4- BIT ARITHMETIC LOGIC UNIT

BECE303L VLSI SYSTEM DESIGN

By

Renumanjari R K - 22BEC1003

Arvindhan K – 22BEC1026

Sherlin Immanuela R - 22BEC1138

Vijay Aditya S – 22bec1147

Teekshitta Ramakrishnan – 22BEC1261

Submitted to

Dr. Umadevi S

SCHOOL OF ELECTRONICS ENGINEERING

VELLORE INSTITUTE OF TECHNOLOGY

CHENNAI - 600127

November 2024

CERTIFICATE

This is to certify that the Project work titled "4 Bit Arithmetic Logic Unit" is being submitted by Renumanjari R K - 22BEC1003, Arvindhan K - 22BEC1026, Sherlin Immanuela R - 22BEC1138, Vijay Aditya S – 22BEC1147 & Teekshitta Ramakrishnan – 22BEC1261 for the course BECE303L VLSI System Design is a record of bonafide work done under my guidance. The contents of this project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University.

Dr. Umadevi S
Associate Professor
School of Electronics Engineering (SENSE),
VIT University, Chennai
Chennai – 600 127.

ABSTRACT

A 4-bit ALU (Arithmetic Logic Unit) is a digital circuit that performs arithmetic and logical operations on 4-bit binary numbers. It’s a core component of many digital systems, including microprocessors and microcontrollers, where it executes fundamental calculations and data manipulations. Designing a 4-bit ALU with less delay and by using minimum hardware plays an important role in the overall performance of the circuit. It ensures quick execution of the results.

A modular circuit is designed in Cadence Virtuoso using 4:1 and 2:1 multiplexer to support both arithmetic and logical operations. The circuit takes two 4-bit input operands, A and B, and uses a 2-bit control line S1 and S0 to select specific operations, along with a 1-bit mode select line C to toggle between logical and arithmetic outputs. This approach leverages the modularity of multiplexers to minimize the circuit complexity while maintaining the flexibility to perform various logical and arithmetic operations based on control inputs. By managing the logical and arithmetic operations within the multiplexer setup, we achieve an efficient design suitable for basic 4-bit processing requirements.

ACKNOWLEDGEMENT

We extend our heartfelt gratitude to Dr. Umadevi ma’am for their invaluable guidance, unwavering support, and mentorship throughout the duration of this mini group project. Their expertise and encouragement have been instrumental in shaping the project's direction and ensuring its successful execution.

We would also like to acknowledge the exceptional contributions of our dedicated team members, whose collaborative efforts, diverse skills, and commitment have significantly enriched the project's outcomes.

Furthermore, we express our appreciation to the technical staff and laboratory personnel for their indispensable assistance in providing the necessary resources, equipment, and infrastructure essential for the practical implementation of the project.

Lastly, we are thankful to the academic institution for fostering an environment that encourages experiential learning, innovation, and the practical application of theoretical knowledge.

The successful completion of this project has been made possible through the collective support, guidance, and contributions of all those mentioned above, and for that, we are truly grateful.

NAME WITH SIGNATURE NAME WITH SIGNATURE

INTRODUCTION

A 4-bit Arithmetic Logic Unit (ALU) is a fundamental digital circuit designed to perform basic arithmetic and logical operations on 4-bit binary inputs. Commonly found in microprocessors, microcontrollers, and other digital systems, a 4-bit ALU enables essential data processing tasks such as addition, subtraction, divisor and bitwise operations (AND, OR, XOR). These operations are crucial in executing instructions and making logical decisions in digital applications.

Optimizing a 4-bit ALU for speed and minimal hardware usage is important, as it impacts the performance, efficiency, and power consumption of the system. In this design, a modular approach is employed, using multiplexers to create a versatile and compact 4-bit ALU capable of quickly switching between arithmetic and logical functions. Through a structured control mechanism, the ALU dynamically selects the desired operation, making it a compact and effective solution for fundamental 4-bit processing needs in digital systems.

Early designs on ALU laid the groundwork for digital computation but often involved extensive hardware, which led to increased delay and power consumption. As digital systems became more complex, researchers focused on optimizing ALU designs to improve performance and reduce power consumption. One common approach was to streamline hardware by using multiplexers to select specific operations based on control inputs, rather than employing separate circuits for each operation. This not only reduced the hardware footprint but also helped minimize propagation delay, making ALUs faster and more efficient.

LITERATURE SURVEY

Arithmetic Logic Units (ALUs) are integral to digital systems, acting as the computational heart of microprocessors. Researchers have explored various designs and implementations to enhance their functionality, efficiency, and adaptability.

Sarangi (2020) presents a VHDL-based implementation of a 4-bit ALU, showcasing its combinational logic design. The ALU performs 12 operations—seven arithmetic and four logical—using a mixed VHDL model tested with Xilinx ISE. The paper emphasizes modular design by cascading identical stages, each controlled by six inputs. Through controlled data inputs to a parallel adder, various arithmetic and logical operations are achieved, underscoring the flexibility of combinational logic.

Yadav (2021) focuses on optimizing power consumption in 4-bit ALUs while ensuring high-speed operations. By integrating parallel computation and carry-lookahead circuits, her design achieves efficient computation of 16 arithmetic and 16 logical operations. This approach leverages a MUX for output selection and uses auxiliary signals like Carry Propagate (P) and Carry Generate (G) for enhanced speed and accuracy, particularly in ripple and carry-lookahead modes.

Zhao (2022) introduces a design with six operations, including addition, 2's complement, and NAND/NOR logic. Unique design blocks, such as 3-to-6 decoders, 4x2-bit AND gates, and 6:1 multiplexers, are used to simplify hardware costs while maintaining functionality. The modular architecture ensures efficient operation selection and minimal resource utilization.

METHODOLOGY

1 Specification of Operations

The ALU was designed to perform the following operations based on the input control signals:

- Logical Operations: OR, AND, XOR
- Arithmetic Operations: Addition, Subtraction, and Division

These operations are selected using a combination of control signals:

- **S1, S0 (2-bit control lines)**: These control the selection of the operation. The operations corresponding to different values of S1 and S0 are:

S1	S0	Logical Process	Arithmetic Process
0	0	OR	ADDITION
0	1	AND	SUBTRACTION
1	0	XOR	DIVISOR

Table 1

C (1-bit mode select): This control signal determines whether the operation is logical or arithmetic.

C	Operation
0	Arithmetic
1	Logical

Table 2

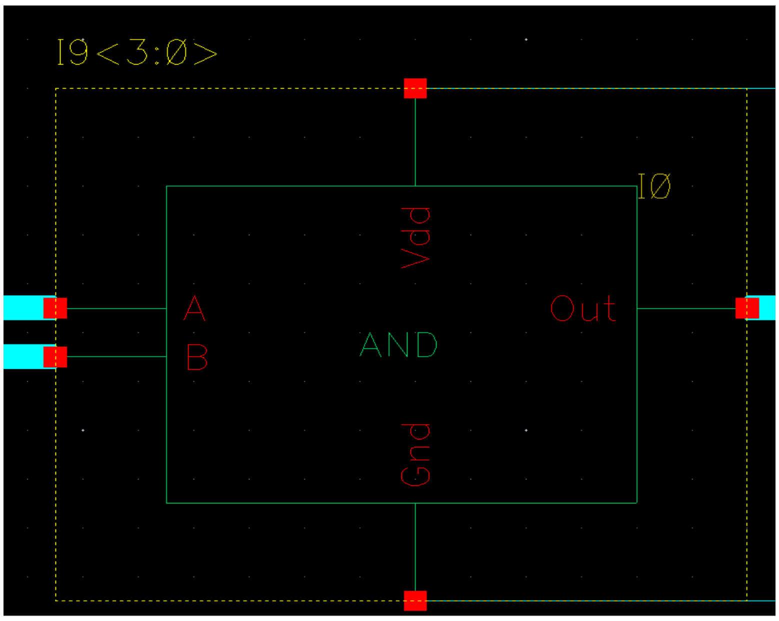
2 Selection of Components

To implement the ALU, multiplexers were chosen as the core components due to their simplicity, versatility, and ability to select between multiple inputs based on control signals. The following components were selected:

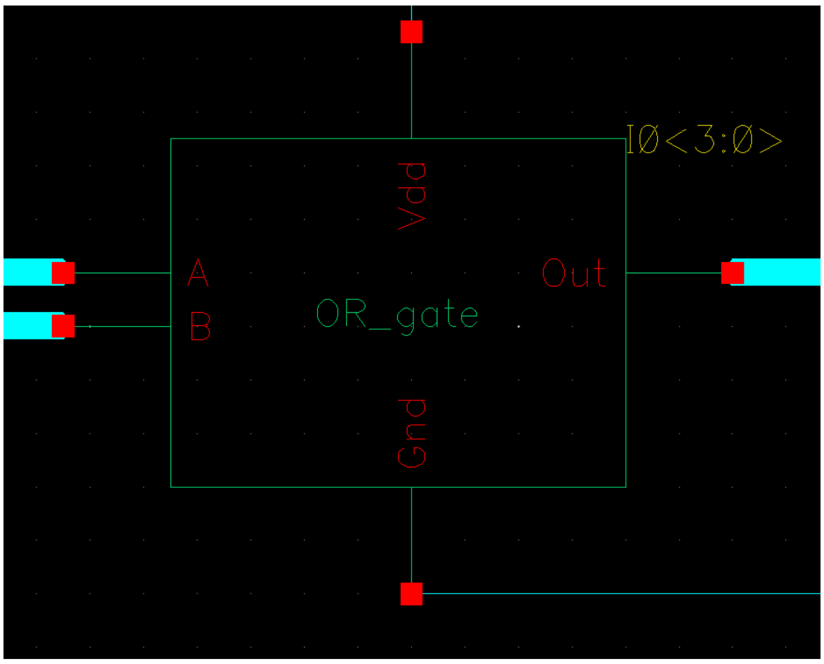
- **4:1 Multiplexer**: Used to select one of the four operations (OR, AND, XOR, ADD/SUB/DIV) based on the 2-bit control input (S1, S0).
- **2:1 Multiplexer:** Used to choose between arithmetic or logical operations based on the mode select signal (C).

3 Circuit Design in Cadence Virtuoso

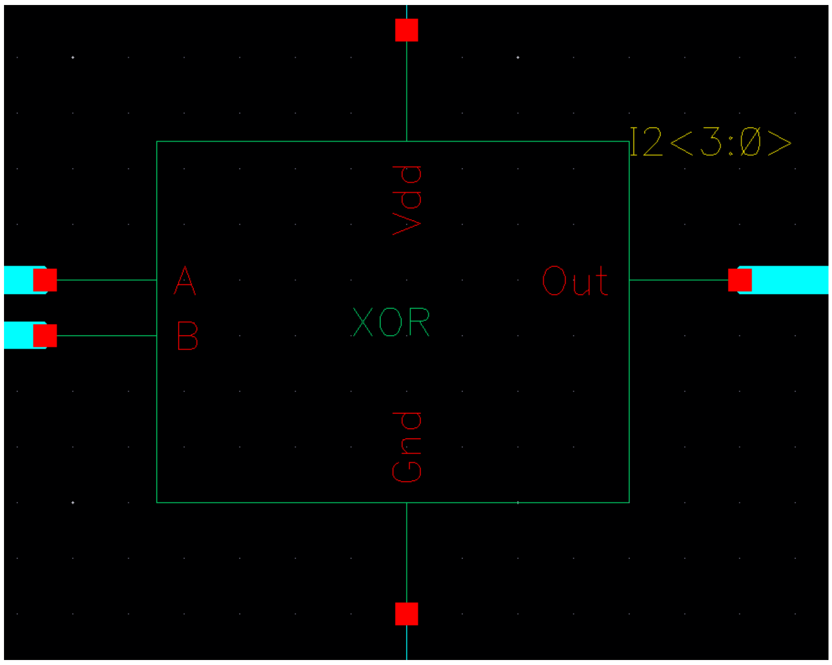
AND Gate



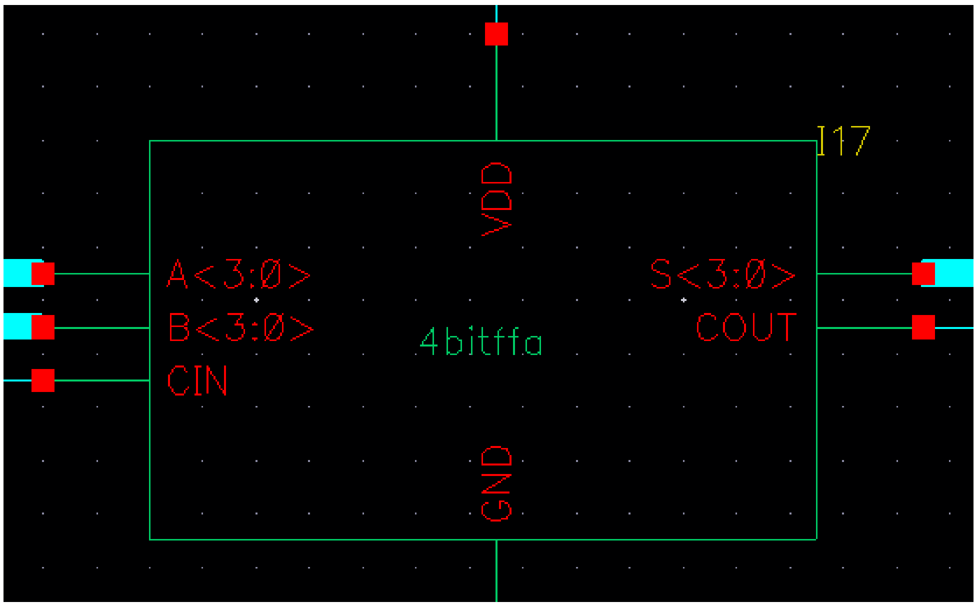
OR Gate



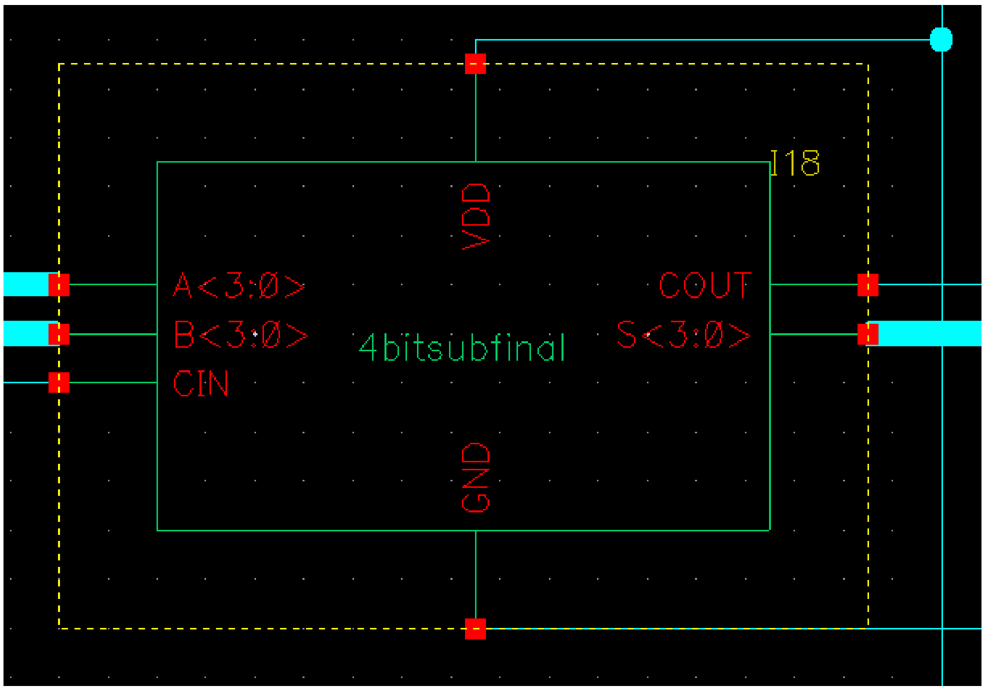
XOR Gate



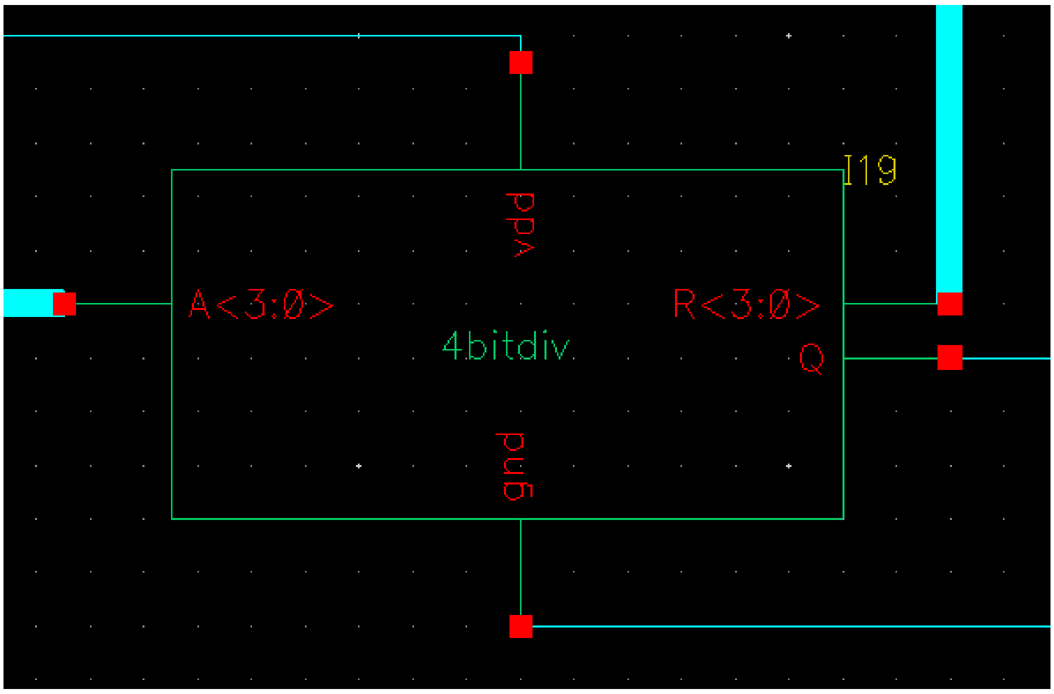
Addition



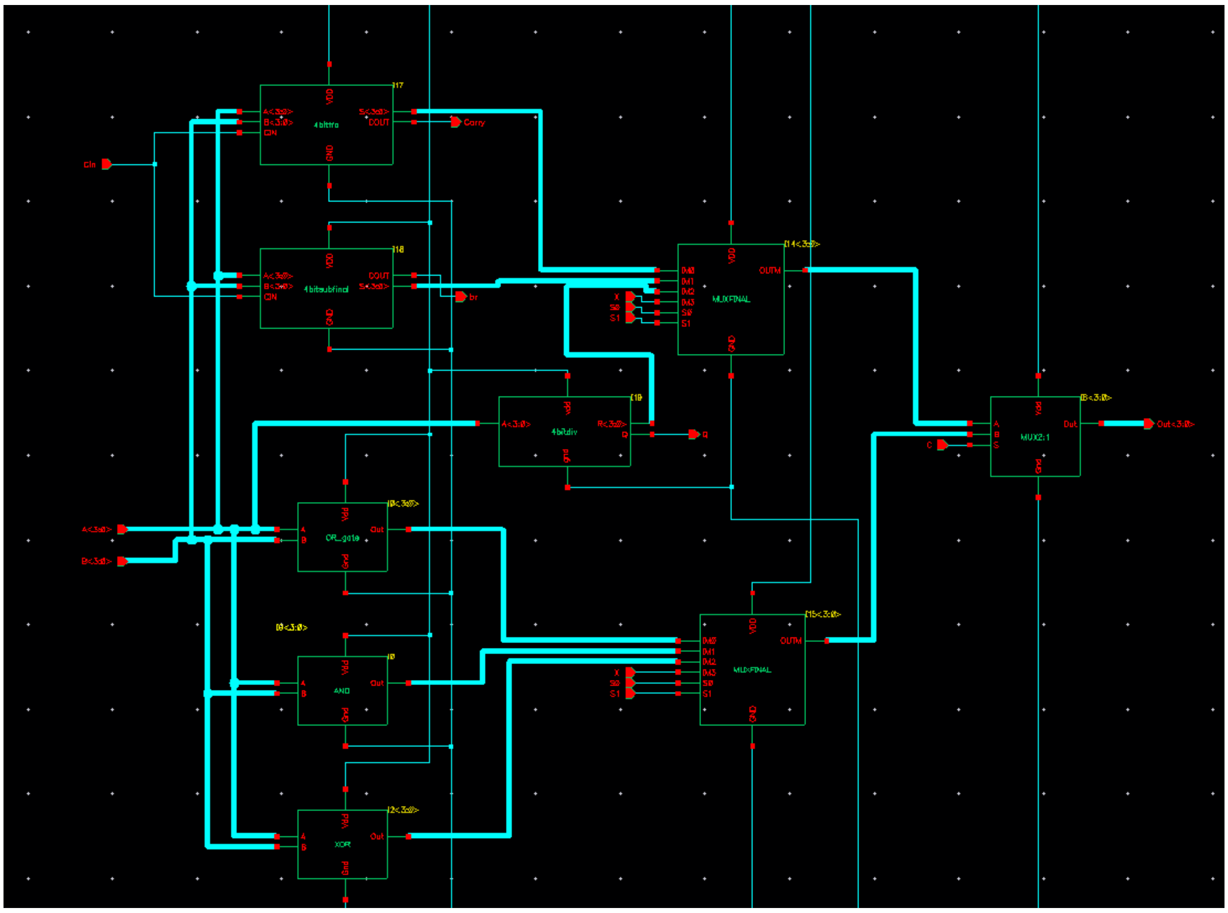
Subtraction

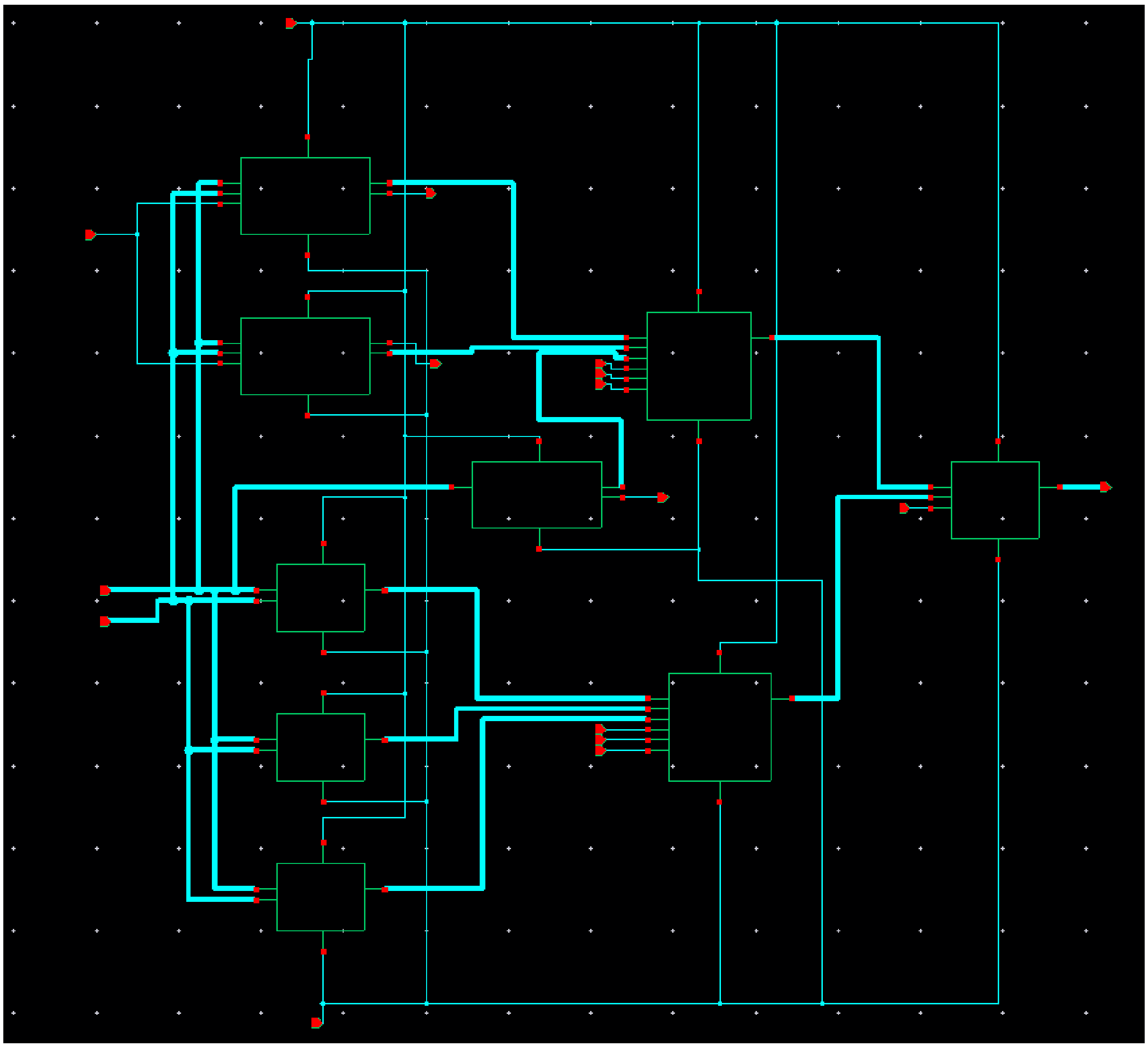


Divisor



Full Schematic





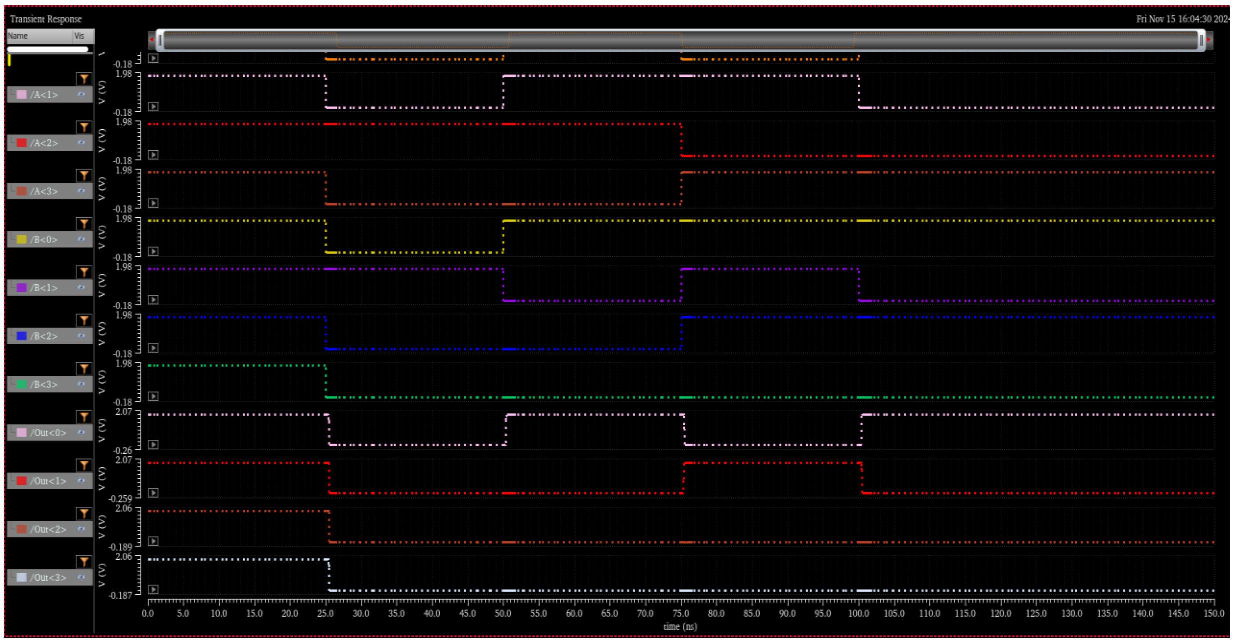
RESULTS

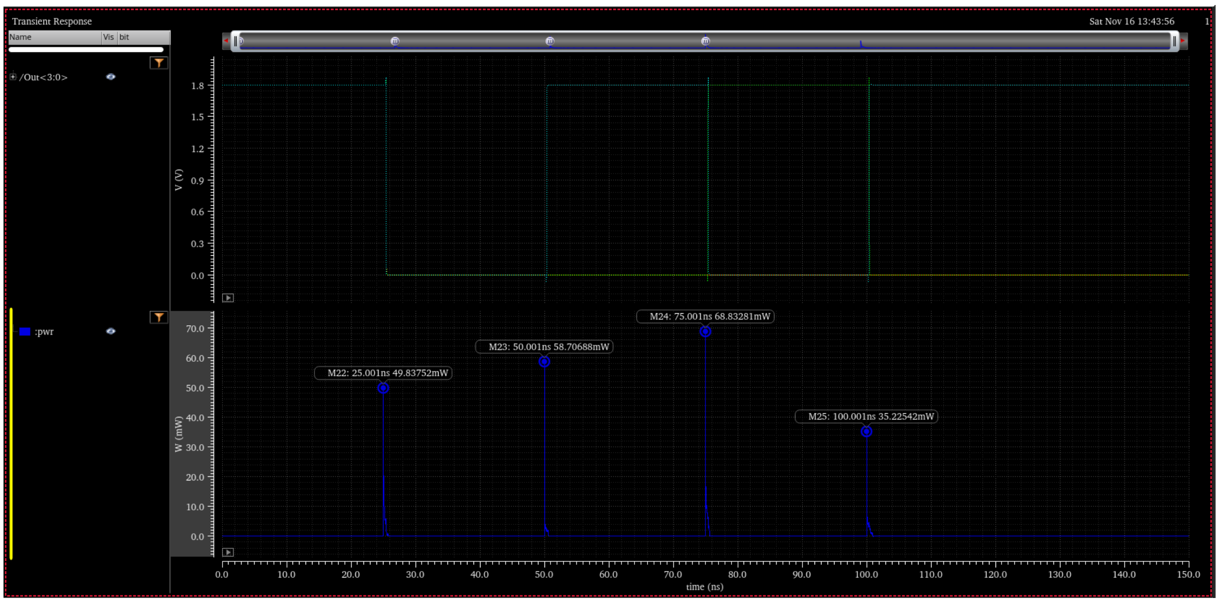
Truth Table:

A	A3	A2	A1	A0
	10011	**11100**	**10110**	**10101**
B	**B3**	**B2**	**B1**	**B0**
	10000	**10011**	**11010**	**10111**
AND	10000	10000	10010	10101
OR	10011	11111	11110	10111
XOR	00011	01111	01100	00010
ADD	10101	11001	11001	00010
SUB	00000	00101	01110	00010
DIV	00001	11100	00100	10101

Logical Unit of ALU:

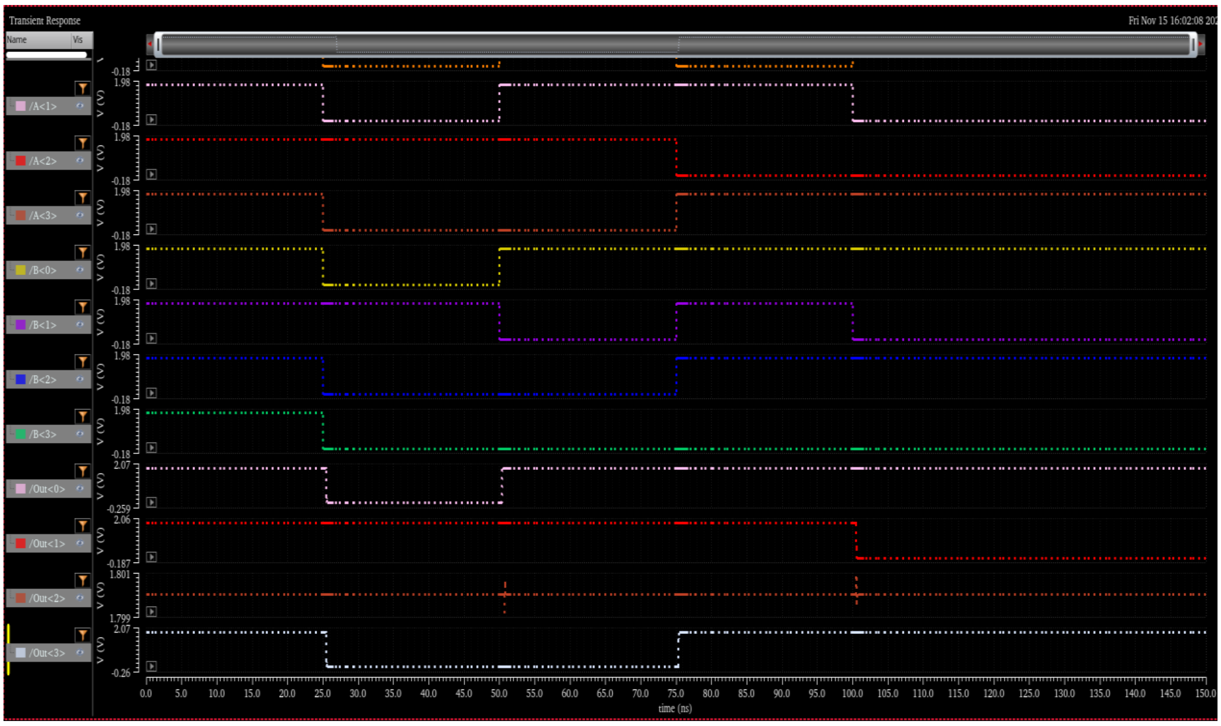
AND Gate:



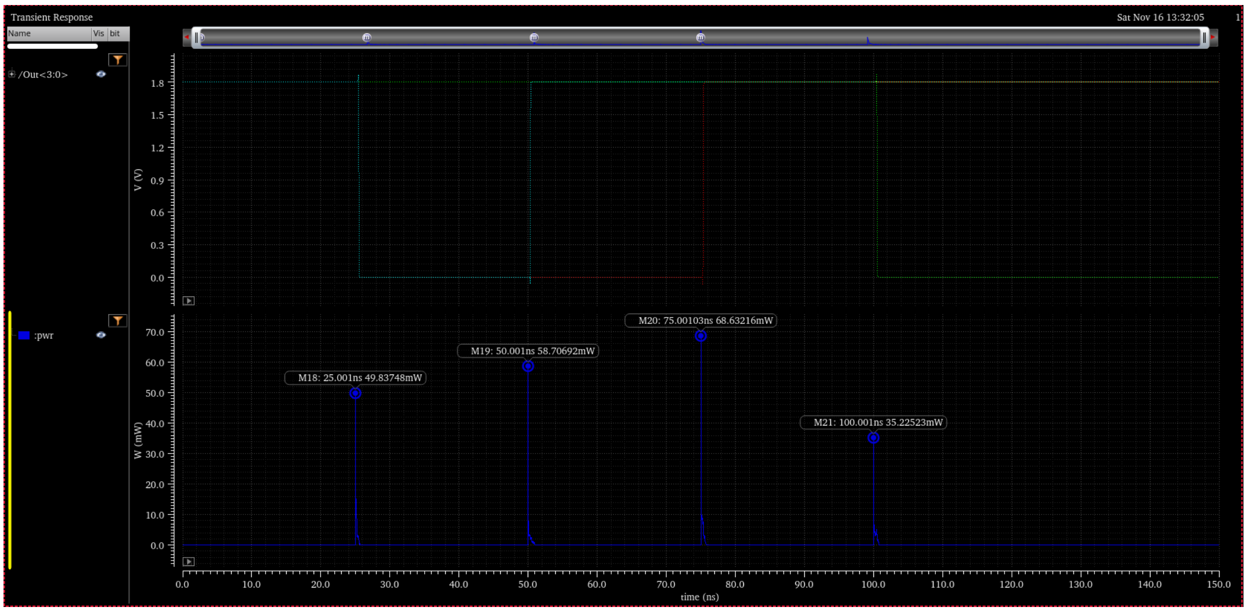


Transient Response
Sat Nov 16 13:43:56
/Out<3:0>
:pwr
M22: 25.001ns 49.83752mW
M23: 50.001ns 58.70688mW
M24: 75.001ns 68.83281mW
M25: 100.001ns 35.22542mW
V (V)
W (mW)
time (ns)

OR Gate

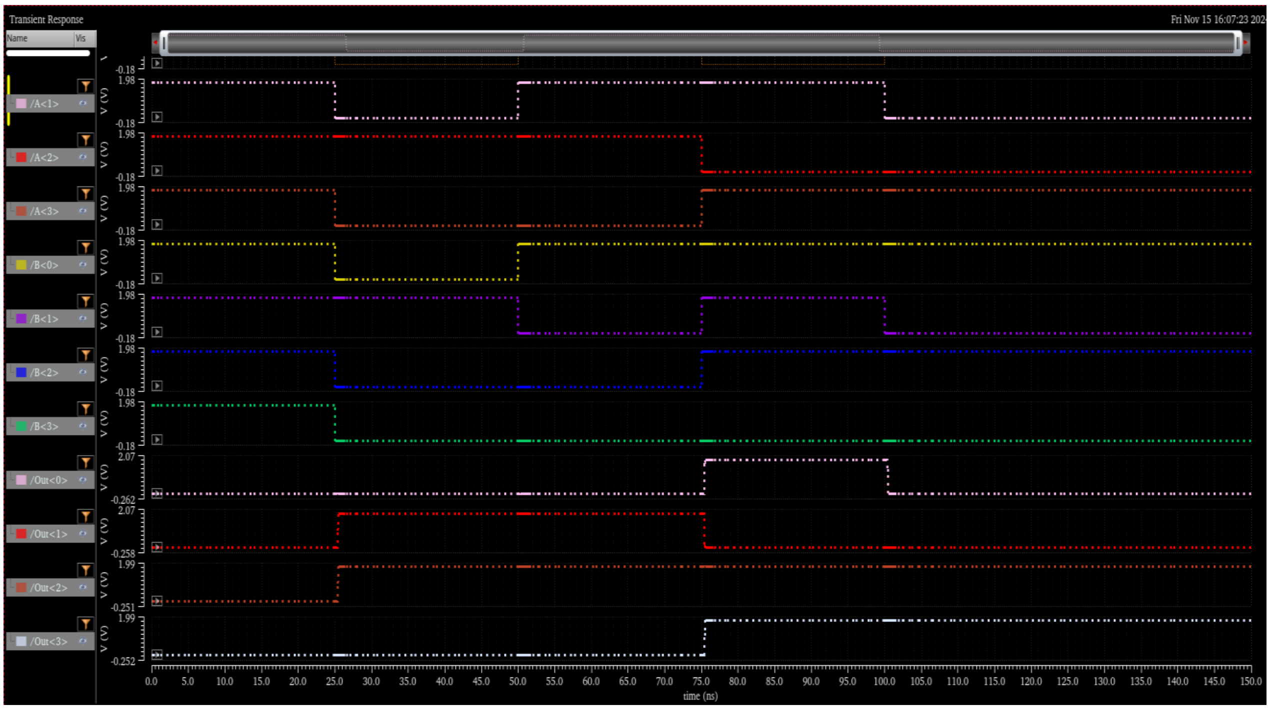


Transient Response
Fri Nov 15 16:02:08 20
/A<1>
/A<2>
/A<3>
/B<0>
/B<1>
/B<2>
/B<3>
/Out<0>
/Out<1>
/Out<2>
/Out<3>
V (V)
time (ns)

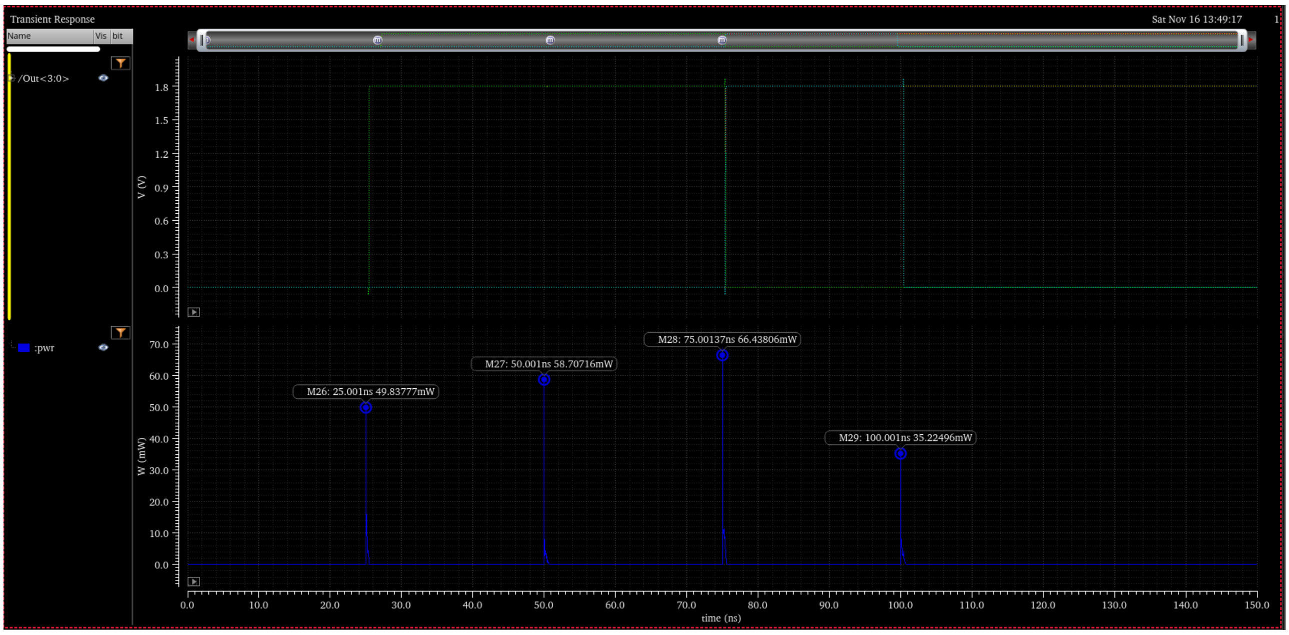


Transient Response
Sat Nov 16 13:32:05
/Out<3:0>
:pwr
M18: 25.001ns 49.83748mW
M19: 50.001ns 58.70692mW
M20: 75.00103ns 68.63216mW
M21: 100.001ns 35.22523mW
V (V)
W (mW)
time (ns)

XOR

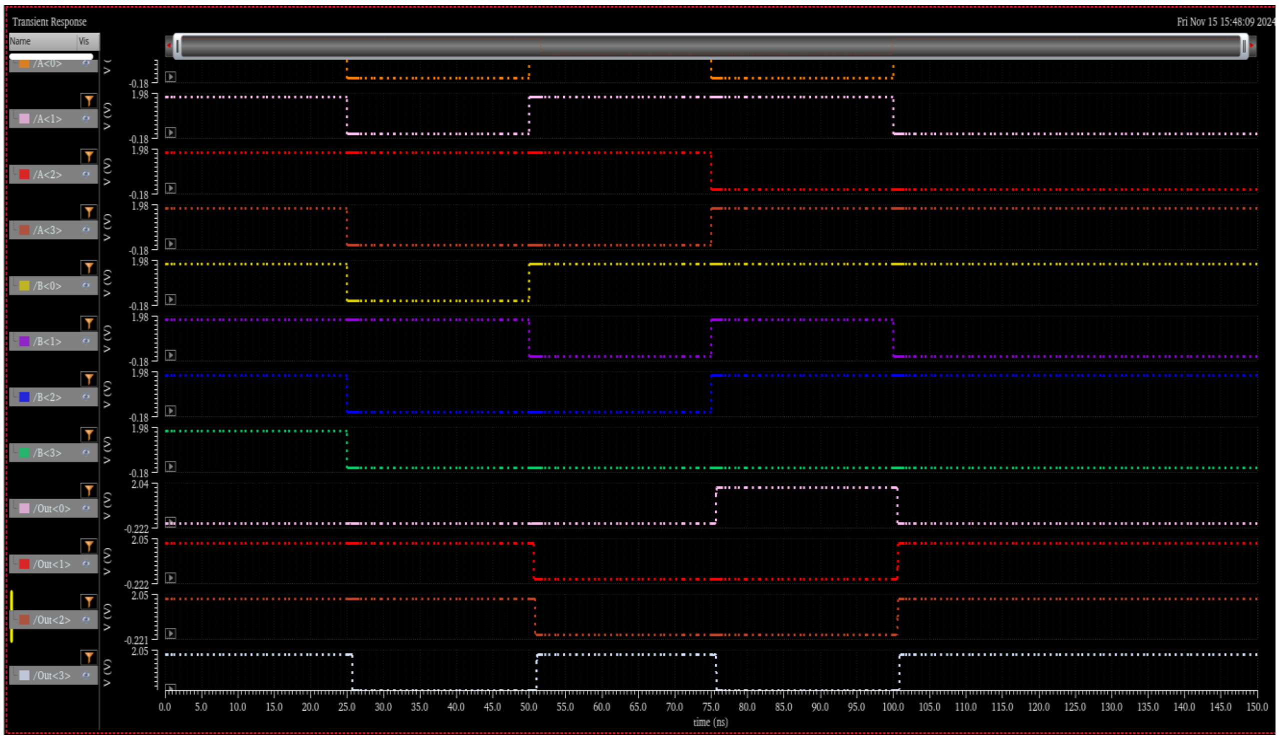


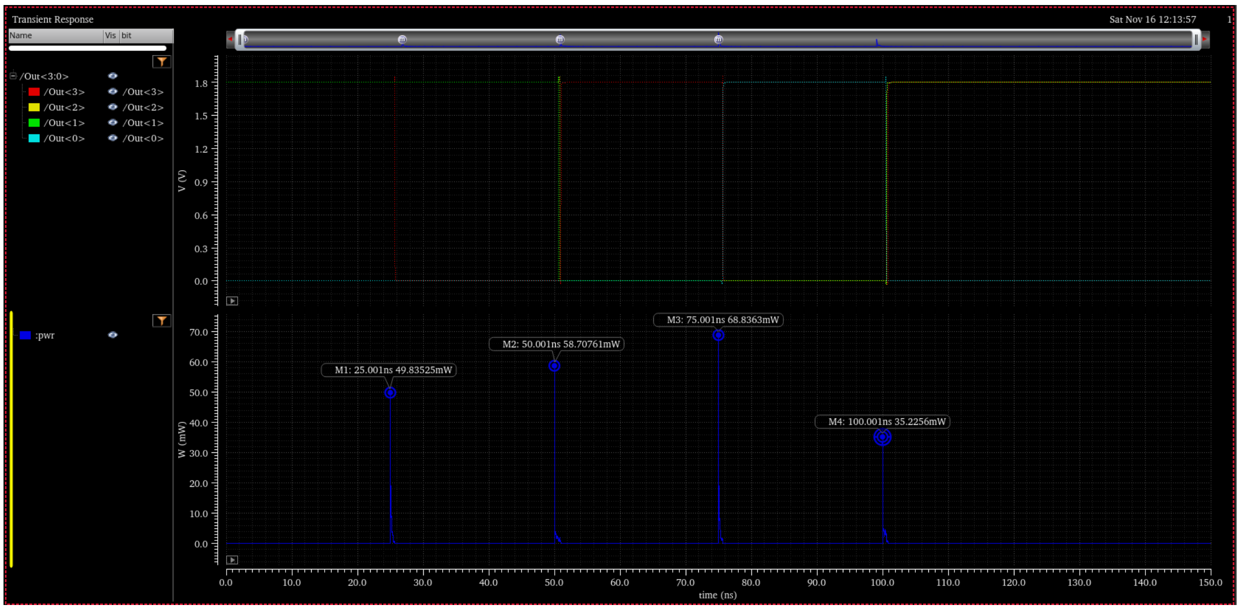
Transient Response
Fri Nov 15 16:07:23 202
/A<1>
/A<2>
/A<3>
/B<0>
/B<1>
/B<2>
/B<3>
/Out<0>
/Out<1>
/Out<2>
/Out<3>
V (V)
time (ns)



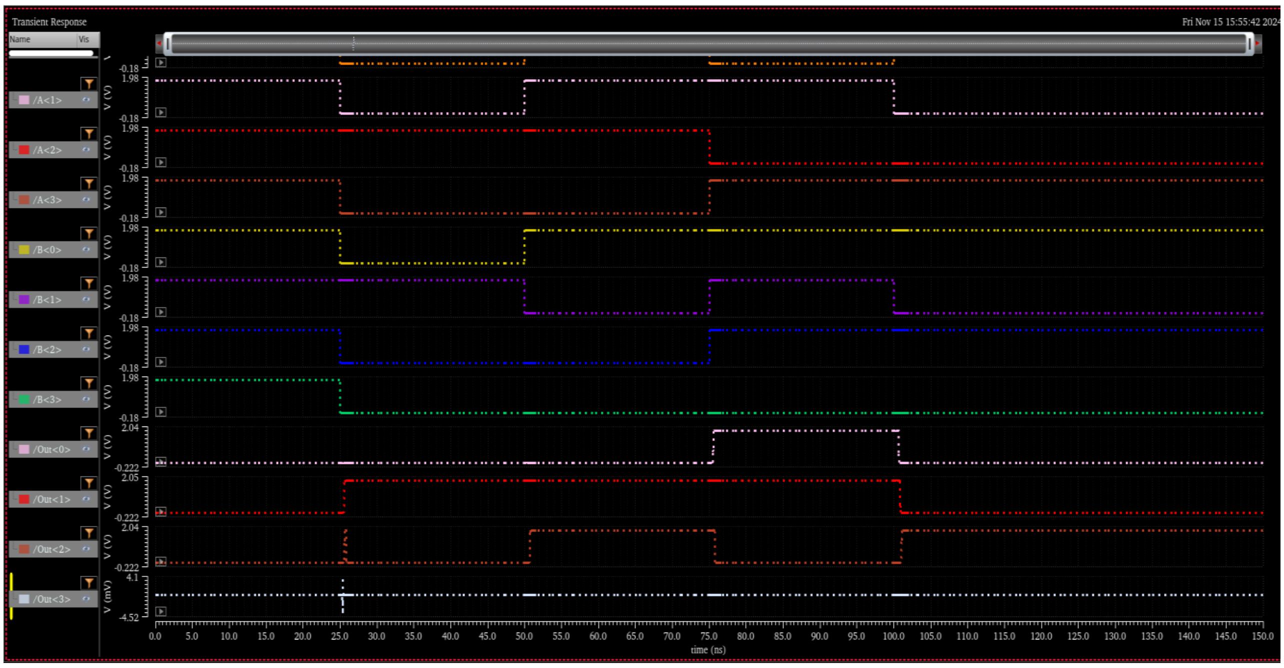
Arithmetic Unit of ALU:

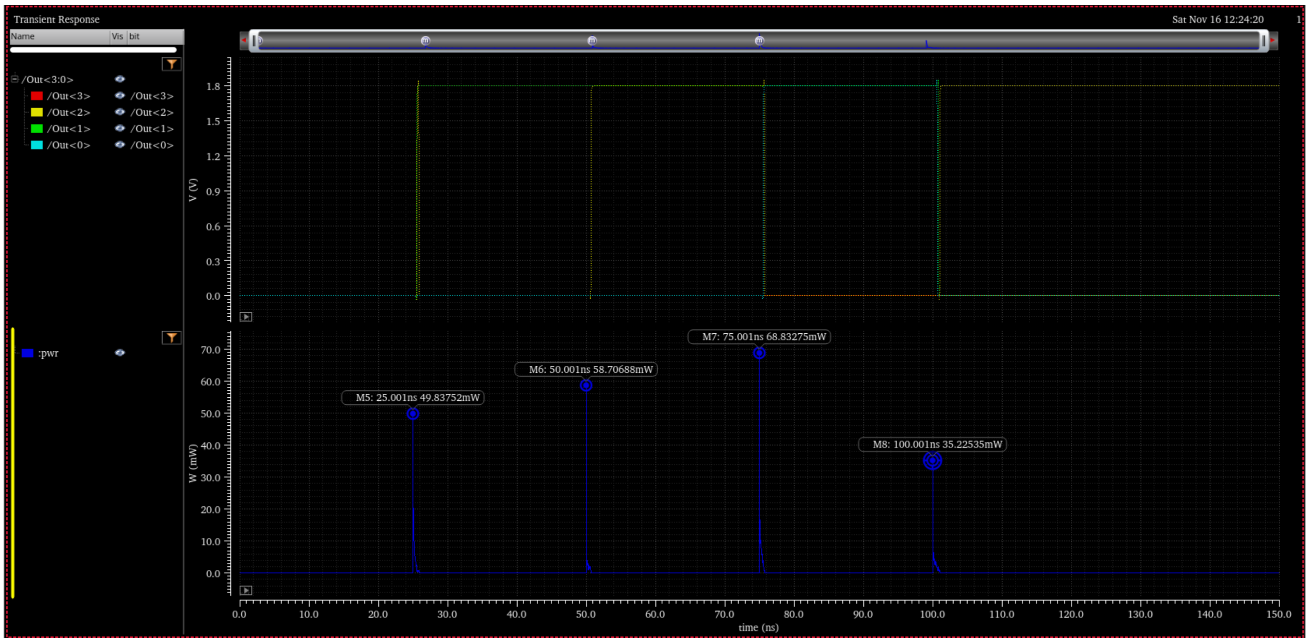
Addition



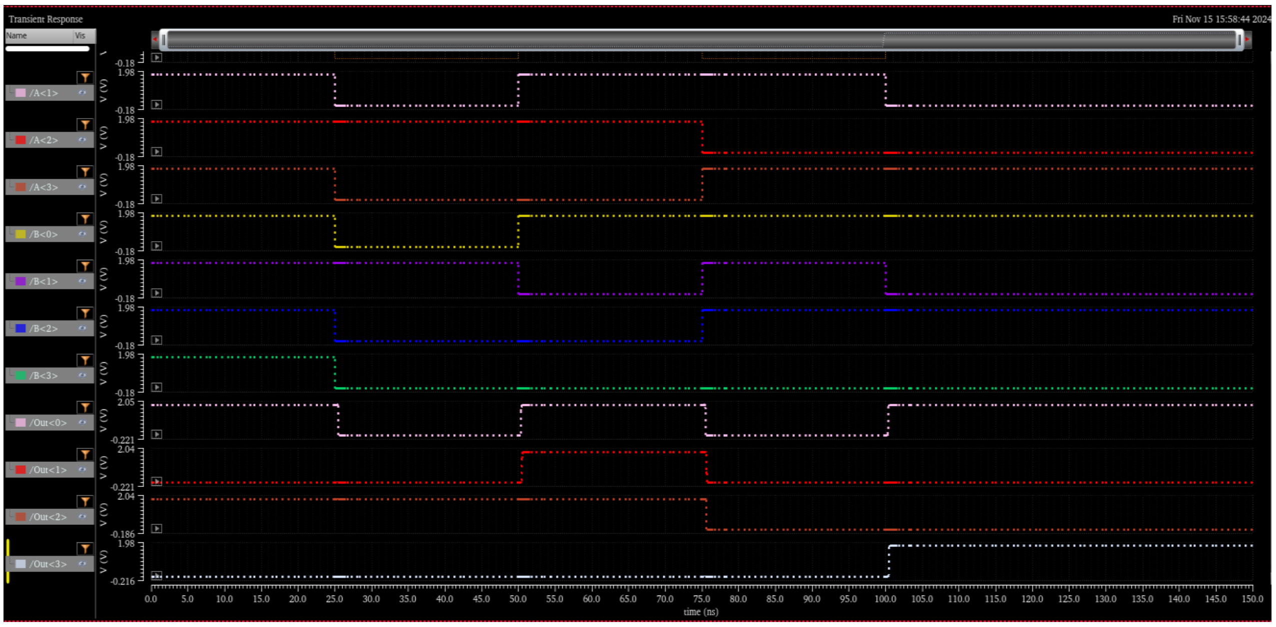


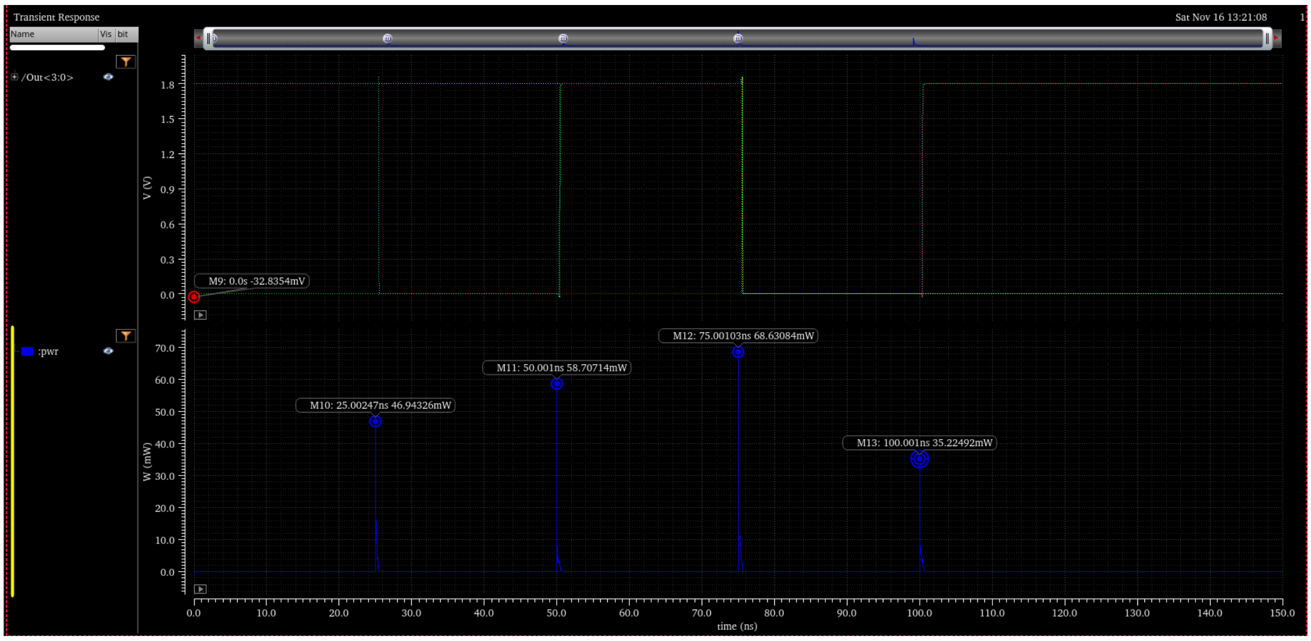
Subtraction




Transient Response
Sat Nov 16 12:24:20
Name
Vis
bit
/Out<3:0>
/Out<3>
/Out<2>
/Out<1>
/Out<0>
:pwr
V (V)
W (mW)
M5: 25.001ns 49.83752mW
M6: 50.001ns 58.70688mW
M7: 75.001ns 68.83275mW
M8: 100.001ns 35.22535mW
time (ns)

Divisor


Transient Response
Fri Nov 15 15:58:44 2024
Name
Vis
/A<1>
/A<2>
/A<3>
/B<0>
/B<1>
/B<2>
/B<3>
/Out<0>
/Out<1>
/Out<2>
/Out<3>
V (V)
time (ns)



Average Power Values

Full Adder	96.92 x 10^-6
Subtractor	111.2 x 10^-6
Division	103.0 x 10^-6
OR	102.2 x 10^-6
AND	110.3 x 10^-6
XOR	101.3 x 10^-6

Delay Values

Adder

OUT<0>	RISING DELAY	664.8 x 10^-12
	FALLING DELAY	552.6 x 10^-12
OUT<1>	RISING DELAY	631.6 x 10^-12
	FALLING DELAY	627.7 x 10^-12
OUT<2>	RISING DELAY	824.8 x 10^-12
	FALLING DELAY	615.7 x 10^-12
OUT<3>	RISING DELAY	1.02 x 10^-9
	FALLING DELAY	706.7 x 10^-12

Subtractor

OUT<0>	RISING DELAY	557.8 x 10^-12
	FALLING DELAY	691.1 x 10^-12
OUT<1>	RISING DELAY	560.5 x 10^-12
	FALLING DELAY	827.7 x 10^-12
OUT<2>	RISING DELAY	-24.40 x 10^-9
	FALLING DELAY	-49.15 x 10^-9
OUT<3>	RISING DELAY	0
	FALLING DELAY	0

Divisor

OUT<0>	RISING DELAY	476.7 x 10^-12
	FALLING DELAY	366.7 x 10^-12
OUT<1>	RISING DELAY	464.9 x 10^-12
	FALLING DELAY	655.7 x 10^-12
OUT<2>	RISING DELAY	-
	FALLING DELAY	598.2 x 10^-12
OUT<3>	RISING DELAY	464.8 x 10^-12
	FALLING DELAY	-

OR

OUT<0>	RISING DELAY	333.3 x 10^-12
	FALLING DELAY	504.1 x 10^-12
OUT<1>	RISING DELAY	-
	FALLING DELAY	514.6 x 10^-12
OUT<2>	RISING DELAY	-
	FALLING DELAY	-
OUT<3>	RISING DELAY	347.1 x 10^-12
	FALLING DELAY	504.1 x 10^-12

AND

OUT<0>	RISING DELAY	367 x 10^-12
	FALLING DELAY	439.1 x 10^-12
OUT<1>	RISING DELAY	367.5 x 10^-12
	FALLING DELAY	461.2 x 10^-12
OUT<2>	RISING DELAY	-
	FALLING DELAY	478.6 x 10^-12
OUT<3>	RISING DELAY	-
	FALLING DELAY	439.1 x 10^-12

XOR

OUT<0>	RISING DELAY	426.8 x 10^-12
	FALLING DELAY	434.6 x 10^-12
OUT<1>	RISING DELAY	424.2 x 10^-12
	FALLING DELAY	435.9 x 10^-12
OUT<2>	RISING DELAY	424.6 X 10^-12
	FALLING DELAY	-
OUT<3>	RISING DELAY	455.3 X 10^-12
	FALLING DELAY	-

BIODATA

	Name: Name: Vijay Aditya S Mobile Number: +91 6380318882 E-mail: vijayaditya.s2022@vitstudent.ac.in
	Name: Arvindhan K Mobile Number: +91 63820 89860 E-mail: arvindhan.k2022@vitstudent.ac.in
	Name: Sherlin Immanuela R Mobile Number: +91 7904203066 E-mail: sherlinimmaunela.r2022@vitstudent.ac.in
	Name: Renumanjari R K Mobile Number: +91 7395926575 E-mail: renumanjari.rk2022@vitstudent.ac.in

	Name: Teekshitta R Mobile Number: +91 7397404729 E-mail: teekshitta.r2022@vitstudent.ac.in