Griffith UNIVERSITY

Integrated Circuit Design

Design Project

4-bit Arithmetic Logic Unit (ALU)

Student Name:

Waleed Umer

Instructor :

Dr Faisal Mohd-Yasin

Due Date: Friday, 6 October 2023

Contents:

Chapter1:

Introduction and project specifications

This report outlines a project within Griffith University's Integrated Circuit Design Program. The project's primary aim is to create a 4-bit Arithmetic Logic Unit (ALU) while implementing techniques to minimize layout area, reduce delay, and optimize power consumption. The project's development takes place using the Electric VLSI Design System software.An Arithmetic Logic Unit (ALU) is a digital device that performs both arithmetic and logical operations, including addition, subtraction, logical AND, logical OR, minimum and maximum selection, and more.The project employs various logic gates to construct the ALU. It supports multiple logical operations detailed in Table 1. Inputs A and B consist of 4-bit values, specifically labeled A0 to A3 and B0 to B3. Outputs are designated as R0 to R3 for most operations, but for multiplication, which requires 8-bit outputs, additional outputs R4 to R7 have been included. The ALU design also includes three selectors (S0, S1, and S2) to facilitate operation selection. Notably, subtraction, negative value computation for A and B, and multiplication results are all represented in two's complement form.

Table 1 Mode of Operations for designed ALU

S2	S1	S0	Arithmetic Operation	Output	C	V
0	0	0	Addition of A and B	A+B	0/1	0/1
0	0	1	Subtraction of B from A	A-B	0/1	0/1
0	1	0	Left shift B by one bit	Left B (put zero in the right most bit)	0	0
0	1	1	Negative A	0-A	0	0
1	0	0	Negative B	0-B	0	0
1	0	1	Absolute(A-B)	|A-B|	0/1	0/1
1	1	0	Multiplication of A and B	AxB	0	0
1	1	1	Minimum selector	Min A, B	0	0

The ultimate representation of the designed ALU is depicted in Figure 1. In this image, two additional outputs, namely "cout" (carry) and "V" (overflow), are visible. These outputs serve as indicators of carry and overflow for operations such as addition, subtraction, and absolute. Table 1 provides a comprehensive overview of these outputs' values. Notably, for all operations except for addition, subtraction, and absolute, both "C" (cout) and "V" are consistently set to 0. However, the values of "C" (cout) and "V" for the three operations mentioned earlier (adder, subtraction, and absolute) are result-dependent, thus capable of assuming values of either 0 or 1.

It's essential to highlight that the MOSIS CMOS technology was employed for the design and implementation of this ALU, showcasing the choice of technology underpinning this project.

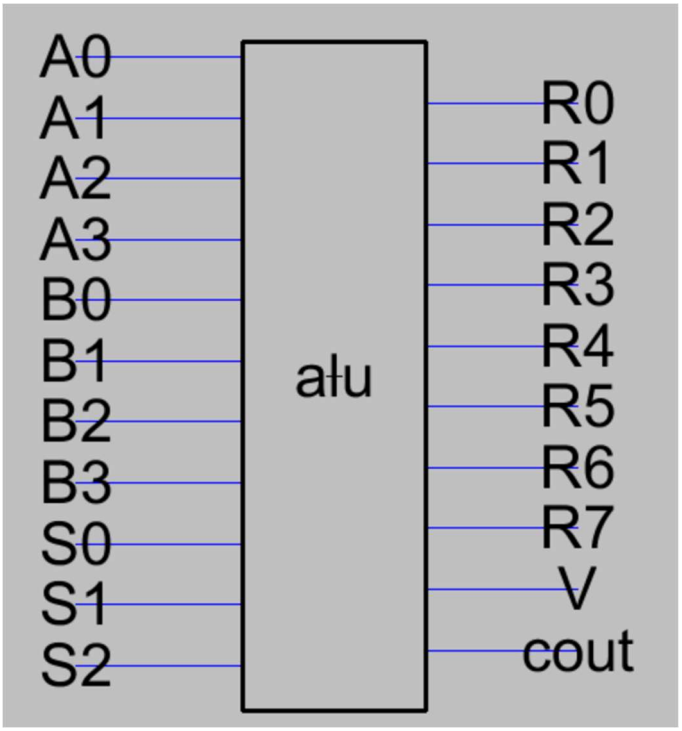

Figure 1 ALU Icon

Chapter 2:

Theory and basic design from literature

All the gates are built from scratch.It is imperative that pMOS transistors have twice the width of nMOS transistors to achieve identical beta values. This equalization is essential because pMOS transistors inherently exhibit lower mobility due to the higher mobility of electrons in silicon compared to that of holes. Typically, the mobility ratio falls within the range of 2 to 3, as stated in [1]. In most cases, the mobility ratio is considered to be approximately 2. This mobility ratio is crucial because if the beta values of nMOS and pMOS transistors are not equal, it can lead to a shift in the switching threshold away from VDD/2, affecting the inverter's normal operation.

The ratio of $\beta p/\beta n$ is set to one, as depicted in Figure 2. The choice of pMOS and nMOS transistor widths differs when it comes to the design of NAND gates. In NAND gates, pMOS transistors are connected in parallel, obviating the need for them to have widths twice that of nMOS transistors.

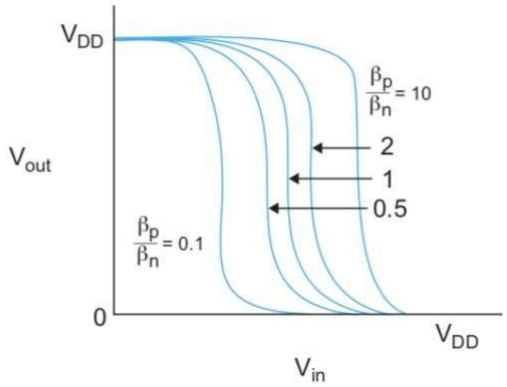

Figure 2 Transistor characteristics of inverters [1]

As highlighted in [3], it's worth noting that increasing the width of transistors at the current stage can effectively reduce delay. Importantly, the length of the transistors remains unaltered and fixed at 2, which aligns with the minimum length requirement for the MOSIS CMOS technology selected for this project. Therefore, by adhering to the minimum length while modifying the width, it becomes feasible to preserve the operational speed, as indicated in [1].

Wires:

As detailed in [1], it's important to recognize that wires within a circuit exhibit characteristics such as resistance and capacitance per unit length. Various model approximations, including the L-model, π-model, and T-model, are employed to represent these wire characteristics. Notably, the π-model, which comprises three segments, has proven to be superior to the other two models, L-model and T-model, demonstrating results with a high degree of accuracy (within 3% deviation) during simulation. In contrast, achieving similar accuracy with the L-model necessitates the use of 100 segments.

In the final design of both the ALU and multiplier, the π-model is employed for cases requiring longer wires. This choice is driven by the recognition that wires with reduced capacitance and resistance contribute to decreased power consumption within the design.

To mitigate the potential for crosstalk issues among wires, measures have been taken to increase the spacing between wires. Moreover, in certain circuit configurations such as negative, absolute, and minimum selector circuits, a wire shielding topology has been implemented to effectively reduce crosstalk problems, as depicted in Figure 3.

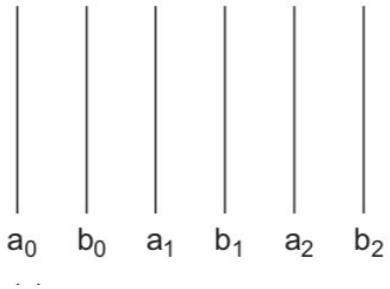

Figure 3 Shielding topology [1]

The minimum contact size, which is set at 4, plays a crucial role in minimizing the resistance and capacitance associated with pins used to connect various metals together.It's worth noting that this design choice aligns with the formula for power as outlined in the lecture notes. This approach not only optimizes resistance and capacitance but also serves as a valuable strategy for reducing overall power consumption.

$$P_{dynamic} = P_{switching} + P_{shortcircuit}$$

And $P_{switching} = CV_{DD2}f_{sw}$

It is undeniable that long wires can introduce delays in the circuit. To mitigate the impact of these long wires, smaller blocks have been specifically designed for large circuits, such as the minimum selector and multiplier. Consequently, the usage of long wires within each circuit is minimized, leading to a reduction in the number of pins required for metal layer changes in scenarios involving wire overlap. This strategic approach contributes to more efficient circuit design.[4]

Chapter 3:

Design & Simulation

In this project, a significant focus was placed on the redesign of all logic gates utilized, with none of the previous logic gates from the laboratory assignment being incorporated. This decision was driven by discrepancies in transistor widths. In the laboratory assignment, the transistor width was determined as 10+ the student's ID, and the pMOS transistor width wasn't set to be twice that of the nMOS transistor. Consequently, this resulted in variations in the speeds of holes and electrons, leading to slower pMOS transistors. To address this challenge, transistor widths of 10 for pMOS and 5 for nMOS were employed to rectify the observed mobility ratio problem during the laboratory assignment. A noteworthy comparison was conducted by designing an Xor gate with a pMOS width of 20 and an nMOS width of 10 to assess the impact of larger transistor widths on time delay. The analysis revealed an increase in drive for the subsequent stage in the circuit when larger transistor widths were utilized.

In this project, a primary objective was the reduction of delay time. This optimization aimed to enhance overall circuit performance by decreasing the delay of the critical path. Among the various circuits designed, the multiplier and minimum selector posed challenges due to their increased complexity, involving numerous gates. To overcome these challenges, a preference was given to nand gates over and gates and or gates. This preference stemmed from the lower Logical effort (g) exhibited by 2-input nand gates, standing at 4/3 compared to the 5/3 for and gates. Additionally, and gates included inverters and nand gates, so opting for nand gates not only improved performance but also reduced the layout area. A prime example of this optimization was evident in the new full adder design (Figure 4), which exclusively featured nand and Xor gates. This resulted in reduced power consumption and layout area due to the decreased number of required pins. Furthermore, the delay was minimized because nand gates exhibited a lower logical effort than and and or gates.

Another critical aspect of this project was the reworking of the multiplier, particularly the M block, which incorporated Xor and full adder logic gates. By adopting the new full adder design, it was possible to significantly reduce power consumption, layout area, and time delay for the multiplier.

Finally, it's important to revisit the formula for dynamic power, as discussed in Chapter 2, to further underscore the significance of these optimizations in terms of power efficiency.

$$P_{dynamic} = P_{switching} + P_{shortcircuit}$$

Indeed, the design of the multiplexer plays a pivotal role in optimizing switching power, which in turn contributes to a reduction in dynamic power consumption. An efficient multiplexer design can significantly impact the overall power efficiency of the system.

In the context of the project, two different metals are employed for connecting transistors within each logic gate and for establishing connections between the inputs and outputs of various blocks. Metal 3 is specifically designated for connecting the VDD (power supply)

and GND (ground) of blocks associated with different operations and the final ALU. This strategic allocation of metals helps in effectively managing power distribution within the system, enhancing its overall performance, and reducing power consumption.

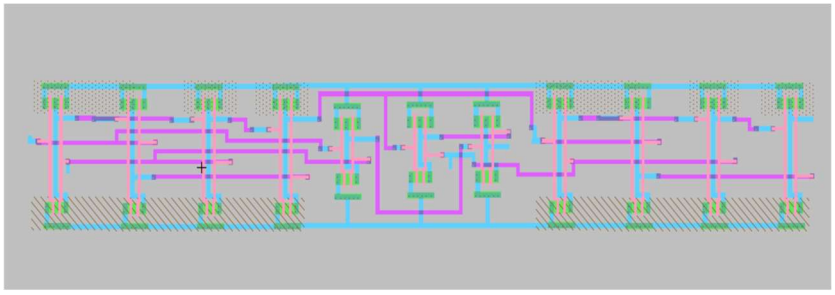

Figure 4 Full adder with pMOS width twice nMOS width for Xor

In the majority of the circuits, a consistent layout strategy has been applied where all the inputs are grouped together on one side, while the outputs are situated on the opposite side. This deliberate arrangement is designed to enhance processing speed.For instance, Figure 5 illustrates the layout of a 2 by 1 multiplexer, which is composed of 'nand' gates and inverters, replacing the conventional use of 'and' and 'or' gates. In the case of an 8 by 1 multiplexer, it is constructed by combining two 4 by 1 multiplexers and one 2 by 1 multiplexer. This modular approach is employed to select the desired operation as the ALU's output. The schematics and layouts for both the 4 by 1 multiplexer and 8 by 1 multiplexer are available in Appendix 1.

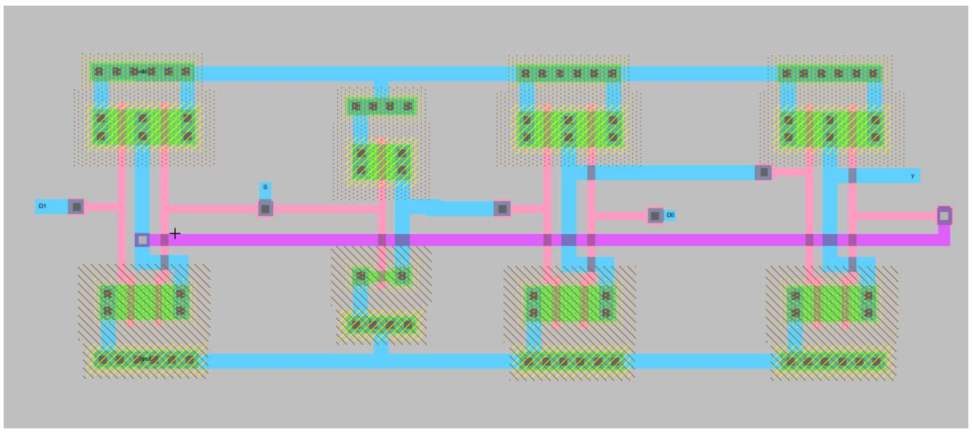

Figure 5 2 by 1 multiplexer layout

In both schematic and layout designs, a unified approach is applied, connecting all the ground and power nodes together. Specifically, the names "Vdd" and "Gnd" are consistently employed as power and ground references in each circuit. All inputs and outputs are exported in both schematic and layout representations, ensuring a seamless integration of components.

Rigorous design rule checking (DRC), electrical rule checking (ERC), and netlist connectivity checking (NCC) have been carried out meticulously to guarantee the absence of errors within each circuit and the final project.

Upon completing the design of each circuit, comprehensive digital design validation has been performed to ensure the accuracy of circuit outputs, verifying their alignment with expected

results. During the initial stages of simulation, a crucial step involves setting all inputs to both '1' and '0' to establish the correct initialization of the circuit.

In the final design of the ALU, three selectors are provided as inputs, each having values corresponding to the operations detailed in Table 1, encompassing a total of 8 different operations. This chapter presents the conclusive simulation results for the ALU, confirming its functionality and correctness.

The blocks designed for ALU:

Adder and Subtractor:

The integration of Adder and Subtractor functionalities into a single circuit design streamlines the overall architecture. This consolidated design includes 'full adder' and 'Xor' gates, providing a versatile platform for both addition and subtraction operations. The input 'M' is intelligently routed to serve as one of the inputs for the 'Xor' gates, and it also connects to the carry-in (cin) input of the initial 'full adder.' This cascading approach ensures that for subsequent 'full adders,' the carry-out (cout) of the previous 'full adder' functions as the carry-in (cin).

Additionally, the design incorporates an essential feature for calculating overflow in both addition and subtraction scenarios. This involves the inclusion of an extra 2-input 'Xor' gate, which receives the cout values from the last two 'full adders.' This mechanism is crucial for accurately determining overflow conditions, ensuring the robustness of the Adder and Subtractor circuit.It's important to note that all power and ground connections for all the gates are consolidated into a single power and ground reference. Figure 18 provides a schematic representation of the Adder and Subtractor block, offering a visual depiction of this circuit configuration.

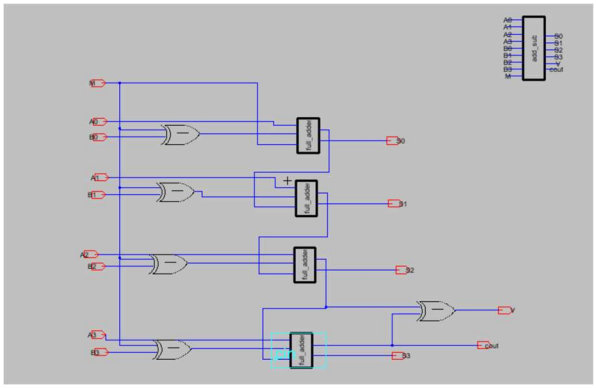

Figure 6 adder_subtractor_schematic

In Figure 7, which represents the Layout design for the Adder and Subtractor block, it is evident that the 'full adders' and 'Xor' gates have been positioned in close proximity to one another. This meticulous arrangement ensures that the software does not flag any spacing errors. Furthermore, a strategic layout approach has been adopted where all the inputs are grouped on one side and all the outputs are placed on the opposite side. This design choice serves to optimize power consumption by reducing the need for additional pins, which would otherwise be necessary to mitigate wire crosstalk and accommodate longer wires. Longer wires would contribute to increased wire capacitance, and more pins inherently possess larger capacitance, both of which have the potential to elevate power consumption. As previously mentioned, higher capacitance values are associated with increased power consumption.

Wire 3 has been specifically designated for connecting power and ground sources. Within the 'full adders,' you can observe the final full adder, as depicted in Figure 4, while the 'Xor' gates featured in the layout are the optimized versions, particularly the second 'Xor' gate showcased in Appendix1, Figure 64. Additional figures and details are available in Appendix 2.1 for reference.

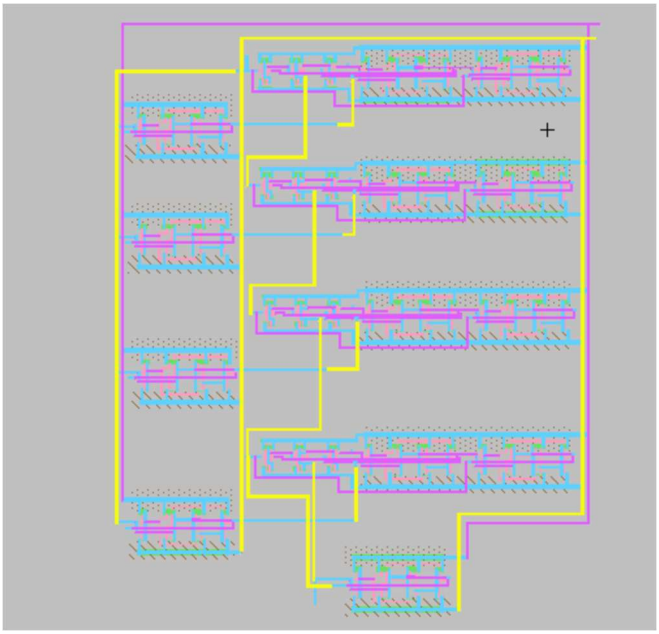

Figure 7 Layout of adder and subtractor

The simulation results clearly indicate that the delay for both addition and subtraction operations is approximately less than 1 nanosecond. However, it's noteworthy that the delay for subtraction is slightly longer compared to addition. In this specific simulation scenario, the inputs A and B are set to 1111 and 1111, respectively. The operation performed by the designed circuit depends on the value of 'M': when M=0, addition is executed, and when M=1, subtraction is performed (Figure 17).Additionally, the combined delay for both addition and subtraction operations is less than 2 nanoseconds. It's worth highlighting that all the outputs, except for S0, exhibit a delay of approximately 1 nanosecond.
For reference, the stimuli used for the 'adder_sub' module are included with the report as a separate file. This module functions correctly, and for the input provided (1111 for both A and B), it yields an output of 0000, confirming the accuracy of both addition and subtraction operations.

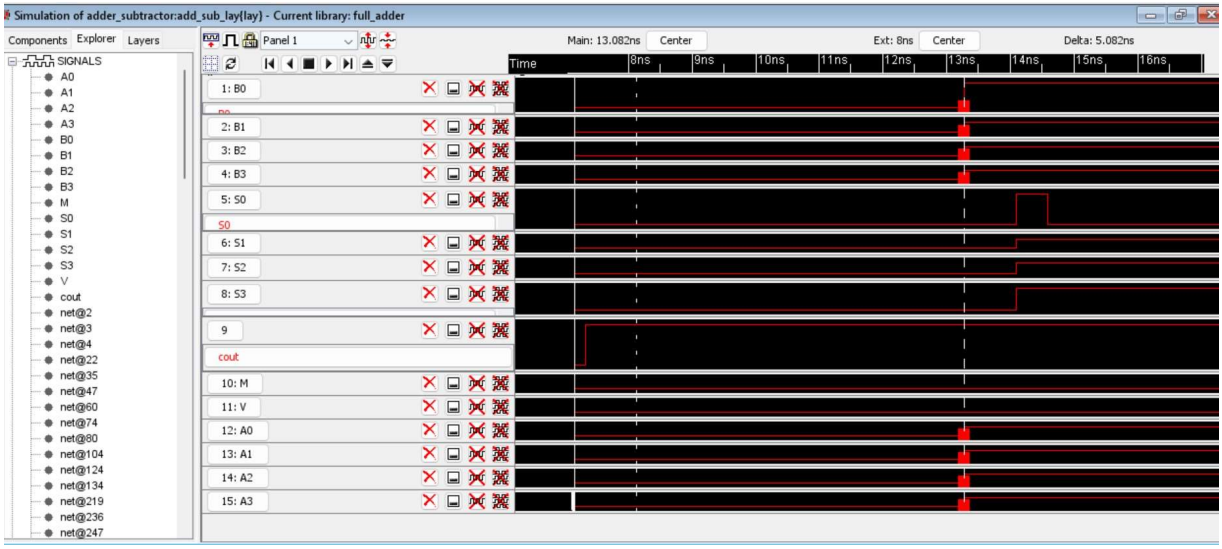

Figure 8 Addition in adder_sub block (A= 1111 and B= 1111, the output is 1110 with cout 1 and v is equal to 0)

Left shift B:

To achieve the left shift functionality for B, the design employs 'inverters' and 2-input 'nand' gates. Initially, the idea was to consolidate all the 'nand' gates and inverters into a single layout and connect them together. However, this approach resulted in a circuit design that occupied a substantial amount of area within the final design.

To optimize space utilization, a different strategy was adopted, where a 1-bit shift was designed first, and then four 1-bit shifts were combined to create the Left shift. This revised layout required significantly less area. 'Nand' gates were preferred over the use of 2 'and' gates and an 'or' gate to minimize space consumption, and as mentioned earlier, 'nand' gates offer the advantage of lower logical effort, contributing to reduced delay.

Moreover, a deliberate layout choice was made to position the outputs and inputs on opposite sides. This layout optimization minimizes the need for additional pins, which would be required if inputs and outputs were placed on the same side to avoid crosstalk issues, as demonstrated in Figure 13.

For reference, the schematic and layout of the 1-bit shift, which is employed in the 4-bit left shift, along with an additional layout of the 4-bit left shift that provides an internal view of the design, are available in Appendix 2.2.

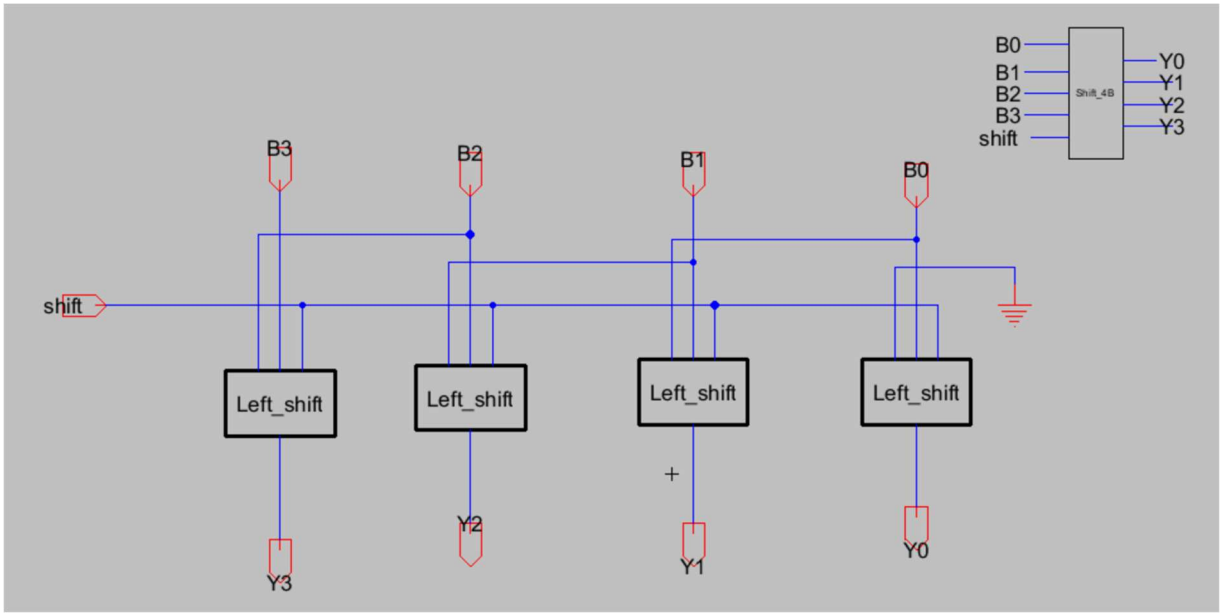

Figure 9 4-bit Left shift B by using four 1-bit shift gates(sch)

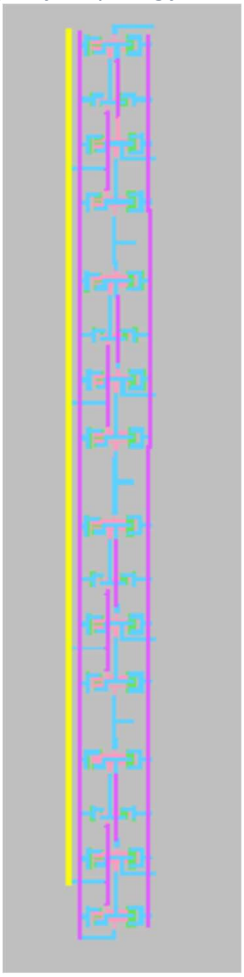

Figure 10 Left_Shift layout

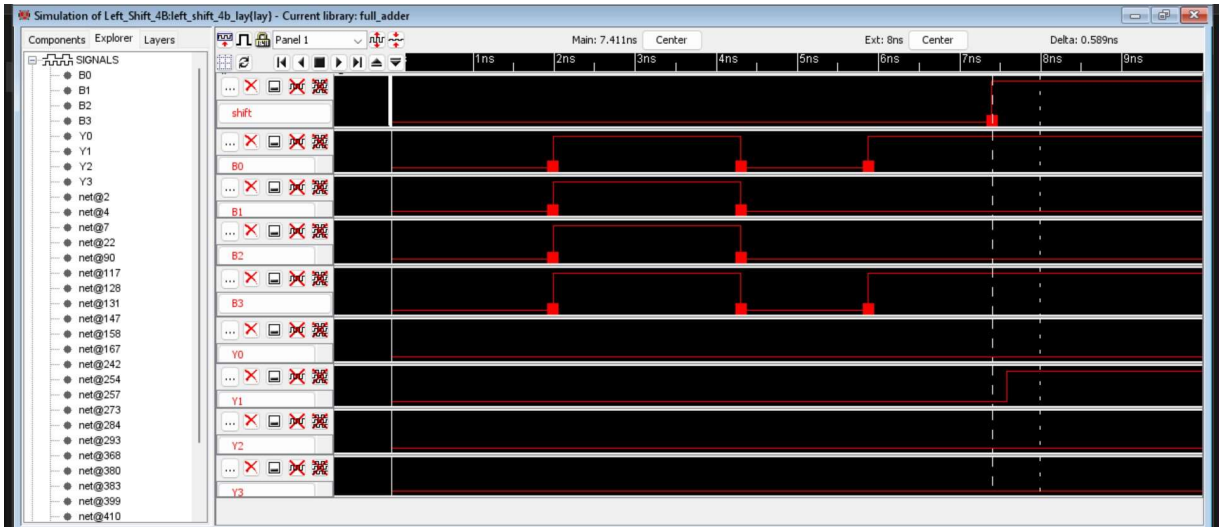

Figure 11 4-bit left shift Sim B=1001, output=0010

During the Friday demonstration, an issue was identified with the output Y0 of the left shift operation. This discrepancy was attributed to one of the inputs, B0, which was erroneously connected to Vdd instead of being grounded as intended.This issue has now been resolved, and subsequent simulation results indicate that the Left shift operation is now functioning as intended.

Negative A or B:

In the pursuit of efficiency and area optimization within the ALU, a unified circuit is devised to handle both negative A and negative B operations. Since these operations involve subtraction from 0, they necessitate the presence of both an adder and a subtractor circuit. To facilitate this, a versatile block labeled 'a' is created to select either the 4-bit input A or B based on the control signals IA and IB.For the correct operation within the final ALU, IA is assigned to connect with S0, ensuring that it remains at a logic high state (1). This choice aligns with the selectors configuration for negative A, which are set to 011. On the other hand, IB is designated to connect with S2, harmonizing with the selectors setup for negative B within the ALU, which are configured as 100.The schematic and layout of the 4-bit Left shift B, as well as block 'a,' are diligently crafted to optimize both functionality and spatial efficiency. In the layout of the 'a' circuit, a thoughtful arrangement places 'nand' gates adjacently, effectively minimizing the overall area usage.Furthermore, the performance of this design shines through as the delay of outputs is consistently under 2 ns, as clearly demonstrated in Figure 14. This meticulous approach ensures that negative A and negative B operations are handled with precision and efficiency within the ALU.

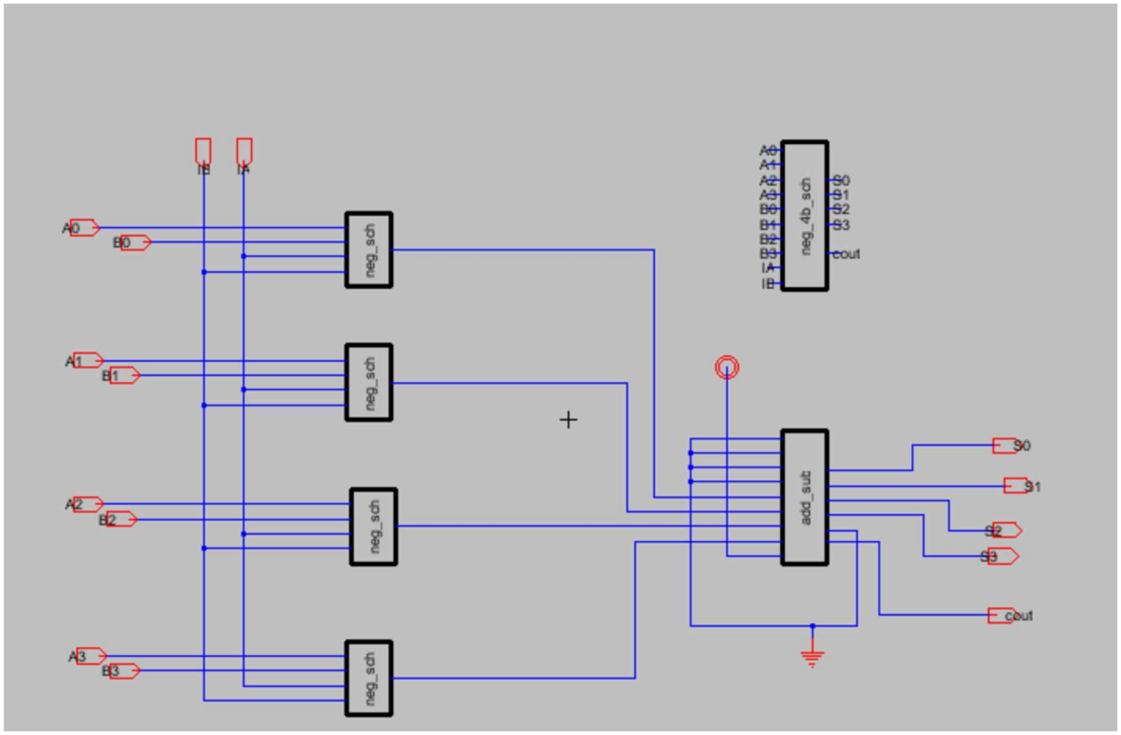

Figure 12 Schematic of 4-bit negative A or negative B

Figure 13 Layout of 4-bit neg A / negative B

The simulation results for the individual negative circuit designs for both A and negative B are presented in Figure 17. In this particular simulation scenario, A is set to 1100, and the resulting negative A is 0100. The outputs are labeled as S0 to S3.

For your reference, the stimuli used for the negative circuit simulations are included with the report in the stimuli folder, providing a comprehensive overview of the input conditions and expected outcomes. These simulation results confirm the correct functionality of the negative circuit designs.

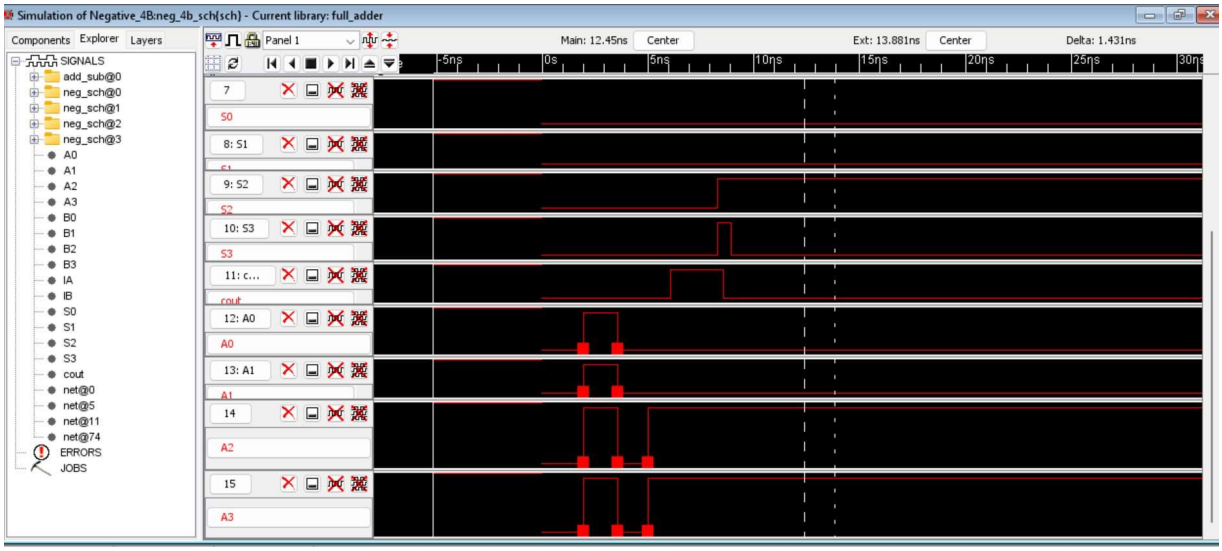

Figure 14 Negative circuit simulation 4 input A= 1100 (two's complement of 4-bit binary strings)

Absolute (A-B):

The design approach for the absolute operation involved finding the minimum number between A and B before performing subtraction. This strategy is more efficient in terms of both area utilization and circuit complexity compared to the initial idea of subtracting B from A and then determining whether the result is negative, which would have required a larger circuit.Figure 19 to Figure 20 illustrate the schematic, layout, and simulation results for the absolute operation. The 'block a' component plays a crucial role in this design by selecting the input that has the maximum value. The minimum selector circuit, in addition to the absolute output, also provides outputs indicating whether A is equal to B or if A is less than B.

The process of selecting the input for the adder/subtractor block is as follows: If the output A<B is equal to 1, indicating that A is less than B, then the inputs of A in circuit 'a' are forced to be NANDed with 0, and this does not affect the selection of B. The inputs of B are NANDed with 1, yielding the opposite of the value of input B. These outputs are then combined to determine the exact value of B. The process is reversed when the output A<B is equal to 0.

For reference, stimuli for the absolute operation are included with the report, demonstrating the expected output for specific input values. The stimuli provided in Figure 17 show the output for A=1100 and B=1001, resulting in an absolute value of 0011.The schematic and layout of 'block a,' as well as an additional layout of the absolute operation, including the internal components of the minimum selector and adder/subtractor, are available in Appendix 2.4. This design approach optimizes both area usage and circuit efficiency.

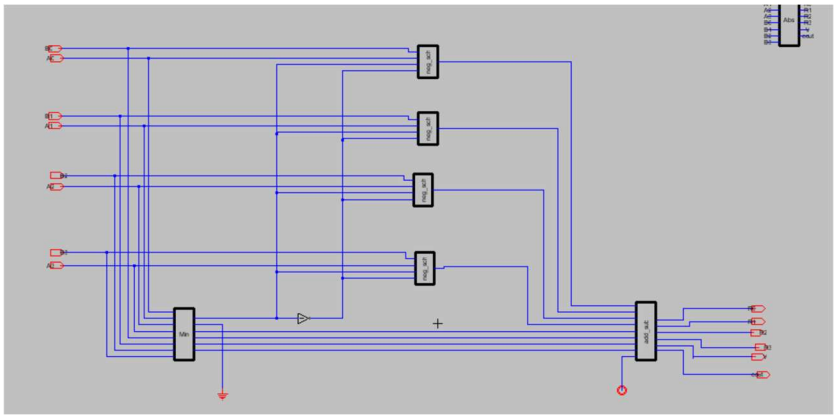

Figure 15 Absolute Schematic

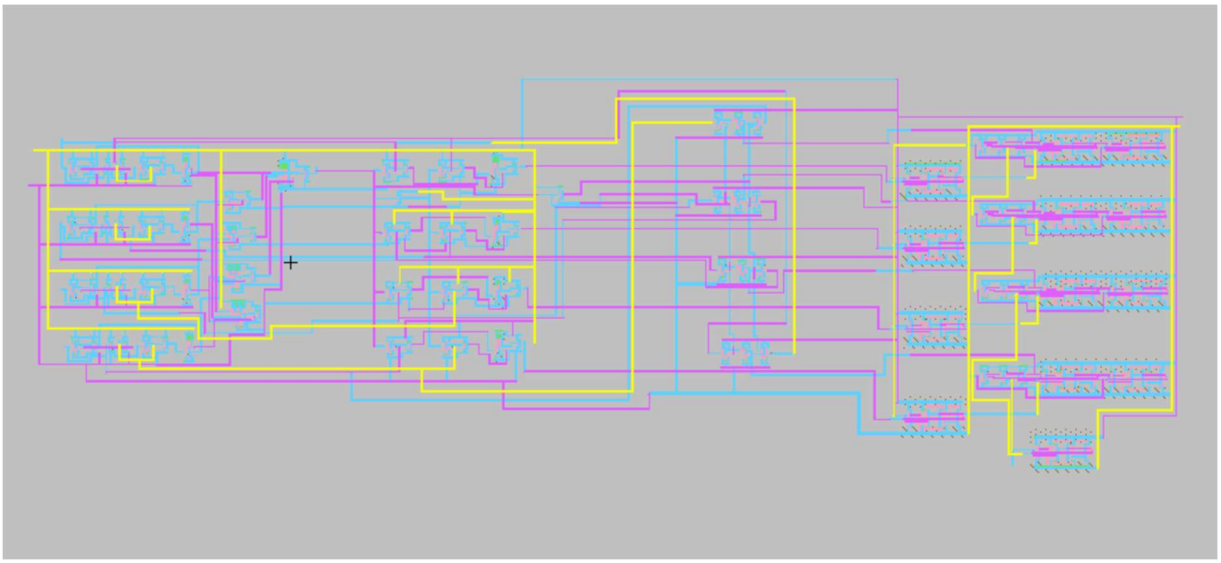

Figure 16 Absolute layout

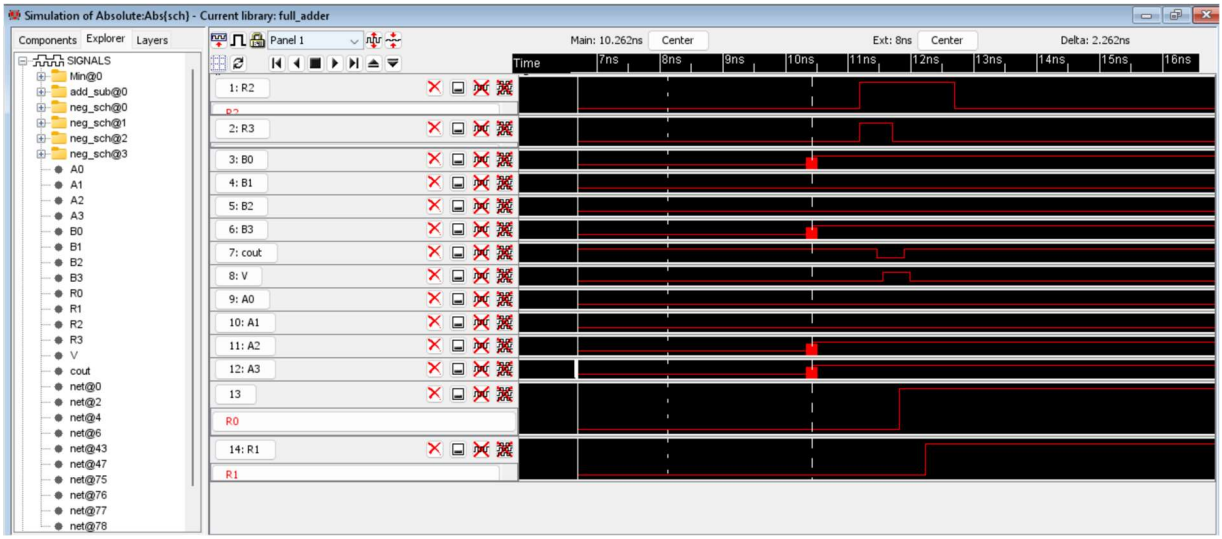

Figure 17 Absolute (A-B)-Simulation outside the ALU

Multiplication:

The design of the multiplication operation within the ALU entails the strategic integration of 'and' gates, adders, and the incorporation of an additional 4-bit output block. Initially, the concept for multiplication is laid out in Appendix 2.4. A detailed examination of the schematic, layout, and simulation results for the multiplication operation is presented in Figure 18 to Figure 19.To ensure the accurate execution of the multiplication operation when the selectors are set to 110, a dedicated block labeled 'M' has been meticulously devised. This block plays a pivotal role in guaranteeing that all the outputs deliver precise multiplication results. In this intricately designed operation, 4-bit 'and' gates come into play, featuring 3 inputs dedicated to selectors and an additional input responsible for receiving the output of the multiplication operation. Xor gates are ingeniously employed to enforce S0=0, with one Xor gate for the input of the 'and' gate and another for the Xor operation between S0 and S1, set to 1 to enable multiplication.A comprehensive insight into the schematic and layout of block 'M,' as well as an exploration of the Extra Block for multiplication, is available in Appendix 2.5. The layout of the multiplication operation is thoughtfully executed, as depicted in Figure 23. The 'M' blocks and 'and' gates are deliberately placed in close proximity, effectively minimizing area consumption. The inclusion of $\pi - model$ wires significantly contributes to the reduction of delay by mitigating resistance and capacitance issues.An astute layout strategy is implemented, ensuring that all inputs and outputs are conveniently aligned on the same side, with outputs thoughtfully positioned opposite to the inputs. This layout optimization serves to enhance power efficiency. The design approach encapsulates the essence of efficiency and precision in handling the multiplication operation within the ALU. However, during a recent demonstration on Friday, a simulation issue was identified with the multiplier circuit. The root cause of the problem was traced back to improper initialization of the circuit, resulting in unexpected behavior. Subsequent simulation results, as showcased in Figure 19, unequivocally validate that following the correct initialization, the multiplier circuit now operates as intended. This successful resolution reaffirms the design's accuracy and functionality.

For reference and comprehensive evaluation, stimuli for the multiplier schematic and layout have been thoughtfully included in the report. These stimuli provide detailed input conditions and anticipated outputs, serving as a valuable reference point for validation.

It's noteworthy that while many aspects of the circuit have been successfully addressed, there remains a specific issue concerning one output, C5 in the layout, which is still encountering an error. Further investigation and resolution efforts may be required to rectify this particular issue, ensuring the overall accuracy of the design.

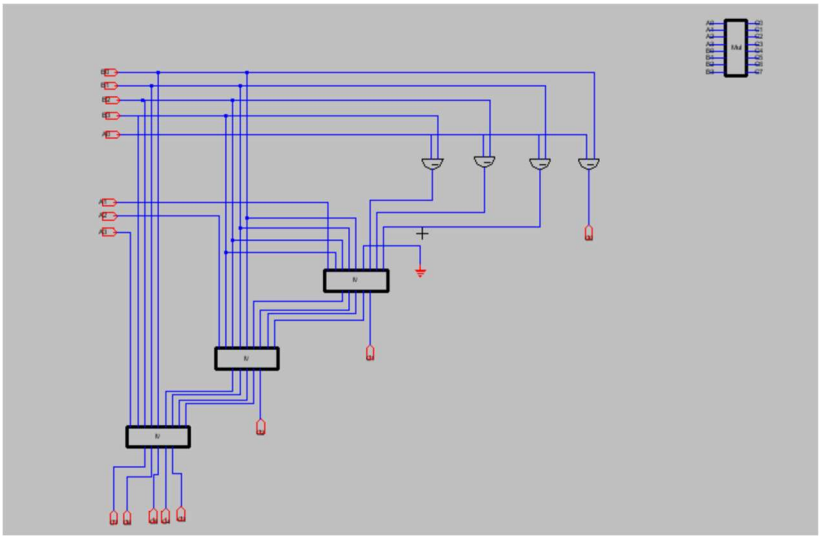

Figure 18 Multiplication circuit design- Schematic

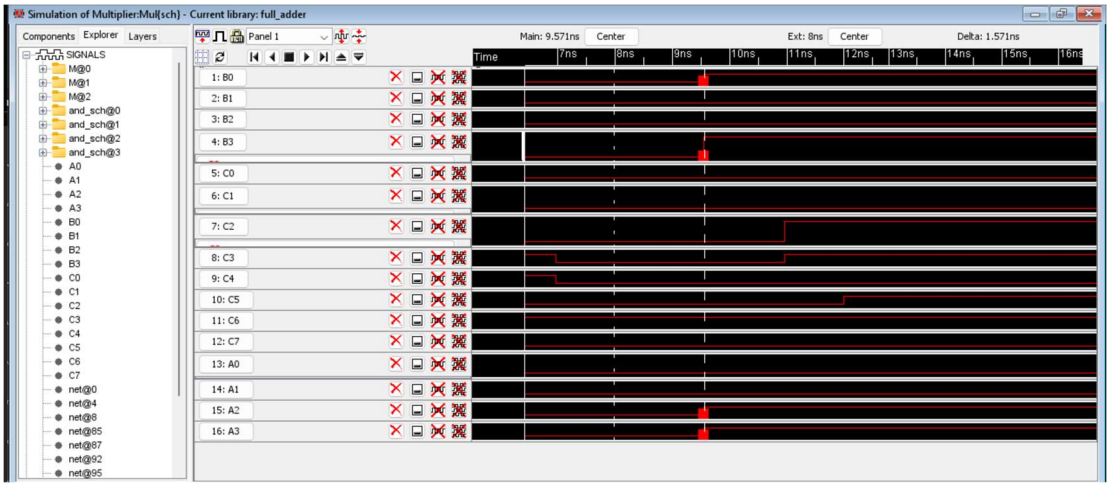

Figure 19 Sim_result of multiplication A=1111, B=1111 and out=01101100

Minimum Selector:

The creation of the minimum selector circuit required a meticulous design process involving the comparison of outputs to determine the minimum value. This involved the development of a 1-bit block circuit that could be extended to construct a 4-bit comparison circuit known as 'min,' serving as the final minimum selector. The design of the minimum selector circuit was achieved by effectively utilizing the 'min' block in conjunction with 'inverter' and 'or' gates.Figure 20 provides a comprehensive schematic representation of the final minimum selector. In instances where circuits necessitated the connection of more than two outputs to gates, a specialized technique was employed to optimize layout efficiency. This technique is notably employed in the minimum selector, which incorporates the 'compare' block. The 'compare' block is equipped with 3-input and 4-input 'and' gates, along with a 4-input 'or' gate. These gates were thoughtfully constructed by utilizing 3-input and 4-input 'nand' gates, in addition to a 4-input 'nor' gate.For an in-depth understanding of the schematic and layout designs of the 3 and 4-input 'nand' and 'and' gates, as well as the 4-bit 'nor' and 'or' gates, please consult Appendix 1. Furthermore, the schematic and layout representations of the 1-bit 'compare' block and the 4-bit 'compare' block ('min') are thoughtfully provided in Appendix 2.6.To gain insight into the layout visualization and simulation outcomes of the minimum selector, you are encouraged to refer to Figure 20 and Figure 21. These components collectively contribute to the seamless functionality of the minimum selector circuit.For your convenience and reference, stimuli corresponding to the absolute operation have been thoughtfully included within the report. It is noteworthy that both the schematic and layout designs for the absolute operation are operating in accordance with the intended specifications, reaffirming the precision and effectiveness of the circuit design.

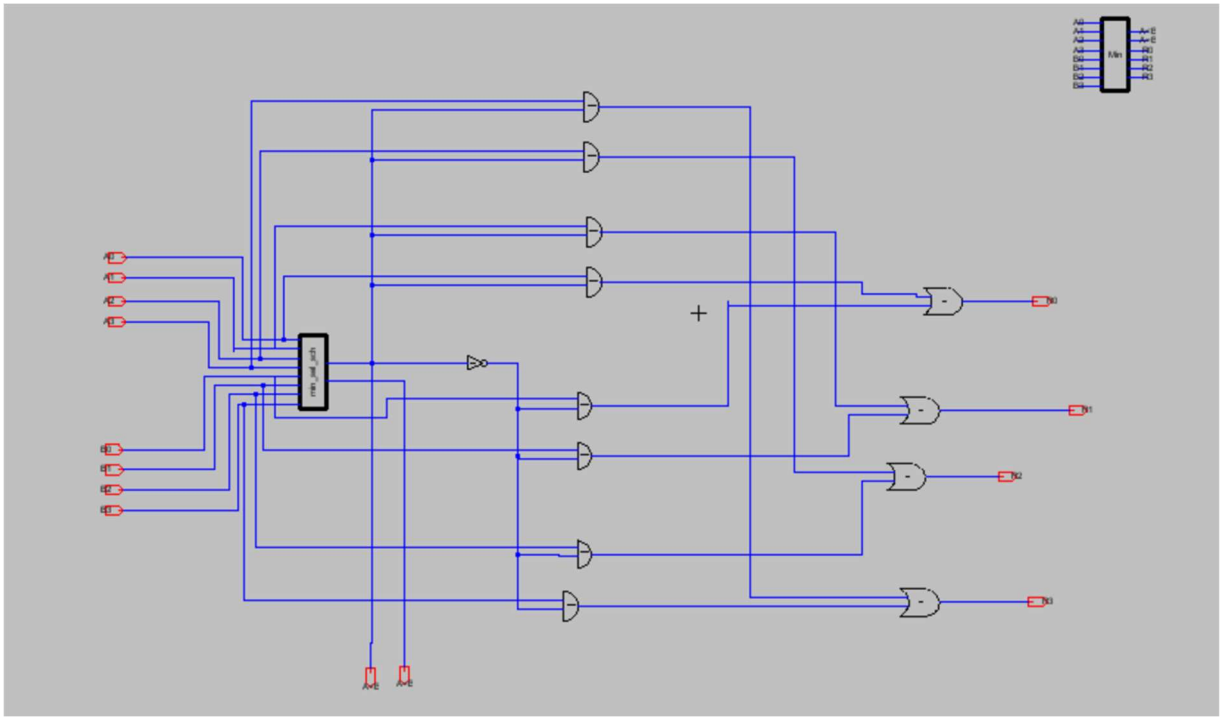

Figure 20 Sch_Minimum selector_ 4-bit A and B

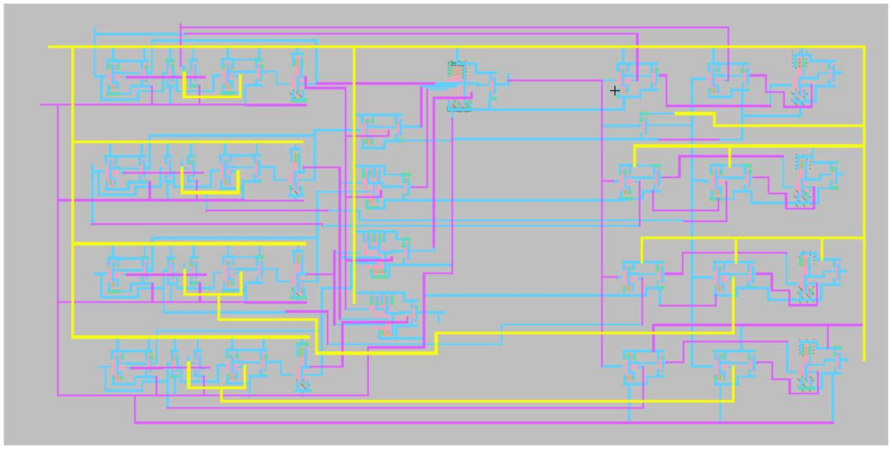

Figure 21 Layout- Minimum selector

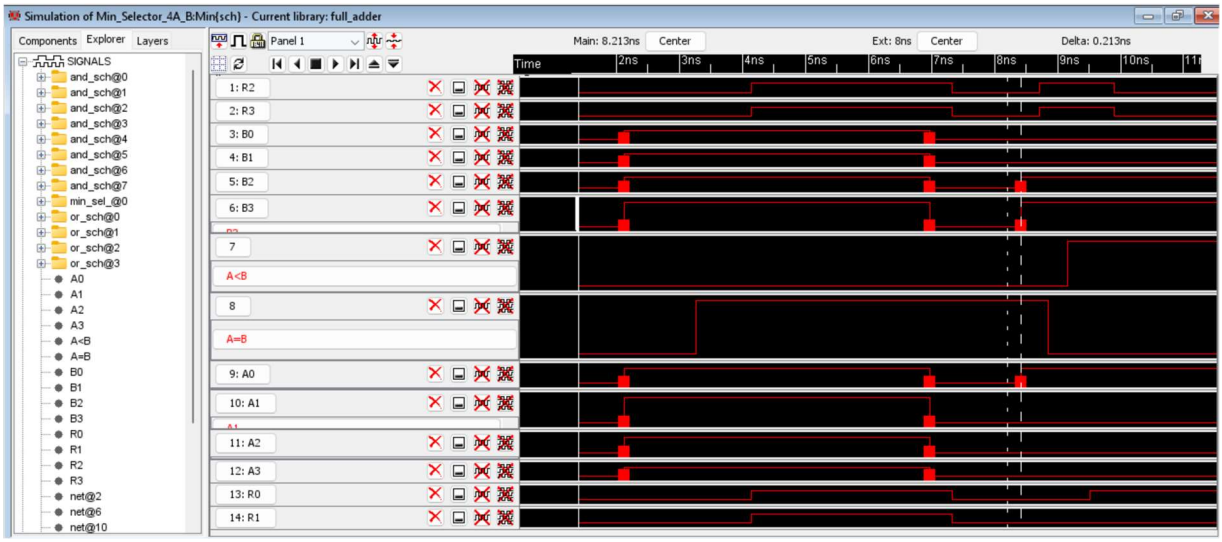

Figure 22 Simulation- Minimum selector (first inputs: A= 1111, B=1111 and out=(A=B), second inputs: A=0001, B=1100 and out=(A<B))

As previously mentioned, the schematic and layout results for the 8 by 1 multiplexer, which is crucial for selecting the ALU operation and is also used for cout and overflow (V), have been provided in Appendix 1.Furthermore, it's noteworthy that the output delay for all individual circuits is less than 2 nanoseconds. This indicates that these circuits are operating efficiently and within the desired timing constraints.

Chapter 4:

Results and Analysis

The performance evaluation of all the designed blocks essential for the 4-bit ALU, in terms of functionality and error validation (DRC, ERC, and NCC), has been meticulously conducted. Chapter 3 of this report provides comprehensive details, including the schematic, layout, and simulation outcomes for each of these blocks.The simulation outcomes, portrayed in Figure 24 to Figure 29, indicate that the total delay of outputs is approximately 1.5 nanoseconds. It is noteworthy that the total delay for all individual blocks was confirmed to be under 2 nanoseconds. This indicates that all blocks are operating effectively and well within the designated timing constraints.The incorporation of pMOS transistors with a width twice that of nMOS transistors has led to a reduction in the total delay for primary blocks such as full adders, Xor gates, and logic gates like or and nor gates. Additionally, for efficient area utilization, most blocks employ nand gates instead of and gates and or gates.Concerted efforts to minimize power consumption in most blocks involve organizing inputs and outputs on the same side and employing the smallest available pin size. This choice is made due to the lower resistance and capacitance of smaller pins, resulting in reduced power consumption.As previously discussed in Chapter 3, the formula for dynamic power is a significant consideration in power optimization, and steps have been taken to mitigate power consumption throughout the design process.

The formula for dynamic power is:

$$P_{dynamic} = P_{switching} + P_{shortcircuit}$$

The formula for switching power, including capacitance, was discussed in Chapter 2. To minimize power consumption, the design focuses on smaller pin sizes and shorter wires, which reduce capacitance. This strategy aligns with the goal of achieving lower power usage in the circuit.

And Multiplexers has an important role to play in decreasing the power consumption.In the final alu design,multiplexor selector pins were not connected according to the design .Upon review,it was found that some of the selector pins were not correctly were not connected at the right export.This issue is rectified now and selector pins are now working in accordance with the design shown in Table 2.

The final schematic design is shown in Figure 23.

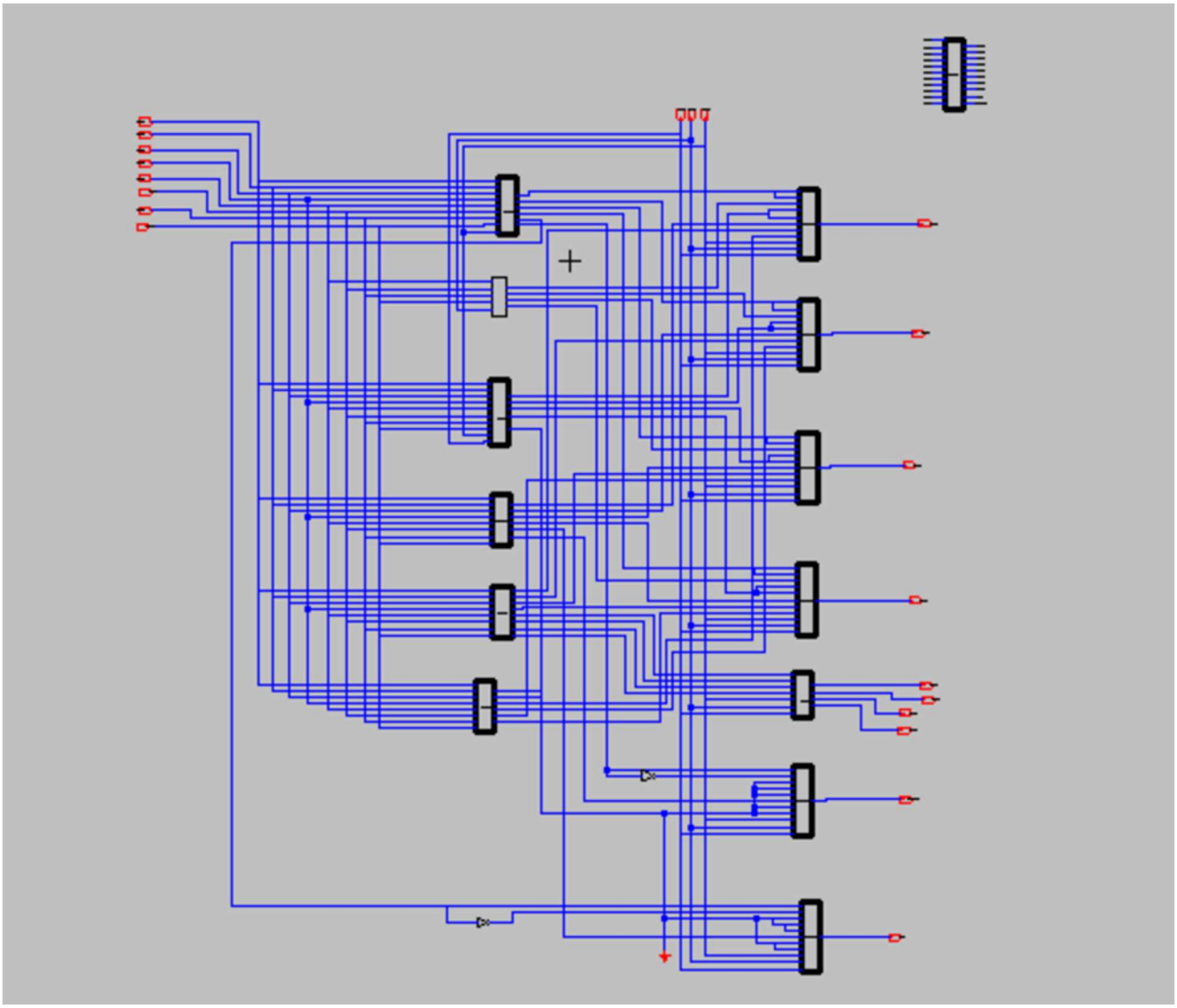

Figure 23 Final design_4-bit ALU

Figure 24 illustrates the addition operation with the selectors set to 000. In this scenario, the 4-bit inputs A and B have values of 1100 and 1001, respectively. The output of this addition operation is equal to 0101, and both the carry-out (cout) and overflow (V) are set to 1. The value of M is 0, indicating that this operation represents addition.

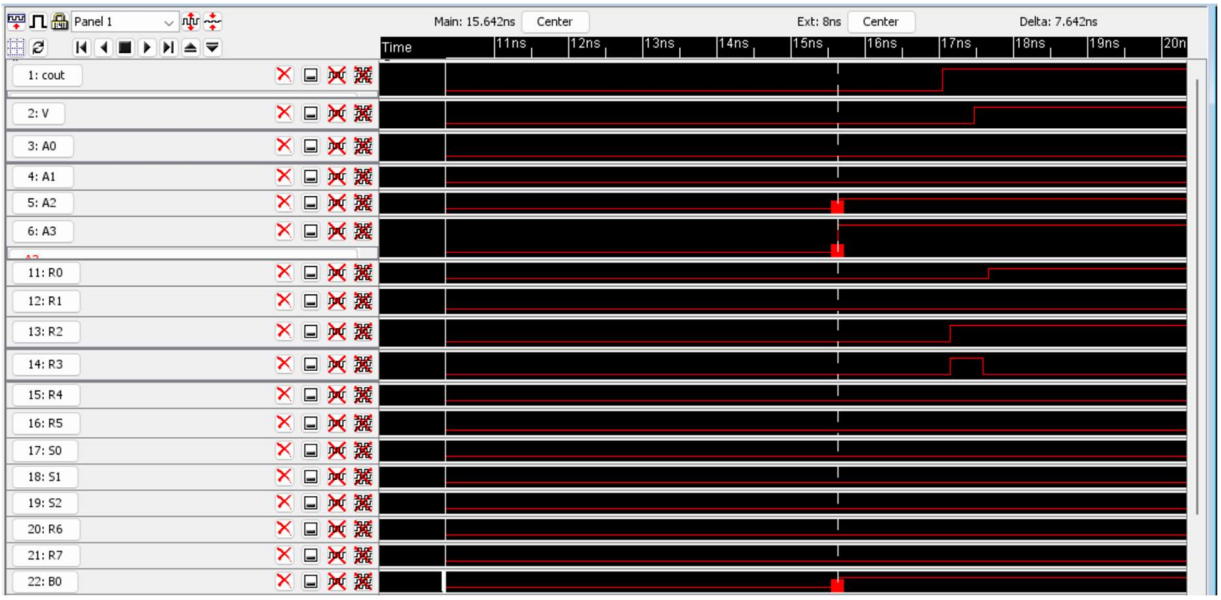

Figure 24 Add_ A and B (A= 1100 and B= 1001, the output is 0101 with cout and v are equal to 1)

Figure 25 depicts the subtraction operation with the selectors set to 001. In this case, the 4-bit inputs A and B have values of 1100 and 1001, respectively. The output of this subtraction operation is equal to 0011, with the carry-out (cout) being 0, and the overflow (V) set to 1. The value of M is 1, indicating that this operation represents subtraction.

Figure 25 Sub_ A from B (A= 1100 and B= 1001, the output is 0111 with cout is equal to 0 and v is equal to 1)

Figure 26 illustrates the left shift operation with the selectors set to 010. In this scenario, the 4-bit input B has a value of 1001. The output of this left shift operation is equal to 0010, with both the carry-out (cout) and overflow (V) set to 0. This operation represents a left shift.

Figure 26 Left shift_4-bit B(B= 1001, the output is 0010 with cout and v are equal to 0)

Figure 27 depicts the negative operation applied to a 4-bit input A. In this configuration, the selectors (S2 S1 S0) are set to 011. The 4-bit inputs A and B have values of 1100 and 1001, respectively. The output of this negative operation is equal to 0100, which represents the two's complement answer for the negation. In this case, both the carry-out (cout) and overflow (V) are set to 0.

Figure 27 Negative 4 input A= 1100 (the output represents two's complement of 4-bit binary strings)

Figure 28 illustrates the absolute operation applied to the subtraction of 4-bit inputs A and B. In this scenario, the selectors (S2 S1 S0) are configured as 101. The 4-bit inputs A and B have values of 1100 and 1001, respectively. The output of this absolute operation is equal to 0011, and it's important to note that the carry-out (cout) is set to 1, while the overflow (V) is set to 0.

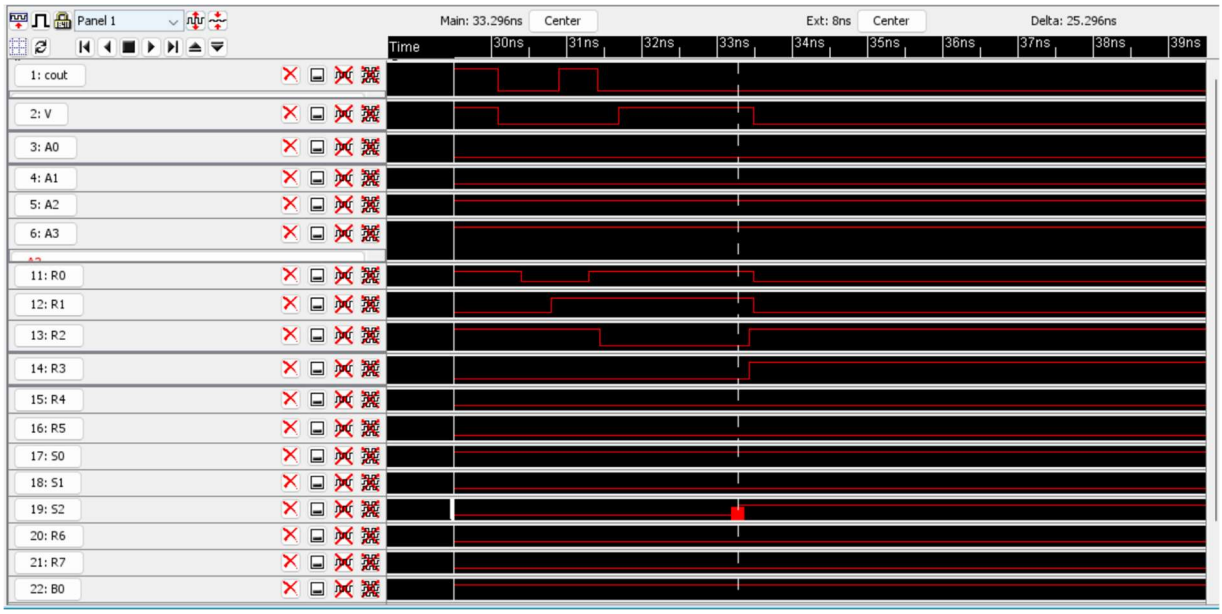

Figure 28 Absolute A-B (A=1100 and B=1001 the output is equal to 0011 cout is 0 and V is 0)

Figure 29 depicts the multiplication operation applied to 4-bit inputs A and B. In this case, the selectors (S2 S1 S0) are set to 110. The 4-bit inputs A and B have values of 1100 and 1001, respectively. The output of this multiplication operation is equal to 01101100, which is represented in eight bits. It's worth noting that the carry-out (cout) is set to 0, and the overflow (V) is also set to 0. However, it's important to mention that the multiplier is not functioning as intended in the final ALU design. The suspected issue is related to the pin connections for the multiplier in the final design. Further investigation is required to resolve this issue. There are no design rule check (DRC) and electrical rule check (ERC)errors in the design.

Figure 29 Multiplication of A and B (A= 1100 and B= 1001 the output is equal to 01100011 instead of 01101100)

Final Layout of the designed ALU is shown in Figure 30 .

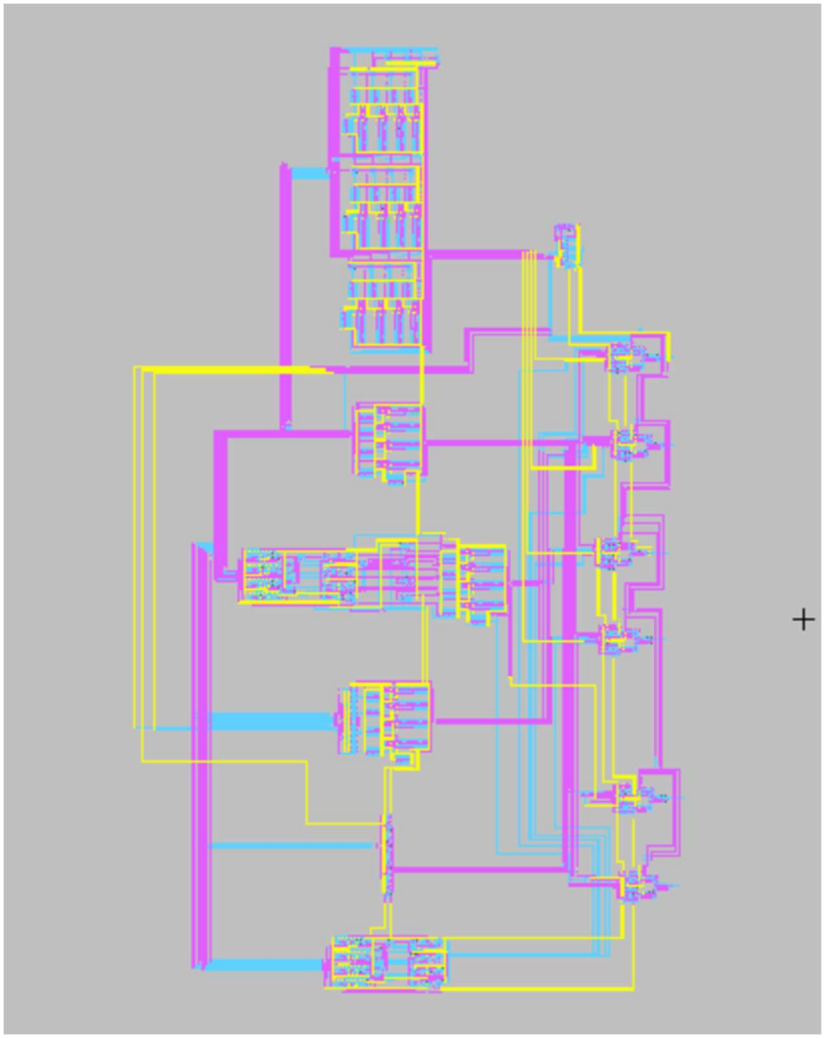

Figure 30 ALU_layout

For your reference, the stimuli for the ALU have been included in the report's stimuli folder, serving as tangible evidence of its proper functionality.

During the Friday demonstration, an issue came to light regarding the functionality of the Multiplexer (Mux). The root cause of this problem was traced back to incorrect connections of its selector pins, which were not in accordance with the intended design outlined in Table 1. However, I'm pleased to report that this issue has since been identified and resolved, and the Mux is now operating in alignment with its intended design.

It's worth noting that among all the circuits, the Multiplier is the sole component not performing as originally intended. However, it's important to emphasize that this issue is expected to be resolved through a more in-depth review and necessary adjustments.

Conclusion:

In conclusion, this project was undertaken with the overarching objective of crafting a 4-bit Arithmetic Logic Unit (ALU) endowed with the capability to execute eight fundamental operations: addition, subtraction, left shift B (two's complement), negative A, negative B, absolute (A-B), multiplication, and minimum selection. In the realm of Very Large Scale Integration (VLSI) design, this endeavor rigorously tackled three paramount optimization facets: delay, area, and power consumption.
Reducing delay was a top priority. To do this,I carefully adjusted the size of transistors and their performance ratios, making pMOS transistors twice as wide as nMOS transistors. This played a crucial role in speeding up our circuit. We also pinpointed and streamlined the most critical pathways to further reduce delay. To save space and power, we used nand gates extensively because they're more efficient than 'and' and 'or' gates. Shielding wires helped reduce interference between them.
For power optimization, we organized inputs and outputs on the same side in most blocks, which saves energy by requiring fewer connections. We also chose smaller pin sizes because they have lower resistance and capacitance, reducing power consumption. Rigorous checks were performed to ensure everything worked correctly.

In our pursuit of top performance, we included wire shielding in the final layout to cut down on delay. However, the presence of a large number of wires in our final design did lead to a slight increase in delay. Future improvements might focus on optimizing wire layouts for even better results.

In conclusion, all our circuits worked exceptionally well, except for the multiplier circuit, which needs some fine-tuning. This project successfully addressed the main concerns in designing complex circuits while achieving the desired functionality for our 4-bit ALU design.

References

[1] N.H.E. Weste and D.M. Harris, Integrated Circuit Design. (Fourth, Global ed.) Boston: Pearson, 2011.

[2] M. R. Stan, "Optimal Voltages and Sizing for Low Power", 12th IEEE Intl. Conf. on VLSI Design*, pp. 428-433, 1999*

[3] K. Singh, A. Jain, A. Mittal, V. Yadav, A.A. Singh, A.K. Jain and M. Gupta, "Optimum transistor sizing of CMOS logic circuits using logical effort theory and evolutionary algorithms," Integration, the VLSI Journal, vol. 60, pp. 25-38, 2018.

[4] K. Prudhvi Raj and Y. Syamala, "Transistor Level Implementation of Digital Reversible Circuits", International Journal of VLSI design & Communication Systems (VLSICS)*, vol. 5, no. 6, December 2014*.

Appendix:

1.Initial logic gates design (Schematic and layout):

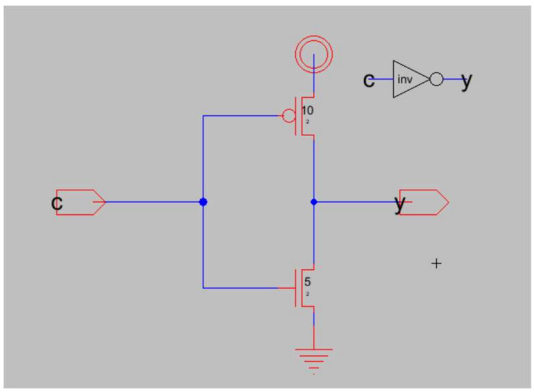

Figure 31 Inverter_ pMOS width twice nMOS width- Schematic

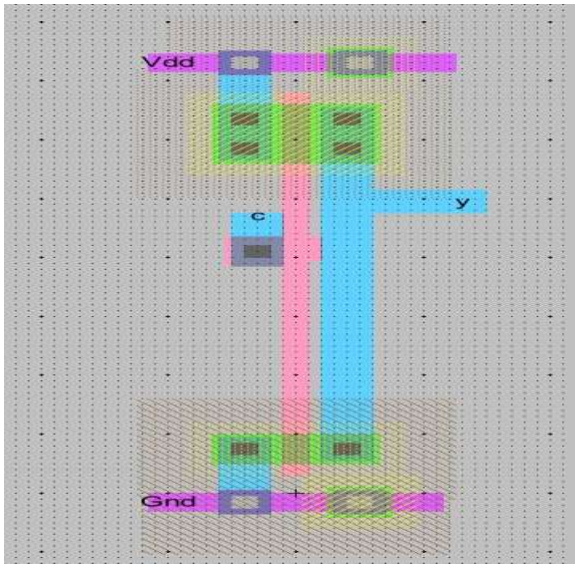

Figure 32 Inverter _ pMOS width twice nMOS width- Layout

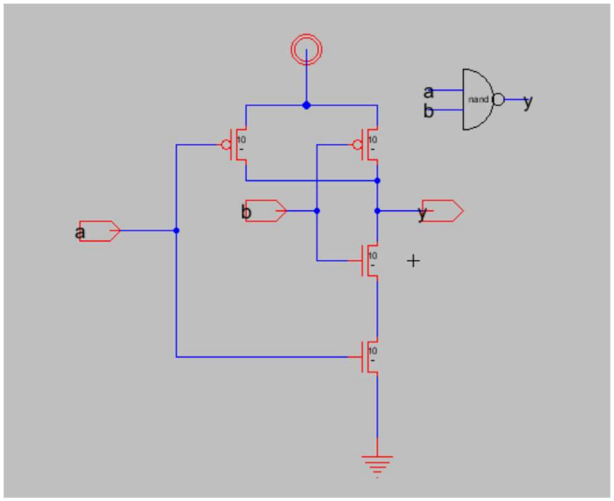

Figure 33 2_ input nand gate-Schematic

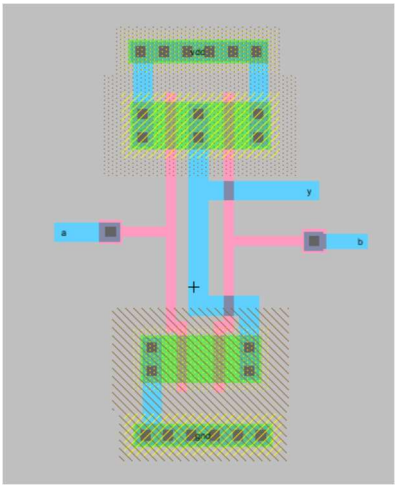

Figure 34 2_input nand gate- Layout

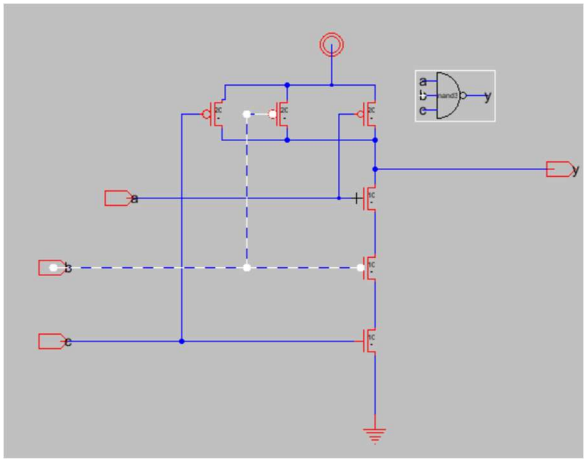

Figure 35 3_input nand gate- Schematic

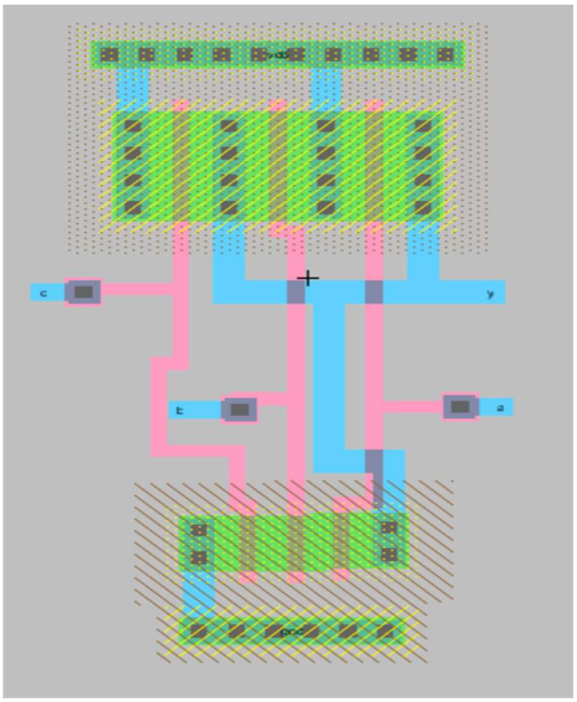

Figure 36 3_ input nand gate- Layout

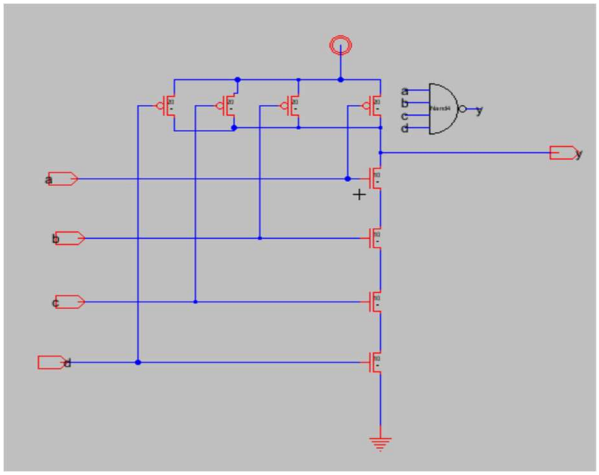

Figure 37 4 _input nand gate- Schematic

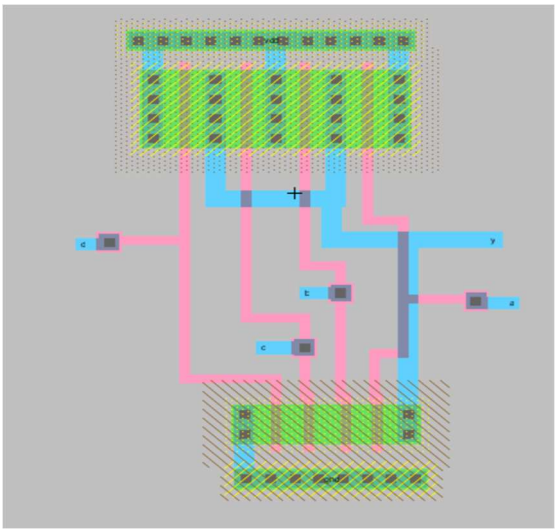

Figure 38 4_ input nand gate- Layout

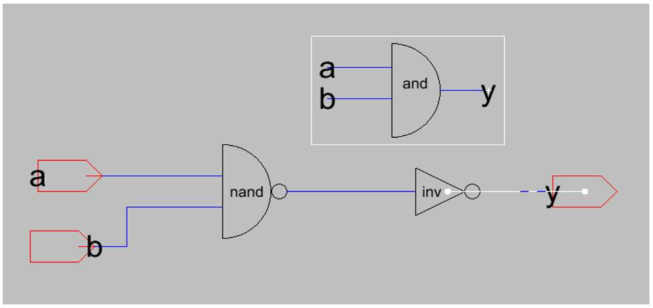

Figure 39 2_ input and gate- Schematic

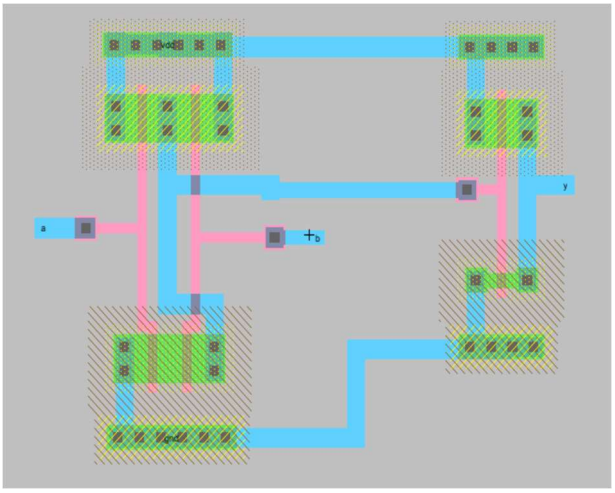

Figure 40 2_ input and gate- Layout

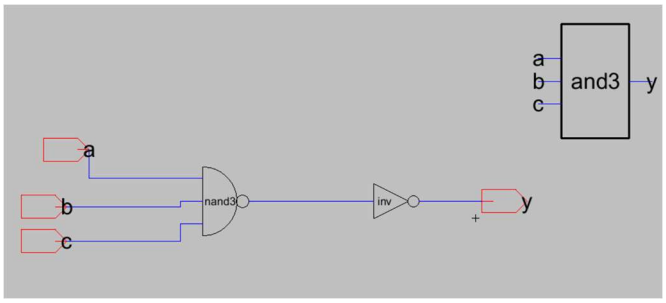

Figure 41 3_ input and gate- Schematic

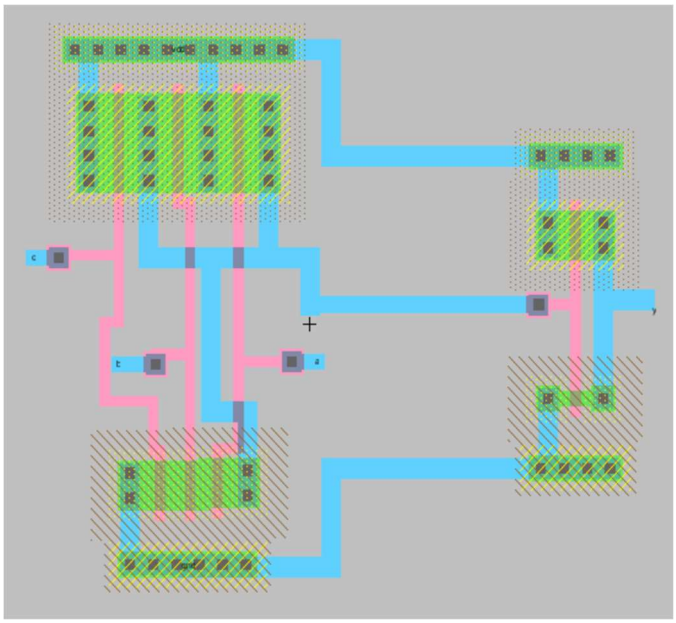

Figure 42 3_ input and gate- Layout

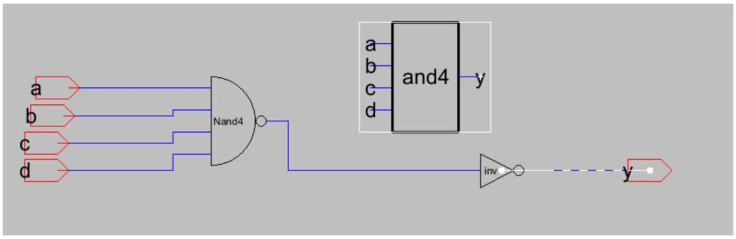

Figure 43 4_input and gate- Schematic

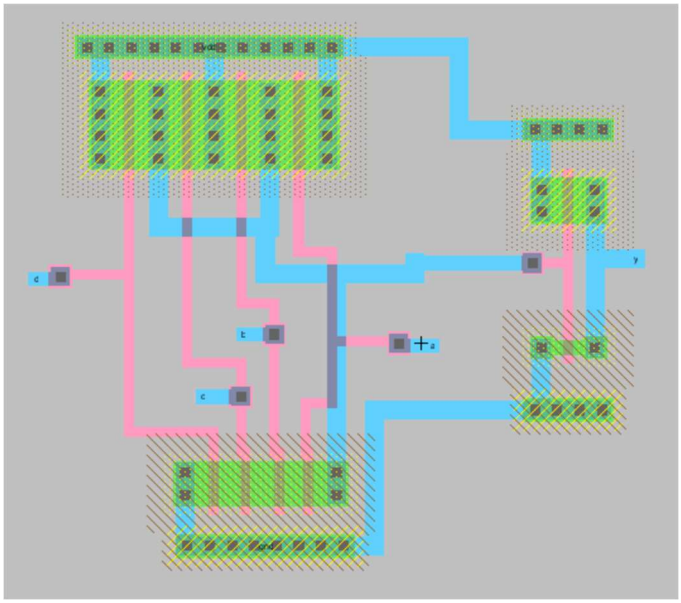

Figure 44 4 _input and gate- Layout

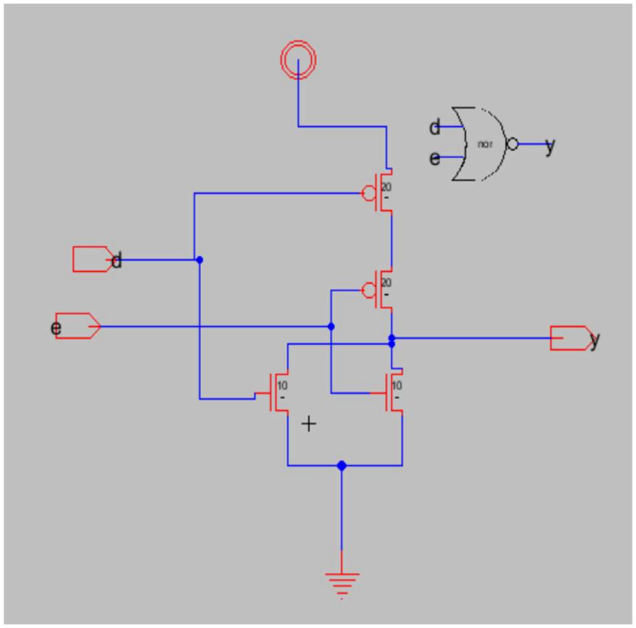

Figure 45 2 _input nor gate-Schematic

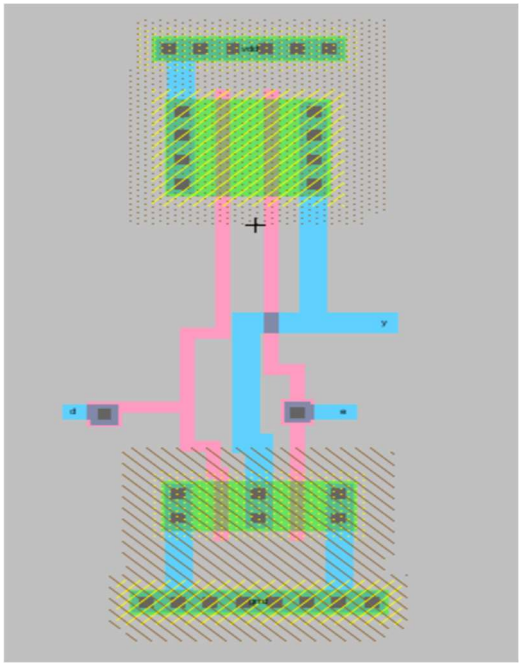

Figure 46 2 _input nor gate-Layout

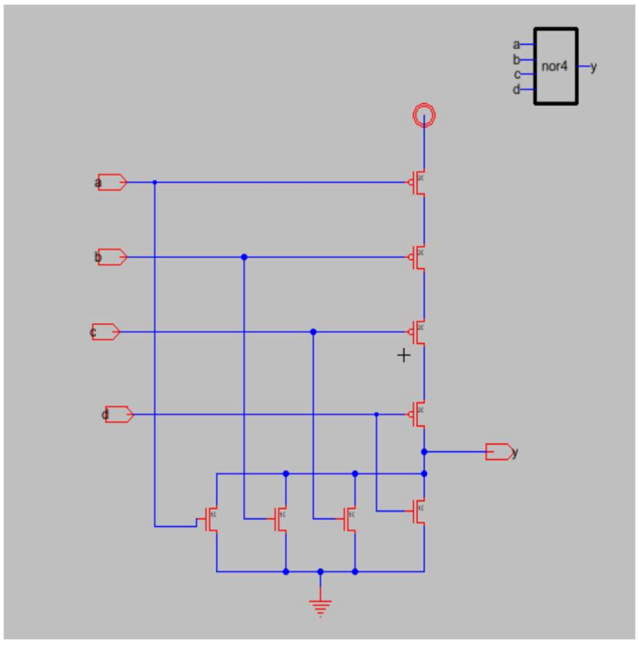

Figure 47 4_input nor gate-Schematic

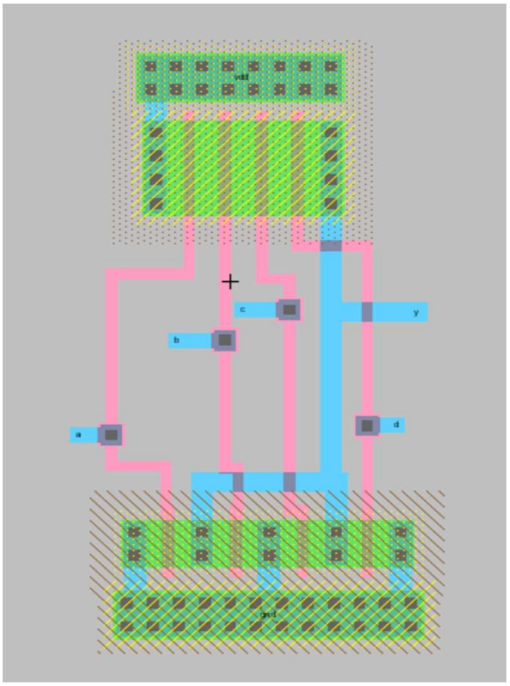

Figure 48 4_input nor gate-Layout

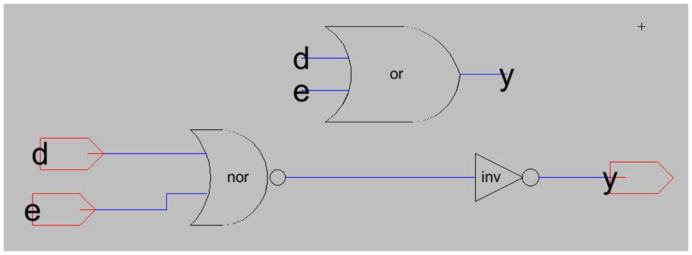

Figure 49 2_ input or gate-Schematic

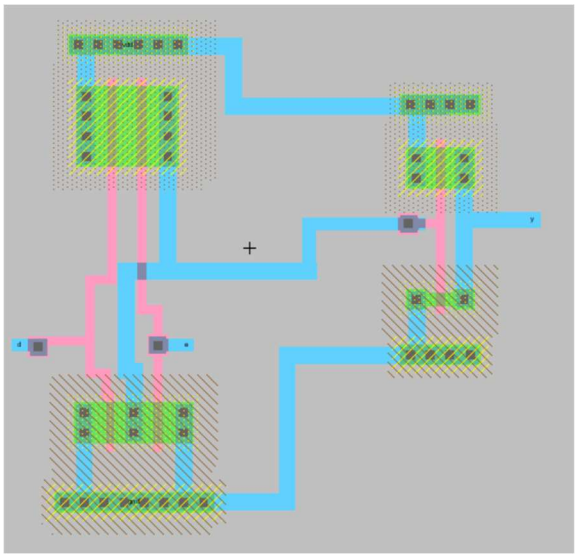

Figure 50 2_input or gate-Layout

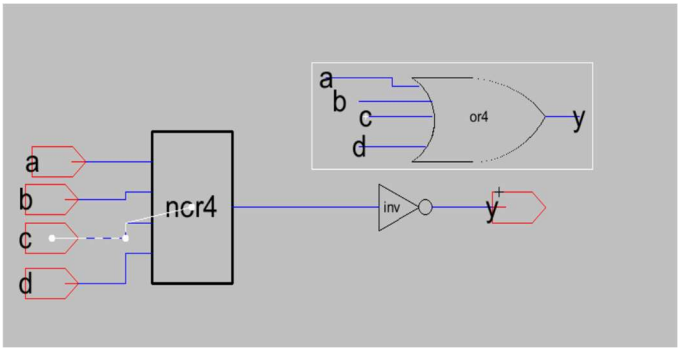

Figure 51 4_input or gate_Sch

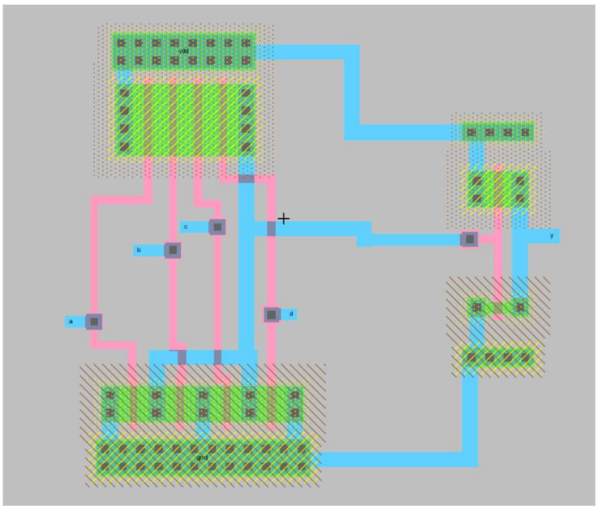

Figure 52 4 input or gate_Layout

Figure 73 shows the first designed Xor gate with nand gates.

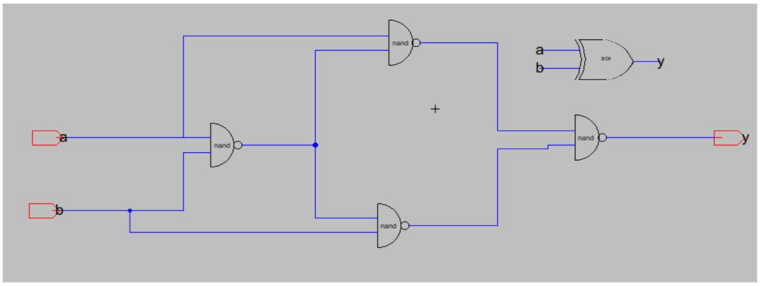

Figure 53 First designed Xor gate_Sch

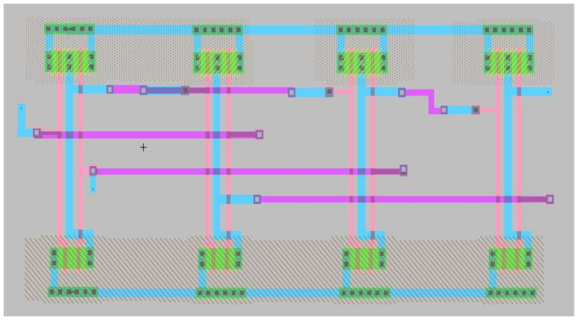

Figure 54 First designed Xor gate_Lay

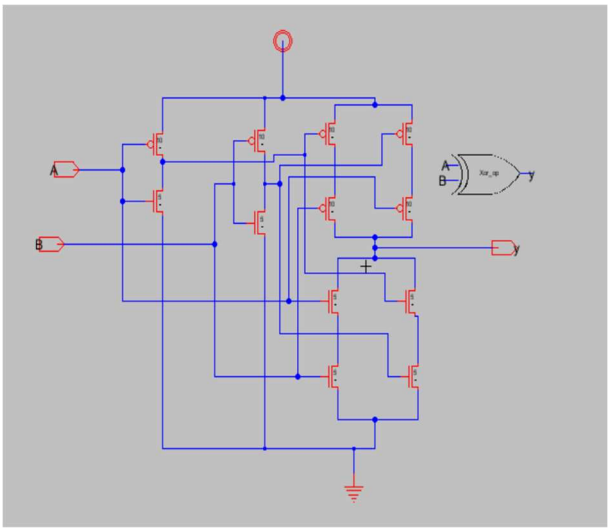

Figure 55 Second designed Xor gate (optimised)_Sch

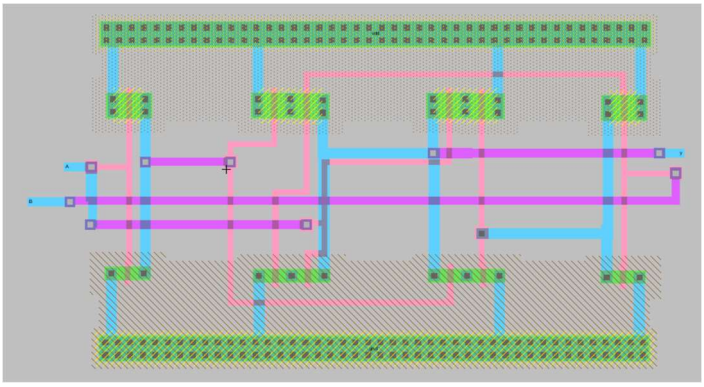

Figure 56 Second designed Xor gate (optimised)_Layout

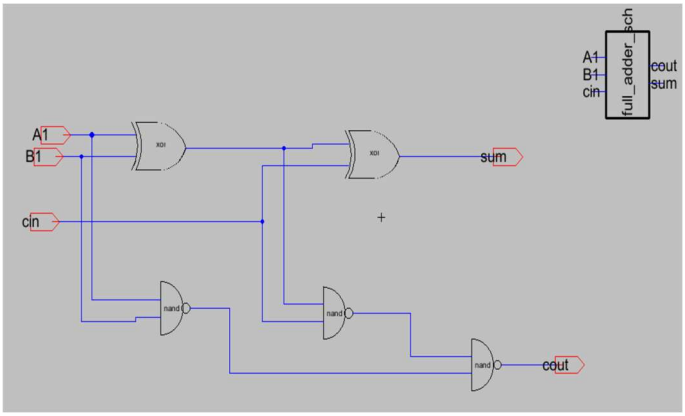

Figure 57Full adder with first xor gate_ Sch

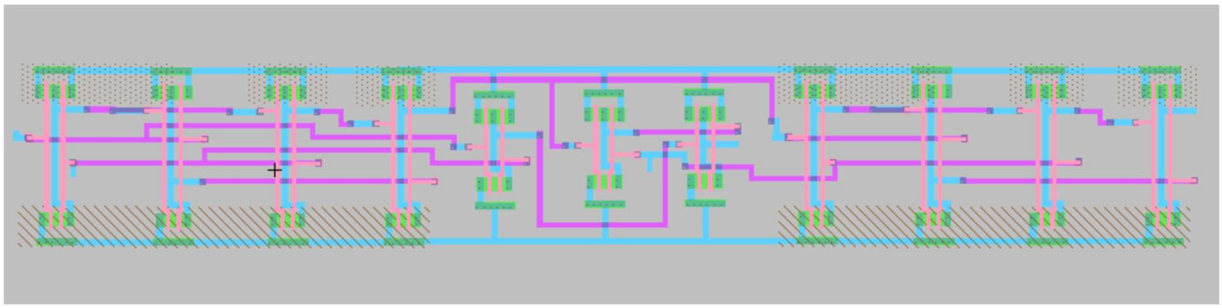

Figure 58 Full adder with first xor gate_ Layout

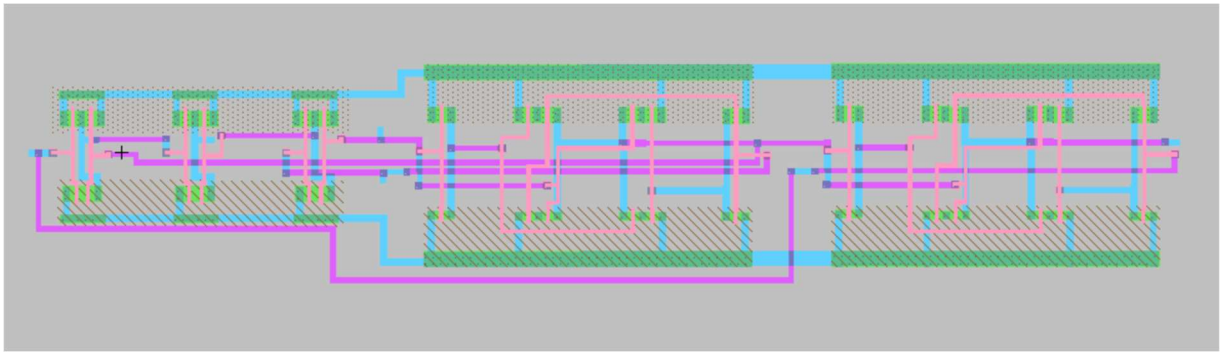

Figure 59 Full adder with optimised Xor gate

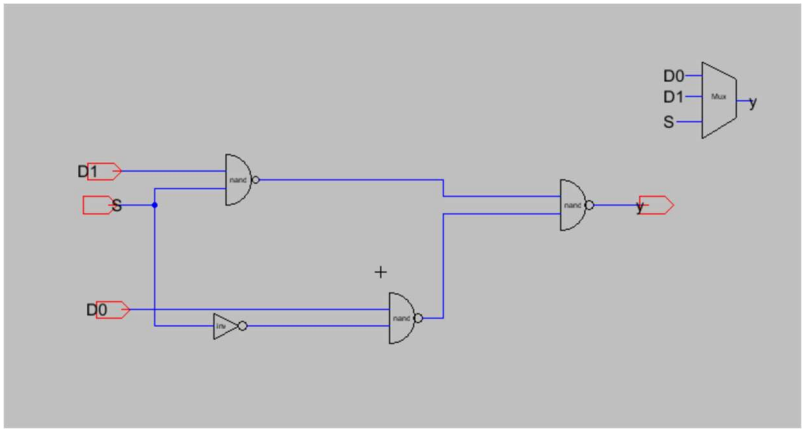

Figure 60 Mux 2 by 1- Sch

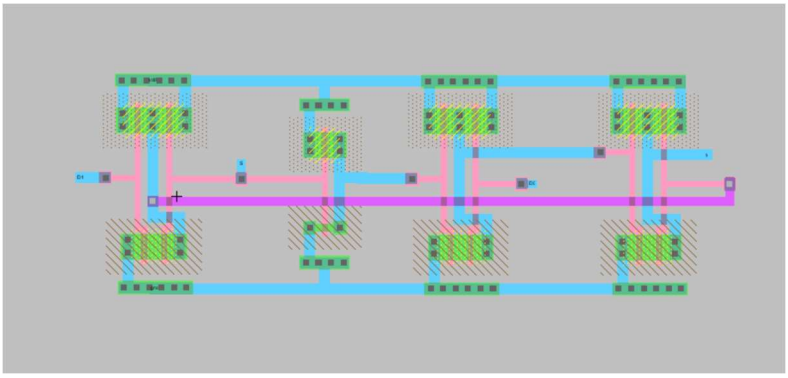

Figure 61 Mux 2 by 1- Lay

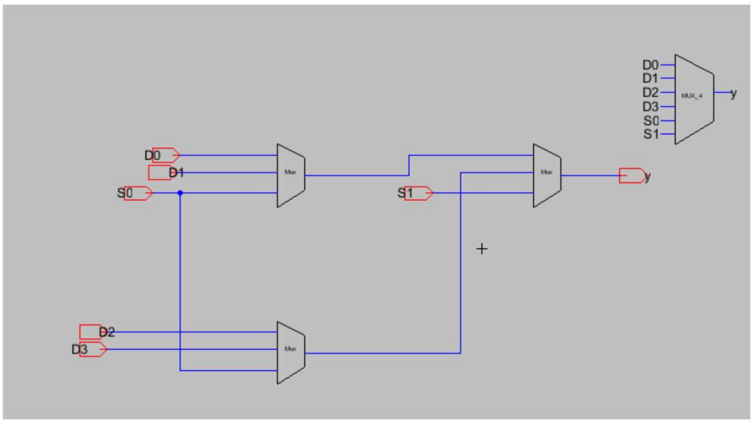

Figure 62 Mux 4 by 1_ Sch

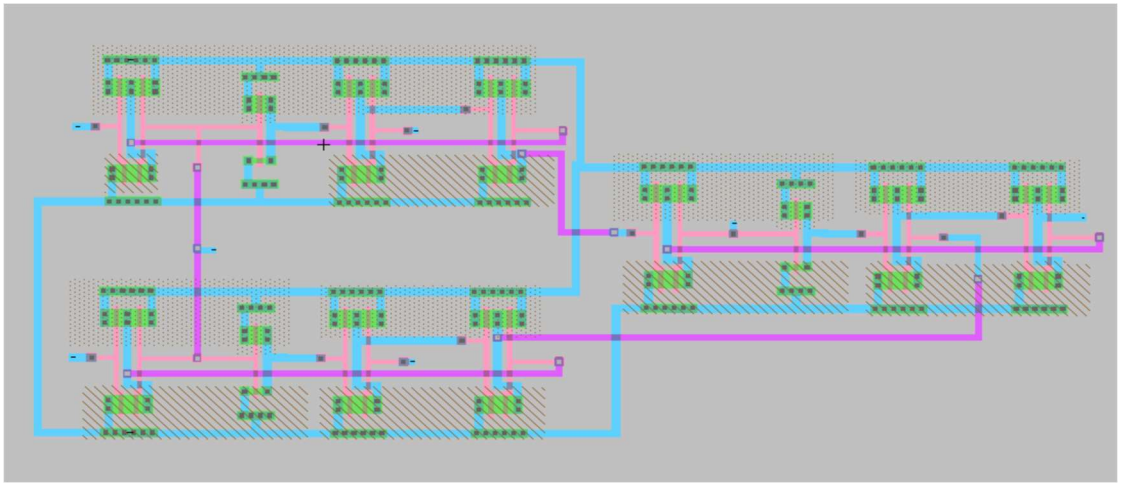

Figure 63 Mux 4 by 1-_Lay

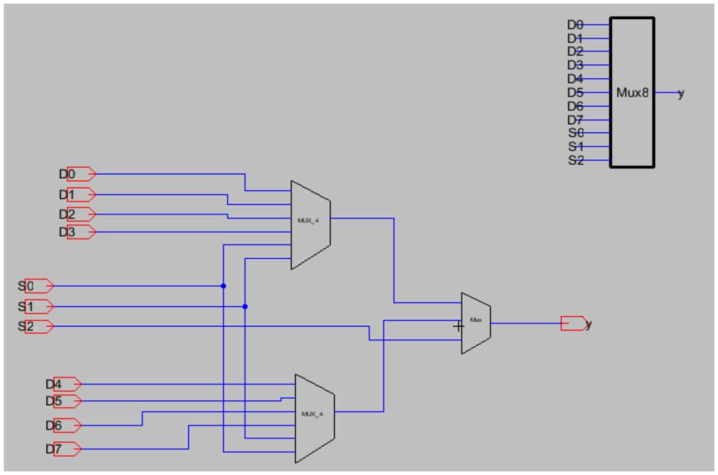

Figure 64 Mux 8 by 1- Sch

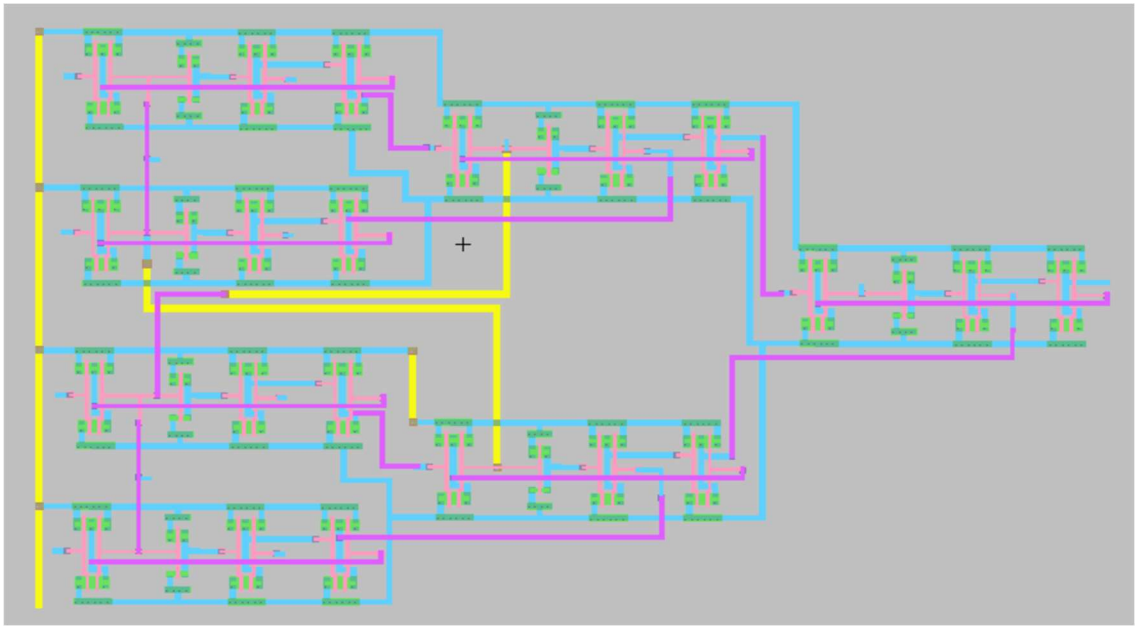

Figure 65 Mux 8 by 1_ Lay

Appendix :

Chapter 3 Additional Layouts

2.1 Adder and Subtractor

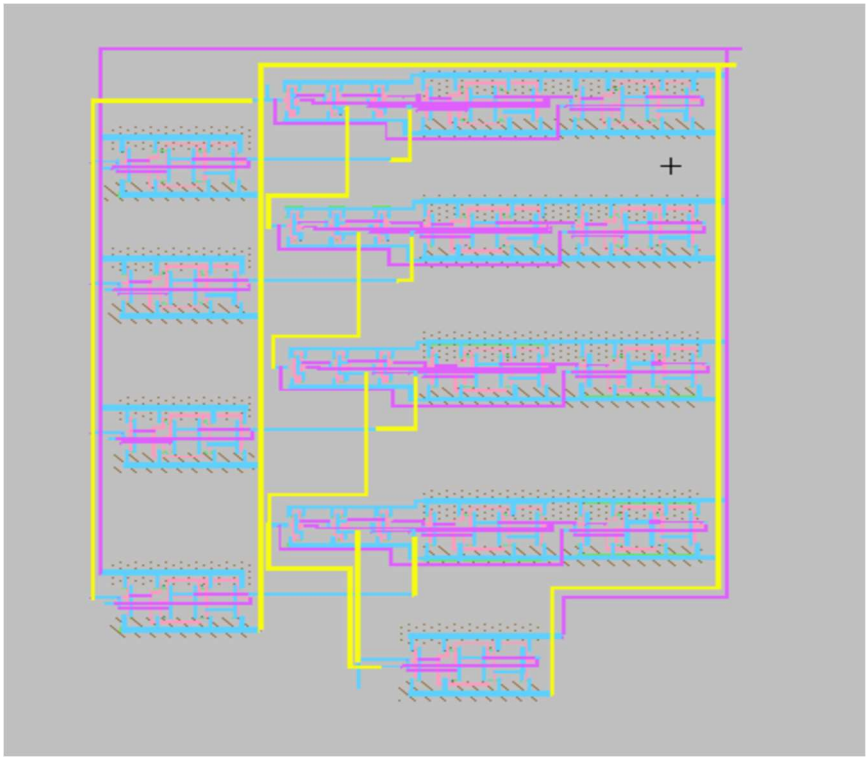

Figure 66 Adder and Subtractor- Extra layout figure

2.2 Left shift B small block:

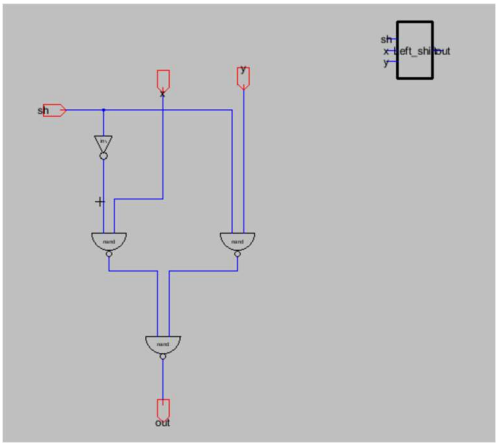

Figure 67 Schematic of 1-bit left shift _ nand gates and inverter

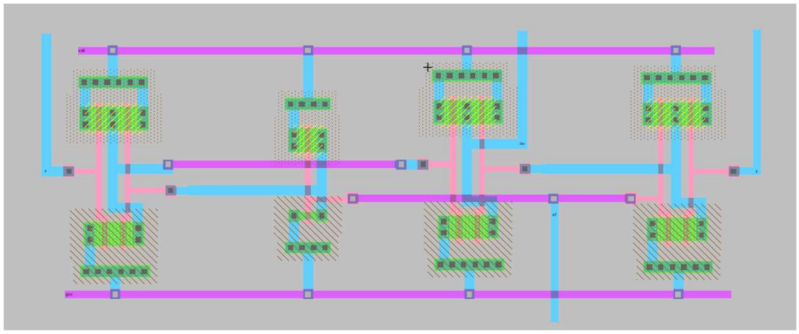

Figure 68 Lay of 1-bit left shift with 3 nand gates and one inverter

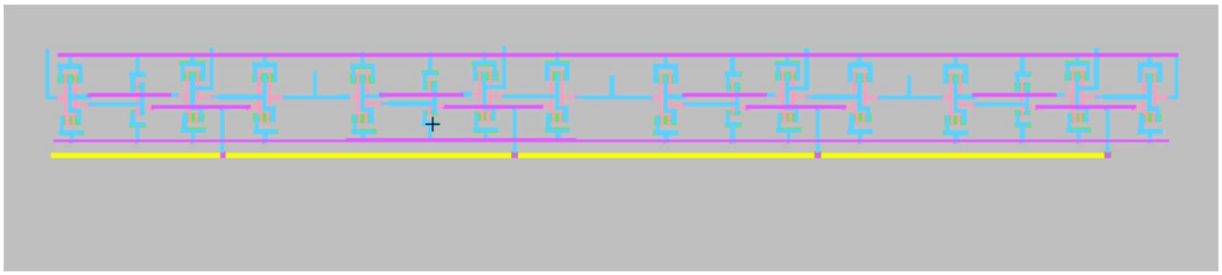

Figure 69 Left shift-4B Figure for layout

2.3 Negative A or Negative B:

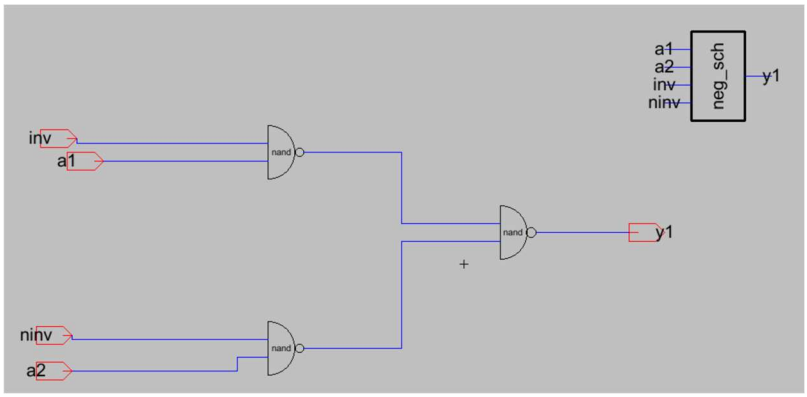

Figure 70 Sch of 'neg' circuit for selecting A or B as an input of negative circuit

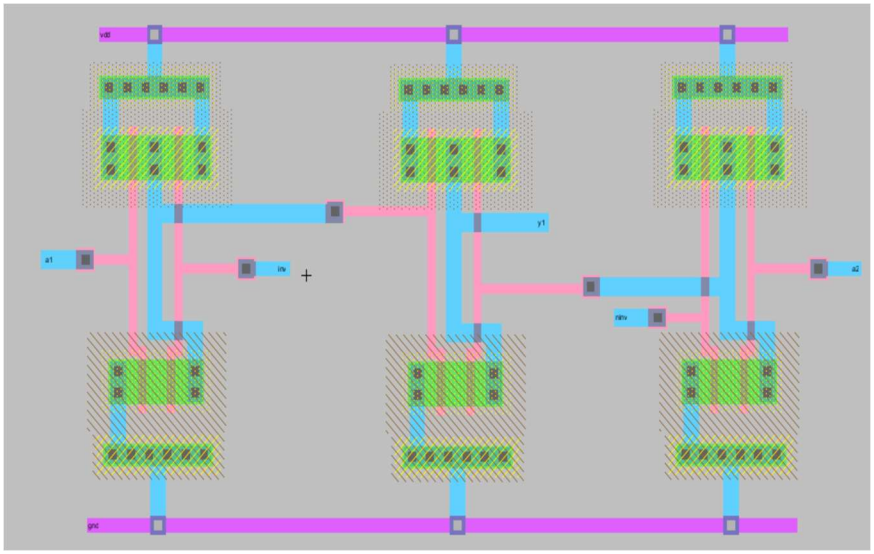

Figure 71 Lay of a circuit for selecting A or B as an input of neg circuit

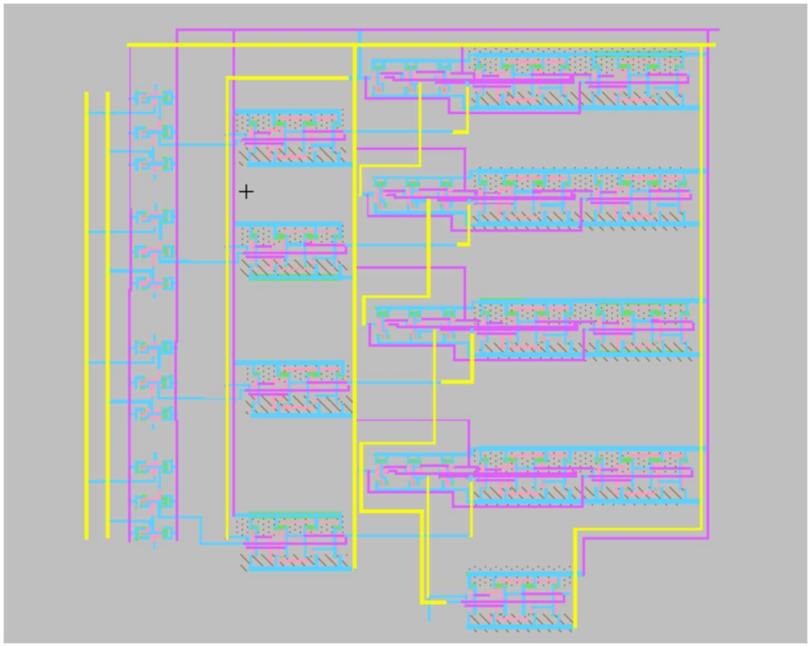

Figure 72 Negative A and Negative B layout

2.4 Absolute:

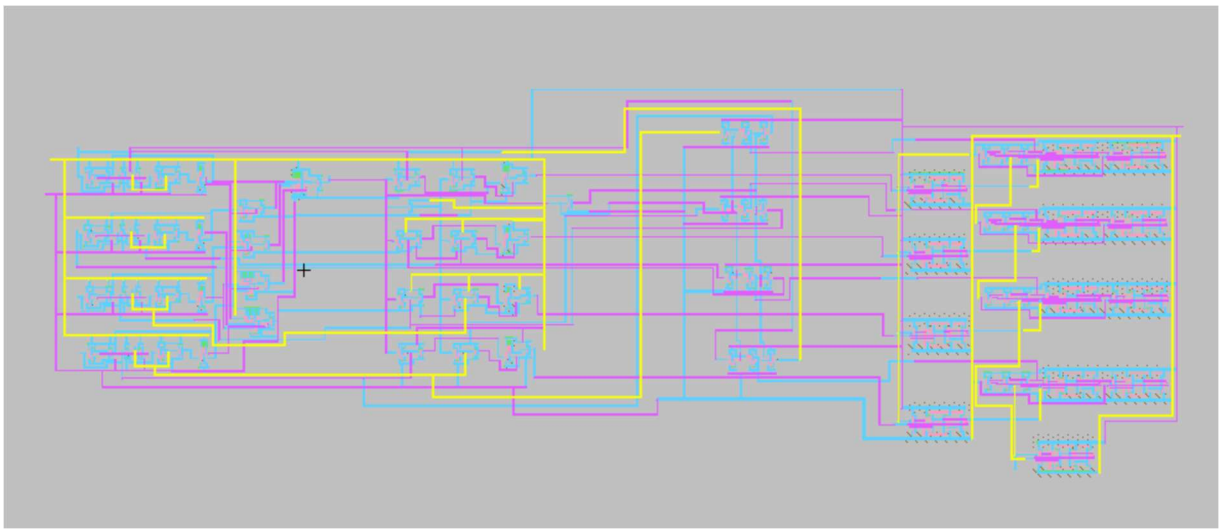

Figure 73 Absolute |A-B|-layout

2.5 Multiplication:

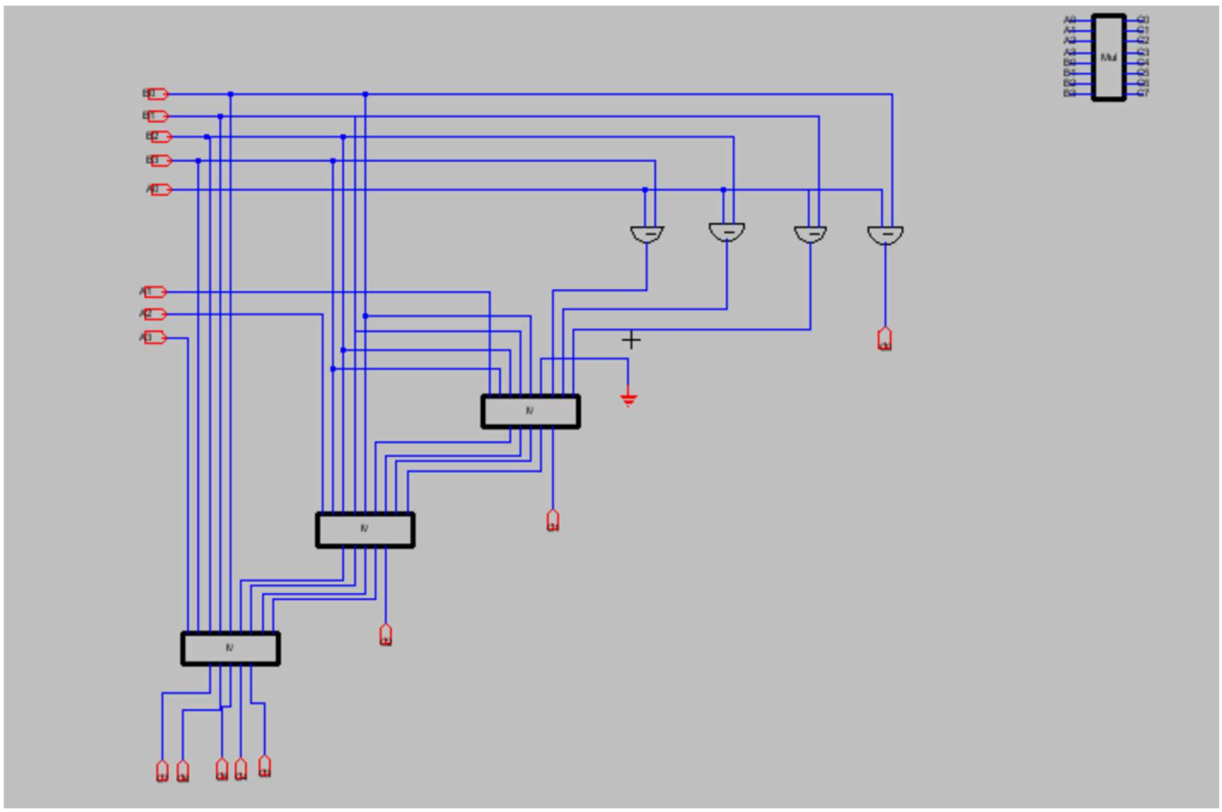

Figure 74 Multiplier Schematic

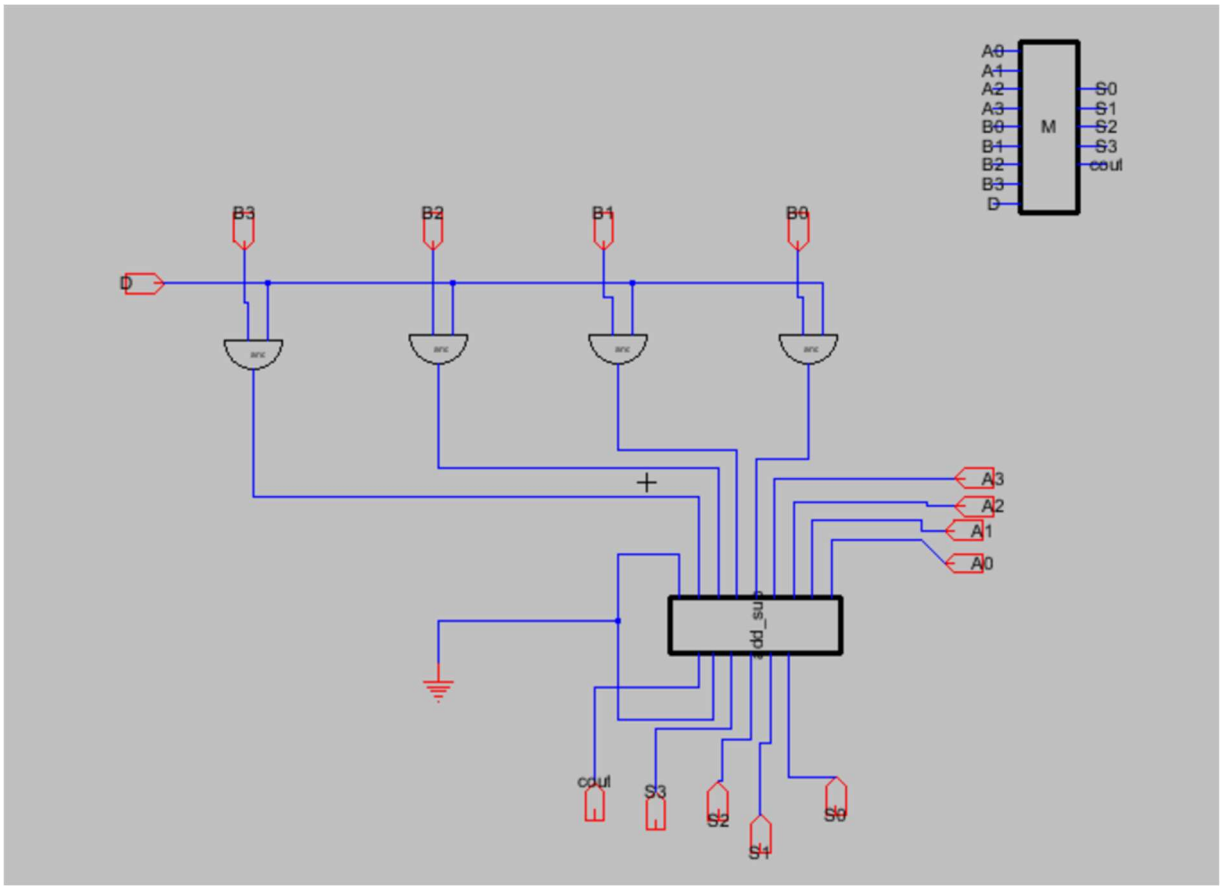

Figure 75 Design of Single Multiplier Block_ Schematic

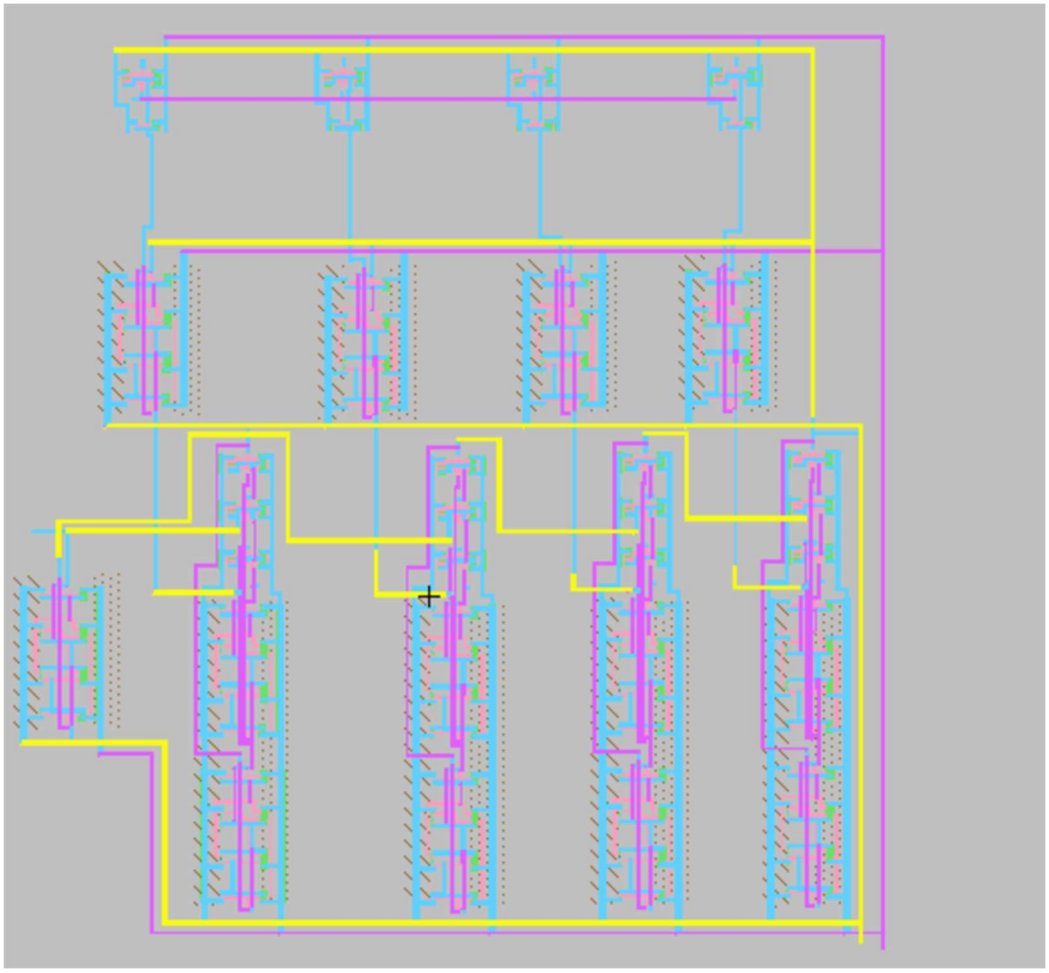

Figure 76 Layout of Single Mul Block

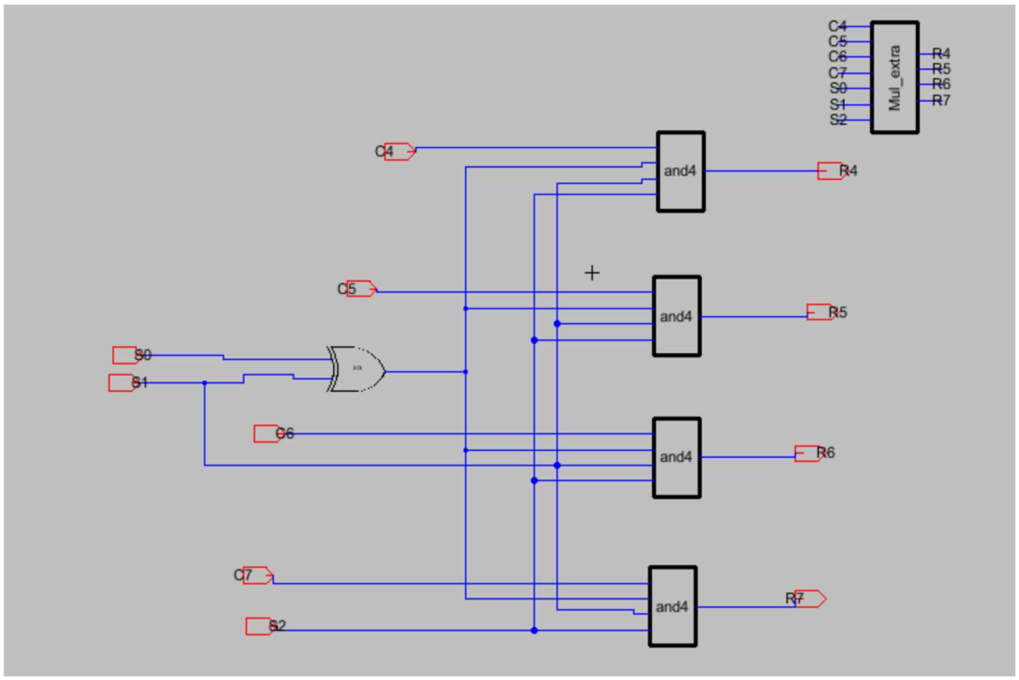

Figure 77 Multiply Extra_schematic

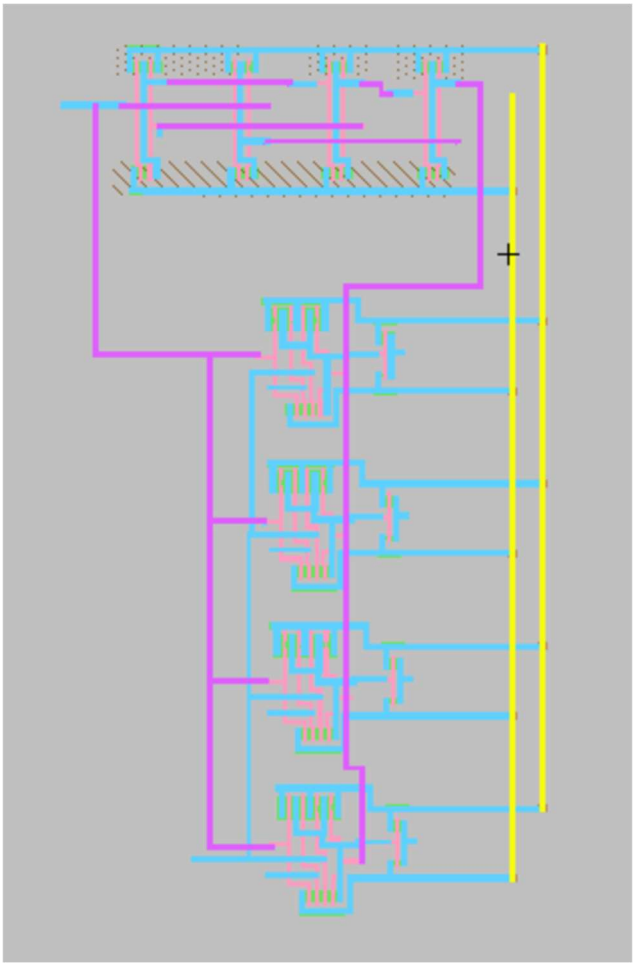

Figure 78 Multiply Extra_Layout

2.6 Minimum selector:

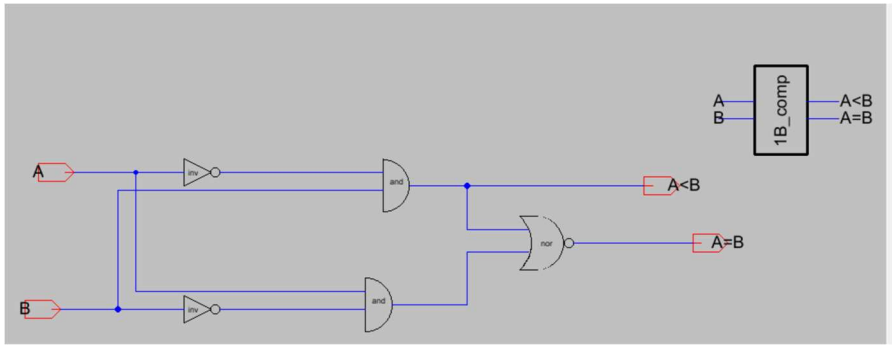

Figure 79 Schematic_1-bit compare

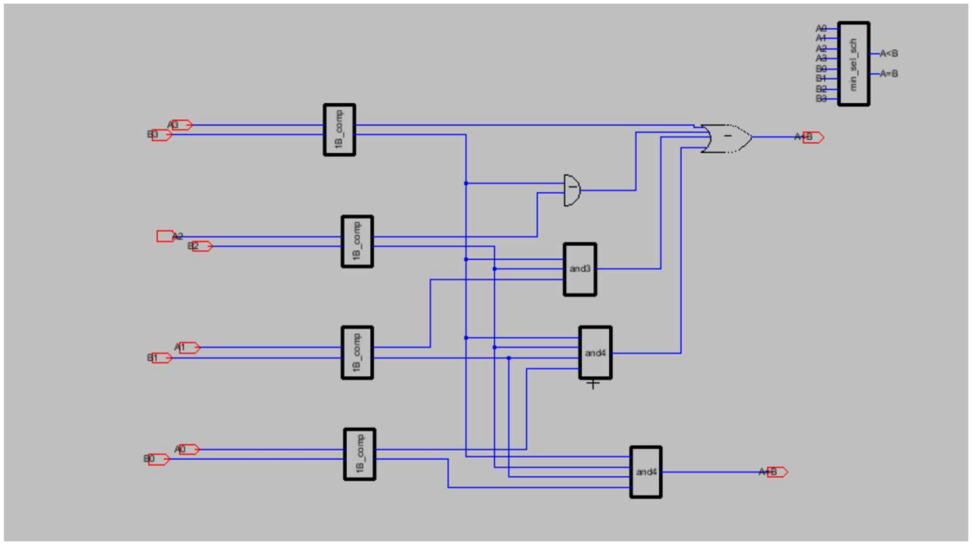

Figure 80 sch_4-bit compare

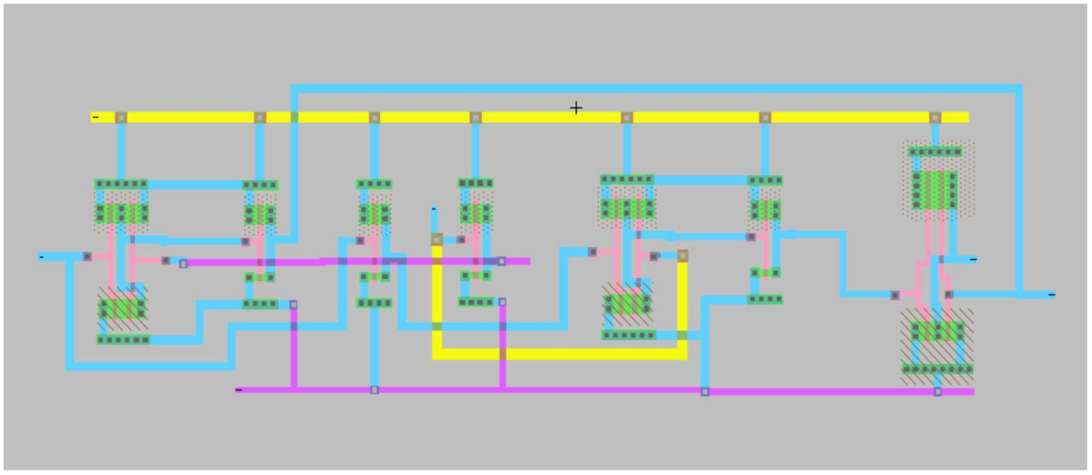

Figure 81 Layout- 1-bit compare

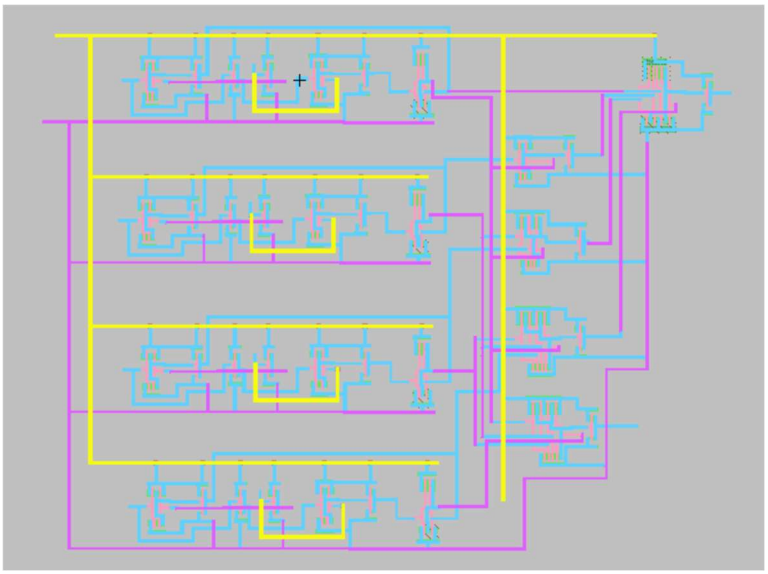

Figure 82 Layout- 4-bit compare (named min)

Appendix 3:

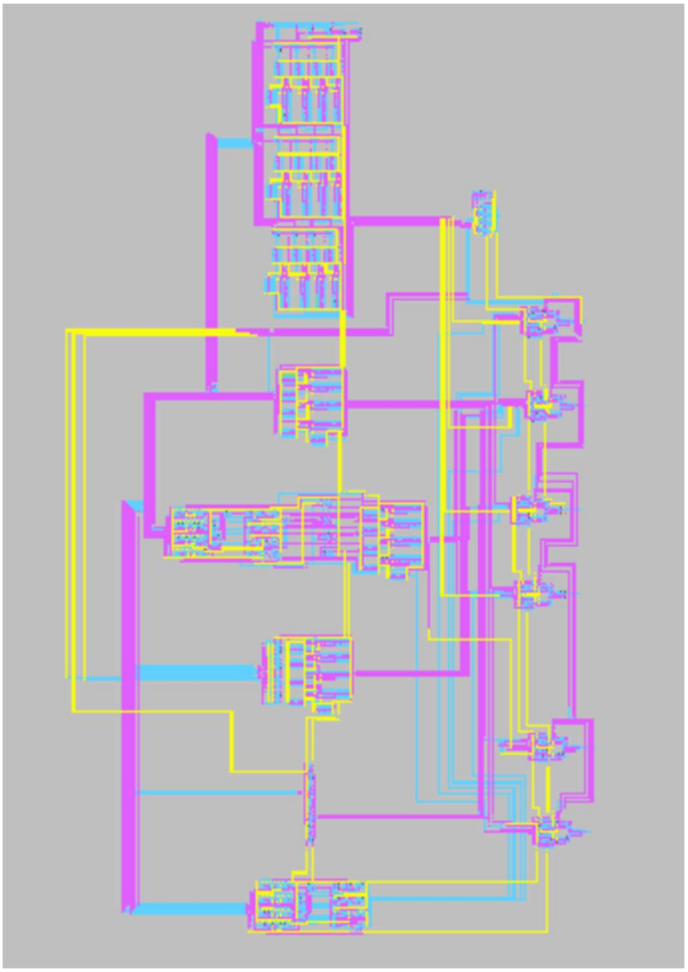

Figure 84 ALU _Layout Block