Project Report

Benjamin Medoff, Shuai Wang

1. Describe in 100 words or less how the provided framework and its component enable a design space exploration.

Design space exploration is the process of testing different design space nodes to determine which ones, together, form the optimal design. The provided framework succeeds in doing this by using our code to generate unique design point configurations, then it runs simulations on multiple configurations where only one design point is changed. From these it decides which design point is best. This is repeated until all 18 design points are completed and the best configuration is found.

2. List the design point chosen by your DSE

Our program found that for EDP, “0 0 2 2 0 5 0 1 3 1 0 0 4 3 0 1 4 3” was the best configuration and for performance “0 0 2 2 0 6 0 2 3 1 0 0 4 3 3 1 5 4” was the best configuration. For the corresponding values, we have listed them in the table below.

3. Fill out the following table as detailed below.

Parameter	Performance	EDP
Pipeline Width	Value = 1 A greater width creates more complexity which means more preprocessing time, and the workload might be highly sequential	Value = 1 More hardware requires more energy to run and almost certainly would not run as efficiently as the main pipeline.
Scheduling	Value = In Order Because this is only run with pipeline width 1 OO does not give us better parallelism and just adds more overhead	Value = In Order Because this is only run with pipeline width 1 OO does not give us better parallelism and just adds more overhead
L1 block size	Value = 32 bytes The larger the block size, the less likely a cache miss because of spatial locality, but too large and L1 access time becomes too	Value = 32 bytes The larger the block size, the more energy we need to handle every cache entry. But with too small block size, we need to

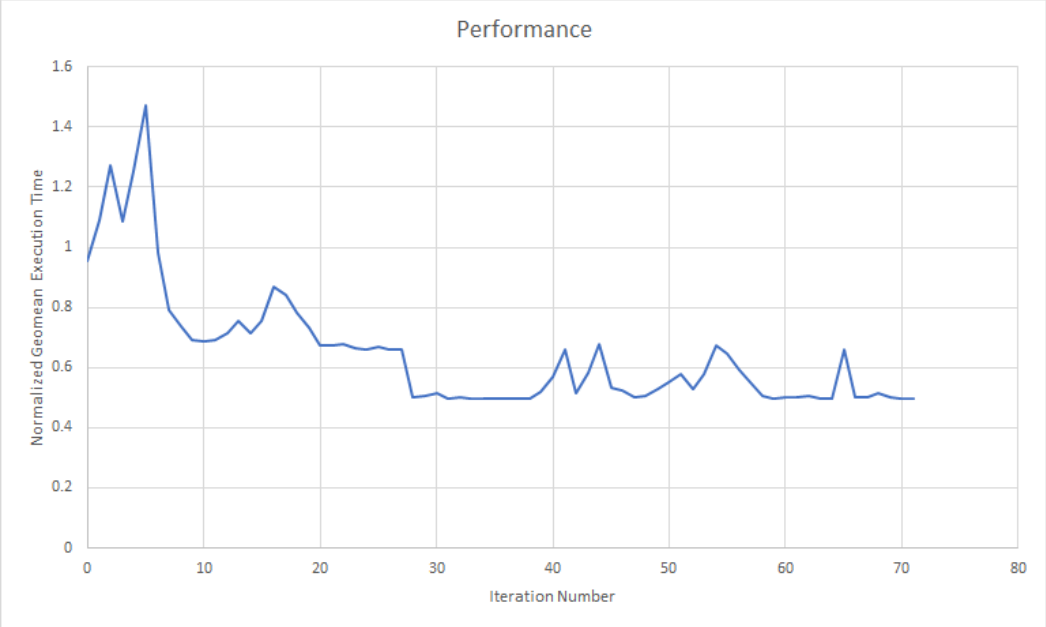
	large	access more blocks than before when one piece of data or instruction, which will cost more energy
L1 data sets	Value = 128 The larger the data sets, means the longer time to traverse the whole cache, so it will take longer time to locate a specific entry in the cache	Value = 128 The larger the data sets, means the longer time to traverse the whole cache, so it will take longer time to locate a specific entry in the cache, that is, more energy cost.
L1 data set associativity	Value = 1 For a given cache size, more set associativity means less entries in the main memory can be cached	Value = 1 For a given cache size, more set associativity means less entries in the main memory can be cached, so we need to access to the main memory more times, that is, more energy cost
L1 instruction sets	Value = 2048 The larger the instruction sets, means the longer time to traverse the whole cache. And too small number of instruction sets means high cache miss rate	Value = 1024 The larger the instruction sets, means the longer time to traverse the whole cache, so the energy cost to traverse the whole cache will increase. And too small number of instruction sets means high cache miss rate, so we need to access to the main memory more times, that is, higher energy cost
L1 instruction set associativity	Value = 1 More set associativity means we can cache less number of entries in L1 instruction cache, which means less hit rate.	Value = 1 More set associativity means we can cache less number of entries in L1 instruction cache, which means we need to access to the main memory more times, that is, more energy

		cost.
L2 sets	Value = 1024 The larger the sets, means the longer time to traverse the whole cache. And too small number of sets means high cache miss rate	Value = 512 The larger the sets, means the longer time to traverse the whole cache, so the energy cost to traverse the whole cache will increase. And too small number of sets means high cache miss rate, so we need to access main memory more times, that is, higher energy cost
L2 block size	Value = 128 bytes Access time is not as important in L2 so a large block size decreases cache miss rate	Value = 128 bytes Access time is not as important in L2 so a large block size decreases cache miss rate. With higher cache hit rate, we can access memory less times, which contributes to lower energy cost.
L2 set associativity	Value = 2 Increasing set associativity decreases cache miss rate, but too much can hurt access time.	Value = 2 Increasing set associativity decreases the number of entries can be cached, so we need to access to main memory more times, which contributes to higher energy cost.
Cache/TLB replacement policy	Value = LRU LRU is more expensive than other replacement policies but is substantially better at decreasing cache miss rates, which contributes to higher performance	Value = LRU LRU is more expensive than other replacement policies but is substantially better at decreasing cache miss rates. With lower cache miss rate, we do not need to access the main memory every time, which contributes to less energy cost
FPU width	Value = 1	Value = 1

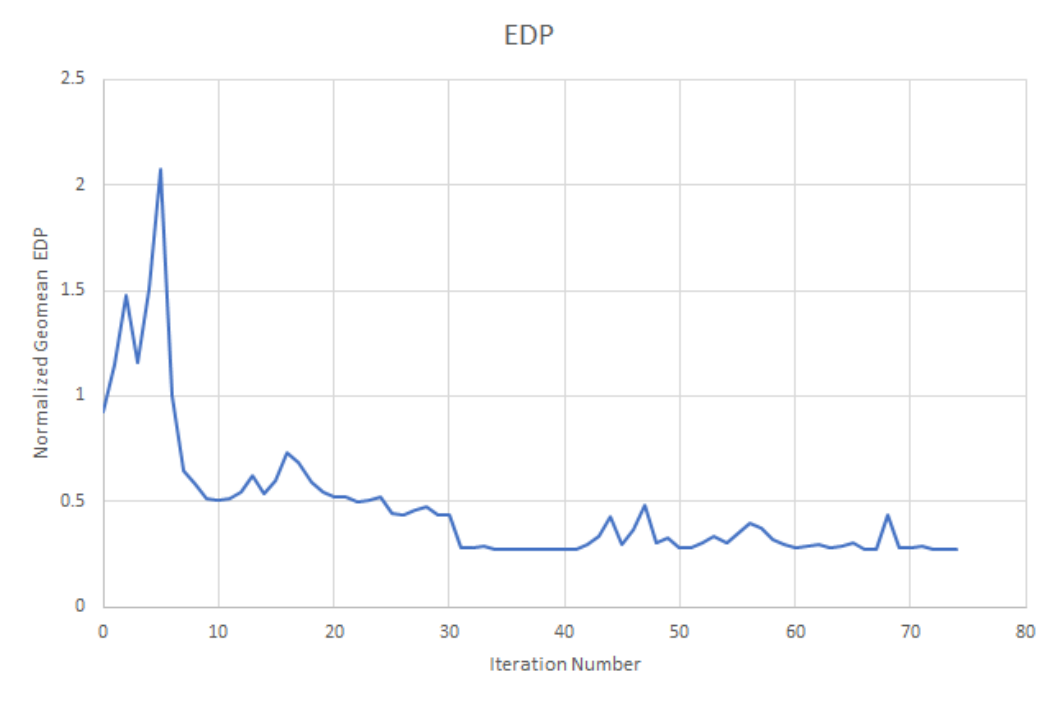
	Without CPU parallelism we can't benefit too much from FPU width.	Without CPU parallelism we can't benefit too much from FPU width.
Branch predictor	Value = 2 level w/ n = 4 2 level predictors are better at recognizing patterns other than TTTTT or NNNNN higher n ads more past info. More number of accurate prediction means higher performance.	Value = 2 level w/ n= 4 2 level predictors are better at recognizing patterns other than TTTTT or NNNNN higher n ads more past info. More number of accurate prediction means less energy cost
Return Address Stack size	Value = 8 A deeper RAS will more consistently smooth out jump instructions	Value = 8 A deeper RAS will more consistently smooth out jump instructions
Branch Target Buffer set associativity	Value = 1024 sets assoc 2 Large number of sets and small associativity means large number of high BTB hit rate, which contributes to high performance	Value = 128 sets assoc 16 Small number of sets and large associativity means we can find the target entry in the BTB easier, which contributes to low energy cost

4. Plots as detailed below

A:

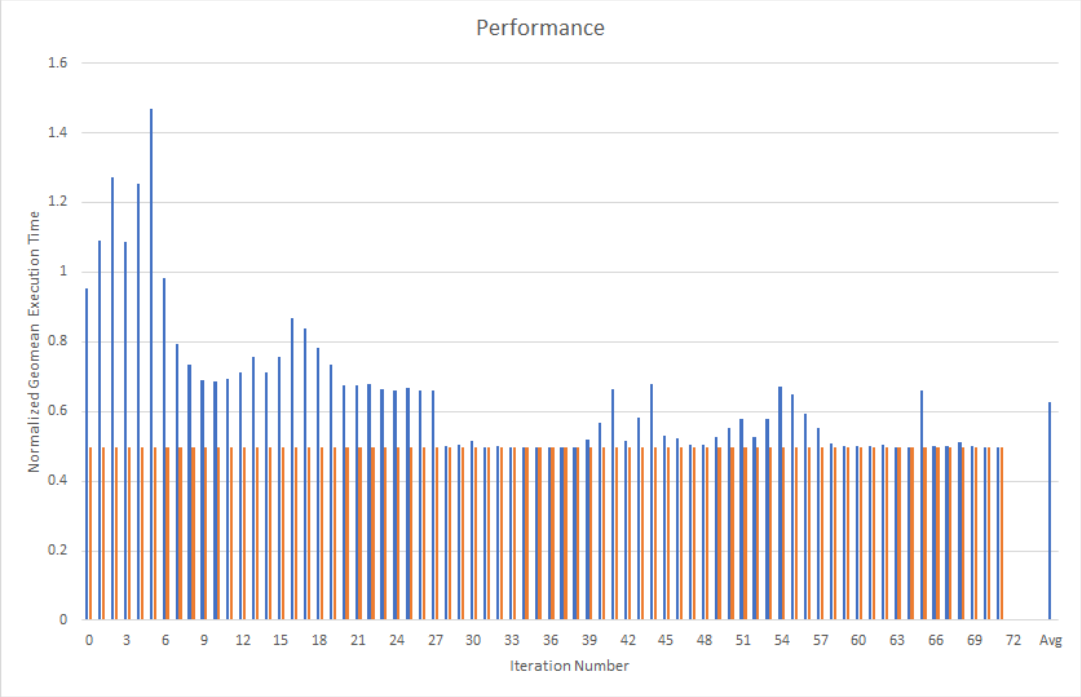


B:



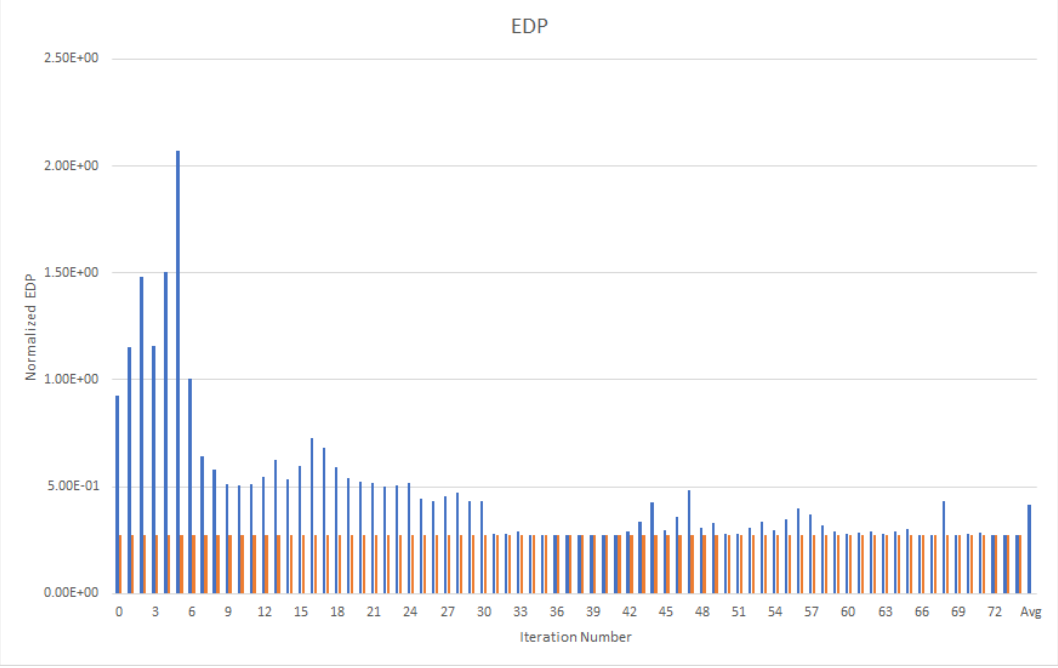
C:

The orange bar is for the best normalized performance, and blue bars are for the normalized performance of every benchmark.



D:

The orange bar is for the best normalized EDP, and blue bars are for the normalized EDP of every benchmark.



5. Describe a more sophisticated heuristic which you expect will perform design space exploration (limited by 1000 design points)

more effectively to find a better performing design (with respect to execution time)

The heuristic that we used in this assignment is simple and relatively effective but does not optimize the process to more effectively find the best configuration. A better heuristic would take into account what design points are most related and make configuration guesses based on past runs. More specifically a better heuristic might look like this: First, start with a base configuration like we do in this project then keep one design point constant and change the other design points related to it (e.g. only the branch design points) to see how they affect the EDP/execution time. Repeat this for each dimension class (Cache, FPU, Branch Prediction, Core). Using this information we craft a couple optimal combinations for each class (for example we may have noticed that a direct mapped L1 cache works better with set associated L2 cache). Then we test different test configurations using these premade class combinations to find the best. Finally we can go through and change each dimension individually to see if we can get any final small optimizations.

6. Elaborate on any 2 new insights you gained while working on this project.

By coding:

i. Both L1 cache and L2 cache are in the CPU. But L1 cache size is less than L2 cache, while the latency of L1 cache is less than the latency of L2 cache

L1 cache size is equal to the multiply of L1 block size, L1 data set and L1data associativity. And similar as L2 cache size. We can found that L1 cache size is less than L2 cache size, but for the latency, L1 cache is less than L2 cache.

This is because L1 is closer to CPU than L2 cache. Basically, the system will access L2 cache one cycle later than L1 cache. L1 cache size is smaller, which makes it easier to find something in a smaller space.

ii. L1 data cache block size is same as L1 instruction cache block size. But they are separate and can be access at the same time.

L1 cache consists of two parts, L1 data cache and L1 instruction cache. They can be access at the same time, which reduces the caching conflict caused by multi-threads or multi-cores.

By validating the results:

i. Tradeoff is everywhere

When we analyze and compare the performance or energy of a specific system using different configurations, there are many tradeoffs among these dimensions. For example, intuitively, we want a L1 cache with larger block size, because the larger block size we have, the less cache miss we will get. But the larger L1 block size means the longer L1 access time. So we should always consider these kinds of tradeoffs when we evaluate the results.

ii. Some choices are interesting.
- Pipeline width and scheduling: The scheduling policy is dependent on the pipeline width we choose. Since we choose pipeline width as 1, in-order scheduling will be a better choice.
- Cache/TLB replacement policy: LRU is a better choice comparing to FIFO and random, although we need to choose the page we want to evict according to their timestamp.
- Brach predictor: 2-bit branch predictor works better 1-bit branch predictor.

7. List of additional resources used(optional)

- CPU cache (Wikipedia)
  URL: https://en.wikipedia.org/wiki/CPU_cache
- Design space exploration (Wikipedia)
  URL: https://en.wikipedia.org/wiki/Design_space_exploration
- Cache Basics (Northeastern University Course Material)
  URL: https://course.ccs.neu.edu/com3200/parent/NOTES/cache-basics.html

8. Additional information or comments(optional)

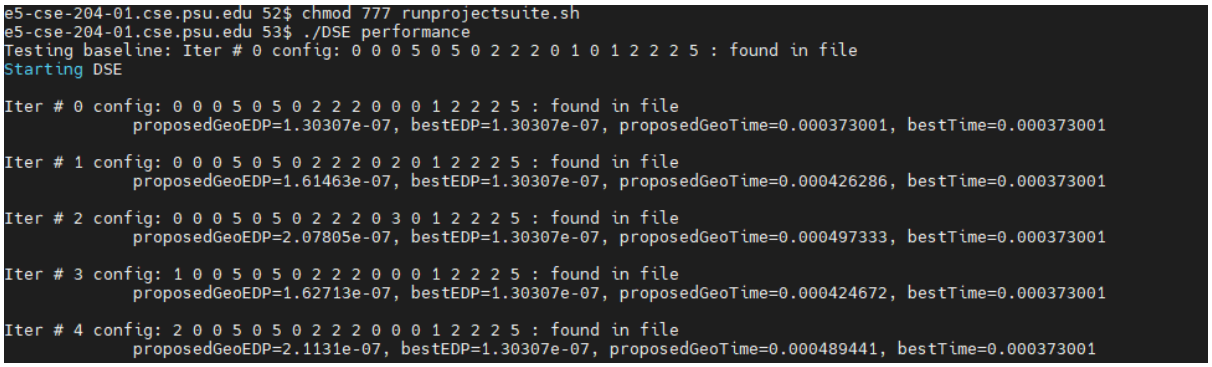
- GitHub URL
  We have uploaded all our source code to the GitHub, the URL is:
  https://github.com/5404BenM/Medoff_Wang_431Proj
- Steps for running the program
  If we want to run the program, first thing we should do is:

```
chmod 777 runprojectsuite.sh
```

Then when we run DSE for performance option:

```
./DSE performance
```

We can get Iter # 0 ~ Iter # 68. Parts of screenshots are shown as follow:



```
Iter # 65 config: 0 0 2 2 0 6 0 2 3 1 0 0 0 3 2 1 5 4 : running simulation
            proposedGeoEDP=7.02447e-08, bestEDP=4.08993e-08, proposedGeoTime=0.000272481, bestTime=0.000194049

Iter # 66 config: 0 0 2 2 0 6 0 2 3 1 0 0 1 3 2 1 5 4 : running simulation
            proposedGeoEDP=4.93154e-08, bestEDP=4.08993e-08, proposedGeoTime=0.000219805, bestTime=0.000194049

Iter # 67 config: 0 0 2 2 0 6 0 2 3 1 0 0 2 3 2 1 5 4 : running simulation
            proposedGeoEDP=4.96874e-08, bestEDP=4.08993e-08, proposedGeoTime=0.000220839, bestTime=0.000194049

Iter # 68 config: 0 0 2 2 0 6 0 2 3 1 0 0 3 3 2 1 5 4 : running simulation
            proposedGeoEDP=5.14918e-08, bestEDP=4.08993e-08, proposedGeoTime=0.000225704, bestTime=0.000194049

returned the same configuration EDP 0 0 2 2 0 6 0 2 3 1 0 0 4 3 2 1 5 4
Time 0 0 2 2 0 6 0 2 3 1 0 0 4 3 2 1 5 4
FINISH
```

When we run DSE for energy option

```
./DSE energy
```

We can get Iter # 0 ~ Iter # 74. Parts of screenshots are shown as follow:

```
e5-cse-204-01.cse.psu.edu 54$ ./DSE energy
Testing baseline: Iter # 0 config: 0 0 0 5 0 5 0 2 2 2 0 1 0 1 2 2 2 5 : found in file
Starting DSE

Iter # 0 config: 0 0 0 5 0 5 0 2 2 2 0 0 0 1 2 2 2 5 : found in file
            proposedGeoEDP=1.30307e-07, bestEDP=1.30307e-07, proposedGeoTime=0.000373001, bestTime=0.000373001

Iter # 1 config: 0 0 0 5 0 5 0 2 2 2 0 2 0 1 2 2 2 5 : found in file
            proposedGeoEDP=1.61463e-07, bestEDP=1.30307e-07, proposedGeoTime=0.000426286, bestTime=0.000373001

Iter # 2 config: 0 0 0 5 0 5 0 2 2 2 0 3 0 1 2 2 2 5 : found in file
            proposedGeoEDP=2.07805e-07, bestEDP=1.30307e-07, proposedGeoTime=0.000497333, bestTime=0.000373001

Iter # 3 config: 1 0 0 5 0 5 0 2 2 2 0 0 0 1 2 2 2 5 : found in file
            proposedGeoEDP=1.62713e-07, bestEDP=1.30307e-07, proposedGeoTime=0.000424672, bestTime=0.000373001
```

```
Iter # 72 config: 0 0 2 2 0 5 0 1 3 1 0 0 4 0 0 1 4 3 : running simulation
            proposedGeoEDP=4.63557e-08, bestEDP=3.84009e-08, proposedGeoTime=0.000222911, bestTime=0.000197122

Iter # 73 config: 0 0 2 2 0 5 0 1 3 1 0 0 4 1 0 1 4 3 : running simulation
            proposedGeoEDP=4.61555e-08, bestEDP=3.84009e-08, proposedGeoTime=0.000222345, bestTime=0.000197122

Iter # 74 config: 0 0 2 2 0 5 0 1 3 1 0 0 4 2 0 1 4 3 : running simulation
            proposedGeoEDP=4.6012e-08, bestEDP=3.84009e-08, proposedGeoTime=0.000221939, bestTime=0.000197122

returned the same configuration EDP 0 0 2 2 0 5 0 1 3 1 0 0 4 3 0 1 4 3
Time 0 0 2 2 0 5 0 1 3 1 0 0 4 3 0 1 4 3
FINISH
```