DirectMap Cache Report

# Abstract:

This report relays the effect of spatial locality on DirectMap Cache by examining the effect of different parameters on a simulated DirectMap Cache written in C++ code. The simulator is fed the memory access requested by a certain program and then it mimics the response of a DirectMap to these requests, and finally prints the responses of interest, mainly misses and hits of data in the cache among other things. To examine the effect of spatial locality, two different programs were considered to be simulated by the simulator, one that respects spatial locality, horizontal access array, and another that violates spatial locality, vertical access array. Each program will then be simulated with two cache write policies namely Write-back and Write-through. For every write policy the DirectMap simulator will run the program in two different cache sizes, 512 and 1024. In each cache size trial, the size of the cache lines is going to be changes to also view the effect of such parameter on the total number of hits and misses. It is assumed that as one increases the cache line size, the cycles taken to load it from memory to cache increases and this happens each time the cacheline size increases twice, for example the time taken to load 4 bytes is the same as 8 bytes but is less than 16 bytes cacheline size. The other cycles are adjusted accordingly so that they all follow the same ratio provided in the Assignment6 pdf file. The outputs of every simulations is presented in the results section below, then the results are analyzed and interpreted with the help of graphs highlighting the most important features.

# Results:

## Horizontal access array:

Constants in all simulations for this program are:

1. Memory Read Access Attempts = 16863316
2. Memory Write Access Attempts = 4751935

### WB:

#### 512-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 2965754 | 18649497 | 2965754 | 1632494 | 24922968 |
| 2 | 1 | 2 | 4 | 8 | 2817579 | 18797672 | 2817579 | 1612117 | 24167252 |
| 4 | 2 | 4 | 8 | 16 | 2740598 | 18874653 | 2740598 | 1600233 | 47528512 |
| 8 | 2 | 4 | 8 | 16 | 1665367 | 19949884 | 1665367 | 1073172 | 30493688 |
| 16 | 4 | 8 | 16 | 32 | 855628 | 20759623 | 855628 | 550133 | 31294304 |
| 32 | 4 | 8 | 16 | 32 | 427306 | 21187945 | 427306 | 272980 | 15572256 |
| 64 | 8 | 16 | 32 | 64 | 219525 | 21395726 | 219525 | 139775 | 15970400 |
| 128 | 8 | 16 | 32 | 64 | 118400 | 21496851 | 118400 | 73759 | 8509376 |
| 256 | 16 | 32 | 64 | 128 | 67960 | 21547291 | 67960 | 41040 | 9602560 |
| 512 | 16 | 32 | 64 | 128 | 42668 | 21572583 | 42668 | 24622 | 5882368 |
| 1024 | 32 | 64 | 128 | 256 | 30083 | 21585168 | 30083 | 16446 | 8060800 |

#### 1024-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 2818197 | 18797054 | 2818197 | 1612102 | 24169604 |
| 2 | 1 | 2 | 4 | 8 | 2743166 | 18872085 | 2743166 | 1600223 | 23774448 |
| 4 | 2 | 4 | 8 | 16 | 2698420 | 18916831 | 2698420 | 1590273 | 47031728 |
| 8 | 2 | 4 | 8 | 16 | 1381397 | 20233854 | 1381397 | 812411 | 24049752 |
| 16 | 4 | 8 | 16 | 32 | 812859 | 20802392 | 812859 | 529326 | 29944176 |
| 32 | 4 | 8 | 16 | 32 | 409817 | 21205434 | 409817 | 264742 | 15028816 |
| 64 | 8 | 16 | 32 | 64 | 208588 | 21406663 | 208588 | 134266 | 15267840 |
| 128 | 8 | 16 | 32 | 64 | 108942 | 21506309 | 108942 | 69119 | 7909760 |
| 256 | 16 | 32 | 64 | 128 | 58975 | 21556276 | 58975 | 36571 | 8455488 |
| 512 | 16 | 32 | 64 | 128 | 34170 | 21581081 | 34170 | 20388 | 4796544 |
| 1024 | 32 | 64 | 128 | 256 | 21731 | 21593520 | 21731 | 12260 | 5920128 |

### WT:

#### 512-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 2965754 | 18649497 | 2965754 | 4751935 | 49878496 |
| 2 | 1 | 2 | 4 | 8 | 2817579 | 18797672 | 2817579 | 4751935 | 49285796 |
| 4 | 2 | 4 | 8 | 16 | 2740598 | 18874653 | 2740598 | 4751935 | 97955744 |
| 8 | 2 | 4 | 8 | 16 | 1665367 | 19949884 | 1665367 | 4751935 | 89353896 |
| 16 | 4 | 8 | 16 | 32 | 855628 | 20759623 | 855628 | 4751935 | 165751968 |
| 32 | 4 | 8 | 16 | 32 | 427306 | 21187945 | 427306 | 4751935 | 158898816 |
| 64 | 8 | 16 | 32 | 64 | 219525 | 21395726 | 219525 | 4751935 | 311148640 |
| 128 | 8 | 16 | 32 | 64 | 118400 | 21496851 | 118400 | 4751935 | 307912640 |
| 256 | 16 | 32 | 64 | 128 | 67960 | 21547291 | 67960 | 4751935 | 612597120 |
| 512 | 16 | 32 | 64 | 128 | 42668 | 21572583 | 42668 | 4751935 | 610978432 |
| 1024 | 32 | 64 | 128 | 256 | 30083 | 21585168 | 30083 | 4751935 | 1220345984 |

#### 1024-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 2818197 | 18797054 | 2818197 | 4751935 | 49288268 |
| 2 | 1 | 2 | 4 | 8 | 2743166 | 18872085 | 2743166 | 4751935 | 48988144 |
| 4 | 2 | 4 | 8 | 16 | 2698420 | 18916831 | 2698420 | 4751935 | 97618320 |
| 8 | 2 | 4 | 8 | 16 | 1381397 | 20233854 | 1381397 | 4751935 | 87082136 |
| 16 | 4 | 8 | 16 | 32 | 812859 | 20802392 | 812859 | 4751935 | 165067664 |
| 32 | 4 | 8 | 16 | 32 | 409817 | 21205434 | 409817 | 4751935 | 158618992 |
| 64 | 8 | 16 | 32 | 64 | 208588 | 21406663 | 208588 | 4751935 | 310798656 |
| 128 | 8 | 16 | 32 | 64 | 108942 | 21506309 | 108942 | 4751935 | 307609984 |
| 256 | 16 | 32 | 64 | 128 | 58975 | 21556276 | 58975 | 4751935 | 612022080 |
| 512 | 16 | 32 | 64 | 128 | 34170 | 21581081 | 34170 | 4751935 | 610434560 |
| 1024 | 32 | 64 | 128 | 256 | 21731 | 21593520 | 21731 | 4751935 | 1219276928 |

## Vertical access array:

Constants in all simulations for this program are:

1. Memory Read Access Attempts = 16863281
2. Memory Write Access Attempts = 4751935

### WB:

#### 512-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 3905916 | 17709300 | 3905916 | 1631295 | 28674024 |
| 2 | 1 | 2 | 4 | 8 | 3812758 | 17802458 | 3812758 | 1611977 | 28146848 |
| 4 | 2 | 4 | 8 | 16 | 3760542 | 17854674 | 3760542 | 1599616 | 55678192 |
| 8 | 2 | 4 | 8 | 16 | 3215622 | 18399594 | 3215622 | 1073247 | 42896928 |
| 16 | 4 | 8 | 16 | 32 | 2155945 | 19459271 | 2155945 | 550219 | 52102128 |
| 32 | 4 | 8 | 16 | 32 | 1601066 | 20014150 | 1601066 | 272402 | 34333920 |
| 64 | 8 | 16 | 32 | 64 | 1331562 | 20283654 | 1331562 | 139888 | 51562816 |
| 128 | 8 | 16 | 32 | 64 | 1196664 | 20418552 | 1196664 | 73846 | 43019392 |
| 256 | 16 | 32 | 64 | 128 | 1127240 | 20487976 | 1127240 | 41084 | 77402112 |
| 512 | 16 | 32 | 64 | 128 | 1093426 | 20521790 | 1093426 | 24645 | 73133824 |
| 1024 | 32 | 64 | 128 | 256 | 1076505 | 20538711 | 1076505 | 16469 | 142008704 |

#### 1024-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 3813375 | 17801841 | 3813375 | 1611959 | 28149172 |
| 2 | 1 | 2 | 4 | 8 | 3763126 | 17852090 | 3763126 | 1599610 | 27849384 |
| 4 | 2 | 4 | 8 | 16 | 3732935 | 17882281 | 3732935 | 1590292 | 55308152 |
| 8 | 2 | 4 | 8 | 16 | 2551488 | 19063728 | 2551488 | 813108 | 33421632 |
| 16 | 4 | 8 | 16 | 32 | 1694919 | 19920297 | 1694919 | 528723 | 44037840 |
| 32 | 4 | 8 | 16 | 32 | 1137652 | 20477564 | 1137652 | 264907 | 26679456 |
| 64 | 8 | 16 | 32 | 64 | 1045759 | 20569457 | 1045759 | 134256 | 42056672 |
| 128 | 8 | 16 | 32 | 64 | 1038264 | 20576952 | 1038264 | 69137 | 37649216 |
| 256 | 16 | 32 | 64 | 128 | 1042826 | 20572390 | 1042826 | 36590 | 71424384 |
| 512 | 16 | 32 | 64 | 128 | 1048155 | 20567061 | 1048155 | 20360 | 69688000 |
| 1024 | 32 | 64 | 128 | 256 | 1051389 | 20563827 | 1051389 | 12286 | 137723008 |

### WT:

#### 512-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 3905916 | 17709300 | 3905916 | 4751935 | 53639144 |
| 2 | 1 | 2 | 4 | 8 | 3812758 | 17802458 | 3812758 | 4751935 | 53266512 |
| 4 | 2 | 4 | 8 | 16 | 3760542 | 17854674 | 3760542 | 4751935 | 106115296 |
| 8 | 2 | 4 | 8 | 16 | 3215622 | 18399594 | 3215622 | 4751935 | 101755936 |
| 16 | 4 | 8 | 16 | 32 | 2155945 | 19459271 | 2155945 | 4751935 | 186557040 |
| 32 | 4 | 8 | 16 | 32 | 1601066 | 20014150 | 1601066 | 4751935 | 177678976 |
| 64 | 8 | 16 | 32 | 64 | 1331562 | 20283654 | 1331562 | 4751935 | 346733824 |
| 128 | 8 | 16 | 32 | 64 | 1196664 | 20418552 | 1196664 | 4751935 | 342417088 |
| 256 | 16 | 32 | 64 | 128 | 1127240 | 20487976 | 1127240 | 4751935 | 680391040 |
| 512 | 16 | 32 | 64 | 128 | 1093426 | 20521790 | 1093426 | 4751935 | 678226944 |
| 1024 | 32 | 64 | 128 | 256 | 1076505 | 20538711 | 1076505 | 4751935 | 1354288000 |

#### 1024-cache-size-map:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache  Line size | CRC | CWC | MRC | MWC | Total Misses | Total Hits | Memory Read Access | Memory Write Access | Total Memory  Access  cycles |
| 1 | 1 | 2 | 4 | 8 | 3813375 | 17801841 | 3813375 | 4751935 | 53268980 |
| 2 | 1 | 2 | 4 | 8 | 3763126 | 17852090 | 3763126 | 4751935 | 53067984 |
| 4 | 2 | 4 | 8 | 16 | 3732935 | 17882281 | 3732935 | 4751935 | 105894440 |
| 8 | 2 | 4 | 8 | 16 | 2551488 | 19063728 | 2551488 | 4751935 | 96442864 |
| 16 | 4 | 8 | 16 | 32 | 1694919 | 19920297 | 1694919 | 4751935 | 179180624 |
| 32 | 4 | 8 | 16 | 32 | 1137652 | 20477564 | 1137652 | 4751935 | 170264352 |
| 64 | 8 | 16 | 32 | 64 | 1045759 | 20569457 | 1045759 | 4751935 | 337588128 |
| 128 | 8 | 16 | 32 | 64 | 1038264 | 20576952 | 1038264 | 4751935 | 337348288 |
| 256 | 16 | 32 | 64 | 128 | 1042826 | 20572390 | 1042826 | 4751935 | 674988544 |
| 512 | 16 | 32 | 64 | 128 | 1048155 | 20567061 | 1048155 | 4751935 | 675329600 |
| 1024 | 32 | 64 | 128 | 256 | 1051389 | 20563827 | 1051389 | 4751935 | 1351073152 |

# Analysis:

In this section will examine the effect of each parameter on making the DirectMap more tolerant towards spatial locality.

## Effect of cacheline size:

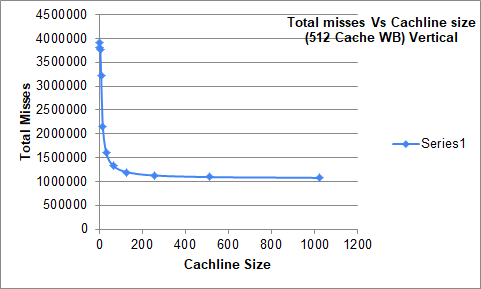
These two graphs show that the number of misses decreases and the number hits consequently increase when we increase the cacheline size

### Horizontal access array:

Also the same pattern exactly follows in the WT policy of horizontal access.

This show that regardless of the cache size or the policy increasing the cacheline size will reduce the misses hence make the cache more tolerant to spatial locality.

### Vertical access array:



Not only is this pattern observed in the horizontal access file but it also holds true for the vertical access program. Thus we conclude that increasing cacheline size in general has a positive effect on spatial locality.

## Effect of cache size:

The effect of cache size is observed in each program, horizontal and vertical. Again we only consider the WB policy since the misses and hits are identical for same cache sizes in the same program, thus by only inspecting one writing policy we can generalize over the other.

One can observe that as the cache size since increase the misses decrease more and the hits increase more compared to a smaller cache. This concludes that increasing the cache size increase the Directmap tolerance for spatial locality since it decreases the misses in both programs. It is to be noted that the same pattern happens in the horizontal access program; however the difference in misses is not as significant as here hence a graph won’t emphasize the effect. The pattern can be checked in the results though.

## Effect of cache-write policy:

As per the results section the cache-write policies have no effect on the misses or the hits in both programs for the same cache size and cacheline size. However they play another important role in changing the number of accesses to memory to write in it when the cache is written.

These graphs below show the effect:

From these two graphs it is shown that WB policy has less memory write accesses and that these accesses decrease as we increase the cacheline size. On the other hand WT has way more memory write accesses that are constant. Misses have negative effect on spatial locality since they require accessing memory to fetch the new tag which delays the program. WT policy have the same effect since it access the memory more than WB hence delay the program more thus we can assume that WB has a more positive effect on spatial locality than WT.

It is to be noted that the same relation also hold for the vertical access program this can be observed from the results table thus we can generalize this conclusion on both.

## Effect of program run:

Here it is noted that the misses in the vertical access array is way more than that in horizontal access array in general for different cacheline sizes and for different cache sizes also since the pattern is apparent in both graphs. Since cache writing policies have no effect on misses of the same cache size and same cacheline size we can conclude that in general the vertical access array program as a negative effect on spatial locality and that the horizontal has a positive effect. This shows that a program can respect or violate the spatial locality regardless of the design of the cache.

# Interpretation:

In this section the data fetched in results and the conclusions reached in the analysis are interpreted according to the logic of the DirectMap cache simulator.

## Interpreting Data:

### Constant Data in the results:

It is noted that the Memory Read Access Attempts and Memory Write Access Attempts are constant in each program.

This is because these attempts are actually the read and the write to the cache, these instructions come from the program itself hence they are unique to it and no matter how many times we run the simulator the program will ask for the same memory addresses to read and write that is why they are constant for each program (in the Pinatrace file there is a constant number of lines that has “R” and lines that has “W” the numbers of these lines doesn’t change by changing parameters of the cache)

### Memory Reads:

It can be also noted the memory reads always equal the number of misses this is because we assume in this simulator that memory read means load the block from memory and put it in in cache, thus whenever there is a miss there is a read from memory so both numbers are always equal.

### Memory Writes:

In all the WT policies for the same program, it is observed that the memory writes are constant. Not only that they also equal the number of cache writes. This is because by definition the write through policy means writing the memory each time the program writes the cache thus these numbers are always equal under the WT policy hence it is constant since writes to caches are constant for the same program as explained earlier.

## Interpreting Conclusions:

### First conclusion:

Since the cacheline size represent how many bytes there are in a cacheline then it only makes since it has positive effects on locality. Since if we have already accessed a byte by loading its block to cache then it is subsequent byte is there in the cacheline also (if the size of the cacheline permits). And since in spatial locality by definition we need to access subsequent bytes, then having a larger cacheline increases the probability we have that byte already in cache hence decreases misses and has a positive effect on spatial locality as there is less accessing to memory is required.

### Second conclusion:

For this we consider we have a DirectMap that is full, all possible places (indices) have loaded cachelines. Then any address from the program with a different tag will result in a miss. Having a larger cache size is will increase the number of addresses from the program needed to fill the cache hence will help decrease the misses more than a smaller cache will do. That is why the cache size showed positive effect on spatial locality since more subsequent bytes can be loaded since there are more cachelines that can support them.

### Third conclusion:

Cache write policies have no direct effect on misses however WB still has positive effect on spatial locality since in spatial locality we load subsequent bytes so probabilities of misses are lower and since WB only access memory in misses then it will make the program faster since we need to access memory less. If there is no spatial locality there will be lots of misses in the cache thus the effect of WB on the program won’t be as obvious. Thus one can assume that WB results in better access cycles for spatial locality and that is their positive effect.

### Fourth conclusion:

Some programs violate the spatial locality by nature in the sense that they call for memory addresses that are very far apart thus the tags of the addresses is different from the ones already loaded, thus even if the index of the new memory address matches one of loaded the tag is different hence this will result in a miss. We note that the vertical access program implements this since subsequent bytes are in the same row and the program access a new row each time thus the probability that the byte read is in the same block is very low since the byte is very far away from all the previous bytes.