# CS311: Assignment 4

Ansh Vivek - 210020002
Atharv Gade – 210020004
Pratham Ingavale - 210020015

March 15,2024

## 1    Observations

Table 1: Observations

| Program  Name | Cycles | Branch Interlock | Data Interlock |
|---------------|--------|------------------|----------------|
| Descending    | 600    | 176              | 143            |
| Even odd      | 16     | 0                | 6              |
| Fibonacci     | 148    | 32               | 34             |
| Palindrome    | 117    | 14               | 50             |
| Prime         | 55     | 12               | 10             |

"Cycles" represents the count of cycles utilized by each benchmark program. "Branch Interlock" represents how many instances occurred when an instruction following the incorrect branch path entered the pipeline. "Data Interlock" re- flects the number of occasions when the OF stage had to stall due to a data hazard.

## 2    Inferences

In the case of evenorodd.asm, where there are no branch statements, it's evident that there were no instances of instructions entering the pipeline on the wrong branch path, resulting in a count of 0 for such occurrences. As the complexity of the code increases, we notice a rise in conflicts, both in terms of instructions entering the pipeline on the wrong branch path and the number of times the OF stage had to stall due to data hazards. Among the five benchmark programs, de-scending.asm exhibits the highest values in these metrics, while evenorodd.asm demonstrates the lowest values. It's worth noting that the number of cycles also

provides an explanation for the code's complexity. However, it's important to highlight that we cannot make a generalization that conflicts increase solely due to program complexity, but there appears to be a trend within these five bench-mark codes where complexity tends to be associated with a higher frequency of conflicts.