Objectives

- To get familiar and understand Basic Logic Gates
- To introduce the gate behavior and logic interpretation
- To install and able to use Logisim Software

Introduction

A Digital Logic Gate is an electronic circuit which makes logical decisions based on the combination of digital signals present on its inputs. That circuit uses digital inputs to make logical decisions and produce digital outputs. Every logic circuit requires at least one input, before it can produce any kind of output. Digital logic inputs and outputs are usually binary.

Logic Gates

Logic gates are the heart of digital electronics. A gate is an electronic device which is used to compute a function on a two valued signal. Logic gates are the basic building block of digital circuits. Basically, all logic gates have one output and two inputs. Some logic gates like NOT gate or Inverter has only one input and one output. The inputs of the logic gates are designed to receive only binary data (only low 0 or high 1) by receiving the voltage input.

We can connect any number of logic gates to design a required digital circuit. Practically, we implement the large number of logic gates in ICs, by which we can save the physical space occupied by the large number of logic gates. We can also perform complicated operations at high speeds by using integrated circuits (IC). By combining logic gates, we can design many specific circuits like flip flops, latches, multiplexers, shift registers etc.

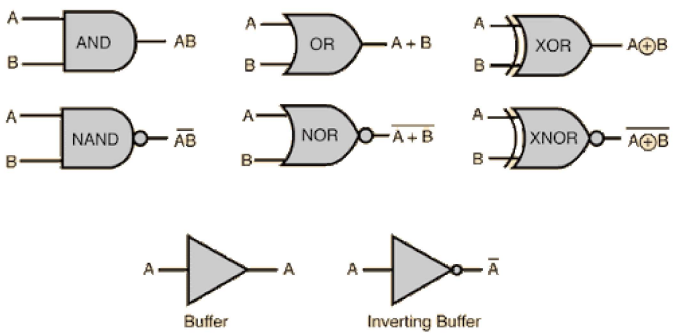


Figure 1: Logic Gates

The 7400 Series of Logic Gates

IC 7400 is the most popular logic family in integrated circuits. The IC 7400 is a 14-pin chip and it includes four 2-input NAND gates. For breadboarding and experimental purposes, we will use different 7400 series Dual-Inline Package (DIP) ICs.

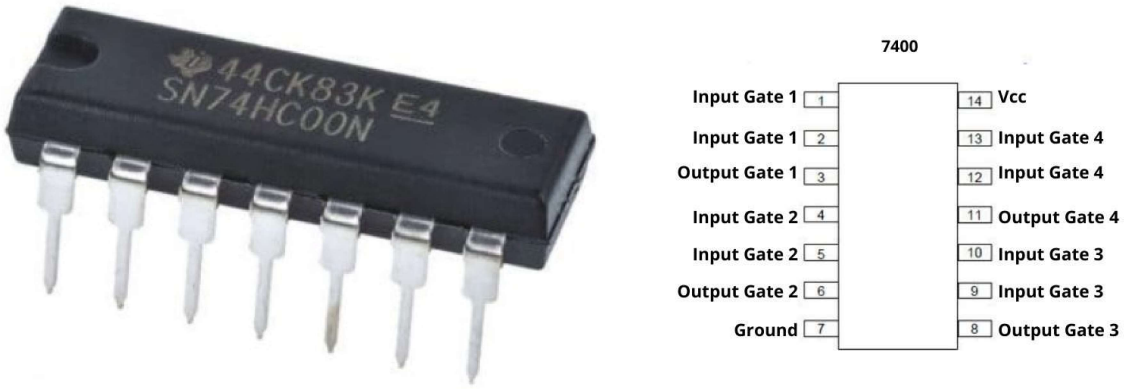


Figure 2

Lab Tasks

1. Install and run Logisim

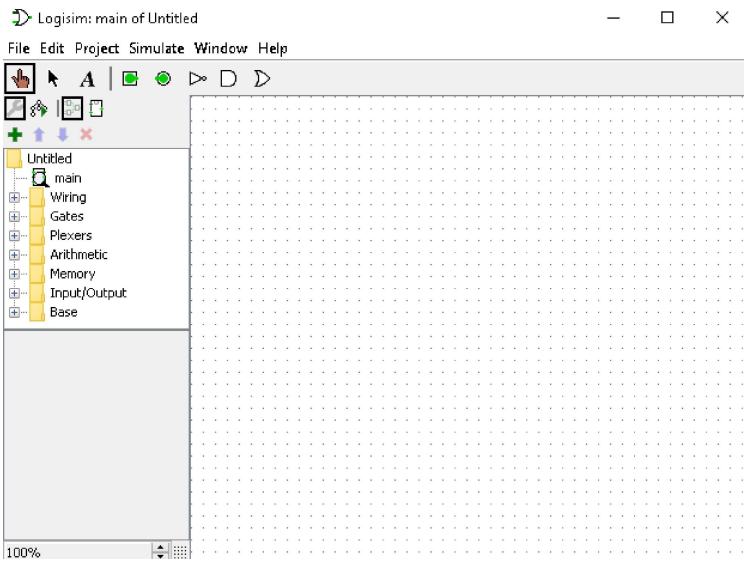
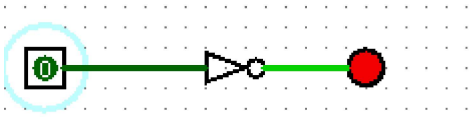


Figure 3: Logisim Main Window

2. Verify each basic logic gate in Logisim and note the Truth Table.

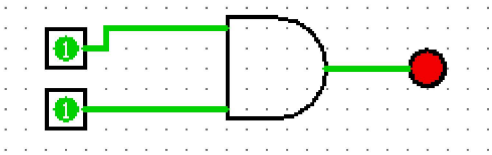
- **NOT Gate**



A	Y
1	0
0	1

Figure 4: NOT Gate

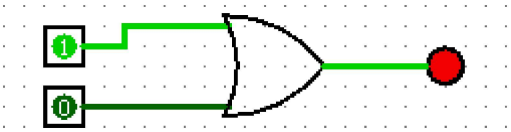
- **AND Gate**



A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Figure 5: AND Gate

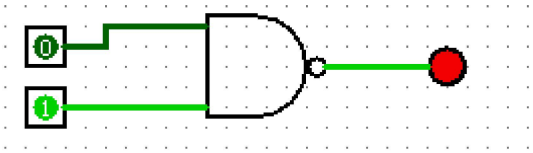
- **OR Gate**



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

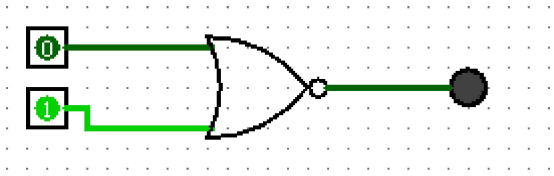
Figure 6: OR Gate

- **NAND Gate** Figure 7: NAND Gate



A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

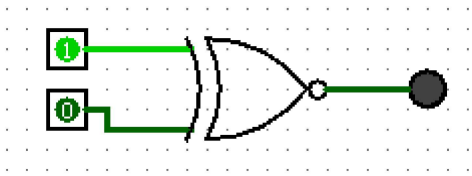
- **NOR Gate**



A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Figure 8: NOR Gate

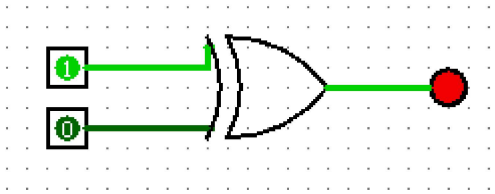
- **XNOR Gate**



A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Figure 9:XNOR Gate

- **XOR Gate**



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Figure 10: XOR Gate

3. Implement NOT gate using NAND Gate(s).

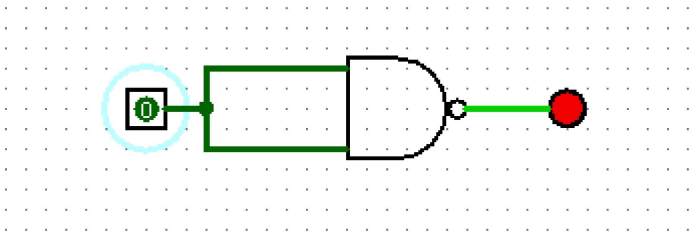


Figure 11: NOT Gate using NAND Gate

4. Implement AND gate using NAND Gate(s).

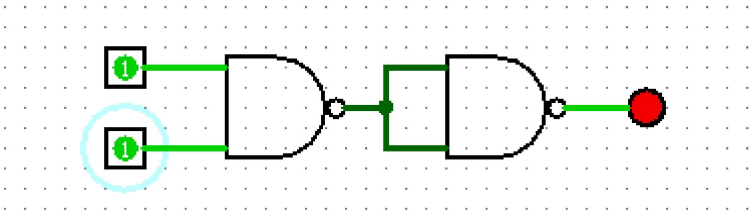


Figure 12: AND Gate with NAND Gate

5. Implement OR gate using NAND Gate(s).

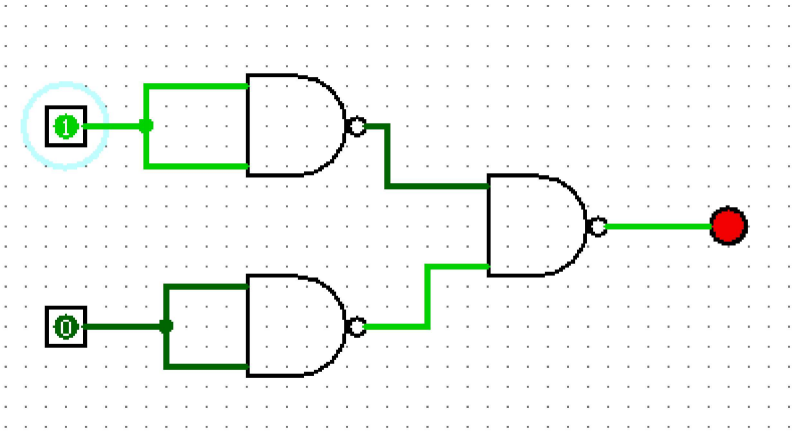


Figure 13:OR Gate with NAND Gate