NC State University

Department of Electrical and Computer Engineering

ECE 463/521: Fall 2015 (Rotenberg)

Project #1: Cache Design, Memory Hierarchy Design

by

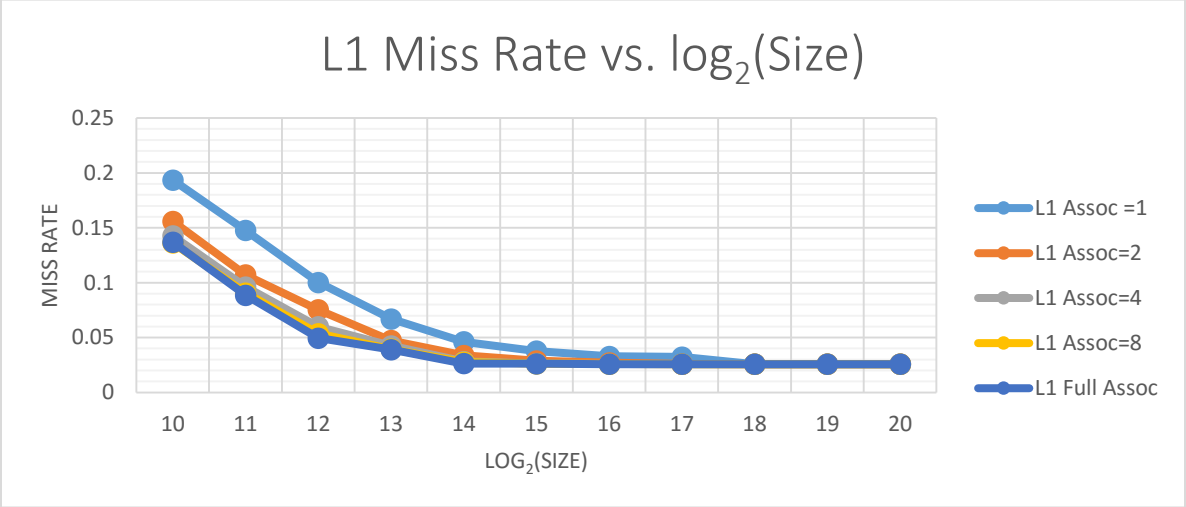
DIVYA ALOK GUPTA

NCSU Honor Pledge: "I have neither given nor received unauthorized aid on this test or assignment."

Student's electronic signature: Divya Alok Gupta
(sign by typing your name)

Course number: 521
(463 or 521 ?)

Graph #1



For a given value of cache associativity, as we increase cache size, the number of capacity and conflict misses decreases. This is because we are have ample space to store data in large cache memories. In these cases, miss rate is dominated by compulsory misses, and capacity and conflict misses contribute minimally to it.
From the graph, we observe same behavior. For given values of cache's associativity (e.g. 1), by increasing cache size, the miss rate decreases up till a point (e.g. 64K) and then flattens out to a constant value (e.g. Miss Rate = 0.0258).

For a given cache size, as we increase cache's associativity, its miss rate decreases. This happens because while we are increasing the number of blocks in a set (essentially increasing associativity), it allows the cache to store more memory blocks which map to the same set. When the cache is made fully associative, we can practically store all memory blocks which map to the same set subject to cache's capacity. Hence, in this case conflict misses will never occur (= zero). The overall miss rate is indicative of the number of compulsory and capacity misses and not of conflict misses. From the graph, we can observe the same phenomenon. For given caches, we can clearly see that as we increase associativity from 1 to fully associative, miss rate decreases and finally saturates to a constant value **0.0258**.

A fully associative cache will only face compulsory misses and capacity misses. When the cache size is large, e.g. 512KB or 1MB, the miss rate is dominated by compulsory misses than capacity misses. Therefore, from the graph we can estimate compulsory miss rate for given caches to be **2.58%.** This will hold good for all cache sizes and associativity values since compulsory misses will be common to all these cases.

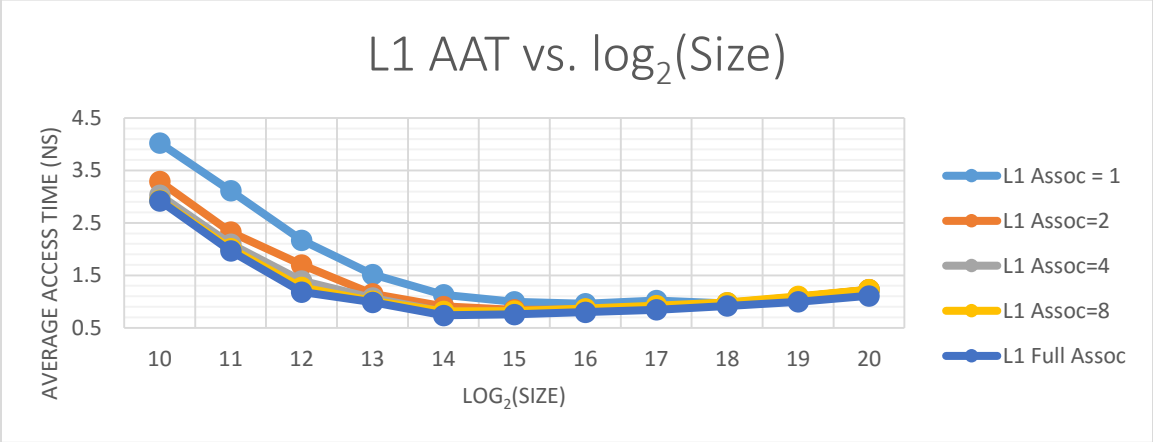
A fully associative cache will only face compulsory misses and capacity misses. We know that the estimated compulsory miss rate is 2.58%. We can estimate the capacity miss rate of each cache size by subtracting compulsory miss rate from total miss rate for the case when caches are fully associative.

Cache Size (Bytes)	Total Miss Rate (Fully Assoc.)	Compulsory Miss Rate	Capacity Miss Rate
1024	0.137	0.0258	0.1112
2048	0.0886	0.0258	0.0628
4096	0.0495	0.0258	0.0237
8192	0.0391	0.0258	0.0133
16384	0.0263	0.0258	0.0005
32768	0.0262	0.0258	0.0004
65536	0.0258	0.0258	0.0000
131072	0.0258	0.0258	0.0000
262144	0.0258	0.0258	0.0000
524288	0.0258	0.0258	0.0000
1048576	0.0258	0.0258	0.0000

Capacity misses are largely dependent on cache size and not on its associativity. Hence, the capacity miss rates calculated for a fully associative cache will hold good for caches having lower values of associativity as well.
Finally, we can use the total miss rate (calculated during cache simulations), capacity miss rate (calculated above) and compulsory miss rate (calculated in previous page) to estimate the conflict miss rate for each cache size and associativity.

Cache Size (Bytes)	Conflict Miss Rate (Assoc. = 1)	Conflict Miss Rate (Assoc. = 2)	Conflict Miss Rate (Assoc. = 4)	Conflict Miss Rate (Assoc. = 8)	Conflict Miss Rate (Fully Assoc.)
1024	0.0565	0.019	0.0057	-0.0007	0
2048	0.0591	0.0185	0.0076	0.0021	0
4096	0.0507	0.0258	0.0104	0.0041	0
8192	0.0279	0.0082	0.0034	0.0004	0
16384	0.0198	0.0075	0.002	0.0014	0
32768	0.0115	0.0026	0.0002	0	0
65536	0.0071	0.0013	0.0001	0.0001	0
131072	0.0065	0.0001	0	0	0
262144	0	0	0	0	0
524288	0	0	0	0	0
1048576	0	0	0	0	0

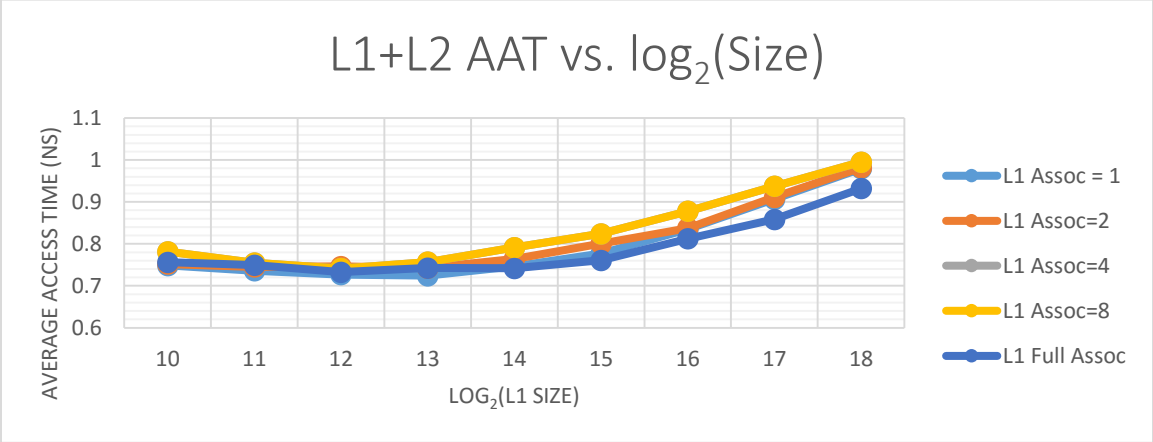
Graph #2



The best AAT is obtained when we are using L1 cache with the following parameters

- Block Size: 32 bytes, Fully Associative, Size: 16Kbytes
- AAT 0.736868 ns

Graph #3



Best configuration of L1 cache (only) yields AAT of 0.736868 ns. The following cache configurations (L1+L2) offer AAT within ±5% to the one offered by L1 cache only.

1. L1 cache (Size 1KB to 16KB Associativity 1) with L2 (Size 512KB Associativity 8)
2. L1 cache (Size 1KB to 16KB Associativity 2) with L2 (Size 512KB Associativity 8)
3. L1 cache (Size 1KB to 16KB Associativity 4) with L2 (Size 512KB Associativity 8)
4. L1 cache (Size 2KB to 8KB Associativity 8) with L2 (Size 512KB Associativity 8)
5. L1 cache (Size 1KB to 32KB Fully Associative) with L2 (Size 512KB Associativity 8)

The best AAT is offered by the following (L1+L2) cache configuration (0.724168219 ns)

- L1 Size 8KB, Associativity 1, Block Size 32Bytes

It offers 1.72% better AAT than the one offered by L1 only cache configuration.

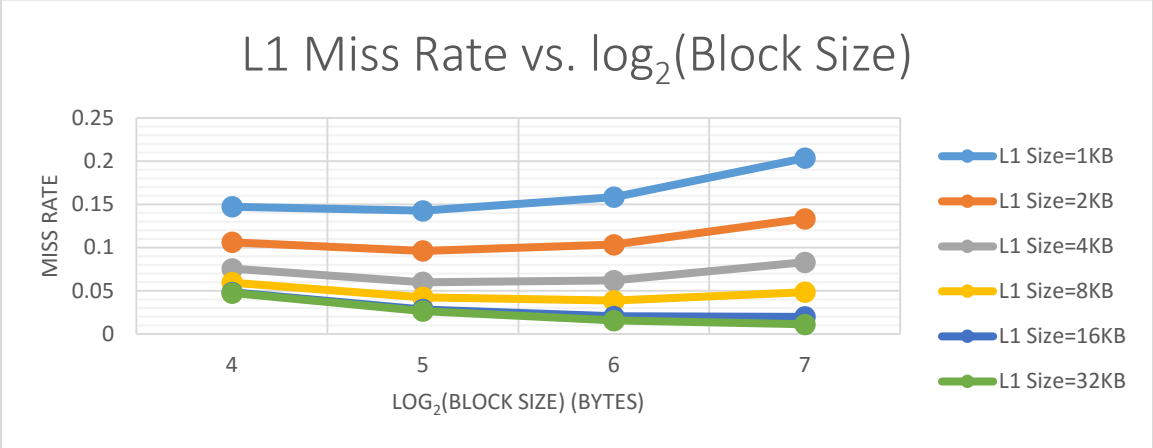
Area of L1 only configuration: 0.063446019 sq. mm

Area of L2 cache (512KB Assoc. 8) = 2.640142073492 sq. mm

L1 Configuration	Area (L1)	Total L1+L2 Area
Size 8K Assoc. 1	0.053293238	2.693435312 sq. mm

Area of L1+L2 combined configuration is **41 times higher** than the L1 (only) configuration.

Graph #4

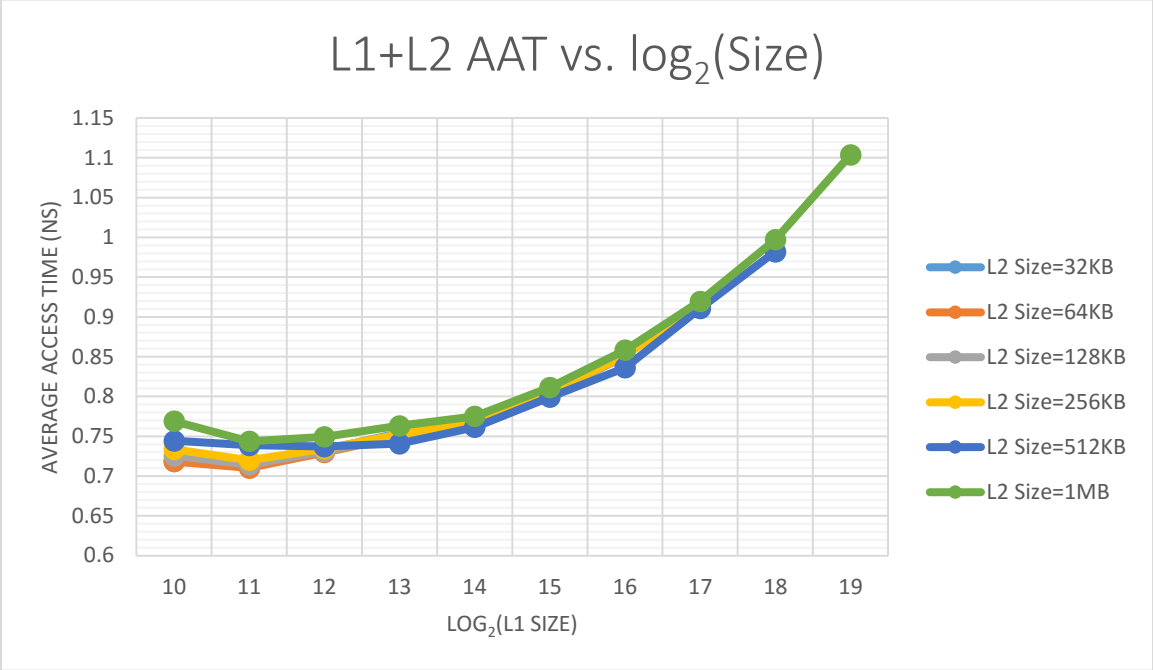


By increasing a cache's block size, we are trying to utilize spatial locality of memory references. While this will be helpful in traversing data structures such as arrays, it may increase the average access time when used with multiple branch statements. Hence, when we use small sized caches, it is preferable to use smaller block sizes where both sequential and non-sequential memory data can be suitably accommodated.
From the graph, we can observe the same behavior. Smaller caches (e.g. 1KB, 2KB) show lower miss rates when their block sizes (e.g. 16 bytes, 32 bytes) are relatively small.

The same reason used for smaller caches is valid for caches with large capacities. By using larger block size in these caches, we can store more data for sequential memory accesses. Since there is adequate space in the cache, we can also store data for non-sequential memory references. Therefore, large block sizes should be used with large sized caches.
In the graph also, we can clearly see that for large cache memories (e.g. 16KB, 32KB) miss rates decrease when the block size is increased to 64 or 128 bytes.

As block size is increased from 16 bytes to 128 bytes, we can see that for smaller cache sizes (e.g. 1KB and 2KB), the miss rate increases. This is because the cache is now polluted with data that is not required in the immediate future. On the other hand for large caches (e.g. 16KB and 32KB), the miss rate decreases as we increase the block size from 16 bytes to 128 bytes. In these cases, we are able to exploit spatial locality of memory references. There is a balance between cache pollution and spatial locality which shifts either side when we change cache size or block size.

Graph #5



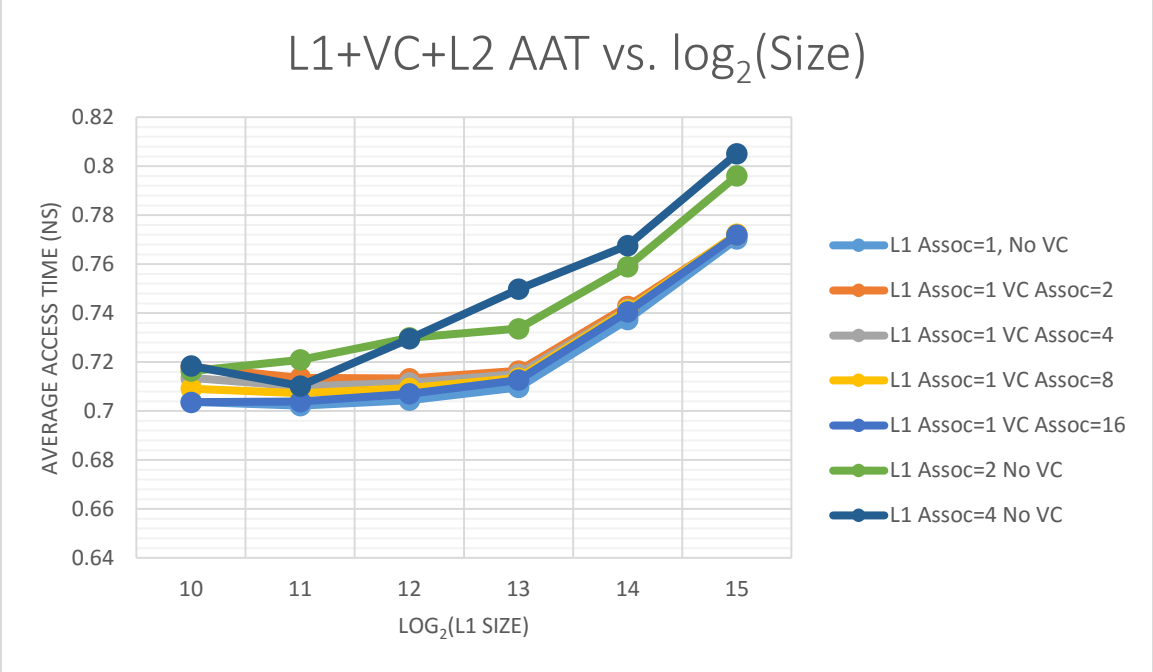
The following configuration yields the lowest AAT

- L1 Size 2KB, Associativity 4, L2 Size 64KB, Associativity 8, Block Size 32 bytes
- AAT 0.710246575 ns

The following configuration has the lowest area and yields AAT within ±5% of the above mentioned configuration

- L1 Size 1KB, Associativity 4, L2 Size 32KB, Associativity 8, Block Size 32 bytes
- Area 0.257285583 sq. mm
- AAT 0.71837788 ns

Graph #6



Hit time for a set associative cache is higher than that of a direct mapped cache because there are overheads involved while performing multiple comparisons with the memory tag. Instead of a set associative cache if we use direct mapped L1 cache with a victim cache, the combined L1+VC combination can store similar number of memory blocks which map to the same set (as that of the set associative cache). Thus, with a victim cache L1 cache projects better associativity without the overhead of higher hit time. Overall, we can expect that this configuration can offer better average access times as compared to set associative cache.

From the graph, we can see that adding a victim cache to a direct mapped L1 cache yields average access times comparable to 2-way set associative L1 cache for the following configurations

- L1 Size 1KB, Victim Cache 2-4 Blocks
- L1 Size 2KB, Victim Cache 2-4 Blocks

The following configuration has the lowest AAT (0.70220337 ns)

- L1 Size 2KB, No Victim Cache

The following configuration has the lowest area and yields AAT within ±5% of the previous configuration.

- L1 Size 1KB, Associativity 2, No Victim Cache
- Area 0.00947173 sq. mm, AAT 0.71634231 ns