EEE 414 Project

16-Bit Carry Bypass Adder

Yağız Mart

22003915

Standard Cell Library

The Beta ratio of the NMOS and PMOS transistors is selected as 4.

Inverter Small (INV): The INV includes the smallest size PMOS and NMOS transistors (PMOS is sized according to beta ratio).

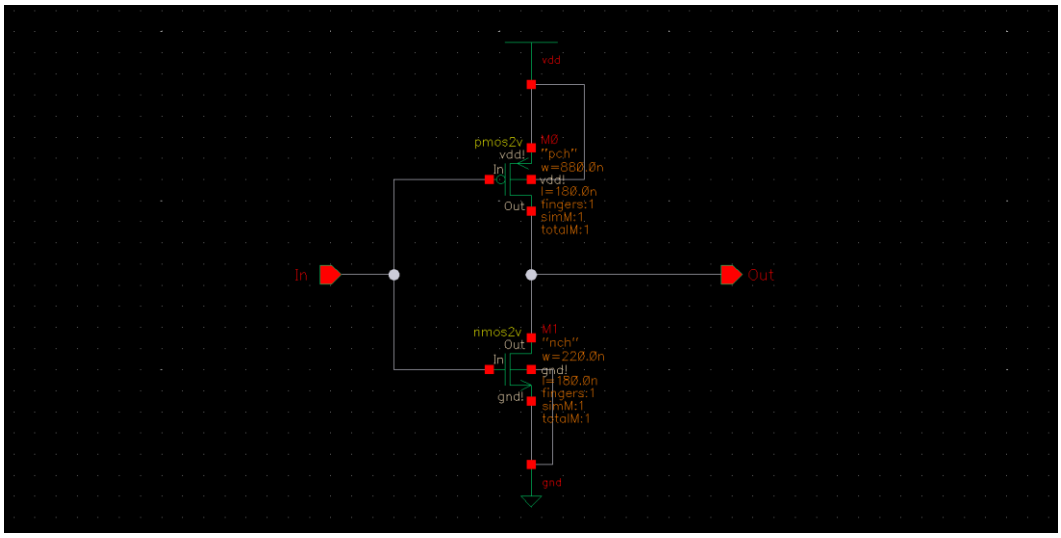

Figure 1:Schematic of INV. Image is taken by “Export Image” of Virtuoso.

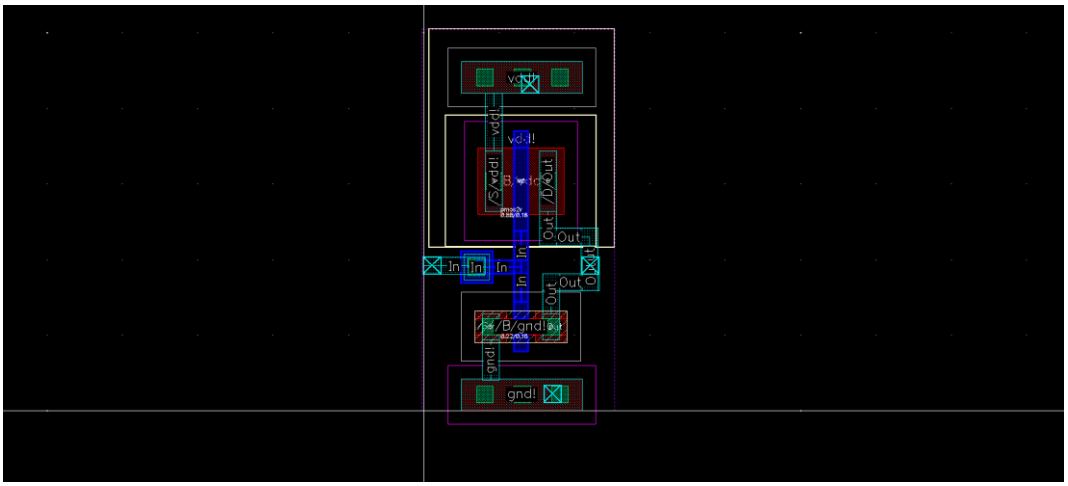

Figure 2: Layout of INV. Image is taken by “Export Image” of Virtuoso

Inverter (INVBIG): This inverter is designed to fasten the circuit and buffering purposes. The sizes of the transistors are two times of the sizes of the transistors of INV.

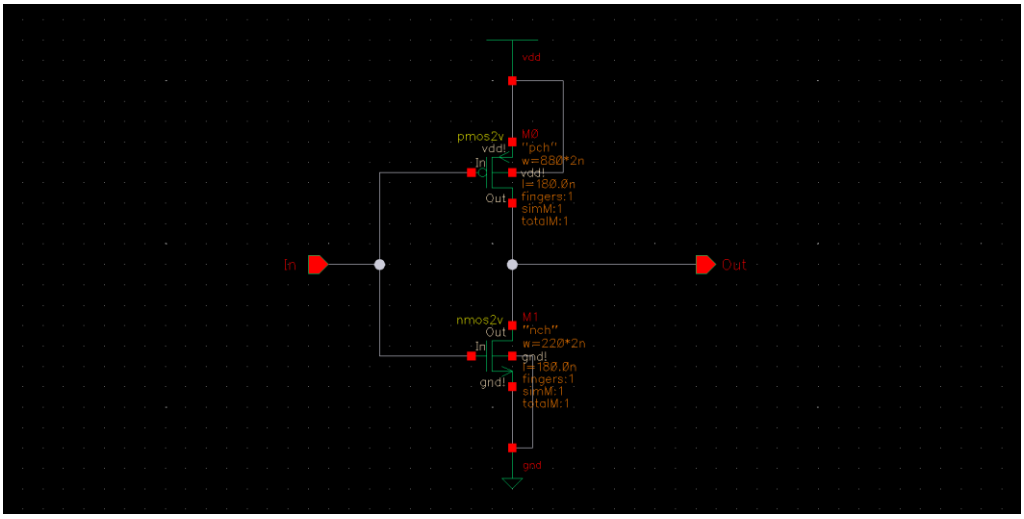

Figure 3: Schematic of INVBIG. Image is taken by “Export Image” of Virtuoso

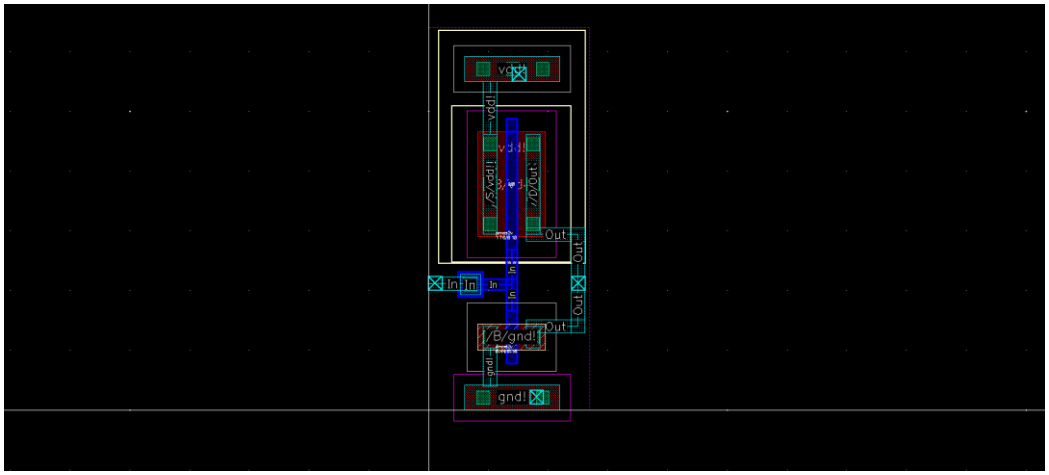

Figure 4: Layout of INVBIG. Image is taken by “Export Image” of Virtuoso

NAND: Two input NAND that is used for generating input of the bypass section. The transistor sizes are selected as the gate has same pullup and pulldown resistance with INV.

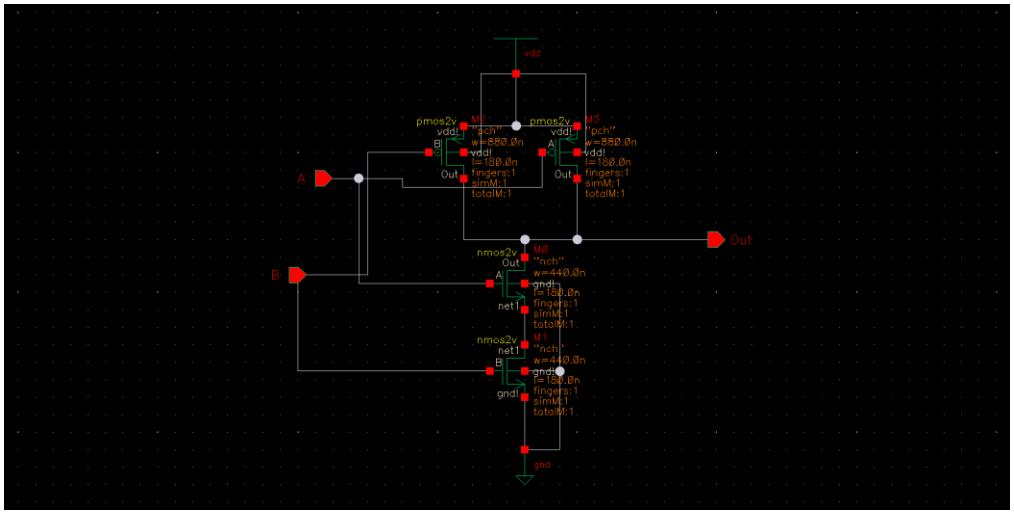

Figure 5:Schematic of NAND. Image is taken by “Export Image” of Virtuoso

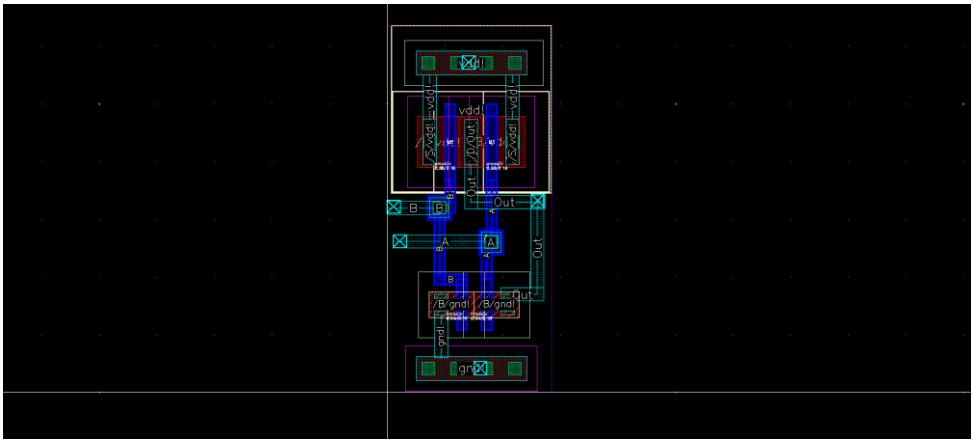

Figure 6: Layout of NAND. Image is taken by "Export Image" of Virtuoso

NOR: Two input NOR that is used for generating input of the bypass section of 4-bit sections. At first, the transistor sizes are selected as the gate has same pullup and pulldown resistance with INV. However, because it does not make much difference in the delay PMOS sizes are halved.

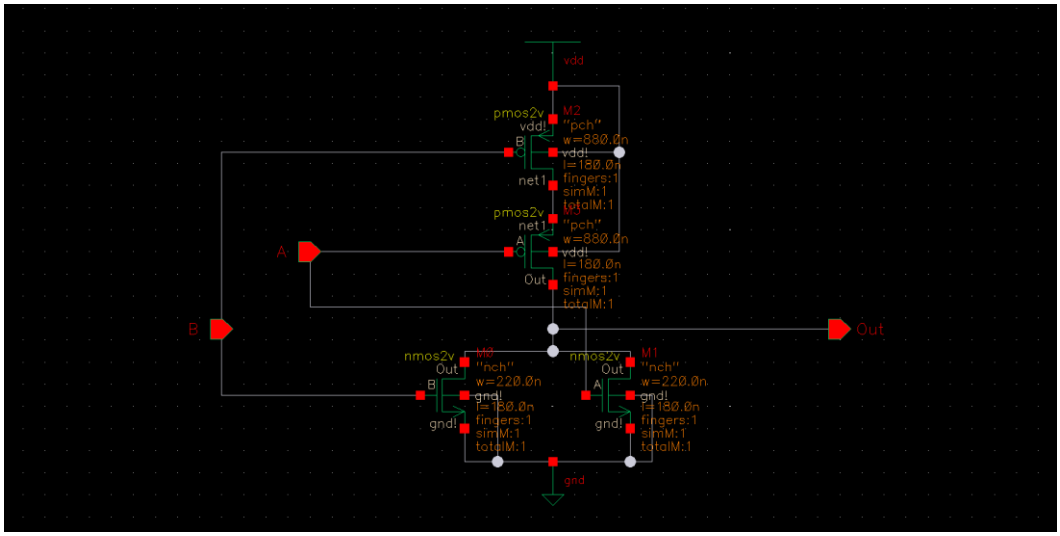

Figure 7:Schematic of NOR. Image is taken by “Export Image” of Virtuoso

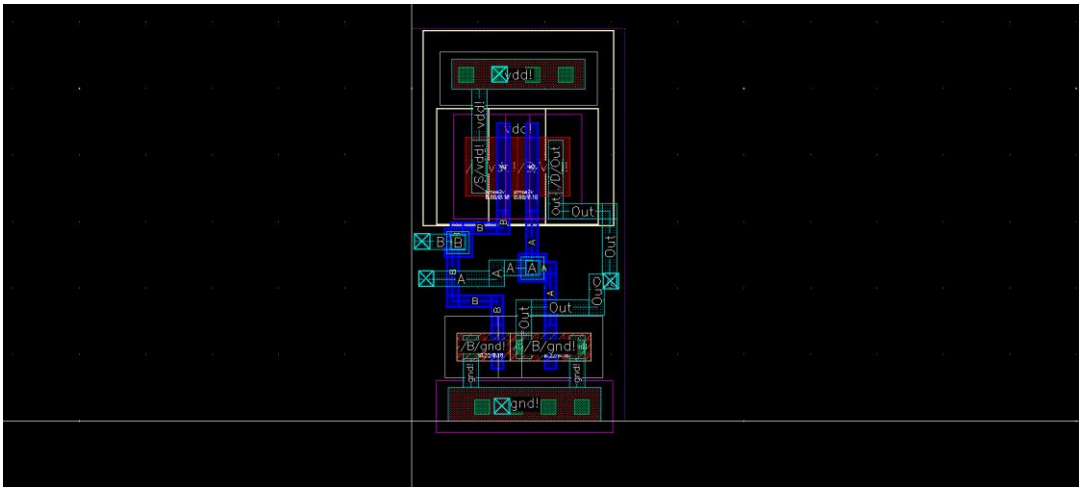

Figure 8: Layout of NOR. Image is taken by “Export Image” of Virtuoso

Multiplexer (TransMux): Two input multiplexer that is built with transmission gates. Used in carry selection and as a XOR gate for the generation of propagate. The transmission gates are used instead

of mirror mux because it has lower propagation delay. PMOS and NMOS transistor sizes are the minimum. This is for a smaller area in the layout and less propagation delay.

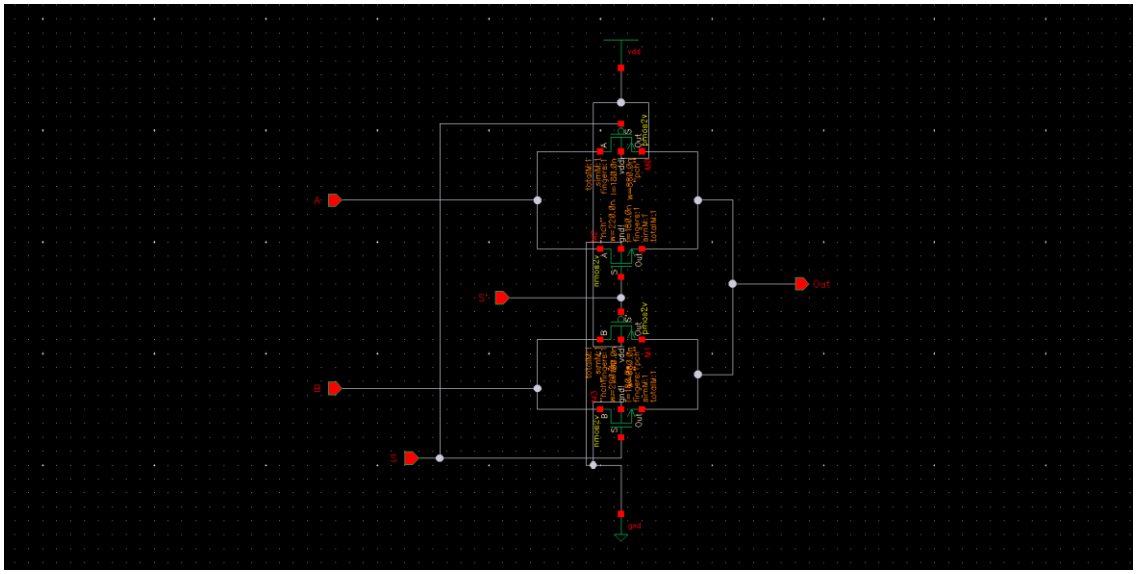

Figure 9: Schematic of TransMux. Image is taken by “Export Image” of Virtuoso

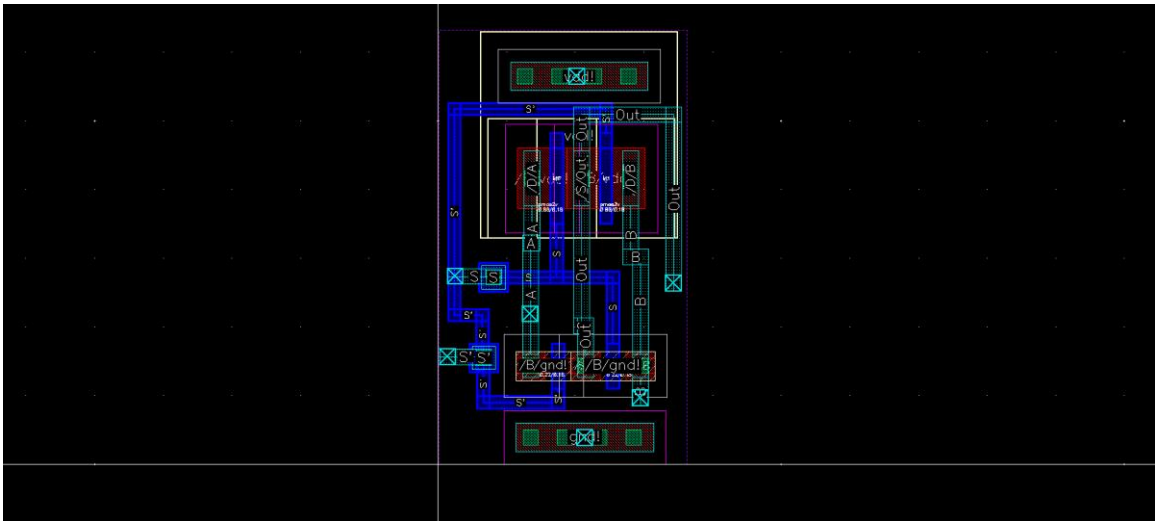

Figure 10: Layout of TransMux. Image is taken by “Export Image” of Virtuoso

Table 1: Transistors’ length and widths. All sizes are in micrometers.

Width/Length	INV	INVBIG	NAND	NOR	TransMUX
NMOS	0.22/0.18	0.44/0.18	0.44/0.18	0.22/0.18	0.22/0.18
PMOS	0.88/0.18	1.76/0.18	0.88/0.18	0.88/0.18	0.88/0.18

Table 2: Low to high, high to low and propagation delay tables with different load capacitors for all the cells. The delays are found with horizontal marker in Virtuoso Ade L.

Capacitor	3fF			24fF			192fF		
Delay	PLH	PHL	P	PLH	PHL	P	PLH	PHL	P
INV	42.46	42	42.23	134.2	171	152.6	869.9	1138	1004
INVBIG	32.53	43.41	37.97	78.49	117.3	97.895	435	702	568.5
NAND	47.69	51.3	49.495	141.7	163.5	152.6	886.8	1053	969.9
NOR	67.01	52.29	59.65	226.35	173.3	199.825	1667	1130	1398.5
TransMux	31.12	33.03	32.075	93.37	108	100.685	631.66	753.49	692.575

The delays are increasing with the increase of the extrinsic load capacitors. This is expected because it is harder to drive a bigger extrinsic load capacitor. Having a smaller load cap can be thought as having a small inverter at the output. It is easier for a gate to drive a small inverter. However, bigger load capacitance at the output can be thought as a more complex gate with maybe bigger transistors. It is harder to drive bigger and complex structures. Therefore, the propagation delay will be higher as the extrinsic load capacitance increases.

The INVBIG is faster than INV and other gates. This is because it has a higher W/L ratio which leads to higher current. Because the driver is stronger, it can drive the output faster. This effect can be observed for all the load capacitance values. The least difference is at 3fF. This is because at lower extrinsic capacitance the intrinsic capacitance has more effect on the delay. As we increase the extrinsic capacitance, it begins to dominate more and more which leads to higher difference between INVBIG and other gates in comparison of propagation delays. NAND has more intrinsic input capacitance than INV (has more gates). This leads to more parasitic delay and more logical effort. However, its effective fan out is less (external load/input cap). At 3fF the dominance of logical effort and parasitic delay is higher, so the delay of the INV is less than NAND. Yet, as external load is increasing, effective fan out begins to dominate the gate delay. Therefore, as we increase the load capacitance the delay of NAND becomes less than INV. The propagation delay of NOR is higher than other gates because I used small width for PMOS transistors which decreases W/L ratio. This leads to less current and lower speed. This can be observed from the PLH of NOR. Because PMOS sizes are smaller than it should be to be the same with INV resistance, the pullup resistance is more. This leads to higher low to high propagation delay which increases overall propagation delay. Moreover, the NOR is always slower than NAND in circuits that has P-type body. This is because the speed of PMOS carriers are less than NMOS carriers. PMOS are series in NOR gate while parallel in NAND. Thus, NAND is faster.

Multiplexer with transmission gate is a different case from all other cells. It is designed with transmission gates to provide high speed to the carry bypass structures. Transmission gate structures provide more speed while having more area in the layout. This is a tradeoff. We can see that its propagation delay is less than other gates.

Full Adder

The full adder that is designed is a mirror adder. This design lowers the number of transistors that is used. It is also a more compact design which is easier to draw the layout. Mostly Weste/Harris asymmetric sizing approach is followed in the carry propagate part. The generate and delete parts are also asymmetric. Carry is the critical signal in the adder design. Therefore, it should be put to the inner part (closer to the output) to make it faster. When carry is in inner parts of the circuit, the internal node capacitors that should be charged is less for the carry. This means that it becomes faster. In addition to that, the asymmetric architecture provides more speed to the carry. The asymmetric structure further

favors the critical input which is carry in because the logical effort and parasitic delay of the carry is decreasing.

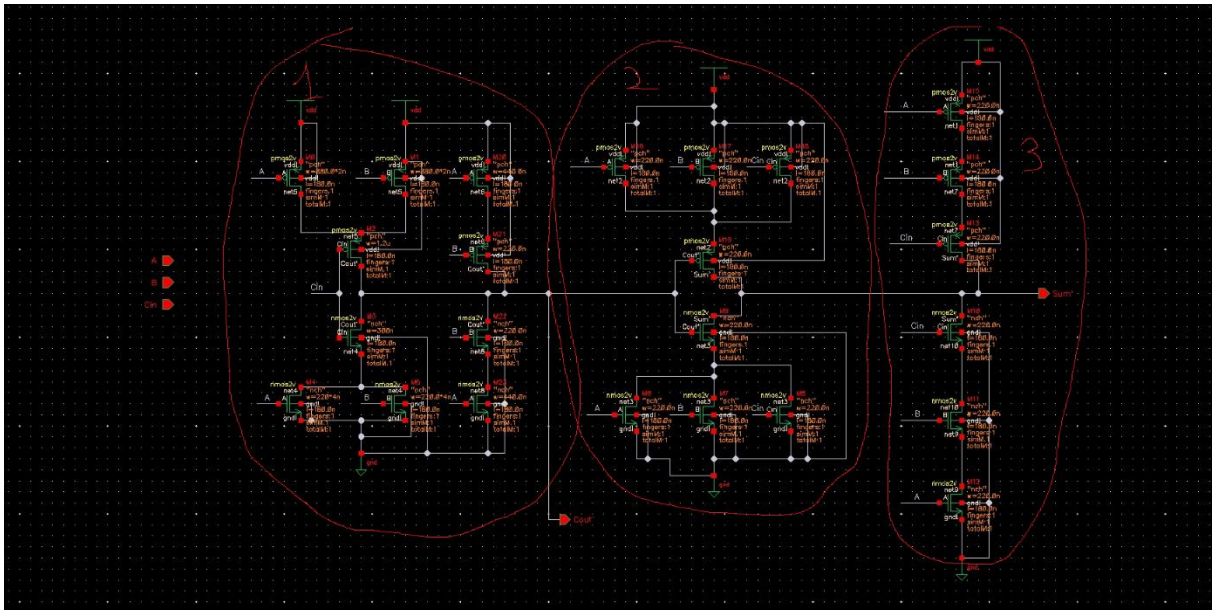

Figure 11: Schematic of Full Adder. Image is taken by “Export Image” of Virtuoso

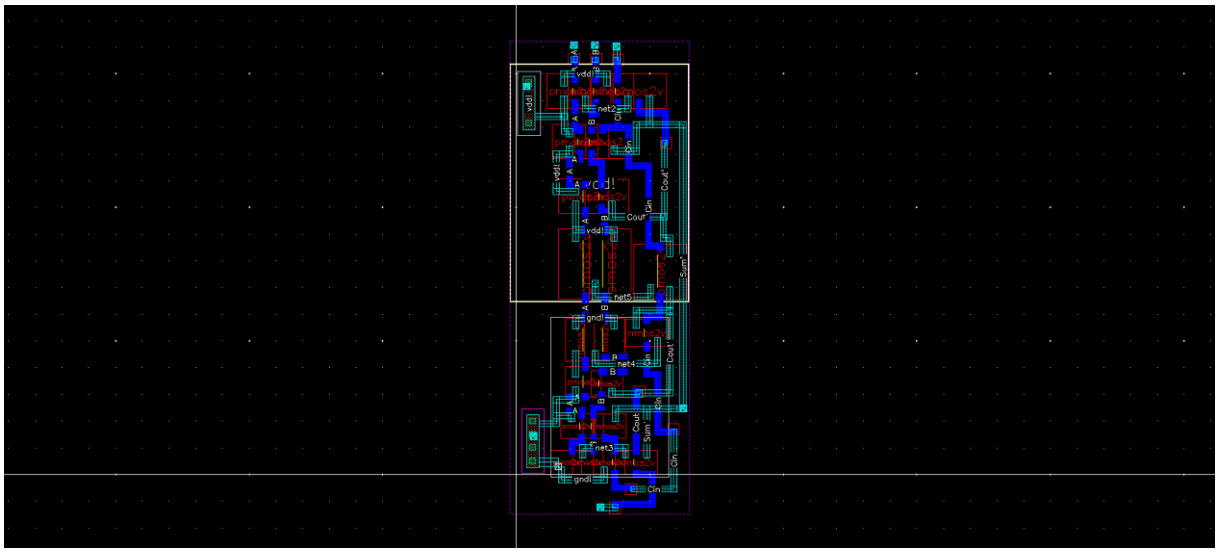

Figure 12: Layout of Full Adder. Image is taken by “Export Image” of Virtuoso.

All the lengths are 0.18 micrometer. The transistors in the second and third sections of the design (sum section) are all minimum size. Their widths are 0.22 micrometer both for PMOS and NMOS transistors. The first section (section related to carry out) is a bit different. In propagate part, two parallel PMOS transistors have 1.76 micrometer width. The below one is 1.2 micrometer. The parallel NMOS transistors are 0.88 micrometer in width. The one that is above them is 0.3micrometer. The transistors that are next to them and connected in series have 0.44 and 0.22 micrometer widths. The inner ones are the 0.22.

There are three important signals for carry output which are propagate, generate and delete. Generate and delete directly generates or deletes carry out without looking at input carry. Propagate is working with carry in. If propagate is 1, it puts the carry in into carry out. Either 1 or 0. The sum is dependent to two sections. If carry out is 0 and either X, Y or carry in is 1 then Sum will be 1. The other case that makes sum 1 is the one that all of the inputs are 1. The mirror adder is working in an inverted way. It

gives inverted carry out and sum. Therefore, I enter inverted inputs to the even (2, 4) full adder blocks in the 4-bit or 2-bit CBA blocks to get the correct values.

Table 3: Truth Table of Mirror Adder

X	Y	Cin	Sum	Cout	Carry Status
0	0	0	0	0	Delete
0	0	1	1	0	Delete
0	1	0	1	0	Propagate
0	1	1	0	1	Propagate
1	0	0	1	0	Propagate
1	0	1	0	1	Propagate
1	1	0	0	1	Generate
1	1	1	1	1	Generate

16-Bit Carry-bypass Adder

The 16-bit Carry-bypass adder is very similar to a ripple carry adder circuit. Only difference is that it checks whether the inputs produce a propagate signals or not. If there is a propagate production, the carry is passed to the next section without waiting for all the full adders of this block. This makes the addition process faster than ripple carry adder in most of the input combinations. However, it takes more area because we use more circuit element. In the design, there are bypass stages. The stages in this design are composed of 2-bit and 4-bit blocks. In 2-bit stage there are 2 full adders and in 4-bit stage there are 4 full adders. The other cells are also allocated according to that. The critical path of CBA is starts from the first block, goes through bypasses at the middle blocks and end with the last block. The critical output is Z16 which is the last sum bit. Decreasing the size of first and last blocks is shortening the critical path. The first and last blocks are 2-bit and the middle blocks are 4-bit sized. The multiplexers are used as XOR to produce propagate signals for each bit. Then, this propagate signals are put into and AND gate that is built by two NAND and one NOR in 4-bit block and one NAND and one INV in 2-bit block. If all the propagates of the stage is 1, the carry is directly propagated from the multiplexer to the next stage, so the circuit does not wait carry from full adders. There are 2 inverters at the end of multiplexer and even full adders. This is for speed and signal quality purposes. As I said before, the mirror adder outputs inverted sum, therefore the odd full adders are used by and output inverter. There are two modules that are 2-bit stage and 4-bit stage. These are connected in the top module to create a 16-bit Carry-bypass Adder. Two inverters are added to the output Z17. This was actually unnecessary but layouts were ready when I realize them, so I did not change it.

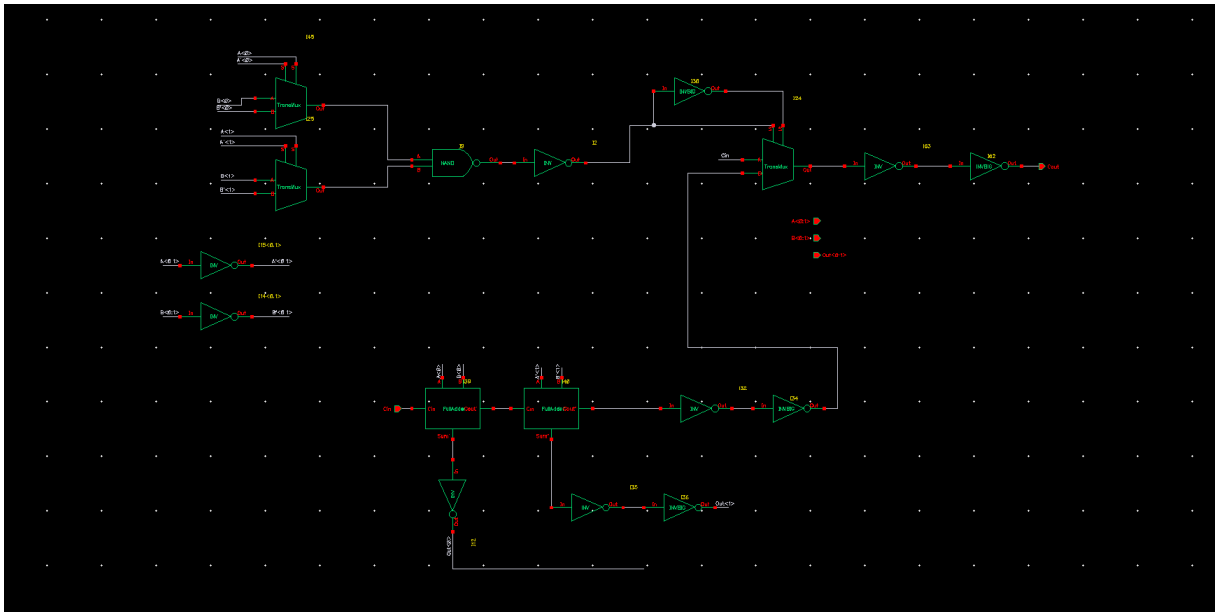

Figure 13: Schematic of 2-bit block of Carry-bypass adder. Image is taken by “Export Image” of Virtuoso.

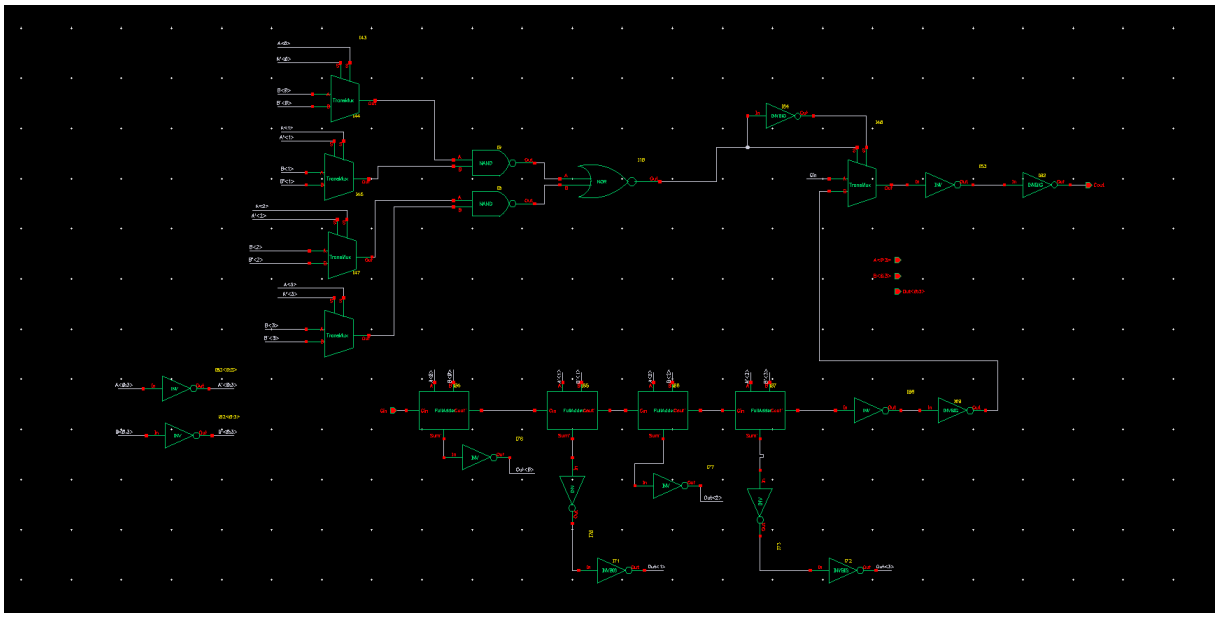

Figure 14:Schematic of 4-bit block of Carry-bypass adder. Image is taken by “Export Image” of Virtuoso.

Inputs are X1:16 and Y1:16. Outputs are Z1:17.

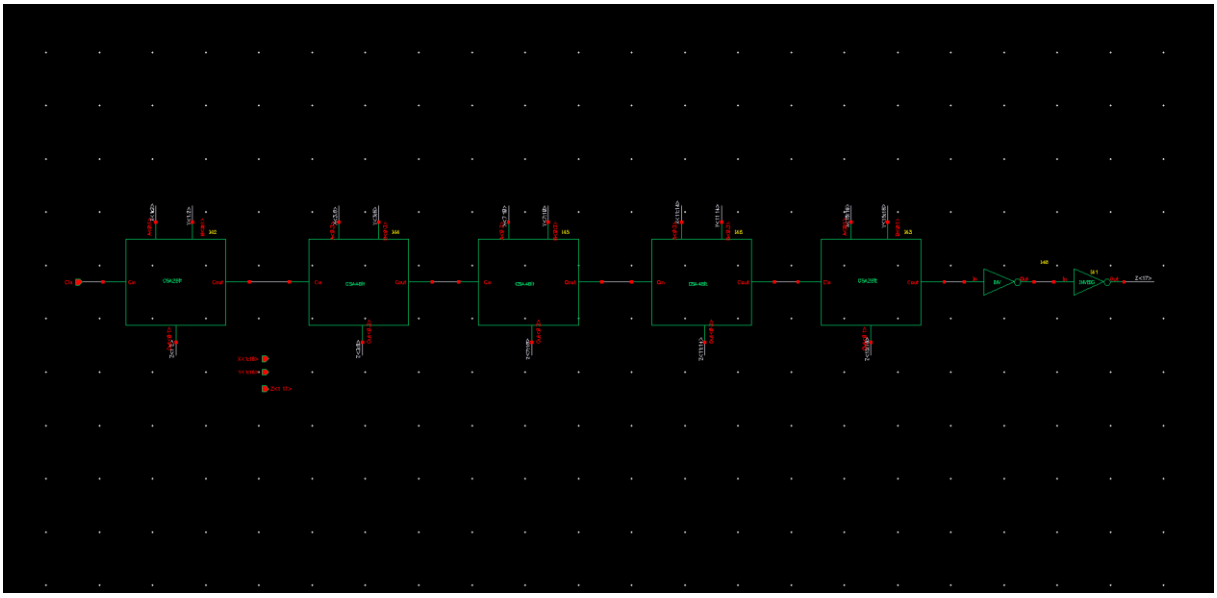

Figure 15: Schematic of 16-bit Carry-bypass Adder. Image is taken by “Export Image” of Virtuoso.

The critical path is the first carry (X1 Y1) should be generated and all the carries of the middle stages should be propagated. The last sum of the last stage (Z16) is the slowest output. For example, this condition can be got by entering

Y = 0111 1111 1111 1111

X = 0000 0000 0000 0001

We need to change the least significant bit from 0 to 1. There are 2^16 different inputs for critical path. We can put 0,0 to the LSB and can make 1,0 other 14 bits and 1,1 or 0,0 the last bit. This is the critical path because if there is a carry that is generated or deleted at the middle stages, the carry will go on from that point and this will shorten the path.

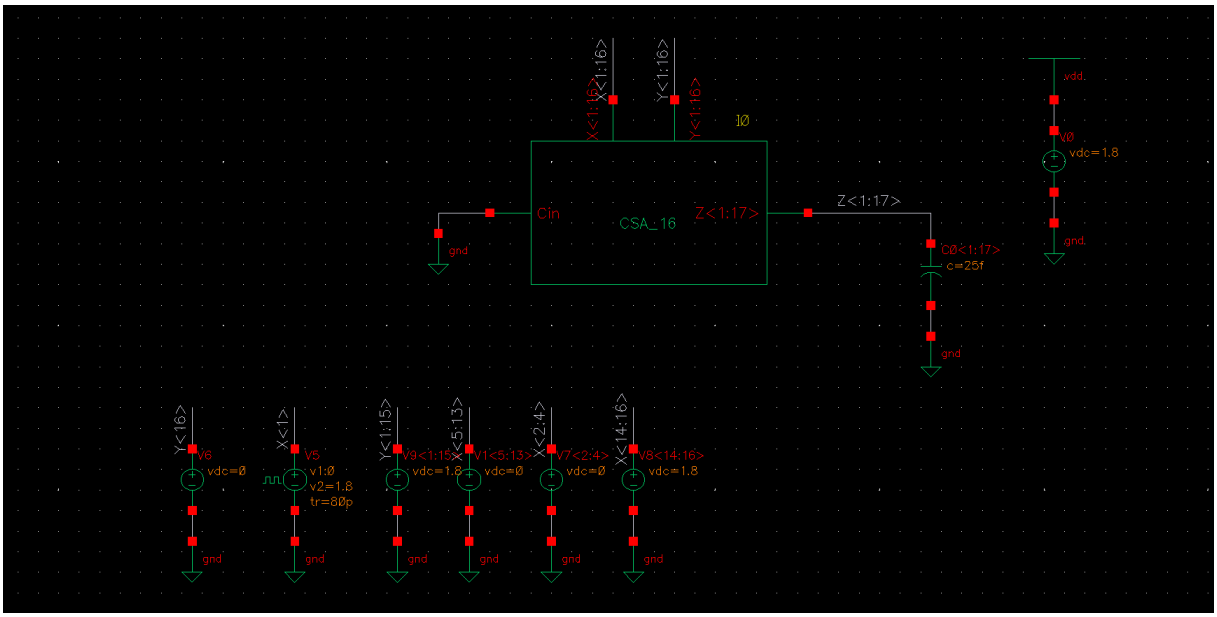

Figure 16: Schematic of 16-bit Carry-bypass Adder Test. Image is taken by “Export Image” of Virtuoso.

Inputs for the critical path delay are:

Y = 0111 1111 1111 1111

X = 0000 0000 0000 0001

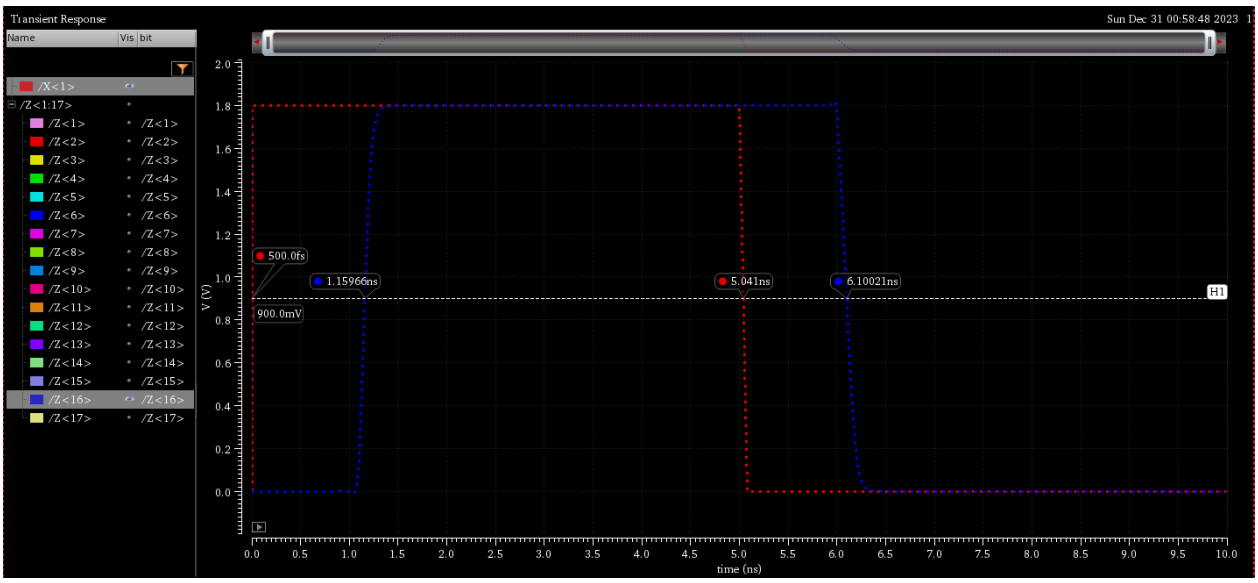

Figure 17: The delay of the schematic. Image is taken by screenshot.

The delay of the Z16 at the schematic is 1.597ns.

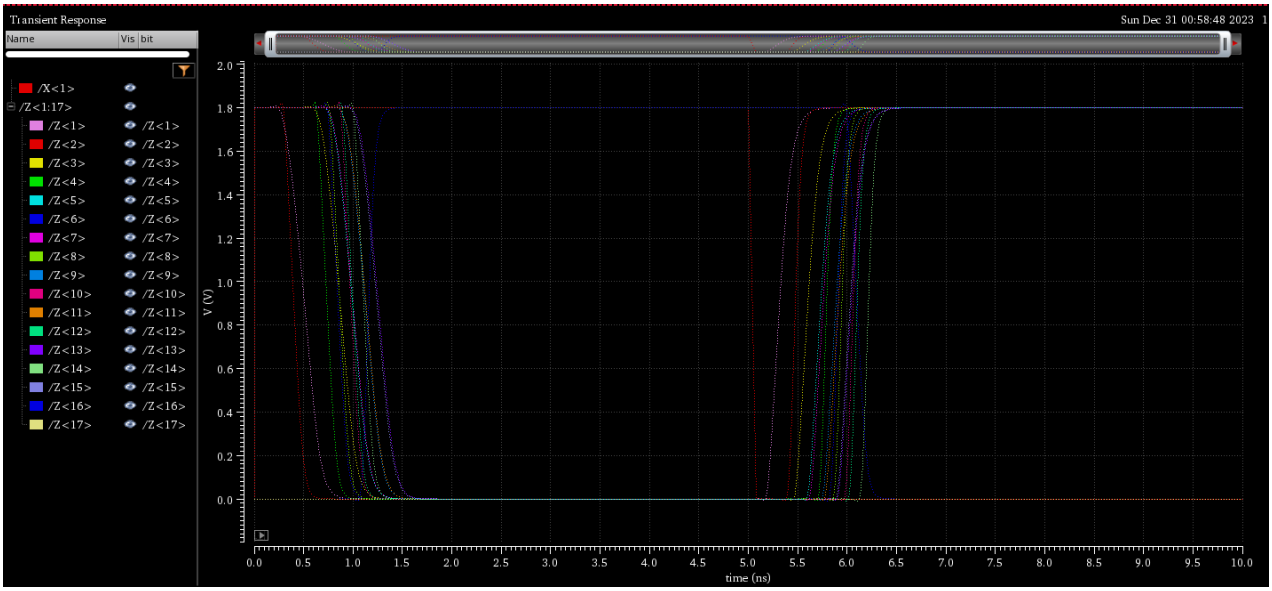

Figure 18: The outputs. Image is taken by screenshot.

16-bit Carry-bypass Adder Layout Area, Delay, and Quality

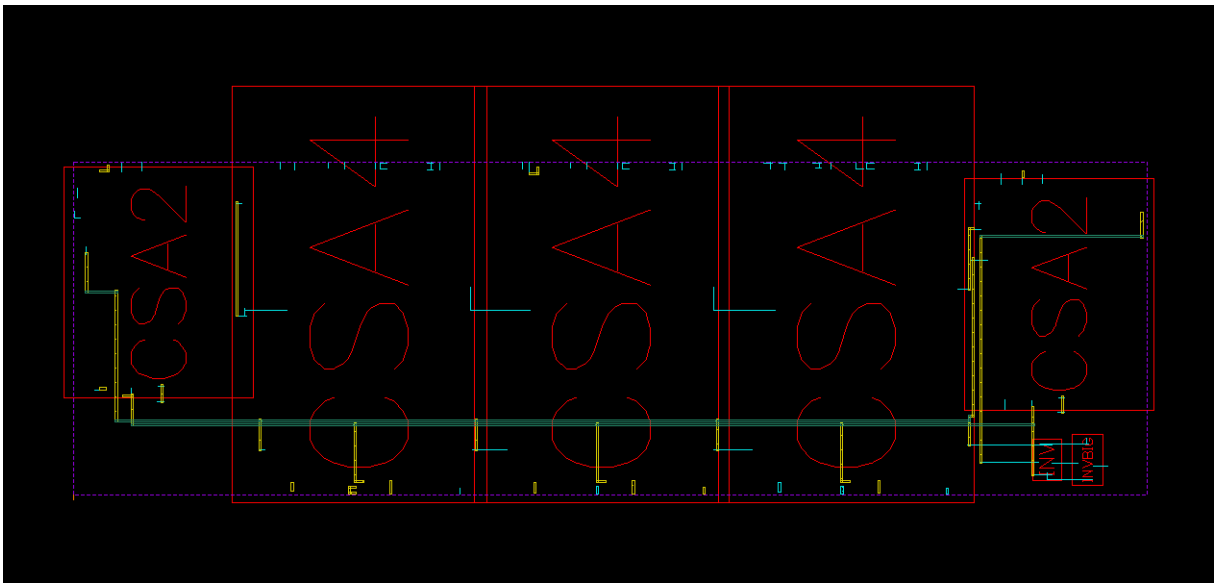

Figure 19: Layout of 16-bit Carry-bypass Adder. Image is taken by “Export Image” of Virtuoso.

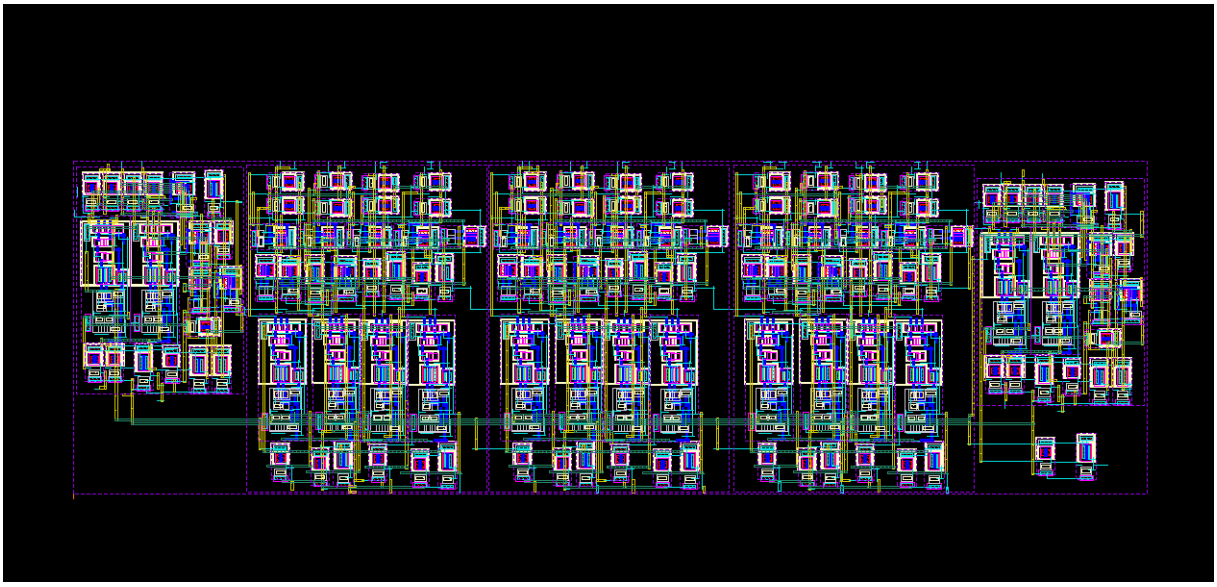

Figure 20: Layout of 16-bit Carry-bypass Adder. Image is taken by “Export Image” of Virtuoso.

Area is 6919.535 micrometer.

Inputs for the critical path delay are:

Y = 0111 1111 1111 1111

X = 0000 0000 0000 0001

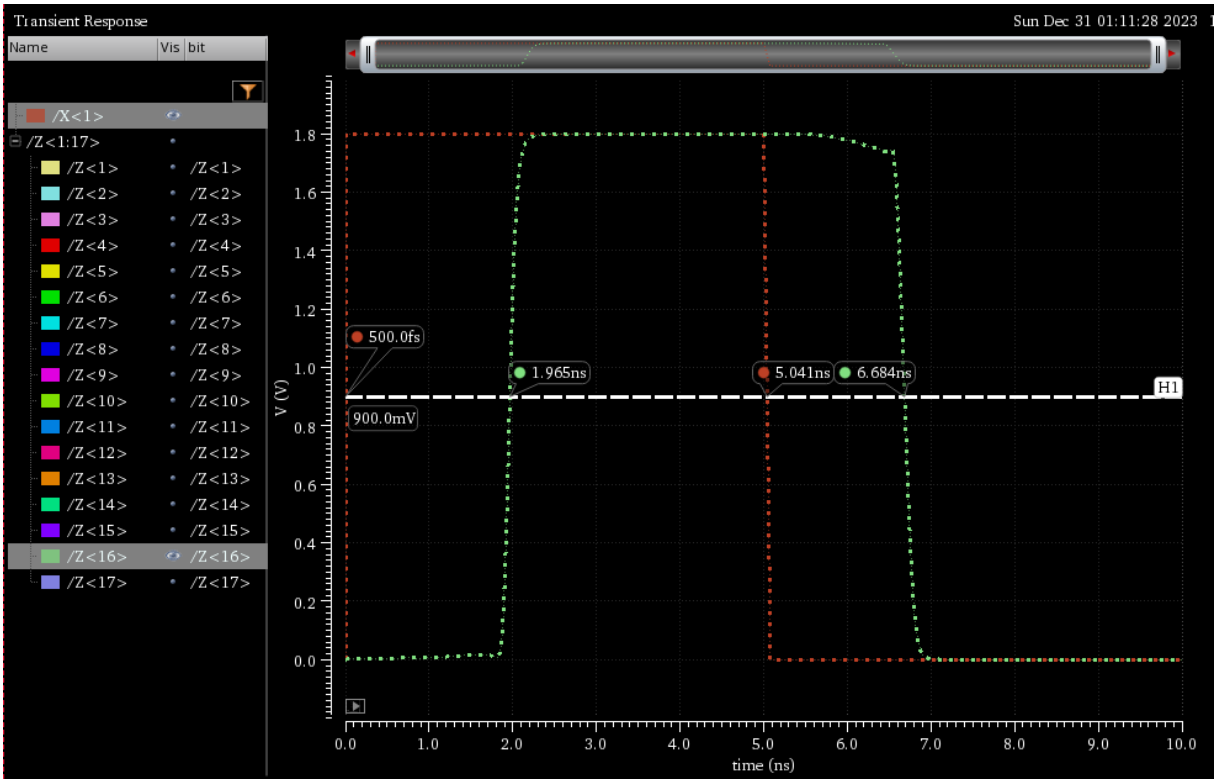

Figure 21: The critical path delay RC extracted. Image is taken by screenshot.

Table 4: R, C and RC extracted delays of 16-bit CBA

	R extracted	C extracted	RC extracted
Delay	1.717ns	1.9202ns	1.965ns

Comparison: All delays of RC, C and R extracted simulations are longer than the schematic one. This was expected. In the schematic, extrinsic wire capacitance and resistance are not considered while calculating the delay. However, when we draw a layout there are capacitances between metal wires, metal wires and polysilicon etc. There are also resistances of these wires. These capacitances and resistances make the circuit slower. The placement of the cells, the drawing of the wires, the placement of the vias, everything effects the delay of the circuit. Layout drawing is an important process. A bad designer can draw a layout of schematic that has much more delay than a layout of a good designer. The layout is the real circuit of the chip. Hence, the extraction of resistance and capacitance are important to measure the delay of the circuit correctly.

Quality = 1/(6919.535*10^-6*1.965*10^-9) = 73.546250205*10^15