Julian Kemmerer
ECEC 621
Simulation Project
Introduction to gem5

Cache Experiment
Experiment Chosen: Keep number of processors fixed at 4, block size fixed at 64B and vary L1 cache capacity : 16KB, 32KB, 64KB

Benchmarks Used: FFT, WaterSpatial

Results:

	FFT				WaterSpatial			
L1 Cache Size (KB)	L1 Cache Avg Miss Rate	L2 Bus Traffic (Trans/Proc)	Execution Cycles	Harmonic Mean IPC	L1 Cache Avg Miss Rate	L2 Bus Traffic (Trans/Proc)	Execution Cycles	Harmonic Mean IPC
16	0.04713	63754.25	28594813	2.0314	0.00813	250241.5	225482432	2.6020
32	0.04598	61257.5	28501404	2.0406	0.00506	162422.5	223032033	2.6296
64	0.04569	60286.5	28446048	2.0465	0.00323	89644.75	220334496	2.6599

The following parameters have the specified behavior as L1 cache size is increased:

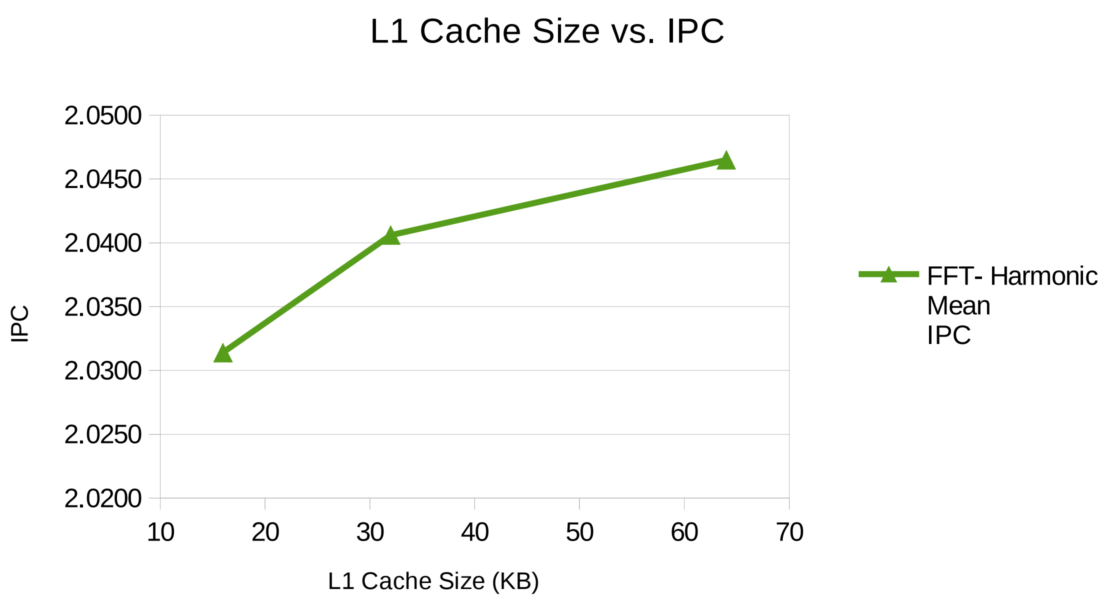
- Miss rate: decreases
- L2 bus traffic: decreases
- Execution cycles: decreases
- IPC: increases

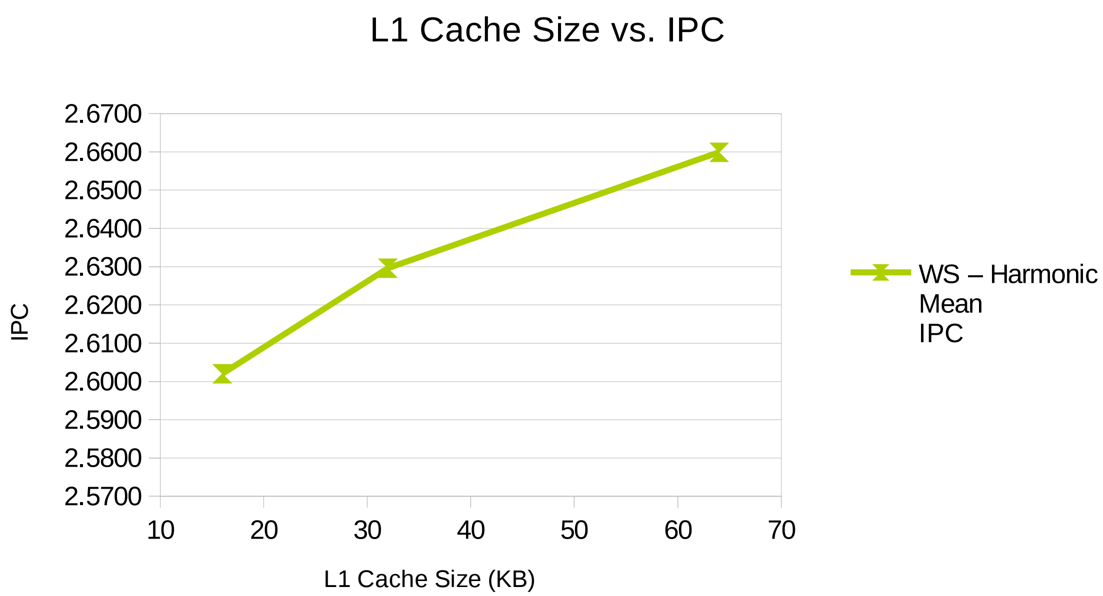
Result Explanations:

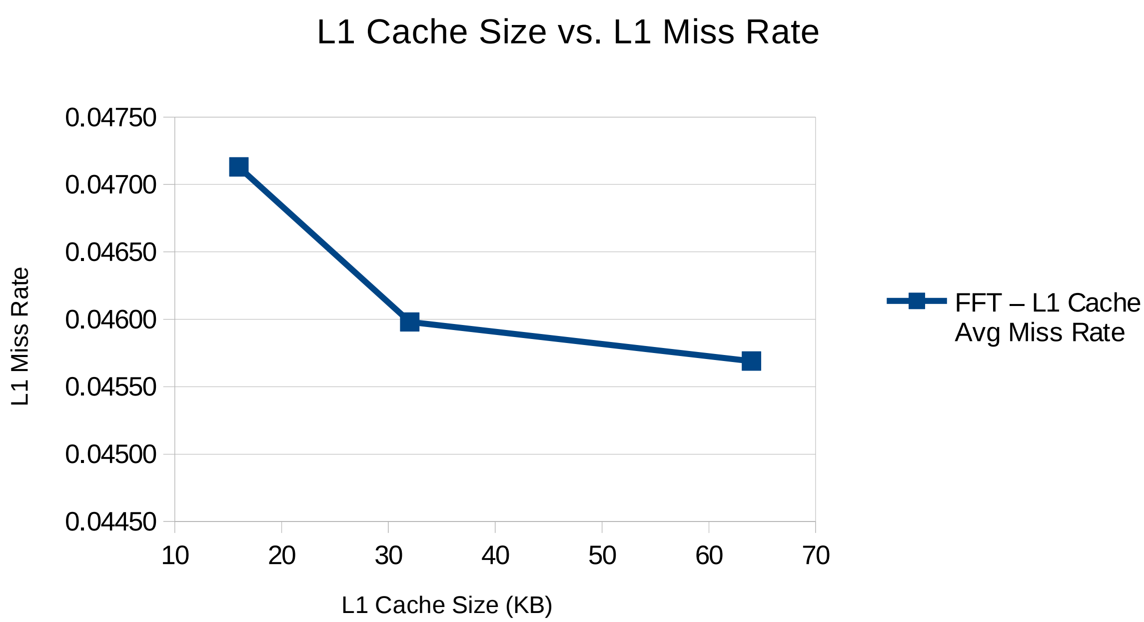
- Miss rate: decreases
    - Since more data can be stored in the L1 cache, it is more likely that the data requested will already be present in the cache → decreasing L1 miss rate.
- L2 bus traffic: decreases
    - Since more data is being found in the L1 cache, less requests to the L2 cache are made → L2 bus traffic decreases.
- Execution cycles: decreases
    - Since extra cycles do not need to be 'spent' accessing L2 or main memory (due to larger L1) the number of total execution cycles decreases.
- IPC: increases
    - When more memory operations can be completed through L1 cache, on average it takes less cycles to complete a single instruction → larger IPC.

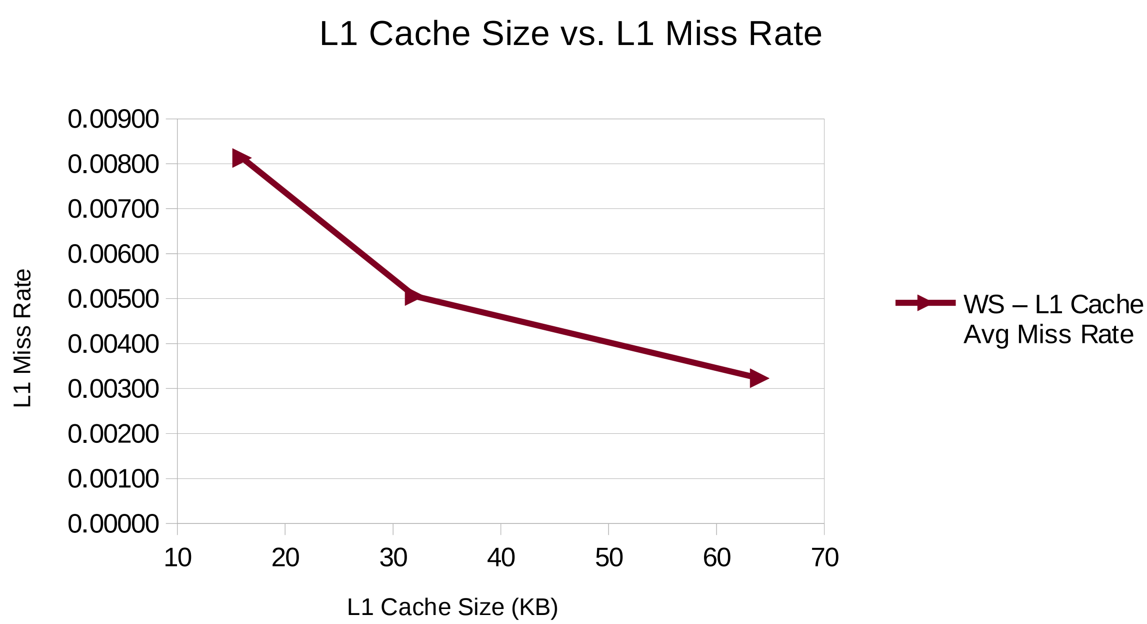
Workload Analysis:

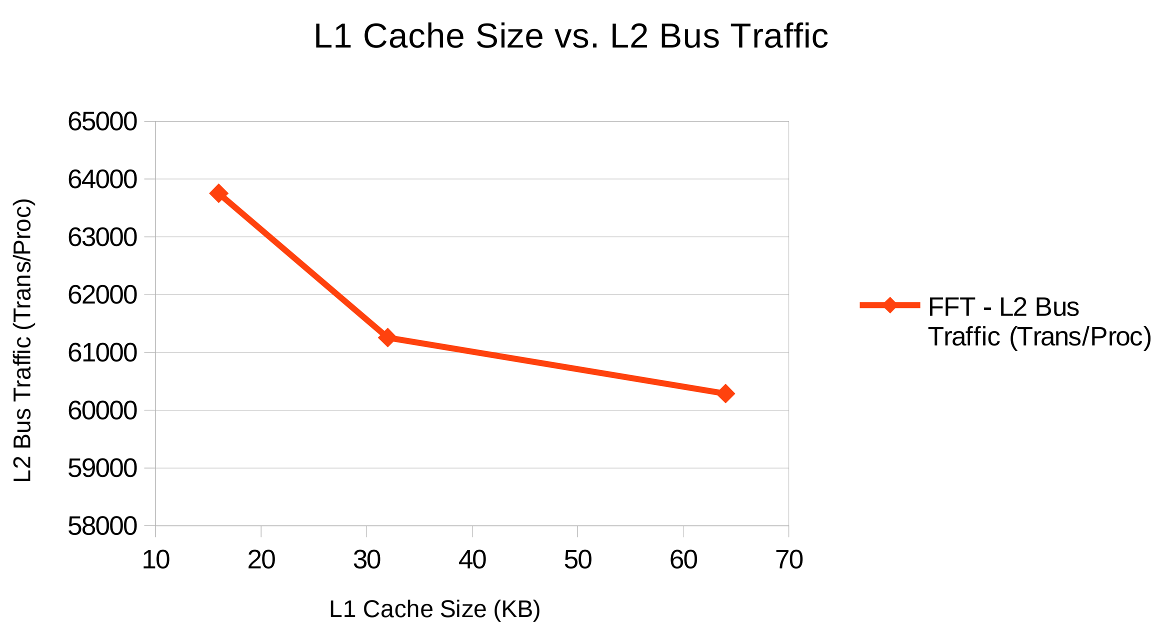
- Miss rate: decreases
  - WaterSpatial shows a sharper decrease in miss rate as the L1 becomes larger. This would indicate that the cache configuration for L1 is such that WaterSpatial benefits more than FFT (i.e. WaterSpatial likely has higher temporal and spacial data locality throughout the code)
- L2 bus traffic: decreases
  - Again, WaterSpatial shows a sharper decrease in L2 bus traffic. This can be directly correlated to the decrease in L1 misses as described above.
- Execution cycles: decreases
  - Both FFT and WaterSpatial have approximately equal decreases in execution cycles (as a percentage of the original number of cycles). This would indicate that both benchmarks have an approximately equal percentage of instructions that benefit from larger L1 cache sizes.
- IPC: increases
  - WaterSpatial shows a larger increase in IPC, this can be directly correlated to the lower miss rate in the L1 cache as described above.



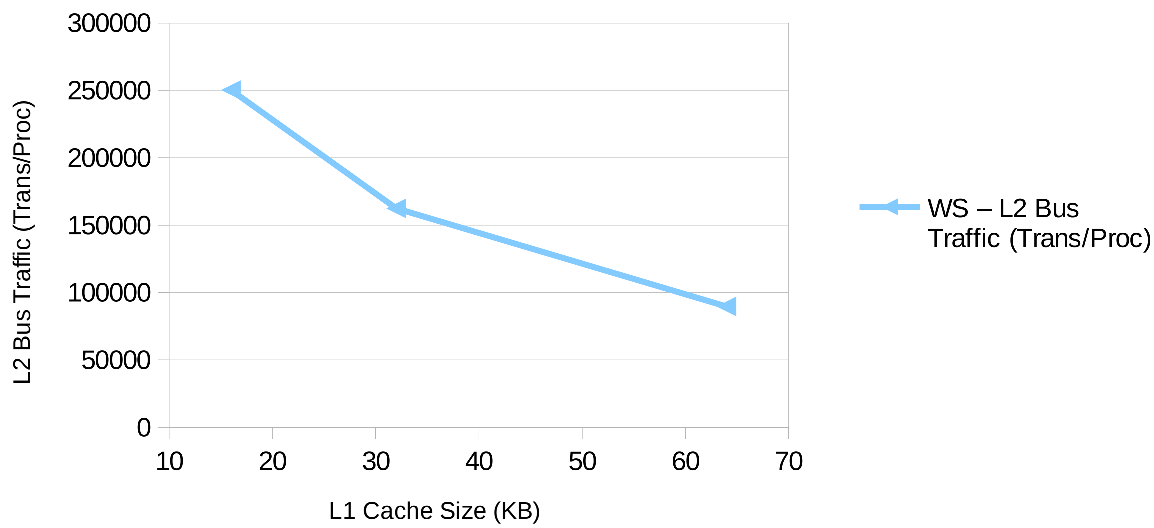

L1 Cache Size vs. IPC
2.6700
2.6600
2.6500
2.6400
2.6300
2.6200
2.6100
2.6000
2.5900
2.5800
2.5700
IPC
10
20
30
40
50
60
70
L1 Cache Size (KB)
WS – Harmonic Mean IPC


L1 Cache Size vs. L1 Miss Rate
0.04750
0.04700
0.04650
0.04600
0.04550
0.04500
0.04450
L1 Miss Rate
10
20
30
40
50
60
70
L1 Cache Size (KB)
FFT – L1 Cache Avg Miss Rate

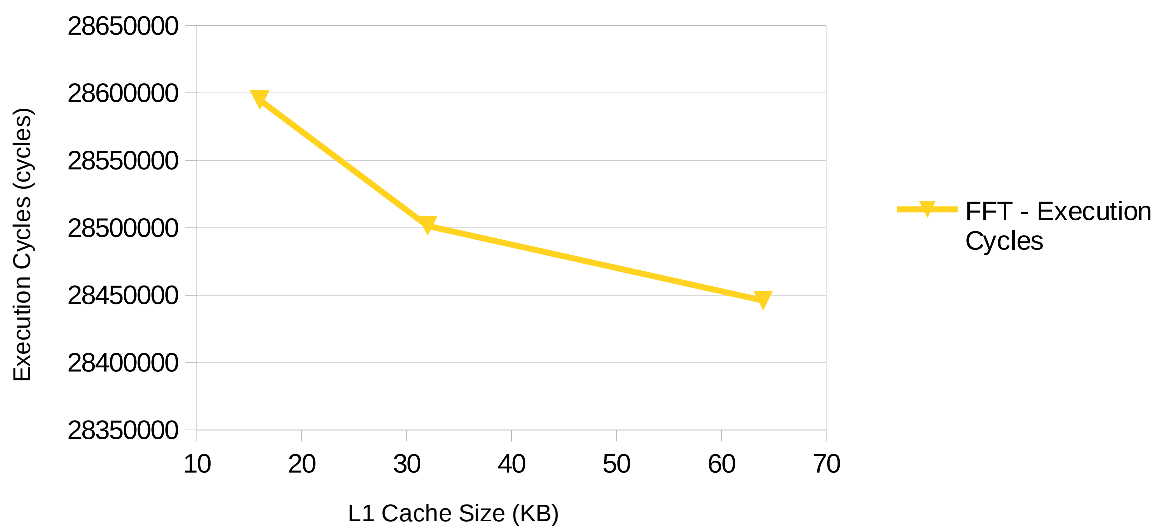

L1 Cache Size vs. L1 Miss Rate
L1 Miss Rate
0.00900
0.00800
0.00700
0.00600
0.00500
0.00400
0.00300
0.00200
0.00100
0.00000
10
20
30
40
50
60
70
L1 Cache Size (KB)
WS – L1 Cache Avg Miss Rate

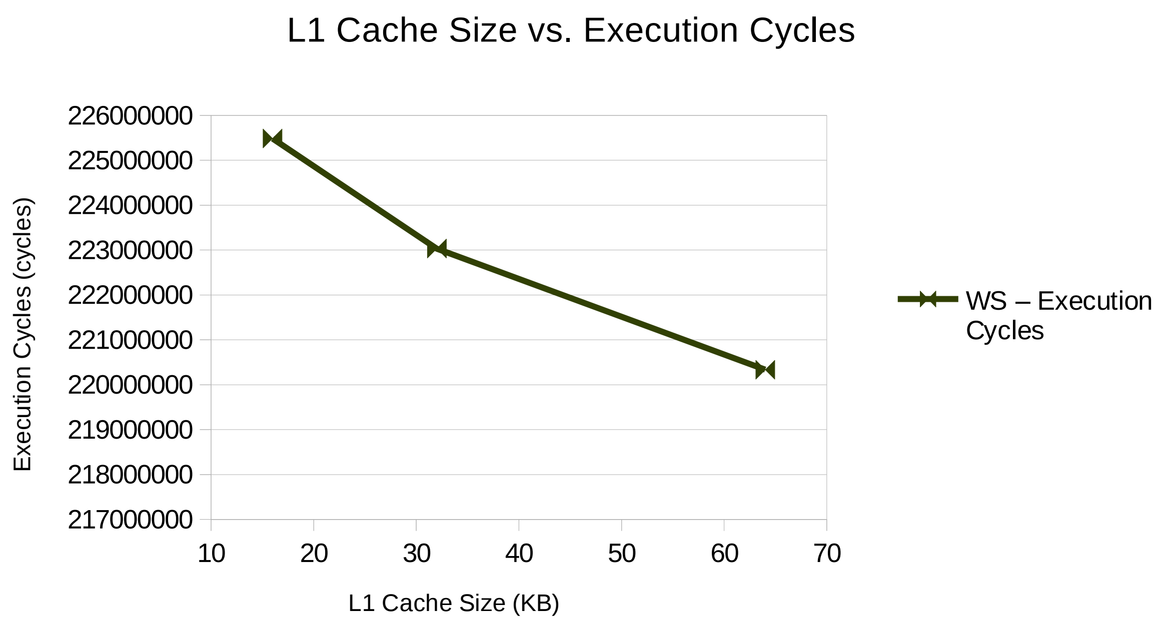

L1 Cache Size vs. L2 Bus Traffic
L2 Bus Traffic (Trans/Proc)
65000
64000
63000
62000
61000
60000
59000
58000
10
20
30
40
50
60
70
L1 Cache Size (KB)
FFT - L2 Bus Traffic (Trans/Proc)

L1 Cache Size vs. L2 Bus Traffic



L1 Cache Size vs. Execution Cycles





Architecture Experiment
Experiment Chosen: numPhysIntRegs : range [64, 128, 256, 512]

Benchmarks Used: FFT, WaterSpatial

Results:

	FFT			WaterSpatial		
Physical Integer Registers	Execution Cycles	Harmonic Mean IPC	Branch Mispredicts	Execution Cycles	Harmonic Mean IPC	Branch Mispredicts
64	31784729	1.7815	44409	227761761	2.5763	2385092
128	28937825	1.9877	52361	223062717	2.6292	2420437
256	28501404	2.0406	51228	223032033	2.6296	2417256
512	28501404	2.0406	51228	223032033	2.6296	2417256

It is interesting to note that no result parameter changed between 256/512 physical integer registers. The result was verified through examining the config.ini file for each benchmark run.

The following parameters have the specified behavior as the number of physical integer registers is increased:

- Execution cycles: decreases
- IPC: increases
- Branch Mispredicts: no correlation

Result Explanations:

- Execution cycles: decreases
  - As the number of integer registers increases, the number of writes to cache, and writes to main memory to free register space is lower → lower number of execution cycles. Keeping more data in registers decreases execution cycles.
- IPC: increases
  - As the number of integer registers increases, more data can be maintained in registers causing less 'long' writes and reads to cache and main memory. This increases the ratio of instructions to cycles (less cycles for the same instruction).
- Branch Mispredicts: no correlation
  - It is likely that the number of branch mispredicts does not depend on the number of integer registers.

Trade-offs Involved When Varying Integer Registers:

- From the collected data, it does not appear that there are trade offs that need to be made when increasing the number of integer registers.
- However, it is possible to imagine that as the number of integer registers increases, the total register memory must have a lower latency to maintain the benefits of the larger number memory.

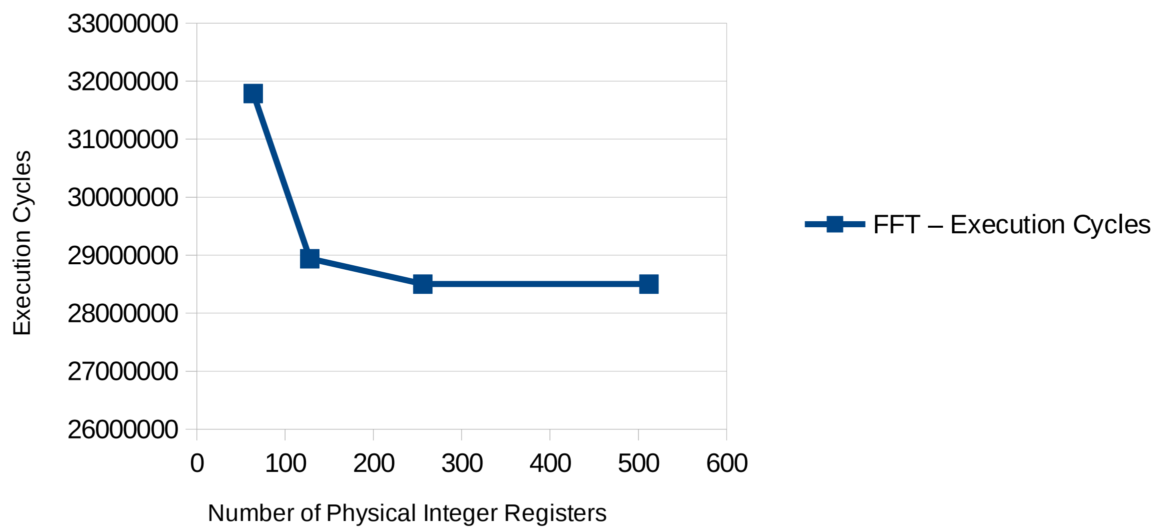
Other Required Changes to Architecture:

- There should be no other architecture changes required for using additional integer registers (other than a larger register file, obviously).

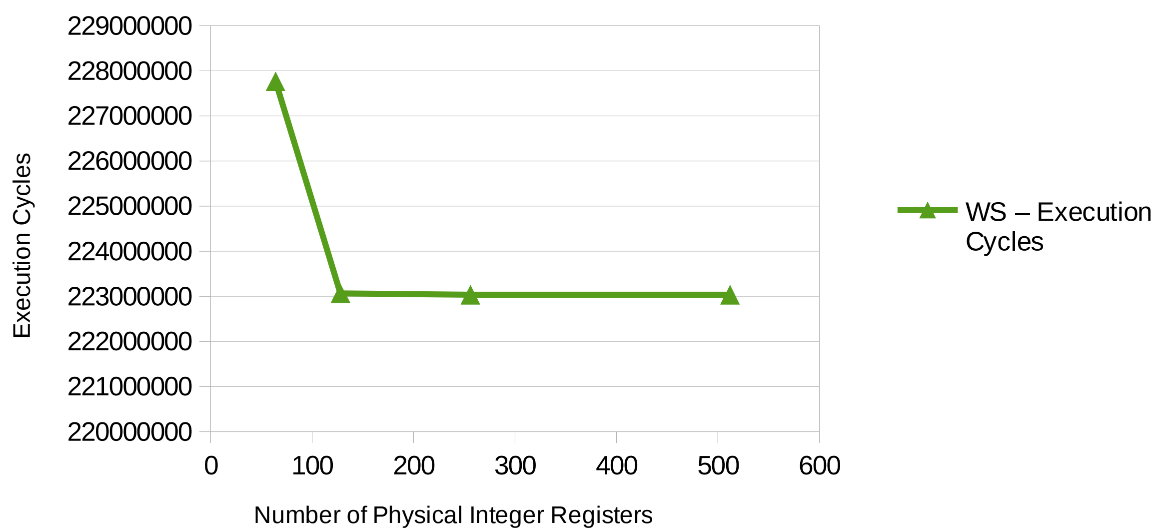
Workload Analysis:

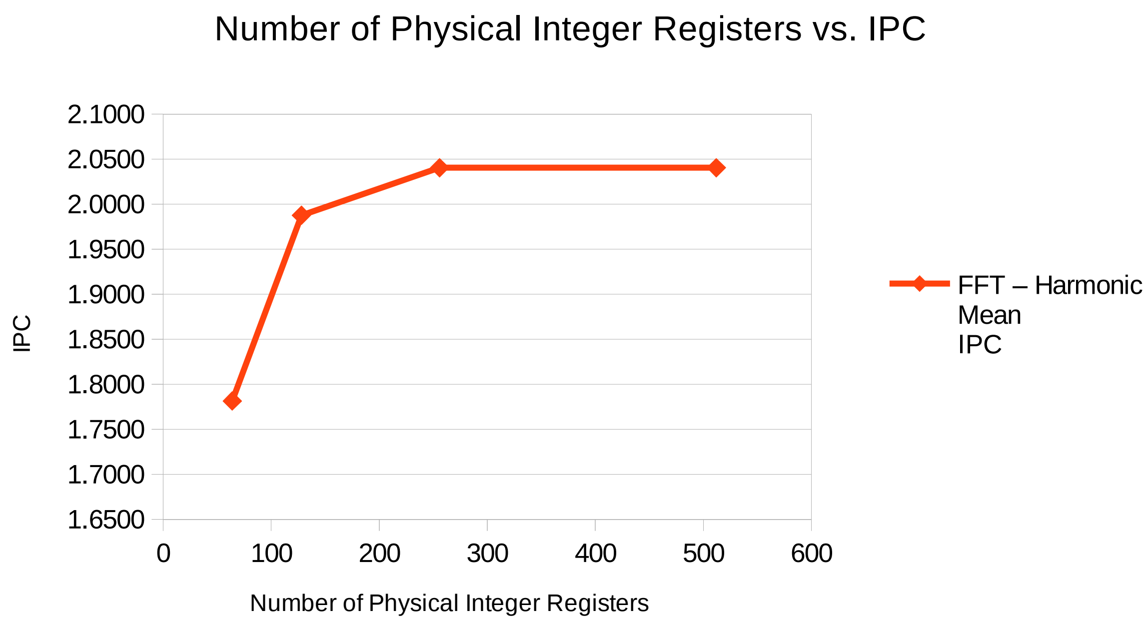
- Execution cycles: decreases
  - Both FFT and WaterSpatial show large decreases in execution cycles from 64 to 128 integer registers. Then both show a smaller decrease from 128 to 256. This would indicate that both benchmarks do not rely heavily on more that 64-128 integer registers.
- IPC: increases
  - The FFT benchmark shows a larger increase in IPC when compared to WaterSpatial. This correlates with the execution cycle decrease. This indicates that both benchmarks execute instructions that take more cycles on average when there are less integer registers to use (related to cache usage).
- Branch Mispredicts: no correlation
  - It is likely that the number of branch mispredicts does not depend on the number of integer registers.

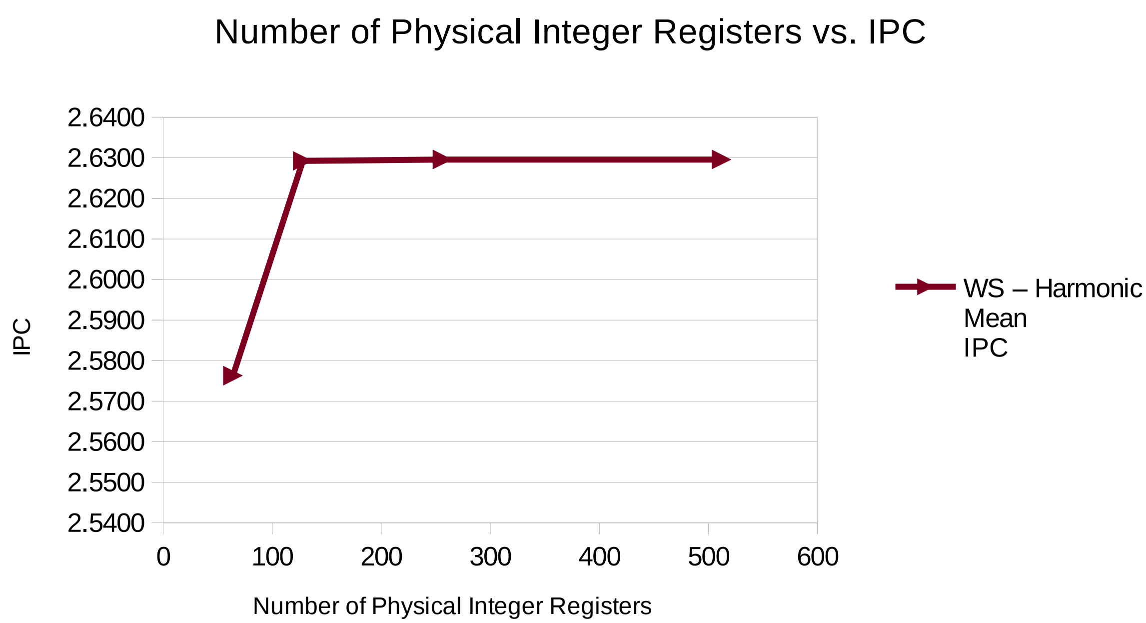
Number of Physical Integer Registers vs. Execution Cycles

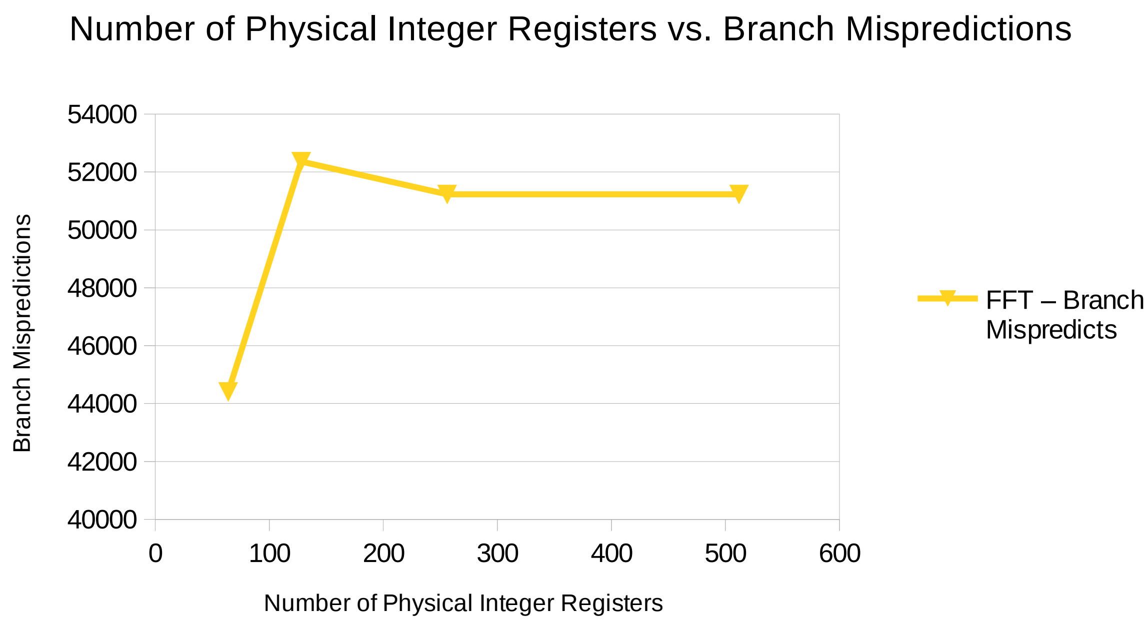


Number of Physical Integer Registers vs. Execution Cycles

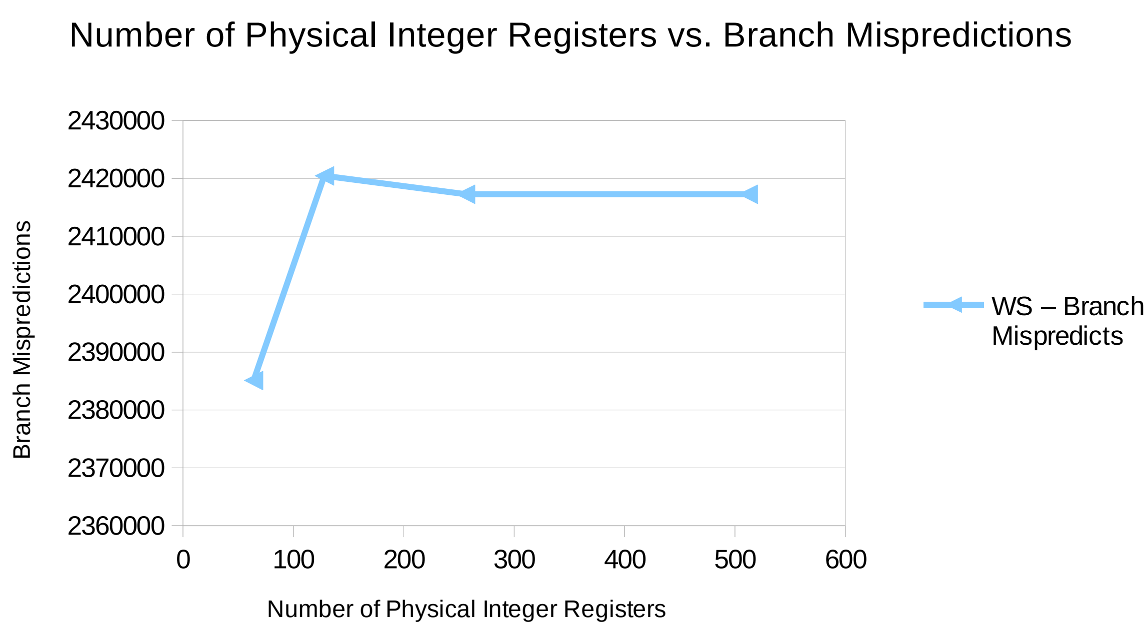



Number of Physical Integer Registers vs. IPC
2.1000
2.0500
2.0000
1.9500
1.9000
1.8500
1.8000
1.7500
1.7000
1.6500
IPC
0
100
200
300
400
500
600
Number of Physical Integer Registers
FFT – Harmonic Mean IPC


Number of Physical Integer Registers vs. IPC
2.6400
2.6300
2.6200
2.6100
2.6000
2.5900
2.5800
2.5700
2.5600
2.5500
2.5400
IPC
0
100
200
300
400
500
600
Number of Physical Integer Registers
WS – Harmonic Mean IPC



Number of Physical Integer Registers vs. Branch Mispredictions
54000
52000
50000
48000
46000
44000
42000
40000
Branch Mispredictions
0
100
200
300
400
500
600
Number of Physical Integer Registers
FFT – Branch Mispredicts



Number of Physical Integer Registers vs. Branch Mispredictions
2430000
2420000
2410000
2400000
2390000
2380000
2370000
2360000
Branch Mispredictions
0
100
200
300
400
500
600
Number of Physical Integer Registers
WS – Branch Mispredicts