Program	Number of instructions executed	Number of cycles taken	Number of times an instruction on a wrong branch path entered the pipeline	Number of times the OF stage needed to stall because of a data hazard	Throughput in terms of number of instructions per cycle
descending.asm (Lab 4 Given Test Case)	366	15325	3468	0	0.023882544861337684
evenorodd.asm (Lab 4 Given Test Case)	7	254	0	0	0.027559055118110236
Lab 2 -> descending.asm	203	8466	1072	0	0.02397826600519726
Lab 2 -> even-odd.asm	7	254	0	0	0.027559055118110236
Lab 2 -> fibonacci.asm	89	3656	359	0	0.024343544857768053
Lab 2 -> palindrome.asm	22	869	158	0	0.02531645569620253
Lab 2 -> prime.asm	13	500	39	0	0.026

Observations:

- The number of cycles has increased many folds due to added functionality of latency of Main Memory, multiplier, divider, and ALU.
- The number of instructions executed is found to be the same. (They have to remain the same for obvious reasons)
- Due to the high latency of Main Memory, there occurs no Data Hazard because as long as the next instruction is fetched from memory, the previous instruction has done execution.
- Now, one cycle is equivalent to 40 (Main memory latency) or more cycles.
- The frequency of instructions entering the wrong path also depends upon the input data values stored in register and memory and whether the branch conditions satisfy true or not, satisfying to true leads to the instruction being fetched on the wrong data path.
- The more number of times branch condition satisfies to true, the more is the number of times instruction entered the wrong path in the pipeline and we need to set that instruction to NOP and hence it will lead to wastage of that cycle and hence increasing the number of cycles.

- The more is the Read After Write operations from the same position, the more will be the Number of times OF stage needed to be stalled and it will therefore also increase the number of cycles because of stalling.

Submitted By:

Mayank Singla