1. **Draw the schematic for a single stage processor and fill in your code in the to run the simulator. (20 points)**

Diagram

Description automatically generated

1. input PC to get address and read instructions.  
2. decode instructions and get data in registers.

3. making ALU operations.

4. (lw)loading data or (sw)write data to memory.

5. write to registers and update PC.

1. **Draw the schematic for five stage pipelined processor and fill in your code to run the simulator. The processor should be able to take care of RAW and control hazards by stalling and forwarding. (20 points)**

Diagram

Description automatically generated

1. IF: Instruction fetch from memory

2. ID: Instruction decode & register read

3. EX: Execute operation or calculate address

4. MEM: Access memory operand

5. WB: Write result back to register

Hazaard:   
1. from EX/MEM:

1a. EX/MEM.RegisterRd = ID/EX.RegisterRs1

1b. EX/MEM.RegisterRd = ID/EX.RegisterRs2

when facing situations that the previous operation is performing ALU operations while the current instruction is reading data in register, the hazard happens and we need to add a stall and forwarding to fix this.

Simply route the contents to be written into RD From EX/MEM column of Pipeline Registers to ID/EX column.

2. from MEM/WB:

2a. MEM/WB.RegisterRd = ID/EX.RegisterRs1

2b. MEM/WB.RegisterRd = ID/EX.RegisterRs2

when facing situations that the previous operation is writing data back to register while the current instruction is reading data in register, the hazard happens and we need to add a forwarding to fix this.

Simply route the contents to be written into RD From EX/MEM column of Pipeline Registers to ID/EX column.

1. **Measure and report average CPI, Total execution cycles, and Instructions per cycle for both these cores by adding performance monitors to your code. (Submit code and print results to console or a file.) (5 points)**

Since single stage design only have one cycle per instruction. Average CPI and IPC are the same which is one.

For five stages, the CPI depend on the ISA specific percentages on different instruction. For example, a SW function which does not include any operation in WB stage will not have 5 cycle for that instruction. The length of the code segment also matters, since the total execution cycles is around total instruction execute + 4. However, as the program grows bigger and longer this constant value of cycles lose its significance.

1. **Compare the results from both the single stage and the five stage pipelined processor implementations and explain why one is better than the other. (5 points)**

Five-stage pipelined can execute more instructions simultaneously, thus increasing the overall performance of the CPU.

Single stage processors are much simpler in design and require less energy while executing the same ISA. The simple design made it endurant facing hazard situation. However, five stage design sacrifice energy usage and stableness in exchange of execution time. Additionally, a five stage pipeline design is more complex than single stages ones.

1. **What optimizations or features can be added to improve performance? (Extra credit 1 point)** 
   1. the code we wrote still include plentiful segment that can be decoupled. Ie. Similar or identical code block are written in different branch of if.
   2. The code is written in one file, which makes it much harder to debug than file written in multiple component composed program
   3. The code have five stage processer and single stage processor together which can be troublesome when programmer want to see the logs.
   4. The program come with some logs feature that can be very useful in terms of debugging. However, a toggle mode for all debugging console print should be included.