Winter Fellowship Report

On

Integrated Circuit Design using Subcircuit feature of eSim

Submitted by

Varad V. Patil

Electrical Engineering Department

SGGSIE&T

Under the guidance of

Prof.Kannan M. Moudgalya

Chemical Engineering Department

IIT Bombay

February 22, 2025

Acknowledgment

We take this occasion to offer our heartfelt gratitude to the FOSSEE, IIT Bombay Team for offering us this wonderful opportunity to work on the design and integration of multiple sub-circuits in eSim. Working on eSim has prov ided us with invaluable insights into various open-source EDA tools for circuit simulation and their applications in the practical world.

We extend our sincere regards to Prof. Kannan M. Moudgalya for his valuable guidance and motivation to throughout this fellowship program.

We would like to express our heartfelt appreciation to the entire FOSSEE team including our mentors Mr. Sumanto Kar, Mrs. Vineeta Ghavri, and Mrs. Usha Vishwanathan for constantly guiding and mentoring us throughout the duration of our internship.

It is with their support that we have been able to fulfill our project demands successfully. Whenever faced with an issue, our mentors were always accessible to help us assess and debug them. Our learnings from them have been invaluable and shall be of paramount importance to us in the future.

Overall, it was a delightful experience interning at FOSSEE and contributing to its growth and I take away some great insights and knowledge from it. As enthusiastic beginners in the semiconductor industry, this internship is a milestone for us in our pursuit of a successful career.

Contents

Chapter 1

Introduction

FOSSEE which stands for Free/Libre and Open Source Software for Education is an organization, based at IIT Bombay, as a remarkable initiative aimed at promoting the use of open-source software in education and research. It was established with the mission to reduce the dependency on proprietary software and to encourage the adoption of open-source alternatives. FOSSEE offers a wide range of tools and resources that cater to various academic and professional needs.

It provides comprehensive documentation, tutorials, workshops, and hands-on training sessions, for empowering students, educators, and professionals to leverage open-source software for their projects and coursework. The organization's commitment to fostering a collaborative and inclusive environment has significantly contributed to the democratization of technology and has opened up new avenues for innovation and learning.

1.1 eSim

eSim, created by the FOSSEE project at IIT Bombay, is a versatile open-source software tool for circuit design and simulation. It combines various open-source software packages into one cohesive platform, making it easier to design, simulate, and analyze electronic circuits. This tool is particularly useful for students, educators, and professionals who need an affordable and accessible alternative to proprietary software.

eSim offers features for schematic creation, circuit simulation, PCB design, and includes an extensive library of components. The Subcircuit feature is a significant enhancement, enabling users to design complex circuits by integrating simpler subcircuits. Through eSim, FOSSEE promotes the use of open-source solutions in engineering education and professional fields, encouraging innovation and collaboration.

1.2 NgSpice

NgSpice is the open-source spice simulator for electric and electronic circuits. Such a circuit may comprise JFETs, bipolar and MOS transistors, passive elements like R, L, or C, diodes, transmission lines and other devices, all interconnected in a netlist.

Digital circuits are simulated as well, event-driven and fast, from single gates to complex circuits and the combination of both analog and digital as well as a mixed-signal circuits. NgSpice offers a wealth of device models for active, passive, analog, and digital elements. Model parameters are provided by our collections, by the semiconductor device manufacturers, or from semiconductor foundries. The user adds her circuits as a netlist, and the output is one or more graphs of currents, voltages and other electrical quantities or is saved in a data file.

1.3 Makerchip

Makerchip is a platform that offers convenient and accessible access to various tools for digital circuit design. It provides both browser-based and desktop-based environments for coding, compiling, simulating, and debugging Verilog designs. Makerchip supports a combination of open-source tools and proprietary ones, ensuring a comprehensive range of capabilities.

One can simulate Verilog/SystemVerilog/Transaction-Level Verilog code in Makerchip. eSim is interfaced with Makerchip using a Python based application called Makerchip-App which launches the Makerchip IDE. Makerchip aims to make circuit design easy and enjoyable for users of all skill levels. The platform provides a user-friendly interface, intuitive workflows, and a range of helpful features that simplify the design process and enhance the overall user experience.

The main drawback of these open source tools is that they are not comprehensive. Some of them are capable of PCB design (e.g. KiCad) while some of them are capable of performing simulations (e.g. gEDA). To the best of our knowledge, there is no open source software that can perform circuit design, simulation and layout design together. eSim is capable of doing all of the above.

Chapter 2

Features Of eSim

The objective behind the development of eSim is to provide an open source EDA solution for electronics and electrical engineers. The software should be capable of performing schematic creation, PCB design and circuit simulation (analog, digital and mixed-signal). It should provide facilities to create new models and components. Thus, eSim offers the following features -

1. Schematic Creation: eSim provides an easy-to-use graphical interface for drawing circuit schematics, making it accessible for users of all levels. Users can drag and drop components from the library onto the schematic, simplifying the design process. Comprehensive editing tools allow for easy modification of schematics, including moving, rotating, and labeling components.

2. Circuit Simulation: eSim supports SPICE (Simulation Program with Integrated Circuit Emphasis), a standard for simulating analog and digital circuits. Users can perform various types of analysis such as transient, AC, and DC, providing insights into circuit behavior over time and frequency.An integrated waveform viewer helps visualize simulation results, aiding in the analysis and debugging of circuit designs.

3. PCB Design: The PCB layout editor allows users to place components and route traces with precision. eSim includes DRC capabilities to ensure that the PCB design adheres to manufacturing constraints and electrical rules. Users can generate Gerber files, which are standard for PCB fabrication, directly from their designs.

4. Subcircuit Feature: This feature enables users to create complex circuits by integrating smaller, simpler subcircuits, promoting modular and hierarchical design approaches. Subcircuits can be reused in different projects, saving time and effort in redesigning common circuit elements.

5. Open Source Integration: eSim integrates several open-source tools like KiCad, Ngspice, and GHDL, providing a comprehensive suite for electronic design automation. Being open-source, eSim is free to use, making advanced circuit design tools accessible without the need for expensive licenses.

Chapter 3

Problem Statement

To design and develop various Analog and Digital Integrated Circuit Models in the form of sub-circuits using device model files already present in the eSim library. These IC models should be useful in the future for circuit designing purposes by developers and users, once they get successfully integrated into the eSim subcircuit Library.

3.1 Approach

Our approach to implementing the problem statement began with examining datasheets from prominent Integrated Circuit (IC) manufacturers such as Texas Instruments, Analog Devices, and NXP Semiconductors. we selected ICs that offer a diverse range of functionalities, including precision amplifiers, comparators, encoders, and audio amplifiers. After building the subcircuits, we tested them to verify basic circuit configurations using NgSpice simulations. The step-by-step roadmap of this process is outlined below :

1. Analyzing Datasheets : The primary step is to browse through various analog and digital IC datasheets, and hence find suitable circuits to implement in eSim, that are not previously included into the eSim library. Check for the detailed schematic of the IC's and once the component values and the truth table is ascertained, then finalise the IC to be created.

2. Subcircuit Creation : After deciding the IC, we start modeling it as a sub-circuit in eSim, using the model files present in the eSim device model library only. The design is strictly according to the information given in the official data-sheets of the ICs.This step also includes building the Symbol/Pin diagram of the IC according to the packaging and pin description given in the data-sheets only.

3. Test Circuit Design : Once the component of the IC is ready, now we can build the test circuits, according to the data-sheets. In this step we build the test cases and test circuits using the component IC.

4. Schematic Testing : Once the test circuits are ready, now it's time to simulate the test circuits so that the output can be obtained in the form of wave-forms and plots. Here we take help of KiCad to NgSpice conversion and Simulation feature in eSim

If the output of the test circuit is not as per expectation, this implies that the test case has failed, and there is some error in the schematic. In such cases we go back to the design phase of the IC or the test circuits, to look for possible errors and then repeat the testing process again after making required changes.

Once the expected output of the test cases are correct and satisfy the expected results, then in such a case the IC is declared successfully working. The test case has been verified and the designing process is complete.

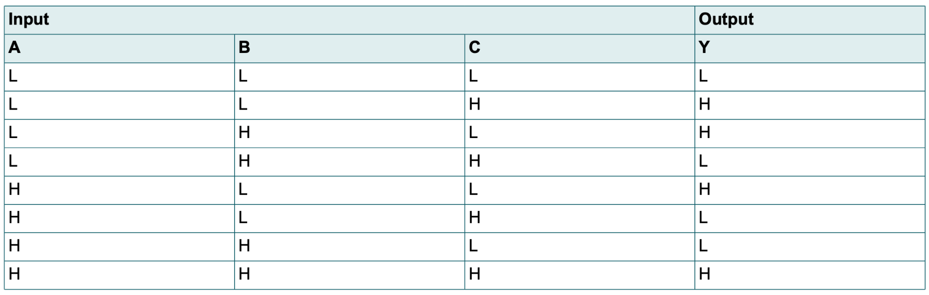
Chapter 4

Integrated Circuit Design

4.1 74LVC1G386

The 74LVC1G386 is used in digital logic applications where an exclusive-OR (XOR) function is needed with an additional control input. It is commonly used in arithmetic circuits, parity generators, and data comparison applications. Its low-voltage operation and high-speed characteristics make it ideal for modern logic systems requiring efficient and compact logic implementation.

Truth Table :



Input			Output
A	B	C	Y
L	L	L	L
L	L	H	H
L	H	L	H
L	H	H	L
H	L	L	H
H	L	H	L
H	H	L	L
H	H	H	H

Figure 4.1: Truth Table

4.1.1 Pin Diagram

The figure shows the physical representation of the 74LVC1G386 IC, indicating the arrangement of its pins. It includes input pins,

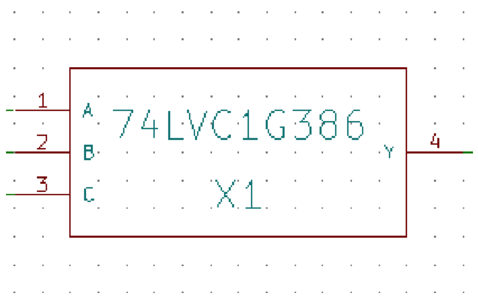


Figure 4.2: Pin Layout

an output pin, a power supply pin, and a ground pin. The pin diagram is essential for correctly integrating the IC into a circuit, ensuring proper connections and functionality in digital applications.

4.1.2 Sub Circuit Layout

The figure represents the internal design of the 74LVC1G386 IC, showing how logic gates and components are interconnected within the chip. This layout determines how the IC processes input signals to generate the required output. Understanding the sub-circuit helps in optimizing circuit performance and ensuring efficient logic operation.

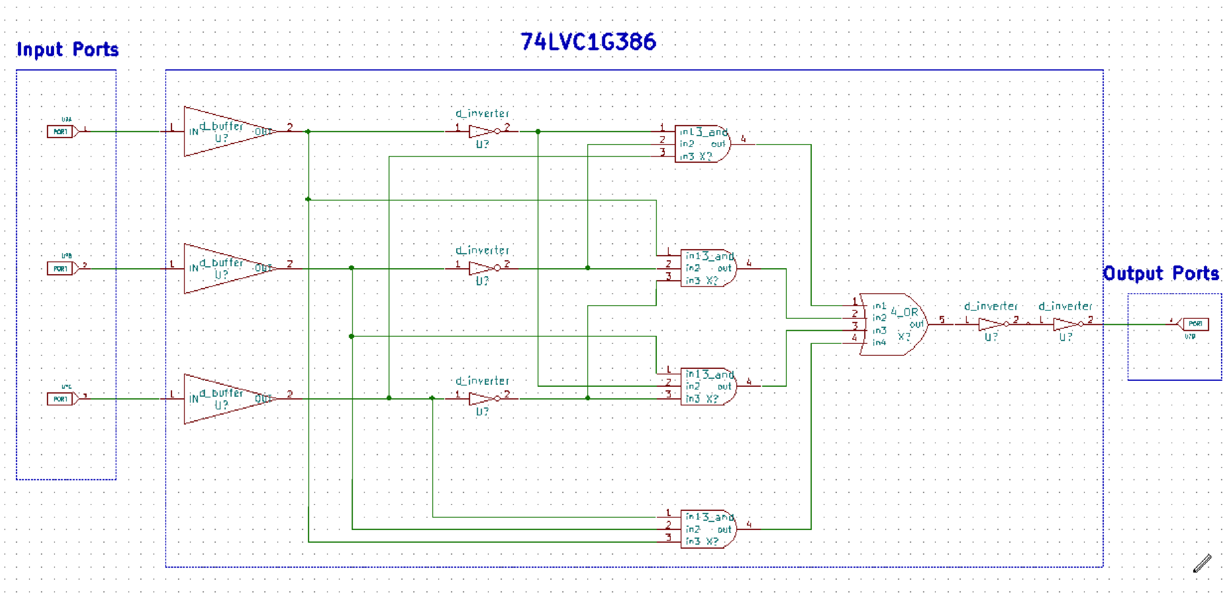


Figure 4.3: Sub Circuit Layout

4.1.3 Test Circuit

The figure illustrates a test setup used to verify the performance of the 74LVC1G386 IC. A test circuit includes a power source, input signal generators, and output monitoring devices such as oscilloscopes or logic analyzers. This setup helps evaluate response time, signal integrity, and overall functionality under different conditions.

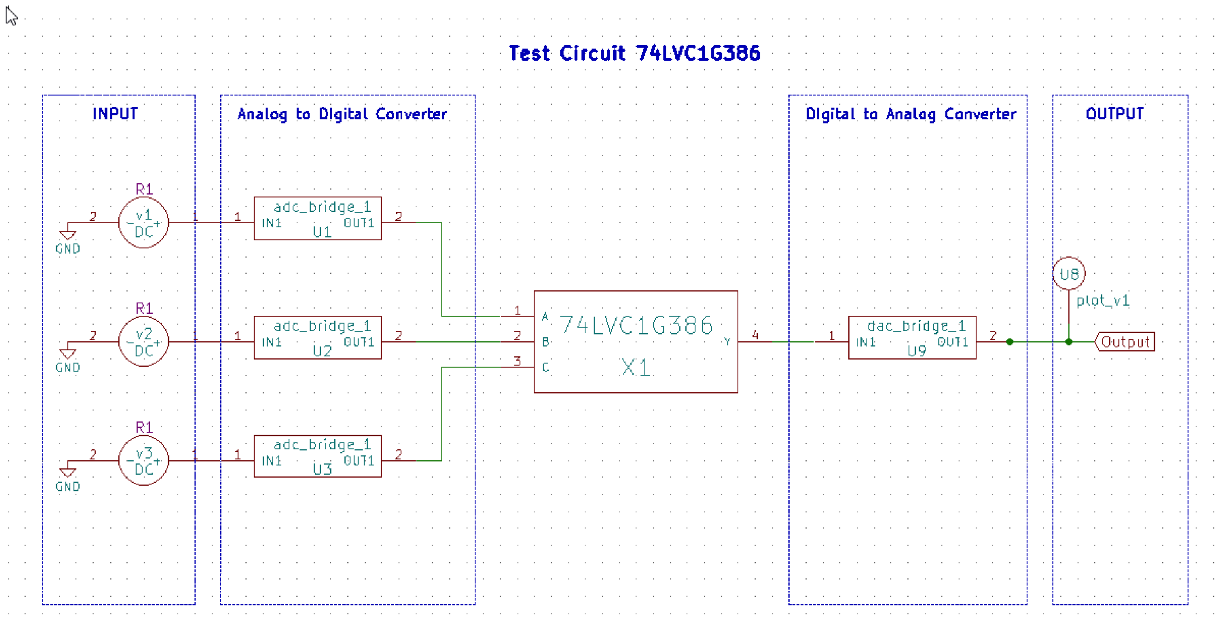
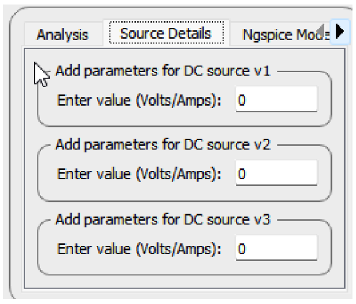


Figure 4.4: Test Circuit Layout

4.1.4 Output Waveform

The figure shows the signal produced at the output pin of the 74LVC1G386 IC after processing the input signals. This waveform represents the XOR logic operation performed by the IC..



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.5: Input

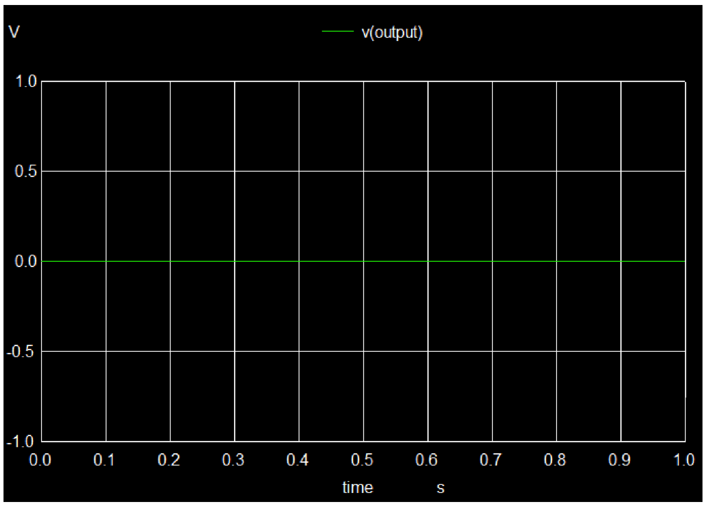
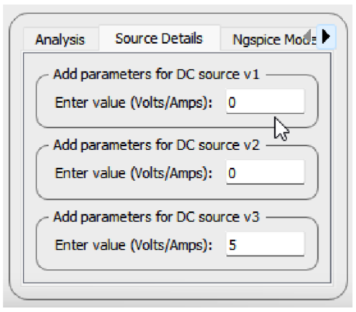


Figure 4.6: Output



Add parameters for DC source v1

Enter value (Volts/Amps): 0

Add parameters for DC source v2

Enter value (Volts/Amps): 0

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.7: Input

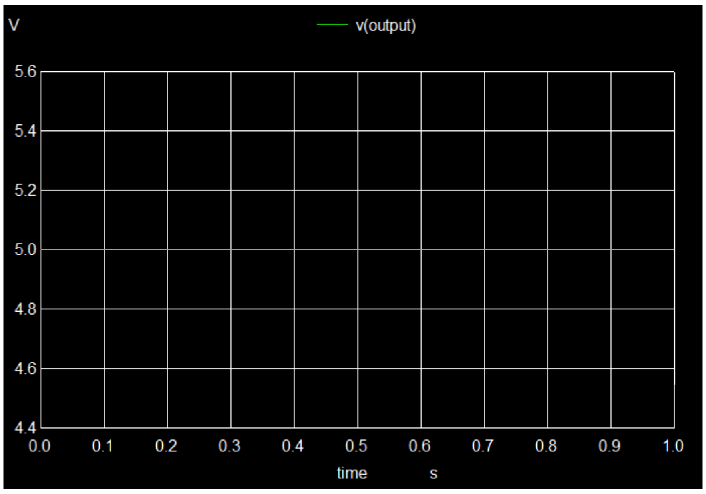
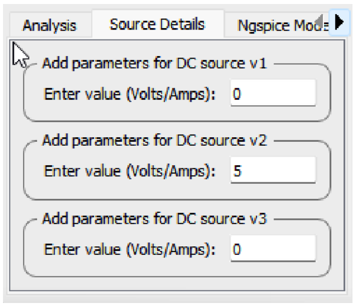


Figure 4.8: Output



Add parameters for DC source v1

Enter value (Volts/Amps): 0

Add parameters for DC source v2

Enter value (Volts/Amps): 5

Add parameters for DC source v3

Enter value (Volts/Amps): 0

Figure 4.9: Input

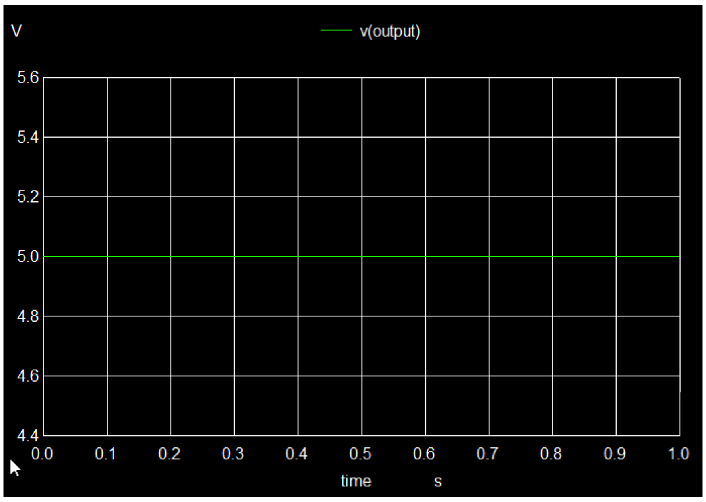
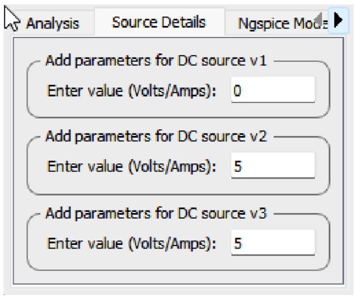


Figure 4.10: Output



Add parameters for DC source v1

Enter value (Volts/Amps): 0

Add parameters for DC source v2

Enter value (Volts/Amps): 5

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.11: Input

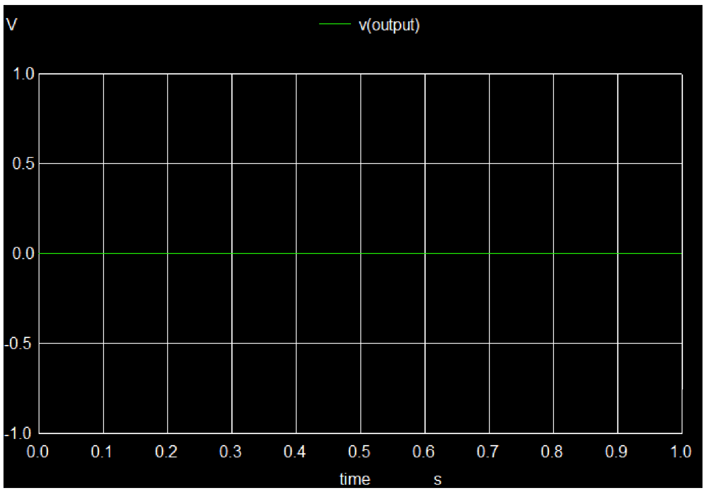
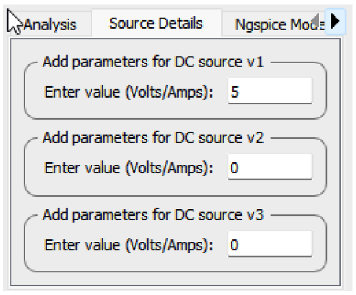


Figure 4.12: Output



Analysis | Source Details | Ngspice Mode

Add parameters for DC source v1
Enter value (Volts/Amps): 5

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.13: Input

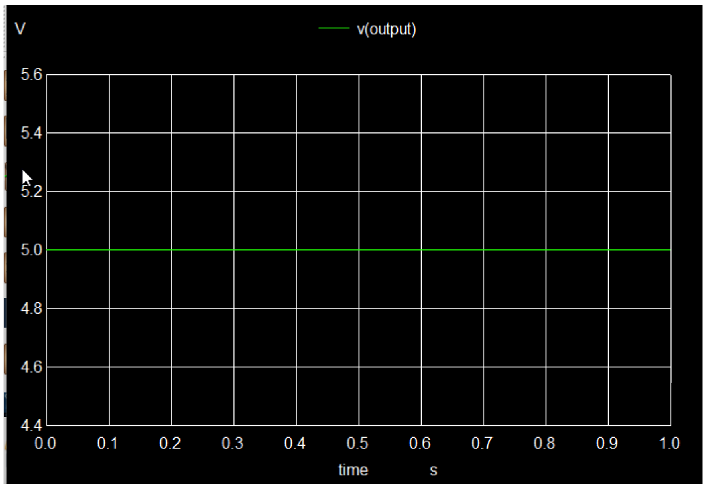
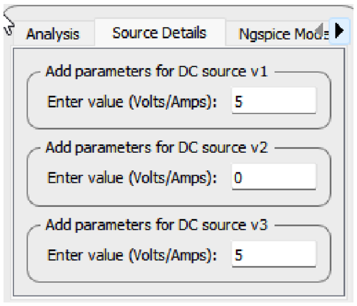


Figure 4.14: Output



Analysis | Source Details | Ngspice Mode

Add parameters for DC source v1
Enter value (Volts/Amps): 5

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.15: Input

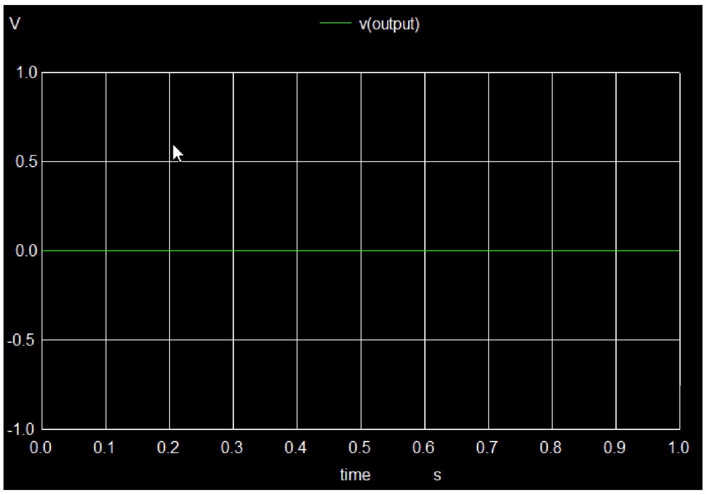
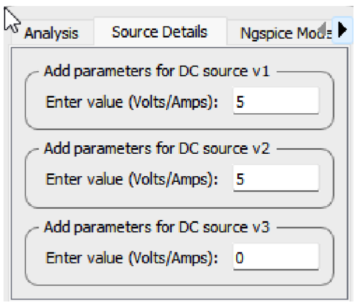


Figure 4.16: Output



Analysis | Source Details | Ngspice Mode

Add parameters for DC source v1
Enter value (Volts/Amps): 5

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.17: Input

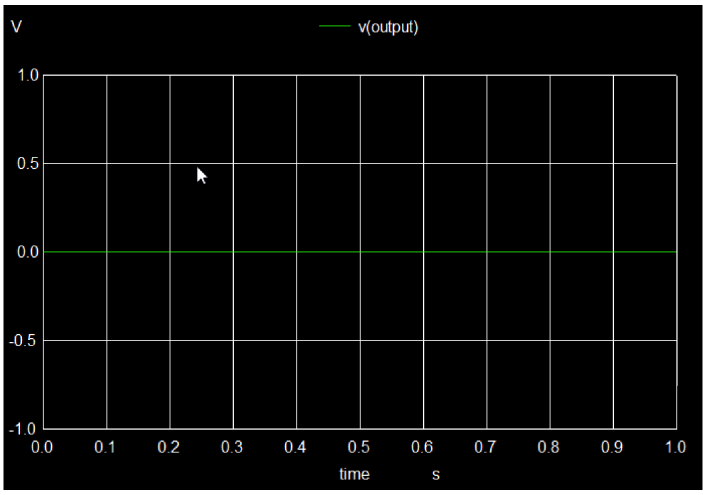
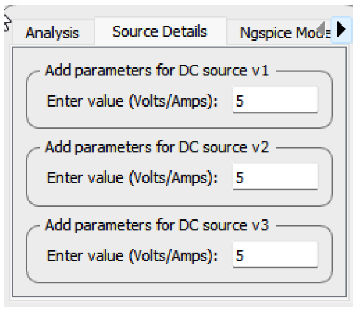


Figure 4.18: Output



Analysis | Source Details | Ngspice Model

Add parameters for DC source v1
Enter value (Volts/Amps): 5

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.19: Input

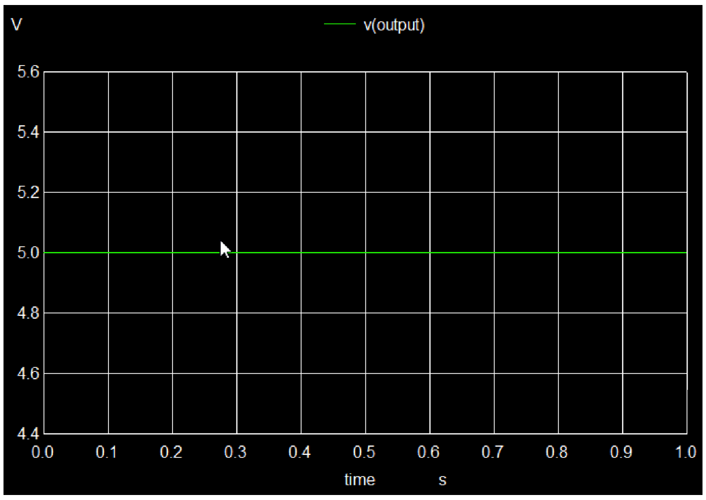


Figure 4.20: Output

4.2 SN74LVC1G57

The SN74LVC1G57 is designed for applications that require configurable logic functions within a single device. It is useful in circuits where multiple Boolean functions need to be implemented with minimal components, such as in control systems, data processing, and signal manipulation. Its flexibility makes it a preferred choice in embedded systems and logic optimization tasks.

Truth Table :

INPUTS			**OUTPUT**
In2	**In1**	**In0**	**Y**
L	L	L	H
L	L	H	L
L	H	L	H
L	H	H	L
H	L	L	L
H	L	H	L
H	H	L	H
H	H	H	H

Figure 4.21: Truth Table

4.2.1 Pin Diagram

The figure shows the physical representation of the SN74LVC1G57 IC, indicating the arrangement of its pins. This IC is a configurable logic gate that allows the selection of different logic functions using control inputs.

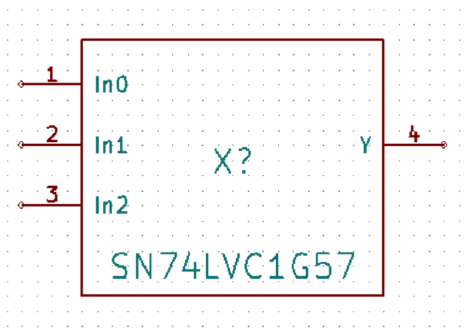


Figure 4.22: Pin Layout

It consists of input pins for logic operations, an output pin for the processed signal, a power supply pin, and a ground pin. Proper pin configuration is crucial for ensuring accurate functionality in digital applications.

4.2.2 Sub Circuit Layout

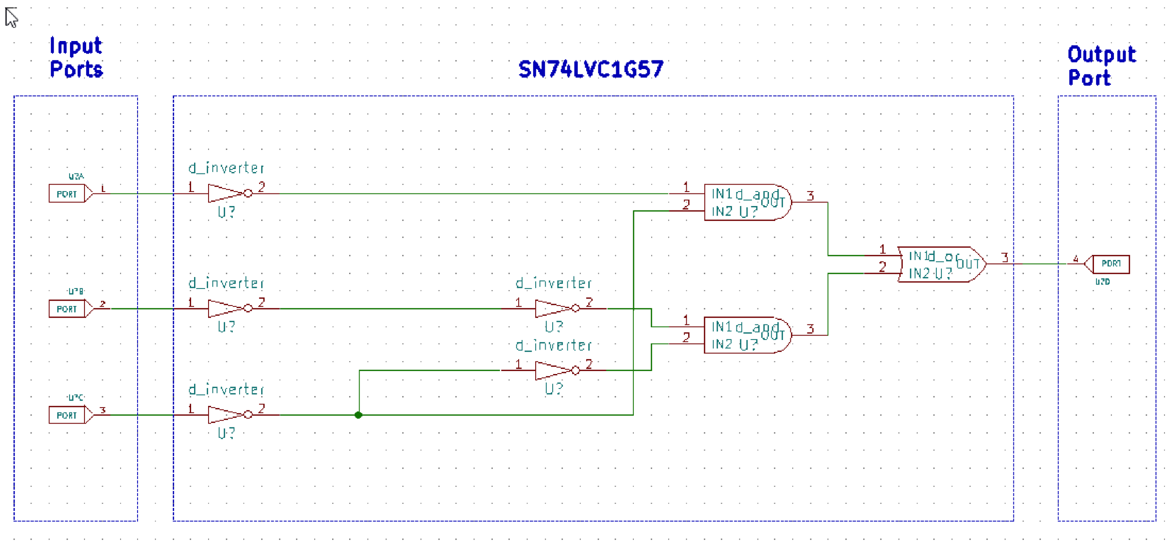


Figure 4.23: Sub Circuit Layout

The figure represents the internal structure of the SN74LVC1G57 IC, showing how its logic components are interconnected. This IC features configurable logic, allowing users to implement various Boolean functions based on input selection. The internal design consists of gates and control mechanisms that determine the final logic operation. Understanding the sub-circuit layout helps in optimizing circuit performance and utilizing the IC effectively.

4.2.3 Test Circuit

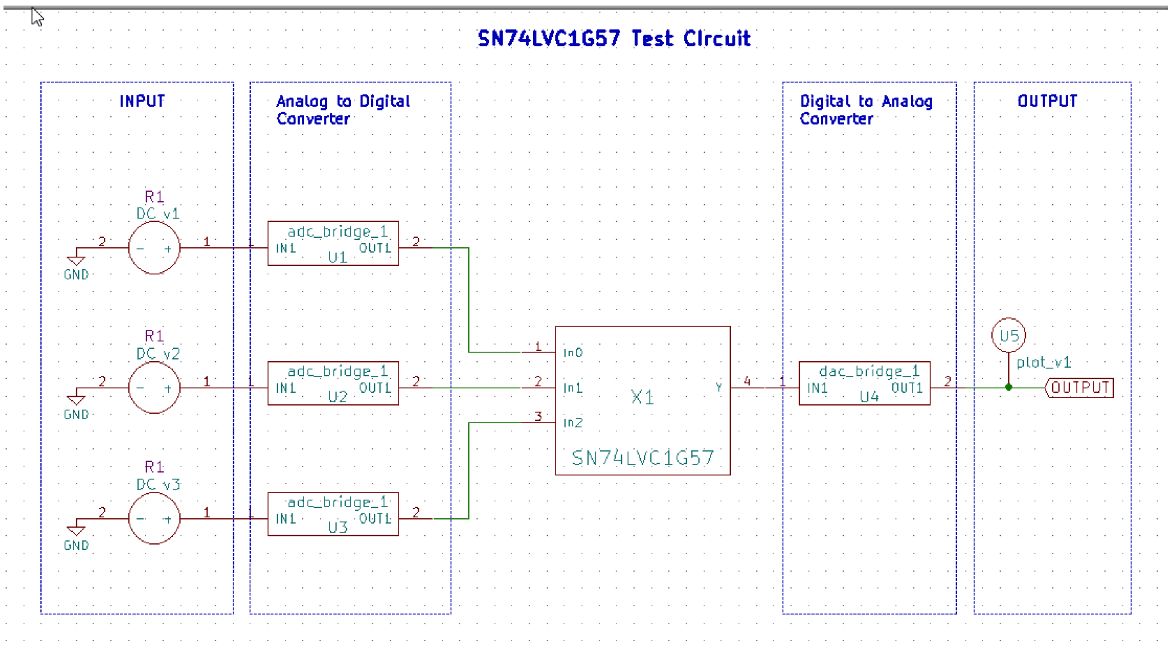
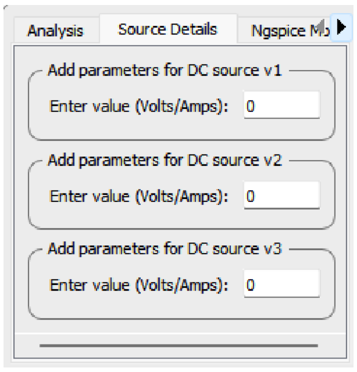


Figure 4.24: Test Circuit Layout

The figure illustrates a test setup designed to verify the operation of the SN74LVC1G57 IC. This test circuit typically includes a power supply, input signal sources, and measuring instruments such as oscilloscopes or logic analyzers. Since this IC provides configurable logic, testing different input combinations helps ensure that it performs the desired logic function correctly under varying conditions.

4.2.4 Output Waveform

The figure represents the signal generated at the output pin of the SN74LVC1G57 IC after processing the input signals. The output waveform reflects the logic function executed by the IC, ensuring the expected response based on input conditions.


Analysis | Source Details | Ngspice M

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.25: Input

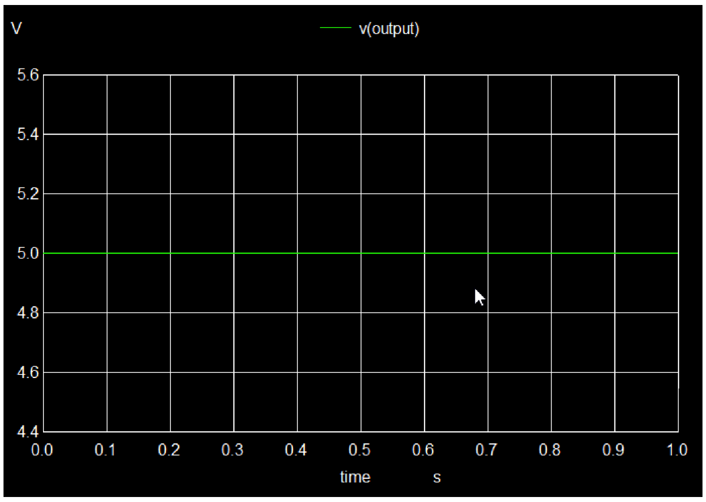
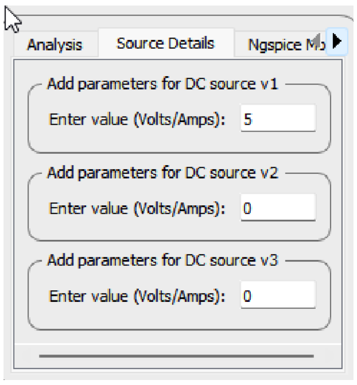


Figure 4.26: Output


Analysis | Source Details | Ngspice M

Add parameters for DC source v1
Enter value (Volts/Amps): 5

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.27: Input

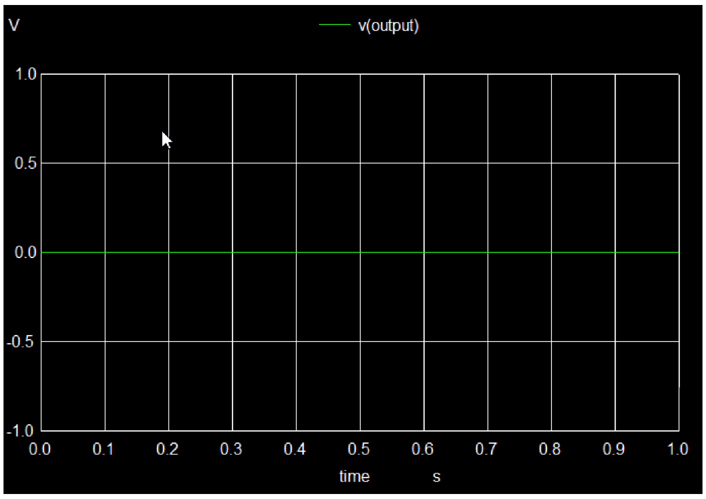
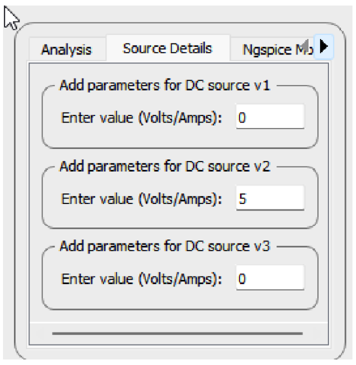


Figure 4.28: Output


Analysis | Source Details | Ngspice M

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.29: Input

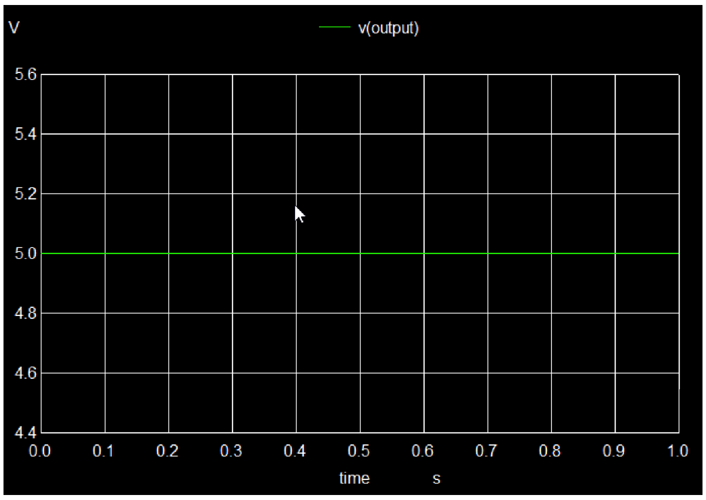
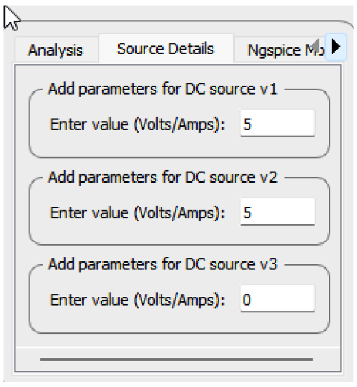


Figure 4.30: Output



Add parameters for DC source v1

Enter value (Volts/Amps): 5

Add parameters for DC source v2

Enter value (Volts/Amps): 5

Add parameters for DC source v3

Enter value (Volts/Amps): 0

Figure 4.31: Input

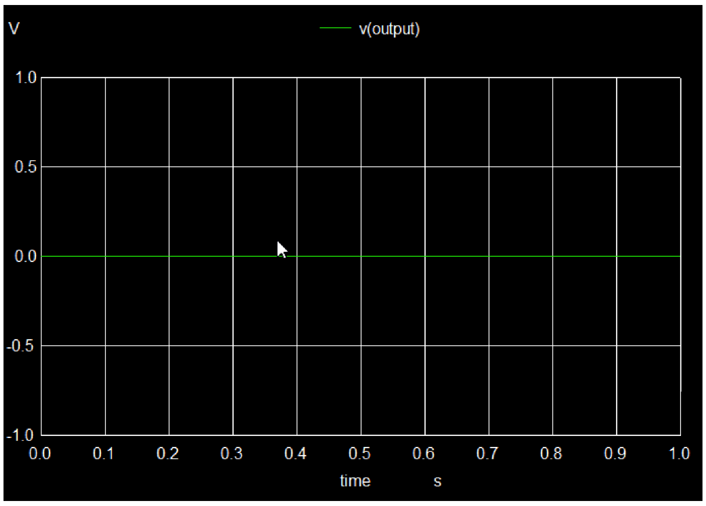
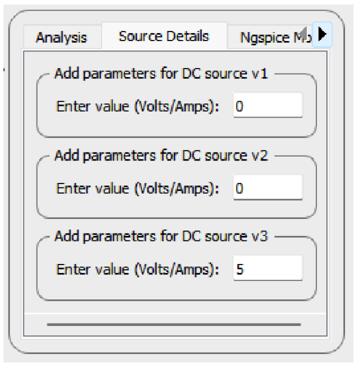


Figure 4.32: Output



Add parameters for DC source v1

Enter value (Volts/Amps): 0

Add parameters for DC source v2

Enter value (Volts/Amps): 0

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.33: Input

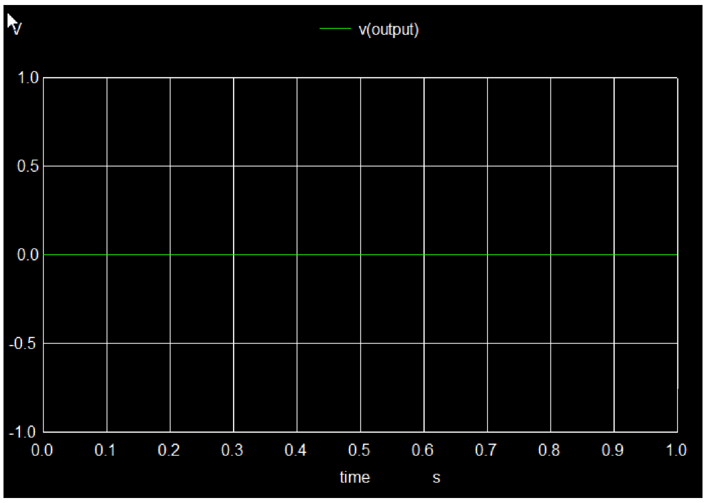
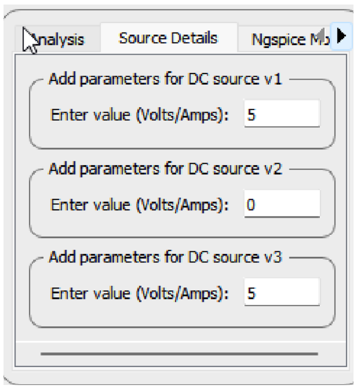


Figure 4.34: Output



Add parameters for DC source v1

Enter value (Volts/Amps): 5

Add parameters for DC source v2

Enter value (Volts/Amps): 0

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.35: Input

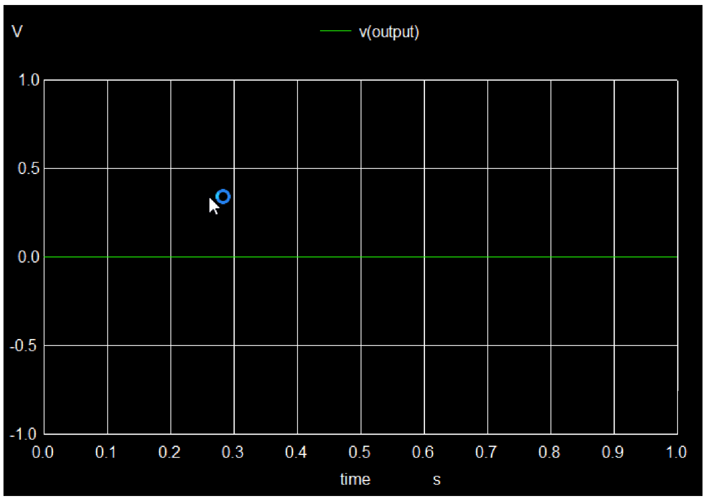
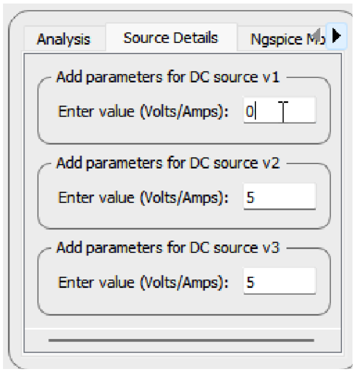


Figure 4.36: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.37: Input

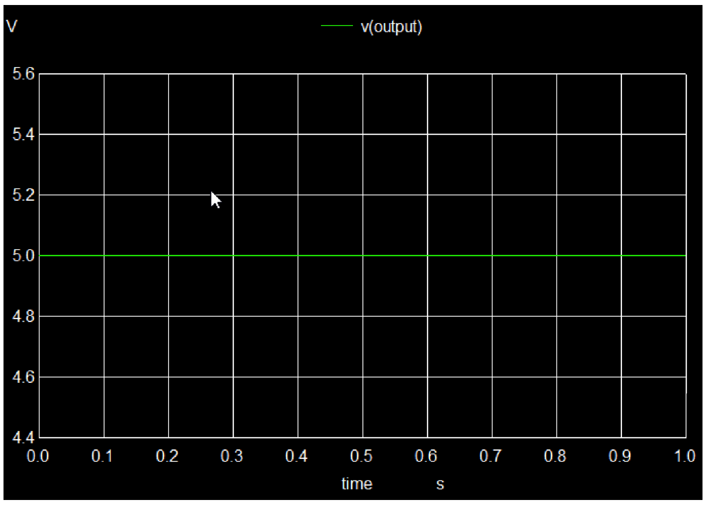
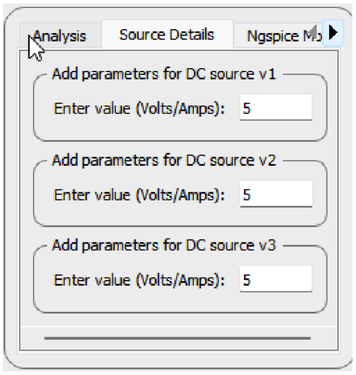


Figure 4.38: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 5
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.39: Input

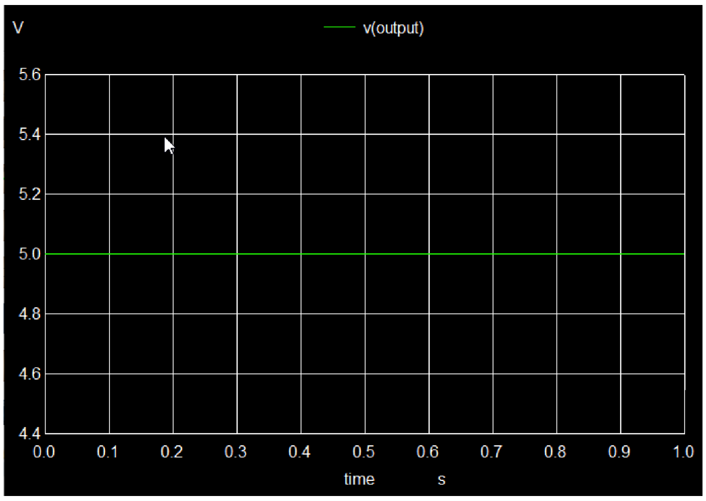
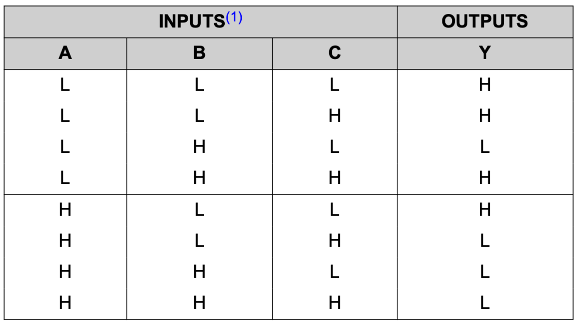


Figure 4.40: Output

4.3 SN74LVC3G98

The SN74LVC3G98 is used in digital circuits where multiple independent logic functions are required. It is particularly useful in applications such as address decoding, data multiplexing, and custom logic implementation. Its ability to configure different logic gates in a single IC reduces circuit complexity and improves design efficiency.

Truth Table :



INPUTS[1]			OUTPUTS
A	B	C	Y
L	L	L	H
L	L	H	H
L	H	L	L
L	H	H	H
H	L	L	H
H	L	H	L
H	H	L	L
H	H	H	L

Figure 4.41: Truth Table

4.3.1 Pin Diagram

The figure shows the physical representation of the SN74LVC3G98 IC, indicating the arrangement of its pins. This IC is a triple 3-input configurable logic gate that can perform multiple logic functions based on its inputs.

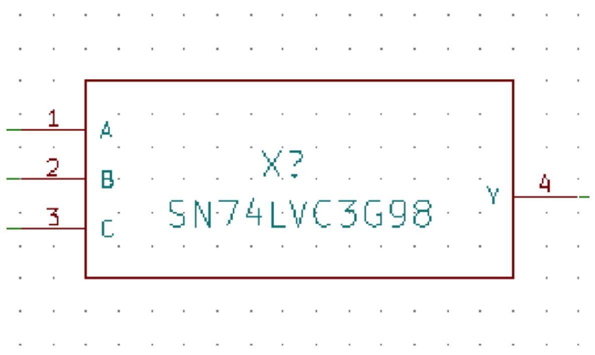


Figure 4.42: Pin Layout

It consists of multiple input pins, an output pin for each logic gate, a power supply pin, and a ground pin. Proper pin configuration is essential for ensuring correct logic operations and efficient circuit integration.

4.3.2 Sub Circuit Layout

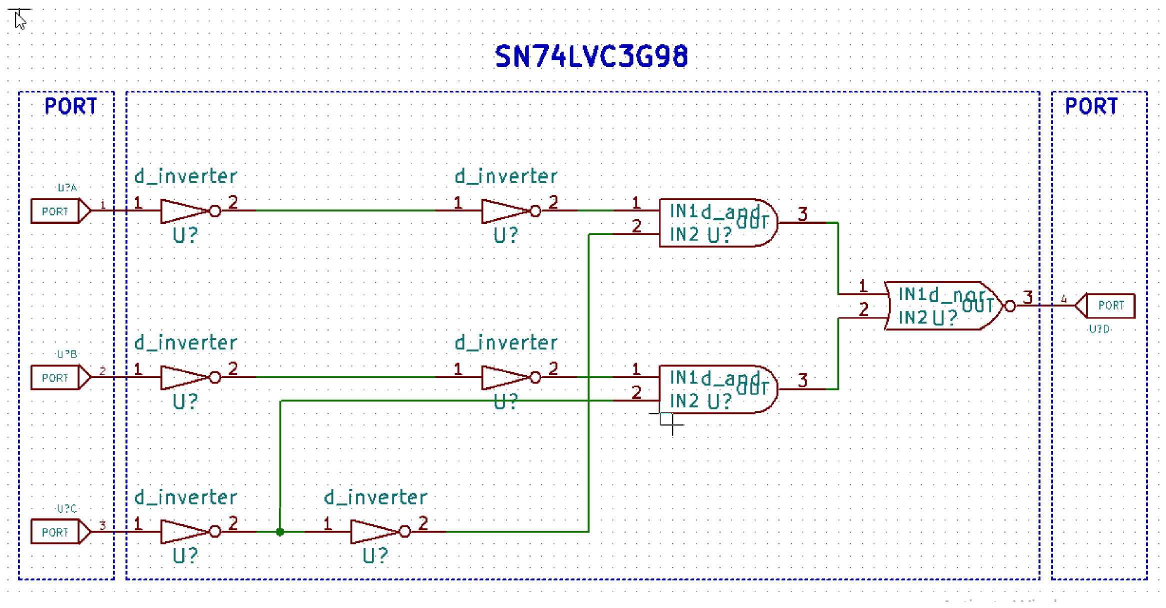


Figure 4.43: Sub Circuit Layout

The figure represents the internal structure of the SN74LVC3G98 IC, showing how its logic gates and internal components are connected. This IC integrates three independent configurable logic gates, allowing it to perform various Boolean functions. The internal design consists of multiple logic elements, enabling flexible circuit design and optimization. Understanding the sub-circuit layout helps in utilizing the IC effectively for digital applications.

4.3.3 Test Circuit

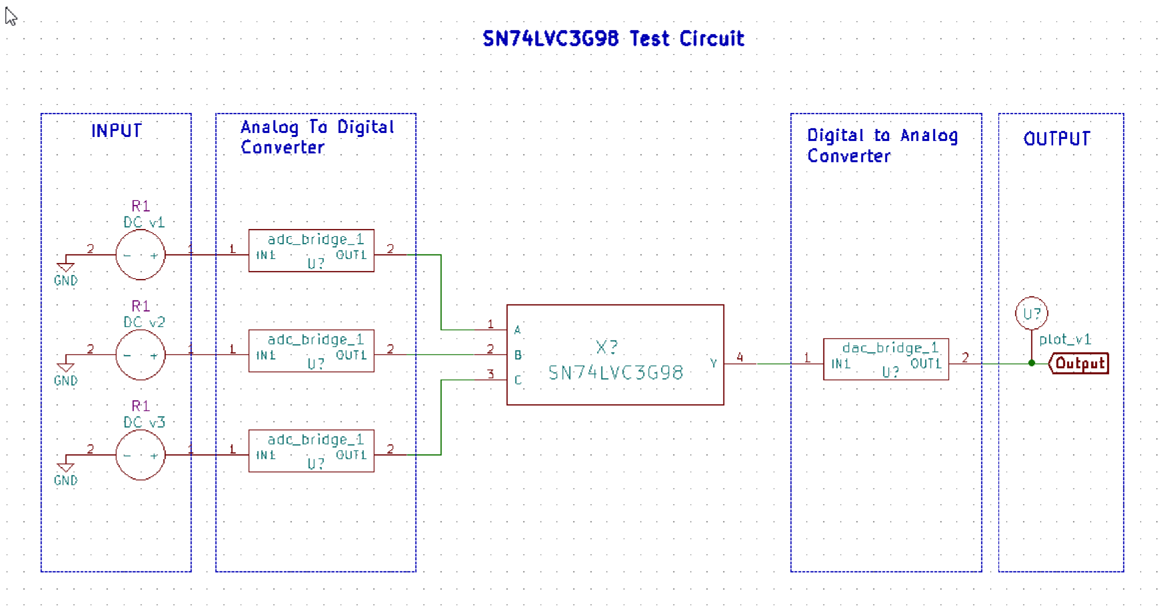
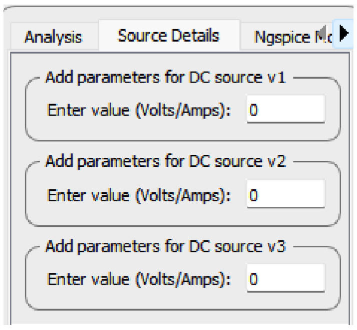


Figure 4.44: Test Circuit Layout

The figure illustrates a test setup designed to verify the operation of the SN74LVC3G98 IC. The test circuit includes a power source, input signal generators, and measurement devices such as oscilloscopes or logic analyzers. Since this IC has multiple configurable logic gates, testing different input combinations ensures that all logic functions operate correctly under varying conditions.

4.3.4 Output Waveform

The figure represents the signals generated at the output pins of the SN74LVC3G98 IC after processing the input signals. Each output waveform corresponds to the logic function performed by the respective gate within the IC.



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.45: Input

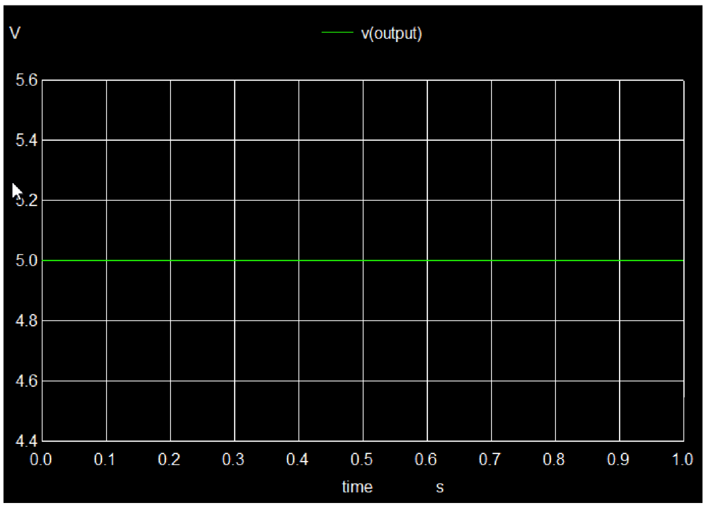
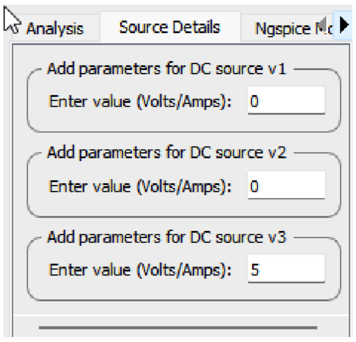


Figure 4.46: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.47: Input

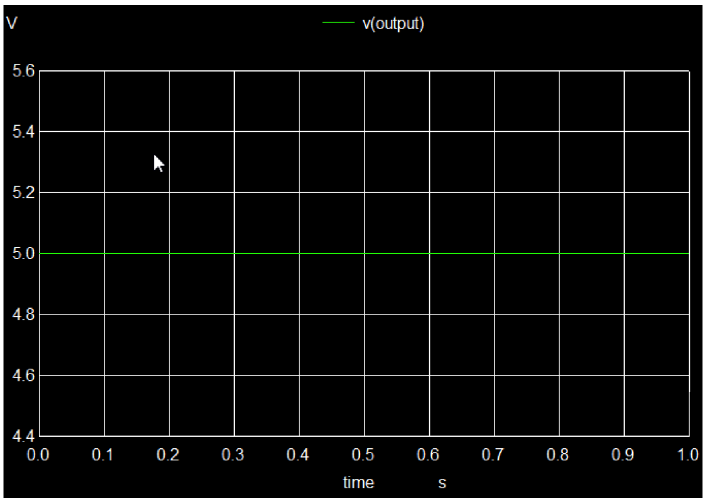
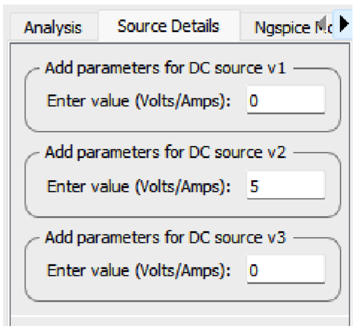


Figure 4.48: Output


Analysis | Source Details | Ngspice Mc

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.49: Input

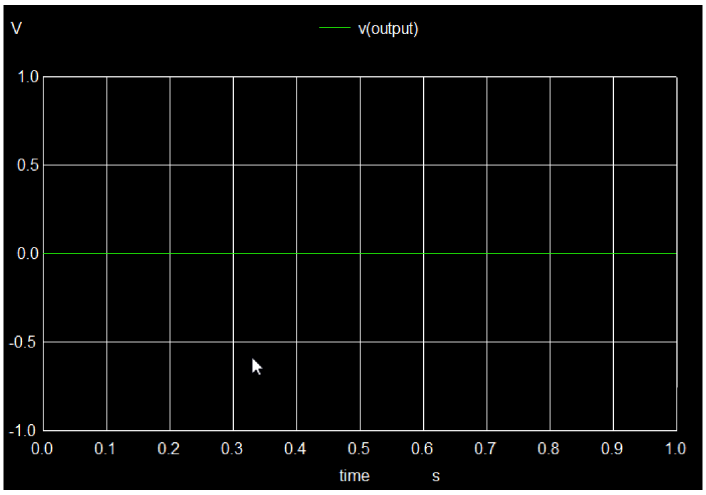
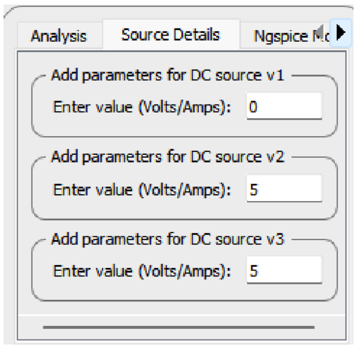


Figure 4.50: Output


Analysis | Source Details | Ngspice Mc

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.51: Input

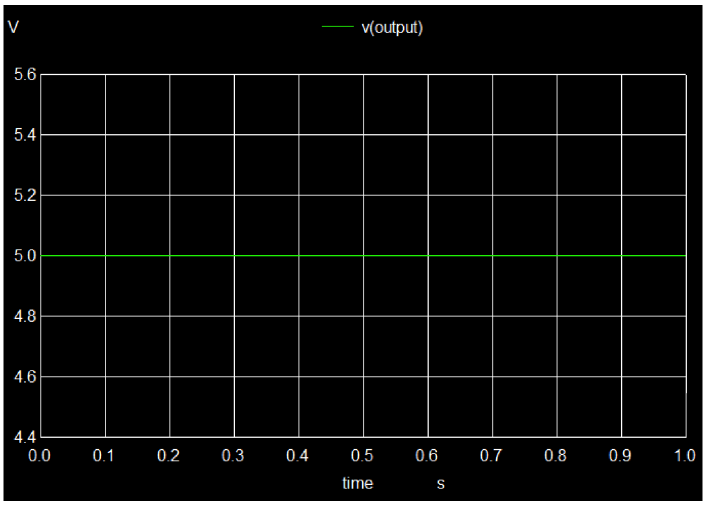
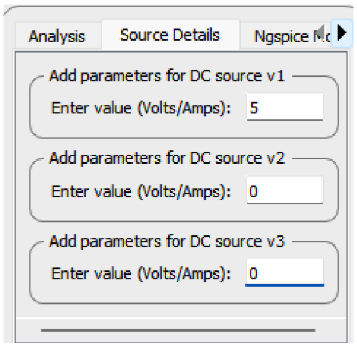


Figure 4.52: Output


Analysis | Source Details | Ngspice Mc

Add parameters for DC source v1
Enter value (Volts/Amps): 5

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.53: Input

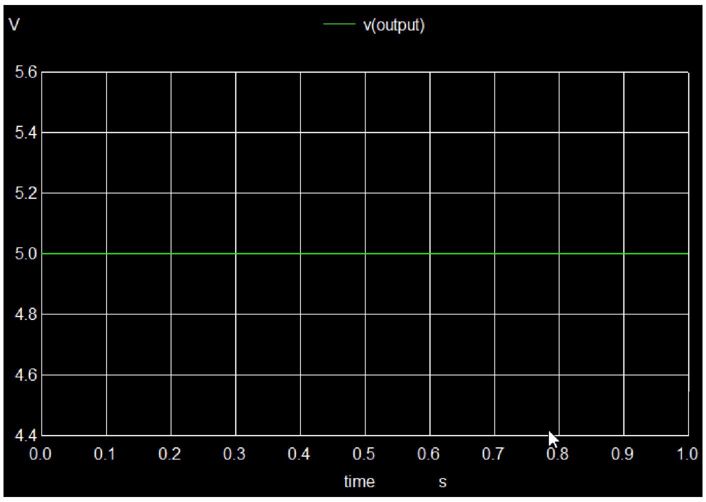
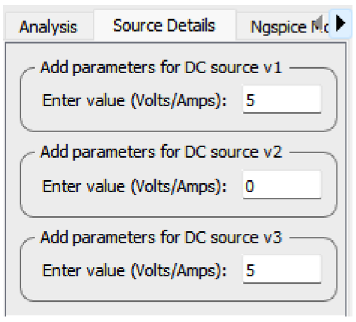


Figure 4.54: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 5
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.55: Input

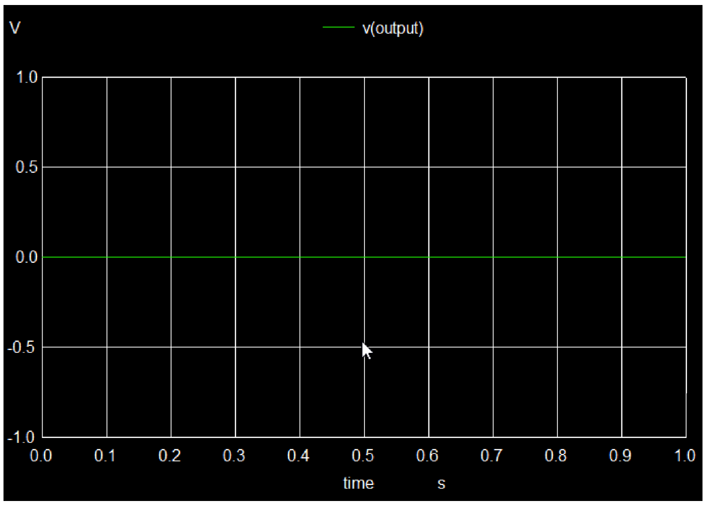
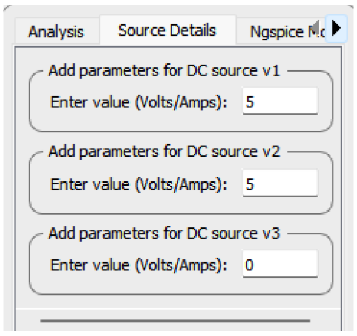


Figure 4.56: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 5
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.57: Input

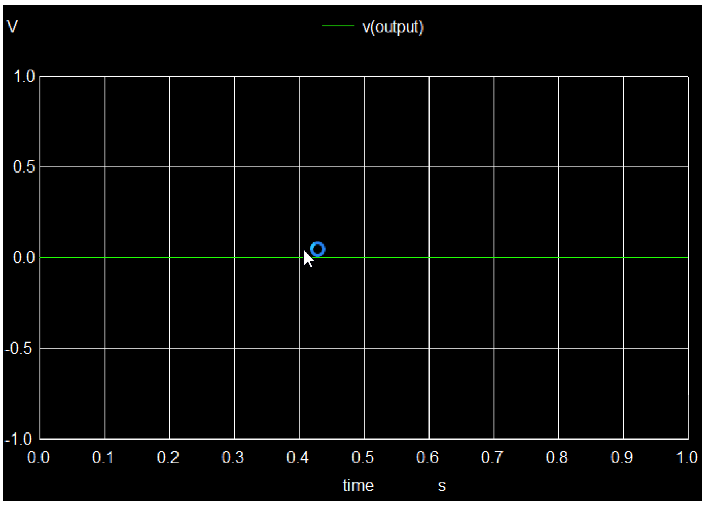
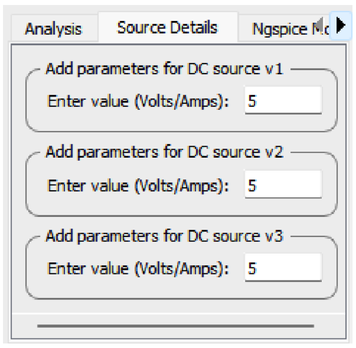


Figure 4.58: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 5
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 5

Figure 4.59: Input

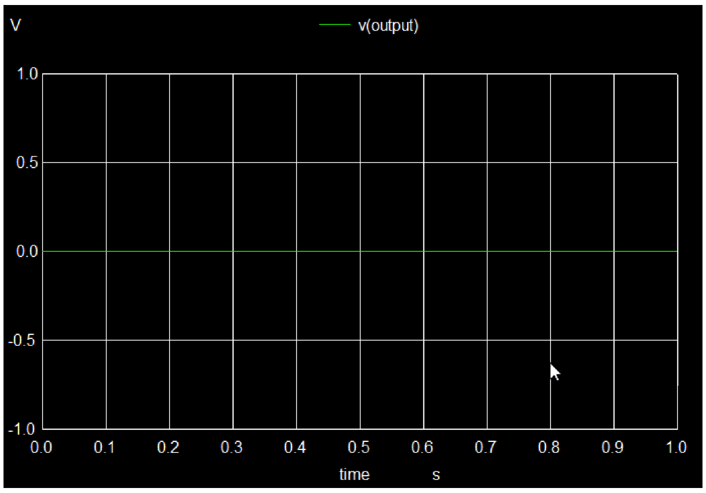
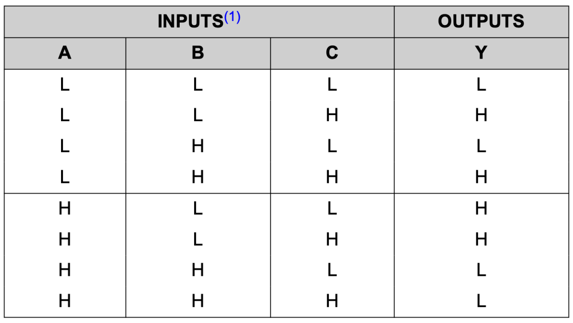


Figure 4.60: Output

4.4 SN74LVC3G58

The SN74LVC3G58 is ideal for applications that require a combination of multiplexing and logic operations. It is commonly used in signal selection, data routing, and conditional logic circuits. The ability to configure logic operations within a single IC allows designers to create efficient and space-saving digital systems.

Truth Table :



INPUTS[1]			OUTPUTS
A	B	C	Y
L	L	L	L
L	L	H	H
L	H	L	L
L	H	H	H
H	L	L	H
H	L	H	H
H	H	L	L
H	H	H	L

Figure 4.61: Truth Table

4.4.1 Pin Diagram

The figure shows the physical representation of the SN74LVC3G58 IC, indicating the arrangement of its pins. This IC is a triple 3-input configurable multiplexer logic gate, meaning it can perform various logic operations based on input selection.

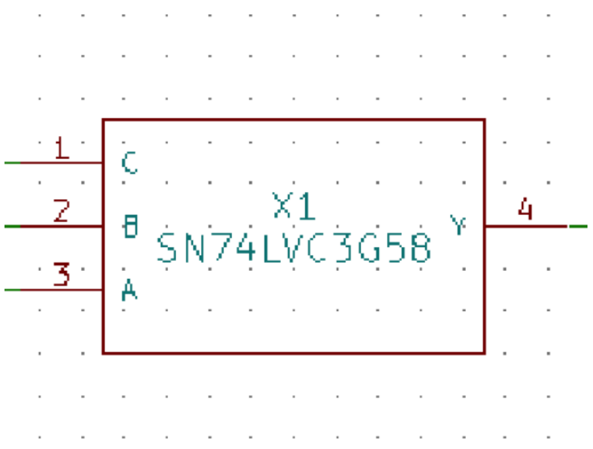


Figure 4.62: Pin Layout

It consists of multiple input pins, an output pin for each logic function, a power supply pin, and a ground pin. Correct pin connections are essential to ensure the desired logic operation and reliable circuit performance.

4.4.2 Sub Circuit Layout

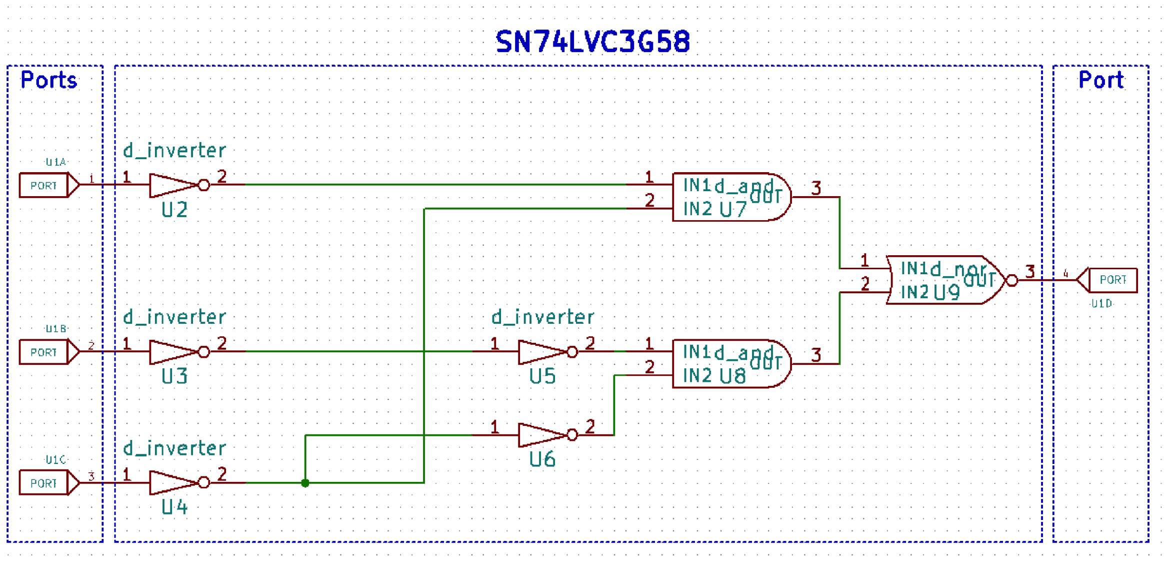


Figure 4.63: Sub Circuit Layout

The figure represents the internal structure of the SN74LVC3G58 IC, showing how its logic components are interconnected. This IC integrates three independent 3-input logic functions that can be configured as multiplexers or other logic gates. The internal design includes logic elements that allow flexible operation, enabling users to customize the logic behavior based on input configurations. Understanding the sub-circuit layout helps in designing optimized digital circuits.

4.4.3 Test Circuit

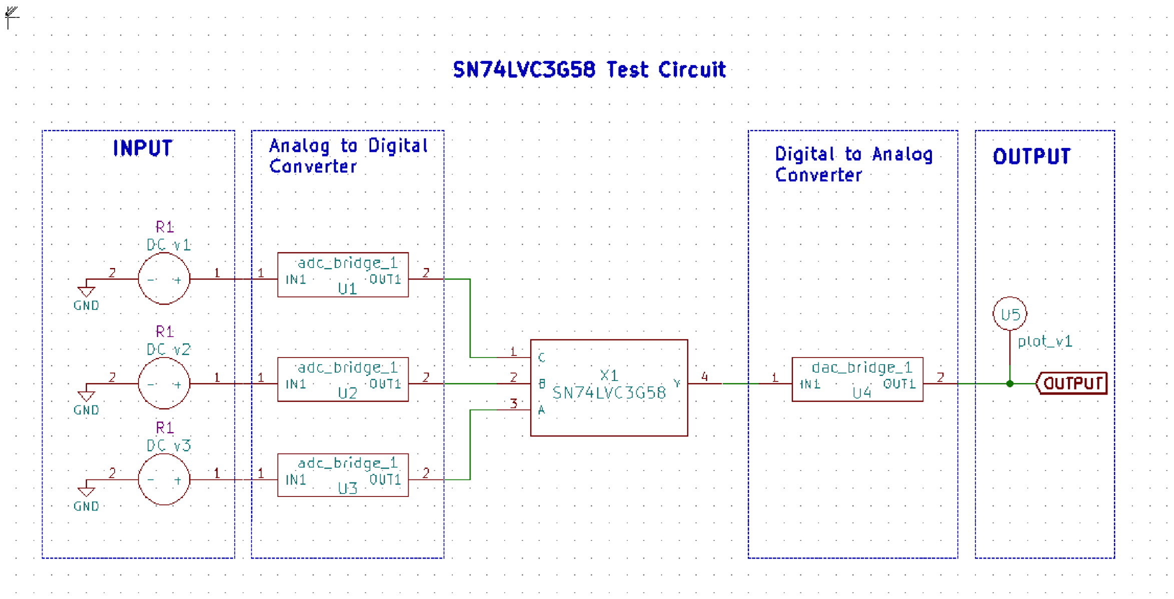
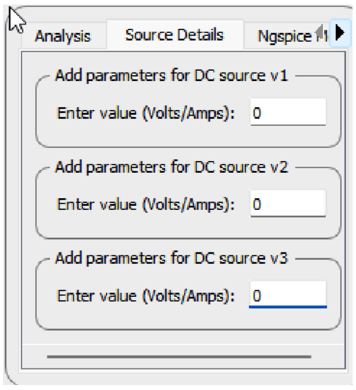


Figure 4.64: Test Circuit Layout

The figure illustrates a test setup used to verify the operation of the SN74LVC3G58 IC. The test circuit typically includes a power source, signal generators for inputs, and measurement instruments like oscilloscopes or logic analyzers. Since this IC provides configurable logic, testing different input combinations ensures that each logic function operates correctly and meets design specifications.

4.4.4 Output Waveform

The figure represents the signals generated at the output pins of the SN74LVC3G58 IC after processing the input signals. Each output waveform corresponds to the logic function performed by the IC, whether as a multiplexer or another logic gate.



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.65: Input

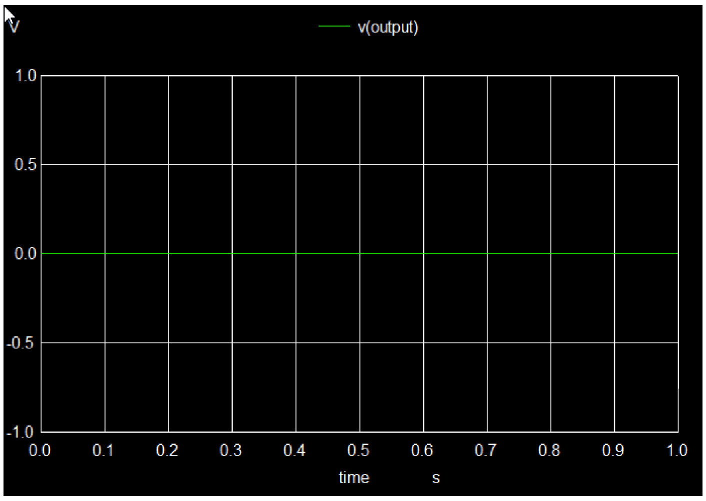
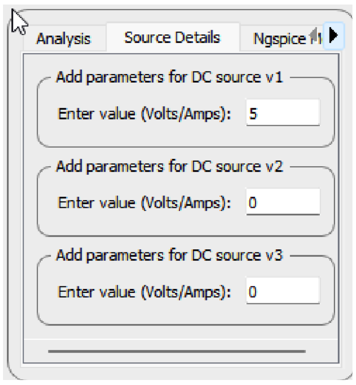


Figure 4.66: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 5
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 0

Figure 4.67: Input

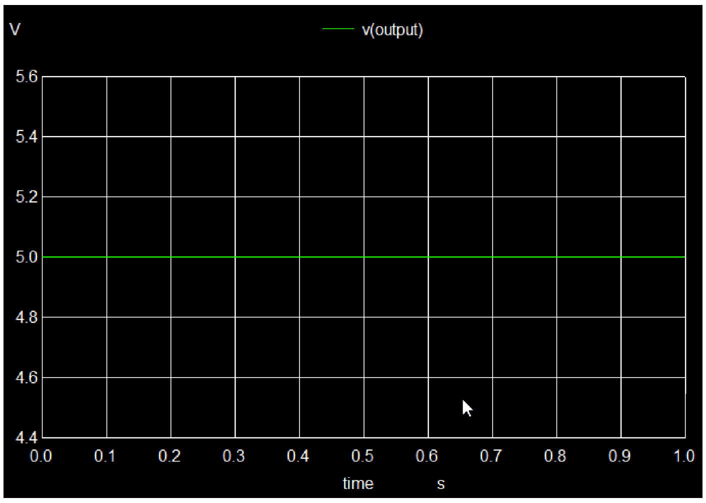
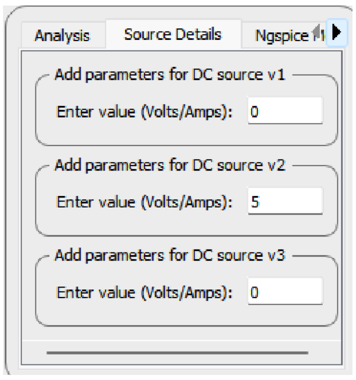


Figure 4.68: Output


Analysis | Source Details | Ngspice

Add parameters for DC source v1

Enter value (Volts/Amps): 0

Add parameters for DC source v2

Enter value (Volts/Amps): 5

Add parameters for DC source v3

Enter value (Volts/Amps): 0

Figure 4.69: Input

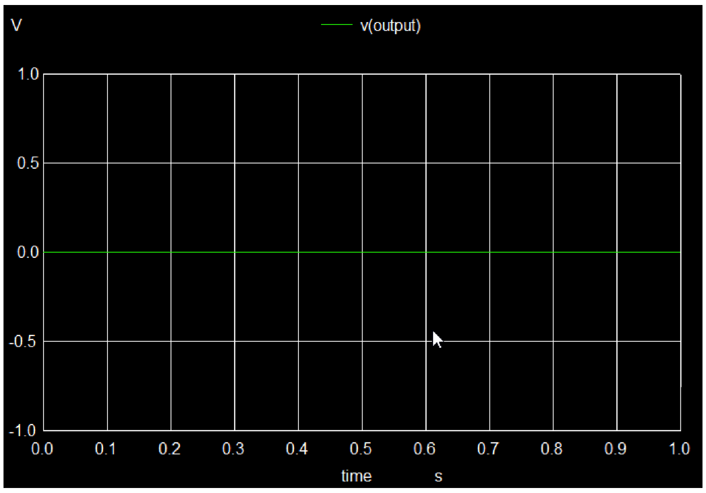
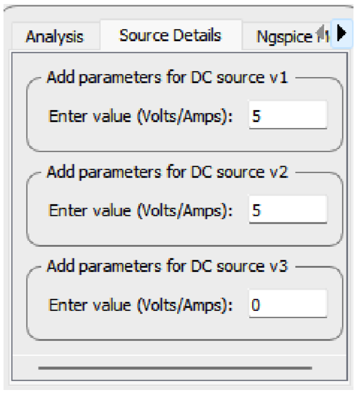


Figure 4.70: Output


Analysis | Source Details | Ngspice

Add parameters for DC source v1

Enter value (Volts/Amps): 5

Add parameters for DC source v2

Enter value (Volts/Amps): 5

Add parameters for DC source v3

Enter value (Volts/Amps): 0

Figure 4.71: Input

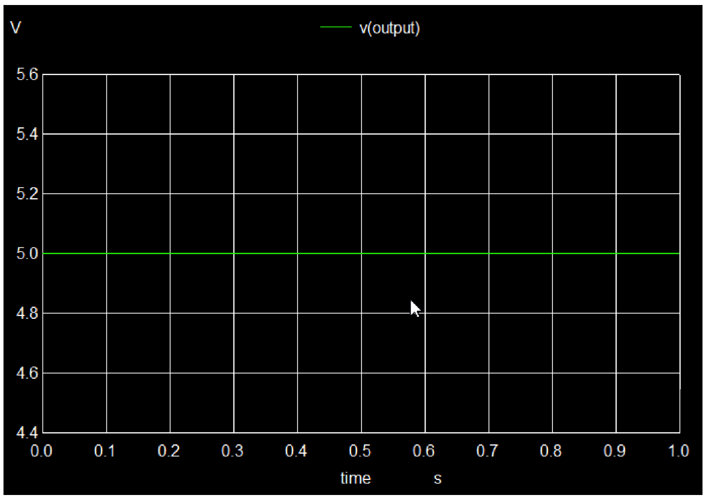
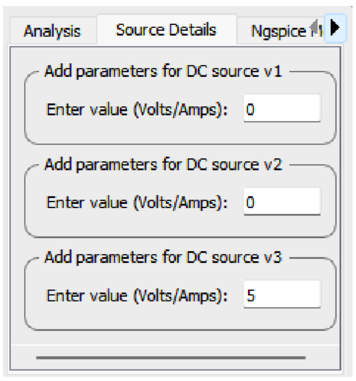


Figure 4.72: Output


Analysis | Source Details | Ngspice

Add parameters for DC source v1

Enter value (Volts/Amps): 0

Add parameters for DC source v2

Enter value (Volts/Amps): 0

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.73: Input

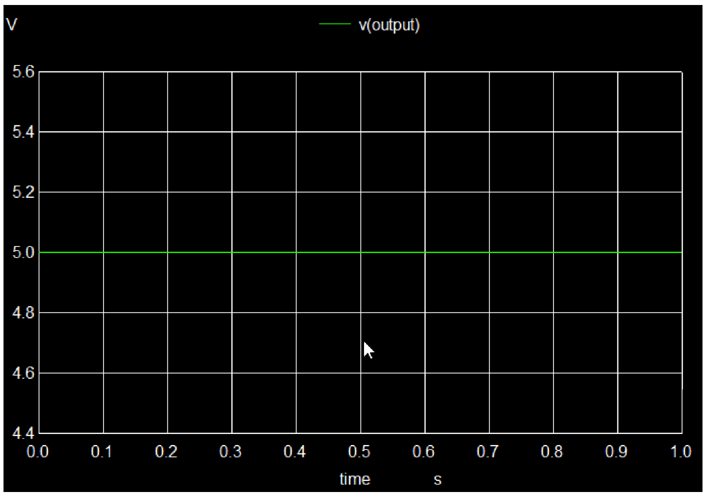
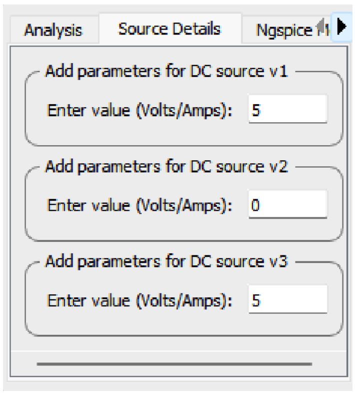


Figure 4.74: Output



Analysis | Source Details | Ngspice

Add parameters for DC source v1

Enter value (Volts/Amps): 5

Add parameters for DC source v2

Enter value (Volts/Amps): 0

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.75: Input

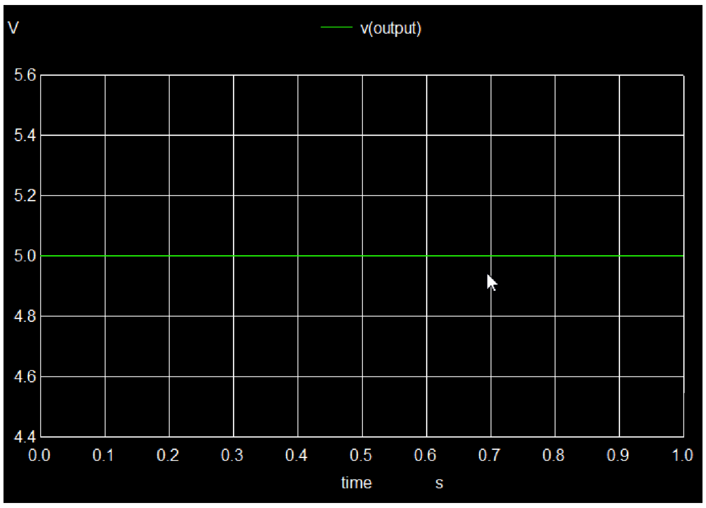
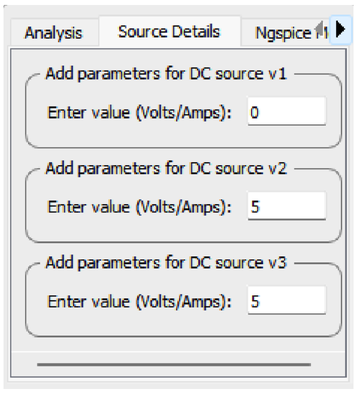


Figure 4.76: Output



Analysis | Source Details | Ngspice

Add parameters for DC source v1

Enter value (Volts/Amps): 0

Add parameters for DC source v2

Enter value (Volts/Amps): 5

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.77: Input

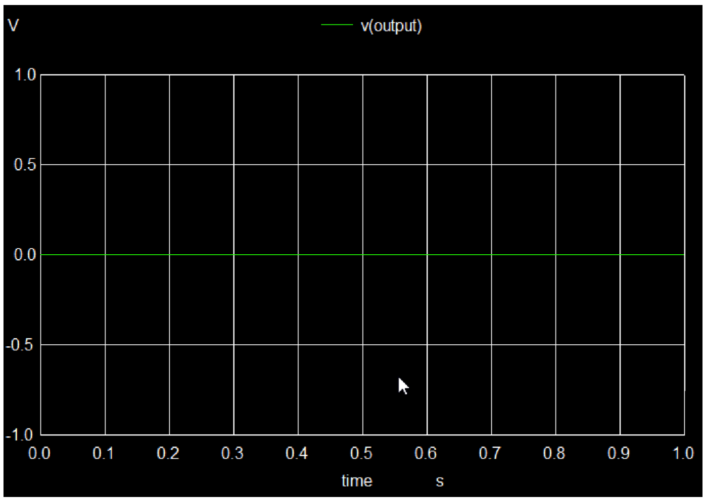
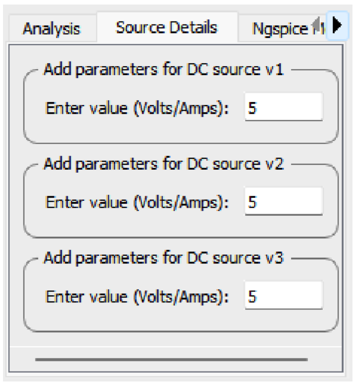


Figure 4.78: Output



Analysis | Source Details | Ngspice

Add parameters for DC source v1

Enter value (Volts/Amps): 5

Add parameters for DC source v2

Enter value (Volts/Amps): 5

Add parameters for DC source v3

Enter value (Volts/Amps): 5

Figure 4.79: Input

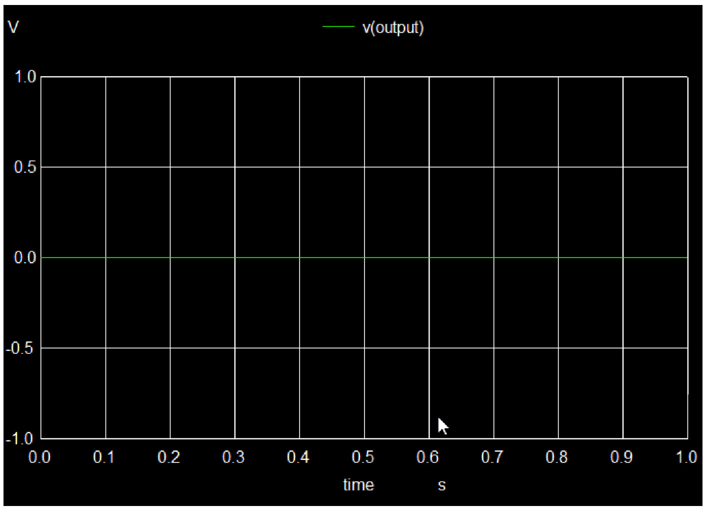
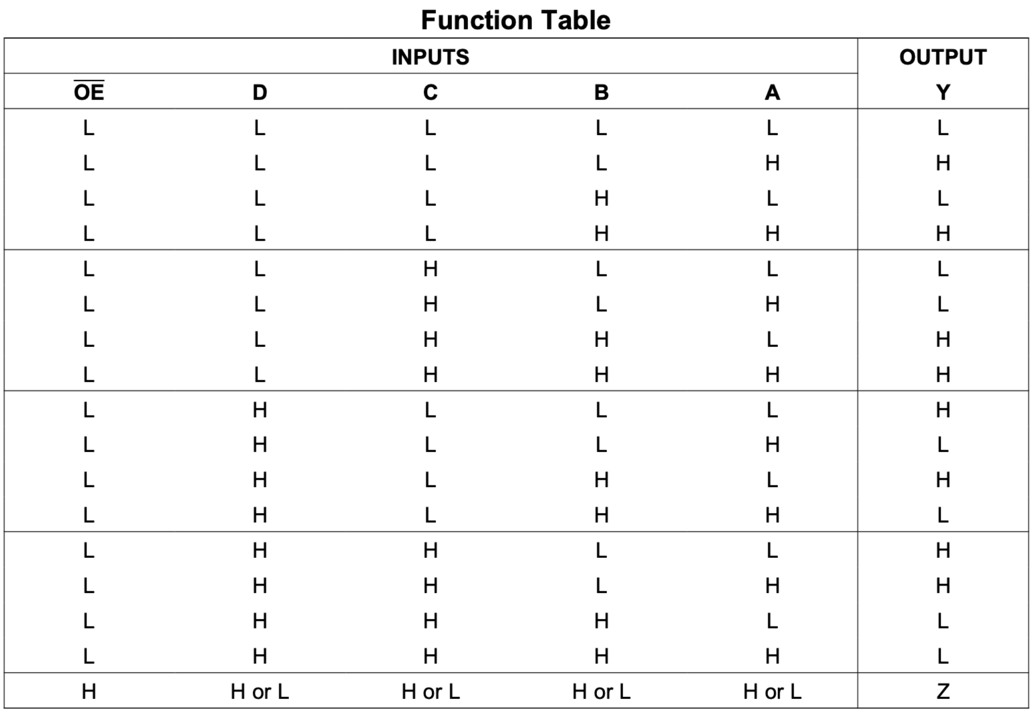


Figure 4.80: Output

4.5 SN74LVC1G99

The SN74LVC1G99 is widely used in applications requiring configurable logic gates for flexible circuit design. It is used in control logic, digital signal processing, and memory address decoding. Its ability to perform multiple logic functions within a single device helps reduce component count and enhances design flexibility in compact electronic systems.

Truth Table :



Function Table

INPUTS					OUTPUT
$\overline{OE}$	D	C	B	A	Y
L	L	L	L	L	L
L	L	L	L	H	H
L	L	L	H	L	L
L	L	L	H	H	H
L	L	H	L	L	L
L	L	H	L	H	L
L	L	H	H	L	H
L	L	H	H	H	H
L	H	L	L	L	H
L	H	L	L	H	L
L	H	L	H	L	H
L	H	L	H	H	L
L	H	H	L	L	H
L	H	H	L	H	H
L	H	H	H	L	L
L	H	H	H	H	L
H	H or L	H or L	H or L	H or L	Z

Figure 4.81: Truth Table

4.5.1 Pin Diagram

The figure shows the physical representation of the SN74LVC1G99 IC, indicating the arrangement of its pins. This IC is a single configurable multiple-function gate that allows users to implement various logic operations.

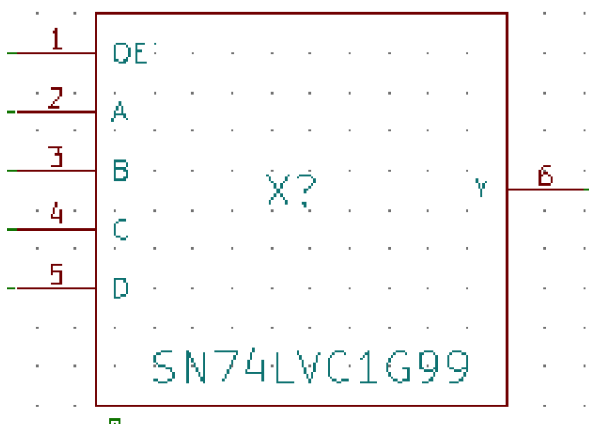


Figure 4.82: Pin Layout

It consists of multiple input pins for logic selection, an output pin for the processed signal, a power supply pin, and a ground pin. Proper pin configuration is necessary for ensuring accurate logic operation and seamless circuit integration.

4.5.2 Sub Circuit Layout

The figure represents the internal structure of the SN74LVC1G99 IC, illustrating how its internal logic components are interconnected. This IC contains a configurable logic function that allows it to perform multiple Boolean operations depending on the input conditions. The internal circuitry includes logic elements and control mechanisms that enable flexibility in digital designs. Understanding this layout helps in optimizing circuit performance and functionality.

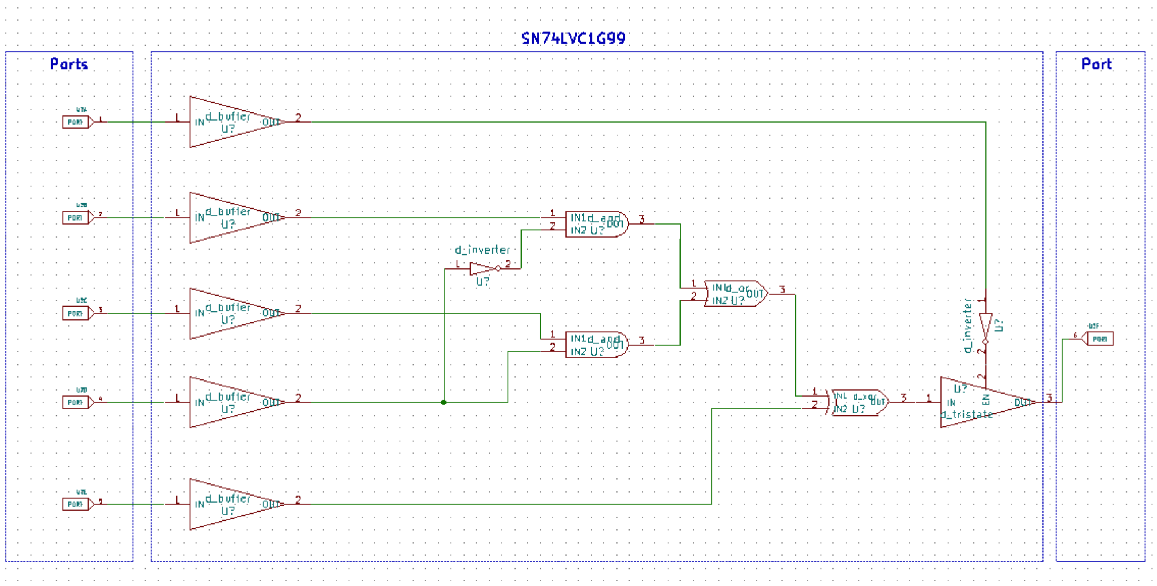


Figure 4.83: Sub Circuit Layout

4.5.3 Test Circuit

The figure illustrates a test setup designed to verify the operation of the SN74LVC1G99 IC. The test circuit typically includes a power supply, input signal generators, and measuring instruments such as oscilloscopes or logic analyzers. Since this IC supports multiple logic functions, testing various input combinations ensures that it performs the intended logic operations correctly under different conditions.

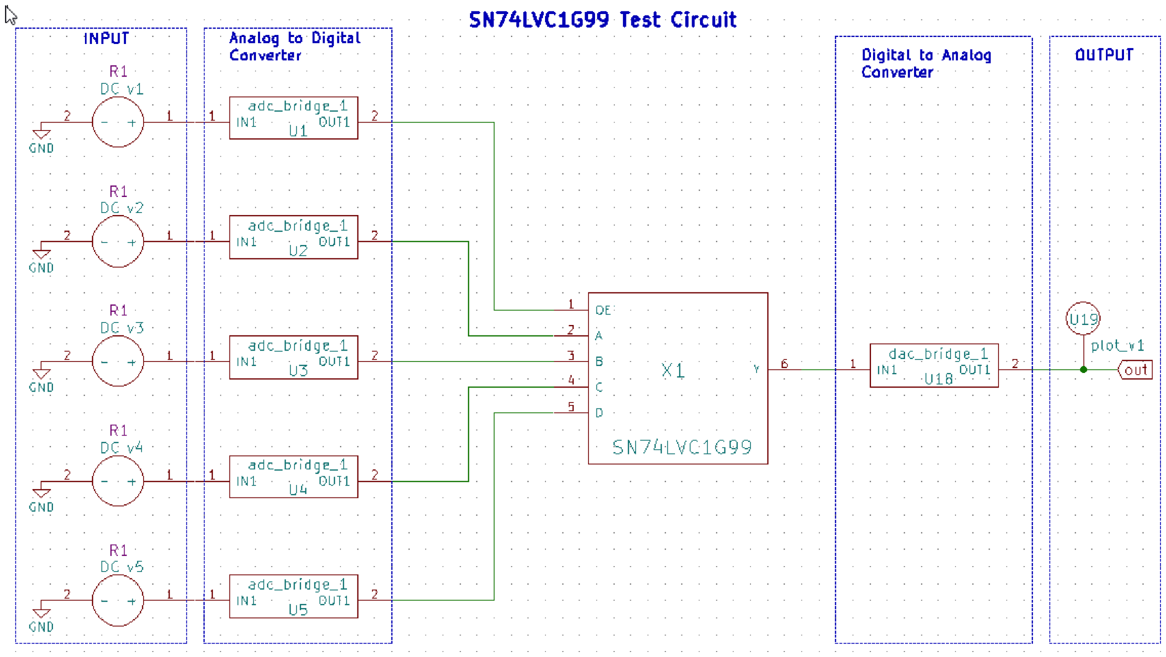
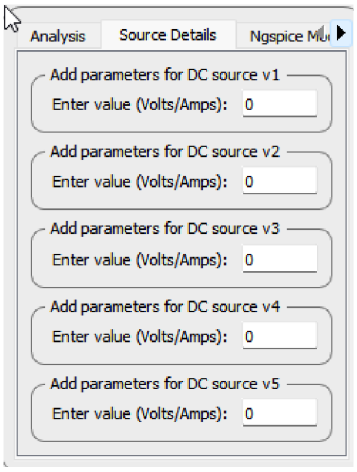


Figure 4.84: Test Circuit Layout

4.5.4 Output Waveform

The figure represents the signal generated at the output pin of the SN74LVC1G99 IC after processing the input signals. The output waveform corresponds to the logic function being executed and should match the expected result based on input conditions. Any deviations in the output signal may indicate incorrect input configurations, power supply fluctuations, or circuit instability.



Analysis | Source Details | Ngspice Mu

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Add parameters for DC source v4
Enter value (Volts/Amps): 0

Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.85: Input

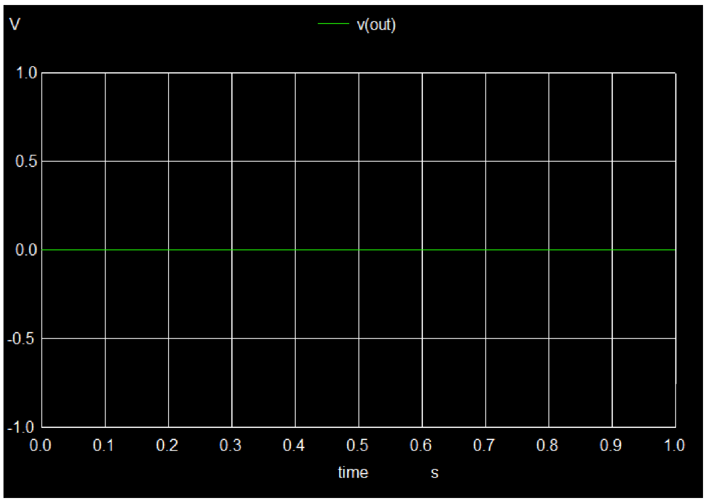
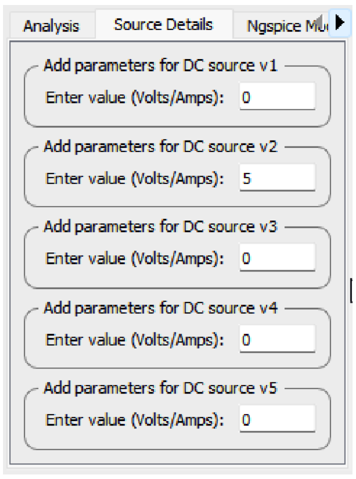


Figure 4.86: Output



Analysis | Source Details | Ngspice Mu

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Add parameters for DC source v4
Enter value (Volts/Amps): 0

Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.87: Input

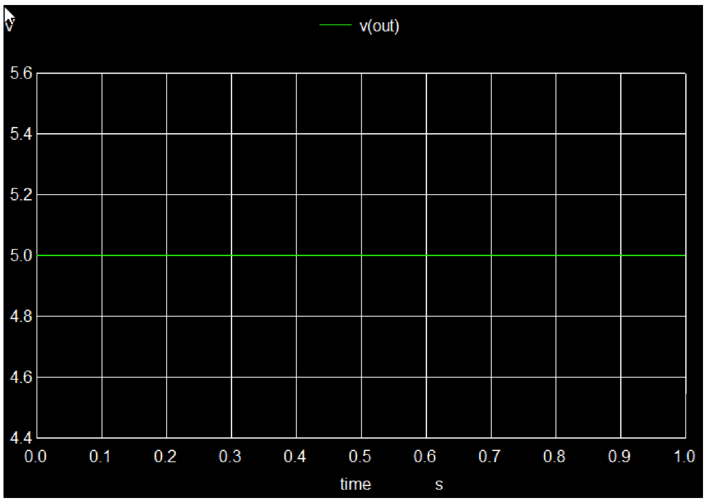
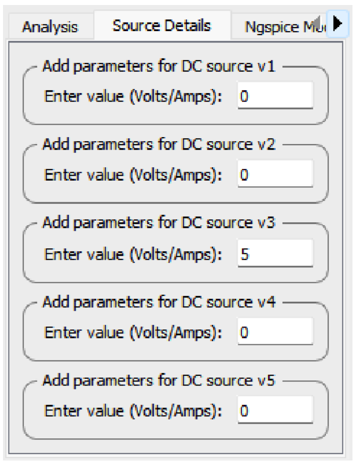


Figure 4.88: Output



Analysis | Source Details | Ngspice Mu

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 5

Add parameters for DC source v4
Enter value (Volts/Amps): 0

Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.89: Input

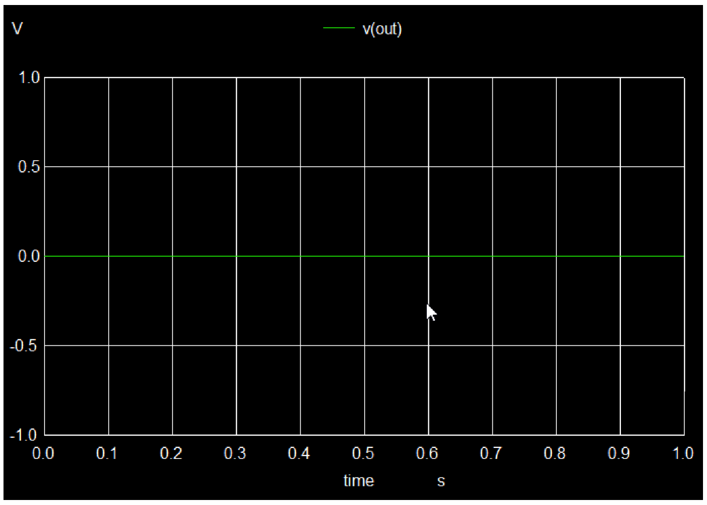
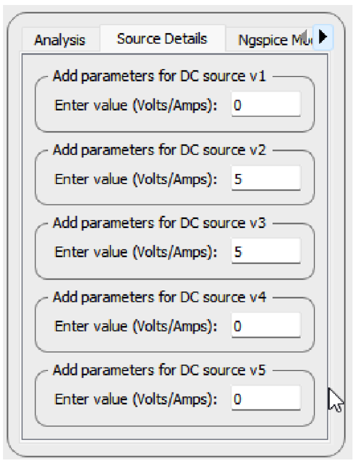


Figure 4.90: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 5
Add parameters for DC source v4
Enter value (Volts/Amps): 0
Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.91: Input

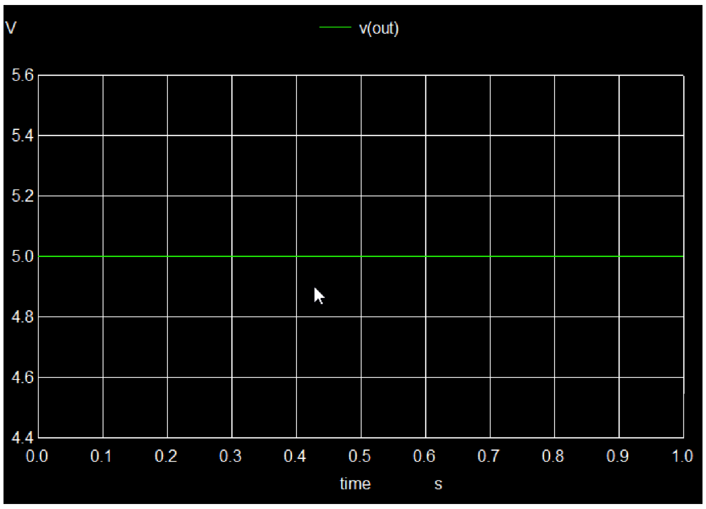
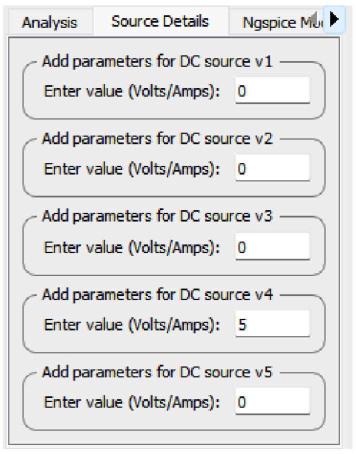


Figure 4.92: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 0
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.93: Input

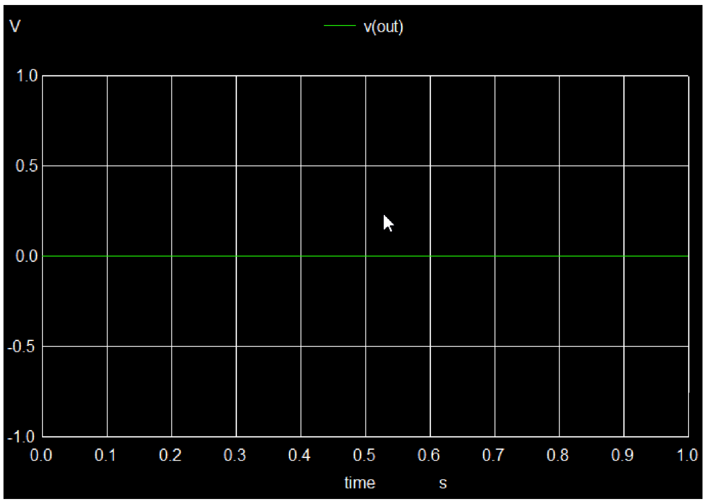
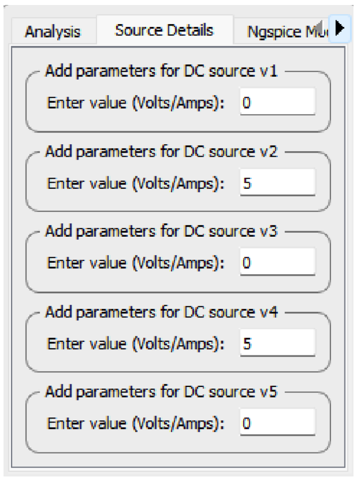


Figure 4.94: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 0
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.95: Input

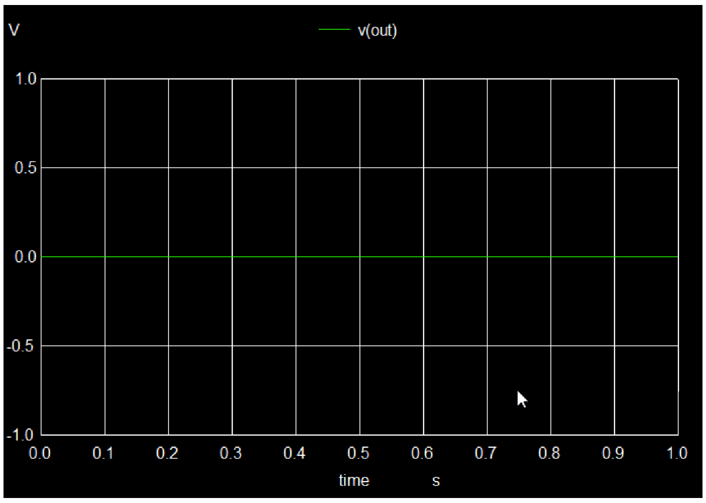
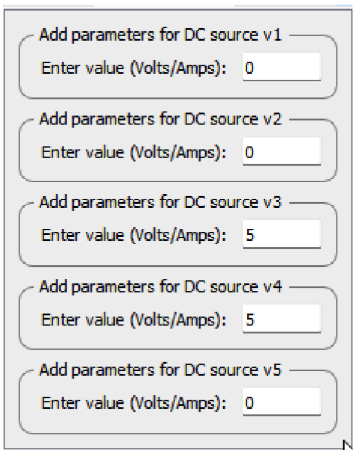


Figure 4.96: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 5
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.97: Input

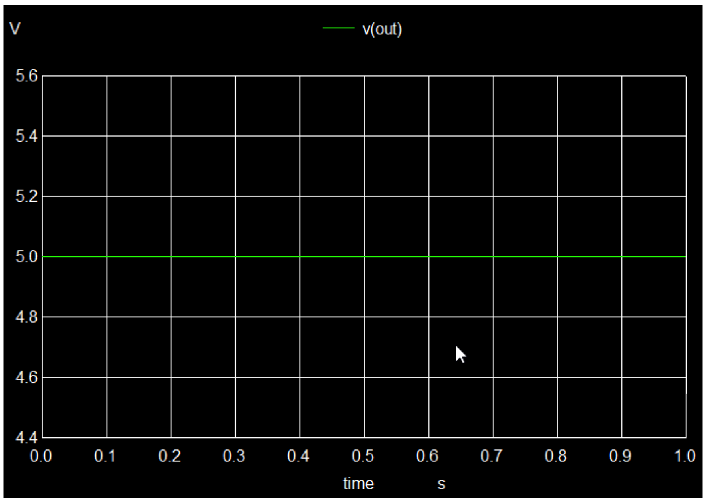
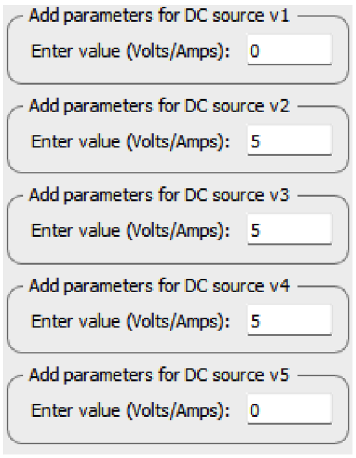


Figure 4.98: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 5
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 0

Figure 4.99: Input

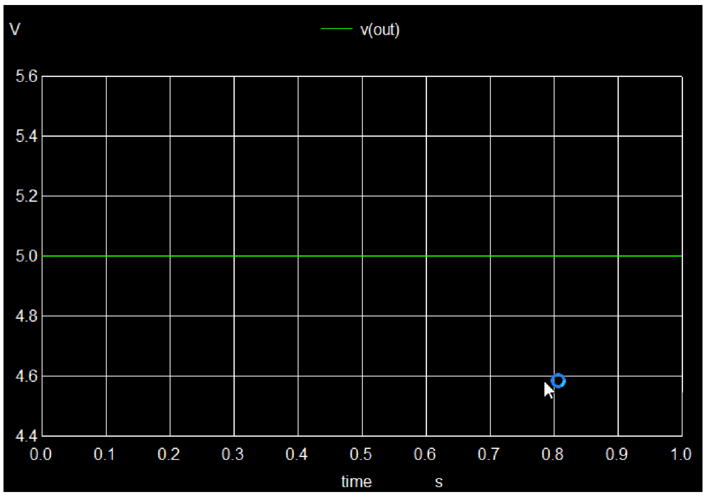
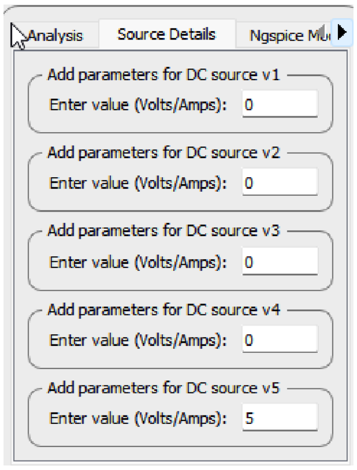


Figure 4.100: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 0
Add parameters for DC source v4
Enter value (Volts/Amps): 0
Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.101: Input

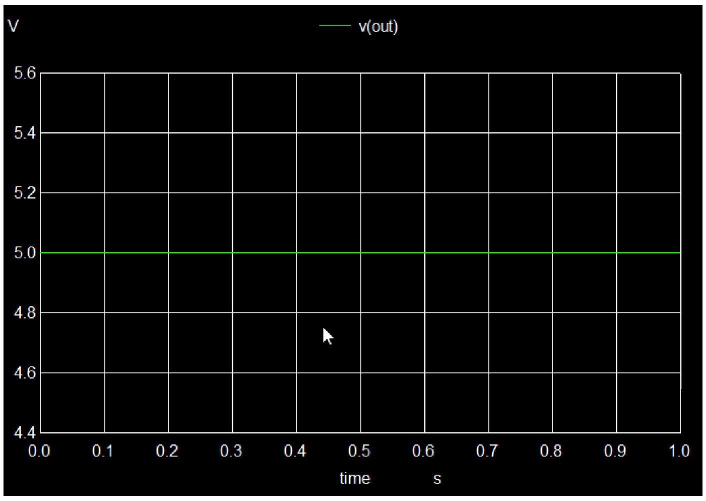
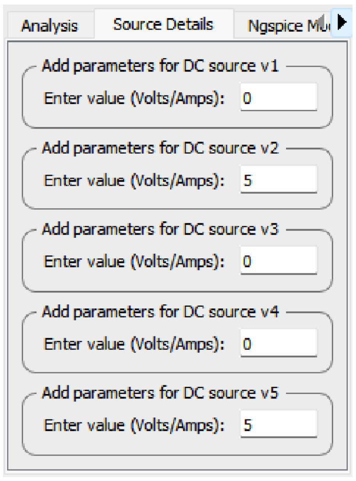


Figure 4.102: Output


Analysis | Source Details | Ngspice Mu

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 0

Add parameters for DC source v4
Enter value (Volts/Amps): 0

Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.103: Input

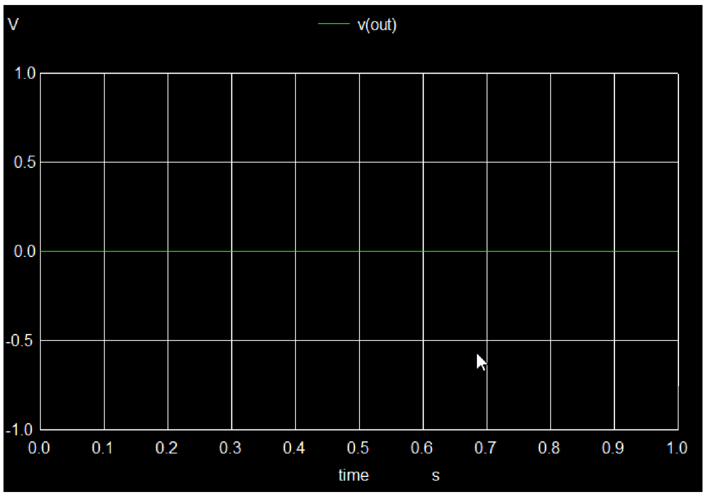
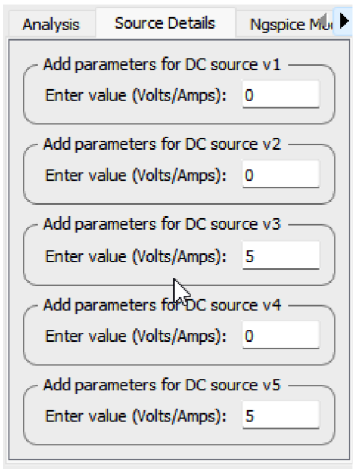


Figure 4.104: Output


Analysis | Source Details | Ngspice Mu

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 0

Add parameters for DC source v3
Enter value (Volts/Amps): 5

Add parameters for DC source v4
Enter value (Volts/Amps): 0

Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.105: Input

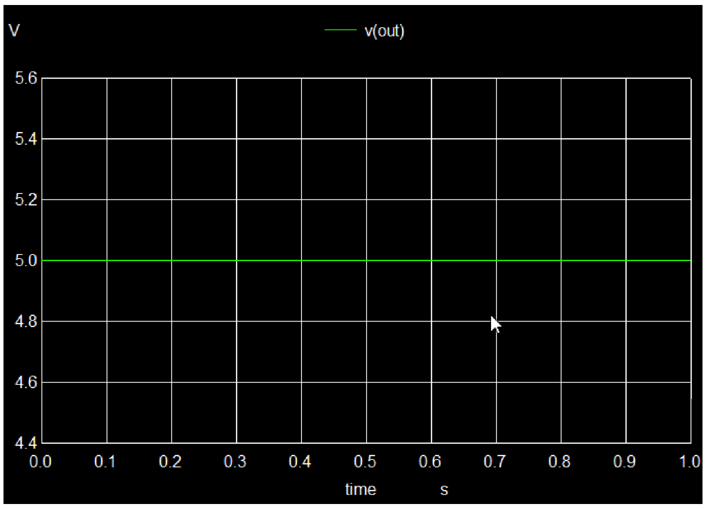
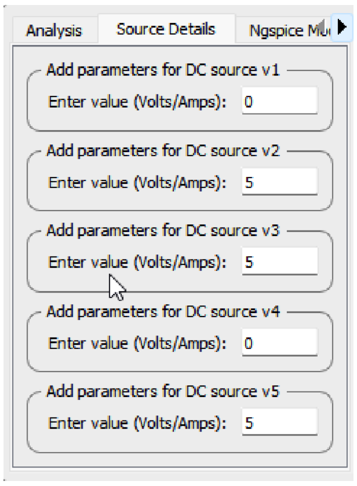


Figure 4.106: Output


Analysis | Source Details | Ngspice Mu

Add parameters for DC source v1
Enter value (Volts/Amps): 0

Add parameters for DC source v2
Enter value (Volts/Amps): 5

Add parameters for DC source v3
Enter value (Volts/Amps): 5

Add parameters for DC source v4
Enter value (Volts/Amps): 0

Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.107: Input

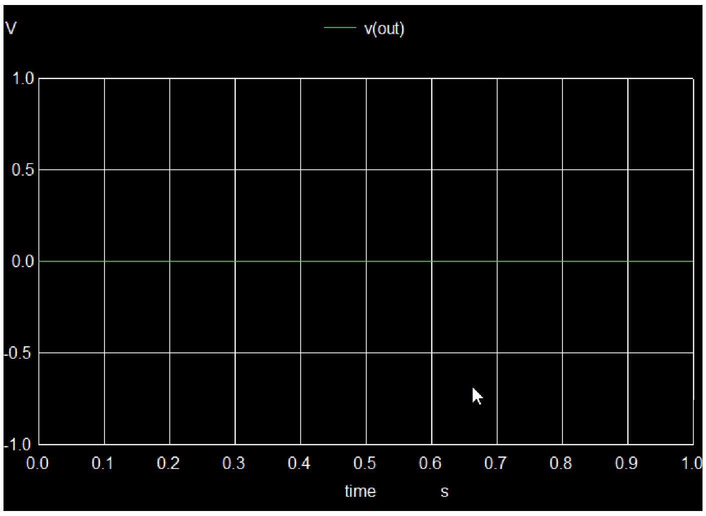
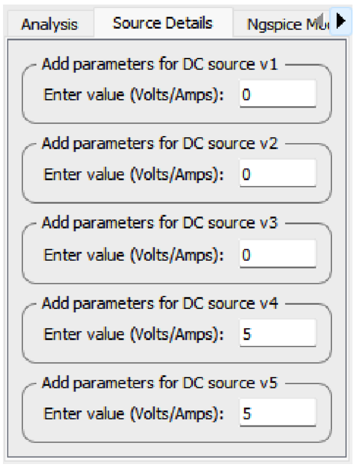


Figure 4.108: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 0
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.109: Input

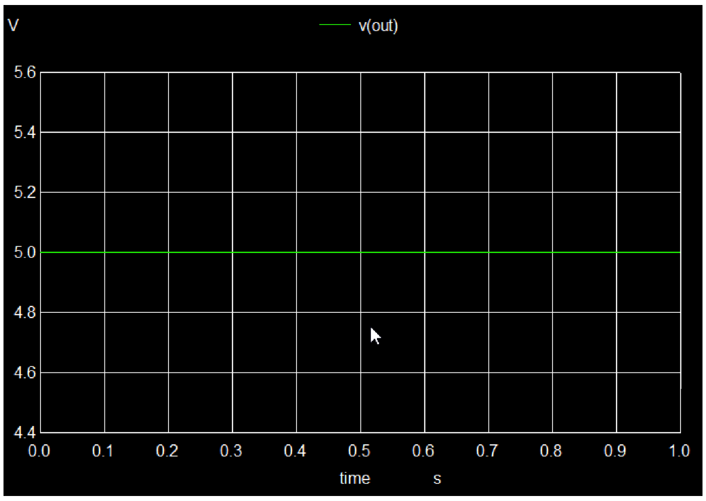
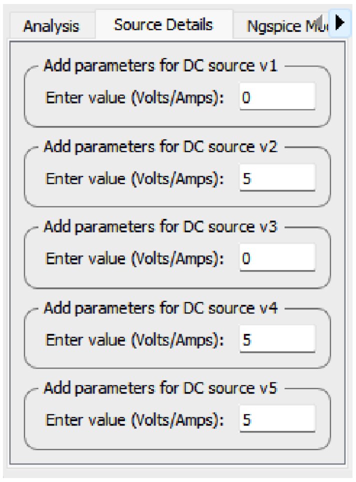


Figure 4.110: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 0
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.111: Input

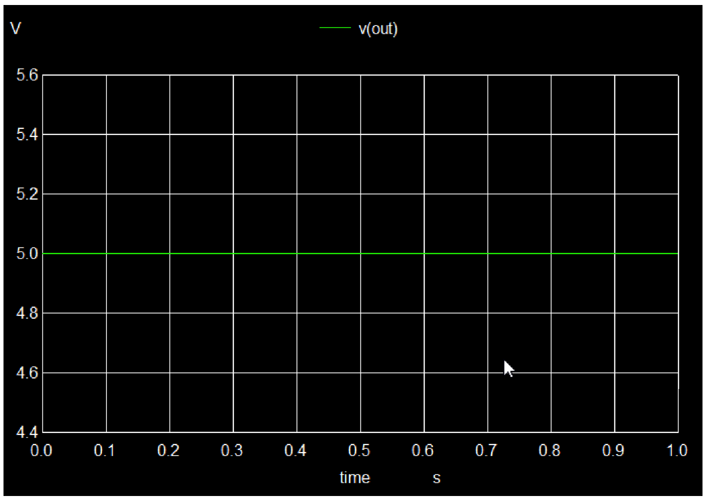
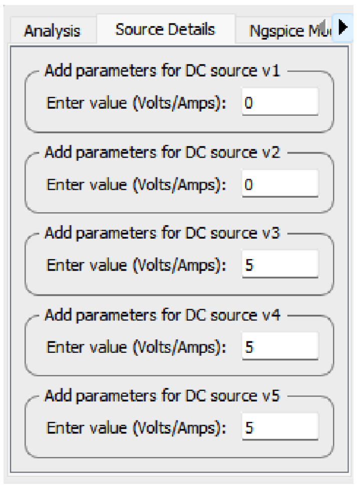


Figure 4.112: Output



Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 0
Add parameters for DC source v3
Enter value (Volts/Amps): 5
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.113: Input

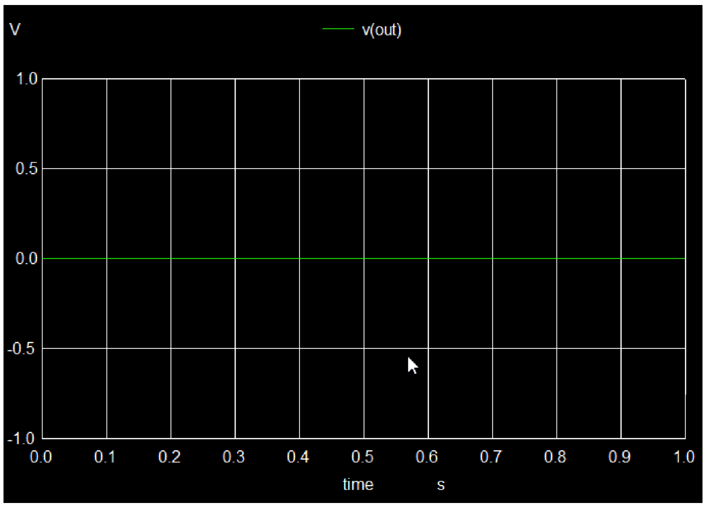


Figure 4.114: Output

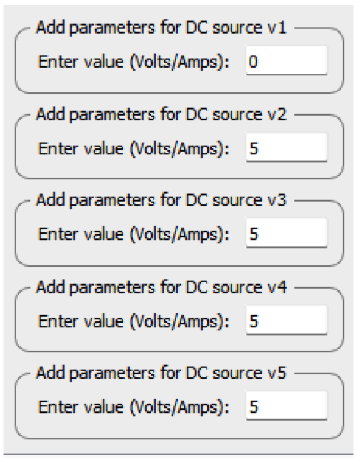

Add parameters for DC source v1
Enter value (Volts/Amps): 0
Add parameters for DC source v2
Enter value (Volts/Amps): 5
Add parameters for DC source v3
Enter value (Volts/Amps): 5
Add parameters for DC source v4
Enter value (Volts/Amps): 5
Add parameters for DC source v5
Enter value (Volts/Amps): 5

Figure 4.115: Input

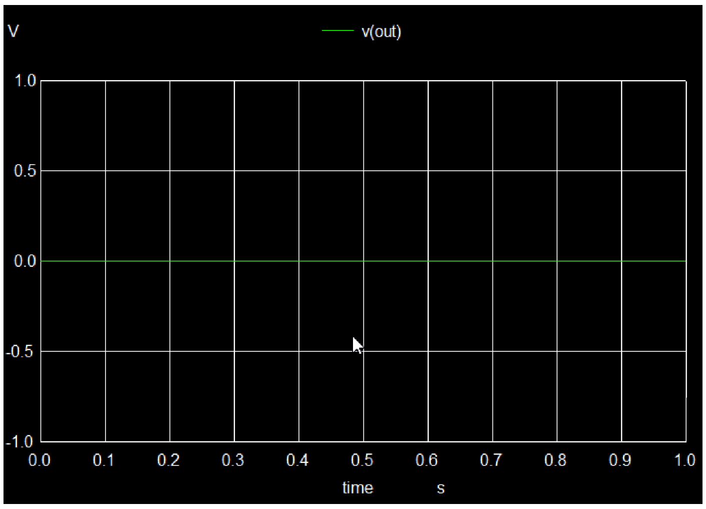


Figure 4.116: Output

Chapter 5

Adapting eSim for macOS: Dependency Management, Compatibility Issues, and Solutions

5.1 Introduction

eSim is an open-source EDA tool designed for circuit design, simulation, and PCB layout. Initially built for Linux (Ubuntu), adapting it to macOS presented various challenges, including dependency management, compatibility issues, and build failures.

5.2 Transition from apt-get to Homebrew

5.2.1 Ubuntu Approach

In Ubuntu, package dependencies were installed using:

```
sudo apt-get install <package_name>
```

Ubuntu's package manager ensures compatibility with Linux distributions but is not available on macOS.

5.2.2 macOS Approach

To replace `apt-get`, Homebrew (`brew`) was used. All `sudo apt-get install` commands were replaced with:

```
brew install <package_name>
```

For example:

```
sudo apt-get install verilator
```

was changed to:

```
brew install verilator
```

5.3 Python Package Installation Differences

5.3.1 Ubuntu

Python dependencies were managed using:

```
pip install <package>
```

or

```
pip3 install <package>
```

5.3.2 macOS

On macOS, explicit use of `pip3 install` was necessary for Python 3 compatibility. The required Python packages such as `watchdog`, `makerchip-app`, and `sandpiper-saas` were installed with:

```
pip3 install watchdog makerchip-app sandpiper-saas
```

5.4 Handling Permissions on macOS

5.4.1 Executable Permissions

On Ubuntu, files were made executable using:

```
chmod +x <file_name>
```

The same command was used on macOS to ensure files were executable.

5.4.2 Gatekeeper and Security Restrictions

macOS uses Gatekeeper, a security feature that blocks unsigned applications. If an application was blocked, users had to allow it manually via:

```
sudo spctl --add <file_name>
```

Although not directly required in the installation script, some users might need to bypass Gatekeeper for certain executables.

5.5 Verilator Configuration and Compatibility Issues

5.5.1 Issues Encountered

On macOS, configuring Verilator caused compatibility problems due to missing dependencies. The standard:

```
brew install verilator
```

command worked, but some versions caused conflicts.

5.5.2 Solution: Identifying Working Versions

Through testing, the working version of dependencies was identified from **16/12/24 onwards**. Higher versions introduced unexpected errors and needed specific patches. The stable version was installed using:

```
brew install verilator@<specific_version>
```

5.6 Package Management Issues

5.6.1 Ubuntu-Specific Issues

On Ubuntu, dependency errors commonly arose due to:

- Outdated package libraries.
- Missing dependencies due to broken `apt` sources.
- Conflicts between pre-installed and new libraries.

5.6.2 Homebrew Issues on macOS

Homebrew Compatibility Problems:

- Some formulae were unavailable or outdated.
- Certain packages required additional paths to be set manually.

For example, Verilator required linking using:

```
brew link verilator
```

Incorrect Path Configurations: Homebrew installs binaries in `/opt/homebrew/bin`, which is not always included in the system path. This was fixed using:

```
export PATH="/opt/homebrew/bin:$PATH"
```

5.7 Version Conflicts and Their Impact

5.7.1 Identified Issues

While certain dependency versions built successfully, newer versions created conflicts. For example:

- The working version of GCC compiled the project without errors.
- A higher GCC version resulted in undefined references during linking.

5.7.2 Solution

Ensuring that only tested versions were installed by using:

```
brew install gcc@<specific_version>
```

5.8 Post-Build Issues: Makefile and NGHDL

5.8.1 Makefile Issues

After the build process, errors appeared while generating the Makefile. This issue was resolved using GCC, which compiled and linked the required components successfully.

5.8.2 NGHDL Compilation Errors

NGHDL still faced errors while creating the Makefile. The `sudo rm` command was updated to:

```
sudo rm -rf <directory>
```

to prevent permission-related problems.

5.9 Clang Compatibility Issues

5.9.1 Problem Statement

Different Clang versions were tested, but the issue persisted. Compilation errors were encountered even with multiple Clang versions.

5.9.2 Possible Solutions

Further debugging is needed to determine the root cause of Clang-related issues. Potential solutions include:

- Using an alternate compiler like GCC.
- Modifying build flags for better compatibility.

5.10 Conclusion

Porting eSim to macOS required:

- Replacing `apt` commands with `brew` equivalents.
- Handling Gatekeeper and macOS permissions.
- Identifying working versions of dependencies.
- Fixing Makefile and NGHDL errors.

While major challenges were resolved, NGHDL and Clang-related issues need further debugging.

Note : eSim Mac installer is still in developing phase.

Chapter 6

Conclusion and Future Scope

Developed a range of subcircuits for both Analog and Digital Integrated Circuits (ICs), strictly following the specifications from their official datasheets. These include essential components like Op-Amps, Voltage Regulators, and Comparators, as well as other foundational digital and analog ICs. Each IC model was thoroughly tested using appropriate test circuits to ensure accurate performance. These IC models are now ready for integration into eSim's subcircuit library, providing a comprehensive set of building blocks for developers and students to use in a wide array of circuit designs and simulations. With continued development, more such IC models are expected to enhance the library's offerings.

The adaptation of eSim for macOS is still in the development phase. While significant progress has been made, including replacing `apt-get` with `brew`, resolving major dependency issues, and ensuring compatibility with Homebrew packages, certain challenges remain unresolved. NGHDL still faces errors during Makefile generation, and Clang-related issues persist despite testing multiple versions. Further debugging and optimization are required to achieve a fully functional macOS installer for eSim. The development is ongoing, and future efforts will focus on resolving these remaining issues to ensure a seamless installation and execution process on macOS.

Bibliography

[1] FOSSEE Official Website. Available: https://fossee.in/about

[2] eSim Official Website. Available: https://esim.fossee.in/

[3] NXP Semiconductors, ”74LVC1G386 Datasheet.” Available: https://www.mouser.com/datasheet/2/302/74LVC1G386_2-57080.pdf

[4] Texas Instruments, ”SN74LVC1G57 Datasheet.” Available: https://www.ti.com/lit/ds/symlink/sn74lvc1g57.pdf

[5] Texas Instruments, ”SN74LVC1G99 Datasheet.” Available: https://www.ti.com/lit/ds/symlink/sn74lvc1g99.pdf

[6] Texas Instruments, ”SN74LVC3G58 Datasheet.” Available: https://www.ti.com/lit/ds/symlink/sn74lvc3g58.pdf

[7] Texas Instruments, ”SN74LVC3G98 Datasheet.” Available: https://www.ti.com/lit/ds/symlink/sn74lvc3g98-q1.pdf