CS M151B: Homework 9

Jonathan Woong

804205763

Spring 2017

Discussion 1A

Wednesday 7th June, 2017

Problem 1. You are a member of the design team of a new computer system. Your task is to nail down the details regarding the implementation of the cache. Some decisions cannot be changed:

- The system is designed for programs where 65% of memory accesses are reads and 35% of memory accesses are writes.
- The block size is 16 bytes.
- The width of the bus between the cache and main memory is 32 bits.
- The cache access time is 5 ns.
- The main memory access time is 68 ns.
- The write policy is write-back.

On a write miss, there are two options:

(I) The main memory is updated but the accessed block is not copied to the cache (write-around). The hit rate is 90% and on average 70% of the blocks in the cache are dirty.

(II) The block is moved to the cache and is updated there (write-allocate). The hit rate is 95% and on average 75% of the blocks in the cache are dirty.

In order to choose between options (I) and (II), you must compute the effective access time for each case and pick the design that will result in a lower access time. Explain any assumption you need to make. Explain briefly the computations you make, i.e., what is the justification for the way you are computing the effective access time. Check your calculations carefully a correct decision requires correct computations.

(I) Assumption: 1 memory access to update main memory. $\frac{16 \text{ bytes}}{32 \text{ bits}} = 4$ memory accesses to fill 1 cache block.

case	**access time (ns)**	**probability**
read hit	5	$0.65 \times 0.9 = 0.585$
clean read miss	$5 + (4 \times 68) = 277$	$0.65 \times 0.1 \times 0.3 = 0.0195$
dirty read miss	$5 + (8 \times 68) = 549$	$0.65 \times 0.1 \times 0.7 = 0.0455$
write hit	5	$0.35 \times 0.9 = 0.315$
write miss	$5 + 68 = 73$	$0.35 \times 0.1 = 0.035$

Effective access time$_I$ $= (5\times0.585)+(277\times0.0195)+(549\times0.0455)+(5\times0.315)+(73\times0.035) = 37.436$ ns

(II) Load block into cache for store miss and load miss.

case	access time (ns)	probability
cache hit	5	0.95
clean cache miss	$5 + (4 \times 68) = 277$	$0.05 \times 0.25 = 0.0125$
dirty cache miss	$5 + (8 \times 68) = 549$	$0.05 \times 0.75 = 0.0375$

$$\text{Effective access time}_{II} = (5 \times 0.95) + (277 \times 0.0125) + (549 \times 0.0375) = 28.8 \text{ ns}$$

$\boxed{\text{Choose option (II), which has a lower effective access time of 28.8 ns.}}$

Problem 2. 5.11.4:

VA size	page size	PTE size
32 bits	8 KiB	4 bytes

Given the parameters shown above, calculate the total page table size for a system running 5 applications that utilize half of the memory available.

$$8 \text{ KiB} = 2^{13} \text{ bytes} \Rightarrow 13 \text{ bit page offset}, 19 \text{ bit page number}$$

$$2^{19} = 0.5 \text{ M PTEs}$$

$$0.5 \text{ M PTEs} \times 4 \text{ bytes/entry} = 2 \text{ MB per page table per application}$$

$$5 \text{ applications} \times 2 \text{ MB} = \boxed{10 \text{ MB for all application page tables}}$$

Problem 3. 5.11.5: Given the parameters shown above, calculate the total page table size for a system running 5 applications that utilize half of the memory available, given a two level page table approach with 256 entries. Assume each entry of the main page table is 6 bytes. Calculate the minimum (I) and maximum (II) amount of memory required.

$$8 \text{ KiB} = 2^{13} \text{ bytes} \Rightarrow 13 \text{ bit page offset}, 19 \text{ bit page number}$$

$$8 \text{ MSB index into first-level} \Rightarrow 11 \text{ bit index into second-level} \Rightarrow 2^{11} = 2048 \text{ entries}$$

(I) 128 first-level entries per application

$$128 \text{ first-level entries} \times 2048 \text{ entries per second-level page table} = 262 \text{ K entries}$$

$$262 \text{ K entries} \times 4 \text{ bytes/entry} = 1048 \text{ KB for all second-level page tables per application}$$

$$256 \text{ entries} \times 6 \text{ bytes} = 1536 \text{ bytes for first-level per application}$$

$$5 \text{ applications} \times (1048 \text{ KB} + 1538 \text{ bytes}) = \boxed{5.12 \text{ MB for all page tables}}$$

(II) 256 first-level entries per application

$$256 \text{ first-level entries} \times 2048 \text{ entries per second-level page table} = 0.5 \text{ M entries}$$

$$0.5 \text{ M entries} \times 4 \text{ bytes/entry} = 2 \text{ MB for all second-level page tables per application}$$

$$256 \text{ entries} \times 6 \text{ bytes} = 1536 \text{ bytes for first-level per application}$$

$$5 \text{ applications} \times (2 \text{ MB} + 1538 \text{ bytes}) = \boxed{10.01 \text{ MB for all page tables}}$$

Problem 4. A virtual memory has a page size of 4096 (2^{12}) words. There are 64 pages of virtual address space but only 5 page frames in real memory. The page table is stored in a special memory module and does not take up space in real memory. The program issues the following sequence of addresses (hex):

```
2f012, d120, 550, d58b, 7194, 30000, 7052, 5550, 2fa02, 744, 7276
```

Before the program begins execution, the real memory is empty.

A) Assume that FIFO replacement is used. Circle the references that will cause a page fault.

[✓ 2f012], [✓ d120], [✓ 550], d58b, [✓ 7194], [✓ 30000], 7052, [✓ 5550], [✓ 2fa02], 744, 7276

B) Assume that LRU replacement is used. Circle the references that will cause a page fault.

[✓ 2f012], [✓ d120], [✓ 550], d58b, [✓ 7194], [✓ 30000], 7052, [✓ 5550], [✓ 2fa02], [✓ 744], 7276

Problem 5. A computer system has the following characteristics:

- The memory is byte addressable.
- The disk address where a virtual page is stored is the virtual page number.
- Processor: 30 bit addresses. 65% of memory accesses are reads.
- Real memory: Size: 128 MiB. Access time: 110 ns. Width of bus to memory is 64 bits. Page size: 16 KiB.
- Cache: Accessed using physical addresses. Size: 512 KiB. Access time: 15 ns. Block size: 64 bytes. Organization: 8-way set-associative. Hit rate: 95%. Write policy: write-back, write-around.
- TLB: Size: 256 entries. Access time: 5 ns. Organization: direct mapped. Hit rate: 98%.

A) Assume that the page fault rate is 0.1% and the average disk access time is 8 ms. Specify what additional information is needed.

- Time to (handle page fault): save current state, transfer control to page fault handling routine, initiate disk access, read from disk, detect operation completion, restart process, transfer page from disk to memory.
- Amount of dirty pages in main memory.
- Amount of dirty cache blocks.
- Levels of page table.

B) Assume that the page fault rate is 0. Compute the effective access time to the memory system. Clearly specify any assumptions you make.

Let d = fraction of dirty cache blocks.

Let k = number of levels of page table.

$$\text{translation time} = 5 \text{ ns} + (0.02 \times 110 \text{ ns} \times k) = (5 + 2.2 \times k) \text{ ns}$$

case	**access time (ns)**	**probability**
read hit	15	$0.65 \times 0.95 = 0.6175$
clean read miss	$15 + (8 \times 110) = 895$	$0.65 \times 0.05 \times (1 - d)$
dirty read miss	$15 + (16 \times 110) = 1775$	$0.65 \times 0.05 \times d$
write hit	15	$0.35 \times 0.95 = 0.3325$
write miss	15 + 110 = 125	$0.35 \times 0.05 = 0.0175$

Effective access time = translation time + memory access time

$$= (5 + 2.2 \times k) + (15 \times 0.6175) + (895 \times 0.65 \times 0.05 \times (1 - d)) + (1775 \times 0.65 \times 0.05 \times d) + (15 \times 0.3325) + (125 \times 0.0175)$$

$$= \boxed{(50.525 + 2.2k + 28.6d) \text{ ns}}$$

Problem 6. Consider a byte-addressable virtual memory system with the following properties:

- 64 bit virtual addresses
- 8 KiB pages
- 38 bit physical address
- 8-way set-associative TLB with 512 page-table entries
- Four level page table

Computer the total amount of storage required for the TLB. Your answer should be the total number of bits required for all storage needed for the TLB. State every assumption you make and explain the calculation.

8 KiB page = $2^{13}B \Rightarrow$ 13 bit page offset

64 - 13 = 51 bit virtual page number

$\frac{\text{1024 entries}}{\text{8 entries per frame}} = 128 = 2^7$ set frames $\Rightarrow$ 7 bit index field

51 - 7 = 44 bit tag field

Virtual address

44	7	13
VPN		page offset
tag	index	page offset

38 - 13 = 25 bit PPN

Physical address

25	13
PPN	page offset

Assume TLB includes:

- 1 valid bit
- 1 dirty bit
- 3 protection bits

One entry = 1 + 1 + 3 + 44 + 25 = 74 bits

1024 entries = (74 × 1024) = $\boxed{\text{75776 bits}}$

Problem 7. Consider a shared-memory multiprocessor where each processor has a local cache. The cache write policy is write-back, write-allocate. Each cache block contains two words. Elements `X[0]` and `X[1]` of array `X` are in the same block. Initially, `X[0]=X[1]=0`. Then, the two processes simultaneously execute the following C code:

P1	**P2**
`X[0]++; print X[1];`	`X[1] += 2; print X[0];`

A) There are several possible results from the execution of this code. Assume that cache coherence is maintained. List all the possible values that may be printed by this program. Your answer must be a list of pairs, where the first element of each pair is the value printed by P1 and the second element of each pair is the value printed by P2. Explain your answers.

step	**P1**	**P2**
1	\$1 ← X[0]	\$1 ← X[1]
2	\$1 ← \$1 + 1	\$1 ← \$1 + 2
3	X[0] ← \$1	X[1] ← \$1
4	\$1 ← X[1]	\$1 ← X[0]
5	jal print	jal print

result	**case**
X[0] = 0	$P2_4$ before $P1_3$
X[0] = 1	$P1_3$ before $P2_4$
X[1] = 0	$P1_4$ before $P2_3$
X[1] = 2	$P2_3$ before $P1_4$
(X[0] = 0 && X[1] = 0) impossible	$P2_4$ before $P1_3$

Outcomes: $\boxed{(0,1),(2,0),(2,1)}$

B) Assume that cache coherence is not maintained. List at least one possible value of the cache block that is an outcome of executing of the code above and is not on the list that you produced in your answer to part A. As with part A, the answer here is at least one pair. Explain your answer.

step	**P1**	**cache 1**	**P2**	**cache 2**
$P1_1$	\$1 ← X[0]	miss, X[0]=0, X[1]=0		
$P2_1$			\$1 ← X[1]	miss, X[0]=0, X[1]=0
$P2_2$			\$1 ← \$1 + 2	
$P2_3$			X[1] ← \$1	hit, X[0]=0, X[1]=2
$P2_4$			\$1 ← X[0]	hit, X[0]=0, X[1]=2
$P2_5$			jal print	
$P1_2$	\$1 ← \$1 + 1			
$P1_3$	X[0] ← \$1	hit, X[0]=1, X[1]=0		
$P1_4$	\$1 ← X[1]	hit, X[0]=1, X[1]=0		
$P1_5$	jal print			

result	**case**
X[1] = 2	$P2_3$
X[1] = 0	$P1_4$

Result: $\boxed{(0,0)}$

C) Assume that cache coherence is maintained using a simple write-invalidate snoopy protocol. Before execution begins, all the block frames in the caches of both processors are invalid. What is the minimum number of cache misses, in both caches, that will occur during the execution of the program. Explain your answer.

If one processor executes all of its code before the other, then we get the minimum number of cache misses. This means that the only misses are compulsory misses, making the minimum number of misses 2.

D) Repeat part C but now determine the maximum number of cache misses, in both caches, that will occur during the execution of the program. Explain your answer.

step	**P1**	**cache 1**	**P2**	**cache 2**
$P1_1$	\$1 ← X[0]	miss, X[0]=0, X[1]=0		
$P2_1$			\$1 ← X[1]	miss, X[0]=0, X[1]=0
$P2_2$			\$1 ← \$1 + 2	
$P2_3$		invalidate (X[0],X[1])	X[1] ← \$1	hit, X[0]=0, X[1]=2, mark as dirty
$P1_2$	\$1 ← \$1 + 1			
$P1_3$	X[0] ← \$1	miss, X[0]=1, X[1]=2, mark as dirty		write-back && invalidate (X[0]=0, X[1]=2)
$P2_4$		write-back X[0]=1, X[1]=2	\$1 ← X[0]	miss, X[0]=1, X[1]=2
$P2_5$			jal print	
$P1_4$	\$1 ← X[1]	hit, X[0]=1, X[1]=0		
$P1_5$	jal print			

Maximum misses = 4