Microprocessors

Super-scalar out-of-order architectural simulator

Project Report

Team 21

Ahmad Elsagheer
Ahmed Kamel
Ahmed Soliman
Mostafa Abdullah

Implementation

1. Memory Organization:
   - MemoryHandler is responsible for dealing with the memory unit.
   - Every cache and the main memory extend a common abstract class “Stroage” which has two methods: fetch and write.
   - Every cache level is composed of Cache sets, each of which is composed of cache blocks. Each cache may have different different size, block size, associativity, writing policy and access time.
   - **Every cache communicates with the next memory level in words, such that the cost of accessing a word from each level in case of a miss is calculated as the reading cost in this level added to the reading cost for every word in this block from the next level.**

2. Processor: the processor unit handles the flow of instructions. It connects reservation stations, reorder buffer, instruction fetch unit and the register file. In each clock cycle, the processor:
   - allows the InstructionFetch unit to fetch a new instruction (if it has already been read from memory) and add it to the instruction queue.
   - issues new instruction from the queue front if there are reservation stations and reorder buffer entries available for them taking into consideration the pipeline width.
   - executes instructions whose operands are ready.
   - write result for instructions that finished their execution stage.
   - commit the head entry of the reorder buffer if it's ready.

3. Simulation
   - The simulator takes memory configurations and constructs the memory handler, then takes the processor configurations and constructs the processor, and then initializes the data in the memory according to the user's request. And the processor is then run until the termination command is committed. Then the required information are printed to the console.

Bonus features added

1. The user can provide an assembly program.
2. Unit Testing with a coverage test 99%

Team Work:

Pair/Triple programming

- Soliman/Teefa/Kammola: Memory Organization
- Teefa/Kammola: Unit Testing/Parser
- Soliman/Teefa/Kammola/Sagheer: Processor units (reservation stations, reorder buffer, processor, integration... etc)
- Soliman/Kammola: Simulator

User guide:

- Enter the memory configurations.

```
Memory configurations:
What is the number of cache levels?
2
What is the number of cycles required to access the main memory?
10
Please enter the configurations for every cache level in the following format:
S L M W A
Where S is the cache size, L is the block size, M is the cache associativity, W
is the writing policy (0 => write back, 1 => write through) and A is the number
of cycles to access this cache level.
Cache level 1
16 4 2 0 2
Cache level 2
32 2 1 0 4
Done configuring the memory hierarchy.
```

- Enter the hardware configurations.

```
Hardware Organization Configurations:
Please enter the pipeline width (the number of instructions that can be issued to
the reservation stations simultaneously):
10
Please specify the size of the instruction buffer (queue):
10
Please enter the number of ROB entries
10
Please enter the number of LOAD reservation stations, and the time to calculate
the address.
2 1
Please enter the number of STORE reservation stations, and the time to calculate
the address.
2 1
Please enter the number of ADD reservation stations, and the time to execute.
3 2
Please enter the number of MULT reservation stations, and the time to execute.
3 1
Please enter the number of NAND reservation stations, and the time to execute.
1 1
```

- Determine the path to the file containing the assembly code.
- Enter the starting address in the memory to write the program.

```
Please enter the path to the assembly program file.
program.asm
What is the initial address in the memory to write the program?
43
```

- Optionally initialize the memory with data.

```
Number of instructions completed is 3
Number of branches encountered is 0
Number of cycles spanned is 125
Data cache:
Level 1: 0 read hits, 0 read misses, 0 write hits, 0 write misses.
Level 2: 0 read hits, 0 read misses, 0 write hits, 0 write misses.
Instruction cache:
Level 1: 1 read hits, 2 read misses, 0 write hits, 0 write misses.
Level 2: 4 read hits, 4 read misses, 0 write hits, 0 write misses.
Number of branch misspredictions: 0
```

4. A list of programs (and associated data if any) you simulated. You should at least provide 3 programs. The programs must cover all instructions supported and one of them at least must have a loop.
Check *test/TrickyProgramsTests.java* file and *test/InstructionsTests.java*.

5. The hardware configurations (including memory hierarchy configurations) you used to simulate each program and the results obtained from each simulation. You must simulate each program with at least two different configurations.

Configurations 1:

- Memory access time: 2 cycles
- 1 cache level
- Cache size: 8 words, block size: 2 words, associativity: 1 way, writing policy: write-back, access time: 1 cycle.
- Pipeline width: 1
- ROB size: 4
- Instructions queue size: 4
- 2 LOAD reservation stations, 1 cycle to calculate address.
- 2 STORE reservation stations, 1 cycle to calculate address.
- 2 ADD reservation stations, 2 cycles to execute.
- 2 MULT reservation stations, 2 cycles to execute.
- 1 NAND reservation station, 2 cycles to execute.
- Program start address: 100
- Memory[32] = 2
- Memory[44] = 3

Configuration 2:

- Memory access time: 10 cycles
- 3 cache level
- Cache 1 size: 4 words, block size: 2 words, associativity: 1 way, writing policy: write-back, access time: 1 cycle.
- Cache 2 size: 8 words, block size: 2 words, associativity: 2 way, writing policy: write-back, access time: 3 cycle.
- Cache 3 size: 16 words, block size: 4 words, associativity: 4 way, writing policy: write-back, access time: 5 cycle.
- Pipeline width: 2
- ROB size: 5
- Instructions queue size: 5
- 1 LOAD reservation stations, 2 cycle to calculate address.
- 1 STORE reservation stations, 2 cycle to calculate address.
- 1 ADD reservation stations, 2 cycles to execute.
- 1 MULT reservation stations, 2 cycles to execute.
- 1 NAND reservation station, 2 cycles to execute.
- Program start address: 100
- Memory[i] = i + 1, 0 <= i < 20

6. A brief discussion of the obtained results.

Results from configurations 1:

```
Number of instructions completed is 27
Number of branches encountered is 5
Number of cycles spanned is 111
Data cache:
Level 1: 1 read hits, 0 read misses, 0 write hits, 1 write misses, hit ratio: 50.00%.
Instruction cache:
Level 1: 48 read hits, 7 read misses, 0 write hits, 0 write misses, hit ratio: 87.27%.
Number of branch misspredictions: 4 with percentage of 80.00%
Time spent to access memory: 89
AMAT: 1.85
IPC: 0.24
```

Results from configurations 2:

```
Number of instructions completed is 27
Number of branches encountered is 5
Number of cycles spanned is 358
Data cache:
Level 1: 1 read hits, 0 read misses, 0 write hits, 1 write misses, hit ratio: 50.00%.
Level 2: 1 read hits, 1 read misses, 0 write hits, 0 write misses, hit ratio: 50.00%.
Level 3: 1 read hits, 1 read misses, 0 write hits, 0 write misses, hit ratio: 50.00%.
Instruction cache:
Level 1: 21 read hits, 23 read misses, 0 write hits, 0 write misses, hit ratio: 47.73%.
Level 2: 39 read hits, 7 read misses, 0 write hits, 0 write misses, hit ratio: 84.78%.
Level 3: 10 read hits, 4 read misses, 0 write hits, 0 write misses, hit ratio: 71.43%.
Number of branch misspredictions: 4 with percentage of 80.00%
Time spent to access memory: 470
AMAT: 14.30
IPC: 0.08
```

Conclusions:

The processor with one level cache is faster in this case because the whole program is read from the memory only once, therefore in the case of one cache level, only the access time of the memory and this level is calculated only once, however with adding an additional cache level, its access time is added to the total access time.

Also, we have noticed that decreasing the number of the reservation station for the heavily used functional units leads to degrading the performance and increasing the number of cycles taken by the program, because some instructions are stalled waiting for free reservation station.