**Shishir Ghorashainee**

CSC 362 Homework Assignment #6

Due Date: Wednesday, November 20

Word Processor all answers, show work for partial credit.

1. Answer the following questions regarding locality of reference and cache hit rates. For each, offer a brief (1-2 sentence) explanation.
   1. What change do we make to memory so that locality of reference can improve cache hit rates?

**We can store related or frequently used data close to each other in memory, like using arrays or grouping data together. This makes it easier for the cache to load and use the data efficiently.**

* 1. Without changing memory, which of sequential, spatial and temporal locality would improve cache hit rates?

= **Without changing memory, temporal locality improves cache hit rates because recently used data is likely to be reused soon. Spatial locality does not help if the memory layout doesn’t match how data is stored.**

* 1. Branches interfere with which form(s) of locality, sequential, spatial, temporal?

= **Branches interfere with sequential locality as they make the program jump to different parts of memory, breaking the order of access.**

* 1. Why does the following code not support locality of reference as it contains loops and arrays?

for(j=0;j<n;j++) for(i=0;i<n;i++)

a[i][j]++;

= **The given code does not support locality of reference because it accesses the array column by column instead of row by row, so this doesn’t use the memory layout efficiently leading to frequent cache misses.**

1. A word addressable computer with a 128-bit word size has 32GB of memory and a directmapped cache of 4096 refill lines where each refill line stores 16 words. Note: convert 32GB to words first.

**Given, 32 GB of Memory Size.**

**So, the number of words = 32 GB / 16 bytes / word = 2GW,**

**and it needs 31 bits for the address.**

**(dividing the address into tag, line, word )**

* 1. What is the format of memory addresses if the cache is direct-mapped?

**= 4096 lines 🡪 12 bits, so 16 – 12 - 3**

* 1. What is the format of memory addresses if the cache is made a 2-way set associative?

**= 2048 lines 🡪 16 – 11 - 4**

* 1. What is the format of memory addresses if the cache is made a 8-way set associative?

**= (4096/8)lines 🡪 512 lines 🡪 18 – 9 - 4**

* 1. What is the format of memory addresses if the cache is made a fully associative?

**= 1 line 🡪 27 – 0 - 4**

1. Read problem #11 on page 404. Use the same memory and cache layout and determine for the 20 memory references below which ones are hits and which ones are misses, and compute the hit rate. Assume the cache is already loaded as follows: block 0 has tag 9, block 1 has tag B, block 2 has tag 0 and block 3 has tag 0. The 20 memory references are, in order: 07, 08, 09, 0F, 10, 11, BB, BC, B8, B9, 93, 94, 95, 0F, 10, 13, 14, 26, 27, 28. Whenever there is a miss, you have to update the cache to keep track of what has been brought in and what has been discarded. Your answer should consist of, for each memory reference, whether it is a hit or miss and what the cache’s overall hit rate is for these 20 memory references.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Address | Block | Tag | Action (hit or miss) | Cache Update |
| 07 | 3 = 11 | 0000 | Miss | Load 0000 |
| 08 | 0 = 00 | 0000 | Hit | Keep 0000 |
| 09 | 0 = 00 | 0000 | Hit | Keep 0000 |
| 0F | 3 = 11 | 0000 | Hit | Keep 0000 |
| 10 | 0 = 00 | 0001 | Miss | Load 0001 |
| 11 | 0 = 00 | 0001 | Hit | Keep 0001 |
| BB | 3 = 11 | 1011 | Miss | Load 1011 |
| BC | 0 = 00 | 1011 | Miss | Load 1011 |
| B8 | 2 = 10 | 1011 | Hit | Keep 1011 |
| B9 | 2 = 10 | 1011 | Hit | Keep 1011 |
| 93 | 3 = 11 | 1001 | Miss | Load 1001 |
| 94 | 0 = 00 | 1001 | Miss | Load 1001 |
| 95 | 0 = 00 | 1001 | Hit | Keep 1001 |
| 0F | 3 = 11 | 1001 | Miss | Load 0000 |
| 10 | 0 = 00 | 0000 | Miss | Load 0001 |
| 13 | 0 = 00 | 0001 | Hit | Keep 0001 |
| 14 | 1 = 01 | 0001 | Miss | Load 0001 |
| 26 | 2 = 10 | 0010 | Miss | Load 0010 |
| 27 | 2 = 10 | 0010 | Hit | Keep 0010 |
| 28 | 0 = 00 | 0010 | Miss | Load 0010 |

**Hit rate = (9 / 20) \* 100 = 45%**

1. A processor’s core has an on-chip L1 cache and an off-chip L2 cache. These are backed up by DRAM. The L1 cache has a hit time of .25 ns and a hit rate of 88.7%, the L2 cache has a hit time of 1.5 ns and a hit rate of 94.2%, and DRAM has an access time of 20 ns and a 100% hit rate.
   1. What is the effective access time for this processor?

**EAT = 0.25ns + 0.113 \* (1.5ns + 0.058 \* 20ns)**

**= 0.551ns**

* 1. We add a second on-chip cache (a new L2) making the existing L2 into an L3 cache. The new L2 has a hit time of .8 ns and a hit rate of 91.9%. What is the effective access time now?

= **EAT = 0.25ns + 0.113 \*(0.8ns + 0.081\* (1.5ns + 0.058 \* 20ns))**

**= 0.365ns**

1. Answer the following questions with regard to cache types.
   1. What makes it possible for a 2-way set associative cache to have a better hit rate than a direct-mapped cache?

= **A 2-way set associative cache improves hit rates by giving each set two possible places to go, reducing conflicts. It uses a replacement strategy, like Least Recently Used (LRU), to predict which sets won’t be needed soon and replaces that one, keeping frequently used data in the cache.**

* 1. For a 2-way set associative cache, what replacement strategy would be best?

= **For a 2-way set associative cache, LRU strategy would be best because it removes the set that won’t be used soon, keeping the frequently used data in the cache.**

* 1. For a fully associative cache, what replacement strategy would be best?

**= For a fully associative cache, a better choice is to randomly pick the replacement strategy.**

* 1. Of the various types of cache, which provides the best hit time? What is it about this type of cache that makes this so?

= A **direct-mapped cache provides the best hit time because each memory block maps to exactly one location in the cache. This makes it very fast to check if the data is in the cache since there’s no need to search multiple locations or use replacement strategies.**

1. We have a computer with an L1 cache (1 ns hit time, 92% hit rate), L2 cache (2 ns hit time, 96% hit rate), DRAM (10 ns, 99.99% hit rate) and solid state drive as virtual memory (5,000 ns, 100% hit rate). There is a TLB which is also an L1 cache (1 ns hit time, 97.5% hit rate) with the page table stored in DRAM (10 ns, 100% hit rate).

a. What is the effective access time?

= **EAT = (1ns + 0.025\*10ns) + (1ns + 0.08\*(2ns +0.04\*(10+0.0001\*5000)))**

**= 2.4436ns**

* 1. Redo a assuming the computer uses a hard disk for virtual memory instead of the SSD, where the hard disk access time is 500,000 ns.

= **EAT = (1ns + 0.025\*10ns) + (1ns +0.08\*(2+0.04\*(10+0.0001\*500000)))**

**= 2.602 ns**

* 1. If DRAM had a 100% hit rate (that is, we didn’t have any virtual memory), how much faster would the computer be (remember, the speedup is slower value / faster value) over the version with the SSD.

= **EAT = 1ns + 0.08\*(2ns + 0.04\*10ns) = 1.192ns**

**The computer will be 2.4436 / 1.192 = 2.05 times faster.**

* 1. Repeat part c with the version with the hard disk.

**= We have, EAT = 1.192ns**

**So, the computer will be 2.602 / 1.192 = 2.1829 times faster.**

1. Assume a computer is word addressable with 1M of addresses (words) and is running a process whose page table is given below. Answer the following questions.
   1. How big is a page/frame in words? Hint: to solve this, determine the number of bits for the page size (based on the number of pages as shown below), then use the addresses in b-d to determine how many bits make up the page/frame offset.

**= A page frame = 2^14 = 16K words**

* 1. What is the physical address for the logical address 001100110011001100?

= **We have, 0011= 3 (page number), and its frame number is 30(from the table), then converting 30 into 6 bits decimal, we get 011110. The address is 011110 001100110011001100**

* 1. What is the physical address for the logical address 100110011001100110?

**= The physical address is 9. Having no frame number, so the address can’t be mapped.**

* 1. What is the physical address for the logical address 010101010101010101?

**= We have 0101= 5, and its frame number is 0, then converting 0 into 6 bits decimal, we get 000000. The address is 000000 010101010101010101**

* 1. How large is this maximum size of the program in words?

= **Total page numbers = 13**

**Maximum page size = 13 \* 16K words= 208K Words**

* 1. What percentage of the entire program is currently in memory?

**= Only 8 / 13 = 61.54% is currently in memory.**

|  |  |  |
| --- | --- | --- |
| Page  Number | Frame  Number | Valid Bit |
| 0 | 21 | 1 |
| 1 | 33 | 1 |
| 2 | 32 | 1 |
| 3 | 30 | 1 |
| 4 | --- | 0 |
| 5 | 0 | 1 |
| 6 | ---- | 0 |
| 7 | 17 | 1 |
| 8 | 5 | 1 |
| 9 | ---- | 0 |
| 10 | ---- | 0 |
| 11 | 27 | 1 |
| 12 | ---- | 0 |