EG 211 Computer Architecture Assignment – 2: Marks: 45
Oct 2023

Guidelines:

- To be done individually or in groups of 2 (not more than 2)
- You can choose to do the design in any language you wish, C, C++, python, Verilog etc
- **What to submit:**
  - **Submit your report (in pdf format) which contains roll numbers and names of the students in the group, some explanation of the code, the program you chose to implement, any assumptions in your processor design, snapshots of result etc.**
  - **Upload codes separately.**
  - **Upload the machine code of the programs separately**
- The submission will be followed by a viva/demo
- When you submit the code, rename the filename of your code to <roll_numbers>_filename.< >
- All codes will run through a plagiarism check. Files found similar will get a 0 for the assignment. Repeat offence will attract Grade penalty on the overall grade
- Submit by Nov 9, 2023, 11:59pm on LMS under Assignment 2
- Marks: 35 + 10 for viva
- The following two tasks can be done in parallel by the two members in the team

1. Non-pipelined MIPS processor design – **15 marks**
   1. Take as input the machine code of instructions generated by the assembler in the previous assignment. The program should do something substantial – such as, matrix multiplication, factorial, convolution and not just a simple add/mul. Store this machine code in instruction memory (array for example) .For this assignment, you can ignore the template code, system calls etc. If you were doing factorial, just have the machine code for computing factorial of N. The value of N can be hardcoded or read from memory in a loop.
   2. Design a processor with 5-stages:
      - To read in the machine code of instructions one after the oher
      - Decode the instruction
      - Execute it in an ALU
      - Perform memory access (Data memory- another array) and
      - Writeback to register file (another array)
      - Repeat this for all instructions in a non-pipelined fashion
      - Keep track of the number of clock cycles. You do not need a clock, but use a counter to keep track of the number of pipeline stages the entire program goes through. Print this out also.

3. Desired output: If you chose a sorting algorithm, the output should be sorted integers (either in ascending or descending order) stored in a certain array. You can also display outputs such as PC, outputs of each pipeline stage etc for better understanding. Print out the number of clock cycles taken by the entire program
4. Execute at least **two programs** on the processor you designed. You can could take the machine code of these programs from other groups also. The program does not carry any marks.

2. Non-pipelined MIPS processor design - **20 marks**
   1. The inputs and outputs remain same as in the non-pipelined scenario.
   2. The IF, ID… WB will occur in a pipelined fashion, that is an instruction is fetched every cycle, and does not wait until the previous instruction is completed.
   3. Assume no structural hazards
   4. You should have data and control hazards in your program ideally!
   5. Keep track of these hazards and resolve them using stalls or forwarding. Forwarding should be done only in case of dependencies.
   6. Control hazards can be resolved using pipeline flushes. Branch predictor need not be implemented for this assignment
   7. Run this again for the same two programs as in the case of a non-pipelined processor. Note the number of clock cycles these programs took to execute. Observe that the pipelined processor executed the programs faster than the non-pipelined processor. You could also choose to print out a pipelined diagram, hazards etc- this is for better conceptual understanding

   Note
   - You are free to use any programming language
   - Instruction and Data memory can be declared as arrays of limited size (need not be 2^{32})
   - Memory can be word addressable (each memory location can store 32bits unlike byte-addressable memory used in MIPS where each memory location can store 8 bits)