180010008/180010019/180020024

Name of Programs	No. of cycles	No. of time OF needs to be stalled due to data hazard	No. of times an instruction on a wrong branch path entered the pipeline
descending.out	543	210	52
evenorodd.out	14	4	0
fibonacci.out	130	40	8
palindrome.out	93	33	7
prime.out	53	19	1

Comments:

1. **descending.out** →

This program is quite long as it has many loops and it goes through them many times. That's why it takes 543 cycles to complete its execution. Due to its length many data dependencies occur so processor stalled many times and same reason for may wrong branches taken as there are so many lines of code being executed.

2. evenorodd.out →

This program is quite short and given input is ‘11’ in benchmark program. There is only one branching in this program which is never taken as input is odd. So that’s why wrong branch input path is zero here. And there are 4 stalls due to data dependencies 2 times (b/w load and divi & b/w sub and addi).

3. fibonacci.out →

Same, this program is also quite long as descending.asm and many loops being executed, so, many cycles, many stalls and many wrong branch paths being taken.

4. palindrome.out →

Same, this program is also quite long as descending.asm and many loops being executed, so, many cycles, many stalls and many wrong branch paths being taken.

5. prime.out →

In this program, there are only 2 branches which will be taken when deciding phase of program comes. This deciding phase comes only once at the end and one out of two branch paths is definitely is going to be correct and one definitely wrong branch path. So, that’s why wrong branch path will always be taken once and stalls in OF done due to data dependencies are also taken.