16 Bit Comparator

Using Nand - Nor Architecture

Osaid Nur
Computer Engineering Department
Birzeit University
ID: 1210733
Ramallah , Palestine

Moath Wajeeh
Computer Engineering Department
Birzeit University
ID : 1210125
Ramallah , Palestine

Ahmad Waleed
Computer Engineering Department
Birzeit University
ID : 1210326
Ramallah , Palestine

***Abstract*—In this world where the electronics and digital devices are almost everywhere, we can consider the magnitude comparator as a key component, in this paper we will show our design for an efficient 16-bit magnitude comparator, we designed a 4-bit comparator firstly and then a 16-bit comparator from the 4-bit one, during our design process we went through multiple stages, firstly we tried to use GDI method, after that we chose to use the usual CMOS design with the usual design for a 4-bit comparator from inverters, AND, OR and NOR gates and this design had 158 transistors, after that we changed the design to be only with inverters, NAND and NOR gates for optimization purposes and we reduced the number of transistors to 126 transistor, in the next step we designed the 4-bit comparator to output only the greater than and less than operators so we reduced the number to 116 transistors, as a result of all the above mentioned optimization trials we got a 16-bit comparator with 584 transistor.**

Keywords—Magnitude comparator, low-power, CMOS design, 16-bit comparator, logic gate minimization

I. INTRODUCTION

Magnitude comparator is a digital device that takes two binary numbers as input and outputs if one of them is greater, less or equal to the other number [1].

Comparators has a crucial and very important role in nowadays electronics field, we can find comparators in too many applications such as communication, sorting, multiprocessing, microcontroller and Analog to Digital Converter (ADC), as a result of that and the wide spread of digital devices the people tend to the devices that have a battery that prolongs for long time as far as possible, so that the need for designing an efficient and low power comparator increases rapidly, figure1 shows the importance of comparison process in sorting [2].

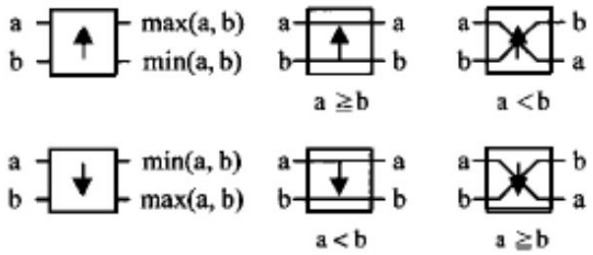


Figure 1: comparing process in sorting [2].

There are many methods to design an optimized comparator such as: GDI, AVLS, and Gate minimization to reduce the number of transistors, to design our 16x16 comparator, we used gate minimization to express the comparator with NAND and NOR gates, we firstly built a 4x4 comparator and then we connected 5 of them to construct a 16x16 bit comparator.

II. 4-BIT COMPARATOR VERSION1

This is the plain design which consists of inverters, AND, OR and NOR gates Maintaining the Integrity of the Specifications

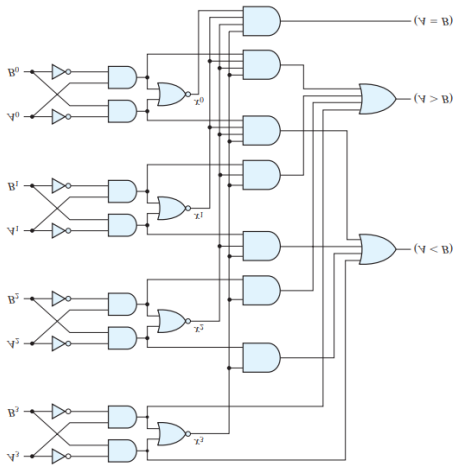


Figure 2: 4-bit comparator version1[3].

The diagram depicted on figure2 is built based on the common algorithm that we as humans use to compare two numbers, we simply check the digits of the number digit by digit, if all digits are equal then the two numbers are equal, else we will keep moving to the next digit till we reach two different digits and based on the value of those two digits we judge if A > B or A < B, to represent the algorithm in more formal mathematic representation we will use the following Boolean expressions:

A. Equality of two bits

We can say that two bits are equal if both of them is 1 or both of them is 0, this can be expressed as shown in the following figure:

$$x_i = A_i B_i + A_i' B_i' \quad \text{for } i = 0, 1, 2, 3$$

Figure 3: Equality of two bits [3]

B. Equality of two numbers

Two numbers are considered equal if all bits in the first number are equal to the bits in the second number as represented in the figure below:

$$(A = B) = x_3x_2x_1x_0$$

Figure 4: Equality of two numbers [3]

C. *Greater than and less than*

The algorithm of checking this part of the comparator has been clarified in the previous lines, and it can be expressed as follows:

$$(A > B) = A_3B_3' + x_3A_2B_2' + x_3x_2A_1B_1' + x_3x_2x_1A_0B_0'$$
$$(A < B) = A_3'B_3 + x_3A_2'B_2 + x_3x_2A_1'B_1' + x_3x_2x_1A'n_0B_0'$$

Figure 5: Checking greater than and less than [3]

III. 4-BIT COMPARATOR VERSION2

To optimize the previous design and reduce some transistors, we converted the logic to be in terms of NAND and NOR gates instead of AND and OR gates, in this step we reduced the number of transistors from 158 to 126 transistor, figure 6 shows the circuit diagram for the comparator after this step.

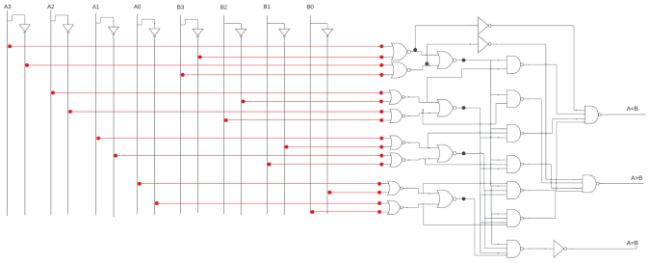


Figure 6: 4-bit comparator version2

IV. 4-BIT COMPARATOR FINAL VERSION

Because we are seeking more optimizations, we redesigned the comparator with nand-nor architecture to output only greater than and less than operators and we took the equal from the 16-bit comparator, with this optimization we reduced the transistors from 126 to 116 for each 4-bit comparator

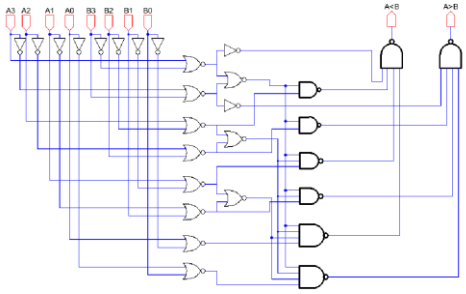


Figure 7: 4-bit comparator final version

The following figures shows the layout and the simulation result of 4-bit comparator outputs which are greater and less than comparison result

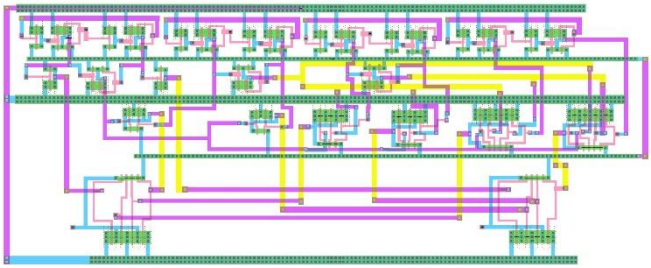


Figure 8: Layout of 4-bit comparator

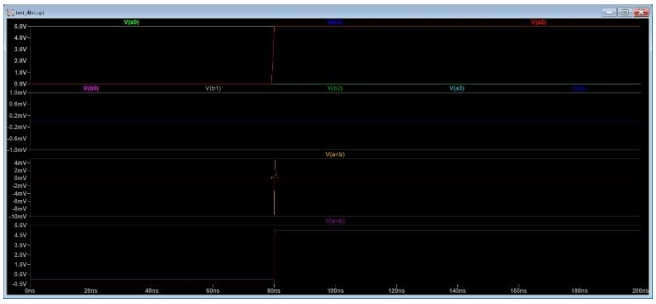


Figure 9: Simulatoin for greater than operation

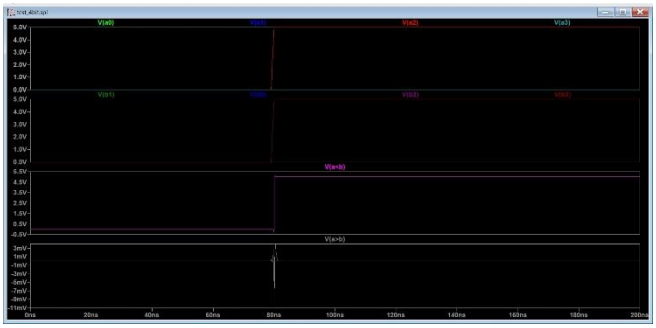


Figure 10: simulation for less than operation

V. GDI TECHNIQUE

Gate diffusion input technique (GDI) is a technique used in design to reduce dynamic and static power by designing the logical gates by low number of transistors, the following figures shows the basic gate element in GDI as well as the table of functions to be represented using the GDI, we tried to use this technique for more optimization but it did not work for unknown reason. [4]

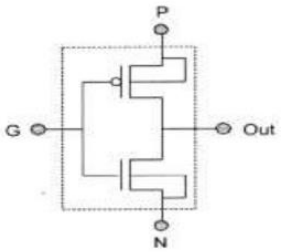


Figure 11: GDI basic gate [4].

TABLE I
FUNCTIONS THAT CAN BE IMPLEMENTED USING BASIC GDI CELL

N	P	G	Out	Function
0	1	A	A'	INVERTER
0	B	A	A'B	FUNCTION1
B	1	A	A'+B	FUNCTION 2
1	B	A	A+B	OR
B	0	A	AB	AND
C	B	A	A'B+AC	MUX
B'	B	A	A'B+B'A	XOR
B	B'	A	AB+A'B'	XNOR

Figure 12: GDI functions table [4].

VI. GATES USED IN THE DESIGN

A. *Inverter*

The inverter gate is used to invert the value of the input, for example it will toggle the 0 to 1 and vice versa, in the following figures the schematic, layout and simulation of the inverter will be shown.

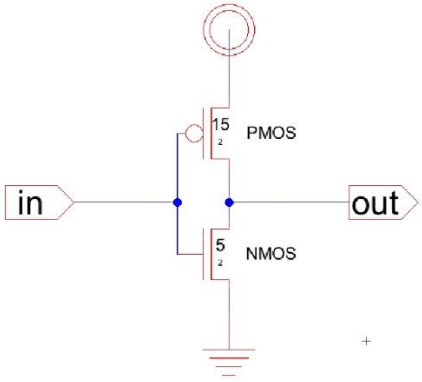


Figure 13: schematic of the inverter

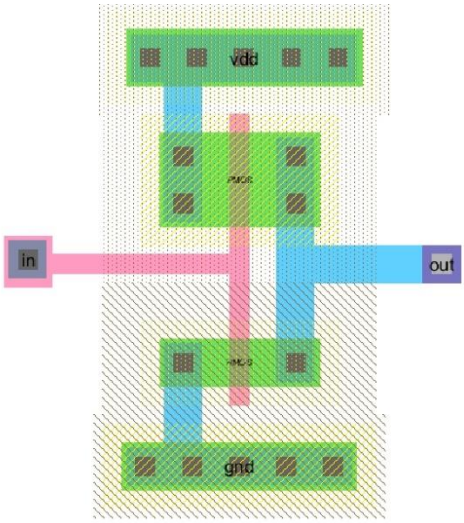


Figure 11: Layout of the inverter

Figure 12: Simulation of the inverter

B. *Nand gate*

For the Nand gate we built 3 different versions, the first has 2 inputs, the second is with 3 inputs and the last one has 4 inputs, the following will show the schematics, layout as well as the simulation for each gate

- 2-input Nand gate

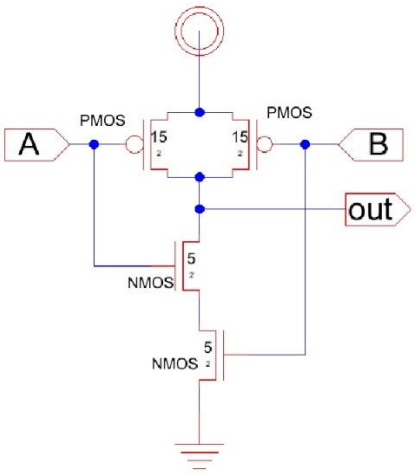


Figure 13: Schematic of 2-input Nand

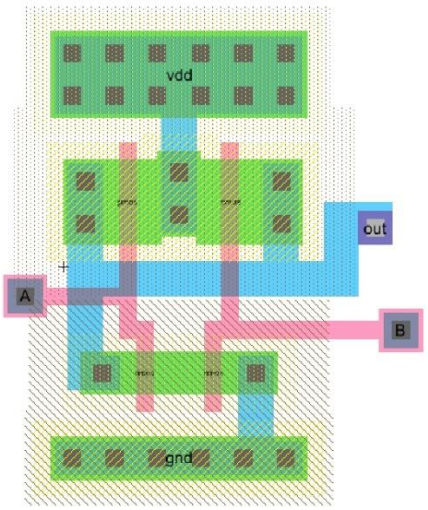


Figure 14: 2-input Nand gate layout

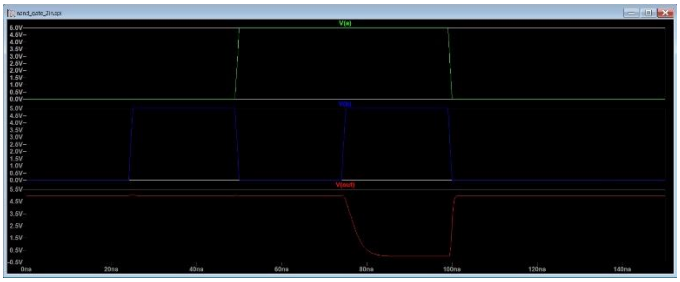


Figure 15: 2-input Nand simulation

- 3-input Nand gate

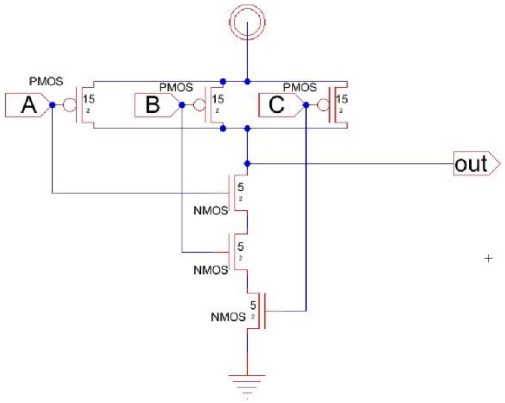


Figure 16: 3-input Nand schematic

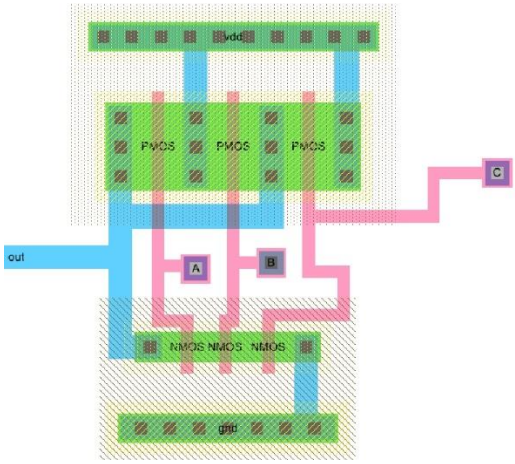


Figure 17: 3-input Nand layout

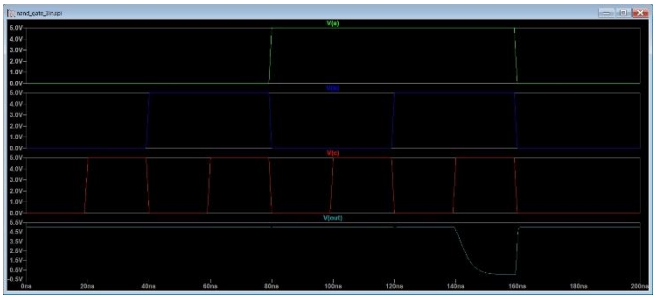


Figure 18: 3-input Nand simulation

- 4-input Nand gate

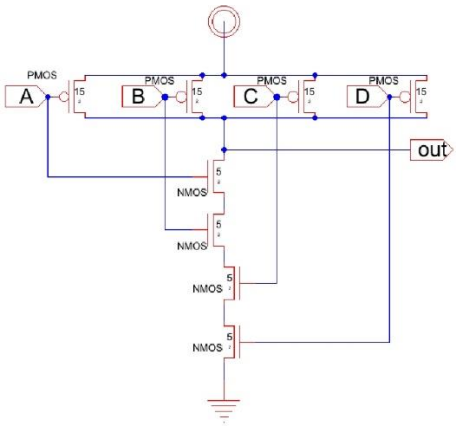


Figure 19: 4-input Nand schematic

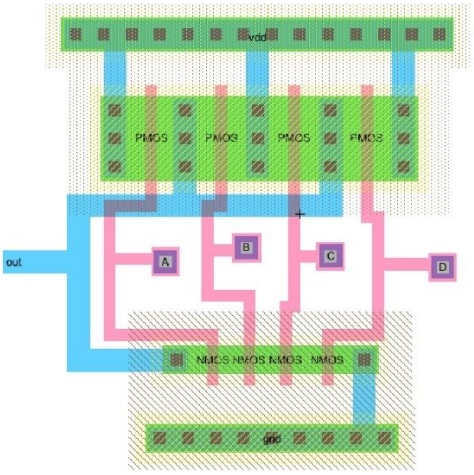


Figure 20: 4-input Nand layout

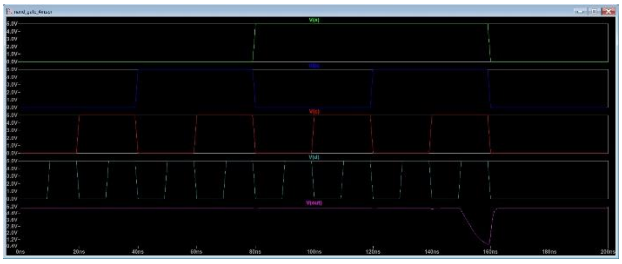


Figure 21: 4-input Nand simulation

C. Nor gate

We used this gate mainly to take the equality output from the 16-bit comparator and its input is the greater than and less than output, in the following snapshots the schematic, layout and simulation of this gate will be shown.

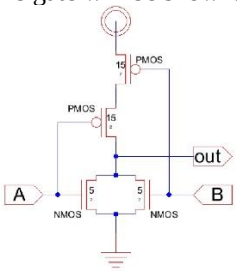


Figure 22: Nor gate schematic

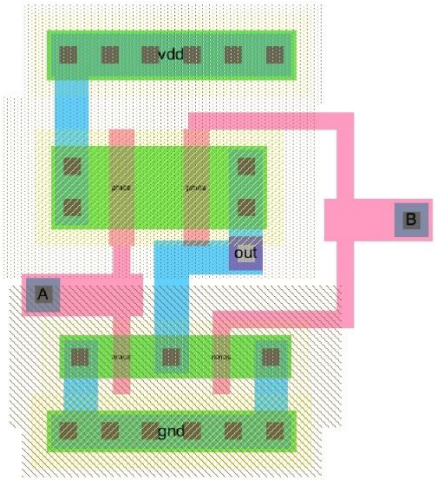


Figure 23: Nor gate layout

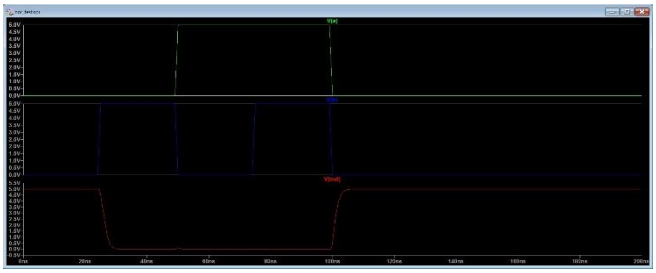


Figure 24: Nor gate simulation

VII. 16-BIT COMPARATOR

We built the 16-bit comparator using 5 4-bit comparators blocks, we connected the output of the first 4-bit comparators with the input of the fifth one, to output the equal operation we took the > and < output and do the NOR operation between then, with this connection we gained some additional optimization.

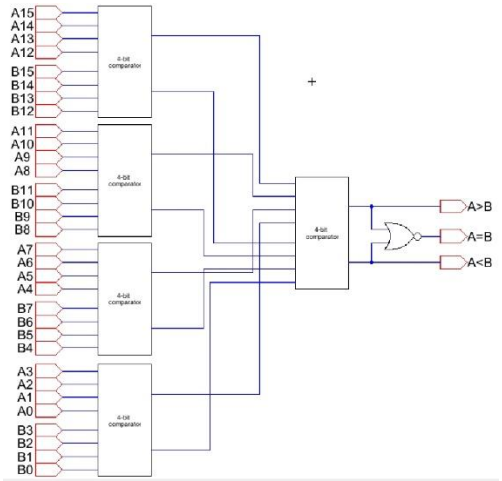


Figure 25: 16-bit comparator schematic

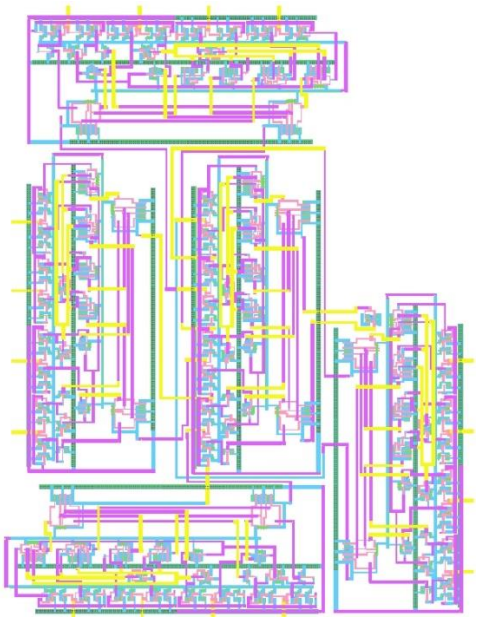


Figure 26: 16-bit comparator layout

Figure 27: 16-bit simulation with greater than operation

Figure 28: 16-bit simulation with less than operation

Figure 29: 16-bit simulation with equal operation

POWER CALCULATIONS

The 16 bit comparator is tested for 3 different test cases , for A=B , A>B , and A>B , the results was as following :
At A=B , the power consumed was

```
Measurement "tr" FAIL'ed
power: AVG(i(vdd)*v(vdd))=-7.32915e-007 FROM 0 TO 2e-007
```

At A>B :

```
Measurement "tr" FAIL'ed
power: AVG(i(vdd)*v(vdd))=-1.67748e-005 FROM 0 TO 2e-007
```

And at A<B:

```
Measurement "tr" FAIL'ed
power: AVG(i(vdd)*v(vdd))=-1.45316e-005 FROM 0 TO 2e-007
```

DELAY CALCULATIONS:

The delay is tested also with the three cases , A>B,A<B , and A=B , the result is as shown :

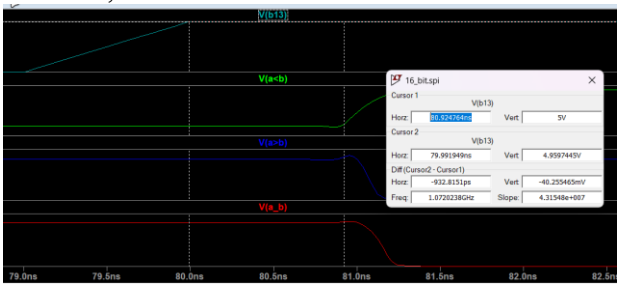


Figure 30. delay when A<B

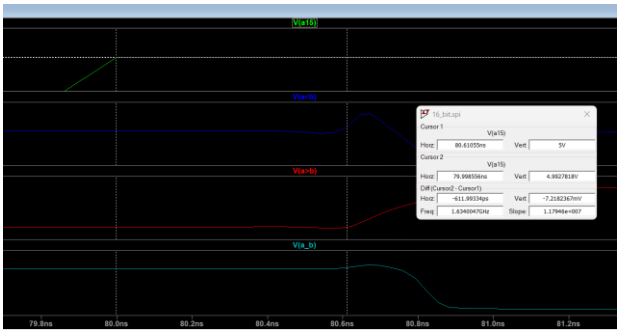
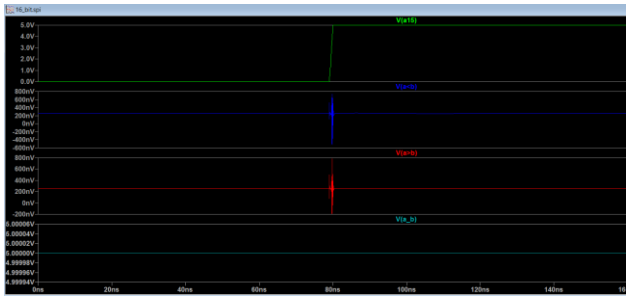


Figure 31. delay when A>B

it's hard to capture the delay for equality because the line is continuous .



CONCLUSION

At the end of this project, we have learned a lot about integrated circuits, CMOS design, comparators as well as schematics and layout of the circuit and calculate the delay, power and area of the circuit, we implemented all of this using electric software in addition to LTspice, we handled a lot of errors and design issues till we reached an optimized digital 16-bit comparator.

ACKNOWLEDGMENT

In this section of the paper, we are willing to express our sincere thanks to Dr.khader mohammad and TA.Raha zabadi for helping us to complete this project and understand the theoretical part of the integrated circuit course using this project.

REFERENCES

[1] https://www.academia.edu/91827442/Design_of_High_Speed_and_Area_Efficient_N_Bit_Digital_Comparator?rhid=29631497504&swp=rr-rw-wc-50007741

[2] https://www.academia.edu/50007741/A_Low_Power_8_bit_Magnitude_Comparator_with_Small_Transistor_Count_using_Hybrid_PTL_CMOS_Logic

[3] Mores mano book

[4] https://www.irjet.net/archives/V6/i7/IRJET-V6I7509.pdf