Digital Design Project Report

Course Name: ESS 102, Digital Design
Student Name: KNV Aditya, Vinay Kusumanchi, Abhijit Dibbidi, T Shantanu
Student ID: IMT2023033 IMT2023608 IMT2023054 IMT2023056
Group ID: 31

30 November 2023

Abstract

Combinational Circuits are circuits made up of different types of logic gates. A logic gate is a basic building block of any electronic circuit. The output of the combinational circuit depends on the values at the input at any given time. Sequential circuits are digital circuits that store and use the previous state information to determine their next state. Unlike combinational circuits, which only depend on the current input values to produce outputs, sequential circuits depend on both the current inputs and the previous state stored in memory elements

1 Combinational Circuit: 4:1 Multiplexer

1.1 Description

A 4:1 multiplexer is a digital logic circuit that has four input lines, one output line, two select lines, and is used to select one of the four input lines and route it to the output line based on the values of the control lines. Inputs (d0, d1, d2, d3): These input lines are the signals you want to choose from and pass to the output. Output (y): The output line is where the selected input is transmitted. Select lines (s0, s1): These select lines determine which of the four inputs is given to the output. Functionality: The select lines (s0 and s1) determine which input line is transmitted to the output.

When s1 = 0 and s0 = 1, d1 is selected. When s1 = 0 and s0 = 1, d1 is selected. When s1 = 1 and s0 = 0, d2 is selected. When s1 = 1 and s0 = 1, d3 is selected.

Applications: 4:1 multiplexers are used in various applications, such as data routing in digital systems, memory address decoding, selecting between multiple input sources, and as a basic building block for constructing more complex circuits.

1.2 Implementation

Sub circuit design :

Boolean Expression: Y = (s1' * s0' * d0) + (s1' * s0 * d1) + (s1 * s0' * d2) + (s1 * s0 * d3)
Here, ' indicates a logical NOT operation, + represents logical OR and * represents logical AND.
So, we can design the circuit using NOT, OR and AND gates

Symbol design :

The 4:1 MUX symbol contains 4 inputs d0, d1, d2 and d3, 2 select lines s0 and s1,
vdd(main voltage), vss(ground voltage) and an output y.

Truth table :

s1	s0	y
0	0	d0
0	1	d1
1	0	d2
1	1	d3

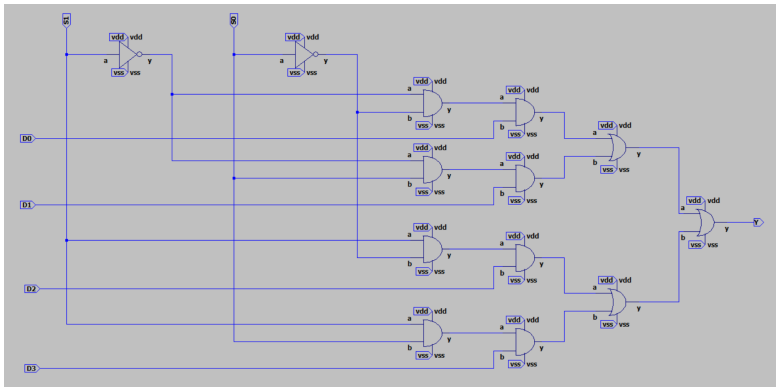

Figure 1: Combinational Schematic

d0 (in V)	d1 (in V)	d2 (in V)	d3 (in V)	s1	s0	y (in V)
5	X	X	X	0	0	5
0	X	X	X	0	0	24.58 * 10^(-9)
X	5	X	X	0	1	5
X	0	X	X	0	1	24.58 * 10^(-9)
X	X	5	X	1	0	5
X	X	0	X	1	0	24.58 * 10^(-9)
X	X	X	5	1	1	5
X	X	X	0	1	1	24.58 * 10^(-9)

1.3 Results :

We took four inputs d0, d1, d2 and d3, two select lines s0 and s1 and we got an output y.

We used 5V for logic 1 and 0V for logic 0. Following were the results when we passed different input voltages in testbench.

As 24.58*10^(-9) is is negligible in comparison to 5, we can consider it to be equal to 0.

We have verified our results using the testbench.

2 Mod-5 Down Counter

2.1 Description

Here is how a modulo-5 down counter works:

1. It requires 3 flip-flops to represent the 5 states from 0 to 4 encoded as binary.
2. Initially, the flip-flops are set to the binary equivalent of 4 (100). This represents the counter starting at the state 4.
3. On each clock pulse, the state transitions to the next lower state, i.e. a decrement operation:

- 100 (4) -¿ 011 (3)
- 011 (3) -¿ 010 (2)
- 010 (2) -¿ 001 (1)
- 001 (1) -¿ 000 (0)
- 000 (0) -¿ 100 (4)

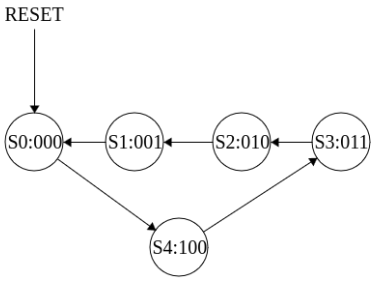

Figure 2: State Transition Diagram

So the state sequence is: 4, 3, 2, 1, 0, 4, 3, 2, 1, 0, ...

As the name suggests, instead of counting from 0-4 then rolling over, this counter rolls over after 5 states to repeat a modulo-5 sequence. In summary, a mod 5 down counter uses 3 flip-flops, is preset to 4, and decrements the state on each clock cycle to go through the repeating sequence 4, 3, 2, 1, 0, 4, 3... This creates a modulo-5 count sequence.

2.2 Implementation

Here is one way to implement a modulo-5 down counter circuit using 3 D flip-flops:

1. Use 3 D flip flops to create a 3-bit counter. Call them FF2, FF1, FF0.
2. Connect the clock to each flip-flop so they switch state on the rising edge.
3. Set the initial state to 100 by tying FF2 high, FF1 low and FF0 low. This stores the state 4.
4. Connect the inverted Q output of FF0 to the D input of FF1.
5. Connect the inverted Q output of FF1 to the D input of FF2.
6. Connect the reset of FF0 to FF2's inverted Q output.
7. Connect the D inputs of FF0 and FF2 to logic HIGH.

This creates the following transition table on each clock pulse:

FF2 FF1 FF0 1 0 0 (100) 4 1 0 1 (101) 3 1 1 0 (110) 2 1 1 1 (111) 1 0 0 0 (000) 0 1 0 0 (100) 4 (restart)

The inversions and resets decrement the count each cycle. The rollover to 100 sets state back to 4.

So this circuit wiring implements a modulo-5 down counter through the state transitions.

2.3 Results

[**?**].

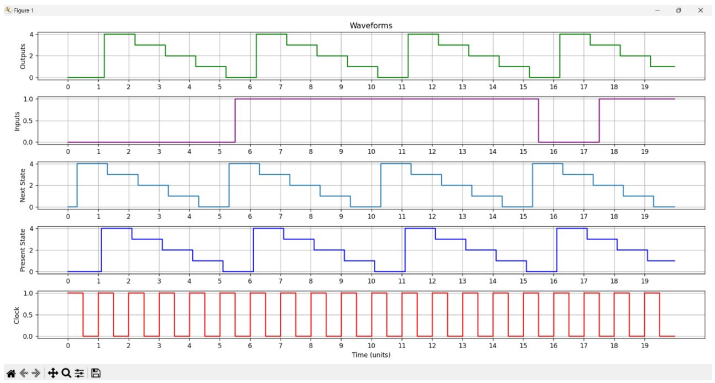

Figure 3: Graph

```
Time: 0.5, State: 000, Output: 000
Time: 1.5, State: 100, Output: 100
Time: 2.5, State: 011, Output: 011
Time: 3.5, State: 010, Output: 010
Time: 4.5, State: 001, Output: 001
Time: 5.5, State: 000, Output: 000
Time: 6.5, State: 100, Output: 100
Time: 7.5, State: 011, Output: 011
Time: 8.5, State: 010, Output: 010
Time: 9.5, State: 001, Output: 001
Time: 10.5, State: 000, Output: 000
Time: 11.5, State: 100, Output: 100
Time: 12.5, State: 011, Output: 011
Time: 13.5, State: 010, Output: 010
Time: 14.5, State: 001, Output: 001
Time: 15.5, State: 000, Output: 000
Time: 16.5, State: 100, Output: 100
Time: 17.5, State: 011, Output: 011
Time: 18.5, State: 010, Output: 010
Time: 19.5, State: 001, Output: 001
```

Figure 4: Terminal Outputs