**CSE 141 Spring 2016 Homework 4**

PID: A11489111

Name: Amit Nijjar

1. The AMD Bulldozer microarchitecture has a 16K, 4-way, 64-byte blocked L1 data cache in each core. This processor uses 64-bit memory addresses. For this processor, please answer the following questions:
   1. When the L1 data cache receives a memory access request, how many bits will be used as “tag”? How many bits will be used as “index”? How many bits will be used as “offset”?

C = ABS

16K = (4)(64)(S)

S = 64

Offset = lg(64) = 6 bits

Index = lg(64) = 6 bits

Tag = 64-6-6 = 52 bits

* 1. In addition to the data array, the cache needs to contain tag arrays and dirty bits that are considered as overheads to the cache. Including these overheads, how many bits are required to build this cache?

16Kb/64 byte blocks = 256 bytes per block / 4 ways = 64

Tag = 52 + 1 dirty bit + 1 valid bit

Setx x ways x (bits per block + tag + dirty bit + verify bit)

Total = 64 x 4 x ( 64 x 8 + 52 + 1 + 1 )

Total = 144,869 bits

1. Consider the following matrix multiplication code  
   int i, j ,k;  
   double \*A, \*B, \*C;  
   A = (double \*)malloc(sizeof(double)\*N\*N);  
   B = (double \*)malloc(sizeof(double)\*N\*N);  
   C = (double \*)calloc(N\*N, sizeof(double));  
   init\_data(A, B, N\*N);  
   for(i = 0; i < N; i++)  
    for(j = 0; j < N; j++)  
    for(k = 0; k < N; k++)  
    C[i\*N+j] += A[i\*N+k]\*B[k\*N+j];  
   // assume load A[i\*N+k], load B[k\*N+j], load C[i\*N+j] then store C[i\*N+j]  
   output\_data(C, N\*N);  
   Assume that the starting address of array A is 0x10000, array B is 0x18000, and array C is 0x20000. Assume N = 64. Please answer the following questions:
   1. How many memory accesses are there in the nested for-loop? How many of them are writes? How many of them are reads?

3 reads and 1 write per execution of C[i\*N+j]+=A[i\*N+k]\*B[k\*N+j];

(((64)^64)^64) x 3 memory accesses

(((64)^64)^64) x 2 reads

(((64)^64)^64) x 1 writes

* 1. Assuming that you have an AMD Phenom II processor, which has a 64KB, 2-way, 64-byte blocked L1 data cache, please estimate the cache miss rate.  
     C = ABS

64KB = 2 x 64 x S

S = 512

Offset = 6 bits

Index = 9 bits

Tag = 64 – 15 = 49 bits

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Address | Tag | Index | Hit? Miss? |
| load A[0\*0+0] | 0x10000 | 0x4 | 0 | miss |
| Load B[0\*0+0] | 0x18000 | 0x6 | 0 | Miss |
| Load C[0\*0+0] | 0x20000 | 0x2 | 0 | Mss, evict 0x4 |
| Load a[0\*1+0] | 0x10004 | 0x4 | 0 | Miss, evict 0x6 |

Will always be evicting the previous data because the cache is a two way cache. Will always miss.

1. You are building a system around a single-issue in-order processor running at 1 GHz and the processor has a base CPI of 1 if all memory accesses are hits. The only instructions that read or write data from memory are loads (20% of all instructions) and stores (5% of all instructions). The memory system for this computer is composed of a split L1 cache that imposes no penalty on hits. Both the I-cache and D-cache are direct mapped and hold 32KB each. You may assume the caches use write-allocate and write-back policies. The L1 I-cache has a 2% miss rate and the L1 D-cache has a 5% miss rate. Also, 50% of all blocks replaced from L1 D-cache are dirty. The 512KB write-back, unified L2 cache has an access time of 10ns. Of all memory references sent to the L2 cache in this system, 80% are satisfied without going to main memory. Also 25% of all blocks replaced are dirty. The main memory has an access latency of 60ns.
   1. What is the overall CPI, including memory accesses?  
      application : 75% ALU 20% loads 5% write

L1 I-cache miss rate 2%; hit time 1 cycle

L1 D-Cache miss rate 5%; hit time 1 cycle

L2 U-Cache miss rate 20%; hit time 10 cycles

Main Memory hit time 60 cycles

Average CPI is 1 + 100%\*(2%(10+20%(1.5\*60))) + 25%\*(5%(10+20%(1.25\*60)))

CPI = 1 + .56 + .3125 = 1.8725 CPI

* 1. You are considering replacing the 1 GHz CPU with one that runs at 2 GHz, but is otherwise identical. How much faster does the system run with a faster processor? Assume the L1 cache still has no hit penalty, and that the speeds of the L2 cache, and main memory remain the same in absolute terms (e.g. the L2 cache still has a 10 ns access time).  
       
     application : 75% ALU 20% loads 5% write

L1 I-cache miss rate 2%; hit time 1 cycle

L1 D-Cache miss rate 5%; hit time 1 cycle

L2 U-Cache miss rate 20%; hit time 10 cycles

Main Memory hit time 60 cycles

Average CPI is 1 + 100%\*(2%(10+20%(1.5\*60))) + 25%\*(5%(10+20%(1.25\*60)))

CPI = 1 + .56 + .3125 = 1.8725 CPI

Faster processor – cut CPI in half since it is twice as fast

Average CPI is .5 + 100%\*(2%(10+20%(1.5\*60))) + 25%\*(5%(10+20%(1.25\*60)))

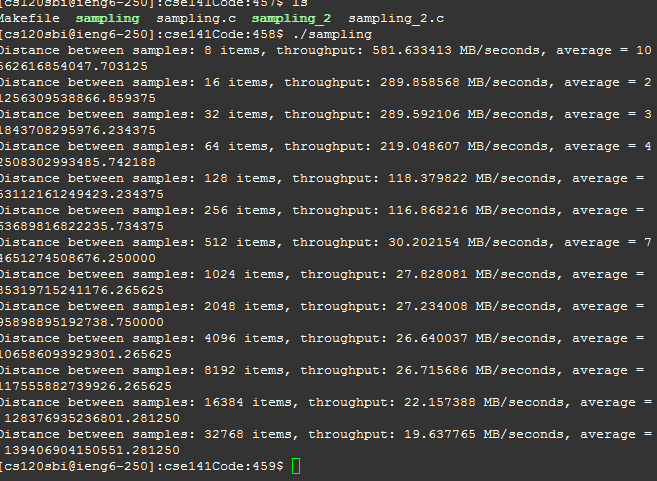
= .5 + .56 + .3125 = 1.3725 CPI

1. Please download the C code from <http://cseweb.ucsd.edu/classes/sp16/cse141-a/homework/homework4.tar.gz> and compile the source files on a machine.
   1. Please find out the cache configuration of the machine and fill the following table:

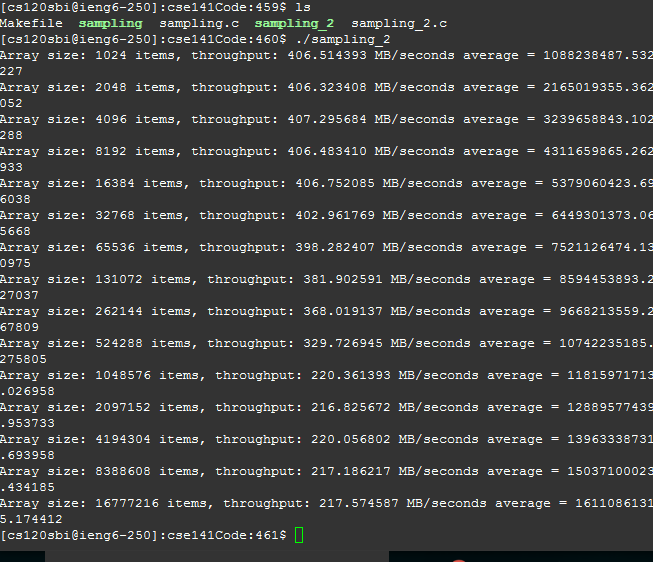
|  |  |  |  |
| --- | --- | --- | --- |
| Memory layer | Capacity | Block size | Way associativity |
| L1 data cache |  |  |  |
| L2 cache |  |  |  |
| L3 cache |  |  |  |

I am unsure how to find the cache configuration. I am using the ing6 servers

* 1. Please execute the program “sampling” and draw a line graph in which the x-axis represents the “distance between samples” and the y-axis represents the “throughput (MB/secs)”.



* 1. Explain the performance curve of the line graph in (2) using the cache configuration on your machine.  
       
     As the distance between samples increases, the throughput decreases. I am unsure about the configuration of the machine, but based on the output, the machine does not have a large capacity to take advantage of spatial locality.
  2. Please execute the program “sampling\_2” and draw a line graph in which the x-axis represents the “array size” and the y-axis represents the “throughput (MB/secs)”.



* 1. Explain the performance curve of the line graph in (4) using the cache configuration on your machine

As the sample array sizes increase, the throughput decreases. I am unsure about the configuration of the machine, but based on the output, the machine does not have a large capacity to take advantage of temporal locality.