MAX ROSS

CACHE SIMULATION

April 13 / 2023 / C++

Introduction

For this cache simulation, I used C++. My simulation allows for variables to be hardcoded for a trace file, cache size, block size, and associativity type. Changing these variables will change if the cache will be simulated as Direct Mapped, Fully Associative, N-way associative, FIFO, LRU, and the vary the size of the other variables. Once the program is done running, it outputs the hit rate to 8 value precision. The test results are fully based on the gcc.trace file present.

Description of Tests

What were the parameters for each test? Include the values of parameters: cache size, block size, associativity, replacement strategy.

Cache Type	Replacement	Cache Size	Block Size	Associativity
Direct	N/A	512	32	N/A
Direct	N/A	1024	32	N/A
Direct	N/A	2048	32	N/A
Direct	N/A	4096	32	N/A
Direct	N/A	8192	32	N/A
Fully	FIFO	512	32	N/A
Fully	FIFO	1024	32	N/A
Fully	FIFO	2048	32	N/A
Fully	FIFO	4098	32	N/A
Fully	FIFO	8192	32	N/A
Fully	LIFO	512	32	N/A

Fully	LIFO	1024	32	N/A
Fully	LIFO	2048	32	N/A
Fully	LIFO	4096	32	N/A
Fully	LIFO	8192	32	N/A
Set	FIFO	512	32	2
Set	FIFO	1024	32	2
Set	FIFO	2048	32	2
Set	FIFO	4098	32	2
Set	FIFO	8192	32	2
Set	LIFO	512	32	2
Set	LIFO	1024	32	2
Set	LIFO	2048	32	2
Set	LIFO	4098	32	2
Set	LIFO	8192	32	2
Set	FIFO	512	32	4
Set	FIFO	1024	32	4
Set	FIFO	2048	32	4
Set	FIFO	4098	32	4
Set	FIFO	8192	32	4
Set	LIFO	512	32	4
Set	LIFO	1024	32	4
Set	LIFO	2048	32	4
Set	LIFO	4098	32	4
Set	LIFO	8192	32	4
Set	FIFO	512	32	8
Set	FIFO	1024	32	8
Set	FIFO	2048	32	8
Set	FIFO	4098	32	8
Set	FIFO	8192	32	8

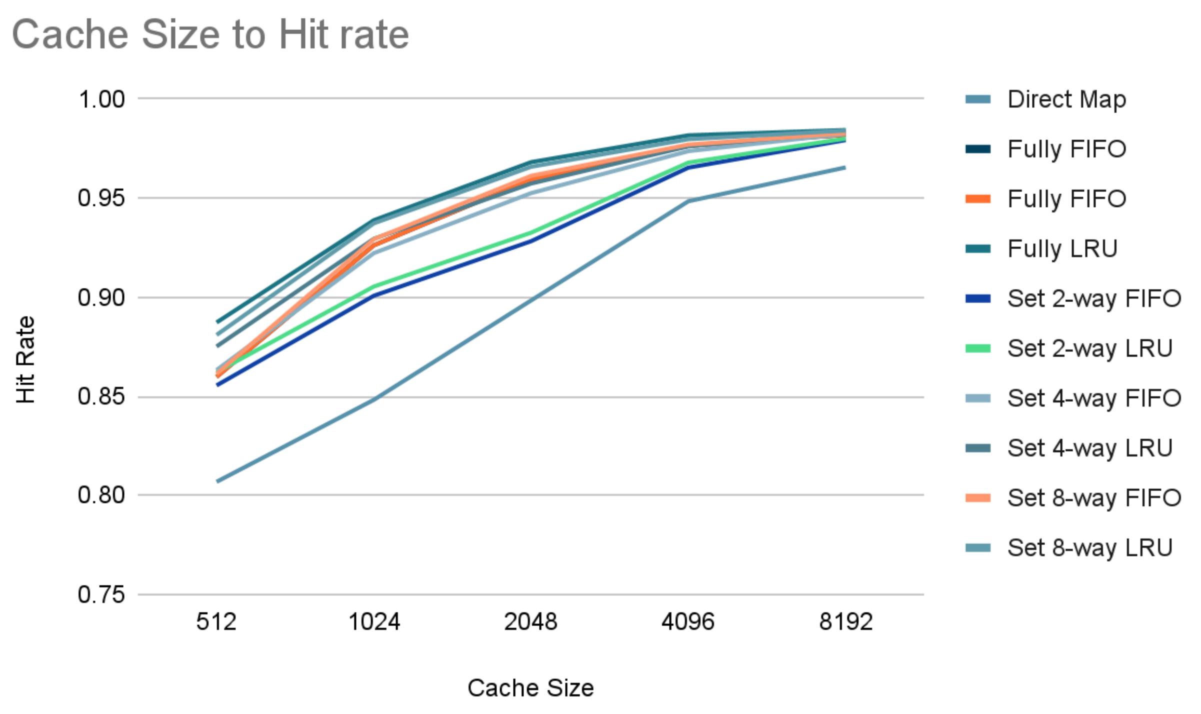
Set	LIFO	512	32	8
Set	LIFO	1024	32	8
Set	LIFO	2048	32	8
Set	LIFO	4098	32	8
Set	LIFO	8192	32	8

Why did you choose these parameters?

I chose these parameters to display how every cache design performs with different cache sizes and types. For every type of cache size, I tested each type of parameter including FIFO, LRU, Direct Map, Fully Associative, 2-way, 4-way, and 8-way associativity.

Results

Size	Direct Map	Fully FIFO	FullyLRU	Set 2-way FIFO	Set 2-way LRU	Set 4-way FIFO	Set 4-way LRU	Set 8-way FIFO	Set 8-way LRU
512	.80684451	.85980341	.88727183	.85550231	.86225646	.86325320	.87515780	.86133729	.88098115
1024	.84823428	.92621824	.93888106	.90075492	.90540313	.92226814	.92949351	.92931898	.93730451
2048	.89839301	.95938008	.96818006	.92828152	.93258261	.95254643	.95739049	.96127078	.96584724
4096	.94844507	.97645065	.98171551	.96541092	.96796869	.97373192	.97617723	.97700719	.97974919
8192	.96551951	.98191525	.98439545	.97931287	.98005946	.98249894	.98350925	.98284605	.98411815



Conclusions

What can you say about cache design (direct mapped/set associative/fully associative)?

- I have observed that the design of the cache directly impacts its performance and hit rate. In the graph, we can see that direct map has the lowest hit rate by a significant amount under every single cache size. While this type of cache is the most simple out of the other types, it shows its inefficiency when compared. This is because in a direct mapped cache, it has a limited amount of spots and it isn’t able to store many copies of data in the same cache block and this causes conflicts and misses when adding to the cache. Next, set associative caches are a lot more complicated than the direct map as they can hold many cache slots. This is why the set associative cache (all types) can be seen in the graph with a higher hit rate than the direct map. Finally, coming in with the highest hit rate is the fully associative cache. It is the most complex cache as it has to search for a victim cache block if the cache has a miss, but this allows for a higher hit rate.

What can you say about replacement policies?

- Replacement policies have an effect on cache performance. As we can see in the graph, LRU performs better than FIFO as FIFO has to replace the last block in the cache. This slows down the program as that last block could still need to be reached often. LRU replacement results in better performance as it brings the least recently used block to the front and that results in less misses.

What can you say about cache size?

- Cache sizes, when increased, result in higher hit rates overall for every testing type. This is because as the cache size is increased, there are more openings for data. The increase in cache size results in the program running slower as it has to traverse more elements and take up more space.