CA Assignment - 2

Swetha Murali - IMT2022018
Akshaya Bysani - IMT2022579

Programs we used:

Factorial:

An input integer will be given and the output will be the factorial of the given input.

Fibonacci:

An input integer (n) will be given and the output will be the nth integer of the Fibonacci series.

Non - pipelined processor:

In the non-pipelined processor, there are dictionaries of opcodes, register numbers, and all the control signals. We also have Instruction memory (IMem) which is the output of our MIPS code run in the MIPS assembler of the previous assignment. Each pipeline stage is represented by a method of the class NonPipelinedProcessor. For each instruction, all the stages are called consecutively in each clock cycle. The inputs for both programs can be given while executing the program.

Output:

This is the output of the non-pipelined processor for the factorial & Fibonacci algorithms.

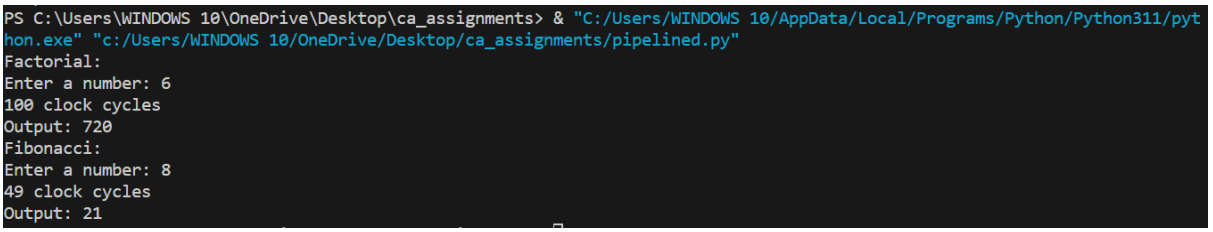
```
PS C:\Users\WINDOWS 10\OneDrive\Desktop\ca_assignments> & "C:/Users/WINDOWS 10/AppData/Local/Programs/Python/Python311/python.exe" "c:/Users/WINDOWS 10/OneDrive/Desktop/ca_assignments/nonpipelined.py"
Factorial:
Enter a number: 6
340 clock cycles
Output: 720
Fibonacci:
Enter a number: 8
175 clock cycles
Output: 21
```

Pipelined processor:

In the pipelined processor, we have started executing the next instruction immediately after the IF stage of the previous instruction and we have used individual queues for each stage for this to happen. This allows us to run different pipeline stages of different instructions within the same clock cycle. We have also taken care of dependencies and forwarding using a "dependencies" dictionary which stores all the destination values in each instruction and popping them after their usage to obtain the most recent value of the register. To handle branching, we assumed that the branch is never taken and flushed the output of future instructions once the branch decision is made by emptying the IF, ID, and EX queues. The inputs for both programs can be given while executing the program.

Output:

This is the output of the pipelined processor for the factorial & Fibonacci algorithms.



As we can see the pipelined processor executed both programs in fewer clock cycles than the non-pipelined processor.