A 16-Bit ALU Design with Gate Diffusion Input (GDI) technique

M L Sriharsha
Department of Electrical Engineering
Indian Institute of Technology
Delhi, India
mlsriharsha98@gmail.com

***Abstract*—A low power, area efficient and fast 16-bit ALU design using GDI technique is presented. Gate Diffusion Input is considered an effective approach for low power and fast integrated circuits with small die area. The circuits are designed in 65nm technology node and performance analysis of the ALU is also presented.**

I. Introduction

In the view of recent growth in connected and portable devices, low power and high speed computation on small silicon footprint is the demand of present day electronics, where fast computations can be performed easily at the edge nodes. Digital circuits designers has explored other than CMOS techniques to design fast, area efficient and low power circuits, GDI is one such technique which promises low transistor count, low power consumption and low propagation delay and hence fast operations. An 'Arithmetic and Logical Unit' (ALU) is the main computational engine for any digital signal processing, microprocessor, micro-controller or any other application specific IC. In this work we have presented design of a 16 bit ALU with GDI technique to achieve fast performance with low power and area consumption.

The paper is organised in following manner: section I talks about introduction.

II. Gate Diffusion Input Technique

Many logic techniques have been developed in last two decades [1] to improve the performance of conventional static CMOS logic. Pass transistor logic (PTL) is one such very famous technique which has been improvised over the years. Most of the PTL implementations suffers with problems of threshold drop and static current, some sort of PTL techniques are invented to solve mentioned issue which are: Transmission Gate, Complementary pass-transistor logic, Double pass transistor logic.

Gate Diffusion Input (GDI) is a PTL technique which solves most of the problems with existing PTL circuits and offers a wide range of logic functions' implementation using just two transistors. GDI function can be explained with the simple GDI cell which looks similar to a CMOS inverter, but with following differences:

1) The cell contains three inputs, G(common gate of nmos and pmos), P (source/drain of pmos) and N (source/drain of nmos).
2) bulk of both pmos and nmos are connected to P and N rather than being connected to VDD and VSS as in CMOS logic.

Table I shows how various input configurations can implement different Boolean functions:

TABLE I
VARIOUS LOGIC FUNCTIONS IMPLEMENTED WITH GDI CELL FOR DIFFERENT INPUT COMBINATIONS

N	P	G	OUT	Function
0	B	A	$\overline{A}B$	F1
B	1	A	$\overline{A}+B$	F2
1	B	A	A+B	OR
B	0	A	AB	AND
C	B	A	$\overline{A}B+AC$	MUX
0	1	A	$\overline{A}$	NOT

Most of these Boolean functions requires 6-8 transistors in standard CMOS or pass transistor logic, but can be implemented in GDI with just two transistors. F1 and F2 are complete logic families, which means any two input function can be realized using F1 and F2.

The original GDI[2] was implemented in SOI technology which allowed a very low power consuming implementation, the translation of GDI cell in CMOS requires little modification [3]. The bulk of nmos and PMOS has to be tied to VDD and VSS otherwise implementation of F2 function would result in shorting of bulk channel diode.

We also use swing restoration transistors in our implementation(as discussed in [2]) to improve the output swings of F1 and F2 logic.

III. Design of ALU

We have used full swing GDI technique for circuit realization.Major circuit blocks are 2×1, 4×1 multiplexers and full adder. Circuit design for these blocks is discussed in detail:

A. Challenges faced

These are the challenges that were faced while designing:

1) There is no GND and VDD terminals attached to transistor, So the only way to discharge is through the input terminals that are attached.
2) To remove the glitch in the waveform capacitance($0 - 0.15fF$) were added in between nodes
3) Sizing of transistor vary from different logical blocks. To generalize the overall sizing $PMOS_{W/L}$ is three time that of $NMOS_{W/L}$
4) Inputs applied to circuits blocks have finite rise and fall time to resemble with practical input

B. 2×1 Multiplexer

A Multiplexer is a switch which selects one of the input based on the select signal. GDI implementation of a 2×1 Mux is shown in figure 1. Transistor count in the implementation is just 4 as compared to 10 in CMOS logic. Schematic and Simulation results of 2×1 multiplexer is shown in the figure below:

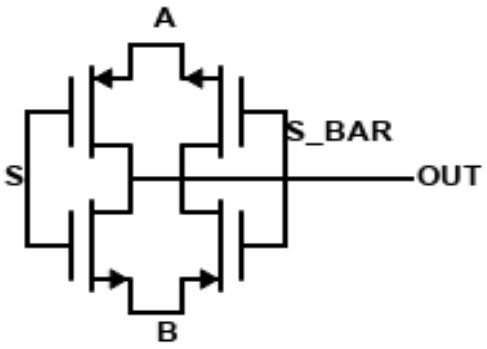

Fig. 1. symbol of 2×1 multiplexer

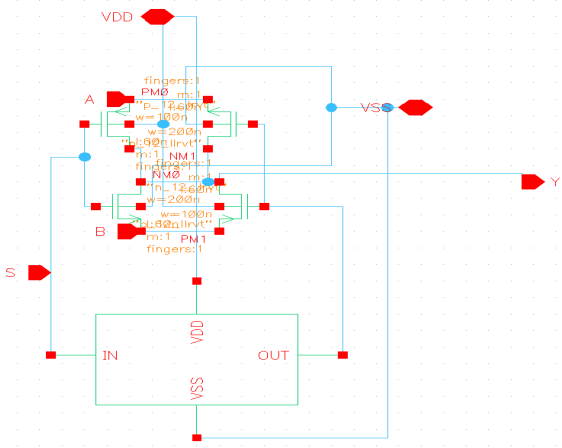

Fig. 2. Schematic of 2×1 multiplexer

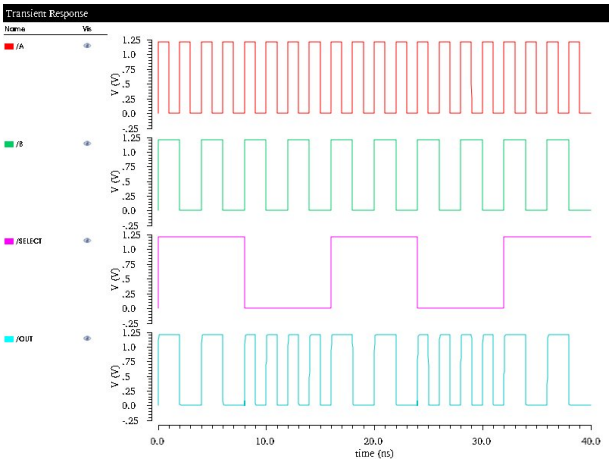

Fig. 3. Simulation results of 2×1 multiplexer

C. 4×1 Multiplexer

Using 2×1 Mux any higher order Mux can be designed. Figure 4 shows 4×1 implementation. Simulation results of 4×1 multiplexer is shown in the figure below:

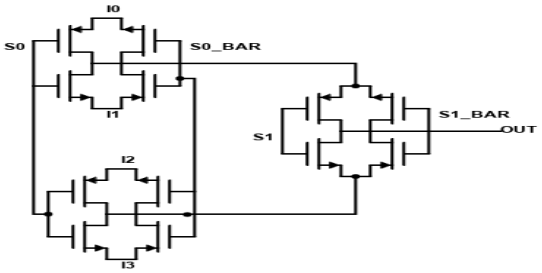

Fig. 4. symbol of 4×1 multiplexer

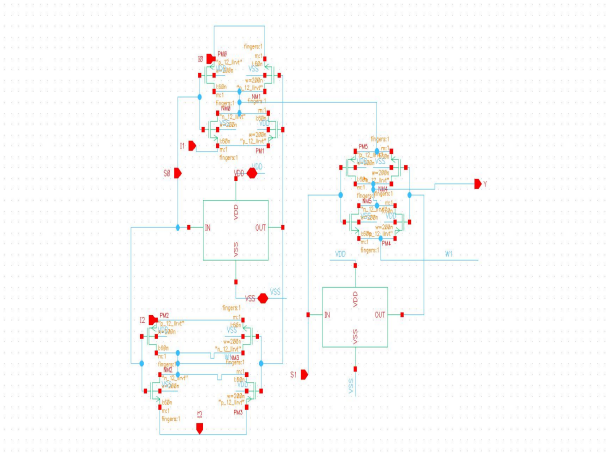

Fig. 5. Schematic results of 4×1 multiplexer

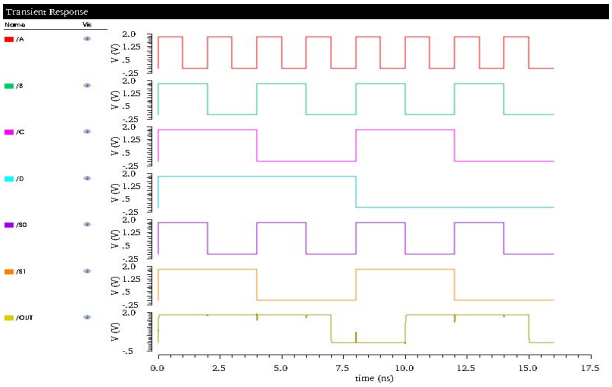

Fig. 6. Simulation results of 4×1 multiplexer

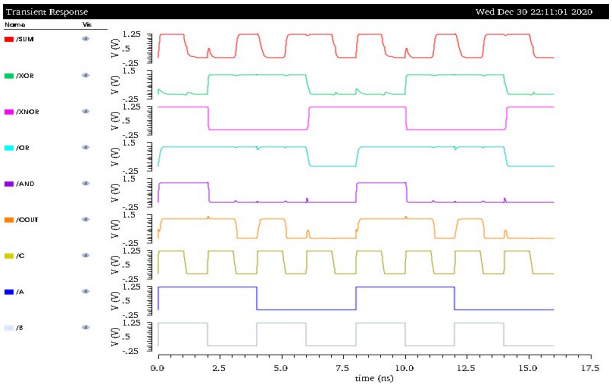

Fig. 9. Simulation results of full adder

D. Full Adder

A full adder performs arithmetic sum of 3-input bits and generates two bit output (sum and carry). Full swing GDI AND, OR and XOR gates are designed to implement full adder as shown in figure 7. Other GDI implementation of full adder circuit [3] exist but the one used in this work [2] is least power consuming. Simulation results of full adder is shown in the figure below:

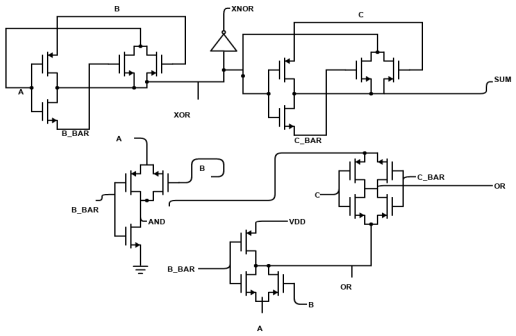

Fig. 7. symbol of Full Adder

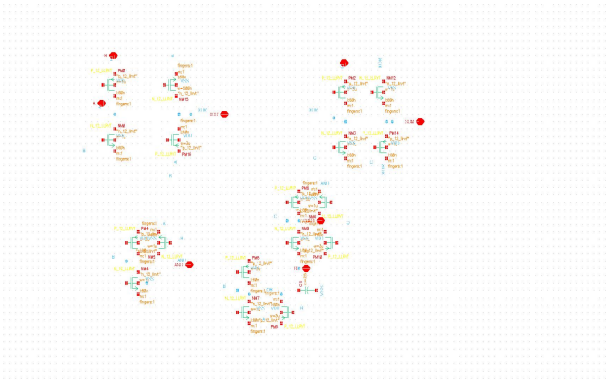

Fig. 8. Schematic results of full adder

E. Arithmetic Logic Unit (ALU)

The designed ALU can perform arithmetic operations: addition, subtraction, increment and decrement, logical operations:AND, OR, XOR and XNOR. A 16-bit ALU is designed which consists of 16 stage of single bit ALU. Each stage of single bit ALU is designed with multiplexers and full adder blocks discussed above. Single bit ALU has 3-inputs A,B and C and 3-select lines S0, S1 and S2. The ALU operation can be selected with these select lines. Table 2 describes ALU operations with select lines:

TABLE II
ALU OPERATIONS WITH SELECT LINES

S2	S1	S0	ALU operation
0	0	0	Decrement
0	0	1	Addition
0	1	0	Subtraction
0	1	1	Increment
1	0	0	AND
1	0	1	XOR
1	1	0	XNOR
1	1	1	OR

1) 1-BIT ALU: The schematic diagram of a 1-bit ALU is presented in figure 10. First be have a 4×1 Mux which passes appropriate B input (B, $\overline{B}$, 0 or 1), based on select lines S0 and S1. Next be have a 2×1 Mux which discriminate between Arithmetic or logical operation since both type of the operations are performed at full adder. Full adder generates all possible ALU outputs and again a 4×1 Mux followed by 2×1 Mux directs the correct signal to Y output based on select lines. The bit of ALU requires only 48 transistors when implemented with full swing GDI technique.

2) 16-BIT ALU: The output of schematic is connected to load (buffer).And inputs that are selected two random values for 16 bits ALU and applied for 4ns4e3g2 each:

1) **A=1101110111011101,B=1001100110011001,C=0**
2) **A=0101010101010101,B=1111111111111111,C=1**

The Schematic diagram is shown in below.

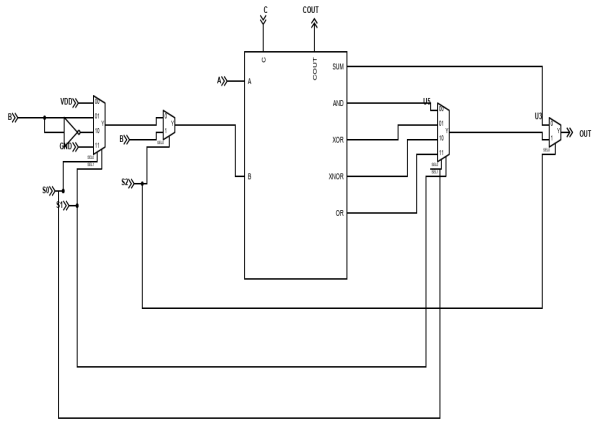

Fig. 10. symbol of 1-bit ALU

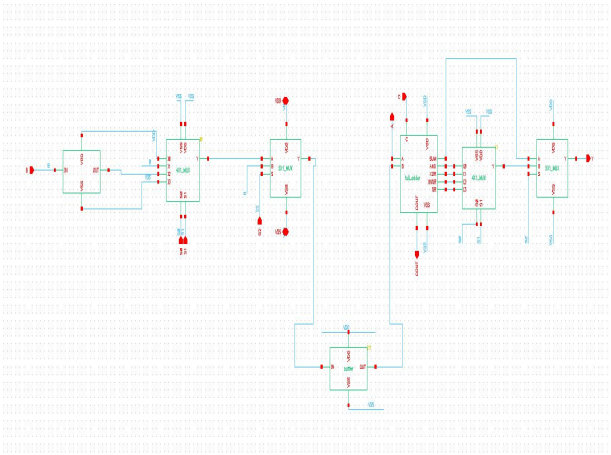

Fig. 11. schematic of ALU 1 bit

In the figure shown below has all the combination of input with all control signal that mentioned in TABLE II

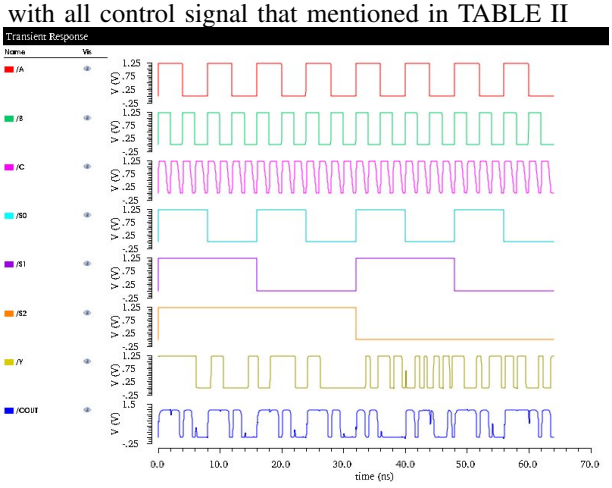

Fig. 12. simulation result of ALU 1 bit with all operations

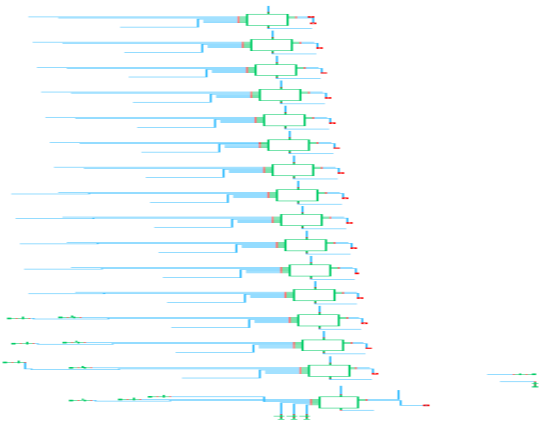

Fig. 13. schematic of ALU 16 bit

3) "000" DECREMENT:

1) **A=1101110111011101,C=0,OUT=1101110111011100**
2) **A=0101010101010101,C=1,OUT=0101010101010101**

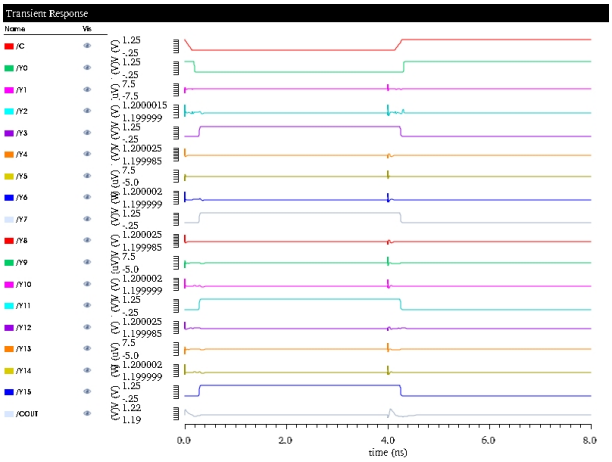

Fig. 14. "000" DECREMENT operation of ALU 16 bit

4) "001" ADDITION:

1) **A=1101110111011101,B=1001100110011001, C=0,OUT=0111011101110110,COUT=1**
2) **A=0101010101010101,B=1111111111111111, C=1,OUT=0101010101010101,COUT=1**

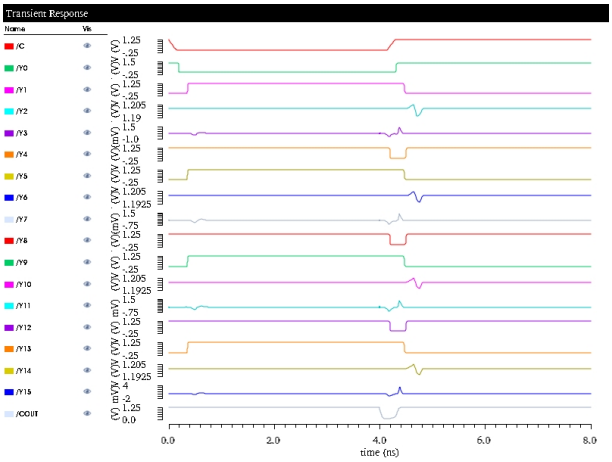

Fig. 15. "001" ADDITION operation of ALU 16 bit

5) "010" SUBTRACTION:

1) **A=1101110111011101,B=1001100110011001, C=0,OUT=0100010001000011,COUT=1**
2) **A=0101010101010101,B=1111111111111111, C=1,OUT=0101010101010110,COUT=0**

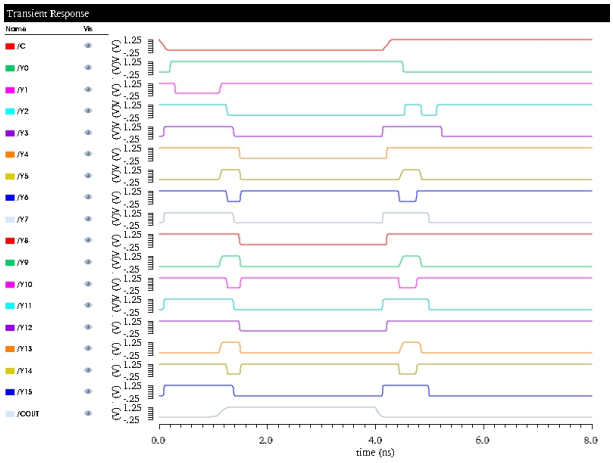

Fig. 16. "010" SUBTRACTIONS operation of ALU 16 bit

6) "011" INCREMENT:

1) **A=1101110111011101,C=0,OUT=1101110111011110**
2) **A=0101010101010101,C=1,OUT=0101010101010110**

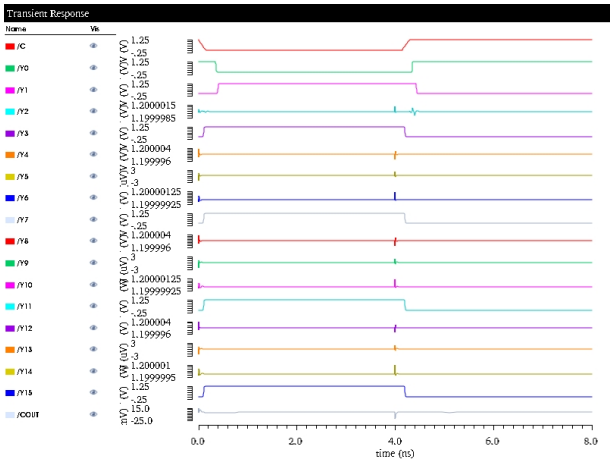

Fig. 17. "011" INCREMENT operation of ALU 16 bit

7) "100" AND:

1) **A=1101110111011101,B=1001100110011001, OUT=1001100110011001**
2) **A=0101010101010101,B=1111111111111111, OUT=0101010101010101**

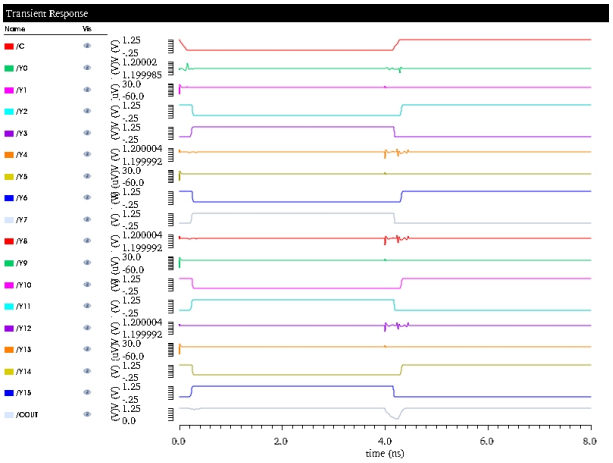

Fig. 18. "100" AND operation of ALU 16 bit

8) "101" XOR:

1) **A=1101110111011101,B=1001100110011001, OUT=100010001000100**
2) **A=0101010101010101,B=1111111111111111, OUT=1010101010101010**

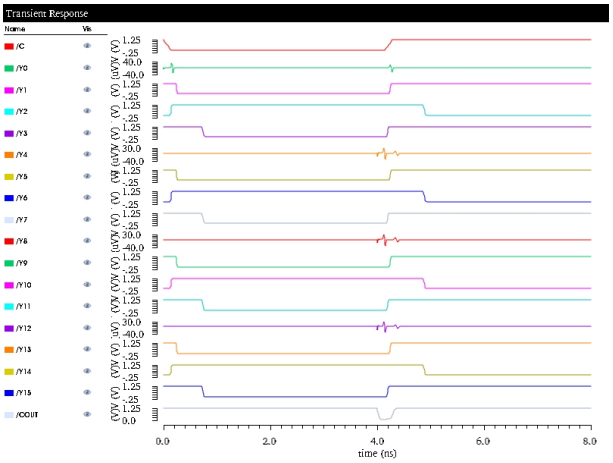

Fig. 19. "101" XOR operation of ALU 16 bit

9) "110" XNOR:

1) **A=1101110111011101,B=1001100110011001, OUT=1001100110011001**
2) **A=0101010101010101,B=1111111111111111, OUT=0101010101010101**

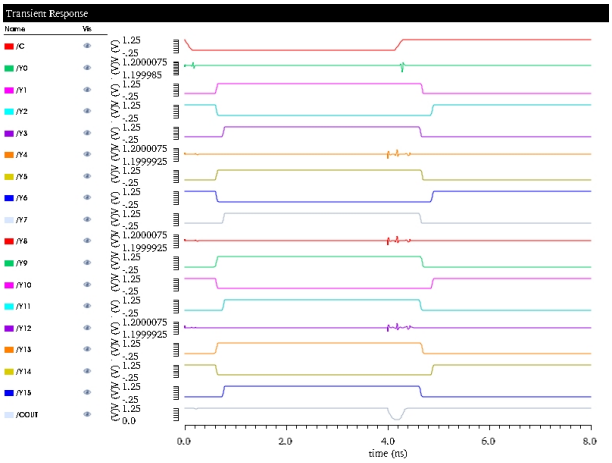

Fig. 20. "110" XNOR operation of ALU 16 bit

10) "111" OR:

1) **A=1101110111011101,B=1001100110011001, OUT=1101110111011101**
2) **A=0101010101010101,B=1111111111111111, OUT=1111111111111111**

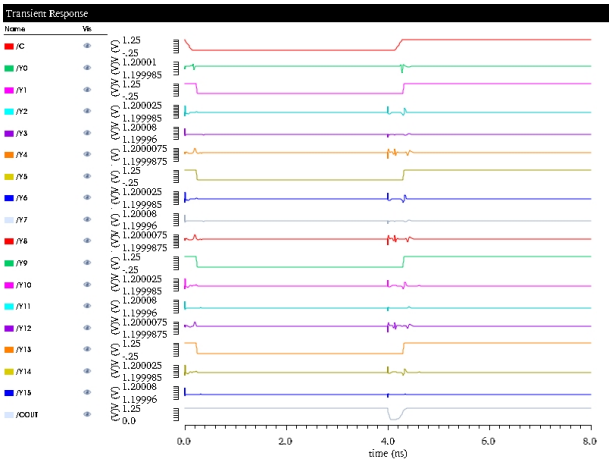

Fig. 21. "111" OR operation of ALU 16 bit

IV. Simulation and Calculations

The 16 bit ALU design is done in 65nm UMC process. Results are simulated using Spectre based simulator. Below table shows the summary of results:

TABLE III
Simulation Results

Design	Technology	Number of Transistors	Power in uW
16-bit ALU	65nm	800	98.83

References

[1] A. Morgenshtein, A. Fish and I. A. Wagner, "Gate-diffusion input (GDI): a power-efficient method for digital combinatorial circuits," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 10, no. 5, pp. 566-581, Oct. 2002, doi: 10.1109/TVLSI.2002.801578.

[2] M. A. Ahmed and M. A. Abdelghany, "Low power 4-Bit Arithmetic Logic Unit using Full-Swing GDI technique," 2018 International Conference on Innovative Trends in Computer Engineering (ITCE), Aswan, 2018, pp. 193-196, doi: 10.1109/ITCE.2018.8316623.

[3] M. Shoba, R. Nakkeeran, "GDI based full adders for energy efficient arithmetic applications", Engineering Science and Technology, an International Journal, vol. 19, no. 1, pp. 485–496, Mar. 2016.