TOMASULO SIMULATOR / SOLVER

By
Devansh Sheth (862188538)
Mahip Shah (862188393)

Introduction:

For our final project, we have implemented a tomasulo simulator. All the details about the project can be found in this report.

Software Dependencies:

We have implemented this project in python3 specifically python 3.7. We have also used pandas to display the result as a dataframe. As a result, to execute the code, python3 and pandas are needed.

Execution Instructions:

To execute the code, run the following command in the terminal:

python tomasulo.py <input_filename> <adder_time> <mult_time> <load_buffers> <adder_slots> <mult_slots> <branch_slots> <branch_taken>

For <input_filename>, you can enter a file which will contain a set of instructions. The instructions our code supports are as following:

Instruction	**Notation**
Load	LW
Store	SW
Add	ADD
Subtract	SUB
Branch	BNE
Multiply	MULT
Divide	DIV

We have included a couple of sample instruction files to test the code in the folder.

For <adder_time>, you can enter the no. of cycles add/subtract instructions will take. For eg. 2

For <mult_time>, you can enter the no. of cycles multiply/divide instructions will take. For eg. 20

For <load_buffers>, you can enter the buffer slots for load/store instructions. For eg. 5
For <adder_slots>, you can enter the no. of reservation stations for add/subtract instructions. For eg. 3

For <mult_slots>, you can enter the no. of reservation stations for multiply/divide instructions. For eg. 2

For <branch_slots>, you can enter the no. of reservation stations for branch instructions. For eg. 2

For <branch_taken>, you can enter 1 if the branch is taken and 0 if the branch is not taken. If the branch is taken, the branch will exit after 4 iterations. If branch is not taken, the branch will exit after the first iteration.

Example of sample command:

python tomasulo.py instruction.txt 2 20 5 3 2 2 1

Implementation Assumptions and Limitations:

Assumptions:

1. If branch taken, branch exits after 4 iterations.
2. Add and Subtract instructions use the same reservation stations and take the same number of cycles.
3. Multiply and Division instructions use the same reservation stations and take the same number of cycles.
4. Load and Store instructions take the same number of cycles.
5. Maximum number of writebacks and commits are the same for one cycle.
6. The cache always hits and its access time is one cycle.
7. The branch unit takes one cycle.

Limitations:

1. Implemented just the Speculative Tomasulo Algorithm.
2. Does not allow multiple issues and multiple commits.
3. Nested loops are not handled.
4. Instructions after the branch will not be executed.

Sample Results:

For our sample runs, we show the output of the speculative tomasulo example from the midterm exam. We have converted the instructions to the appropriate format which the code supports.

Instructions:

LW R5 R1
LW R6 R2
LW R7 R3
ADD R8 R6 R5
ADD R9 R8 R7
ADD R1 R1 R10
ADD R2 R2 R11
ADD R3 R3 R12
SW R9 R1
BNE R4 R1

a) Branch Taken

Output:

```
(base) C:\Users\malvi\Desktop>python tomasulo.py instruction.txt 2 20 5 3 2 2 1
                  Issue  Execution Memory Write Back  Commit
0  LW R5 R1           1          2      3          4       5
1  LW R6 R2           2          3      4          5       6
2  LW R7 R3           3          4      5          6       7
3  ADD R8 R6 R5       4          6      -          8       9
4  ADD R9 R8 R7       5         10      -         12      13
5  ADD R1 R1 R10      6          8      -         10      14
6  ADD R2 R2 R11      9         12      -         14      15
7  ADD R3 R3 R12     11         14      -         16      17
8  SW R9 R1          12         13     14          -      18
9  BNE R4 R1         13         14      -          -      19
10 LW R5 R1          14         15     16         17      20
11 LW R6 R2          15         16     17         18      21
12 LW R7 R3          16         17     18         19      22
13 ADD R8 R6 R5      17         19      -         21      23
14 ADD R9 R8 R7      18         23      -         25      26
15 ADD R1 R1 R10     19         21      -         23      27
16 ADD R2 R2 R11     22         25      -         27      28
17 ADD R3 R3 R12     24         27      -         29      30
18 SW R9 R1          25         26     27          -      31
19 BNE R4 R1         26         27      -          -      32
20 LW R5 R1          27         28     29         30      33
21 LW R6 R2          28         29     30         31      34
22 LW R7 R3          29         30     31         32      35
23 ADD R8 R6 R5      30         32      -         34      36
24 ADD R9 R8 R7      31         36      -         38      39
25 ADD R1 R1 R10     32         34      -         36      40
26 ADD R2 R2 R11     35         38      -         40      41
27 ADD R3 R3 R12     37         40      -         42      43
28 SW R9 R1          38         39     40          -      44
29 BNE R4 R1         39         40      -          -      45
30 LW R5 R1          40         41     42         43      46
31 LW R6 R2          41         42     43         44      47
32 LW R7 R3          42         43     44         45      48
33 ADD R8 R6 R5      43         45      -         47      49
34 ADD R9 R8 R7      44         49      -         51      52
35 ADD R1 R1 R10     45         47      -         49      53
36 ADD R2 R2 R11     48         51      -         53      54
37 ADD R3 R3 R12     50         53      -         55      56
38 SW R9 R1          51         52     53          -      57
39 BNE R4 R1         52         53      -          -      58
```

b) Branch Not Taken

Output:

```
(base) C:\Users\malvi\Desktop>python tomasulo.py instruction.txt 2 20 5 3 2 2 0
                 Issue  Execution Memory Write Back  Commit
0 LW R5 R1           1          2      3          4       5
1 LW R6 R2           2          3      4          5       6
2 LW R7 R3           3          4      5          6       7
3 ADD R8 R6 R5       4          6      -          8       9
4 ADD R9 R8 R7       5         10      -         12      13
5 ADD R1 R1 R10      6          8      -         10      14
6 ADD R2 R2 R11      9         12      -         14      15
7 ADD R3 R3 R12     11         14      -         16      17
8 SW R9 R1          12         13     14          -      18
9 BNE R4 R1         13         14      -          -      19
```

Expected Output:

	Instruction	Issue	Exec	Mem Access	Write CDB	Commit
1	LD R5, 0(R1)	1	2	3	4	5
2	LD R6, 0(R2)	2	3	4	5	6
3	LD R7, 0(R3)	3	4	5	6	7
4	DADD R8, R6, R5	4	6-7		8	9
5	DADD R9, R8, R7	5	10-11		12	13
6	DADDIU R1, R1, #8	6	8-9		10	14
7	DADDIU R2, R2, #8	9	12-13		14	15
8	DADDIU R3, R3, #8	11	14-15		16	17
9	SD R9, -8(R1)	12				18
10	BNE R4, R1, LOOP	13	14			19
11	LD R5, 0(R1)	14	15	16	17	20

As we can observe, the code output matches the expected output.

Contribution :

Work	**Devansh**	**Mahip**
Code	50%	50%
Report	50%	50%

We worked together for the coding as well as for the report.