

Introduction to VLSI

FINAL PROJECT REPORT

Group-10

Library	Library Owner	ID	username
ronabraham_final	Ron Abraham	927110072	ronabraham
shaikohen_final	Shai Kohen	330065541	shaikohen
rahulsxena_final_proj	Rahul Saxena	925141426	rahulsxena
nyagaka_final	Benjamin Nyagaka	901281675	nyagaka

The project is contained in the **nyagaka_final** library which is owned by Benjamin Nyagaka and the cells he designed are:

1. 2_to_4_decoder_2
2. 4_bit_3_input_OR_2
3. 4_bit_AND_2
4. 4_bit_MUX_2
5. 4_bit_register_2
6. Bit_extended_XOR_2 (for subtraction)
7. 5_bit_register_2
8. Overview_circ_2 (Top level view of the circuit)

The 4-bit XOR is in the **ronabraham_final** library which is owned by Ron Abraham and the cell he designed is:

1. Xor block

The Barrel Right Shift Register is in the **shaikohen_final library** which is owned by Shai Kohen and the cell he designed is:

1. Barrel shifter

The Adder and the ALU are in the **rahulsxena_final_proj** library which is owned by Rahul Saxena and the cells he designed are:

1. Adder_dup
2. ALU_2

Schematic and Layout of each cell

nyagaka Library

2-to-4 decoder

Schematic print

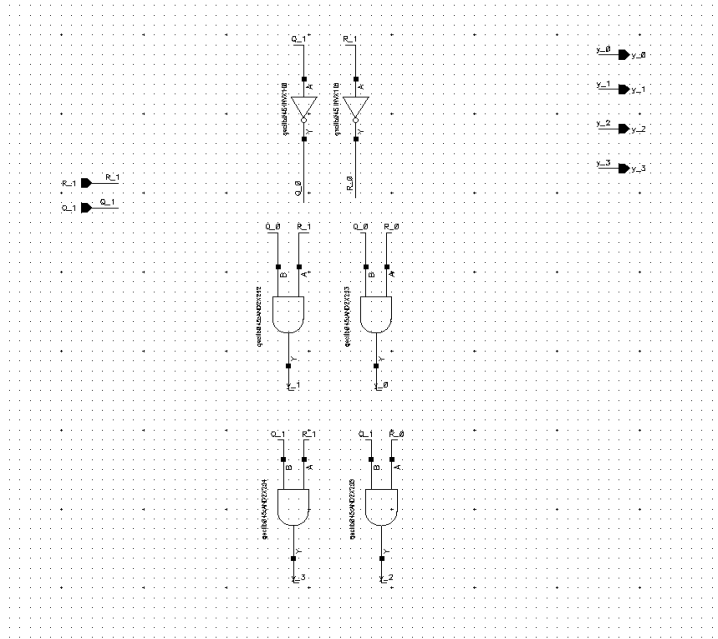


Figure 1: 2-to-4 decoder schematic

Layout print

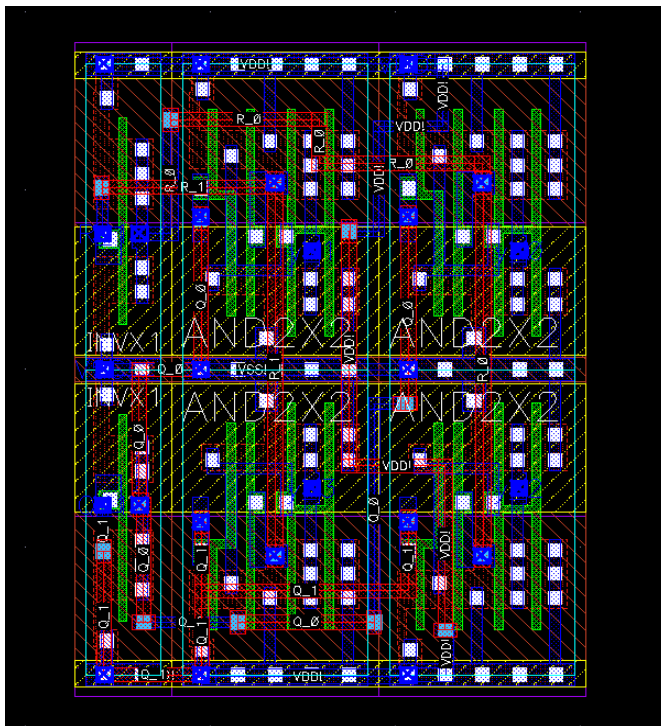


Figure 2: 2-to-4 decoder layout

4-bit 3-input OR

Schematic Print

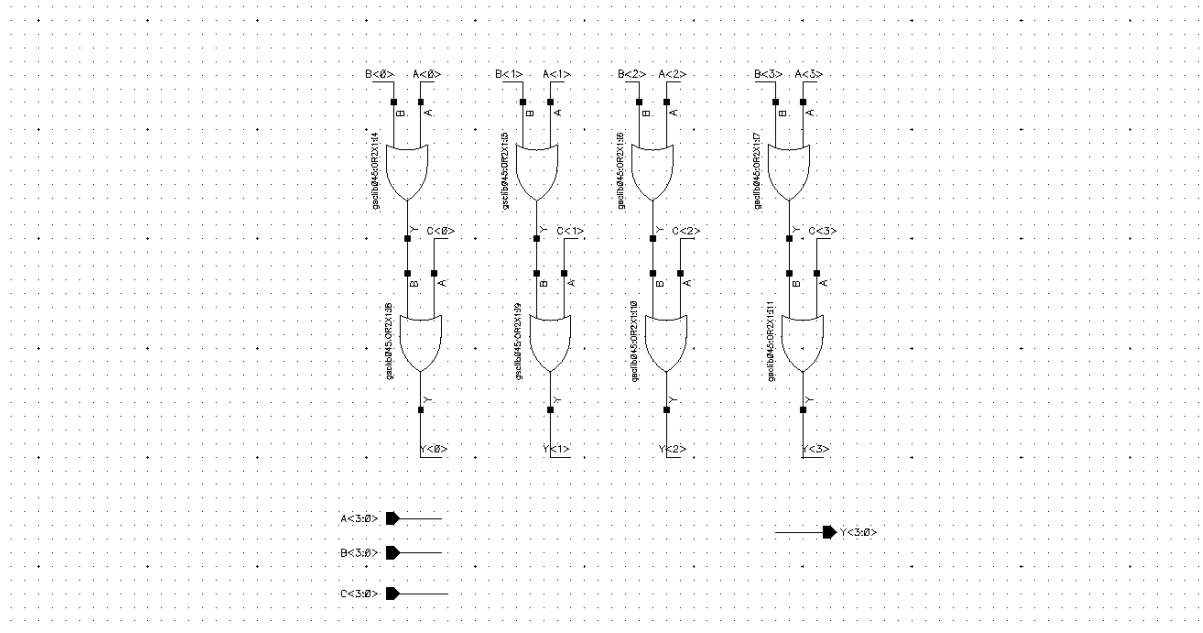


Figure 3: 4-bit 3-input OR schematic

Layout Print

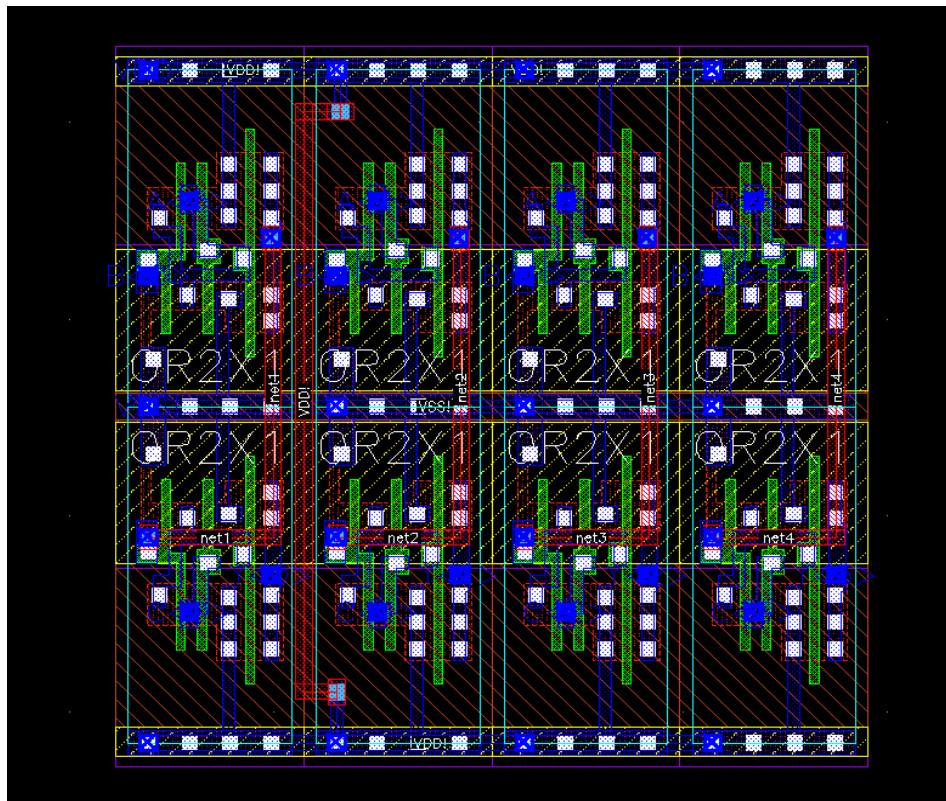


Figure 4: 4-bit 3-input OR layout

4-bit AND

Schematic Print

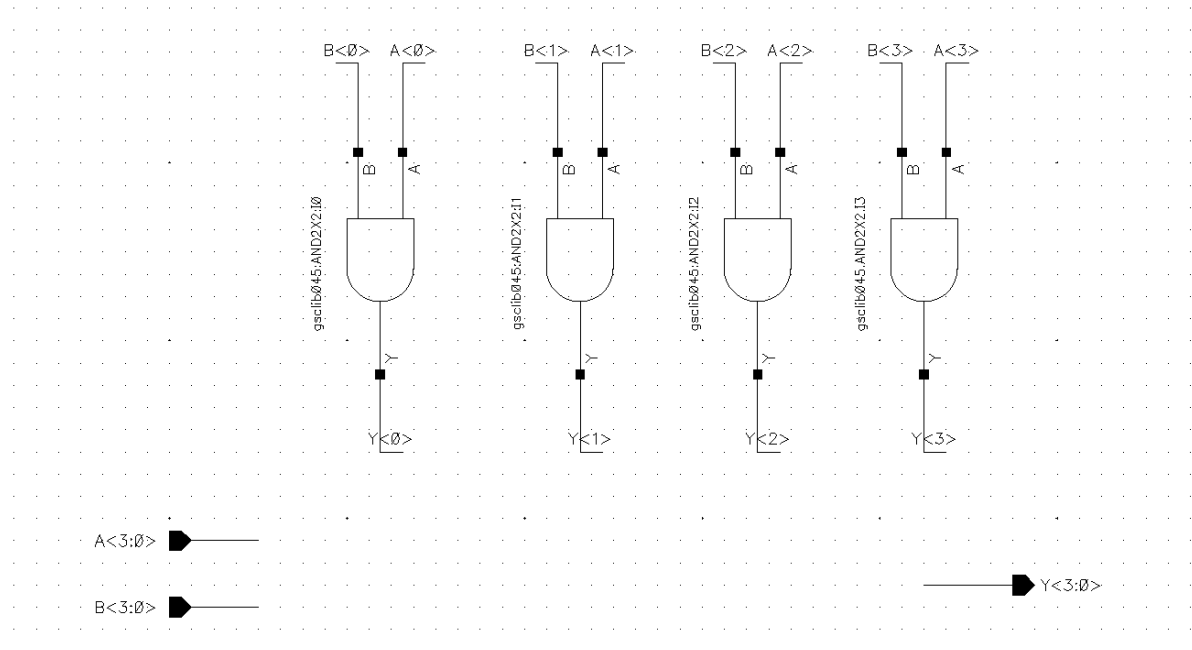


Figure 5: 4-bit AND schematic

Layout Print

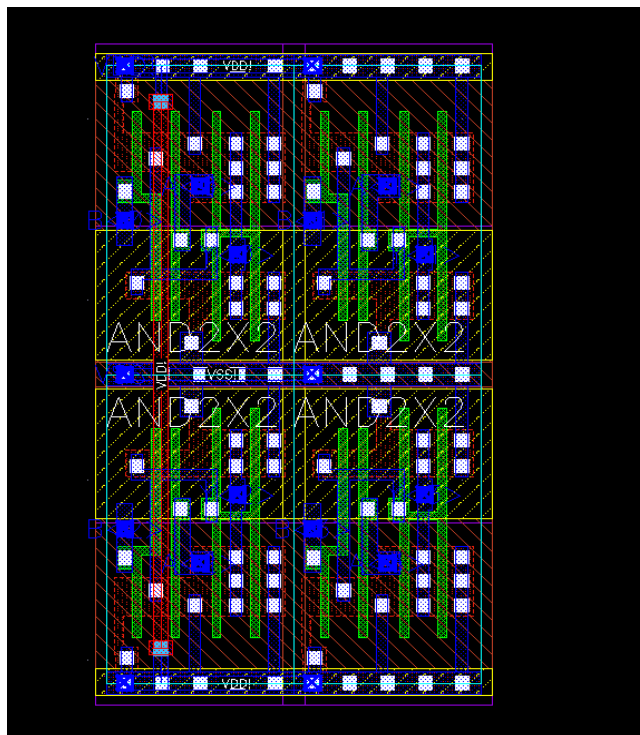


Figure 6: 4-bit AND layout

4-bit MUX

Schematic Print

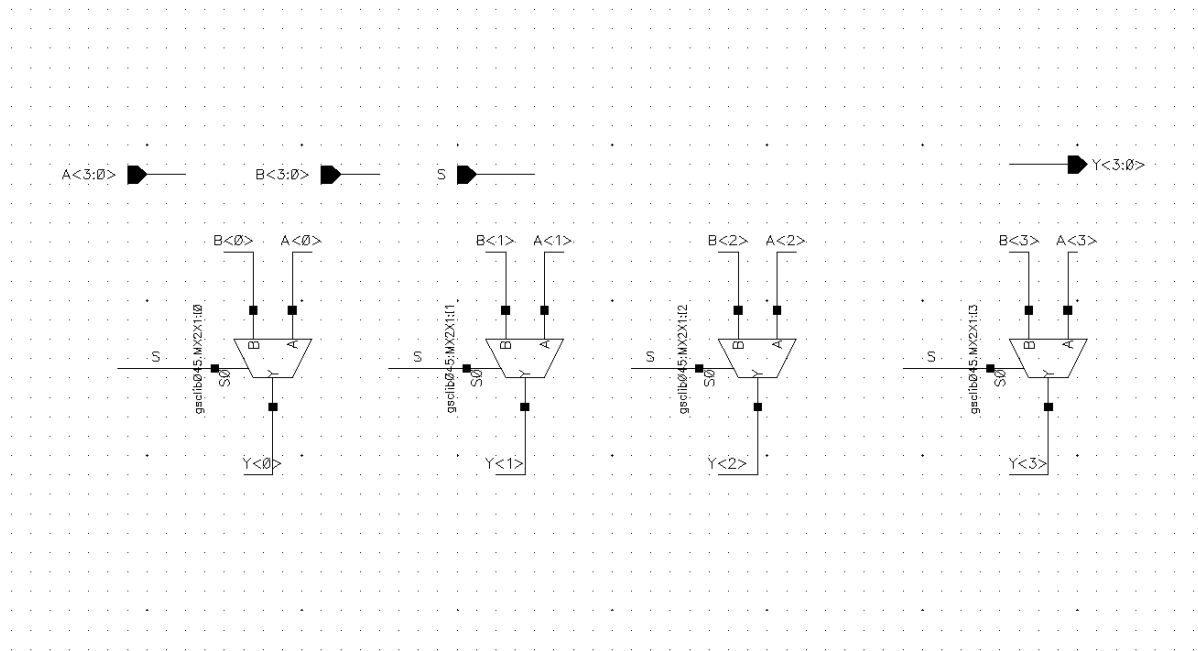


Figure 7: 4-bit MUX schematic

Layout Print

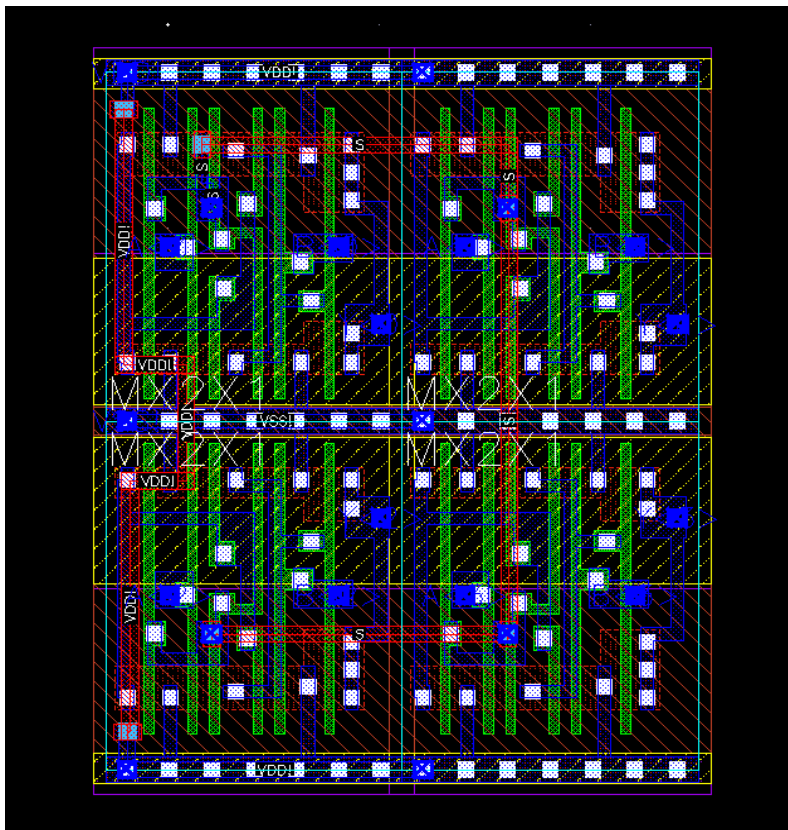


Figure 8: 4-bit MUX layout

4-bit Register

Schematic Print

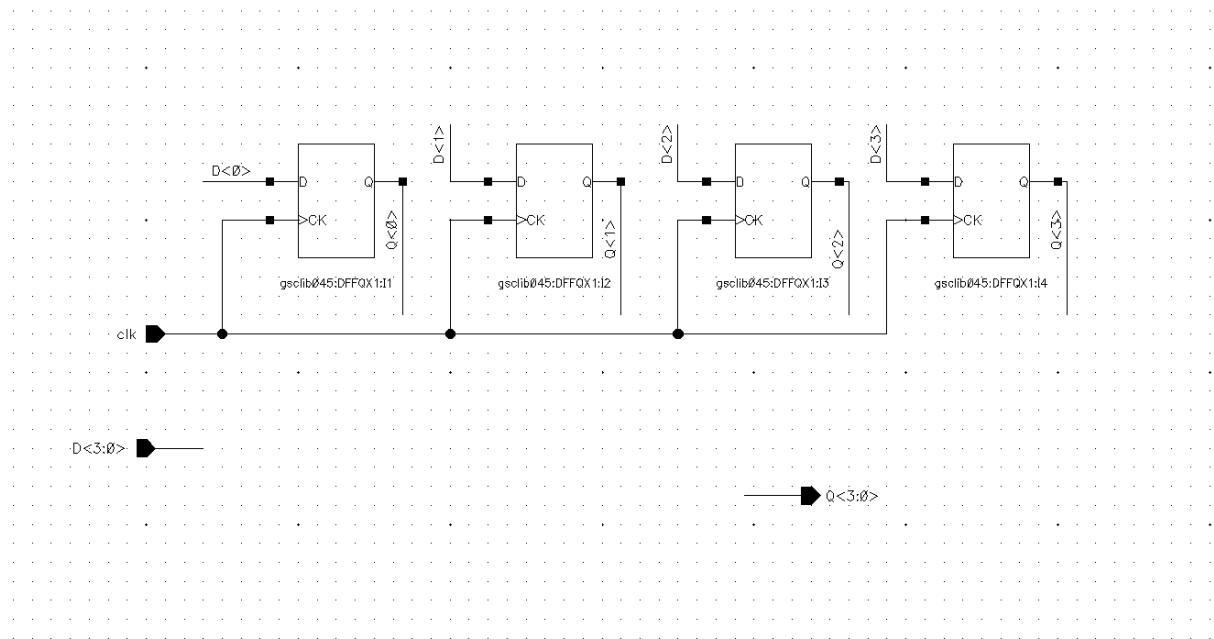


Figure 9: 4-bit register schematic

Layout Print

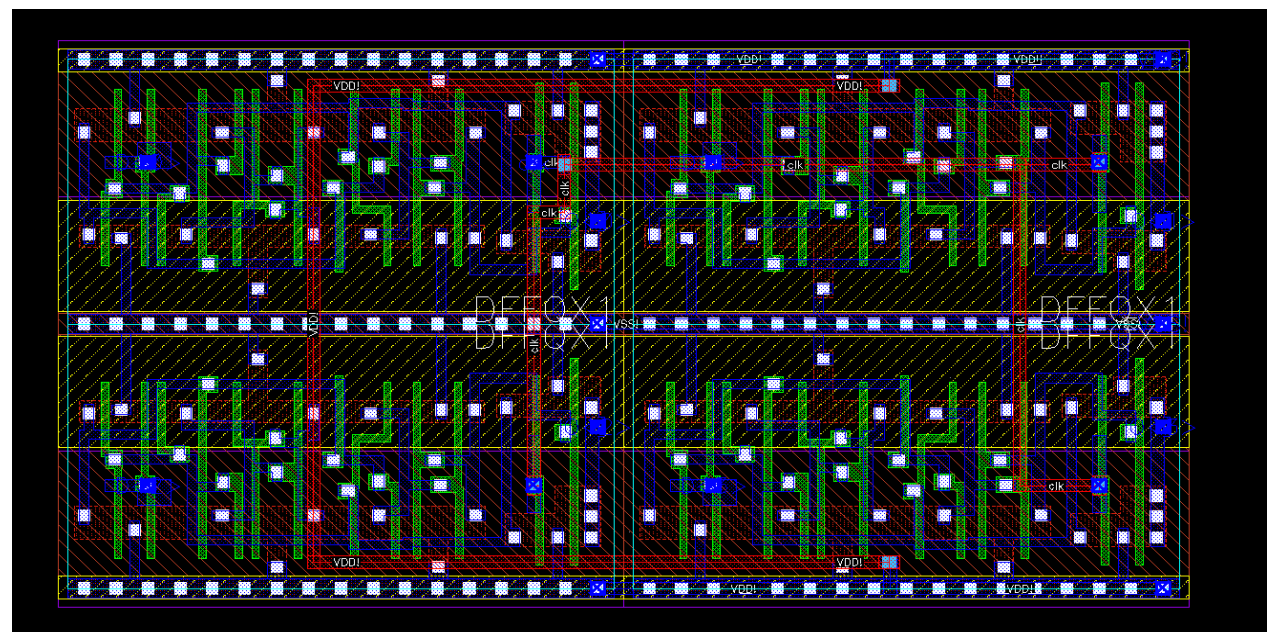


Figure 10: 4-bit register layout

5-bit Register

Schematic Print

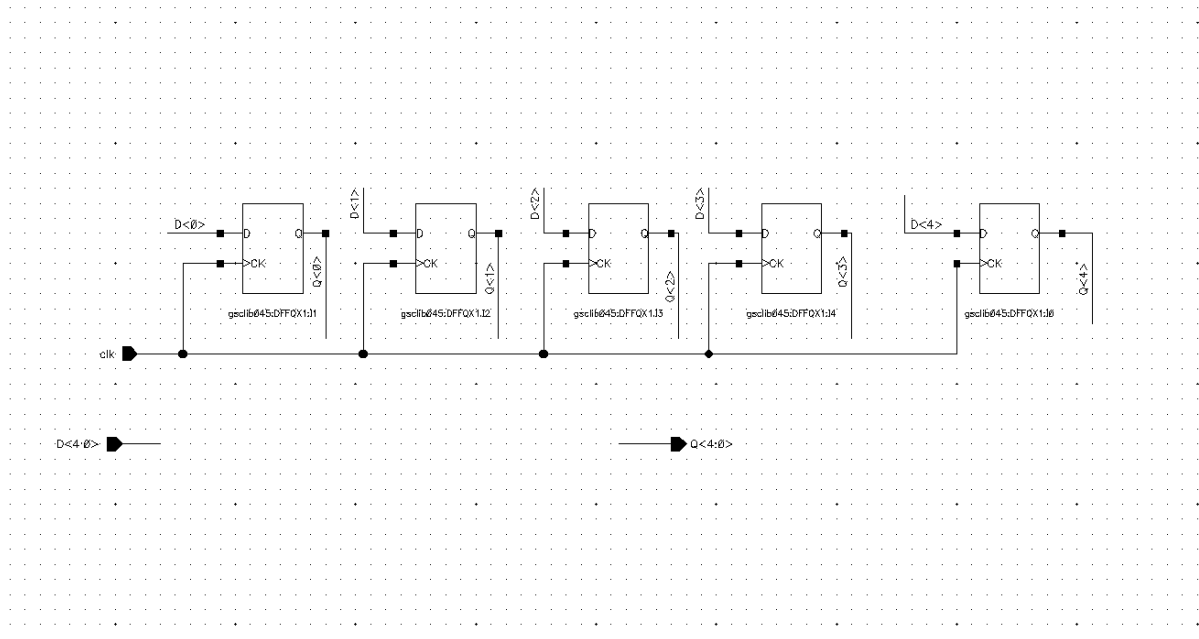


Figure 11: 5-bit register schematic

Layout Print

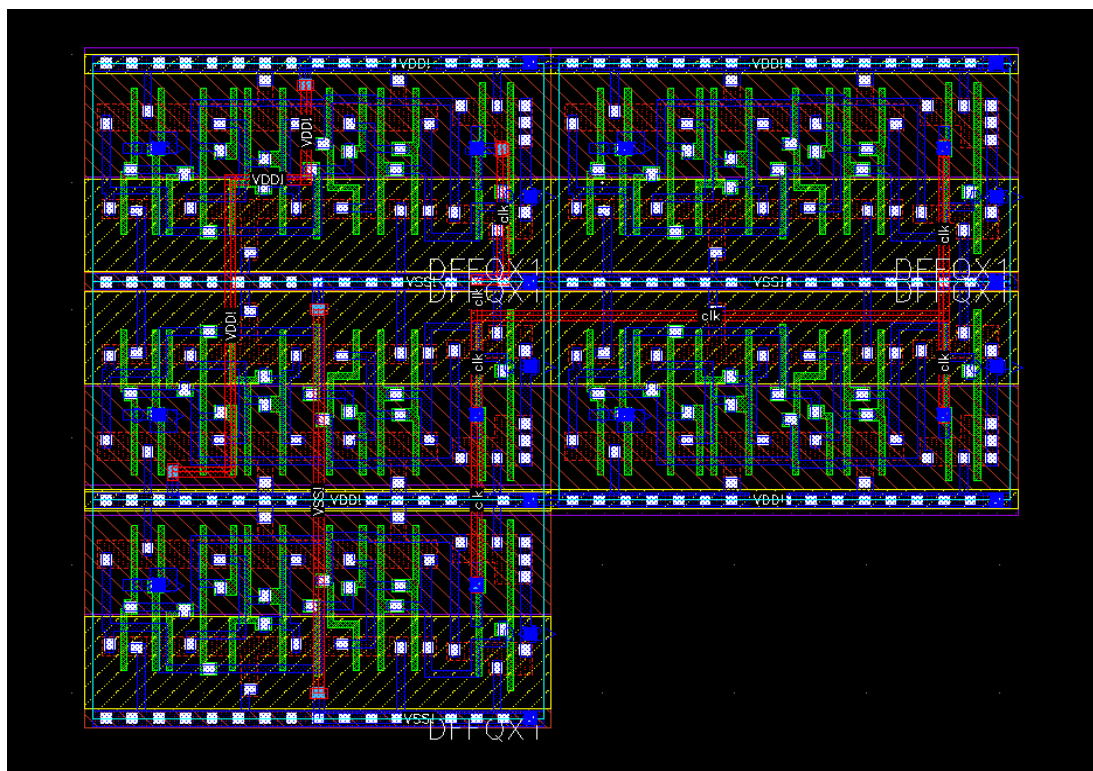


Figure 12: 5-bit register layout

Bit-extended XOR

Schematic print

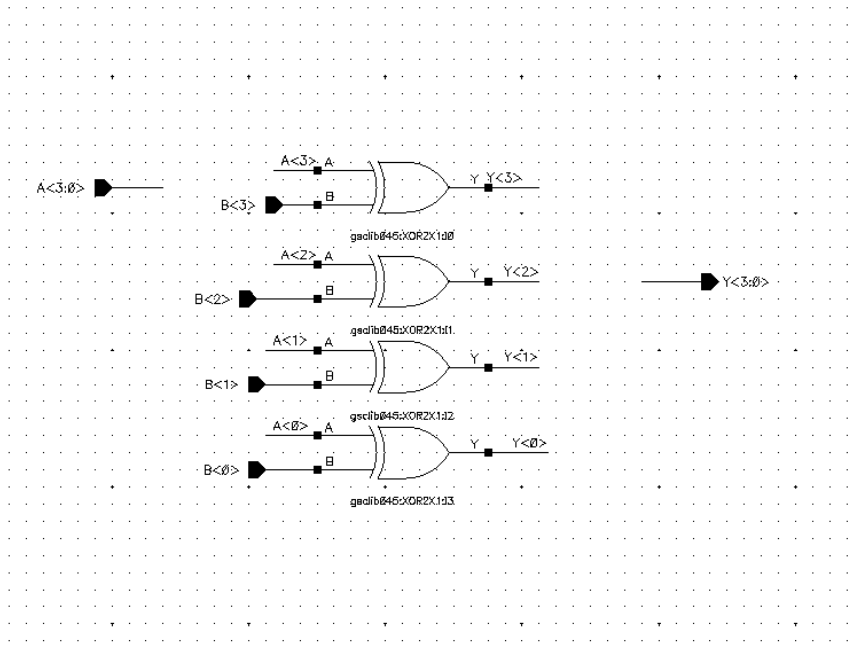


Figure 13: XOR bit extend schematic

Layout Print

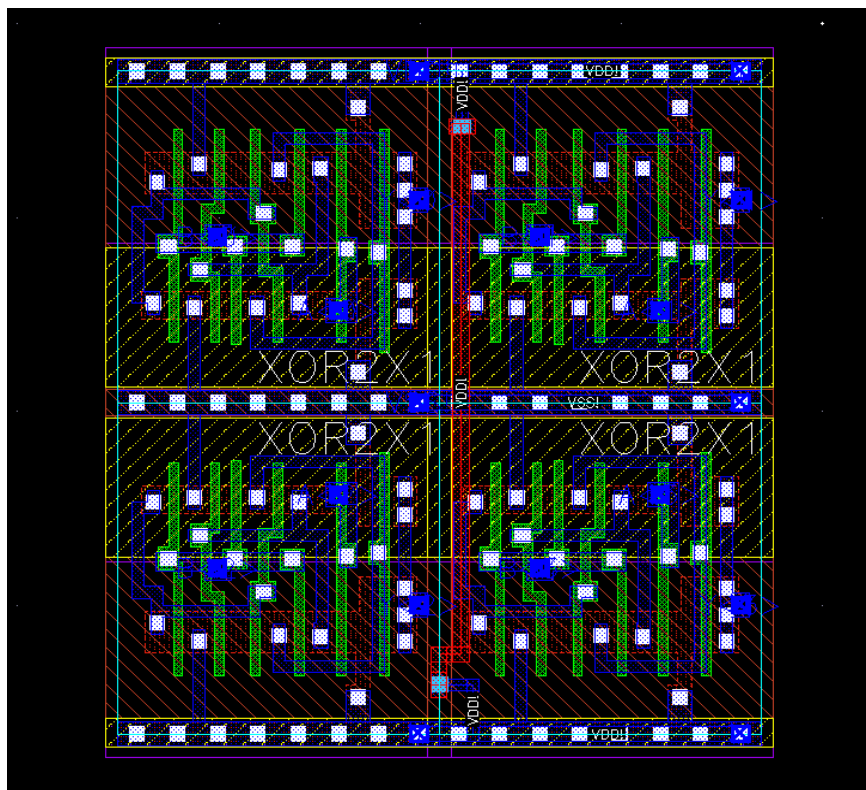


Figure 14: Bit extended XOR layout

Overview circuit

Schematic Print

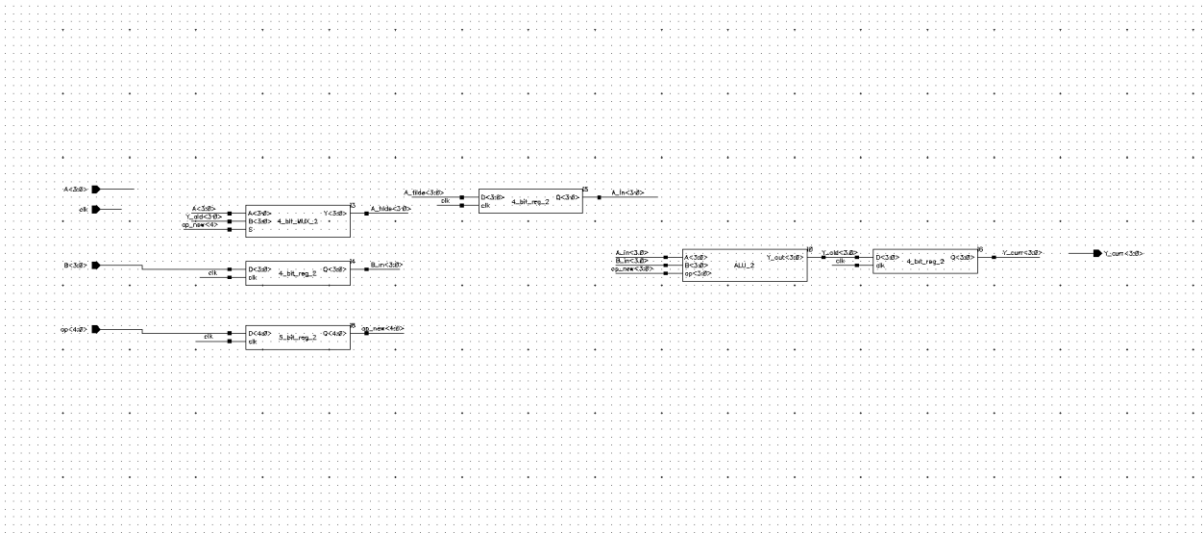


Figure 15: Overview circuit schematic

Layout Print

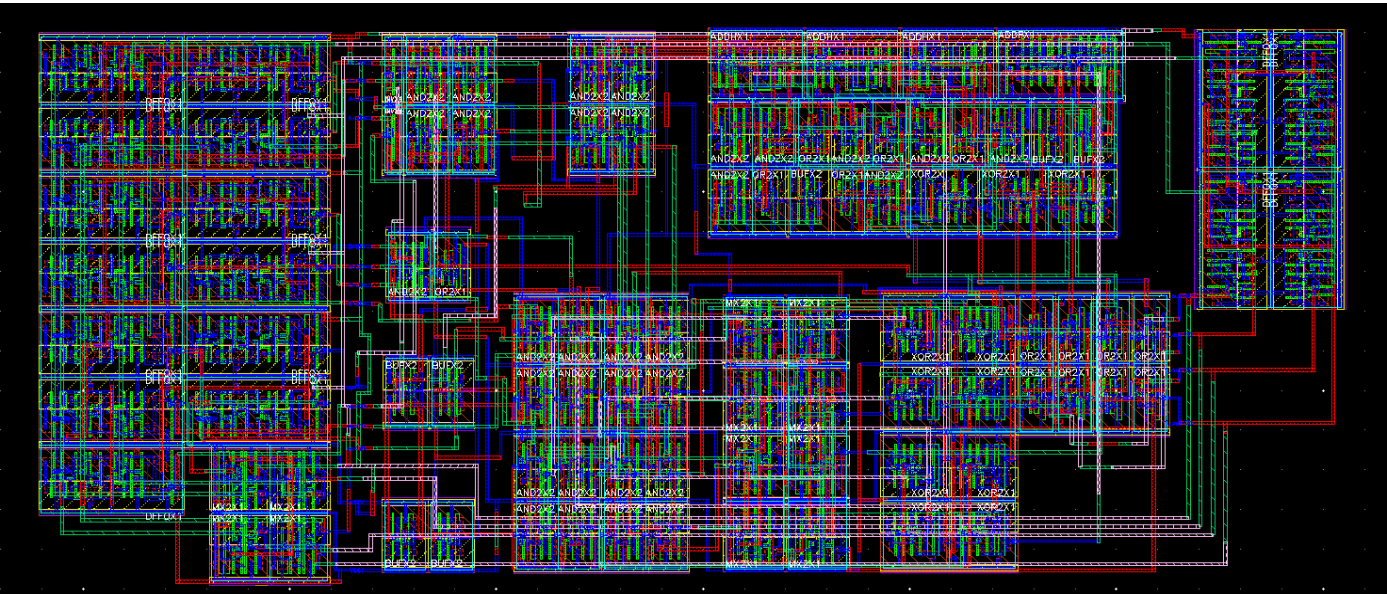


Figure 16: Overview circuit layout

ronabraham library

4-bit XOR

Schematic Print

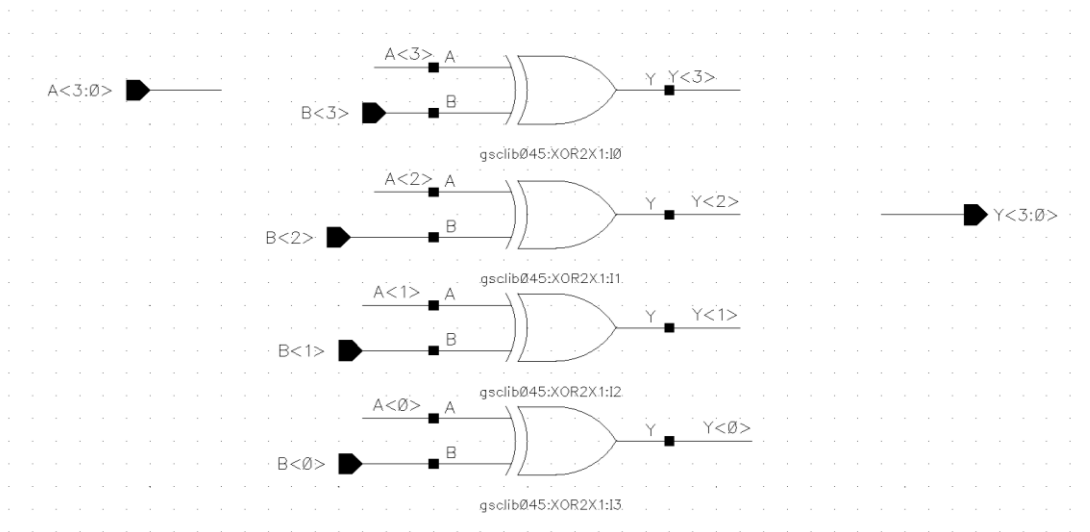


Figure 17: 4-bit XOR schematic

Layout Print

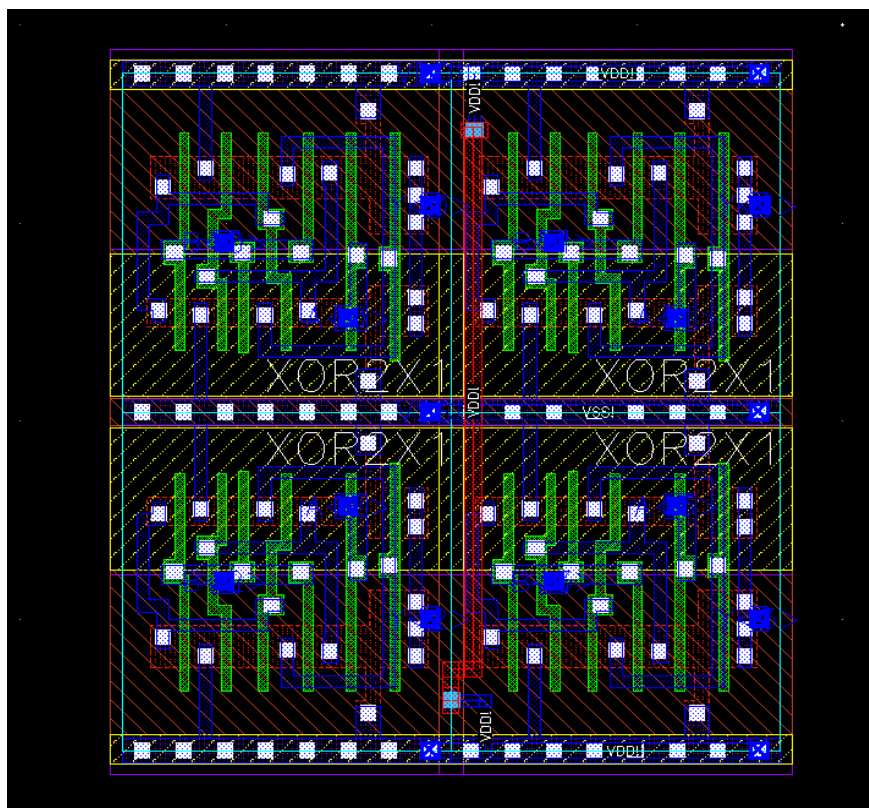


Figure 18: 4-bit XOR layout

shaikhohen library

Barrel Right-Shift register

Schematic Print

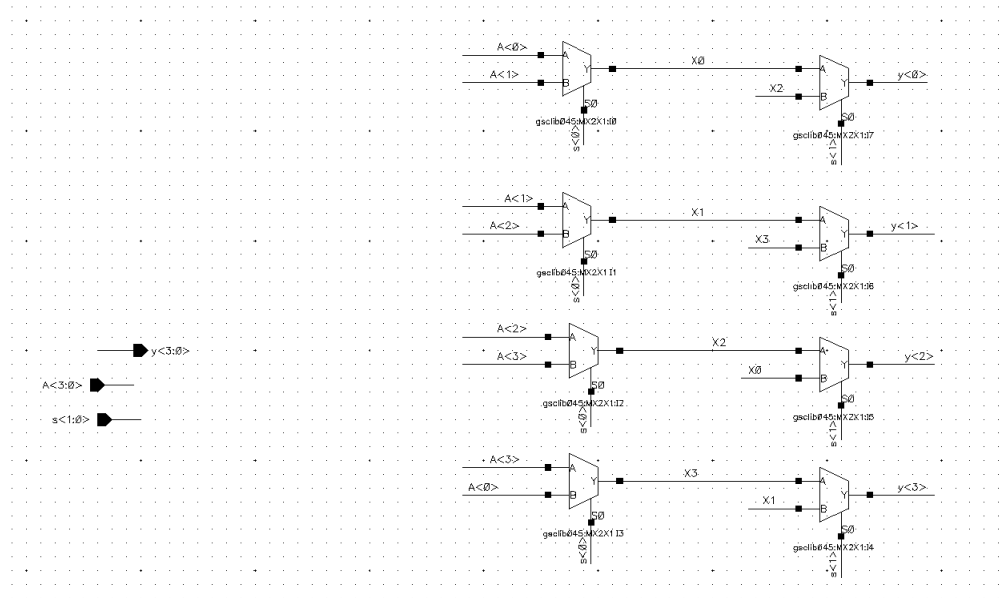


Figure 19: Barrel Shift Register schematic

Layout Print

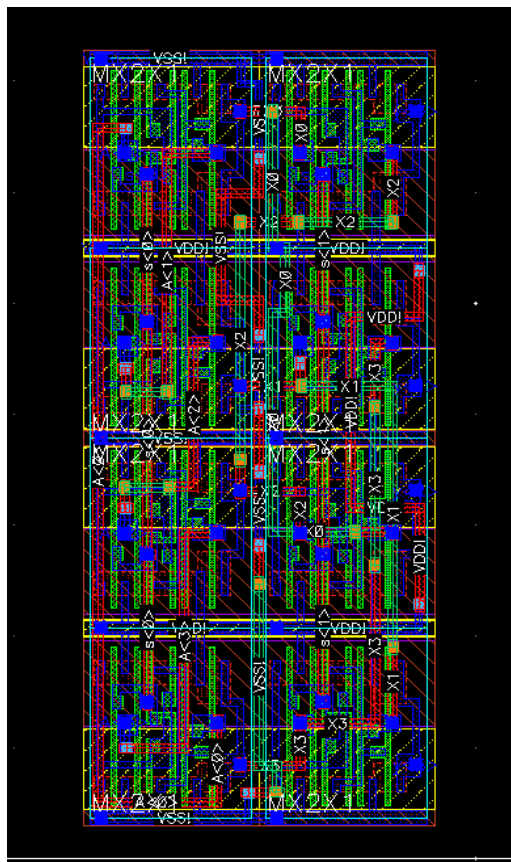


Figure 20: Barrel Shifter Layout

rahulxena library

Adder block

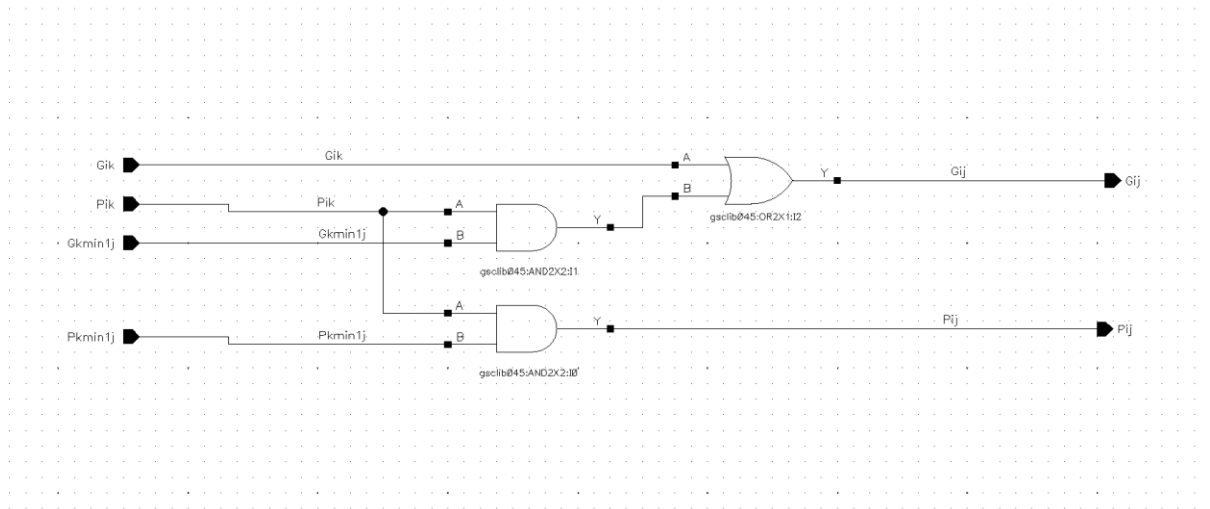


Figure 21: Black cell schematic

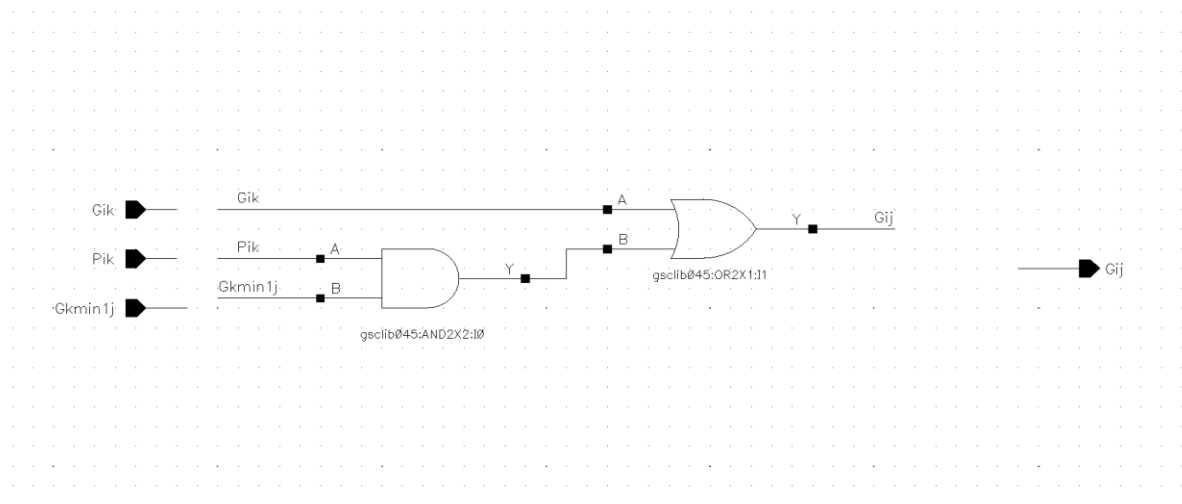


Figure 22: Grey cell schematic

ALU block

Schematic Print

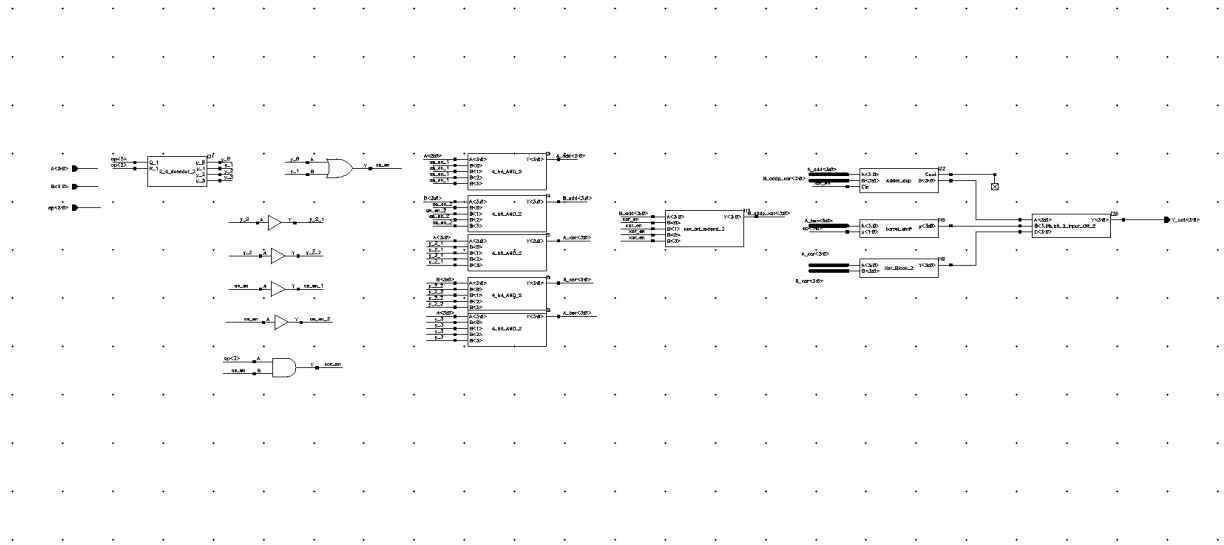


Figure 25: ALU schematic

Layout Print

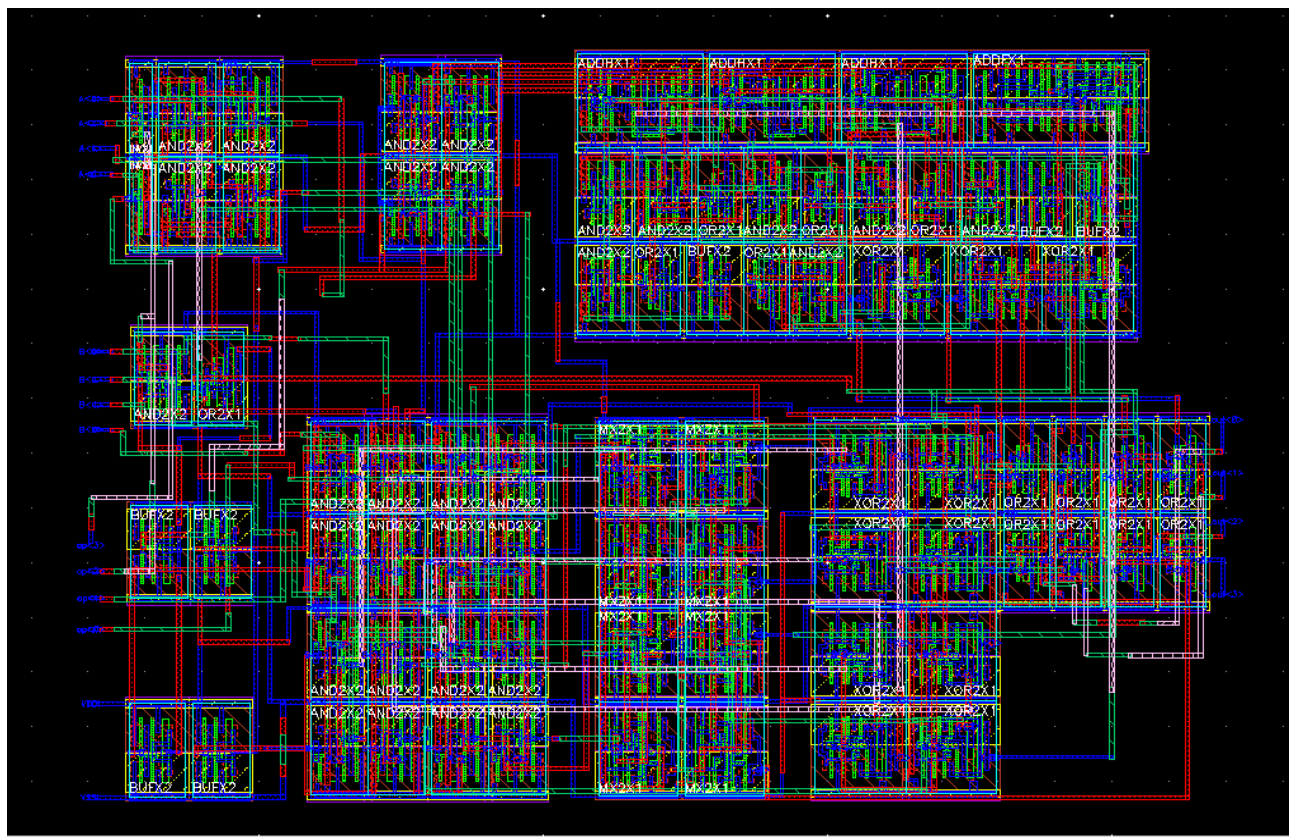


Figure 26: ALU layout

Floor Plan

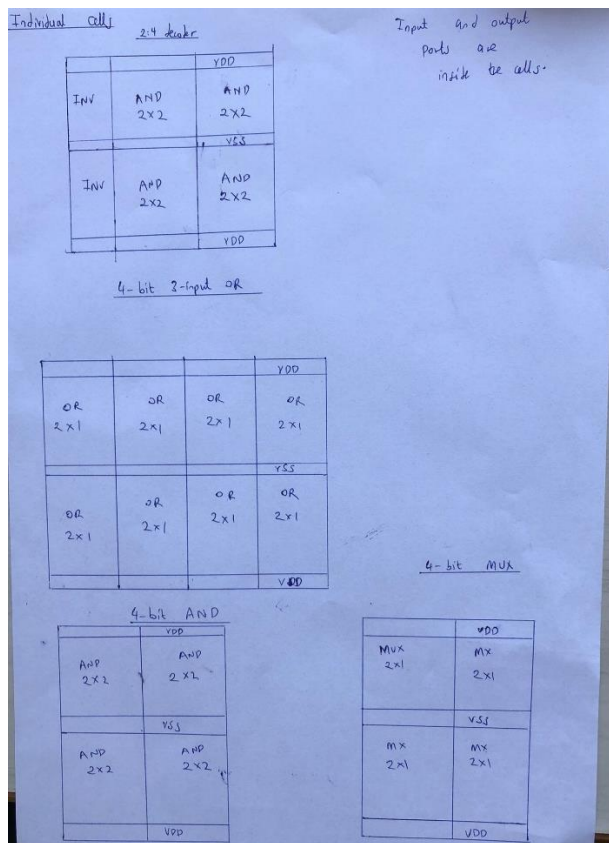


Figure 27: cells design for the layout 1

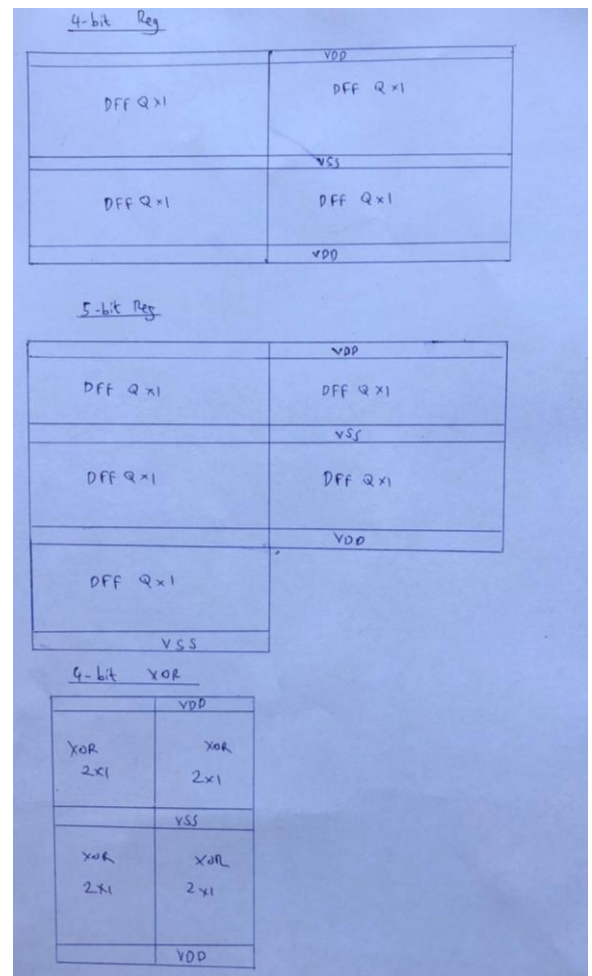


Figure 28: cells design for the layout 2

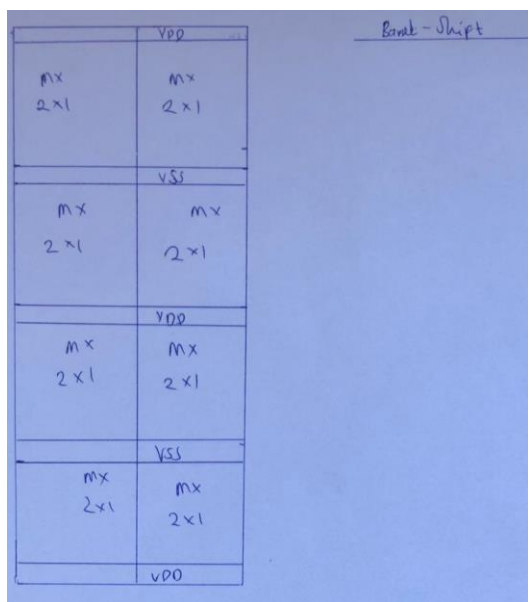


Figure 29 cells design for the layout 3

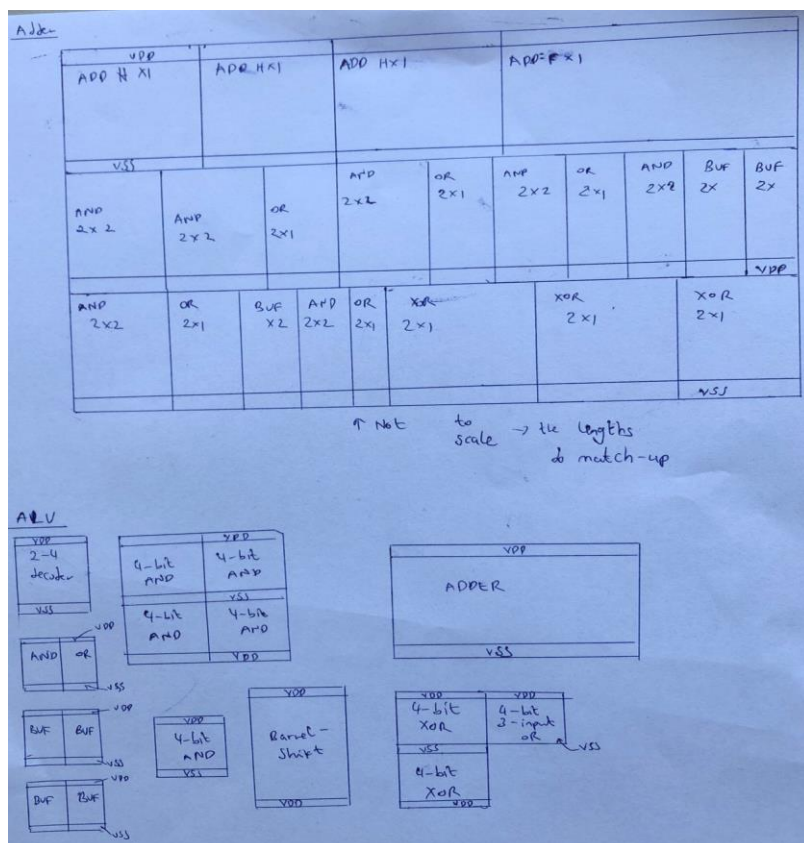


Figure 30: ALU and Adder design

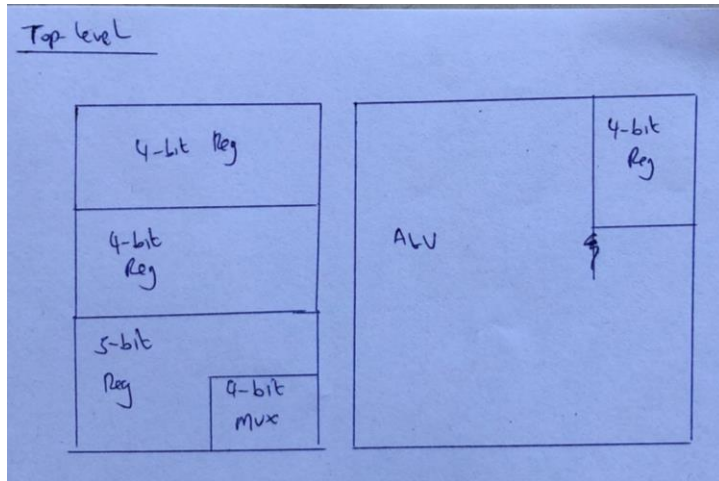


Figure 31: top level circuit design

Realization area

The final realization area is $31.615\mu\text{m}$ by $13.92\mu\text{m}$ = $440.0808\mu\text{m}^2$ which is less than the projected area of $442.0992\mu\text{m}^2$ which thus satisfies the constraint of the realization area not occupying more than 150% of the total area of the cells. In addition to this, only 4 layers of metal were used in the making of the final circuit with every individual block before the ALU using only 3 layers of metal at maximum.

DRC and LVS confirmation

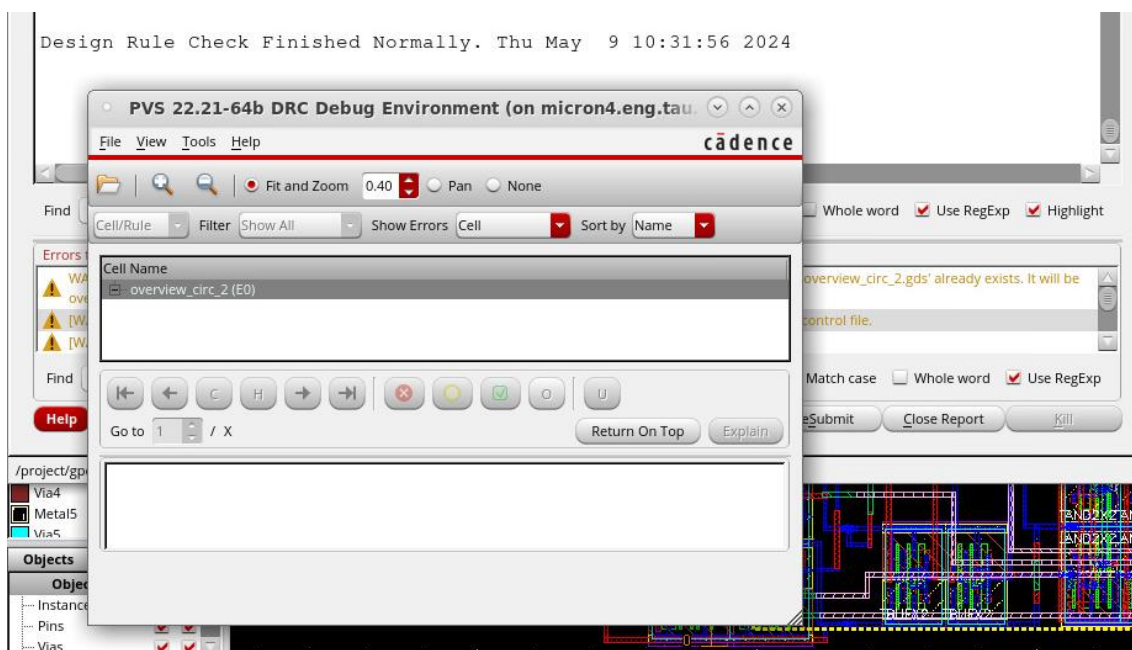


Figure 32: DRC confirmation of no errors on final layout

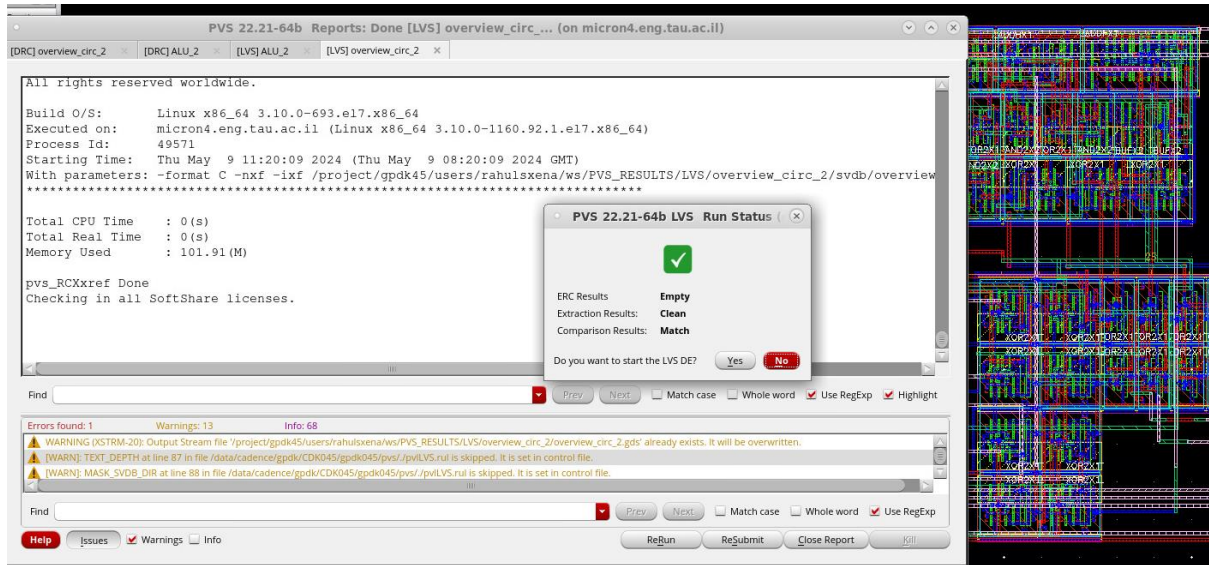


Figure 33: Successful LVS Run on final layout with no errors

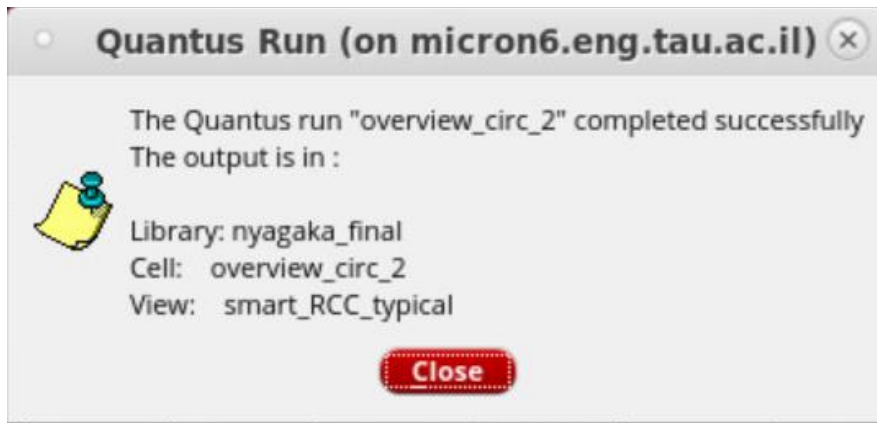


Figure 34: Successful extraction of QRC on final layout

Critical path delay and maximal frequency evaluation

A note for this and all tests going forward, the results were obtained for inputs with a rise and fall time of 1 femtosecond which is below the 50ps limit stated in the outline.

Based off the setup of the ALU, it is determined that the critical path is the SUBTRACTION operation which goes from:

clock input of B reg → 4-bit AND → 4-bit XOR → ADDER → 4-bit-OR → input of output reg Y

We begin by measuring the delay of the logic between the 2 clocks and we note that it was measured for different voltages and with and without QRC extraction below.

Different input patterns were used to yield the maximal delay between the rising edge of the input and the rising edge of the output and the one pattern that yielded the longest delay was when we had all 1's at the output to yield 1111 i.e. A=0 and B=1 to yield A-B=-1.

Name	Type	Details	Value	Plot	Save	Spec
filter	signal	/Y<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/A<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/B<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/op<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit2	expr	delayMeasure(VT("/B<2>") VT("/Y<1>") tedge1 "rising" "value1 0.6 7e...	4.142n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit3	expr	delayMeasure(VT("/B<0>") VT("/Y<2>") tedge1 "rising" "value1 0.6 7e...	4.18n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit11	expr	delayMeasure(VT("/A<3>") VT("/Y<3>") tedge1 "rising" "value1 0.6 7e...	4.152n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 35: Tcritical without QRC for VDD = 1.2V

Name	Type	Details	Value	Plot	Save	Spec
filter	signal	/Y<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/A<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/B<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/op<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit2	expr	delayMeasure(VT("/B<2>") VT("/Y<1>") tedge1 "rising" "value1 0.6 7e...	4.441n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit3	expr	delayMeasure(VT("/B<0>") VT("/Y<2>") tedge1 "rising" "value1 0.6 7e...	4.545n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit11	expr	delayMeasure(VT("/A<3>") VT("/Y<3>") tedge1 "rising" "value1 0.6 7e...	4.476n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 36: Tcritical with QRC for VDD = 1.2V

Name	Type	Details	Value	Plot	Save	Spec
	signal	/Y<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
	signal	/A<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
	signal	/B<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
	signal	/op<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit2	expr	delayMeasure(VT("/B<2>") VT("/Y<1>") tedge1 "rising" "value1 0.35 7e...	19.6n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit3	expr	delayMeasure(VT("/B<0>") VT("/Y<2>") tedge1 "rising" "value1 0.35 7e...	19.86n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit11	expr	delayMeasure(VT("/A<3>") VT("/Y<3>") tedge1 "rising" "value1 0.35 7e...	19.79n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 37: Tcritical without QRC for VDD=0.7V

Name	Type	Details	Value	Plot	Save	Spec
filter	signal	/Y<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/A<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/B<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
filter	signal	/op<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit2	expr	delayMeasure(VT("/B<2>") VT("/Y<1>") tedge1 "rising" "value1 0.35 7e...	21.48n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit3	expr	delayMeasure(VT("/B<0>") VT("/Y<2>") tedge1 "rising" "value1 0.35 7e...	22.34n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tcrit11	expr	delayMeasure(VT("/A<3>") VT("/Y<3>") tedge1 "rising" "value1 0.35 7e...	21.77n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 38: Tcritical with QRC for VDD=0.7V

The **delayMeasure** function proved useful as it enabled us to measure the delay that we have described above.

The above results are tabulated below.

	$V_{DD} = 1.2V$	$V_{DD} = 0.7V$
t_{crit} (without QRC)	4.18n	19.86n
t_{crit} (with QRC)	4.545n	22.34n

To calculate the maximal frequency or minimal period we use the derivation from the lecture that $T_{min} \geq t_{cq} + t_{logic} + t_{su}$

Where in our case t_{su} is the setup time, t_{cq} is the delay from the rising clock edge to when the output of the input register goes high and t_{logic} is the logic delay of the circuit. We've already found t_{logic} and all that's left to find is t_{su} and t_{cq} .

Some conclusions can be made from the above evaluations:

- 1) A decrease in VDD is detrimental to the delay of the circuit as covered in the course. The lower VDD is, the longer it takes for the circuit to respond hence lowering the overall maximal frequency that the circuit can achieve.
- 2) Including the QRC extraction increases the delay. This makes sense as often, the simulations that don't take into account the parasitic resistances and capacitances and are far too optimistic and once the parasitics are extracted we get a more pessimistic but also realistic view of the operation of the circuit.

t_{cq} evaluation

We begin by finding t_{cq} as this will later assist us in finding the setup time.

To get t_{cq} we simply set a testbench that triggers the outputs of the shift registers to go high and measure the delay from the rising clock edge before the output goes high to the rising edge at the output. Delay measure is once again useful in this regard and the results are shown below and subsequently tabulated.



The screenshot shows a simulation tool interface. On the left, a 'Design Variables' panel lists parameters: T_clk (100n), VDD (1.2), t_delay_vdd (-1n), t_delay_vdd_0.5 (0.5*T_clk), trise (0.002*T_clk), tfall (trise), and k (0). On the right, a table displays simulation results for various signals and expressions. The table has columns for Name, Type, Details, Value, Plot, Save, and Spec. The 'expr' row shows a delay measurement result of 4.066n.

Name	Type	Details	Value	Plot	Save	Spec
actual_tcq_test_VDD_12	signal	/Q<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Simulator_spectre	signal	/D<3:0>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Analyses	signal	/clk		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
tran 0.16*VDD("T_clk")	expr	delayMeasure(VT("clk") VT("Q<3:0>") ?edge1 "rising" ?nth1 2 ?value1 0.6 ?edge2 "rising" ?value2 0.6)	4.066n	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 39: t_{cq} measurement at $V_{DD} = 1.2V$ with and without QRC

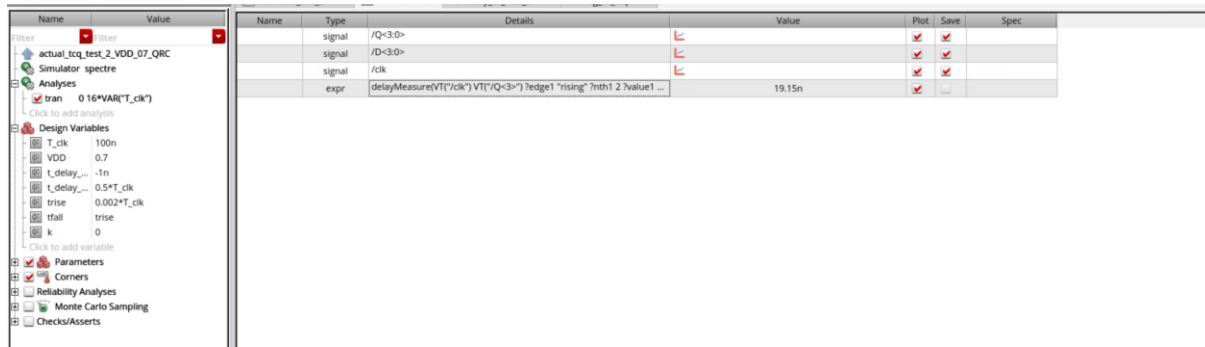


Figure 40: tcq measurement at VDD = 0.7V with and without QRC

	$V_{DD} = 1.2V$	$V_{DD} = 0.7V$
t_{cq} (with and without QRC)	4.066n	19.15n

Immediately from the above we can see that the delay is basically identical with and without parasitics and this similarly makes sense. When making the layout for our register, there was a minimal use for wiring and as a result, the performance of the register is pretty much identical to that of the ideal simulation.

As mentioned in the previous evaluation, the decrease in VDD increases the delay. In the course, we mention that increasing V_{DD} (up to a certain point) is one of the ways in which we can reduce the propagation delay of our gates. Thus, a decrease in VDD is only cause of increased delay.

Setup time evaluation

Setup time is the minimum time for which the input must be stable before the clock edge rises in order for proper sampling to occur.

To find this, we use the method shown in the recitation where you kind of brute force using a sweep on the delay of the inputs and then after finding a reasonable range of values, use pass-fail testing based on the tcq delay of the cycle.

The pass-fail testing works such that if the delay between the rising clock edge at the time we expect the input to rise is much greater than t_{cq} , then the test fails else if the delay is within reason, then the test passes. This makes sense as if the setup time constraint isn't met then the input signal will be sampled in the next clock cycle. Below are the results presented visually but also tabulated.

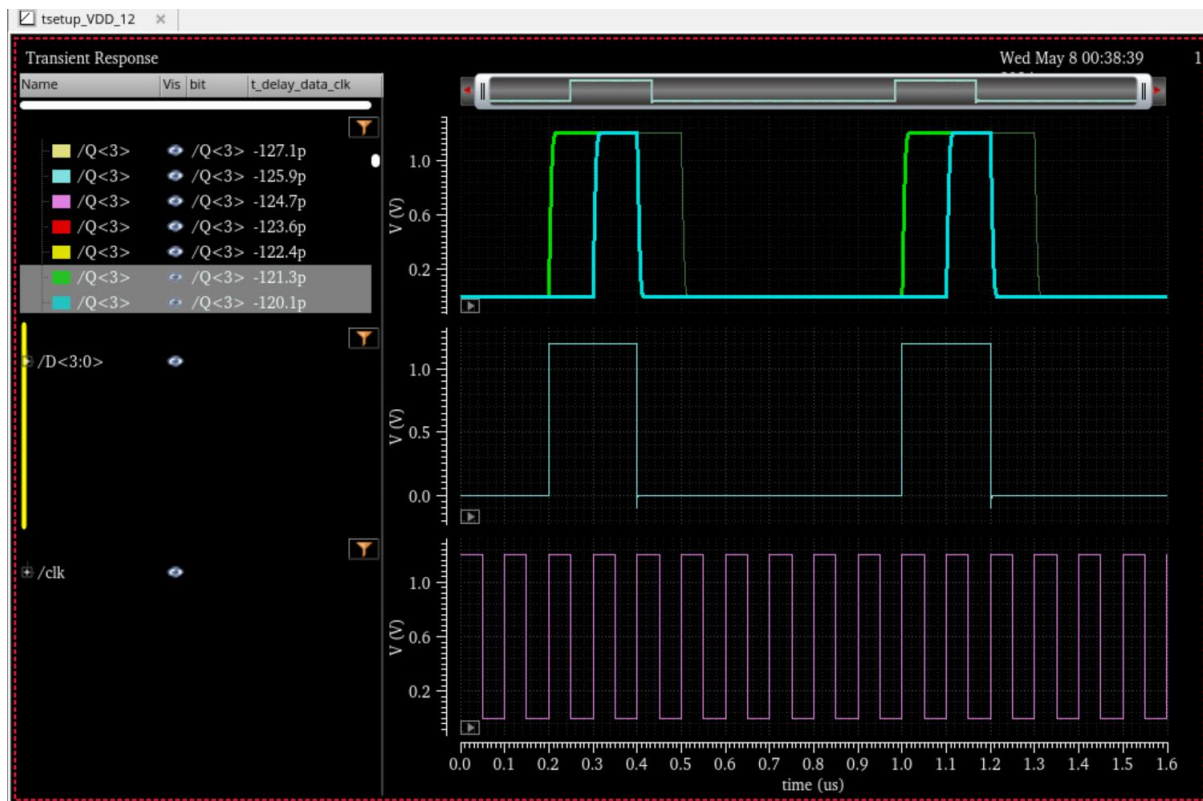


Figure 41: tsetup graph visual for VDD=1.2V

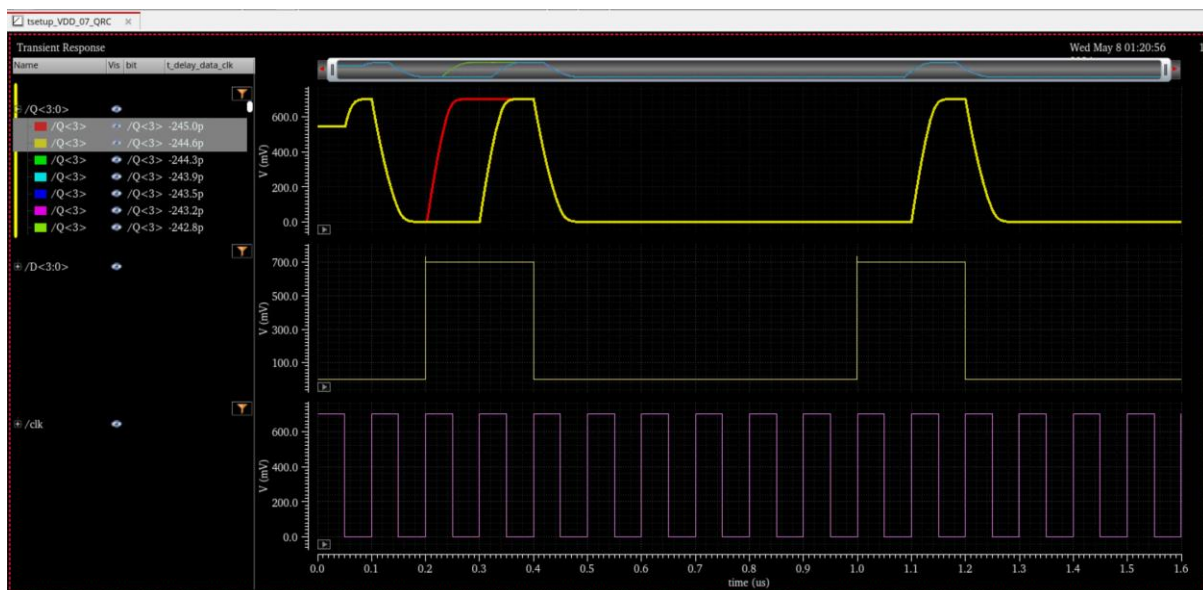


Figure 42 tsetup graph visual for VDD=0.7V

It must be noted that in the above graphs, we were unable to take a print as shown in the recitations as the print wouldn't properly display the transition we were trying to show above.

Next we use pass-fail testing to narrow down the actual t_{su} value.

Parameters: t_delay_data_clk=-244p					
1	tsetup_VDD_07	/Q<3:0>			
1	tsetup_VDD_07	/D<3:0>			
1	tsetup_VDD_07	/clk			
1	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		119.2n	tol 20n 10% fail
Parameters: t_delay_data_clk=-244.6p					
2	tsetup_VDD_07	/Q<3:0>			
2	tsetup_VDD_07	/D<3:0>			
2	tsetup_VDD_07	/clk			
2	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		119.2n	tol 20n 10% fail
Parameters: t_delay_data_clk=-245.1p					
3	tsetup_VDD_07	/Q<3:0>			
3	tsetup_VDD_07	/D<3:0>			
3	tsetup_VDD_07	/clk			
3	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		21.28n	tol 20n 10% pass
Parameters: t_delay_data_clk=-245.7p					
4	tsetup_VDD_07	/Q<3:0>			
4	tsetup_VDD_07	/D<3:0>			
4	tsetup_VDD_07	/clk			
4	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		20.68n	tol 20n 10% pass
Parameters: t_delay_data_clk=-246.2p					
5	tsetup_VDD_07	/Q<3:0>			
5	tsetup_VDD_07	/D<3:0>			
5	tsetup_VDD_07	/clk			
5	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		20.42n	tol 20n 10% pass

Figure 43: t_{su} graph 1 for VDD=0.7V

Parameters: t_delay_data_clk=-246.8p					
6	tsetup_VDD_07	/Q<3:0>			
6	tsetup_VDD_07	/D<3:0>			
6	tsetup_VDD_07	/clk			
6	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		20.15n	tol 20n 10% pass
Parameters: t_delay_data_clk=-247.3p					
7	tsetup_VDD_07	/Q<3:0>			
7	tsetup_VDD_07	/D<3:0>			
7	tsetup_VDD_07	/clk			
7	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		20.06n	tol 20n 10% pass
Parameters: t_delay_data_clk=-247.9p					
8	tsetup_VDD_07	/Q<3:0>			
8	tsetup_VDD_07	/D<3:0>			
8	tsetup_VDD_07	/clk			
8	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		19.95n	tol 20n 10% pass
Parameters: t_delay_data_clk=-248.4p					
9	tsetup_VDD_07	/Q<3:0>			
9	tsetup_VDD_07	/D<3:0>			
9	tsetup_VDD_07	/clk			
9	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		19.81n	tol 20n 10% pass
Parameters: t_delay_data_clk=-249p					
10	tsetup_VDD_07	/Q<3:0>			
10	tsetup_VDD_07	/D<3:0>			
10	tsetup_VDD_07	/clk			
10	tsetup_VDD_07	delayMeasure(VT("clk") VT("/Q<3>") ?edge1 "rising" ?nth1 2 ?value1 0.35 ?edge2 "rising" ?value2 0.35)		19.76n	tol 20n 10% pass

Figure 44: t_{su} graph 2 for VDD=0.7V

Parameters: t_delay_data_clk=-117p					
1	tsetup_VDD_12	/Q<3:0>			
1	tsetup_VDD_12	/D<3:0>			
1	tsetup_VDD_12	/clk			
1	tsetup_VDD_12	delayMeasure(V...		102.3n	< 5n fail
Parameters: t_delay_data_clk=-117.9p					
2	tsetup_VDD_12	/Q<3:0>			
2	tsetup_VDD_12	/D<3:0>			
2	tsetup_VDD_12	/clk			
2	tsetup_VDD_12	delayMeasure(V...		102.3n	< 5n fail
Parameters: t_delay_data_clk=-118.8p					
3	tsetup_VDD_12	/Q<3:0>			
3	tsetup_VDD_12	/D<3:0>			
3	tsetup_VDD_12	/clk			
3	tsetup_VDD_12	delayMeasure(V...		102.3n	< 5n fail
Parameters: t_delay_data_clk=-119.7p					
4	tsetup_VDD_12	/Q<3:0>			
4	tsetup_VDD_12	/D<3:0>			
4	tsetup_VDD_12	/clk			
4	tsetup_VDD_12	delayMeasure(V...		102.3n	< 5n fail
Parameters: t_delay_data_clk=-120.6p					
5	tsetup_VDD_12	/Q<3:0>			
5	tsetup_VDD_12	/D<3:0>			
5	tsetup_VDD_12	/clk			
5	tsetup_VDD_12	delayMeasure(V...		2.321n	< 5n pass

Figure 45: t_{su} graph 1 for VDD=1.2V
















Parameters: t_delay_data_clk=-121.4p					
6	tsetup_VDD_12	/Q<3:0>			
6	tsetup_VDD_12	/D<3:0>			
6	tsetup_VDD_12	/clk			
6	tsetup_VDD_12	delayMeasure(V...		2.308n	< 5n pass
Parameters: t_delay_data_clk=-122.3p					
7	tsetup_VDD_12	/Q<3:0>			
7	tsetup_VDD_12	/D<3:0>			
7	tsetup_VDD_12	/clk			
7	tsetup_VDD_12	delayMeasure(V...		2.306n	< 5n pass
Parameters: t_delay_data_clk=-123.2p					
8	tsetup_VDD_12	/Q<3:0>			
8	tsetup_VDD_12	/D<3:0>			
8	tsetup_VDD_12	/clk			
8	tsetup_VDD_12	delayMeasure(V...		2.304n	< 5n pass
Parameters: t_delay_data_clk=-124.1p					
9	tsetup_VDD_12	/Q<3:0>			
9	tsetup_VDD_12	/D<3:0>			
9	tsetup_VDD_12	/clk			
9	tsetup_VDD_12	delayMeasure(V...		2.301n	< 5n pass
Parameters: t_delay_data_clk=-125p					
10	tsetup_VDD_12	/Q<3:0>			
10	tsetup_VDD_12	/D<3:0>			
10	tsetup_VDD_12	/clk			
10	tsetup_VDD_12	delayMeasure(V...		2.302n	< 5n pass

Figure 46: t_{su} graph 2 for VDD=1.2V

From the above, the setup times is tabulated below. It should be noted that the setup times with and without QRC were pretty much identical so these graphs suffice.

	$V_{DD} = 1.2V$	$V_{DD} = 0.7V$
t_{su} (with and without QRC)	0.1214n	0.2451n

So from the above, we find the minimal time periods for each scenario:

Source Voltage	QRC	Without QRC
VDD = 1.2	(0.1214+4.066+4.545)n=8.7324n	(0.1214+4.066+4.18)n = 8.3673n
VDD=0.7	(0.2451+19.15+22.34)n = 41.735n	(0.2451+19.15+19.86)n = 39.255n

Taking reciprocal of minimum period gives maximum frequency:

Source Voltage	QRC	Without QRC
VDD = 1.2	114.516 MHz	119.513 MHz
VDD = 0.7	23.96 MHz	25.474 MHz

For the t_{cq} measurement and t_{logic} , it's crucial to have the threshold in delay measure set to 0.6V for VDD = 1.2V and 0.35 for VDD =0.7V else you risk erroneous measurements.

From the final evaluated values above, we can conclude that the supply voltage has a very huge effect on the minimal period of the circuit. This is due to the increase in the propagation delays which was already explained above. Similarly, there is a deviation in the minimal period for the same VDD but with extracted QRC parameters. This was also explained above, but in summary j QRC extracted circuits are closer to practicality and in reality, interconnects and vias contribute to increased RC delays.

Simulation results of proper working

Top view circuit test with calculated maximal frequencies

operation	feedback	Op<4>	Op<3>	Op<2>	Op<1>	Op<0>
Addition	No	0	0	0	*	*
Subtraction	No	0	0	1	*	*
XOR	No	0	1	0	*	*
Shift by 2	No	0	1	1	1	0
Addition	Yes	1	0	0	*	*
Subtraction	Yes	1	0	1	*	*
XOR	Yes	1	1	0	*	*
Shift by 2	Yes	1	1	1	1	0

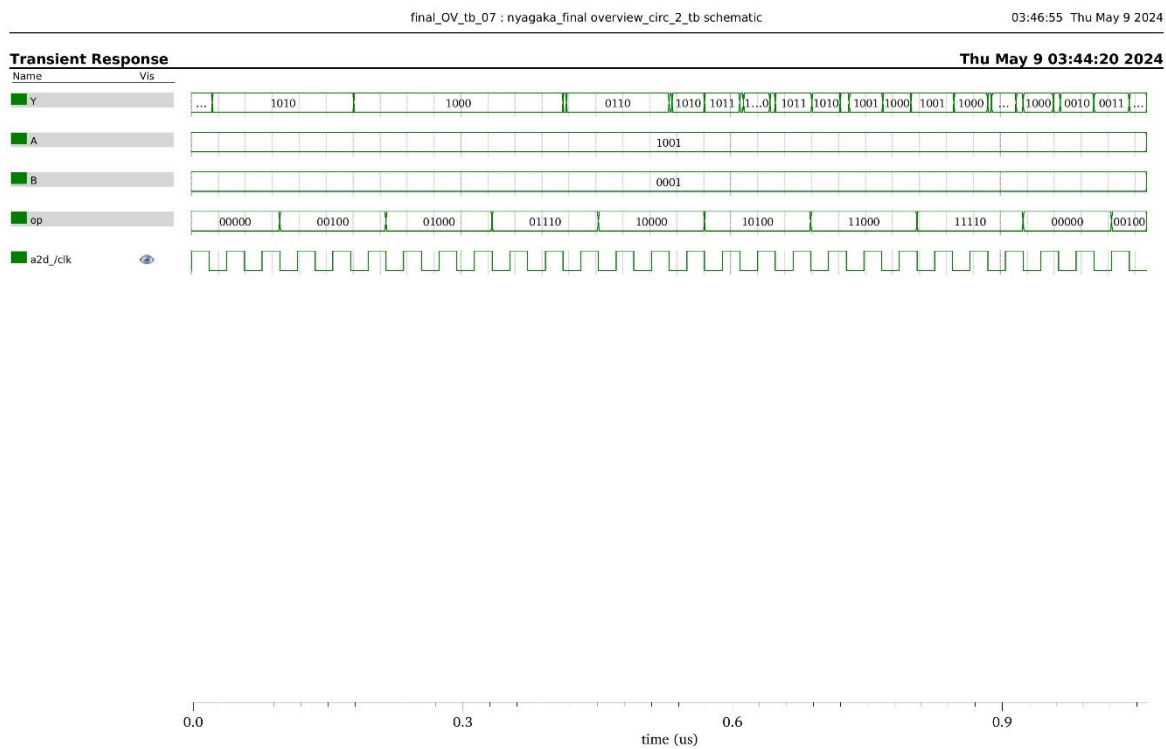
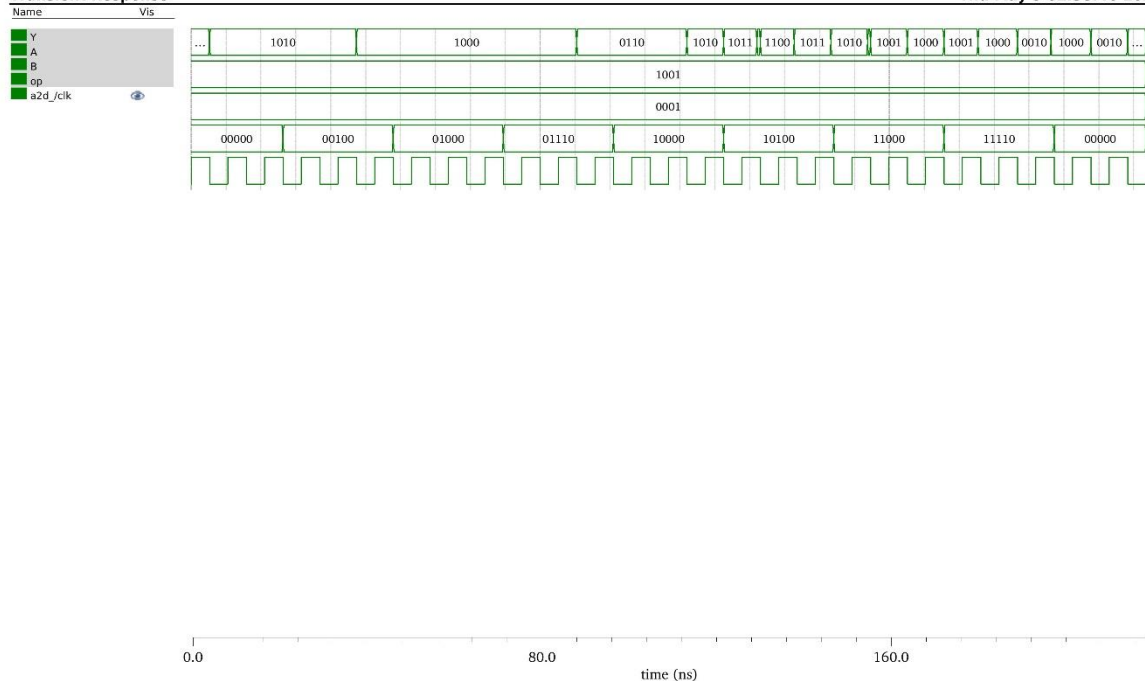


Figure 47: Final circuit functioning for VDD=0.7V

Transient Response

Thu May 9 02:38:46 2024

Figure 48: Final circuit functioning for $V_{DD}=1.2V$

A and **B** are the signals at the input before the registers, **op** is the opcode before the register and **Y** is the output of the circuit at the output register.

The graphs may seem cutoff but that's a result of the way they're extracted from the software. The circuits are functional, and it should be noted that for $V_{DD} = 1.2V$ the clock period is $8.4n$ and for $V_{DD} = 0.7V$ the clock period is $39.3n$ as a result of the differing minimal periods.

The circuits function correctly as it takes 2 clock cycles (2 rising edges) to change output according to opcode for the ALU (without any change in feedback bit) and 3 clock cycles (3 rising edges) to change to and from feedback. This is how we designed our circuit to perform and it works exactly this way. In addition, the outputs subject to the delay after changing operation are in accordance with the opcode.

Steps taken to optimize circuit performance

We make use of **data gating or input gating**. We AND the inputs to all the blocks with an enable bit. The enable bit is 0 if the block is inactive and 1 if the block is active. As a result, whenever a block isn't in use its output is set to zero. This conserves dynamic power. It increased the area of layout as we had to AND the inputs to every block with a respective select bit but in the long run it proved useful to decreasing power consumption.

In addition to the above, we **made use of 4 buffers to decrease the fan-out** of certain signals from 8 to 4. From the course, we know that the optimal fan-out is $3.6 \sim 4$ thus it was imperative to use these buffers to reduce the fan-out and further reduce the delay of the overall circuit.

Conclusions/Remarks

One of the issues we came across in the latter stages was the spacing between white boxes along the VDD and VSS line. The DRC rule is a minimum distance of 0.06um between white boxes pictured below. This wasn't clear from the start and had we known prior, we'd have approached the initial design of the blocks a bit differently so as to allow us to minimize the area even further by spacing the blocks in the placement rows such that the spacing between the white boxes is always a constant.

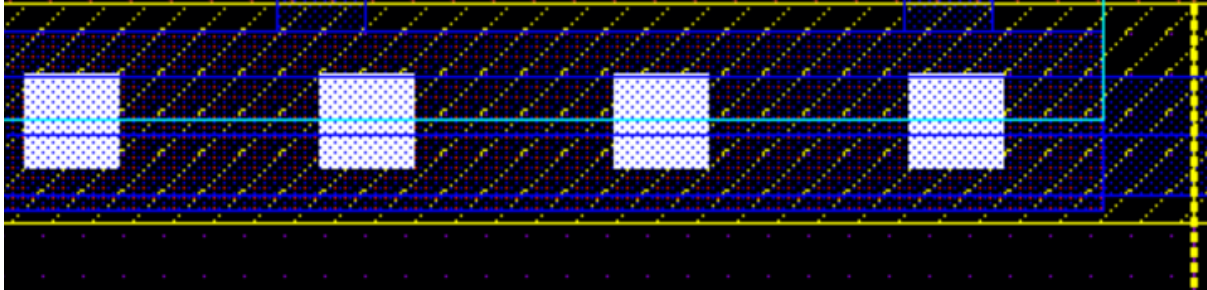


Figure 49: white block spacing that needs to be maintained