Report of Marwa7a Team

Member:

1. Omar Amin 46-1014 T-21 omar.seifallah@student.guc.edu.eg
2. Ahmed Sameh 46-0466 T-21 ahmed.alshabrawi@student.guc.edu.eg
3. Mai Alaa 46-2507 T-18 mai.mohamedammar@student.guc.edu.eg
4. Mohamed Abdelfattah 43-3148 T-21 mohamed.husien@student.guc.edu.eg
5. Abdelrahman Mostafa 46-1292 T-21 abdel-rahman.mosta@student.guc.edu.eg

Code Structure and Approach:

The code is made up of a constructor, a main method, 11 other methods, and a lot of global variables, arrays, array lists, and Java swing components.

- Constructor is used for the GUI it creates the three frames and a lot of buttons and panels then combine everything together in a specific format with some extras like font colors, button colors and icon image.
- Main method only calls the instructor
- actionPerformed method contains what every button in the program does and gets also the input form the user in text fields for the registers and the latencies and saves all the instructions in a string r and string rr (r for only display in JFrame as it is written in html).
- read method reads the file r that was split by coma and parenthesis in an array stArr and counts the number of instructions and number of registers and puts all registers in a sorted array list.
- insert method puts all the instructions and registers in the queue and we create the queue table by making a 2D array using the number of instructions as rows and the same goes for the registers table.
- simulation method contains three sections one for the issue one for the execute and one for the write back and they are marked by comments. In the issue part we go and check if we are not at cycle 0 then we issue the instruction by putting the cycle number in the issue column then we put the register in column i in the busy list as we will use it and then we check on the registers needed (j and k) if they are busy or not and we fill the reservation stations according to the type of the instruction. In the execute part we check if the (j and k) are ready to use or not yet if yes we start decreasing the timer or the latency of the instruction at every cycle and we fill the execute column by the cycle number we executed this instruction in. Finally in the write back part we check if the instruction finished executing and if yes we return its value for the other instructions in the reservation stations that are waiting for its value then they start to execute. Finally the register i is removed from the busy list.
- refresh method is used every time we press next alongside the simulation as when the simulation method is executed there will be changes in the data of the

tables so to display these changes we need to create the tables again with same size and position but with the new data from the previous cycle.

- printArr method is used only to print arrays and 2D arrays rather than looping every time.
- printArr1 is a special print method only for the instructions table (or queue table) as the names of the columns are quite large for the CLI to match with the data so we decreased the typing only for the CLI.
- printCDB prints the final values that are written to the CDB and is printed at the end of the program when the simulation finishes.
- printAll prints all of the tables in the CLI every cycle like the GUI but in CLI.
- checkEOP which checks if we are at the end of the simulation and the simulation has finished by making sure that every instruction has finished executing and wrote back the results and this method is executed every time we press next in the simulation if it returns true the red next button turns green and says "END" which gets us to the CDB.

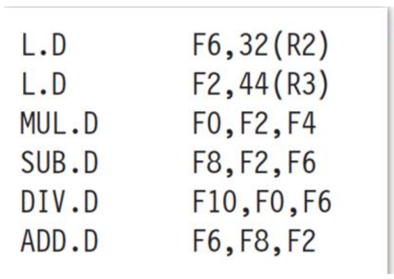
Testing and Test Cases:

- First test was manual entry of the instructions using the text field and buttons and entering the latencies manually (Note if you left a latency blank it will automatically be the default value from lecture 12).
- Another test was uploading a text file that contains the instructions
- Main test was using the exact same code from lecture 12, it is available at the end of the report, the goal was getting the exact same results, and we did.
- Final tests where testing the errors like leaving the text field blank and pressing on an instruction or entering submit without entering any instruction or even uploading an empty file with no instruction.

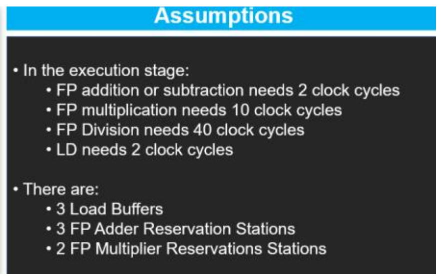
Default Latencies:

- S.D: 1
- L.D: 2
- ADD.D: 2
- SUB.D: 2
- MUL.D: 10
- DIV.D: 40

Code of Lecture 12:



```
L.D      F6,32(R2)
L.D      F2,44(R3)
MUL.D    F0,F2,F4
SUB.D    F8,F2,F6
DIV.D    F10,F0,F6
ADD.D    F6,F8,F2
```



Assumptions

- In the execution stage:
  - FP addition or subtraction needs 2 clock cycles
  - FP multiplication needs 10 clock cycles
  - FP Division needs 40 clock cycles
  - LD needs 2 clock cycles
- There are:
  - 3 Load Buffers
  - 3 FP Adder Reservation Stations
  - 2 FP Multiplier Reservations Stations