VLSI Design Project Report

Course Code:	EC2.201
Term:	Monsoon Semester 2023
Project:	Four bit CMOS ALU

By: Koustubh Jain,
ID. 2023122003
`koustubh.jain@research.iiit.ac.in`

Date: 2 December 2023

Contents

1 Introduction

The objective of the project is to design a 4-bit ALU using CMOS technology. The ALU takes in two, 4-bit inputs `A[0:3]` and `B[0:3]` along with a 1-bit carry input `Cin`. It gives a 4-bit output `F[0:3]` and a 1-bit carry output `Cout`. The computations/operations performed on these 4-bits is controlled using 2 control inputs `M1,M0`.

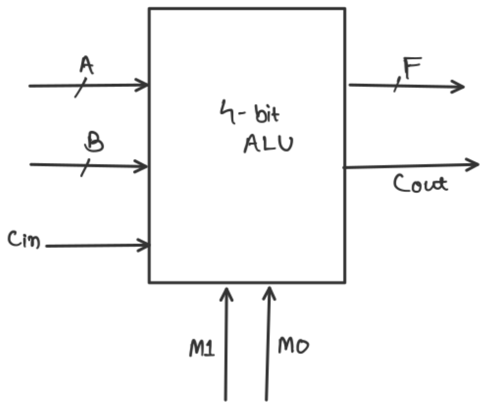


Figure 1: ALU to be implemented

Modes of Operation of ALU		
Mode of Operation	M1	M0
Addition	0	0
Subtraction	0	1
Comparator	1	0
AND	1	1

Table 1: Various modes of operation of ALU

When the ALU is in Comparator mode, only **MSB of `F` and `Cout`** indicate result [See 2.2]. The 4-bit ALU itself is made up of four 1-bit ALUs, please see 16 for more details.

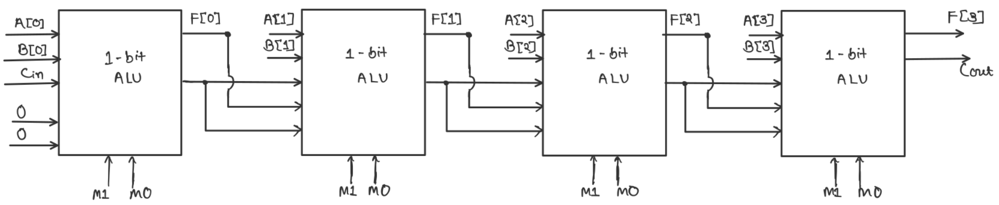


Figure 2: Internal structure of 4-bit ALU

2 1-bit ALU

The 1-bit ALU takes three 1-bit inputs `A`,`B` and `C`. Besides these it also takes in two bits `C1` and `C0` which are used only in comparator mode of the ALU. Each 1-bit ALU is capable of performing the same set of operations as described in 1. The 1-bit ALU gives two, 1-bit outputs, `F` and `Cout`. `F` is the output of the operation performed by the ALU and `Cout` is the carry output in case of addition and subtraction.

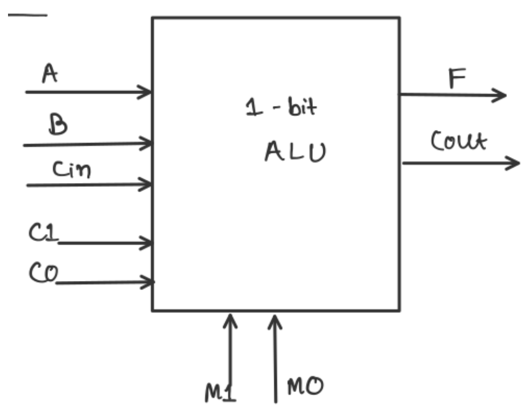
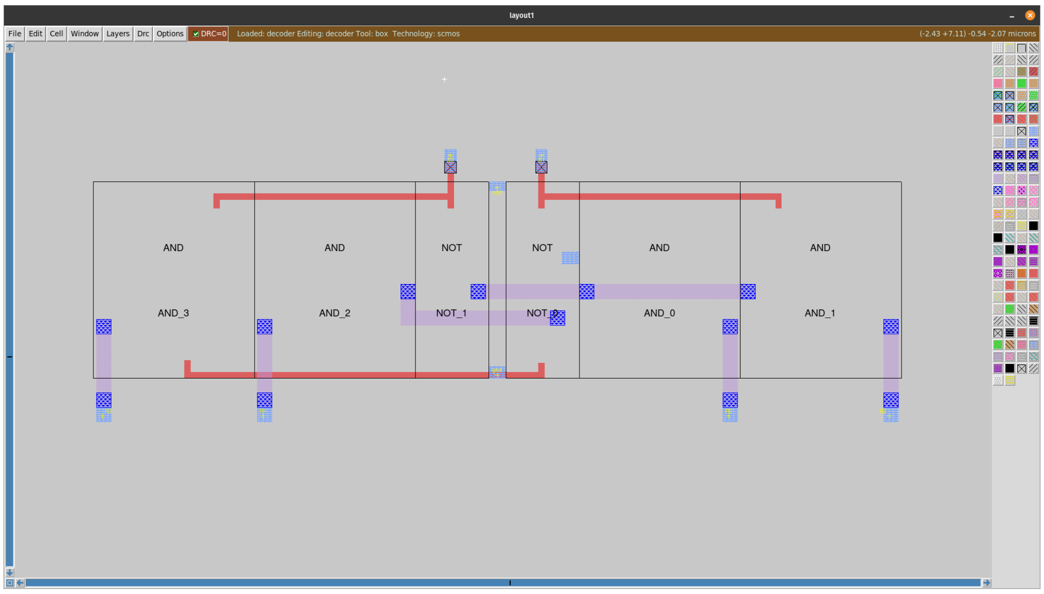


Figure 3: 1-bit ALU

We use a `2:4` decoder to choose mode of operation of ALU.Inputs `M1.M0` are given as inputs to this decoder and its outputs mapped according to 1 act as enable lines for the various operations in the ALU.

2.0.1 Decoder

The `2:4` decoder has the following layout: It has 2 inputs, `S1,S0` which are

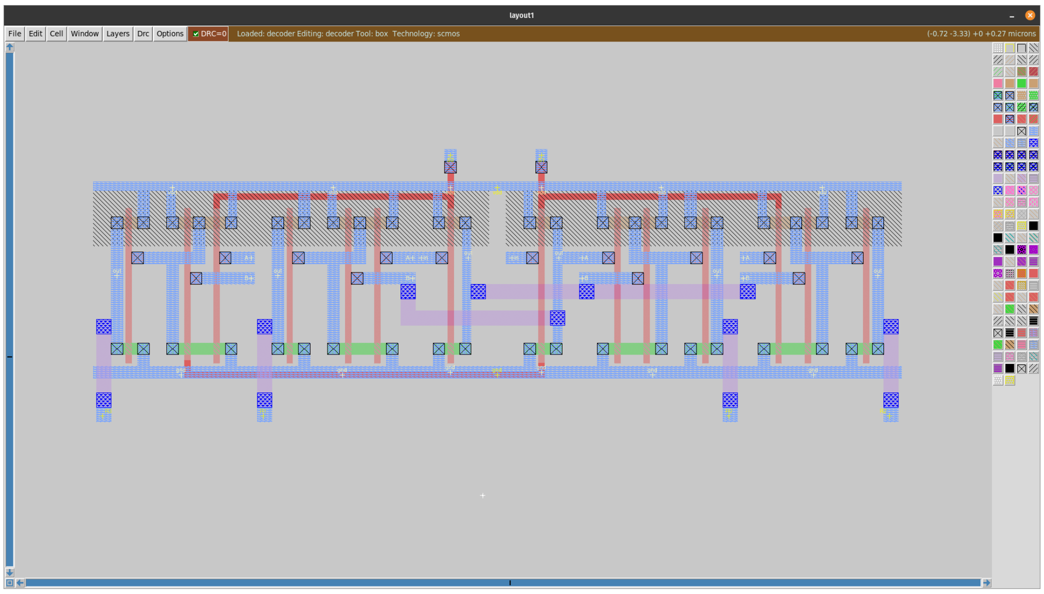


Loaded: decoder Editing: decoder Tool: box Technology: scmos

(-2.43 +7.11) -0.54 -2.07 microns

Figure 4: Block level view of decoder

the control inputs `M1,M0` and it has 4 outputs, `R0,R1,R2,R3`



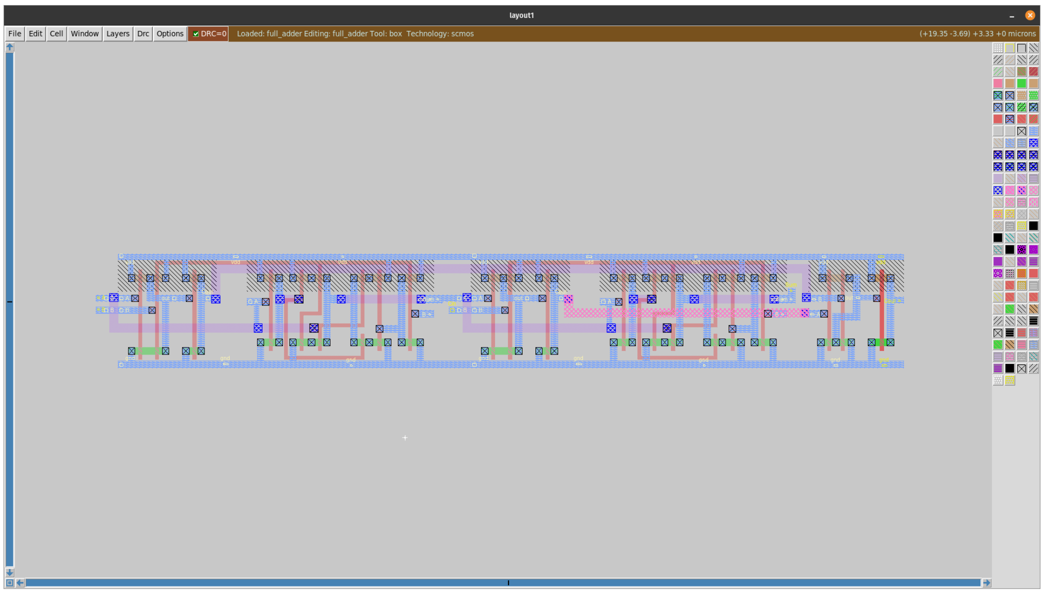
Loaded: decoder Editing: decoder Tool: box Technology: scmos

(-0.72 -3.33) +0 +0.27 microns

Figure 5: Expanded Layout of decoder

2.1 Adder/Subtractor

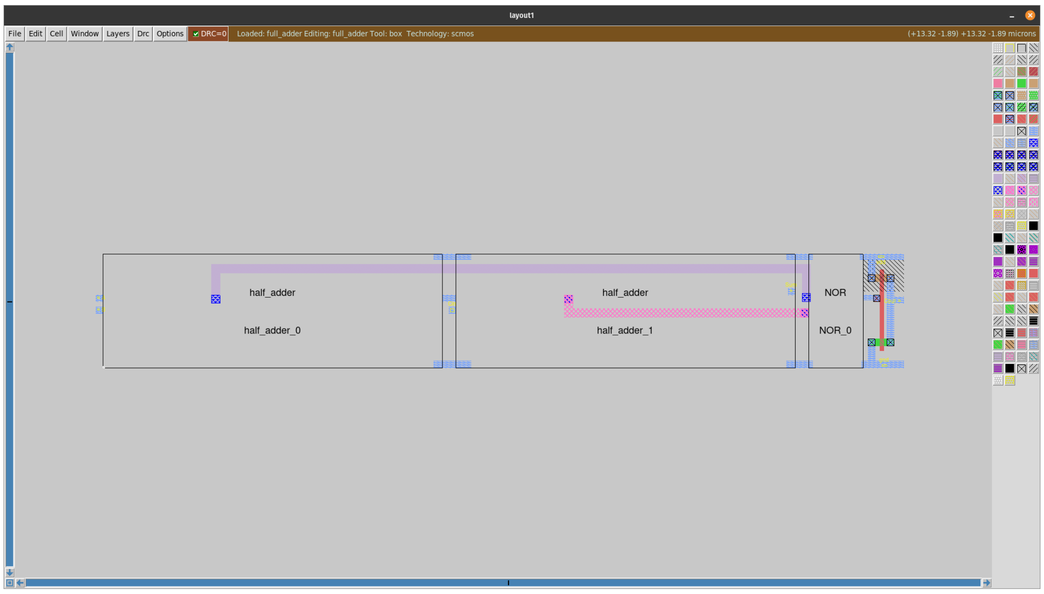
Addition is carried out using a carry look ahead full adder comprised of 2 half-adders. The same adder is used to make a full subtractor by complementing the first input and taking a complement of the output of the adder.



Loaded: full_adder Editing: full_adder Tool: box Technology: scmos

(+19.35 -3.69) +3.33 +0 microns

Figure 7: Expanded view of the full adder



Loaded: full_adder Editing: full_adder Tool: box Technology: scmos

(+13.32 -1.89) +13.32 -1.89 microns

Figure 6: Block level view of the full adder

2.2 Comparator

When ALU is in comparator mode, the output is indicated by `F` and `Cout`.The comparator performs comparison of the current bits, `A` and `B` while knowing the result of the comparison of the previous bits from bits `C1,C0`. It then gives final output of comparison of all the bits upto the present bits on the basis of these four bits which is encoded in `Z1,Z0` which are mapped to `F,Cout` of the respective ALU. The information is encoded in the bits `C1,C0` and `Z1,Z0`

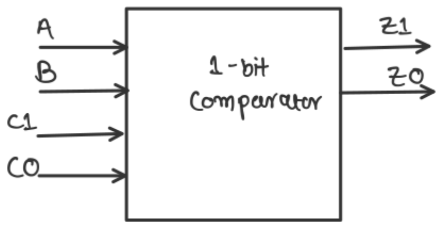
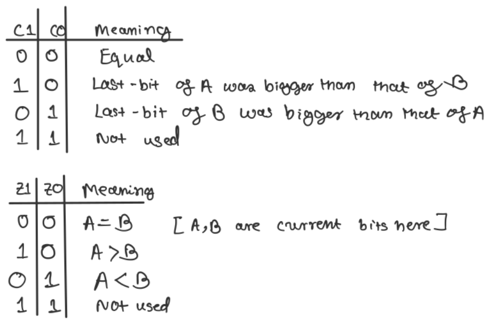


Figure 8: Comparator

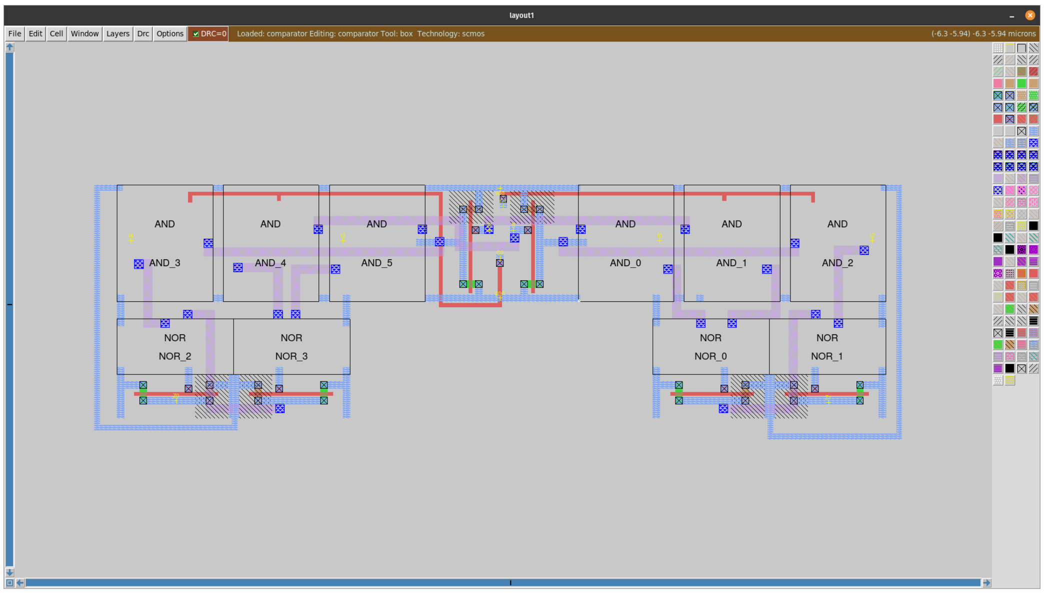
as follows: When `A=B` i.e. both the present bits are same then, `Z1,Z0` are set to be same as `C1,C0`. It has the following CMOS layout:



C1	C0	Meaning
0	0	Equal
1	0	Last-bit of A was bigger than that of B
0	1	Last-bit of B was bigger than that of A
1	1	Not used

Z1	Z0	Meaning
0	0	A = B [A, B are current bits here]
1	0	A > B
0	1	A < B
1	1	Not used

Figure 9: Comparator Logic



layout1

File Edit Cell Window Layers Drc Options DRC=0 Loaded: comparator Editing: comparator Tool: box Technology: scmos (-6.3 -5.94) -6.3 -5.94 microns

Figure 10: Block Level view of the comparator

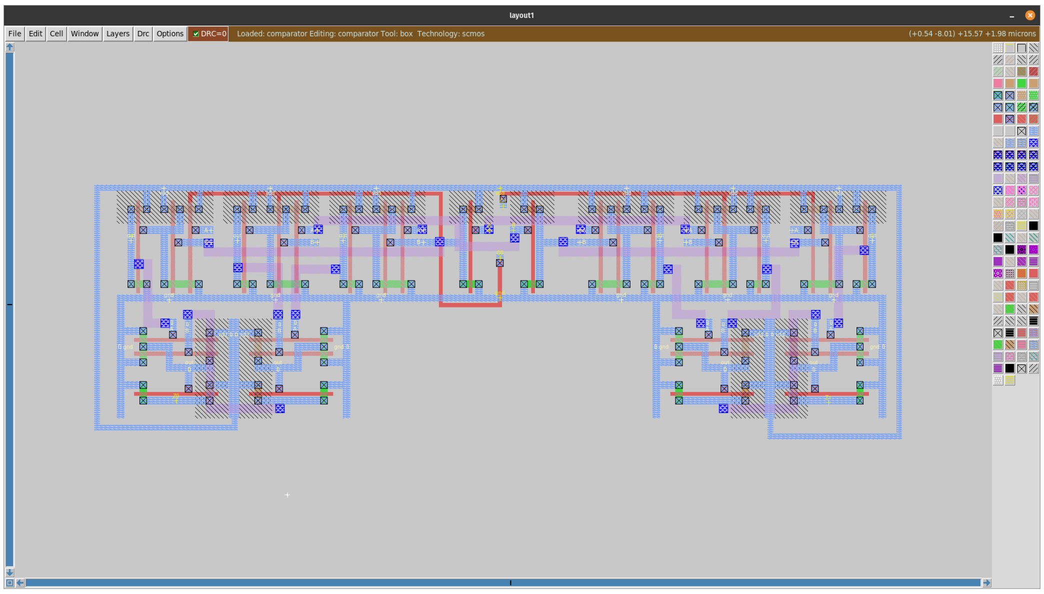

layout1
File Edit Cell Window Layers Drc Options DRC=0
Loaded: comparator Editing: comparator Tool: box Technology: scmos
(+0.54 -8.01) +15.57 +1.98 microns

Figure 11: Expanded view of comparator layout

2.3 Conclusion

Finally upon connecting all the components appropriately, we get the following layout for the 1-bit ALU:

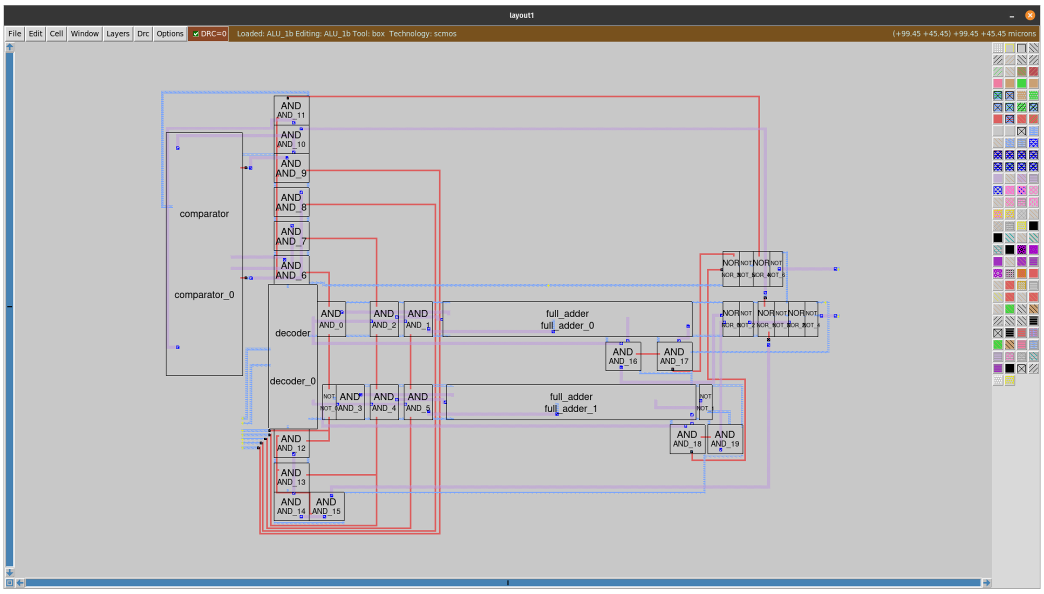


Figure 12: Block level view of 1-bit ALU layout

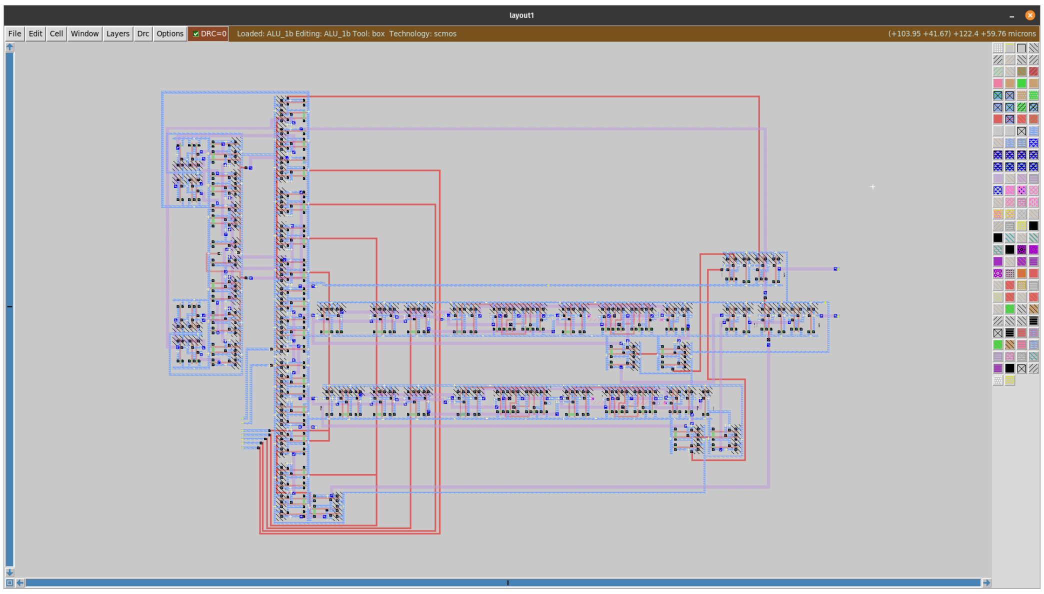

layout1
File
Edit
Cell
Window
Layers
Drc
Options
DRC=0
Loaded: ALU_1b Editing: ALU_1b Tool: box Technology: scmos
(+103.95 +41.67) +122.4 +59.76 microns

Figure 13: Expanded view of 1-bit ALU layout

As evident from the above sections we have made use of a hierarchical implementation which is easy to replicate, reproduce and extend further than 4-bits just as easily. The 4-bit ALU is just four 1-bit ALUs which have been connected with appropriate inputs. Each 1-bit ALU in itself is then a decoder connected to various functional blocks such as adder, subtractor, comparator, AND block. Some of these functional blocks can themselves be broken down into further sub-blocks such as the full adder which comprises of two half adders.

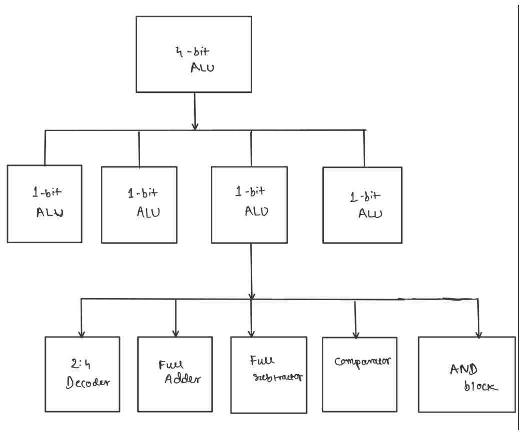


Figure 14: Summary of Hierarchical design implementation

Finally the 4-bit ALU has the following layout:

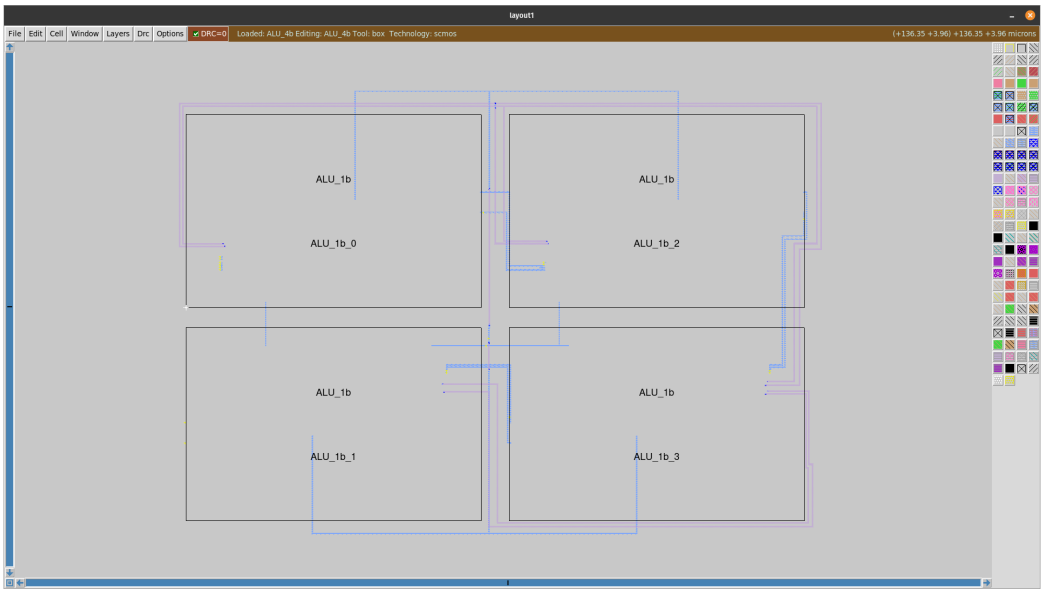


Figure 15: Block level view of 4-bit ALU layout

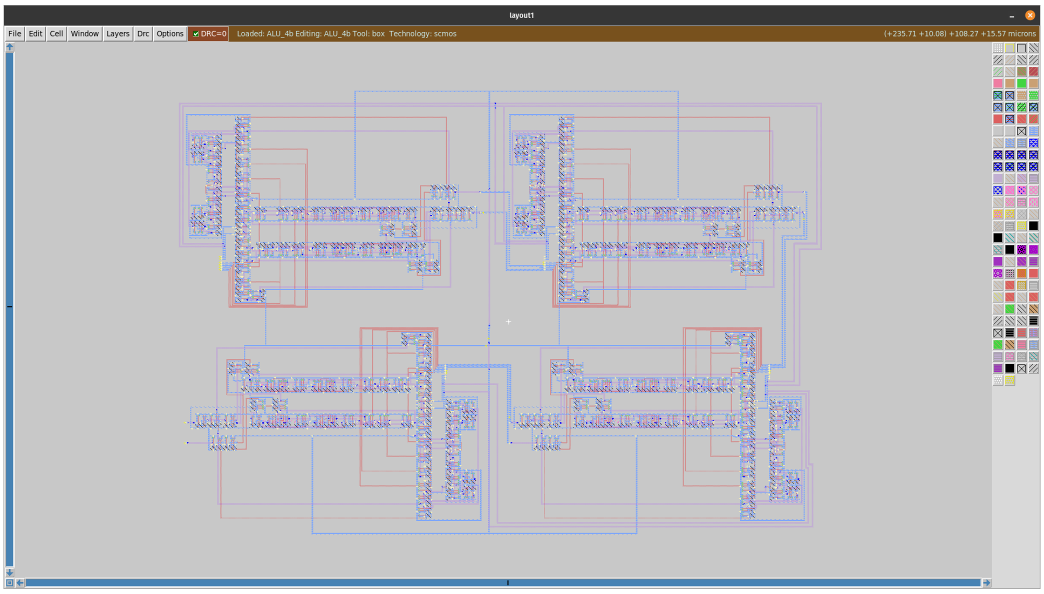

layout1
File
Edit
Cell
Window
Layers
Drc
Options
DRC=0
Loaded: ALU_4b Editing: ALU_4b Tool: box Technology: scmos
(+235.71 +10.08) +108.27 +15.57 microns

Figure 16: Expanded view of 4-bit ALU layout

3 Delay Analysis of 4-bit ALU

NOTE: All the scripts for delay analysis can be found inside the folder titled `Delay_Analysis`. All the delay analysis scripts are also written in `SPICE` and hence are in `.cir` format.

3.1 Addition Mode

While measuring delay with respect to input `A[0:3]` the parameters and inputs to the 4-bit ALU have been set as follows:

```
    ***For Delay Analysis***

    ***For Addition wrt A***
    Vin1 S0 GND 0
    Vin2 S1 GND 0
    Vin3 A0 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin4 A1 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin5 A2 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin6 A3 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin7 B0 GND 0
    Vin8 B1 GND 0
    Vin9 B2 GND 0
    Vin10 B3 GND 0
    Vin11 Cin GND supply
    Vin12 C1 GND 0
    Vin13 C0 GND 0

    .tran 1n 0.4u
```

To ensure that the output changes only due to input `A` and taking into account the maximum delay scenario, `A` is being pulsed with a time period of 0.2 µs and `B` has been set to 0000_2. Then the delays due to each bit of `A` with respect to each bit of the output `F` are measured as follows:

```
    echo "Delay due to A0" > delay_analysis_add_wrt_A.txt
    meas tran trise00 trig v(A0) val=0.5*supply rise=1 targ v
   (F0) val=0.5*supply rise=1
    meas tran tfall00 trig v(A0) val=0.5*supply fall=1 targ v
   (F0) val=0.5*supply fall=2
    let tpd00 = (trise00 + tfall00)/2
    echo "tpd00 = $&tpd00" >> delay_analysis_add_wrt_A.txt
```

We measure `rise-rise` and `fall-fall` delay for the input with respect to the various output bits because the output rises as the input rises and it falls simultaneously with the input in Addition mode of operation.

We then calculate the delays due to `A0` with respect to the rest of the output bits i.e. `F1,F2,F3 and Cout`. These value of these delays is printed into an appropriately named text file where `tpdXY` denotes propagation delay due to bit `X` of input with respect to bit `Y` of output.So, `tpd13` would represent delay due to `A1` with respect to `F3`. Please note that `tpdX4` generally denotes delay due to bit `X` of input with respect to the carry out bit, `Cout`.

While measuring delay with respect to input `B[0:3]` the parameters and inputs to the 4-bit ALU have been set as follows:

```
    ***For Delay Analysis***

    ***For Addition wrt B***
    Vin1 S0 GND 0
    Vin2 S1 GND 0
    Vin3 A0 GND 0
    Vin4 A1 GND 0
    Vin5 A2 GND 0
    Vin6 A3 GND 0
    Vin7 B0 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin8 B1 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin9 B2 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin10 B3 GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin11 Cin GND supply
    Vin12 C1 GND 0
    Vin13 C0 GND 0

    .tran 1n 0.4u
```

Delays for input `B` are measured just like how they are measured for input `A` as described above.

While measuring delays with respect to the carry input, `Cin` the parameters and inputs to the 4-bit ALU have been set as follows:

```
    ***For Delay Analysis***

    ***For Addition wrt Cin***
    Vin1 S0 GND 0
    Vin2 S1 GND 0
    Vin3 A0 GND supply
    Vin4 A1 GND supply
    Vin5 A2 GND supply
    Vin6 A3 GND supply
    Vin7 B0 GND 0
    Vin8 B1 GND 0
    Vin9 B2 GND 0
    Vin10 B3 GND 0
    Vin11 Cin GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin12 C1 GND 0
    Vin13 C0 GND 0

    .tran 1n 0.4u
```

To ensure that any time varying changes in output are caused due to `Cin` only, its been set as the only time varying input. And we have set input `A` to 1111_2 and `B` to 0000_2 to take care of the worst case scenario i.e. when the maximum switching can occur. We are measuring `rise-fall,fall-rise` delay with respect to all the outputs except for `Cout` because it will rise when the input will rise and fall with the input, as such for `Cout` we measure `rise-rise,fall-fall` delay.

3.2 Subtraction Mode

To measure delays in subtraction mode with respect to input `A` the same set of inputs, parameters and methods were used. But to measure the delays with input `B` we measured `rise-rise,fall-fall` delay with respect to `Cout` only. Because in a full subtractor when we subtract 1 from 0 with input borrow 1, we get difference 0 with a borrow out 1 i.e. the difference falls as input rises and borrow out rises with the input.

And to measure delays with respect to the borrow in `Cin` we use the following set of inputs and parameters to the 4-bit ALU:

```
    ***For Delay Analysis***

    ***For Subtraction wrt Cin***
    Vin1 S0 GND supply
    Vin2 S1 GND 0
    Vin3 A0 GND 0
    Vin4 A1 GND 0
    Vin5 A2 GND 0
    Vin6 A3 GND 0
    Vin7 B0 GND 0
    Vin8 B1 GND 0
    Vin9 B2 GND 0
    Vin10 B3 GND 0
    Vin11 Cin GND pulse(0 supply 0 0.5p 0.5p 0.1u 0.2u)
    Vin12 C1 GND 0
    Vin13 C0 GND 0

    .tran 1n 0.4u
```

It is evident from the truth table of a full subtractor that the borrow input can affect the output in terms of delay only when both A and B are same. Hence, the choice of setting both inputs to 0. And since we want any delays to be caused just by the borrow input, `Cin` is the only time varying input. We measure `rise-rise and fall-fall` delay for the same.

3.3 Comparator Mode

For the comparator mode, we use same set of inputs for A and B as addition mode but the ALU itself is set to comparator mode and we measure rise-rise and fall-fall delay. No delay is measured with respect to Cin since it has no physical connection to the circuit and hence can not contribute any delay to the outputs of the comparator.

3.4 AND-ing Mode

For AND-ing mode, we use same set of inputs for A and B as addition except since we don't need Cin,C0 and C1 and since they don't have any connection to this part of the circuit they are not taken into account.