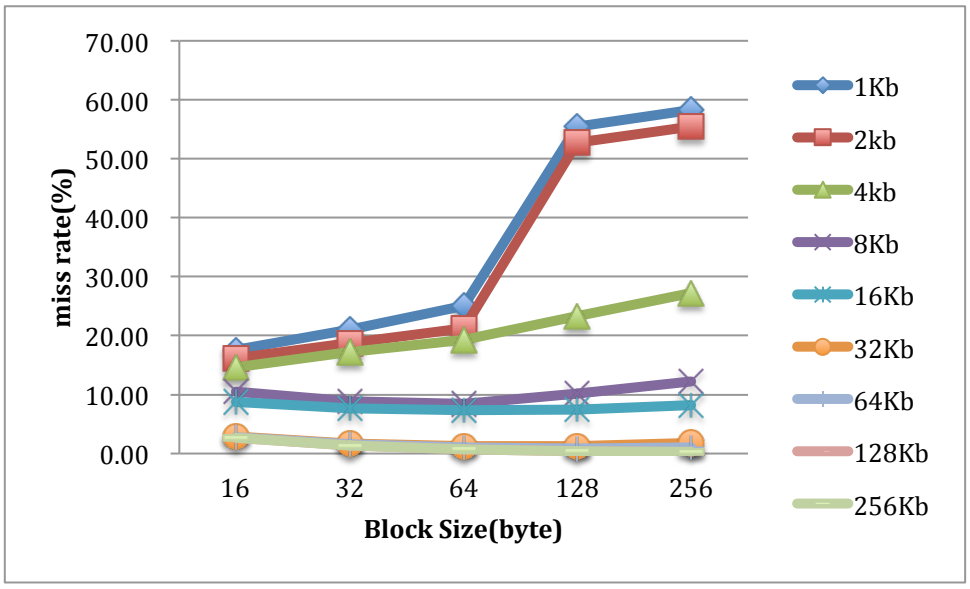
Computer Organization

Experiment result:

a) Fix the associativity on 1



Kbyte	1	2	4	8	16	32	64	128	256
b_size	miss rate(%)								
16	17.59	16.1694	14.6371	10.4853	8.72408	2.89315	2.79234	2.63105	2.63105
32	21.056	18.7126	17.2048	8.87097	7.70161	1.69499	1.55818	1.38825	1.38825
64	25.086	21.2169	19.2569	8.43174	7.35599	1.23128	1.01094	0.75173	0.75173
128	55.4709	52.7074	23.2604	10.2074	7.42944	1.2572	0.85542	0.42051	0.42051
256	58.2618	55.5098	27.2552	12.2595	8.18692	1.82892	1.05271	0.25058	0.25058

由折線圖可以看到 cache size 越大且 block size 固定時，可以讓 miss rate 越小，是因為 cache size 大 cache size 除以 block size，則 entries 越多，所以在 cache 中的 data 比較不會因為 index 相同而被取代。

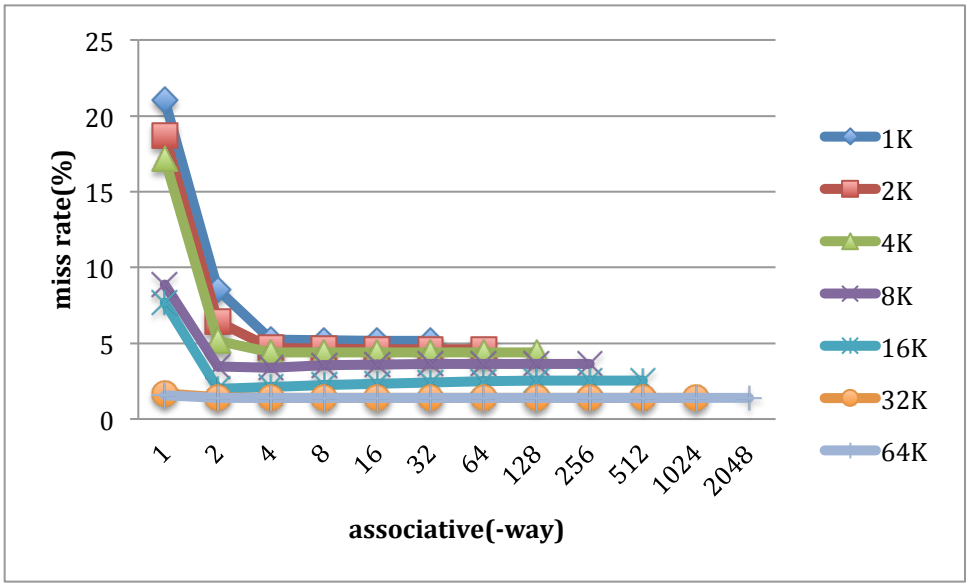
在 cache size 較小時，如 1Kb、2Kb、3Kb，當 block size

越大，entries 變小，miss rate 上升很多，因為 data address 的 index 重複的機會變大，所以更容易被取代，再次使用同個 data 時就會 miss。而且 block 在 128 256 時更明顯，因為 entries 的數量變得太少了。

但當 cache size 更大時，entries 從 cache size 上升而上升的好處變大，所以能解決 block size 變大時，entries 變少的問題，是 entries 數量仍比較足夠。

當 cache size 從 1Kb 到 8Kb 時能看到折線圖較大的差異，但在 128Kb 256Kb 的折線圖一樣。因為在讀 data 時，會有一定的 miss 量是不能避免的，所以越大 cache size 的好處差異越小。

b) Fix the block size on 32(byte)



n-way	1	2	4	8	16	32	64	128	256	512	1024	204
cachesize												
1	21.0556	8.51526	5.22321	5.18865	5.15697	5.16705						
2	18.71	6.43577	4.68606	4.64718	4.59533	4.59821	4.59533					
4	17.2048	5.17281	4.38364	4.38796	4.39948	4.39804	4.4038	4.4038				
8	8.87097	3.47062	3.38278	3.54407	3.61031	3.63623	3.64487	3.64343	3.64631			
16	7.70161	1.99165	2.1227	2.24798	2.33007	2.41359	2.51008	2.54464	2.53888	2.54608		
32	1.69499	1.45305	1.40841	1.39257	1.38825	1.38825	1.38825	1.38825	1.38825	1.38825	1.38825	
64	1.55818	1.38969	1.38825	1.38825	1.38825	1.38825	1.38825	1.38825	1.38825	1.38825	1.38825	1.3882
128	1.38825											
256	1.38825											

當 cache size 上升 entries 越小，miss rate 因次下降。Associative 越大則，可以看到 miss rate 下降很多，address 要放到相對應 index 的 block，如果是 1-way 的話，相同的 index 的就只能有一個 block，2-way 就兩個，以此類推。若是原本 1-way 的話，不同 data 但有同樣 index 依序被讀時，就會要取代，再次讀取會增加 miss，所以越大的 associative 可以讓 miss rate 相較下降很多。

當 cache size 為 128Kb、256Kb 時，不管是多少 associative 的 miss rate 都是 1.38825%，因此我認為那是最優化的 miss rate 不會在下降了。故也沒有放到折線圖中。

Problems you met and solutions:

LRU 的部分想比較久。最後寫了一個變數 w、current_w，給每一個 cache 裡的資料設權重，讓他在讀取資料時，每次都會+1，當要被 replace 時，比較權重。若權重就大則代表越近的時候

使用過。

Debug 的時候，因為 Trace.txt 的檔案較大，不可能輸出 miss/hit instruction 來找 bug 而且很容易頭腦打結。Trace1.txt 檔案又太小，其實蠻難 debug 的，還好最後有用出來了。

再來是檔案輸出時，原本想要寫一個 array 全部一起跑完，但這樣一次其實會輸出很多，會眼花繚亂，而且第二題，每個的 fully associate 也不太一樣。所以一題一題 comment 掉在輸出。也因為這樣 cpp 檔案有點亂亂的。另外輸出的部分都有放在 excel 檔中了。

Summary:

這學期幾乎都在寫 verilog，終於在最後碰到 c++，此次 lab 也是應用老師給的知識，然後 n-way 的時候要再想一下。其他都蠻順利的。開心終於做完了。