# Semester Long Internship Report

On

## IC Design Using subcircuit in eSim

Submitted by

**Deepika Chokka**

Under the guidance of

**Prof. Kannan M. Moudgalya**

Chemical Engineering Department

IIT Bombay

**july 2, 2025**

# Acknowledgement

I would like to express my heartfelt gratitude to the FOSSEE team at IIT Bombay for giving me the invaluable opportunity to work on the eSim project. Being selected for this fellowship was a moment of immense pride, and the experience has been truly enriching both technically and personally.

My sincere thanks go to Prof. Kannan M. Moudgalya for initiating and leading this impactful platform, which empowers students like me to explore the world of open-source electronics. I am especially grateful to Mr. Sumanto Kar for his constant guidance, patience, and encouragement throughout the project. Your support—from understanding datasheets and research articles to effectively using tools like GitHub and eSim—has been instrumental in shaping my learning journey. The constructive feedback during reviews helped me approach problems with greater clarity and confidence.

When I began this fellowship, I had limited hands-on experience with simulation tools and felt uncertain about my ability to complete a project independently. However, through this opportunity, I have developed not only technical skills but also discipline, time management, and self-confidence. I would also like to thank my fellow interns and team members for fostering a collaborative, energetic, and friendly environment that made the journey enjoyable and memorable.

This experience has significantly contributed to my personal and professional growth, and I am proud to have been part of the FOSSEE eSim team. I look forward to applying the knowledge I have gained here to future endeavors and to contributing meaningfully to the open-source community. Thank you once again to everyone who made this experience so impactful and rewarding.

# ABSTRACT

This project focuses on the design, simulation, and analysis of digital circuits and industry-standard logic ICs using the open-source Electronic Design Automation (EDA) tool, eSim, developed by FOSSEE, IIT Bombay. The primary goal is to gain practical, hands-on experience in digital system design, both through the implementation of custom logic-based circuits and by recreating the internal logic of well-known ICs based on their datasheets. The project involves the design of fundamental digital systems including a Digital Lock System, Vedic Multiplier, Binary Comparator, and Frequency Divider. These circuits were chosen for their relevance in real-world embedded applications and their value in reinforcing the understanding of both combinational and sequential logic.

In addition to these circuits, the project also focuses on reconstructing the functionality of several standard digital ICs. These include the 74HC280 Even and Odd Parity Checker, 7447 BCD to Seven Segment Decoder,7442 BCD to Decimal Decoder and 74150 16*1 Multiplexer. The logic designs for these ICs were derived directly from their respective datasheets, including truth tables and logic diagrams, and implemented using eSim's schematic editor. This method provided valuable insight into the internal workings of these devices and the logic used in their construction.

All designs were tested through detailed simulation using eSim's NgSpice engine. Output waveforms were analyzed to verify the correctness of the circuits under various test conditions. By combining theoretical understanding with practical implementation, the project strengthens core concepts in digital electronics such as Boolean algebra, logic simplification, sequential state transitions, and digital arithmetic. Moreover, this work promotes the use of open-source tools in academic and research environments, encouraging accessibility and innovation in digital system design. Overall, the project bridges the gap between textbook-level learning and circuit-level realization, laying a strong foundation for advanced studies in embedded systems, FPGA development, and VLSI design.

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

Digital electronics forms the core of nearly all modern technological systems. From mobile phones and computers to industrial automation and aerospace systems, digital logic is the driving force behind processing, decision-making, and control. At the heart of digital electronics lie fundamental building blocks such as logic gates, arithmetic circuits, flip-flops, and encoders, which are carefully combined to create complex systems capable of performing advanced functions.

The foundation of digital system design is not limited to understanding theoretical principles. Practical implementation, simulation, and validation are equally important, particularly in today's fast-paced development cycles. The shift from traditional design methodologies to Computer-Aided Design (CAD) and Electronic Design Automation (EDA) tools has dramatically transformed how designers and engineers work. These tools provide a virtual environment for circuit creation, simulation, testing, and debugging before physical implementation.

However, a significant challenge in academia and small-scale development has been the reliance on expensive proprietary software, which restricts accessibility and scalability. In response to this challenge, the open-source community has developed robust alternatives. One such powerful tool is eSim, developed by FOSSEE (Free/Libre and Open Source Software for Education) at IIT Bombay.

eSim combines the power of Ngspice and KiCad, providing a flexible platform for both analog and digital circuit design. Its open-source nature allows unrestricted access for educational institutions and individual learners, promoting hands-on learning without the burden of license fees. The tool supports schematic capture, SPICE-based simulation, and visualization of results, all within an integrated environment.

This project aims to leverage eSim for the complete digital design workflow. From creating custom circuits to replicating the internal logic of standard ICs like 74HC280 and 7447, eSim provides an ideal platform for developing a deep and practical understanding of digital systems.

## 1.1 Motivation

Digital electronics courses often emphasize learning through prebuilt ICs, simulation of basic logic circuits, or pre-written HDL modules. While effective to some extent, such methods can limit conceptual understanding. This project aims to address that gap by adopting a hands-on, bottom-up design approach, focusing on reconstructing the internal logic of real-world ICs using their datasheets and implementing custom-designed circuits from scratch.

One key motivation for this work is the ability to interpret and translate IC datasheets into functional digital circuits. ICs such as the 74HC280 (Even and Odd Parity Checker), 7447 (BCD to Seven Segment Decoder), 7442 (BCD to Decimal Decoder), and 74150 (16*1 Multiplexer) are commonly used components in a wide variety of applications. However, their inner workings often remain a black box for students. By recreating these ICs using logic gates and verifying their functionality in simulation, students gain a strong conceptual grounding that textbooks alone cannot provide.

In addition, custom logic systems such as the Digital Lock System, Vedic Multiplier, Binary Comparator, and Frequency Divider were chosen for their practical relevance and conceptual richness. These circuits are widely used in authentication systems, arithmetic logic units, control systems, and signal processing applications. Designing and simulating these circuits from the ground up enhances critical thinking and problem-solving skills essential for careers in embedded systems, VLSI, and digital hardware design.

Another motivational factor is the exploration and advocacy of open-source tools in engineering education. Tools like eSim not only make learning more inclusive but also align with current trends in open hardware development, where collaboration, transparency, and accessibility play key roles. By using open-source tools, this project also contributes to the broader movement toward democratizing technology education.

## 1.2 Objectives

The objectives of this project are clearly defined to meet educational, technical, and practical learning goals. They are outlined as follows:

**Design and Simulation of Custom Digital Circuits:**

To design digital systems such as a Digital Lock, Vedic Multiplier, Binary Comparator, and Frequency Divider from scratch. To simulate these circuits in eSim using SPICE-compatible models. To validate their behavior using waveform analysis and truth table comparisons.

**Implementation of Standard Logic ICs:**

To study and analyze the internal logic diagrams, truth tables, and functional behavior of ICs such as 74HC280, 7447, 7442, and 74150.To recreate the logic-level implementations of these ICs in eSim using basic gates, flip-flops, and combinational logic components.To test and verify their outputs against standard datasheet specifications.

**Promoting Use of Open-Source EDA Tools:**

To demonstrate the capabilities of eSim as a comprehensive platform for digital logic design.To encourage the adoption of free and open-source tools in academic curricula.To highlight the role of accessible technology in skill development and engineering innovation.

**Bridging Theory with Practical Implementation:**

To move beyond textbook knowledge by integrating theoretical concepts with schematic-level realization.To develop design skills, logic optimization techniques, and simulation analysis capability.To foster a mindset of experimentation and exploration in digital electronics.

## 1.3 Overview of eSim Tool

FOSSEE (Free/Libre and Open Source Software for Education) is an initiative taken by the National Mission on Education through Information and Communication Technology (ICT), Ministry of Human Resource Development (MHRD), Government of India which has successfully developed various opensource tools and promotes the use of these tools in improving the quality of education and helping every individual avail these sources free of cost. The software is being developed in such a way that it can stay relevant with respect to the commercial softwares.

**eSim**

eSim is a free/libre and open source EDA tool for circuit design, simulation, analysis and PCB design developed by FOSSEE, IIT Bombay. It is an integrated tool built using free/libre and open source software such as KiCad, Ngspice, NGHDL and GHDL.

**NgSpice**

Ngspice is a general purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analysis. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent voltage and current sources, four types of dependent sources, lossless and lossy transmission lines (two separate implementations), switches, uniform distributed RC lines, and the five most common semiconductor devices: diodes, BJTs, JFETs, MESFETs, and MOSFET.Implementing some standard Integrated Circuits (IC) as subcircuits into the eSim library.

**Approach**

The approach used by me to implement the problem statement is to first look into the datasheets of some prominent Integrated Circuits manufactured by companies like Texas Instruments, Analog Devices, NXP Semiconductors among others. The ICs are so chosen such that there is a variety of utilities served. For example, the ones I have included range from precision amplifiers, comparators, and drivers to audio amplifiers etc. The subcircuits built are then tested to verify basic circuit configurations through NgSpice simulations.

The point-by-point roadmap of the process is as follows:

Browse through datasheets of relevant ICs that are not previously included into the eSim library. Check for the detailed schematic of the IC and implement the same in the eSim subcircuit builder. Convert the subcircuit from Kicad to NgSpice and create a pin diagram for the IC so that it gets added to the library. Then, the next part involves verifying the subcircuit through a test circuit. We create a new project and build a relevant circuit to test the IC. Convert the test circuit from Kicad to NgSpice and simulate it. We verify the waveforms from the datasheet and assess if the IC operates as per standards. The same process is followed for all other subcircuits.

Subcircuit Builder Method

Subcircuit is a way to implement hierarchical modelling. Once a subcircuit for a component is created, it can be used in other circuits. eSim provides an easy way to create a subcircuit. The following section deals with the step-by-step method of creating a subcircuit.

Subcircuit Creation

The steps to create subcircuit are as follows:After opening the Subcircuit tool, click on New Subcircuit Schematic button. It will ask the name of the subcircuit. Enter the name of subcircuit (without any spaces) and click OK. After clicking OK button it will open KiCad schematic. Draw your circuit which will be later used as a subcircuit. Once you complete the circuit, assign port to the node of your circuit which will be used to connect with the main circuit. PORTS can be found in the components section in the editor. These act as linkages to inputs and outputs of the main circuit. As the next step, save the subcircuit and generate a KiCad netlist for the same. Now, to use this as a subcircuit, create a block in the KiCad Eescchema and follow the below steps:

i. Go to library viewer of Eschema

ii. Choose the current working library as the eSim_Subckt.

iii. Click on create a new component with reference X.

iv. Start drawing the subcircuit block.

Update and save it Close the Eeschema window and click on Convert KiCad to Ngspice button in subcircuit builder tool. This will convert the KiCad spice netlist to Ngspice netlist. And it will save your subcircuit into eSim repository, which you can add in your main circuit.

## 1.4 Methodology

**Circuit Selection**

The initial step involved selecting circuits fundamental to digital system design and relevant to real-world applications.Examples include:Digital Lock System – Showcasing sequential logic and FSMs. Parity Checkers – Demonstrating error detection basics. Comparators and Adders – Highlighting combinational logic applications.Frequency Divider – Introducing clock manipulation and timing. Selections were based on:Academic relevance (e.g., syllabus alignment). Practical demonstration potential.Compatibility with the eSim tool.

**Logic Design** Each circuit was logically designed using:Truth Tables to define input-output relationships.Karnaugh Maps (K-maps) for Boolean simplification.State Diagrams in sequential designs like the digital lock and frequency divider. This ensured optimized logic expressions that translated well into schematics.

**Schematic Implementation** The simplified logic was then implemented schematically using: eSim's KiCad interface, a schematic capture tool. Logical gates, flip-flops, and ICs were interconnected using digital components from the built-in and custom libraries.

**Simulation Setup** Circuits were simulated using:Ngspice backend, integrated into eSim.Testbenches with controlled input vectors and timing diagrams to observe behavior.Voltage sources and pulse generators to simulate clock inputs and toggles.

**Validation and Debugging** This step involved:Analyzing waveform outputs for expected logic levels using the simulation viewer.Debugging errors like incorrect logic levels, race conditions, or missing transitions.Iterative refinement of the schematic or logic expression based on observed faults.

**Documentation** Each stage was thoroughly documented:Screenshots of schematics and waveforms.Simulation logs capturing output for reference.Observations and insights noted to highlight learning points and encountered issues.

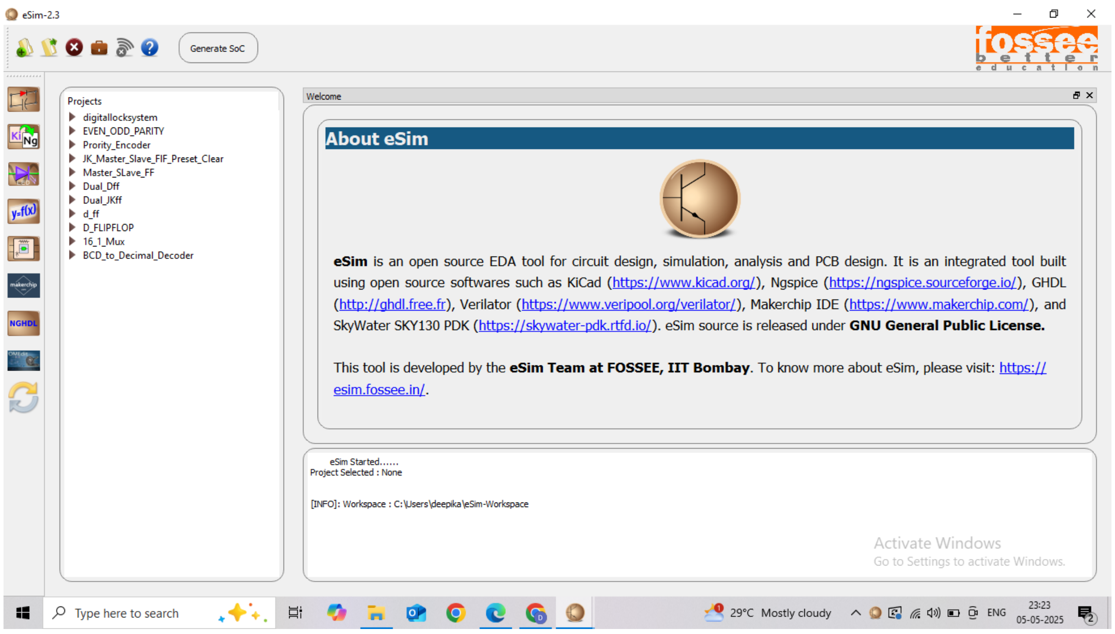
**Challenges Faced Limited Documentation in eSim** Advanced digital simulations lacked official guides or tutorials.This necessitated forum consultations, GitHub issue tracking, and exploring open-source code behavior.Through this, a better understanding of how Ngspice integrates with digital logic simulations was achieved.

**Component Library Constraints** eSim's built-in libraries did not cover all required components, especially for: Multi-bit adders. BCD to seven-segment decoders. JK and D flip-flops with edge-triggering.This required:Custom symbol creation in KiCad.SPICE model modification or creation to match intended functionality. This strengthened skills in customizing simulation environments and interpreting datasheets.

**Timing Issues in Sequential Circuits** Circuits like the Frequency Divider demanded precise clock synchronization.Challenges included: Race conditions due to improper flip-flop triggering. Misalignment between clock pulse widths and flip-flop response time.Addressing this involved:Tweaking PULSE generator parameters. Understanding the setup and hold time concepts.Simulating with delayed input signals to stabilize transitions.

## 1.5 eSim installation in Windows OS

1. Download eSim-2.0 install.exe from https://esim.fossee.in/downloads
2. Disable the antivirus (if any). Now, double click on the exe file to start the installation process. If a window appears, click Yes to complete the installation.
3. By default eSim will be installed in C drive, under an auto-generated FOSSEE Folder. Note that installation directory can neither be in ”Program Files” nor contain spaces in its path.
4. eSim icon will be created on desktop. You can double click on the eSim icon created on the Desktop after installation.



About eSim

**eSim** is an open source EDA tool for circuit design, simulation, analysis and PCB design. It is an integrated tool built using open source softwares such as KiCad (https://www.kicad.org/), Ngspice (https://ngspice.sourceforge.io/), GHDL (http://ghdl.free.fr), Verilator (https://www.veripool.org/verilator/), Makerchip IDE (https://www.makerchip.com/), and SkyWater SKY130 PDK (https://skywater-pdk.rtfd.io/). eSim source is released under **GNU General Public License.**

This tool is developed by the **eSim Team at FOSSEE, IIT Bombay**. To know more about eSim, please visit: https://esim.fossee.in/.

```
    eSim Started......
Project Selected : None

[INFO]: Workspace : C:\Users\deepika\eSim-Workspace
```

Figure 1.1: eSim Main GUI

# CHAPTER 2

# SCHEMATIC CREATION

The first step in the design of an electronic system is the design of its circuit. This circuit is usually created using a Schematic Editor and is called a Schematic. eSim uses Eeschema as its schematic editor. Eeschema is the schematic editor of KiCad. It is a powerful schematic editor software. It allows the creation and modification of components and symbol libraries and supports multiple hierarchical layers of printed circuit design.

## 2.1 Familiarizing the Schematic Editor interface

Fig. 2.1 shows the schematic editor and the various menu and toolbars. We will explain them briefly in this section.

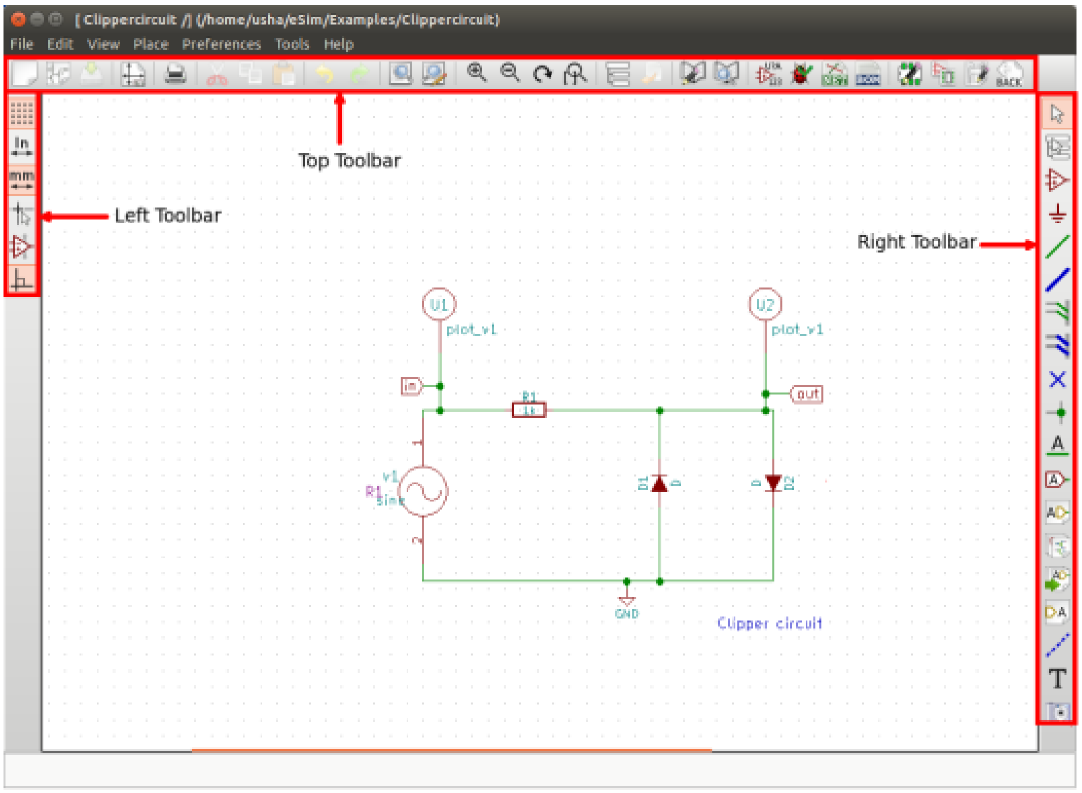

[ Clippercircuit /] (/home/usha/eSim/Examples/Clippercircuit)
File Edit View Place Preferences Tools Help
Top Toolbar
Left Toolbar
Right Toolbar
U1
plot_v1
U2
plot_v1
in
out
R1
v1
sine
D1
D2
GND
Clipper circuit

Figure 2.1: Schematic editor with the menu bar and toolbars marked

### 2.1.1 Top menu bar

File - The file menu items are given below:

1. New - Clear current schematic and start a new one
2. Open - Open a schematic
3. Open Recent - A list of recently opened files for loading
4. Save Schematic project - Save current sheet and all its hierarchy.
5. Save Current Sheet Only - Save current sheet, but not others in a hierarchy.
6. Save Current sheet as - Save current sheet with a new name.
7. Page Settings - Set preferences for printing the page.
8. Print - Access to print menu.
9. Plot - Plot the schematic in Postscript, HPGL, SVF or DXF format
10. Close - Close the schematic editor.

Place - The place menu has shortcuts for placing various items like components,wire and junction, on to the schematic editor window.Preferences - The preferences menu has the following options:

Component Libraries - Select component libraries and library paths. This enables the user to add the libraries, if the libraries are not loaded in the Eeschema.Schematic Editor Options - Select colors for various items, display options and set hot keys.Language - Shows the current list of available languages. Use default.Import and Export - Contain options to load and save preferences and import/ export hot key configuration files.

### 2.1.2 Top toolbar

Some of the important tools in the top toolbar are discussed below. They are marked in Fig. 2.2.

1. Save - Save the current schematic
2. Print - Print the schematic
3. Navigate schematic hierarchy - Navigate among the root and sub-sheets in the hierarchy

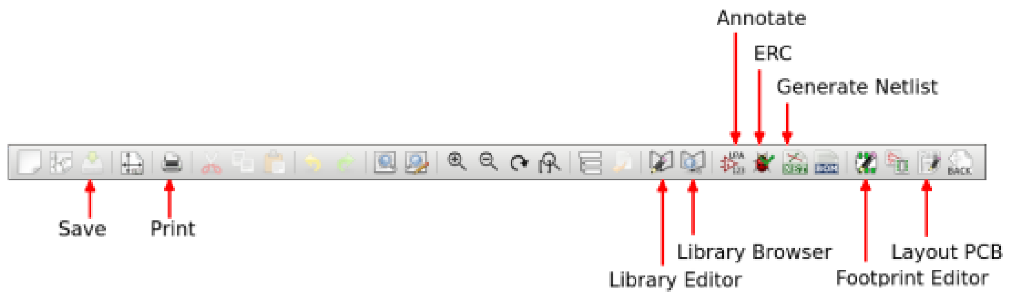


Figure 2.2: Toolbar on top with important tools marked

4. Library Editor - Create or edit components.
5. Library Browser - Browse through the various component libraries available
6. Annotate - Annotate the schematic
7. Check ERC - Do Electric Rules Check for the schematic
8. Generate Netlist - Generate a netlist for PCB design(.net) or for simulation(.cir).
9. Create BOM - Create a Bill of Materials of the schematic
10. Footprint editor - Map each component in the PCB netlist to a footprint
11. Layout PCB - Lay tracks between the footprints to get the PCB layout

### 2.1.3 Toolbar on the right

The toolbar on the right side of the schematic editor window has many important tools. Some of them are marked in Fig. 2.3. Let us now look at each of these tools and their uses. Place a component - Load a component to the schematic.Place a power port - Load a power port (Vcc, ground) to the schematic.Place wire - Draw wires to connect components in schematic.Place bus - Place a bus on the schematic.Place a no connect - Place a no connect flag, particularly useful in ICs.Place a local label - Place a label or node name which is local to the schematic.Place a global label - Place a global label (these are connected across all schematic diagrams in the hierarchy).Create a hierarchical sheet - Create a sub-sheet within the root sheet in the hierarchy. Hierarchical schematics is a good solution for big projects.Place a text or comment - Place a text or comment in the schematic.

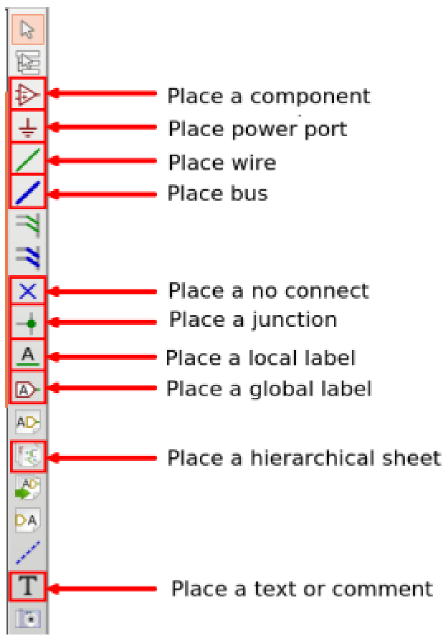


Figure 2.3: Toolbar on right with important tools marked

### 2.1.4 Toolbar on the left

Some of the important tools in the toolbar on the left are discussed below. They are marked in Fig. 2.4. Show/Hide grid - Show or Hide the grid in the schematic editor.

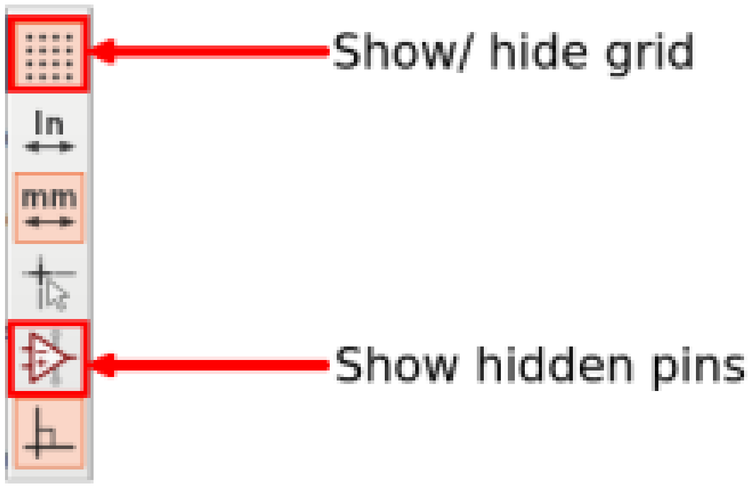


Figure 2.4: Toolbar on left with important tools marked

Pressing the tool again hides (shows) the grid if it was shown (hidden) earlier.Show hidden pins - Show hidden pins of certain components, for example, power pins of certain ICs.

## 2.2 Schematic creation for simulation

Opening the Schematic EditorOnce the project is created, eSim opens the schematic editor interface, which is based on KiCad. This is where you will place all the components and interconnect them to build your desired circuit. Click on the “Place Component” tool or press the shortcut key A to access the component library. From the library, search for and place all the necessary components, such as ICs like the 7442 decoder, voltage sources, toggle switches (for input), LEDs (for output indication), resistors, and the VCC and GND symbols for power connections.

Wiring the Circuit After placing the components, use the “Place Wire” tool (shortcut W) to draw wires and establish electrical connections between pins. Connect the inputs of the IC to switches or voltage sources, the outputs to LEDs (via resistors, if necessary), and ensure that the IC is properly powered by connecting the VCC and GND pins.

Annotating the Schematic Once wiring is complete, annotate the schematic using the “Annotate Schematic” option under the Tools menu. This automatically labels all the components in a standardized manner (e.g., U1 for IC, R1 for resistor, etc.).

Assigning SPICE Simulation Models For simulation purposes, it’s essential to assign appropriate SPICE models to the components. Right-click on each component and choose “Edit Spice Model” to attach a .model or .subckt file if the component is not natively supported. For commonly used ICs like the 7442, eSim often includes predefined models.

Generating the Netlist With the circuit ready, generate a netlist by navigating to “Tools” and selecting “Generate Netlist.” Choose the Ngspice format and save the netlist.Running the Simulation Then open the Ngspice simulation window provided by eSim. Here, define the type of simulation you wish to perform—typically a transient analysis such as .tran 0 50ms. If you’re using any custom or external models, remember to include the relevant .include or .lib statements.

Viewing Simulation Results To observe the behavior of your circuit, place plot probes (like plot_v) on the schematic at points of interest (such as input or output lines). Run the simulation, and eSim will display the corresponding waveforms.

# CHAPTER 3

# SUBCIRCUIT BUILDER

Subcircuit is a way to implement hierarchical modeling. Once a subcircuit for a component is created, it can be used in other circuits. eSim provides an easy way to create a subcircuit. The following Fig. 3.1 shows the window that is opened when the SubCircuit tool is chosen from the toolbar.

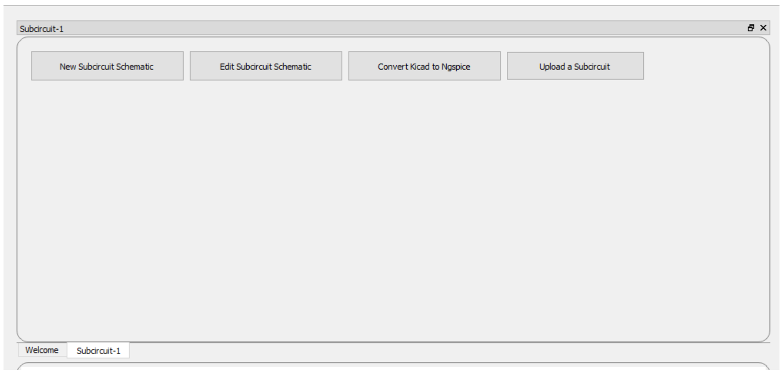


Figure 3.1: Subcircuit Window

## 3.1 Creating a SubCircuit

The steps to create subcircuit are as follows. After opening the Subcircuit tool, click on New Subcircuit Schematic button. It will ask the name of the subcircuit. Enter the name of subcircuit (without any spaces) and click OK as shown in Fig. 3.2.After clicking OK button it will open KiCad schematic. Draw your circuit which will be later used as a subcircuit. e.g the Fig. 3.3 shows the half adder circuit. Once you complete the circuit, assign a PORT to each open node of your circuit which will be used to connect with the main circuit. The port should match with the number of input and output pin. The circuit will look like Fig. 3.4 after adding PORT to it. The PORT component can be found in the eSim Miscellaneous library as shown in Fig. 3.5. Select a different port for each node (input or output), the PORT has 26 such components named alphabetically as Unit A, Unit B to Unit Z, meaning you can create a subcircuit

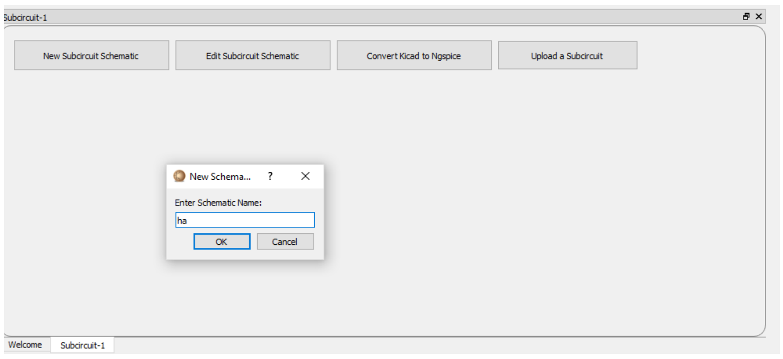


Figure 3.2: New Sub circuit Window

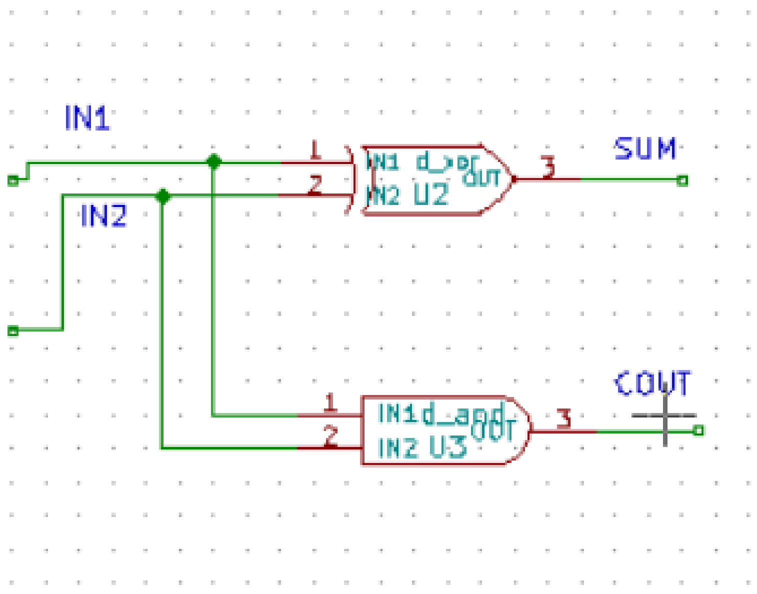


Figure 3.3: Inner circuit of the subcircuit

up-to 26 pins(input, output combined).Next step is to save the schematic and generate KiCad netlist.To use this subcircuit in other schematics, create a block in the schematic editor by following steps given below as one should have a symbol corresponding to the newly created subcircuit that can be used in other schematics: Go to library browser of the schematic editor. It is an ”open book with a pencil in its middle” icon on the top toolbar.Select the Current Library as eSim Subckt shown in Fig. 3.6 Click on create a new component from the top toolbar.Give the same name that was used for creating the new subcircuit’s internal diagram, refer Fig. 3.2.Choose designator as X. If any other reference designator other than X is used for subcircuit, your subcircuit will not be recognised during simulation.Similarly, reference designator are as follows for different types of components. D is for diode, Q is for transistors, J is for FET. The user needs to choose the appropriate reference based on the library in which they wish to add a model.Start drawing the subcircuit block by using the drawing tools from the right

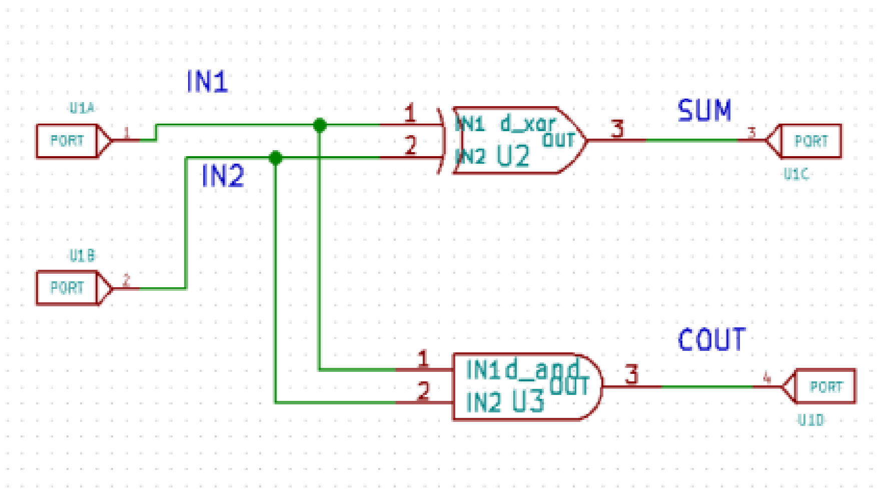


Figure 3.4: Half-Adder Subcircuit

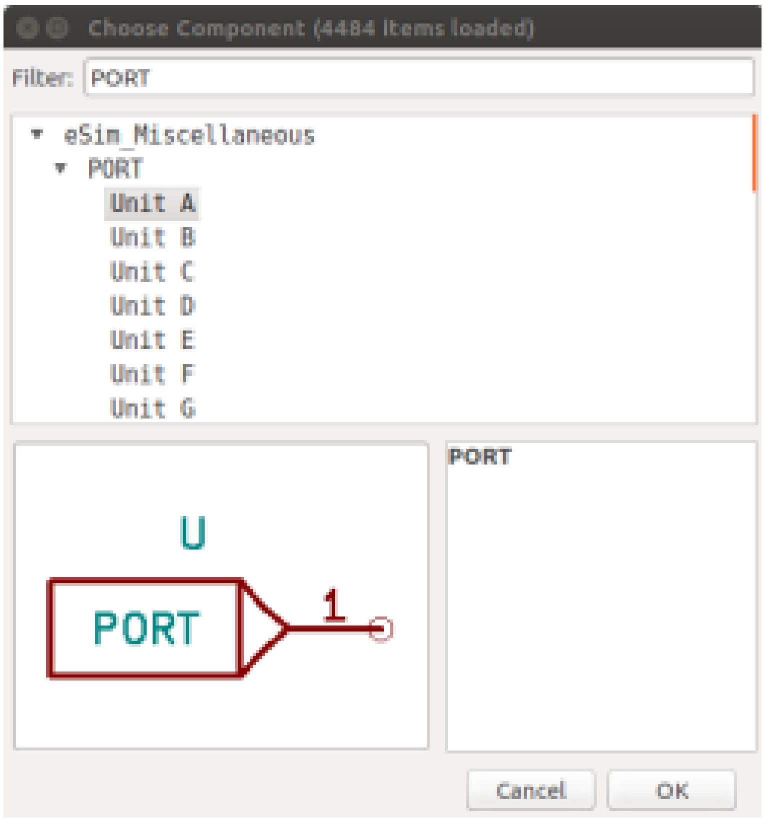
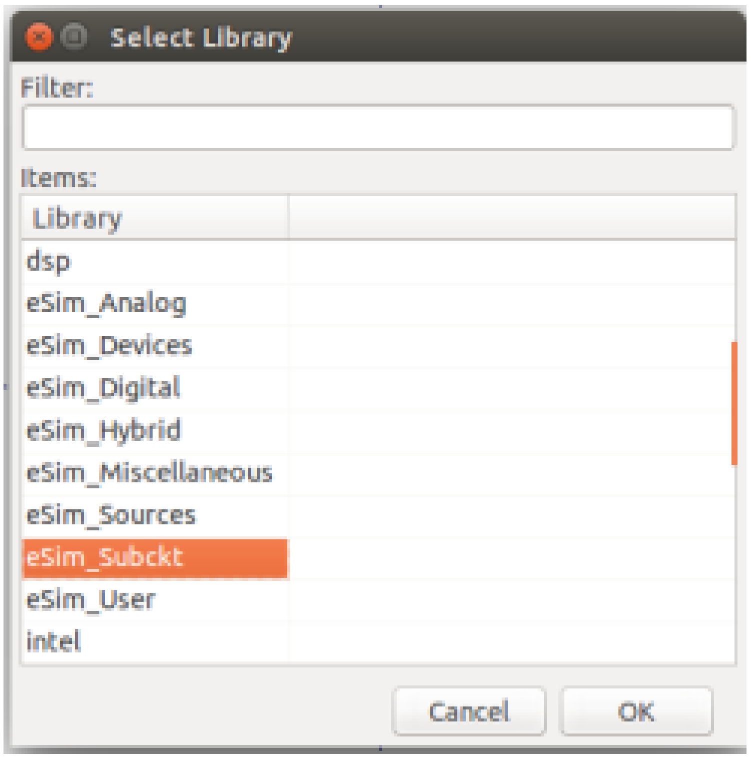


Figure 3.5: Selection of PORT component

taskbar. Here we have used Add graphic rectangle to component body. You can start drawing with a point to point click on the editor.

To add pins select Add pins to components from the right taskbar. Give the Pin Name as IN1 and Pin Number as 1. The pin number has to match with the Port name. Example Port A is mapped to pin 1. Select the Orientation as right or left accordingly. The Electrical Type has to be chosen as Input for nodes which will act as Input in the



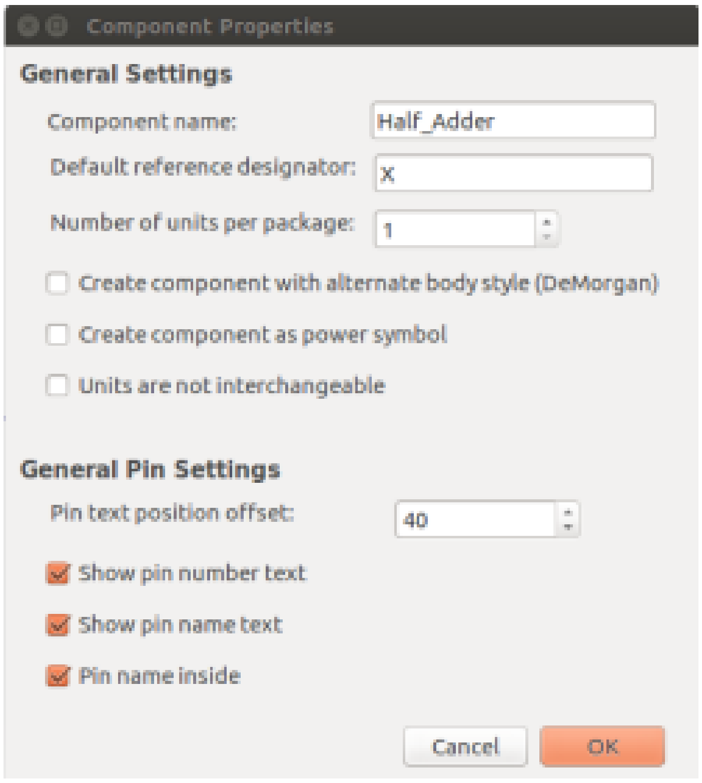
dsp
eSim_Analog
eSim_Devices
eSim_Digital
eSim_Hybrid
eSim_Miscellaneous
eSim_Sources
eSim_Subckt
eSim_User
intel

Figure 3.6: Selecting Working Library

subcircuit you are creating. Similar logic is for output nodes. We would recommend to declare the ports as either Input, Output or Passive. The final block of the subcircuit would look as shown in Fig. 8.8. Pins should be attached properly. Labels(Names to the PORTs) should be given such that it is intuitive and someone other than you should be able to understand and use that block with least amount of hassle. In order to save this file, press Ctrl+S keys and click yes for confirmation purposes.Note : A good practice to retain this created subcircuit would be to take a backup of this library. To do that, click on File from the library editor window and select the Save Current Library as option. A location needs to be selected, please select eSim-Workspace as the location for storing this file and give relevant name e.g. eSim-Subckt-backup. Later other users can use this in their circuits.

**Specifying parameters for generating the .sub file**

A .sub file is nothing but textual representation that is passed to the simulator which essentially informs the simulator about the nodes, and behavior of the subcircuit block. Remember the Fig. 3.4 circuit? It will be saved in a .sub file once we complete this process! Switch to the eSim main window and click on Convert KiCad to Ngspice button in the subcircuit builder tool. as shown in Fig. 3.1 You need not assign any


Component Properties

General Settings

Component name: Half_Adder

Default reference designator: X

Number of units per package: 1

☐ Create component with alternate body style (DeMorgan)

☐ Create component as power symbol

☐ Units are not interchangeable

General Pin Settings

Pin text position offset: 40

☑ Show pin number text

☑ Show pin name text

☑ Pin name inside

Figure 3.7: Creating New Component

values in the transient parameters section. Assign the values to any voltage or current sources present in your internal circuit, if any.

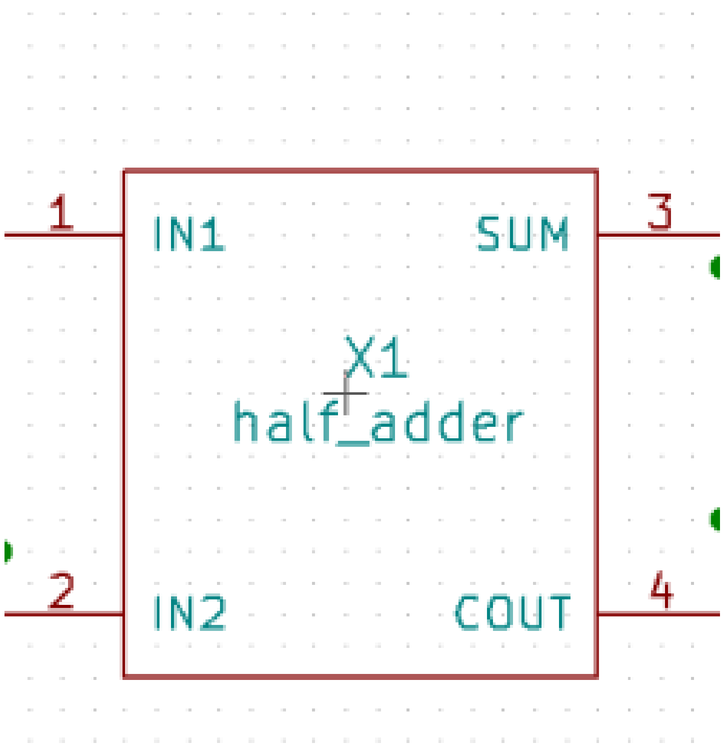


Figure 3.8: Half-Adder Subcircuit Block

## 3.2 Edit a Subcircuit

The steps to edit a subcircuit are as follows After launching the Subcircuit tool, click on Edit Subcircuit Schematic button. It will open a dialog box where you can select any subcircuit for editing.After selecting the subcircuit it will open it in the schematic editor, where you can edit the subcircuit.Next step is to save the schematic and generate the .cir netlist.If you have edited the number of ports then you have to change the block exaplained in section Creating a Subcircuit accordingly.

**NOTE**

User can also import or append the schematic of different projects in the current page using the Append Schematic Sheet from the File menu. This will import(copy) the schematic that user has defined to the current schematic editor page.User can also import the model in the part library editor page using the option Import Component from the top toolbar.

## 3.3 Upload subcircuit

Using this feature, one can import an existing subcircuit file into eSim environment. You necessarily need not create the schematic for this.Download the required subcircuit's .sub file from many online resources/repositories.Upload this file using the upload subcircuit feature.Upon uploading following checks will be made, and only and only if the checks are satisfied, the file will be uploaded. The checks are as following :The uploaded file should have the extension .sub The name of the file, say for example is omega.sub, then the content of the file must start with .subckt omega and end with .ends omega. Any line that starts with asterisk sign(*) is considered as a comment in these types of files.

Hence, the file technically starts with .subckt.If above conditions are satisfied, then the file will be automatically placed in a folder that carries the same name as that of the .sub file will be created in ../SubcircuitLibrary/ directory.Once above steps are verified, proceed to create a block as shown in Fig. 3.8 and name should be same as that of the corresponding .sub file uploaded earlier. Pins of this block should match the number of pins stated in the .sub file.

# CHAPTER 4

# IC-BASED DESIGNS

## 4.1 IC 78HC280

The 74HC280-Q100; 74HCT280-Q100 is a 9-bit parity generator or checker. Both even and odd parity outputs are available. The even parity output (PE) is HIGH when an even number of data inputs (I0 to I8) is HIGH. The odd parity output (PO) is HIGH when an odd number of data inputs are HIGH. Expansion to larger word sizes is accomplished by tying the even outputs (PE) of up to nine parallel devices to the final stage data inputs. Inputs include clamp diodes. It enables the use of current limiting resistors to interface inputs to voltages in excess of VCC.

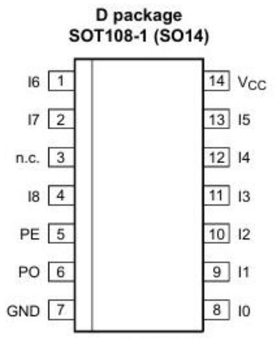


Figure 4.1: 74HC280 9-bit Even/Odd Parity Generator/Checker IC used for parity checking applications in digital systems.

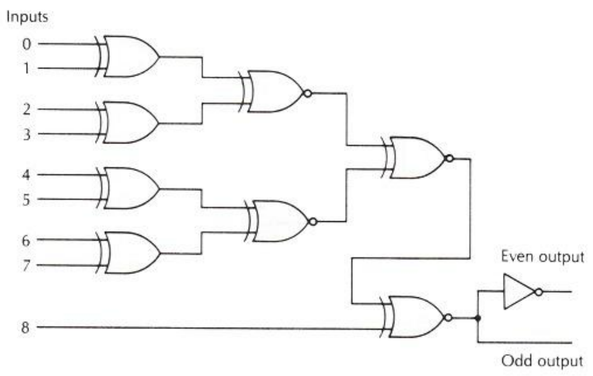


Figure 4.2: Logic diagram of a 9-bit parity checker circuit using XOR and NOT gates to generate even and odd parity outputs.

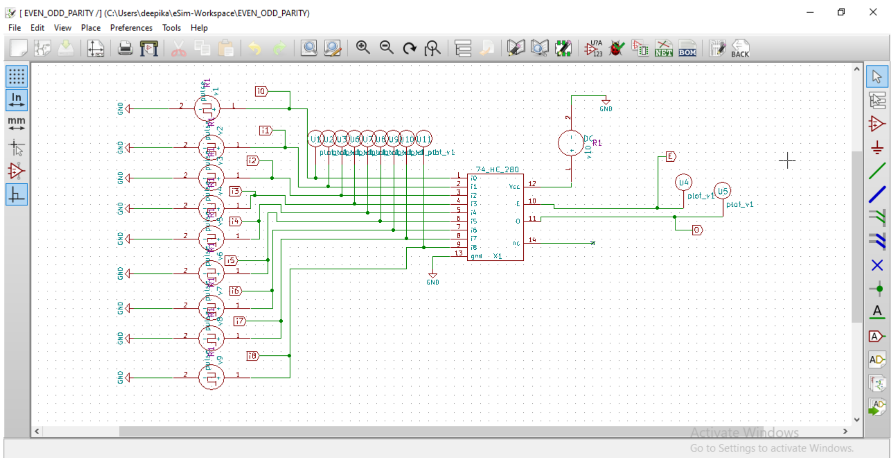

[ EVEN_ODD_PARITY /] (C:\Users\deepika\eSim-Workspace\EVEN_ODD_PARITY)
File Edit View Place Preferences Tools Help
74_HC_280
GND
DC
R1
plot_v1
Activate Windows
Go to Settings to activate Windows.

Figure 4.3: Even-Odd Parity Checker Circuit designed in eSim using 8-bit input switches and logic blocks for parity detection.

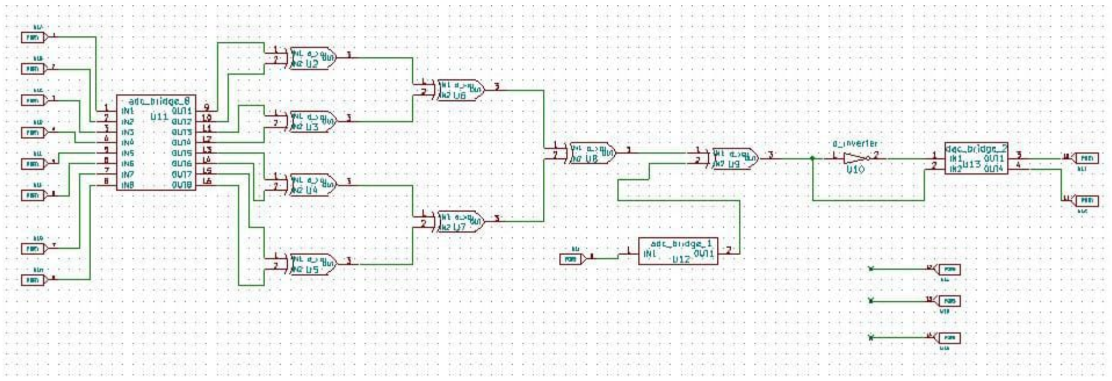


Figure 4.4: Internal subcircuit of Even-Odd Parity Checker using XOR gates to determine parity logic in eSim.

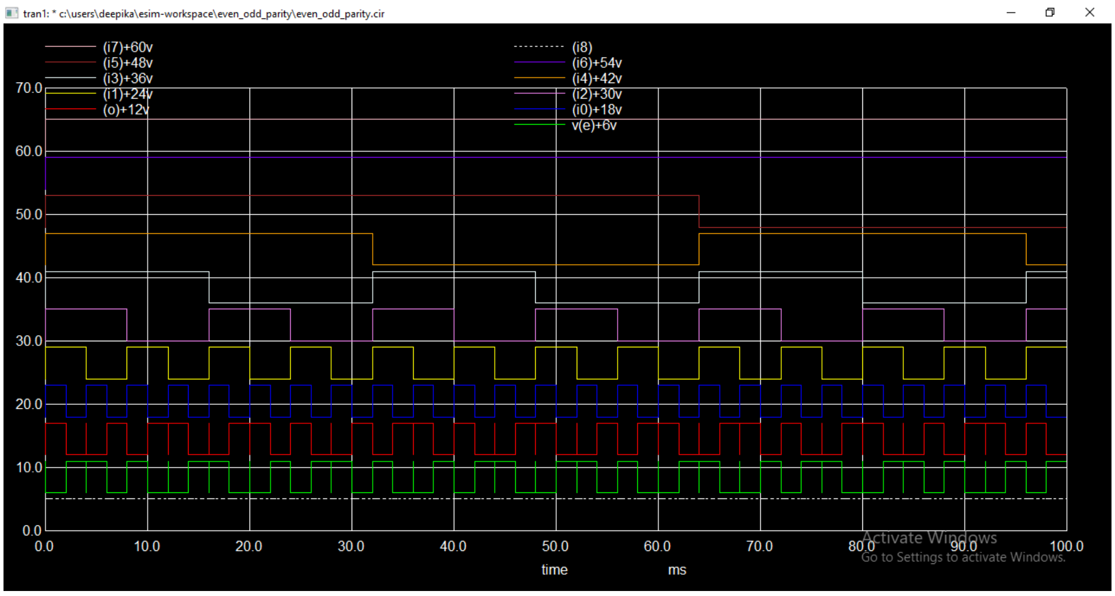


Figure 4.5: Timing diagram of 9-bit Even-Odd Parity Checker showing input waveforms and corresponding parity output response over time.

## 4.2 IC 74LS48

The 74LS48 is a BCD (Binary-Coded Decimal) to 7-segment decoder/driver, designed to convert 4-bit BCD inputs into control signals for 7-segment dis plays. This IC is commonly used in digital systems to display numerical data. BCD Input Handling: Accepts 4-bit BCD inputs (labelled A, B, C, D) and decodes them to drive a 7-segment display.Active Low Outputs: Out puts are designed to drive common-anode displays, meaning a low output (logic 0) turns on the corresponding segment.

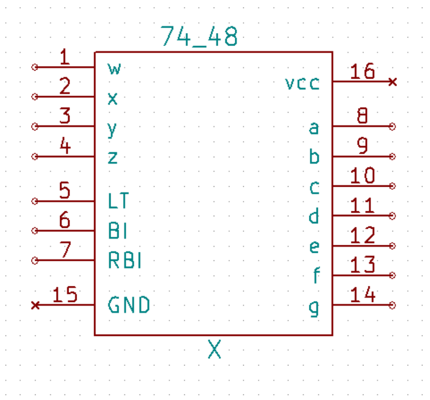


Figure 4.6: Pin configuration of 7448 BCD to 7-segment decoder/driver IC.

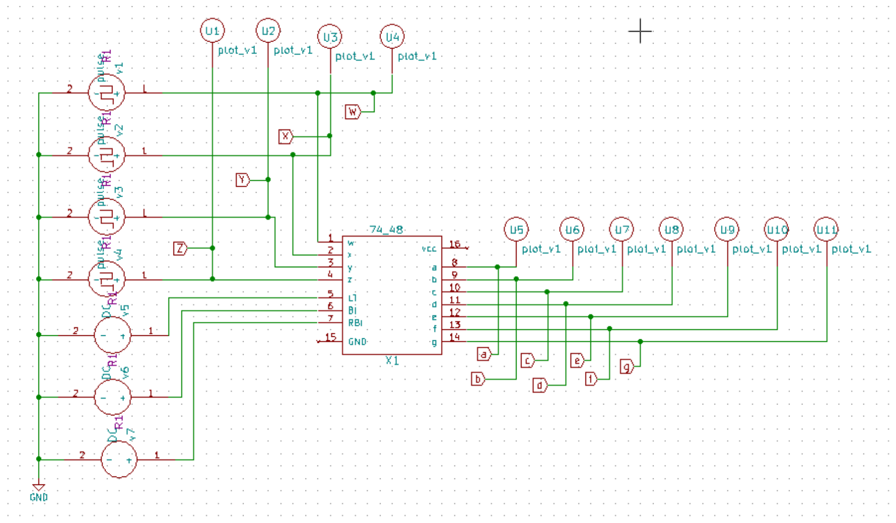


Figure 4.7: Logic gate-level circuit implementation of a 7-segment display system.

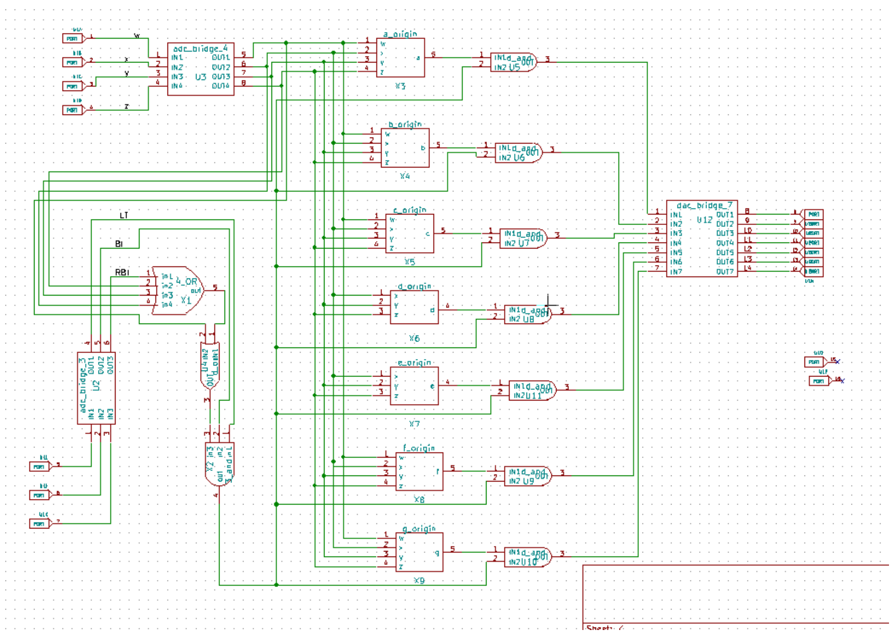


Figure 4.8: Schematic diagram of BCD to 7-segment decoder using 7448 IC.

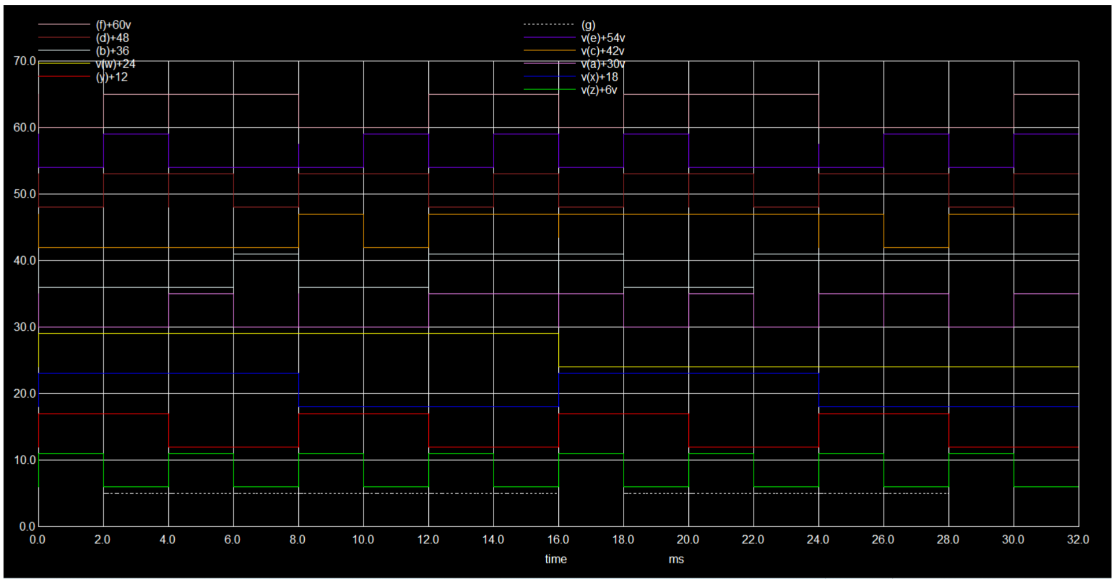


Figure 4.9: Timing waveform analysis of BCD inputs and corresponding 7-segment outputs.

The 7448 is a BCD to 7-Segment Decoder/Driver IC. It accepts a 4-bit Binary Coded Decimal (BCD) input and converts it into the corresponding output to drive a common cathode 7-segment display. It includes active-low outputs and features blanking input (BI), lamp test (LT), and ripple-blanking input/output (RBI/RBO) functions for additional display control.

## 4.3 IC 74LS42

These monolithic BCD-to-decimal decoders consist of eight inverters and ten four-input NAND gates. The inverters are connected in pairs to make BCD input data available for decoding by the NAND gates. Full decoding of valid input logic ensures that all outputs remain off for all invalid input conditions.

The ’42A and "LS42 feature inputs and outputs that are compatible for use with most TTL and other saturated low-level logic circuits. DC noise margins are typically one volt.

The SN5442A and SN54LS42 are characterized for operation over the full military temperature range of 55°C to 125°C. The SN7442A and SN74LS42 are characterized for operation from 0°C to 70°C.

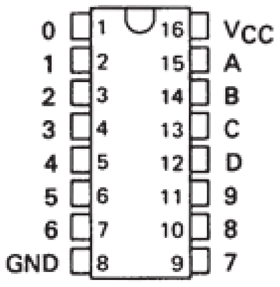


Figure 4.10: Pin configuration of 7442 BCD to Decimal Decoder IC.

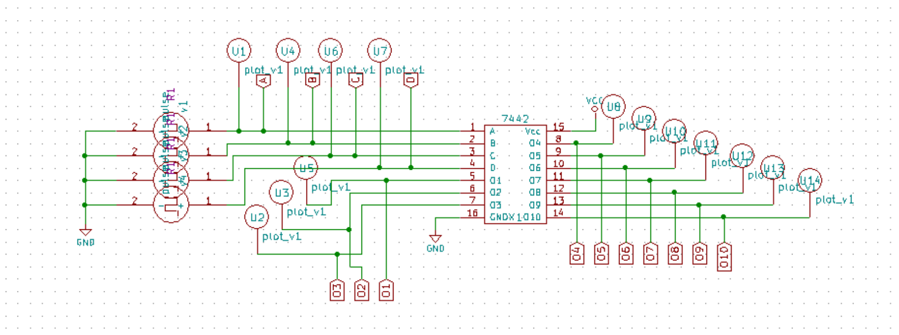


Figure 4.11: Logic-level circuit implementation of BCD to Decimal decoding using 7442.

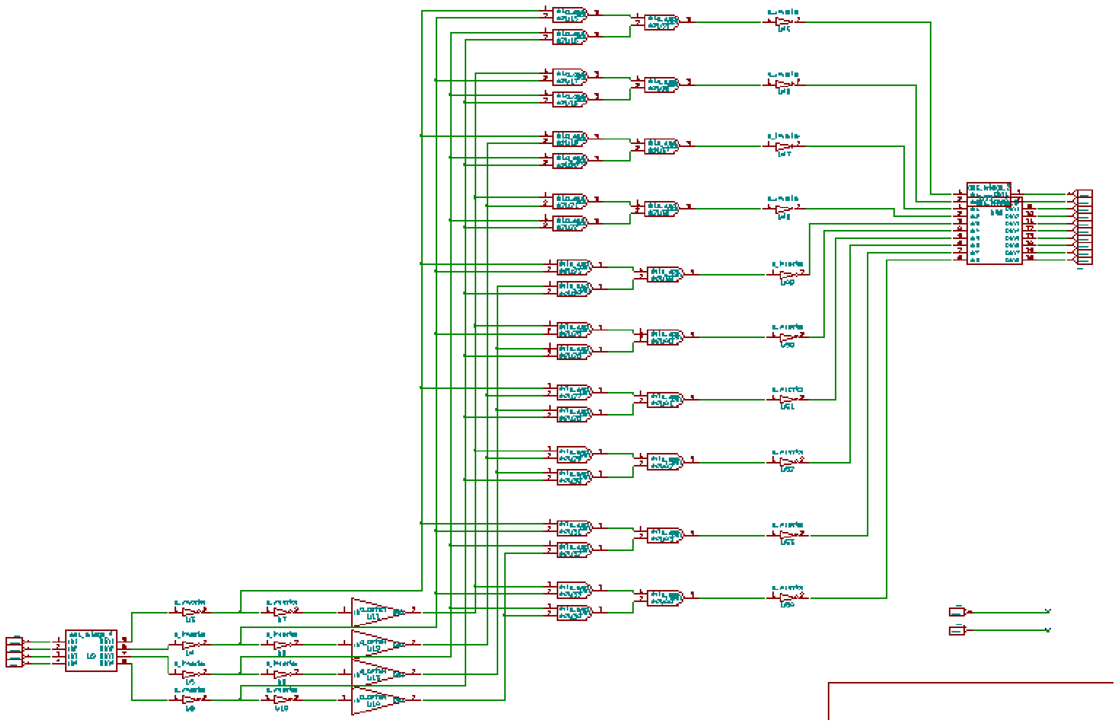


Figure 4.12: Internal Subcircuit of BCD to Decimal decoding using 7442

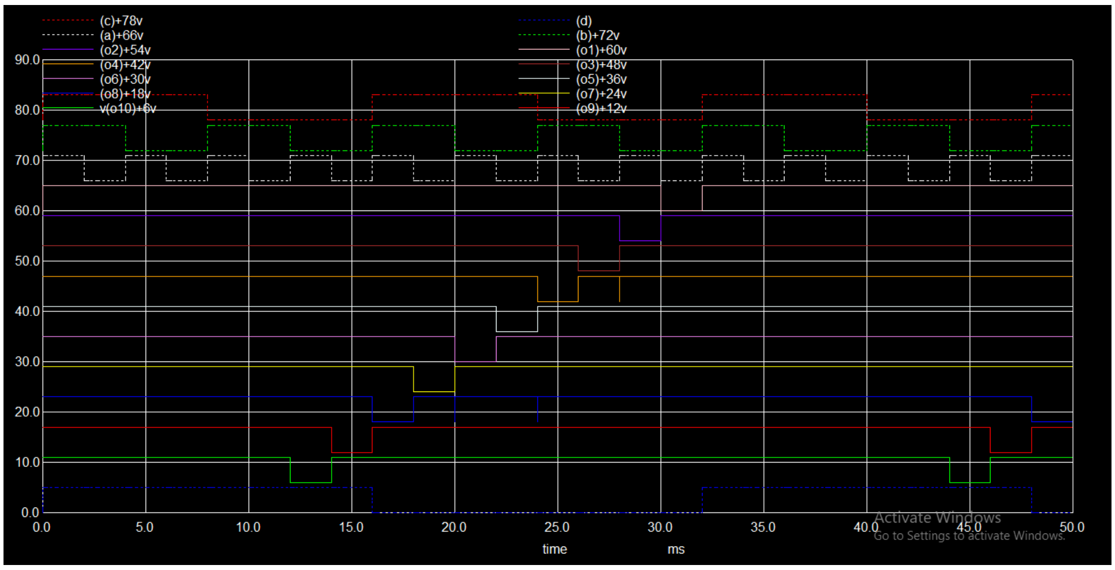


Figure 4.13: Timing waveform analysis of BCD inputs and corresponding decoded outputs of 7442.

## 4.4 IC DM74150

These data selectors/multiplexers contain full on-chip decoding to select the desired data source. The DM74150 selects one-of-sixteen data sources. The DM74150 has a strobe input which must be at a LOW logic level to enable these devices. A HIGH level at the strobe forces the W out put HIGH and the Y output (as applicable) LOW. The

DM74150 features an inverted (W) output only.

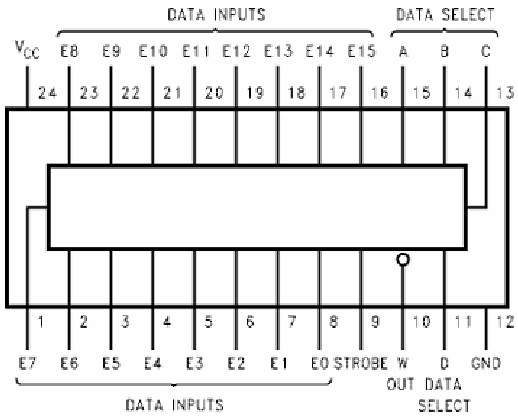


Figure 4.14: Pindiagram of DM74150

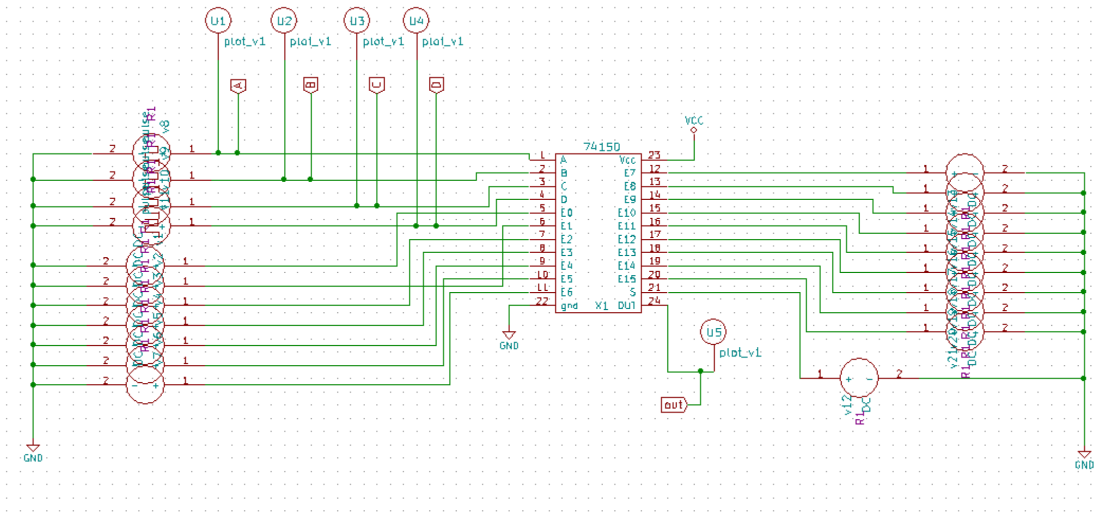


Figure 4.15: Main Circuit implementation of DM74150

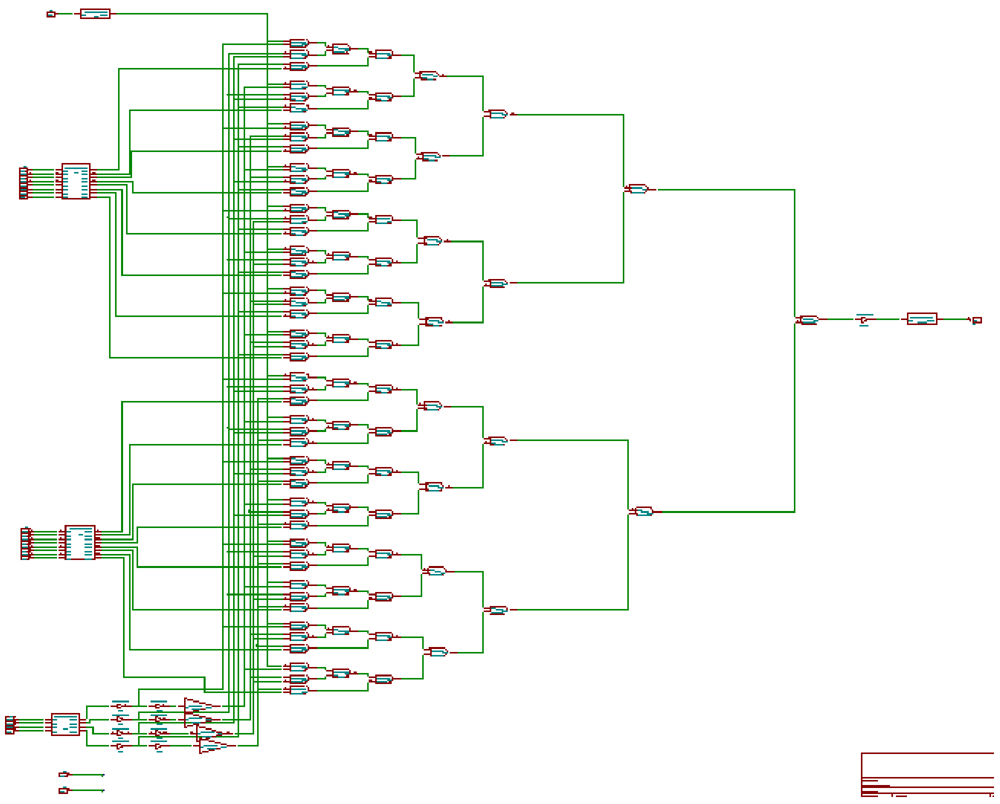


Figure 4.16: Internal Subcircuit of DM74150

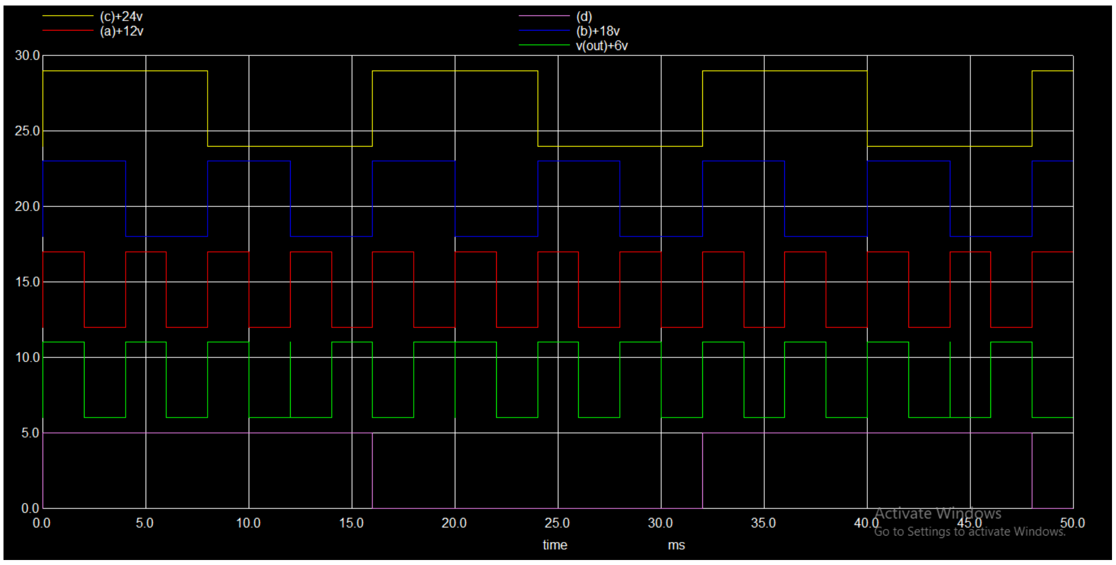


Figure 4.17: Timing diagram of DM74150

# CHAPTER 5

# CONCLUSION

In conclusion, this project has demonstrated the potential and versatil ity of eSIM technology in modern communication systems. Through detailed analysis, design, and implementation, we have explored how eSIMs can sim plify device connectivity, enhance security, and support the growing demand for seamless global mobile access. Unlike traditional SIM cards, eSIMs of fer flexibility, remote provisioning, and better integration with IoT devices and smartphones. Our findings confirm that eSIM adoption can significantly improve user experience, reduce logistical challenges, and pave the way for smarter, more efficient networked ecosystems. As eSIM technology continues to evolve, it is poised to become a core component of future communication infrastructures.

## REFERENCES

[1] https://static.fossee.in/esim/installation-files/eSim_Manual_2.0.pdf

[2] https://assets.nexperia.com/documents/data-sheet/74HC_HCT280_Q100.pdf

[3] https://www.scribd.com/document/401388520/Frequency-Divider-D-Flip-flops-docx

[4] https://onlinecourses.swayam2.ac.in/aic20_sp59/preview

[5] https://spoken-tutorial.org/tutorial-search/?search_foss=eSim&search_language=English

[6] https://esim.fossee.in/downloads/tutorials

[7] https://www.ti.com/lit/ds/symlink/cd74ac283.pdf?ts=1744054257069&ref_url=https%253A%252F%252Fwww.google.com%252F

[8] https://www.ti.com/lit/ds/symlink/sn54ls47.pdf?ts=1744259397102&ref_url=https%253A%252F%252Fwww.google.com%252F

[9] https://www.ti.com/lit/ds/sdls109/sdls109.pdf

[10] https://www.ti.com/lit/ds/symlink/sn54s151.pdf?ts=1746457418774ref_url=https%253A%252F%252Fwww.google.com%252F