Ministerul Educaţiei și Cercetării al Republicii Moldova Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică

COMPUTER ARCHITECTURE

Laboratory work 2:

Simulating digital logic circuits

Elaborated:
st. gr. FAF-213 Konjevic Alexandra

Verified:
asist. Univ Vladislav Voitcovschi

Chişinău – 2023

Objective:

Use a digital logic simulator to implement various circuits.

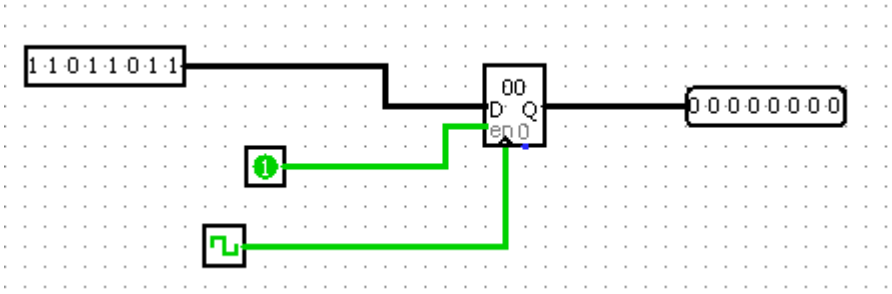
Introduction:

Logisim is a powerful digital logic simulator and design tool that allows users to create and simulate digital circuits in a graphical user interface. It is a free, open-source software that provides a user-friendly environment for designing and testing digital logic circuits. Logisim can be used to create complex circuits that incorporate logic gates, arithmetic circuits, sequential circuits, and other digital components. With Logisim, users can create, edit, and simulate circuits using a drag-and-drop interface. The software provides a library of digital components that can be used to build circuits, and users can also create their own custom components using the software's built-in tools.

Variant: 22

Tasks:

22. Implement a register with clock that memorizes an entry signal into an array of bits



The register implemented above in Logisim is a basic digital circuit that is used for storing a sequence of binary digits or bits. It consists of a group of flip-flops that are interconnected in such a way that they can store binary values and output them as a sequence.

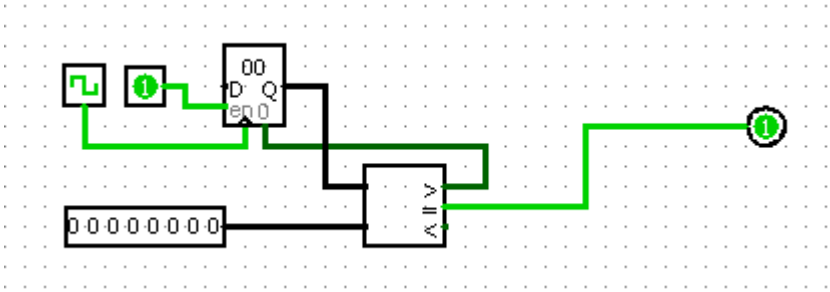
The register has three input pins: "D" for the data input, "CLK" for the clock input, and "CLR" for the clear input. The data input pin "D" is used to set the value of the bit sequence that will be stored in the register. The clock input pin "CLK" is used to synchronize the operation of the flip-flops and ensure that the data is stored in the register at the correct time. The clear input pin "CLR" is used to

reset the register to a known state, usually all zeros.

The register also has an output pin labeled "Q", which outputs the stored bit sequence. The bit sequence is updated on the rising edge of the clock signal, which means that the data is stored in the register only when the clock input transitions from low to high.

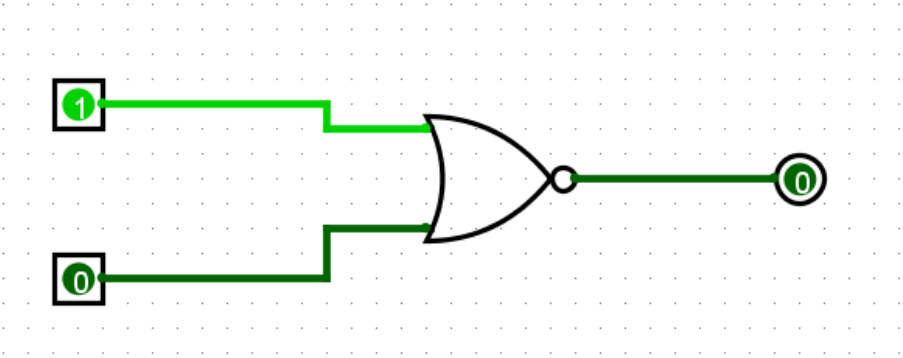
Overall, the register with a clock that we implemented in Logisim is a simple and useful digital circuit that can be used for various applications, such as data storage, data transfer, and data manipulation.

27. Build an oscillation circuit that generates an oscillation with a specified period.



This circuit builds a clock that outputs high on each tick of the input clock and accepts the clock's duration as an input. To determine when to output high and when to reset the tick counter, it utilizes a comparator.

5. Implement a NOR gate that will have at least one “1” input and the output “0”.

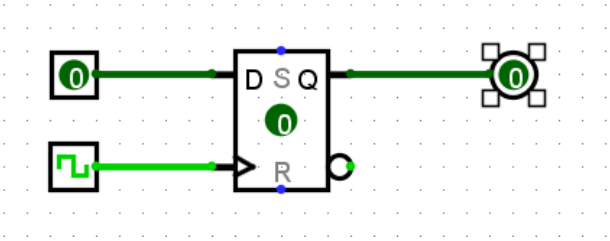


A NOR gate is a logic gate that produces a logical 0 output (or low output) only if all of its inputs are logical 1s (or high inputs). Otherwise, its output is a logical 1 (or high output). The symbol for a NOR gate is a circle with an inverted "OR" written inside, and it has two or more input terminals and one output terminal.

The output of a NOR gate is the logical negation of an OR gate. In other words, if all inputs of an OR gate are 0, the output is 0. If any input is 1, the output is 1. The NOR gate inverts this behavior, so if all inputs are 1, the output is 0. Otherwise, the output is 1.

NOR gates are commonly used in digital circuits for implementing logic functions such as Boolean algebra expressions and for building other logic gates like NAND, XOR and XNOR gates. They are also used for memory storage, timing control, and signal conditioning. NOR gates can be implemented using different technologies, such as diodes, transistors, or CMOS technology, depending on the specific application requirements.

10. Implement a d flip flop that will retain the input state when the clock is activated.



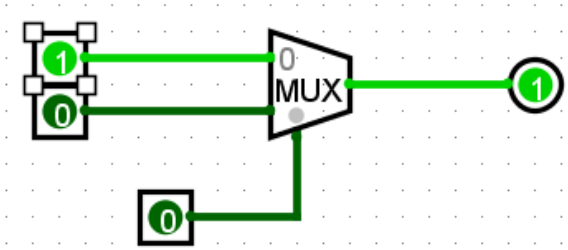
A D flip-flop, also known as a data flip-flop, is a type of flip-flop circuit that stores a single bit of data. It has two inputs, D (data) and CLK (clock), and two outputs, Q (output) and Q' (complement of output).

A D flip-flop that retains its input state when the clock is activated is called an edge-triggered D flip-flop. It changes its output state only on the rising or falling edge of the clock signal, depending on the type of flip-flop. When the clock is high, the D input is transferred to the output Q on the rising edge of the clock. When the clock is low, the output Q retains its previous state. This is sometimes called a "transparent latch" behavior, where the output follows the input when the clock is high, and holds its state when the clock is low.

A D flip-flop can be constructed using various logic gates, such as NAND gates or NOR gates, but the simplest implementation uses two cross-coupled NAND gates or two cross-coupled NOR gates.

D flip-flops are widely used in digital electronic circuits for storing and manipulating binary data. They are especially useful in sequential logic circuits, where the output depends not only on the current input but also on the previous state of the circuit.

15. Multiplexer with two entries and one selection bit



A multiplexer (MUX) is a digital circuit that selects one of several input signals and outputs it based on a selection input. It is a fundamental building block in digital circuit design and is commonly used in applications such as data selection, signal routing, and control circuitry.

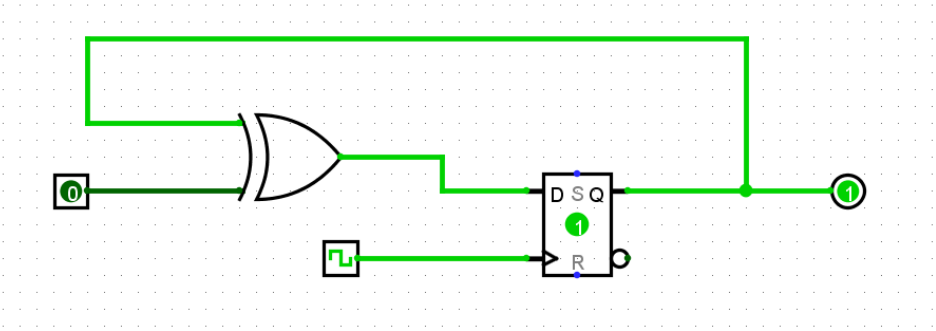
A simple two-input, one-selection-bit multiplexer can be described as follows: the multiplexer has two input signals, denoted as "Input 0" and "Input 1", and a selection input, denoted as "Select". The output of the multiplexer is denoted as "Output".

When the selection input is low (0), the multiplexer selects and outputs the first input signal (Input 0). When the selection input is high (1), the multiplexer selects and outputs the second input signal (Input 1).

The operation of a multiplexer can be thought of as a switch that connects one of several input signals to the output based on the state of the selection input. The selection input determines which input signal is selected, and the output of the multiplexer is the selected input signal.

Multiplexers can have more than two inputs and more than one selection bit, depending on the specific application. The design and implementation of a multiplexer can vary depending on the desired functionality and the available resources. However, the basic principle of selecting one input signal from several based on a selection input remains the same.

21. Implement a t flip flop



A T flip-flop is a type of flip-flop circuit that toggles its output (i.e., switches its state) based on the value of its "toggle" input T. One way to implement a T flip-flop using a D flip-flop and an XOR gate is as follows:

- Connect the D input of the D flip-flop to the output Q of the flip-flop.
- Connect the XOR gate inputs to the T input and the output Q of the flip-flop.
- Connect the output of the XOR gate to the D input of the flip-flop.
- Connect the clock signal to the clock input of the flip-flop.

In this implementation, when the T input is high, the XOR gate will invert the value of the output Q and feed it back to the D input of the flip-flop. This means that when the clock signal transitions from low to high, the flip-flop will latch the inverted value of Q, effectively toggling the output.

On the other hand, when the T input is low, the XOR gate will output the same value as the current output Q, which means that the output of the flip-flop will remain unchanged when the clock signal transitions.

Overall, this implementation allows you to create a T flip-flop using a D flip-flop and an XOR gate, which can be a useful building block in digital circuit design.

Conclusion:

In conclusion, the laboratory work was a valuable opportunity to gain hands-on experience with digital logic circuits and the Logisim simulation software. Throughout the course of the project, I successfully implemented a range of different constructions including a clock, oscillation circuit, NOR gate, D flip flop, multiplexer, and T flip flop.

One of the most significant benefits of the laboratory work was the ability to experiment with different circuit designs and configurations. This allowed me to see firsthand how small changes to the circuit can have a significant impact on its behavior. Through trial and error, I was able to refine my designs and optimize their performance, which is a crucial skill in the field of digital logic.

Finally, the laboratory work allowed me to gain practical experience with Logisim, which is a widely used simulation software in the field of digital logic design. By becoming proficient with Logisim, I have developed a valuable skillset that will be beneficial in my future academic and professional pursuits.