Project 1 CMPEN 431 REPORT

By Samarth Tehri and Aly Ghallab

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

   The provided framework enabled design space exploration by allowing us to test and modify different combinations of processor parameters to achieve the best configuration with regards to performance and EDP. It allowed us to run simulations with different configurations to find the best one for each test. We used a heuristic (#2) (as mentioned in the project 1 instructions document) to determine the transversal order of dimensions to find the best configuration under 1000 iterations.

2) List the design point chosen by your DSE:

   Performance Design Point Values: (1, inorder, 64, 1024, 1, 1024, 1, 512, 128, 4, “f”, 1 "-bpred comb -bpred:comb 1024”, 8, “512 4” , 1, 3, 9)

   EDP Design Point Values: (1, inorder, 32, 1024,1,1024,1,512,128,2 “f”, 1, "-bpred comb -bpred:comb 1024”, 8, “256 8”, 3, 3, 9)

   Performance Design Point Indices: 0 0 3 5 0 5 0 1 3 2 1 0 4 3 2 0 2 4

   EDP Design Point Indices: 0 0 2 5 0 5 0 1 3 1 1 0 4 3 1 2 2 4

3) Fill out the following table as detailed below

Parameter	Performance	EDP
Width	Index = 0 Value = 1	Index = 0 Value = 1
	Why = less instruction pipelines meant less task/context switching,saving time	Why = Increasing instruction pipelines increases energy consumption
Scheduling	Index = 0 Value = inorder	Index = 0 Value = inorder
	Why = inorder is faster due to slight edge in multitasking and speec	Why = Scheduling miscalculations and mishaps increase run time and therefore energy consumption
l1block	Index = 3 Value = 64	Index = 2 Value = 32
	Why = more cache allows for faster	Why = Cache is power hungry,

	lookup times, since many values are stored in the cache, therefore does not miss as frequently.	therefore lower cache means less power allocated (to unused values), hence longer time
dl1sets	Index = 5 Value = 1024	Index = 5 Value = 1024
	Why = Faster lookup time due to sets	Why = Less energy consumed due to faster lookup
dl1assoc	Index = 0 Value = 1	Index = 0 Value = 1
	Why = having 1 way associativity speeds up the process by not thread/multi scheduling, and does not increase extra clock cycle	Why = 1 way associativity reduces the risk of scheduling errors hence power consumption
il1sets	Index = 5 Value = 1024	Index = 5 Value = 1024
	Why = Faster lookup time due to sets	Why = Less energy consumed due to faster lookup
il1assoc	Index = 0 Value = 1	Index = 0 Value = 1
	Why = having 1 way associativity speeds up the process by not thread/multi scheduling, and does not increase extra clock cycle	Why = 1 way associativity reduces the risk of scheduling errors hence power consumption
ul2sets	Index = 1 Value = 512	Index = 1 Value = 512
	Why = Having a ul2 cache decreases chance of accessing memory on a cache miss hence greater performance	Why = Accessing Memory requires more power as opposed to having a ul2 hit, however a larger ul2 means more power consumed.
ul2block	Index = 3 Value = 128	Index = 3 Value = 128
	Why = more cache allows for faster lookup times, since many values are stored in the cache, therefore does not miss as frequently.	Why = Cache is power hungry, therefore lower cache means less power allocated (to unused values), hence longer time
ul2assoc	Index = 2 Value = 4	Index = 1 Value = 2
	Why = having 4 way associativity speeds up the process by not thread/multi scheduling, and does not increase extra clock cycle	Why = 2 way associativity reduces the risk of scheduling errors hence power consumption, increasing associativity increases power consumption

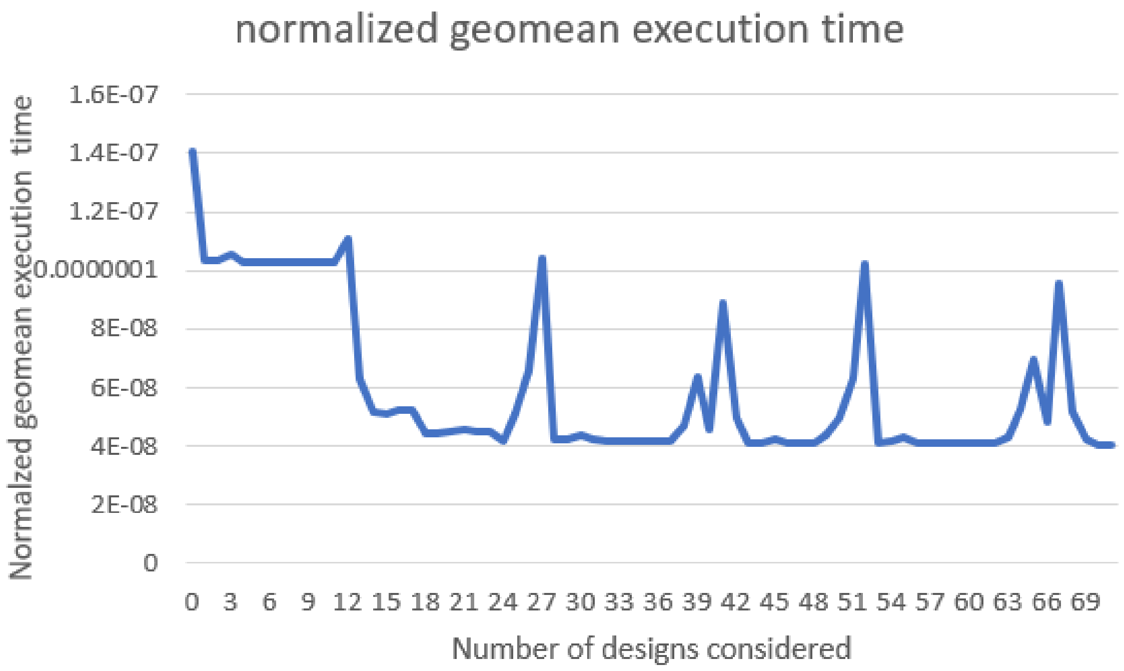
replacepolicy	Index = 1 Value = “f”	Index = 1 Value = “f”
	Why = first in first out increases performance as most recently used cache stays, while least recent ones are evicted	Why = intelligently evicting data while allowing recent caches to stay reduces the eviction energy used, therefore more energy efficient.
fpwidth	Index = 0 Value = 1	Index = 0 Value = 1
	Why = Having a higher fpwidth increases latency, and the simulator likely didn't do floating point operations extensively	Why = having a higher fpwidth increases latency, and energy consumption, hence the minimum width the better.

branchsettings	Index = 4
	Value = "-bpred comb -bpred:comb 1024”
		Index = 4
		Value = "-bpred comb -bpred:comb 1024”

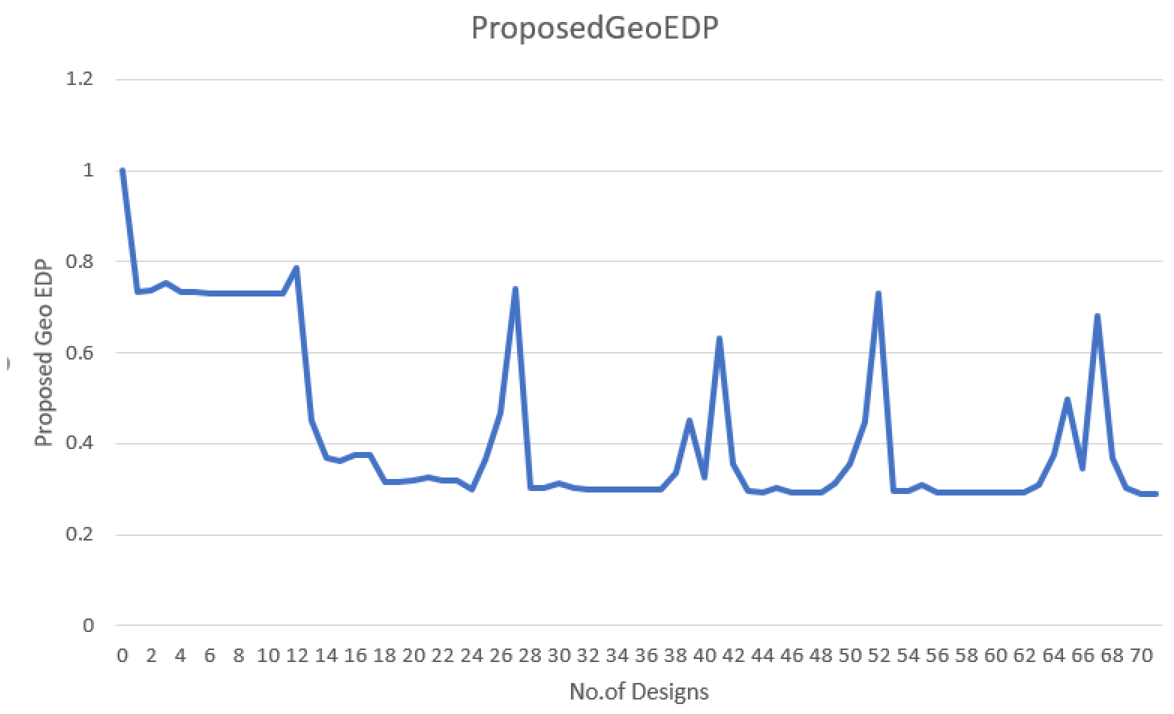
	Why = Using a combination branch predictor gives better predictions allowing for greater performance.	Why = When a branch predictor is more accurate, less mispredictions occur which minimize energy wasted
ras	Index = 3 Value = 8	Index = 3 Value = 8
	Why = RAS store return addresses for function calls, which prevents stalling and unintended jumps making the program more efficient	Why = having an RAS of the appropriate size decreases stalling minimizing energy wasted.
btb	Index = 2 Value = “512 4”	Index = 1 Value = “ 256 8”
	Why = Having a larger btb is better for performance depending on the number of branches.	Why = A smaller btb reduces power consumption per clock cycle but sacrifices performance.
dl1lat	Index = 0 Value = 1	Index = 2 Value = 3
	Why = Cache latency Value determined by other values in our heuristic via formula.	Why = Cache latency Value determined by other values in our heuristic via formula.
il1lat	Index = 2 Value = 3	Index = 2 Value = 3
	Why = Cache latency Value determined by other values in our heuristic via formula.	Why = Cache latency Value determined by other values in our heuristic via formula.
ul2lat	Index = 4 Value = 9	Index = 4 Value = 9

	Why = Cache latency Value determined by other values in our heuristic via formula.	Why = Cache latency Value determined by other values in our heuristic via formula.

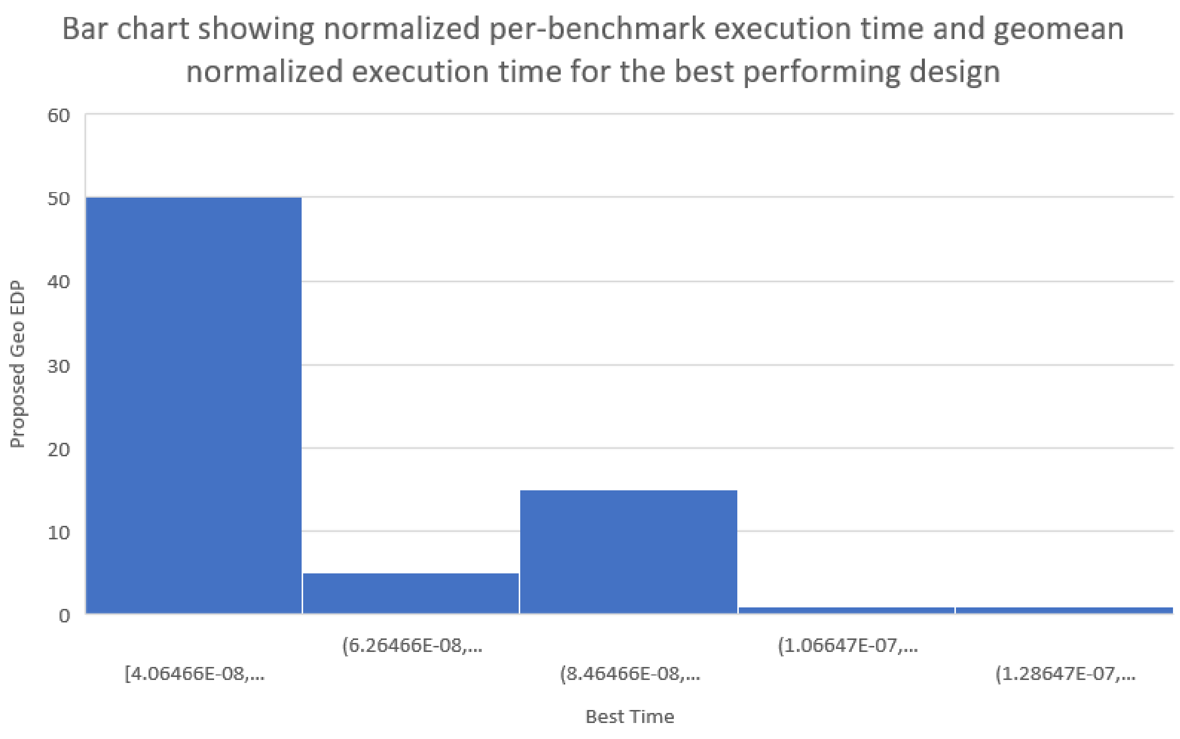
4) Plots as detailed below
   A) Line plot of normalized geomean execution time (yaxis) for each considered point vs number of designed considered (xaxis)



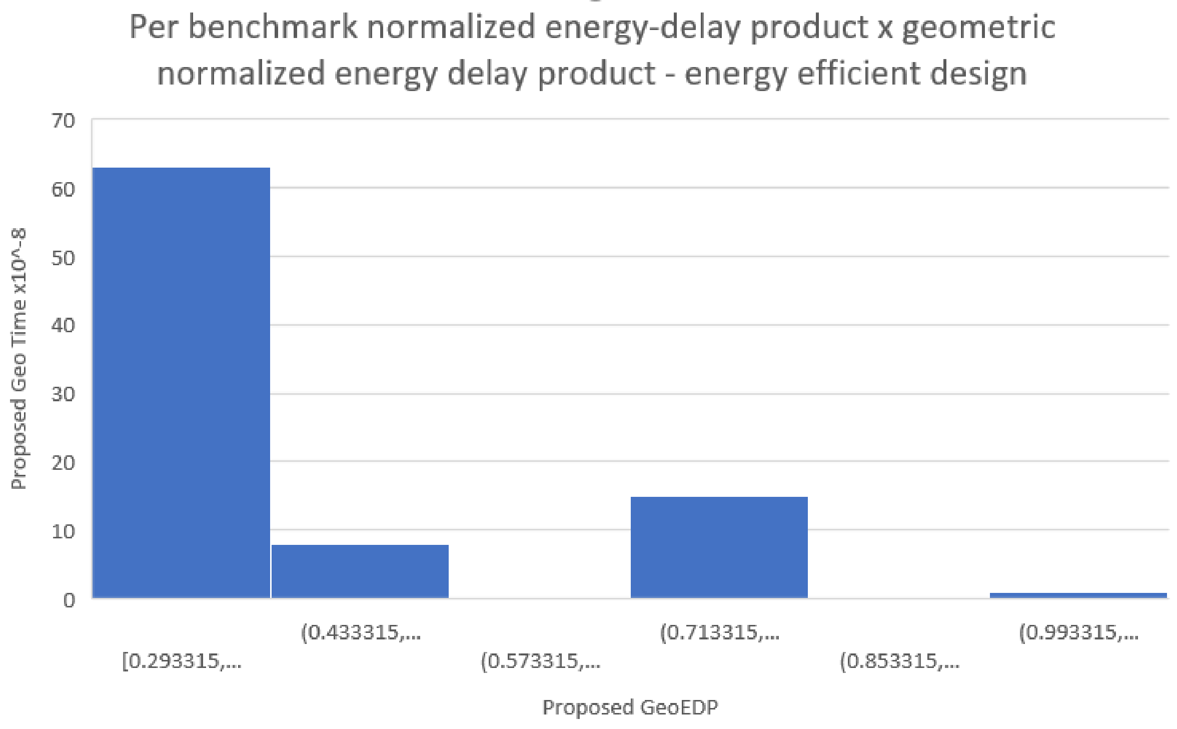
B) Line plot of normalized geomean of energy-delay product (yaxis) vs number of designs considered



C) Bar chart showing normalized per-benchmark execution time and geomean normalized execution time for the best performing design



D) Bar chart showing per-benchmark normalized energy-delay product and geomean normalized energy delay product for the most energy-efficient design found



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)

Our heuristic takes around less than 100 design points according to the code in the Simplescalar framework. With 1000 points we can develop a more complex heuristic. An issue with how our heuristic is set up is that it optimizes each dimension in a linear fashion hence it does not take into account that choosing a less than optimal value for one dimension may lead to a much better configuration overall. A solution is to have a heuristic that traverses in different orders with multiple passes over each traversal order to ensure the heuristic covers multiple designs and is then able to decide which configuration is optimal. It could also be improved by evaluating multiple dimensions at once; however, depending on the dimension cardinality the number of dimensions must be minimal if we were to stay within 1000 design points.

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

We learned how different design parameters affect a processor's performance (measured via execution time) and a processor's energy consumption. We also learned how we can use a simulator such as simplescalar to assist in design space exploration using a heuristic and not relying on impractical exhaustion methods yet achieving accurate results for our designs.

7) List of additional resources used (optional)
   Canvas -> cmpen 431 files
   Microsoft Excel for the Plots
8) Additional information or comments (optional)
   It is a very tough project and takes a lot of time to code. And debugging along with testing for all cases took so much time. Also this project was during exam week, so other exams collided with projects which made project more tough