Assignment - 8

Aarunish Sinha - 2018CS10321
Sharique Shamim - 2018CS10388

In this assignment we extend our MIPS Simulator from the Assignment 7. We have introduced the idea of pipelining into the processor. Whenever we have a data hazard or a control hazard we stall the pipeline and let the stall-causing-instruction execute. We do not deal with structural hazard here since we are designing a software simulator.
We have tokenised and stored each instruction in a custom struct 'Instruction'. We have five functions to execute an instruction namely IF, ID, EX, MEM and WB. The STALL and DATA_HAZARD_STALLING functions help with stalling the pipeline in case of a possible data hazard and the BRANCH_STALLING function helps to stalling the pipeline in case of branching or in general a control hazard.
We have printed the active instruction at every stage of the pipeline by specifically stating the name of the stage and the exact instruction in that particular stage. In the end we have printed the register file, occupied memory elements, total number of clock cycles passed and the Instruction Per Cycle(IPC) as done in Assignment 7.
We have provided six test cases along with the submission and have explained test case 1 below,

```
lw $0 1024
lw $1 1025
lw $2 1026
sub $7 $0 $1
add $8 $1 $2
lw $3 1027
add $9 $2 $3
sw $7 1031
sw $9 1032
exit
```

In the above instruction set the would be a data hazard in the "add $9 $2 $3" due to the load word instruction above it. Hence the pipeline is stalled for 2 extra clock cycles.
The total clock cycle count is 15 and the Instruction Per Count is 0.6.

We are also attaching the screenshot of outputs of all the test cases in the submission folder for reference.