Assignment 3
Bottleneck Analysis using Tejas

Om Patil (200010036)
Pranav Talegaonkar (200010041)

Contents

Task

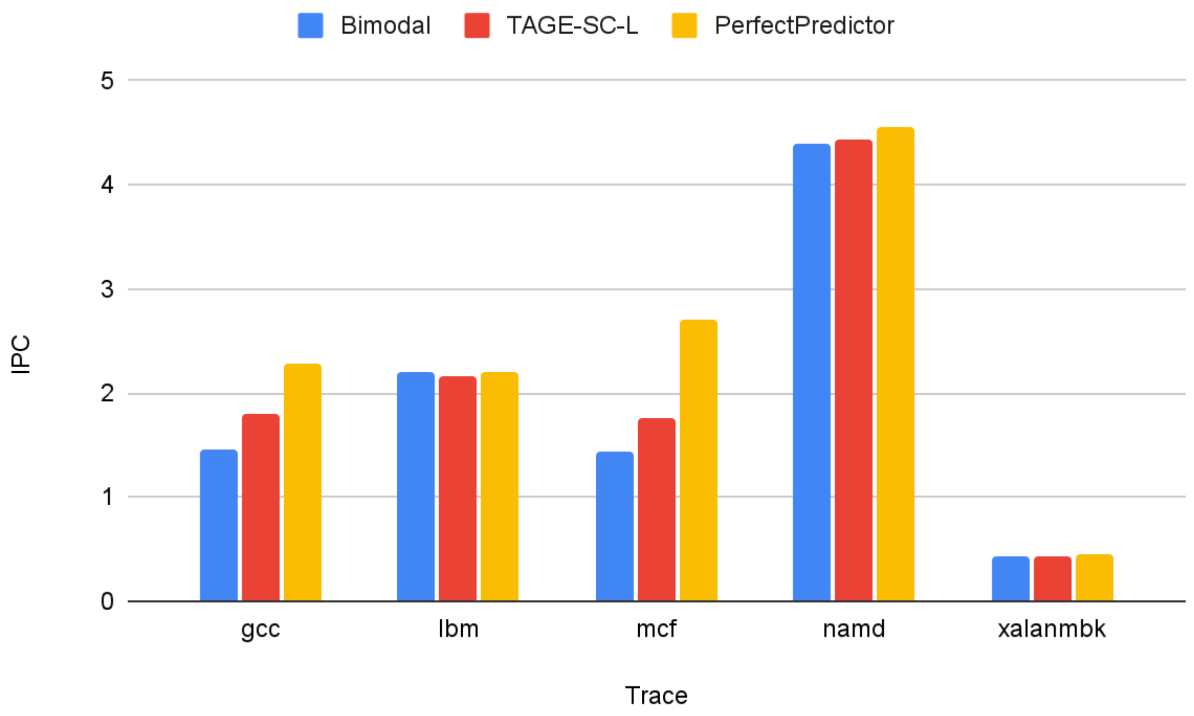
The task is to identify the bottlenecks for various SPEC CPU2017 benchmarks and to identify the parameters that affect the performance of the benchmarks the greatest.

Experiments

To determine the bottlenecks for various SPEC CPU2017 benchmarks, we run various experiments that measure the benchmark performance as we vary specific parameters of the processor. As the IPC of the processor is directly proportional to the performance of the benchmark and the processor frequency and number of instructions in the benchmark are fixed, the IPC can be taken as a measure of performance.

Details of individual experiments are described below.

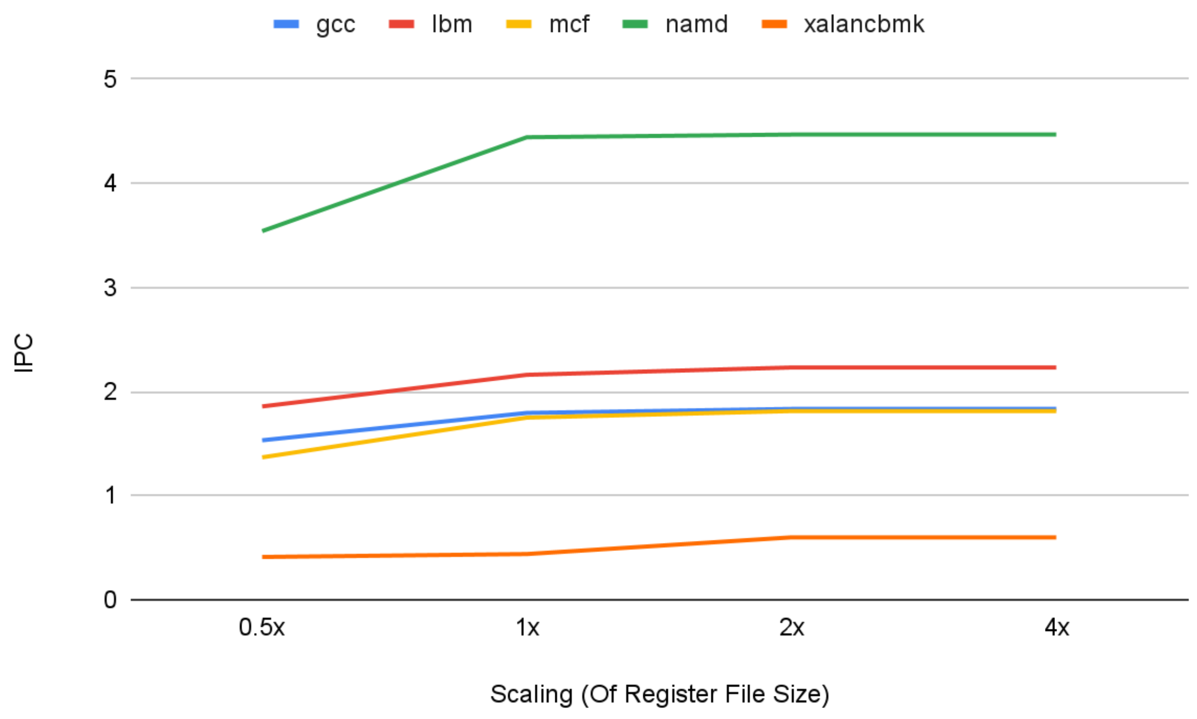
Branch Predictor



In this experiment, we change the branch predictor used to see how it affects the performance of the benchmarks. We also compared this to the performance of a Perfect Branch Predictor to understand if bottlenecks due to control hazards are encountered.

As we can see, the gcc and mcf benchmarks are affected heavily by branch prediction accuracy, i.e. they are bottlenecked by it. On the other hand, namd is slightly affected by branch prediction accuracy, i.e. it is bottlenecked by it to a lesser extent. lbm and xalancbmk are barely affected and are not bottlenecked by branch prediction accuracy.

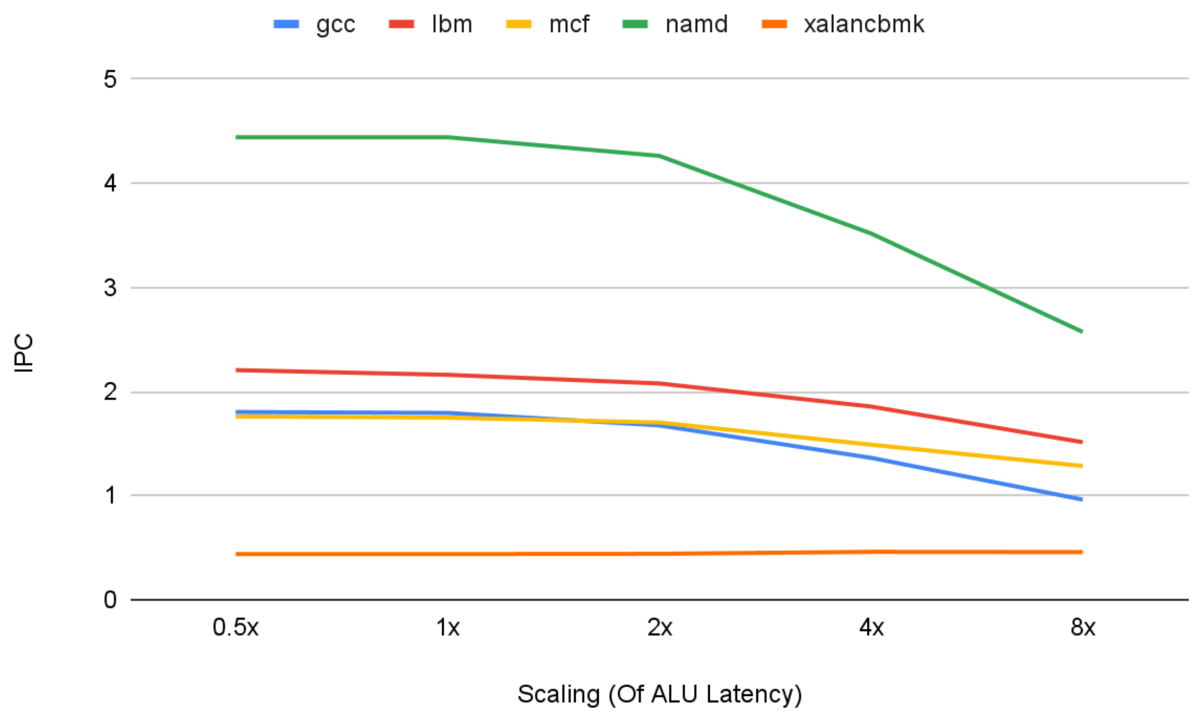
Register File Size



In this experiment, we change the size of the register file to see how it affects the performance of the benchmarks. We vary the register file size by scaling the number of physical registers by a factor of 0.5, 1, 2 and 4.

As we can see, the gcc, lbm, mcf and namd benchmarks are affected by the size of the register file till a scaling factor of 1x, and xalancbmk is affected till a scaling factor of 2x. Hence, we can conclude that gcc, lbm, mcf and namd are bottlenecked by the register file size, while xalancbmk is bottlenecked by it to a greater extent.

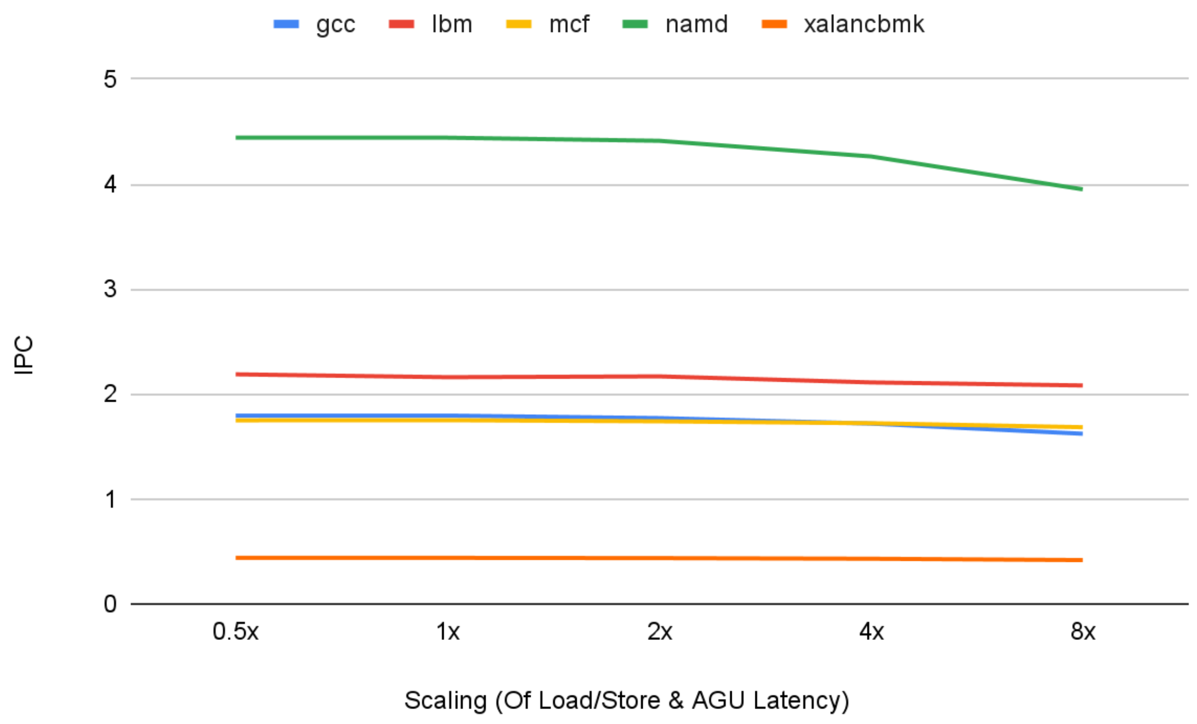
ALU Latency



In this experiment, we change the latency of the ALU to see how it affects the performance of the benchmarks. We vary the ALU latency by scaling the ALU latency by a factor of 0.5, 1, 2, 4 and 8.

It can be seen that namd and gcc take a performance hit post a scaling factor of 1x, while lbm and mcf take a performance hit post a scaling factor of 2x. xalanmbk is not affected by the ALU latency. Hence, we can conclude that namd and gcc are significantly bottlenecked by the ALU latency, while lbm and mcf are bottlenecked by it to a lesser extent. In contrast, xalancbmk is not bottlenecked by the ALU latency.

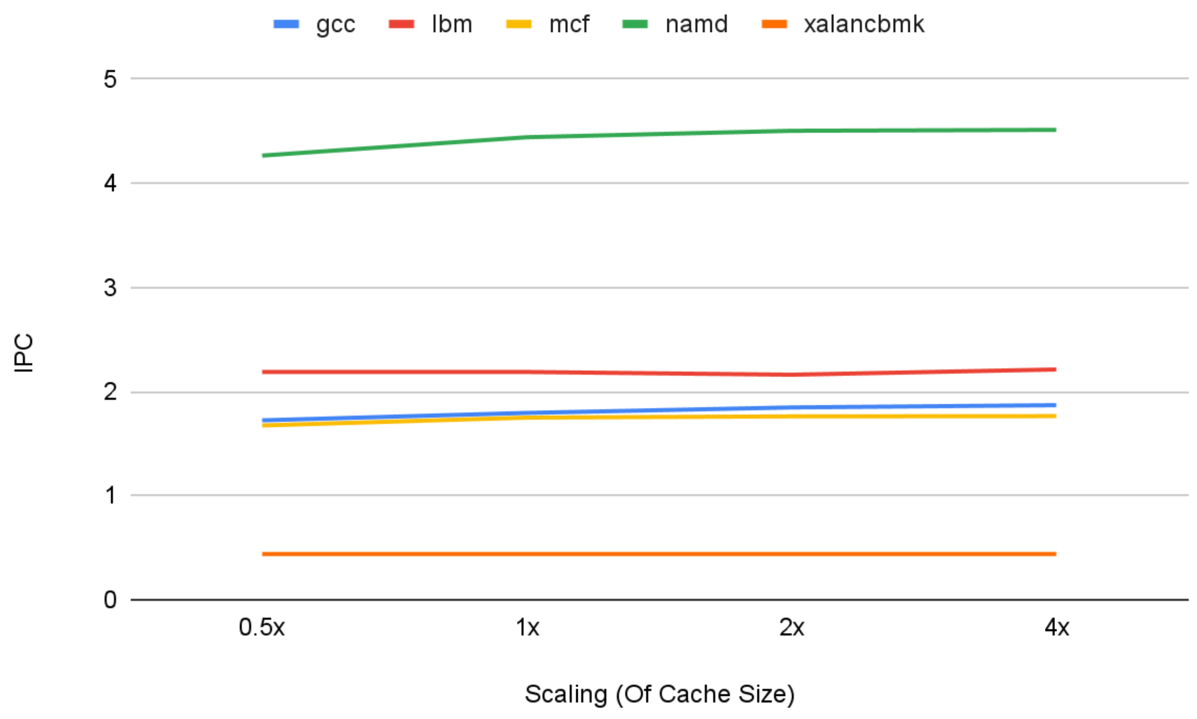
Load/Store & AGU Latency



In this experiment, we change the latency of the Load/Store and AGU to see how it affects the performance of the benchmarks. We vary the Load/Store and AGU latency by scaling the latency by a factor of 0.5, 1, 2, 4 and 8.

It can be seen that namd and gcc take a performance hit post a scaling factor of 2x, while lbm and mcf take a slight performance hit post a scaling factor of 4x. xalanmbk is not affected by the Load/Store and AGU latency. Hence, we can conclude that namd and gcc are significantly bottlenecked by the Load/Store and AGU latency, while lbm and mcf are very slightly bottlenecked by it. In contrast, xalancbmk is not bottlenecked by the Load/Store and AGU latency.

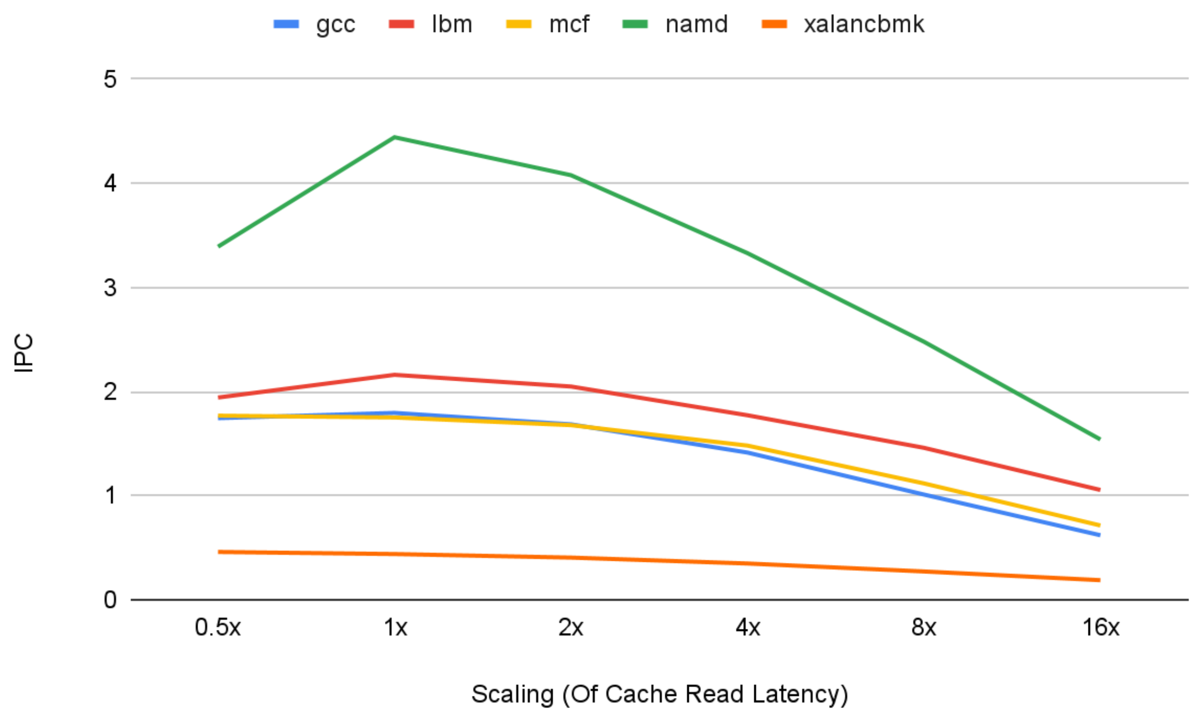
Cache Size



In this experiment, we change the size of the L1 & L2 caches to see how it affects the performance of the benchmarks. We vary the cache size by scaling the cache size by a factor of 0.125, 0.25, 0.5, 1, 2, and 4.

It can be seen that namd and gcc are reasonably affected by the L1 & L2 cache size below a scaling factor of 1x, while lbm and mcf are only slightly affected below a scaling factor of 1x. xalanmbk is not affected by the L1 & L2 cache size. Hence, we can conclude that namd and gcc are bottlenecked by the L1 & L2 cache size, while lbm and mcf are slightly bottlenecked by it. In contrast, xalancbmk is not bottlenecked by the L1 & L2 cache size.

Cache Read Latency

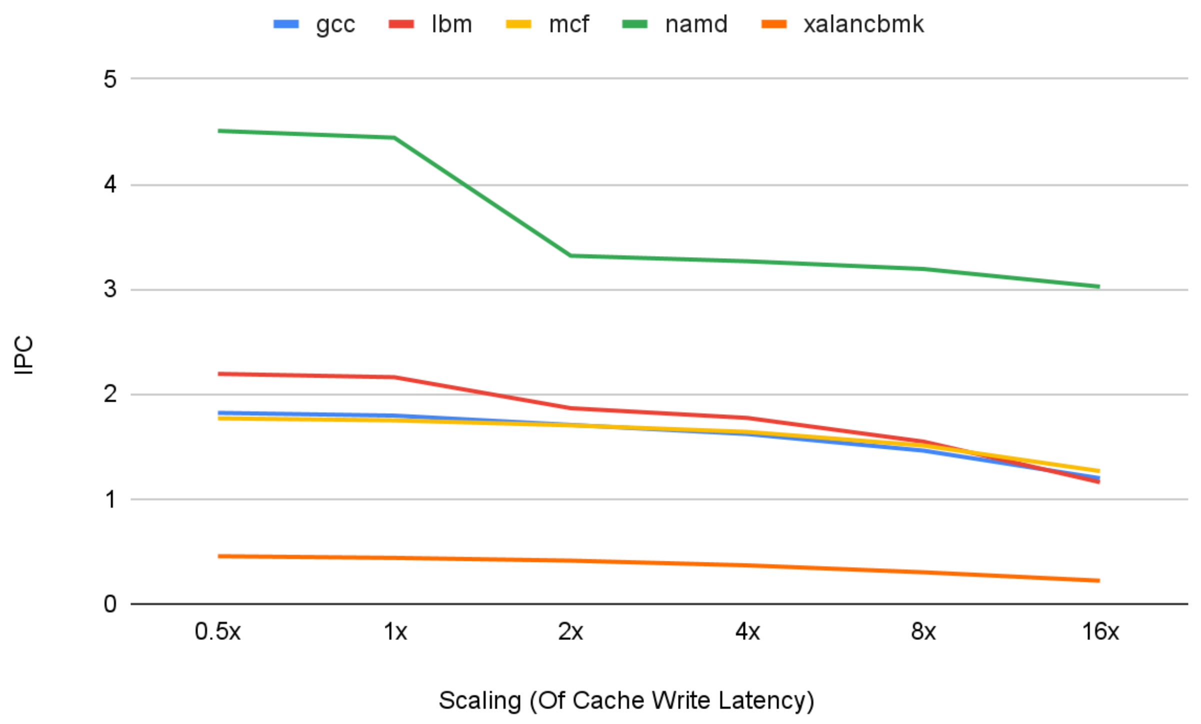


In this experiment, we change the latency of the L1 & L2 caches to see how it affects the performance of the benchmarks. We vary the latency by scaling the latency by a factor of 0.5, 1, 2, 4, 8 and 16.

It can be seen that namd and gcc take a performance hit post a scaling factor of 1x, while lbm, mcf and xalancbmk take a slight performance hit post a scaling factor of 1x. Hence, we can conclude that namd and gcc are bottlenecked by the L1 & L2 cache latency, while lbm, mcf and xalancbmk are slightly bottlenecked by it.

A peculiar observation is that gcc, lbm and namd performance drops when the L1 & L2 cache latency is scaled by a factor of 0.5x. We believe that this is because the L3 cache is not scaled hence, due to backpressure, the L1 & L2 caches are not able to operate at their full capacity.

Cache Write Latency



In this experiment, we change the latency of the L1 & L2 caches to see how it affects the performance of the benchmarks. We vary the latency by scaling the latency by a factor of 0.5, 1, 2, 4, 8 and 16.

It can be seen that namd and lbm take a significant performance hit post a scaling factor of 1x, while gcc, mcf, and xalancbmk take a slight performance hit post a scaling factor of 1x. Hence, we can conclude that namd and lbm are bottlenecked by the L1 & L2 cache latency, while gcc, mcf and xalancbmk are slightly bottlenecked by it.

Conclusion

gcc

From the above experiments, we can conclude that gcc is significantly bottlenecked by the following:

- Branch Prediction Accuracy
- ALU Latency
- Load/Store & AGU Latency
- Cache Size

- Cache Read Latency

gcc is slightly bottlenecked by the following:
- Register File Size
- Cache Write Latency

lbm

From the above experiments, we can conclude that lbm is significantly bottlenecked by the following:
- Cache Write Latency

lbm is slightly bottlenecked by the following:
- Register File Size
- ALU Latency
- Load/Store & AGU Latency
- Cache Size
- Cache Read Latency

lbm is not bottlenecked by the following:
- Branch Prediction Accuracy

mcf

From the above experiments, we can conclude that mcf is significantly bottlenecked by the following:
- Branch Prediction Accuracy

mcf is slightly bottlenecked by the following:
- Register File Size
- ALU Latency
- Load/Store & AGU Latency
- Cache Size
- Cache Read Latency
- Cache Write Latency

namd

From the above experiments, we can conclude that namd is significantly bottlenecked by the following:
- Register File Size
- ALU Latency
- Load/Store & AGU Latency

- Cache Size
- Cache Read Latency
- Cache Write Latency

namd is slightly bottlenecked by the following:
- Branch Prediction Accuracy

xalancbmk

From the above experiments, we can conclude that xalancbmk is significantly bottlenecked by the following:
- Register File Size

xalancbmk is slightly bottlenecked by the following:
- Cache Read Latency
- Cache Write Latency

xalancbmk is not bottlenecked by the following:
- Branch Prediction Accuracy
- ALU Latency
- Load/Store & AGU Latency
- Cache Size