國立高雄大學資訊工程學系 計算機結構期末考考卷

姓名： 學號：

1. (10%) Please compare Write invalidate protocol and Write update protocol.

A. Write invalidate protocol

i. To ensure that a processor has exclusive access to a data item before it writes that item.

ii. It invalidates other copies on a write.

B. Write update or write broadcast protocol:

i. To updates all the cached copies of a data item when that items is written.

ii. Because a write update protocol must broadcast all writes to shared cache lines, it consumes considerably more bandwidth.

Invalidate:

+ Multiple writes by the same processor to the cache block only require one invalidation
+ No need to send the new value of the data (less bandwidth)
– Caches must be able to provide up-to-date data upon request
– Must write-back data to memory when evicting a modified block

Usually used with write-back caches (more popular)

Update:

+ New value can be re-used without the need to ask for it again
+ Data can always be read from memory
+ Modified blocks can be evicted from caches silently
– Possible multiple useless updates (more bandwidth)

Usually used with write-through caches (less popular)

2. (20%) There are numerous techniques for improving the performance of caches. Some reduce the frequency of misses, some reduce the miss penalty, some reduce hit times, and so on. Many optimizations involve trade-offs, making one performance factor worse in return for reducing another. In some cases, the so-called optimization can actually hurt performance. For the following optimizations, briefly summarize the trade-offs involved and how it helps or can hurt performance. Be as specific as possible, for example, if an optimization reduces conflict misses, say so, rather than just say “it reduces misses”.

A. Higher associativity

i. Reduce conflict misses

ii. Increase hit time

- Decreases conflict misses by hashing data
- More complex replacement protocol

B. **Larger block size**

i. Take advantage of spatial locality
ii. Reduce compulsory misses
iii. Increase the miss penalty

+ Decreases compulsory and capacity misses by "prefetching" data
+ Increases capacity misses by decreasing number of unique blocks in cache
- Increases memory traffic
- May increase miss penalty

C. **Bigger caches**

i. Reduce capacity misses
ii. Potentially longer hit time of the larger cache memory
iii. Higher cost and power

D. **Non-blocking caches**

i. **allow data cache to continue to supply cache hits during a miss**
ii. **reduces the effective miss penalty by working during miss vs. ignoring CPU requests**

+ Allows multiple outstanding cache misses, allowing misses to overlap
+ Increases *memory system bandwidth*
- Requires dynamically scheduled processor to fully exploit
- Complex hardware

3. (20%) Please discuss the following compiler optimization techniques that can improve the cache performance. You can give an example code and explain it

A. *Merging Arrays*: improve spatial locality by single array of compound elements vs. 2 arrays

B. *Loop Interchange*: change nesting of loops to access data in order stored in memory

C. *Loop Fusion*: Combine 2 independent loops that have same looping and some variables overlap

D. *Blocking*: Improve temporal locality by accessing "blocks" of data repeatedly vs. going down whole columns or rows

4. (12%) Please explain the three types of cache misses.
   A. Compulsory
   B. Capacity
   C. Conflict (collision)

   Please write the equation of the average memory access time.

- Compulsory

  The very first access to a block cannot be in the cache, so the block must be brought into the cache.
- Capacity

  If the cache cannot contain all the blocks needed during execution of a program, capacity misses will occur because of blocks being discarded and later retrieved.
- Conflict (collision)

  If the block placement strategy is set associative or direct mapped conflict will occur because a block may be discarded and later retrieved if too many blocks map to its set.

$$\text{Average Memory Access Time} = \text{Hit Time} + \text{Miss Rate} \times \text{Miss Penalty}$$

5. (15%) Please describe the cache coherence protocols: snooping protocol and directory protocol. Please explain the limitations of the snooping protocol as the number of processors grows.

   A. Directory based — Sharing status of a block of physical memory is kept in just one location, the directory

   B. Snooping — Every cache with a copy of data also has a copy of sharing status of block, but no centralized state is kept

   i. All caches are accessible via some broadcast medium (a bus or switch)

   ii. **All cache controllers monitor or snoop on the medium to determine whether or not they have a copy of a block that is requested on a bus or switch access**

   Any centralized resource in the system can become a bottleneck

6. (16%) Please describe the advantages and the disadvantages of “Fine-Grained Multithreading”, and “Course-Grained Multithreading”, respectively.

Fine-Grained Multithreading

A. Advantage is it can hide throughput losses that arise from both short and long stalls, since instructions from other threads executed when one thread stalls

B. Disadvantage is it slows down execution of individual threads, since a thread ready to execute without stalls will be delayed by instructions from other threads

Course-Grained Multithreading

A. Advantages

i. Relieves need to have very fast thread-switching

ii. Doesn't slow down thread, since instructions from other threads issued only when the thread encounters a costly stall

B. Disadvantage is hard to overcome throughput losses from shorter stalls, due to pipeline start-up costs

iii. Since CPU issues instructions from 1 thread, when a stall occurs, the pipeline must be emptied or frozen

iv. New thread must fill pipeline before instructions can complete

7. (10%) Please describe the following schemes to increase instruction fetch bandwidth: Branch-Target Buffers and Return Address Predictors.

Branch-Target Buffers

A. Predicts next instruct address, sends it out *before* decoding instructuction

B. PC of branch sent to BTB

C. When match is found, Predicted PC is returned

D. If branch predicted taken, instruction fetch continues at Predicted PC

Return Address Predictors

A. Small buffer of return addresses acts as a stack

B. Caches most recent return addresses

C. Call ⇒ Push a return address on stack

D. Return ⇒ Pop an address off stack & predict as new PC

8. (30%) Please complete the examples based on the example snooping protocol and the example directory protocol in the text book.