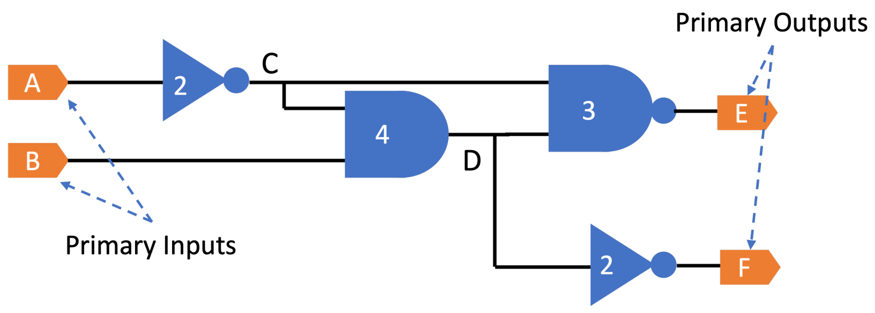
COL215P Software Assignment 1

Input and Output Delays in a Circuit

August 2023

Abhinav Rajesh Shripad (2022CS11596)
Aneeket Yadav (2022CS11116)



Contents

1 Problem Description

1.1 Part_A

To compute the delay of every primary output in a combinational circuit (defined as the earliest time when the output is ready), given the circuit representation and delay information of the individual gates.

1.2 Part_B

The problem entails determining the specific timing for preparing input signals to achieve designated outputs at desired times. This involves accounting for inherent gate delays and arranging input signals accordingly.

2 Design Consideration

2.1 Gate Representation

To encapsulate gate properties encompassing input1, input2, output, and delay, we introduced a versatile Gate class. Notably, the inverter gate was strategically fashioned to have both the two input parameters to be equal. This innovative approach eliminates the need for an additional class dedicated solely to a single input parameter.

2.2 Auxiliary Functions

2.2.1 `read_circuit`

This function systematically enumerates the inputs, outputs, and internal signals associated with the circuit, initializing their values to predefined states. Specifically, for part A, all values are set to 0, while for part B, input and intermediate signals are initialized to infinity. The rationale behind this approach will become evident during the subsequent algorithmic discussion. Concurrently, it creates gate objects corresponding to the specified parameters, laying a robust groundwork for subsequent operations.

2.2.2 `gate_delays`

Creates global variables for gates which store the inherent delay caused by a particular gate.

2.2.3 `calculate_output_delays`

This is employed for Part A.The purpose of this function is self-explanatory.

2.2.4 `required_output_delays`

Responsible for storing the output delays as per the specification of the problem in the final output variables.This is employed for Part B

3 Algorithm Description

The implementation is primarily composed of three functions which are as explained below:

3.1 `Topological_Sort`

This function sorts gates to align with their processing sequence. It addresses the potential disparity between input order and actual connections in `circuit.txt`. The output-linked gate, which might be first, is positioned appropriately. For part B, this order is reversed.

1. A dictionary is managed, mapping each output and intermediate signal to 0 and each input signal to 1. This indicates whether a signal is processed.

2. Gate processing ensues: If both inputs are marked as 1 in the dictionary, the gate is appended to the sorted gates list, and its output is flagged as 1 in the dictionary. This cycle iterates until all gates are processed.

3. Consequently, the resulting sorted gates list conforms to the required order.

3.2 `calculate_output_delay`

The function is composed of two main components that contribute to the total output delay of a gate:

1. **Later Input Delay**: Depending on the arrangement of the gates in the circuit, some inputs might take longer to arrive at the gate than others. This delay in the arrival of input signals is referred to as the "input delay." When calculating the output delay of a gate, it's important to consider the input signal that arrives last because this will be the main contributing factor to the overall delay.

2. **Inherent Gate Delay**: Inherent Gate Delay: Each gate in a digital circuit has an inherent delay associated with it. This is the time it takes for the gate to process the input signals and produce the corresponding output. It's a fixed delay that doesn't change based on the input values.

So, mathematically,

$$total\_output\_delay = later\_input\_delay + inherent\_gate\_delay$$

The function iterates over a list `sorted_gates` which is the output of the `topological_sort` function.Following the above specified logic,at its termination,the output variables store the least time at which the corresponding outputs are ready.

3.3 `calculate_input_delays`

This specific function distinguishes this problem from the previous one. The effective delay of a gate is calculated as follows:

$$Effective\ delay\ of\ gate = \ Delay\ of\ later\ signal\ +\ Inherent\ delay$$

Thus, the delay of the later signal can be derived as:

$$Delay\ of\ later\ signal\ \ =\ Effective\ delay - Inherent\ delay$$

From basic mathematics

$$a = min(a, max(a, b))$$

Thus, we can further deduce that

$$input1 = min(input1, delay\ of\ later\ signal)$$

$$input2 = min(input2, delay\ of\ later\ signal)$$

These inputs subsequently serve as outputs of the preceding gates. The input parameter of the function is `sorted_gates` which is the output of the `topological_sort` function.As the loop progresses, we traverse the circuit in reverse order, ultimately determining the times at which inputs must be supplied concerning the attainment of outputs.

4 Time Complexity Analysis

4.1 Helper Function

The helper functions such as `read_circuit`,`read_gate_delays`, `required_output_delays` go through each line of the input program only once and thus have time complexity as $O(n)$ where n is the size of the input for each particular case.

4.2 `Topological_sort`

The time complexity of this function is based on the ordering of the input given to us. If it is perfectly ordered it takes $O(n)$ time and in worst case scenario when all the gates are given in reverse order it takes $n-1$ iteration for first gate to be processed and similarly $n-2$ for the next and so on for others. Thus the worst case time complexity is $O(n^2)$.

4.3 `calculate_output_delays(sorted gates)`

Since the for loop iterates over all the gates,the time complexity for this function is $O(n)$ where n is the number of gates.

4.4 `calculate_input_delays(sorted gates)`

Since the for loop iterates over all the gates,the time complexity for this function is $O(n)$ where n is the number of gates.

Thus the time complexity of the program in worst case scenario is $O(n^2)$

5 Testcases

We have tested our code on few of the testcases which helped up in resolving any hitherto unforeseen problem in our program.

1. `Testcase_1:-`It encapsulates the idea when 2 inputs interacts in multiple way with one another.
2. `Testcase_2:-`It represents a simple circuit having multiple gates at the same level. It shows that the order of processing wont matter for such "equi-levelled" gates.
3. `Testcase_3:-`It considers a circuit when output of multiple gates interact with the input of some-other gates and considers it effects.
4. `Testcase_4:-` It is a similar testcase as 3 considering different connections.
5. `Testcase_5:-`It considers the case when multiple paths exists from input to output.

Through the use of these test cases,we have verified the outcomes of the following processes-

1. Effect of varying the delay time of different gates
2. Increasing the number of inputs
3. Increasing the number of gates
4. Permuting gates having longer inherent delays near the input and output sides of the circuit

6 Submission Details

The submission consists of a `.zip` file which includes the testcase folders and `main.py` file. Each testcase folder contains the following items:

1. `main.py`
2. `circuit.txt`
3. `gate_delays.txt`
4. `input_delays.txt`
5. `output_delays.txt`
6. `required_delays.txt`