CSE 5350
Computer Architecture II
Fall 2023, © DL, UTA, 2023

Programming Assignment
CAT2: Computer Architecture 2 "VM"
Due: On BlackBoard
Part 1 Due Nov 10, 23:00
Part 2 Due Nov 17, 23:00
Part 3 Due Nov 28, 23:00
Part 4 Due Dec 1, 23:00

Description: The CAT2 "Virtual Machine" (VM) implements some of the functionality needed to run some CAT2 "benchmark" programs. The skeleton code for CAT2 is provided in Python, you may wish to use a different version of Python, and feel free to re-write the functionality in any programming language you wish (please discuss with the GTA and instructor, first.) Please use the provided skeleton code concepts: a 5 cycle micro-operation with the included instruction set, memory, and the ability to show (dump) the contents and any meta-data of registers and memory. You may add additional features.
You will need to add several CAT2 instructions to the VM as well as modify/extend/ write new "test" programs (benchmarks.)

Part 1
+ Implement the "Store" instruction. You will need it to benchmark.
+ Write a small (CAT) program to add two 1 x N vectors (that is, add the corresponding values in two lists of numbers) for example: [1, 2, 3] + [4, 5, 6] -> [5, 7, 9]
Where the count, N will be less than 32
+ Instrument the CAT VM to count number of instructions executed, clock cycles, and total number of memory references

Part 2
+ Implement (CAT) data hazard detection, as if CAT were pipelined with a 5 stage Pipeline that is, implement a scoreboard (or similar) to detect data hazards)
+ Implement control (branch) hazard detection and appropriate stalls (no need to pipeline, unless you wish to)
+ Show where the previous hazards exist (in the source asm/binary code), and the scoreboard results,
Show why these hazards exist, how long it is necessary to stall.
+ Implement one bit branch prediction for CAT
+ You will need to be able to show the values/contents of the previous data, that is what is: what is in the scoreboard, and the branch predictor table as CAT2 instructions are executed.

Part 3
+ Implement a L1 cache memory (in addition to RAM)
+ For the CAT architecture implement cache memory configurations for each of the following configuraions:
DM - Direct Mapped (Unified):
2 words wide, 4 entries (lines) (2 x 4)
4 words wide, 4 entries
2 words wide, 8 entries
SP - Split I and D (Instructions, Data):
I: 2 x 2, D: 2 x 2
I: 4 x 2, D: 4 x 4
SA - 2-way set associative:
Unified: 2 x 8
+ Optional/Bonus: and then add a simple L2: 8 x 8
+ You will test the performance of each using benchmarks that you created as well as supplied code. You should use your "vector add" (add two arrays) as a test program, you may need to add more instructions (code) and data to get good results

+ Please assume:
  L1 takes 1 clock cycle, L2 takes 4 clock cycles, main memory takes 25 clock cycles.
+ For each of these combinations, show:
  Number of memory references (I, D), Hit rates, contents of cache,
  performance (time)
  Use your benchmarks and/or code from exams (large enough code and data)

Part 4
+ Add the following superscalar instructions:
  Vector add (add elements similar to previous benchmark, element by element),
  Vector sum (sum elements in an array, in one operation)
+ Note: you must modify the 5 stage implementation

Note: You will need to demo the assignments to the GTA and instructor,
more details to follow