REPORT ON

INNOVATIVE ASSIGNMENT:

FIBONACCI GENERATOR

2CS507

DIGITAL ELECTRONICS

Project Prepared by:

Hardattsinh Mangrola
23BCE088

Dhruva Kothari
23BTM019

School of Engineering
Department of Computer Science and Engineering
Institute of Technology
Nirma University
Ahmedabad 382481

Nov 2024

INDEX

Sr. No.	Topics	Page No.
1.	Introduction	3
2.	Component Selection	4
3.	Use of the components	5
4.	Working of the circuit	7
5.	Summary	9

1. INTRODUCTION

In this project, a Fibonacci Sequence Generator is implemented to generate and display Fibonacci numbers, a critical sequence in mathematical analysis and computational applications. The Fibonacci sequence is defined by a series of integers beginning with 0 and 1, where each subsequent term is the sum of the previous two terms. This sequence is not only fundamental in mathematics but is also widely applied in areas such as computer science, data structure algorithms, cryptography, and financial modeling, among others.

The system employs an efficient iterative approach to compute and display Fibonacci numbers, enabling fast computations and clear demonstration of sequence generation. Optimized for performance, the generator lets users select specific terms or generate a range, with results presented in a user-friendly digital format. This project highlights practical applications of iterative methods and the mathematical foundations of Fibonacci sequences.

2. COMPONENT SELECTION

In the hardware implementation – manual, as well as automated circuit of the Fibonacci Sequence Generator, the components selected with their output and input sizes are as follows:

COMPONENT	I/P SIZE	O/P SIZE
Registers (A, B, C)	1 input each (data line)	1 output each
Adder	2 inputs (A, B)	1 output (C)
Controlled Buffers	1 input (data line)	1 output
SR latch	2 inputs (set, rest)	1 output
Demultiplexer	1 input (data line), 1 selection line	Multiple outputs
Counter	1 input (clock)	1 output (select for DMX)
Clock	-	1 output
Inverter (NOT Gate)	1 input	1 output
NAND Gate	2 inputs	1 output
Reset Line	1 input	1 output
Clock Line	1 input	1 output
Probe	1 input	Display output in decimal
Falling Edge Detector	1 input	1 output (pulse signal)

3. USE OF THE COMPONENTS

- **Registers (A, B, and C)**:
  - Registers are used to store the values of the Fibonacci sequence temporarily. Each register (A, B, and C) is updated based on the values of the other registers in each cycle, allowing the sequence to progress.
  - Register A holds the older Fibonacci term, B holds the more recent term, and C holds the next term in the sequence.
- **Adder Circuit**:
  - The adder takes the values from registers A and B, adds them together, and outputs the result to register C, which stores the next term in the Fibonacci sequence.
- **Control Buffers**:
  - Control buffers are used to manage the input of different signals into a single register input without causing conflicts. For instance, Register B needs either the value 1 or the value from Register C as an input.
  - By activating the control buffers selectively, the circuit decides which input to pass through to the registers.
- **SR (Set-Reset) Latch**:

- The SR latch is a memory circuit that holds a binary state (0 or 1). It helps manage when specific control signals are active.
- In this setup, the SR latch controls the buffer selection, deciding whether to initialize B with a 1 or update it with C's value.

Clock Pulses and Edge Detection:

- Registers in this circuit do not continuously update; instead, they capture new values only when they receive a rising edge clock pulse (signal changing from 0 to 1). The clock pulses act as a timing mechanism for sequentially updating registers.
- A falling edge detector may be used to trigger certain operations precisely when a signal drops from 1 to 0, helping to manage transitions between states.

De-multiplexer and Counter for Automation:

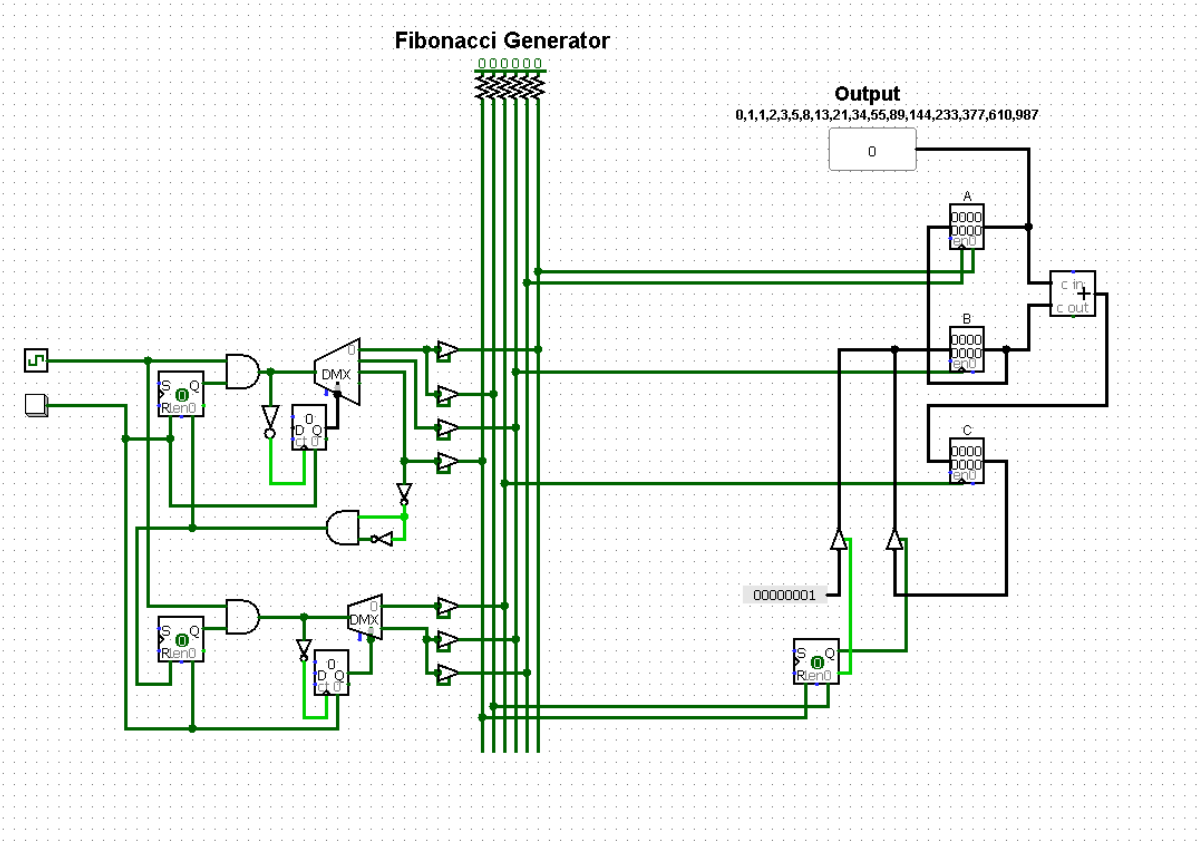
- A de-multiplexer with a counter is used to automate the control signals sequentially, simulating a loop that manually updates each register in the proper order. The counter increments with each clock pulse, allowing the circuit to move through the stages of updating A, B, and C without manual input.

4. WORKING OF THE CIRCUIT

The Fibonacci generator circuit starts in an initial configuration where registers A and B are set to 0 and 1, respectively, and C is updated based on A and B's sum.

1. **Initialization**:
   - Register A is set to 0 by pulsing its reset line, and Register B is set to 1 by pulsing the control buffer connected to it. The SR latch is initially in a state where the control buffer connecting C to B is inactive.
2. **Main Loop**:
   - The circuit then enters a loop where the following steps happen in sequence:
     - The adder calculates A+BA + BA+B and outputs this to register C.
     - A pulse updates register C to store this value, representing the next Fibonacci term.
     - Register A is updated to hold the previous value of register B, effectively shifting it one step back in the sequence.
     - Register B is updated to hold the value of C, setting it up to add with A in the next cycle.
     - The cycle repeats, generating Fibonacci numbers in each loop iteration.

3. **Manual Operation with Buttons**:
   - Manual operation is possible by connecting buttons to each control line, allowing the user to pulse each register individually.
   - By following the sequence (C, A, B), the user can manually simulate the Fibonacci sequence by pressing buttons in the correct order.
4. **Automated Control**:
   - Using a de-multiplexer, counter, and clock, the circuit can automatically generate the sequence. The counter advances with each clock pulse, controlling the de-multiplexer to sequentially activate the signals that update A, B, and C.
   - A clock gate controlled by an SR latch and NAND gates can start and stop the automatic sequence.



5. SUMMARY

This project outlines software implementation of a Fibonacci Sequence Generator using components like registers, an adder circuit, control buffers, SR latches, and automated control elements such as a demultiplexer and counter. By configuring registers A, B, and C to store and update Fibonacci terms and using an automated clock pulse mechanism, the circuit generates Fibonacci numbers in sequential cycles. The setup supports both manual and automated operations, allowing for flexible testing and control.

Future Scope: This project could be expanded to incorporate higher-order sequence generators or to optimize the circuit for reduced power consumption and space efficiency, which would be beneficial in embedded systems.

Learning: This project provided hands-on experience in digital circuit design and enhanced understanding of sequential logic, timing control, and hardware-based computation, laying a strong foundation for more complex sequence generation and automated digital systems.