hit	hit	hit
6	miss	hit	miss	miss
9	hit	miss	hit	hit
17	hit	hit	hit	hit

1. 最终的 Cache:

Index	0	1	2	3	4	5	6	7
Address	-	17	-	19	4	5	6	-
Index	8	9	10	11	12	13	14	15
Address	56	9	-	43	-	-	-	-

2. 最终的 Cache:

Index	0	4	8	12
Address	16-19	4-7	8-11	-

3. 最终的 Cache:

Index	0	1	2	3	4	5	6	7
Address	56	17	-	43	4	5	6	-
Address	8	9	-	11	20	-	-	-

4. 最终的 Cache:

Index	0
Address	17
Address	9
Address	6
Address	5
Address	43
Address	4
Address	11
Address	56
Address	19
Address	20
Address	8
Address	1
Address	-
Address	-
Address	-
Address	-

6

1. 增加是因为有 spatial locality 得到了更多相邻近的地址，减少是因为 temporal locality，只能存贮更少的最近访问的地址。
2. 不一定，因为不同的 block size 对应的 miss penalty 不一样，需要结合二者综合考虑。

7

1. $T_{average} = t_{L1} + m_1 \times t_{p1} = t_{L1} + m_1 \times (t_{L2} + m_2 \times t_{p2})$
2. $m_{global} = m_1 \times m_2$
3. $t_{p2} = (20 + 1) \times (512/32) = 336$ cycles
4.
   - 对于 L1 来说:31-11:Tag 10-2:Index 1-0:Offset
5. 对于 L2 来说:31-17:Tag 16-4:Index 3-0:Offset
6. 总的 CPI 增加:

$$T_{inc} = m_1 \times p_{L1} = m_1 \times (t_{L2} + m_2 \times p_{L2}) = 4\% \times (8 + 40\% \times 336) \approx 5.7\,\text{cycles}$$

8 Cache Organization

1. 最少是全关联，最多是直接映射。
2. 最少的是直接映射，最多的是全关联。
3.
   - 51
   - 0
   - 51

9 Victim Caches

1. 是在 Cache 和下一级存贮之间的一个全关联的 Cache，一般容量较小。在 Cache 中没找到时，通过找 Victim Cache 从而减少 miss penalty。
2. 一个是在开始之前取一些，一个是在退出之后留一些。两者存在相似之处，都是通过额外得到一部分 Data 来降低 L1 Cache 的 miss penalty，同时，它们本身也都是 Cache。

10 Loop Ordering

Problem A

- 总 miss:

$$8 * 128 = 1024$$

- 总 miss:

$$128 * 8 = 1024$$

Problem B

- 只需要 1 行即可。
- 需要 128 行。

Problem C

- 总 miss 也为 1024
- 总 miss 也为 1024
- 不能，因为本题中两个 loop 的 miss 都是一定发生的，增大 Cache 并不能减少这样的 miss。
- 可以，因为本题中两个 loop 的 spatial locality 很高。

11 2.11

1. - 使用的话:$120 + 16 = 136$ cycles
   - 不使用的话:$120 + 32 * 4 = 248$ cycles
2. 相比而言，该方法不需要使用额外硬件，同时能够有效降低 miss penalty。在 Block size 较大时，效果会更好。而 Block size 较小时，多级 Cache 效果会更好。

12 2.12

1. 16B，和 L2 Cache 的相匹配。
2. 2 倍的加速，因为 nonmerging 只使用一半的带宽。
3. Blocking 的会等到数据写入完毕，Nonblocking 的则会直接从 write buffer 中读取。

13 2.13

1. 应该为 1Gbit，因此需要 $16 + 2 = 18$ 个 chips，数据读写的带宽为 4bits。
2. 数据的带宽是 64bits, 因此长度为 4。
3. - 对于 DDR2-667:$667 * 8 = 5336$MB/s
   - 对于 DDR2-533:$533 * 8 = 4264$MB/s