ECE M116C - CS M151B Computer Architecture Systems - UCLA

Homework 5 Deadline: Monday, Dec. 9, 11:59 PM

Fall 2024 (Upload it to Gradescope.)

Author: Krish Patel

Q1. Assume a memory hierarchy with a 16-way set-associative L1 cache with 64 sets, an 8-way set-associative L2 cache with 256 sets, and a victim cache with 4 entries. The virtual address space is 16GB, the page size is 2KB, and the cache block size is 64 bytes. Each way contains one block. The physical address space is 1GB.

Answer the following questions. (For each part, you need to show your work and how you calculated this number.)

a) **What is the maximum size of the cache that enables us to have a virtually-indexed physically tagged cache with no aliasing?**

To prevent aliasing:

Index width + Block offset width <= page offset width
Page offset width = log(2^11) = 11

Block size = 64 bytes, block size width = log (2^6) = 6

Index width = Log (64(Number of sets)) = 6

6+6 > 11, thus, this condition is not met. We have to reduce number of sets

16 way set associativity: cache size = Block size * associativity * Number of sets = 512 bytes

b) **How many virtual pages does this memory have?**
Virtual pages = Virtual Address Space / page size = 2^(34-11) = 2^23 pages

c) **How many physical pages does this memory have?**
Physical pages = Physical address space / page size = 2^(30-11) = 2^19

d) What is the size of the flat (one-level) page table (per process) in this system?

Page table entry size = 19 bits (closest to 3 bytes). Table size = 2^23 * 3 bytes ~ 24 MB

e) If we use a 2-level page table design with 10 bits for the first level and the rest for the second level, what would be the <u>maximum</u> size of this two-level page table?

First level = 2^10 entries
Second level = 2^(23-10)= 2^13 entries (size = 24 KB)
total size of the two level page table = 2^10* 24KB = 24 MB

f) If the 8 most significant bits of the virtual address always stay zero (i.e., the process never uses those addresses), what would be the size of the 2-level page table?

34 - 8 = 26 bits
New virtual pages = 2^(26-11) = 2^15 pages
Flat page table size = 2^15 * 3 = 96KB

1st level size = 2^10 * 3 = 3KB
2nd Level size = 2^13 * 3KB = 24 KB
Total size = 27 KB

Q2. Assume we have a memory hierarchy with an 8-way set-associative L1 cache with 64 sets and a TLB with 4 entries. The TLB is fully associative. We have an 8-way set-associative cache with 128 sets. The virtual address space is 4GB, the page size is 4KB, and the cache block size is 4 bytes. Each way contains one block. The physical address space is 512MB. Assume that we store 8 bits of flags and/or metadata for each page.

Also, assume that the hit time and miss rate for the memory hierarchy are as follows.

Unit	Hit Time (cycles)	Miss Rate
TLB	1	.01%

Cache (L1)	**1**	**2%**
Cache (L2)	**10**	**5%**
MM	**150**	**.0001%**
Disk	**10,000**	**0%**

Answer the following questions (*show your work for each part*).

a) **What is the size of TLB?**
TLB entry = 20 bits(vpn) + 19 bits (PPN) + 8 bits(metadata) = 47 bits ~ 6 bytes
TLB size = 4 entries * 6 bytes = 24 bytes

b) **If we have a flat page table that is stored in the main memory (MM), what is the average access time (AAT) for this memory system? Assume that the cache is physically indexed physically tagged (PIPT).**

AAT = Hit time + Miss Rate * miss penalty

1+0.01%*150 + (1+2%*(10+0.05*150) = 1.015+1.35 = 2.365 cycles*

c) **If the cache is VIPT, what is the new AAT?**

AAT = 1+2%*(10+150) = 4.2 cycles

Q3. Assume that we have a multi-core system with the following instruction sequences:

Initial Values: `Per Thread: x1=0, x2=0  Global: flag = 0, Money = 1`

THREAD 1

```
x1 = Money;
Money = Money + 1;
fence rw;
acquire (lock);
x1=4;
release (lock);
```

THREAD 2

```
while(flag==0);
x2 = Money + x1;
```

THREAD 3

```
fence rw;
x2 = Money;
acquire (lock);
flag = 1;
fence rw;
Money = Money + 1;
release (lock);
```

a) **If the memory model is TSO, what are the possible values for `x1` in Thread 2 and x1 in Thread 1? (Show the sequence by denoting numbers as T1.1, T1.2, etc. where Tx.y means the Yth instruction in Thread X).**

Initial values: x1 = 0, x2 = 0, Money = 1
Possible values: x1 = 1,x2 = 2 or x1 =4, x2 = 2

(For each unique answer, show only one possible sequence that would create that value.)

b) Repeat part (a) for the Release Consistency memory model. Does anything change? Why or why not?

Release consistency adds ordering constraints around synchronization primitives(eg acquire, release). This doesn't change the possible values for x1 and x2, as fences already enforce necessary ordering.