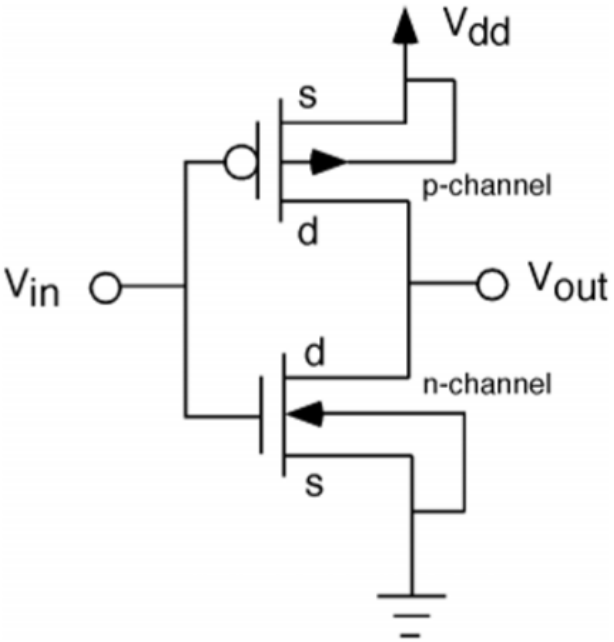
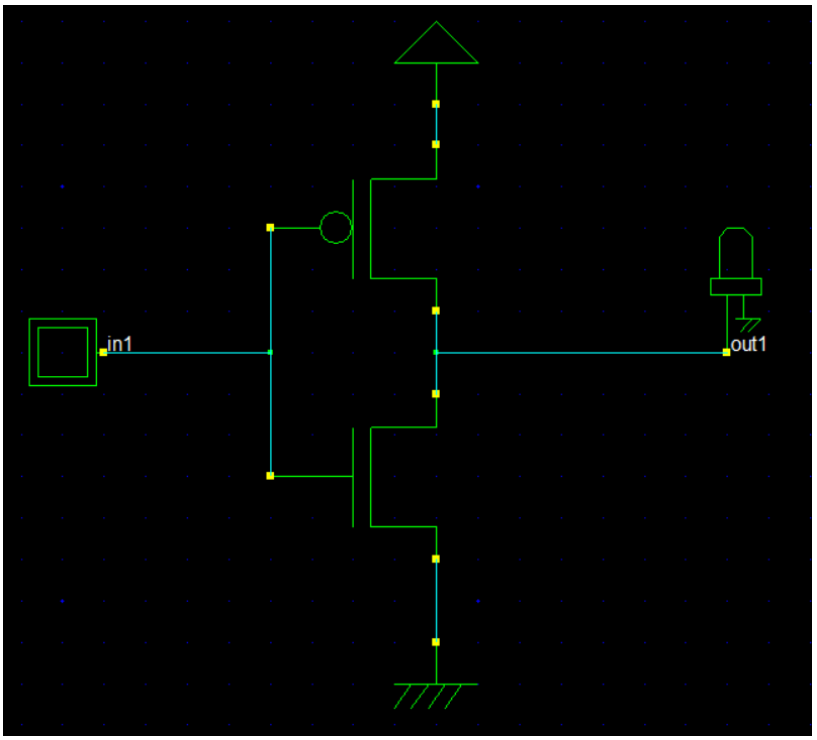
Verna Dorian-Alexandru 341C1
Referat VLSI

1. Inversor
    a. Schema circuitului (CMOS/PMOS)
    Am preluat schema circuitului de la adresa:
    https://www.tutorialspoint.com/vlsi_design/vlsi_design_mos_inverter.htm

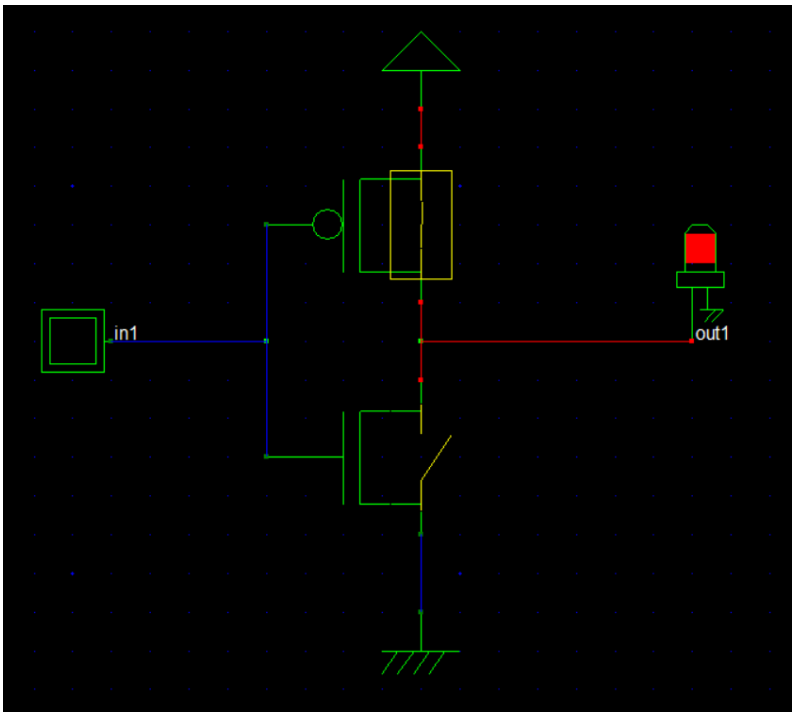


    b. Implementarea în DSCH

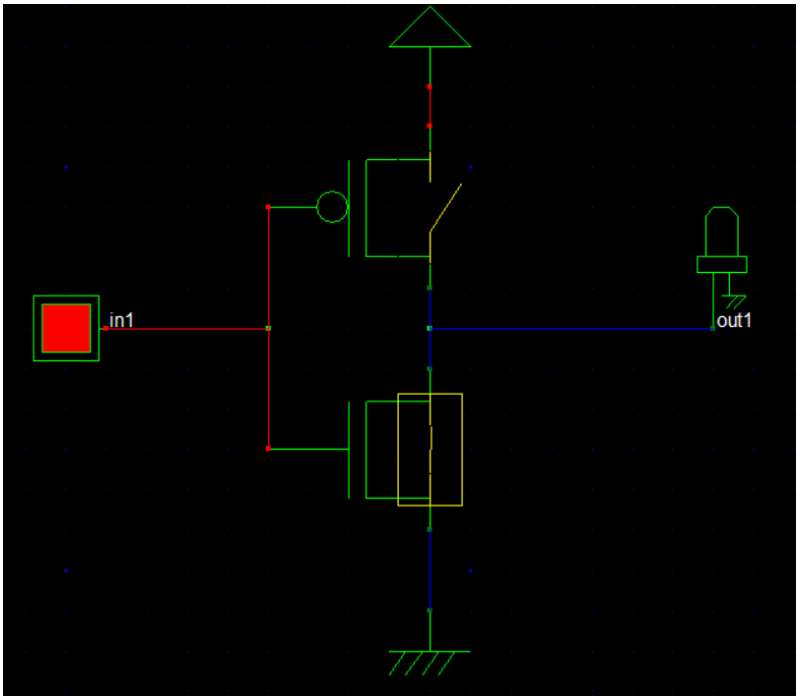


c. Simularea în DSCH

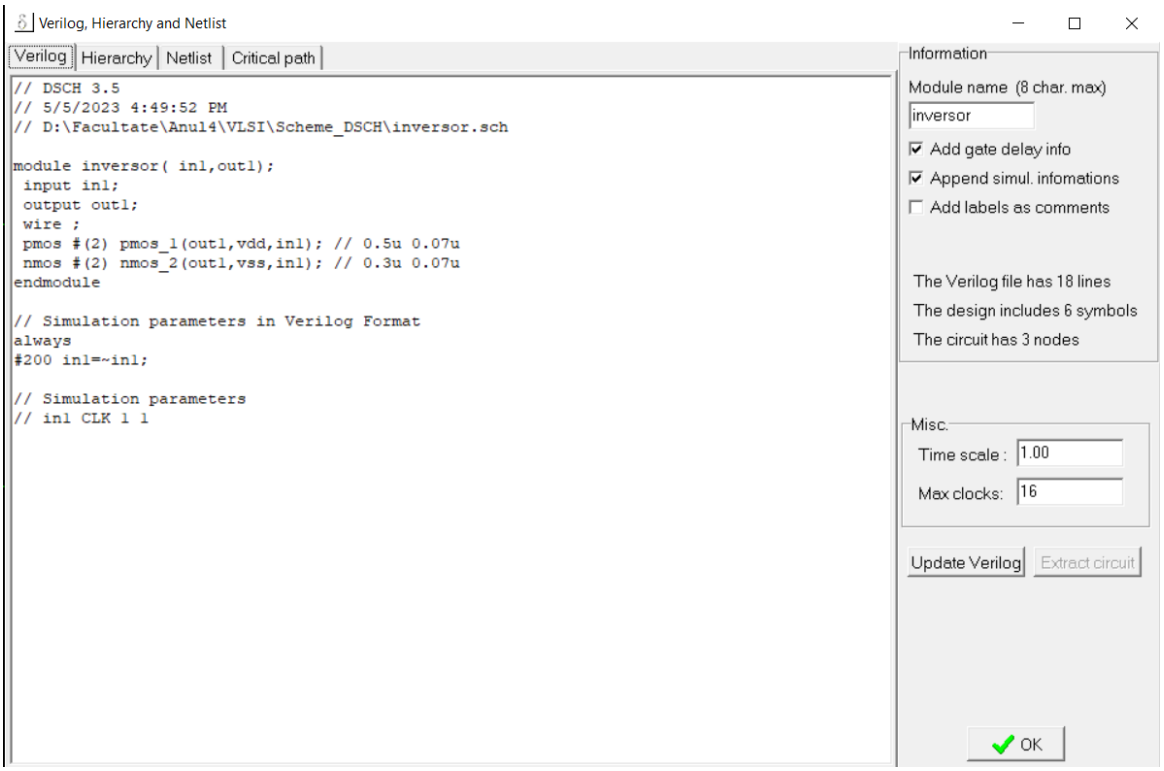
i. Inputs: in1 - 0, Outputs: out1 - 1



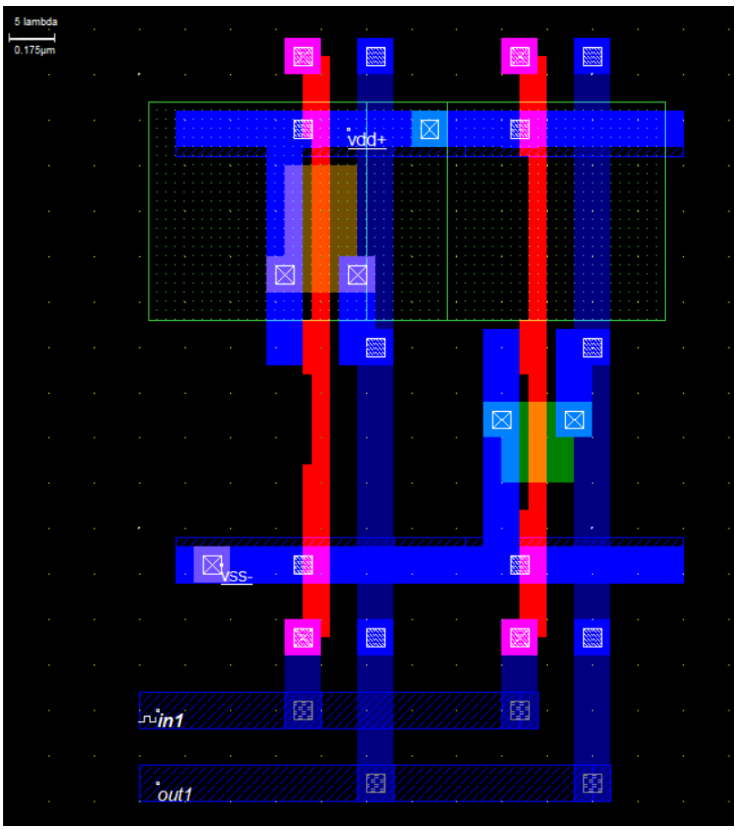
ii. Inputs: in1 - 1, Outputs: out1 - 0



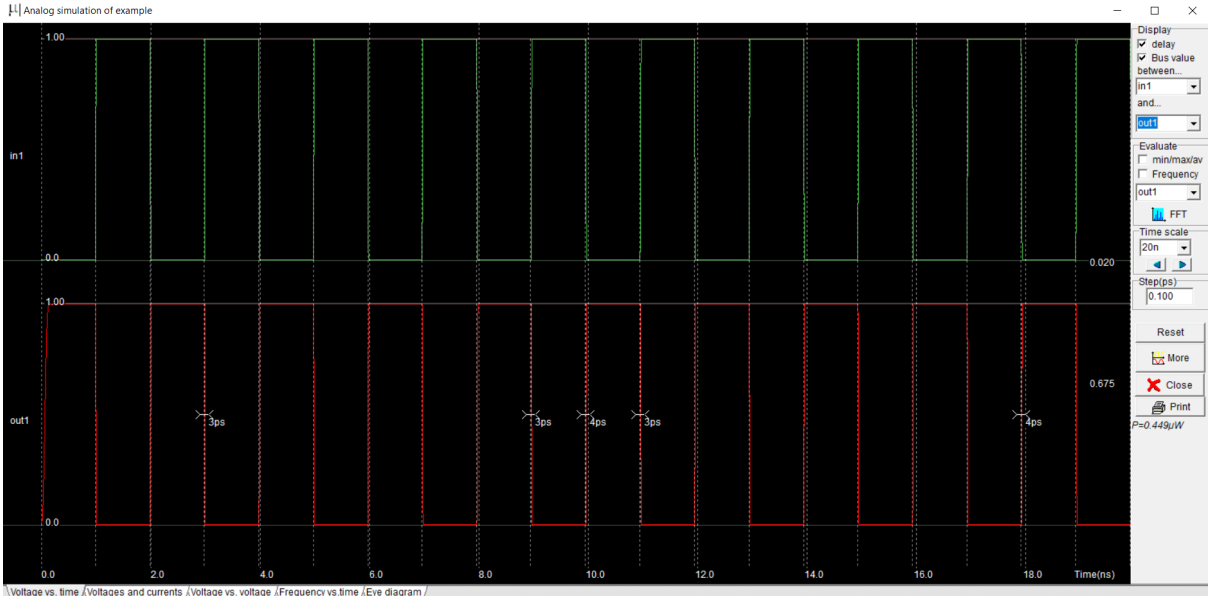
d. codul Verilog obținut



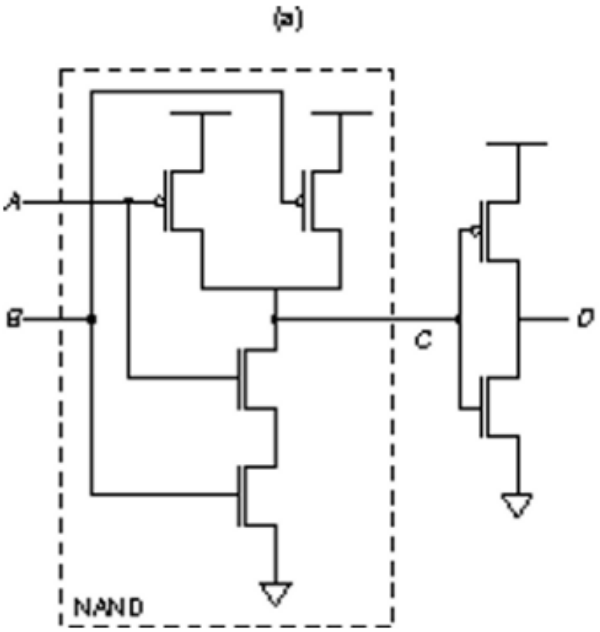
e. Implementarea în Microwind



f. Simularea obținută în Microwind



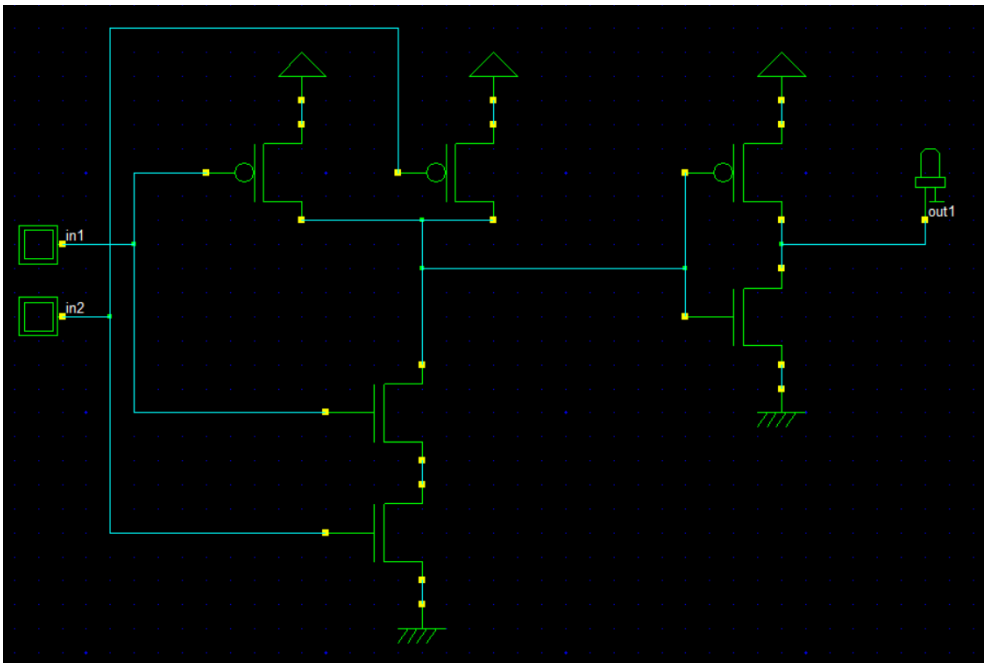
2. AND
   a. Schema circuitului (CMOS/PMOS)
   Am preluat schema circuitului de la adresa:
   https://electronics.stackexchange.com/questions/226023/cmos-and-gate-implementation



(b)

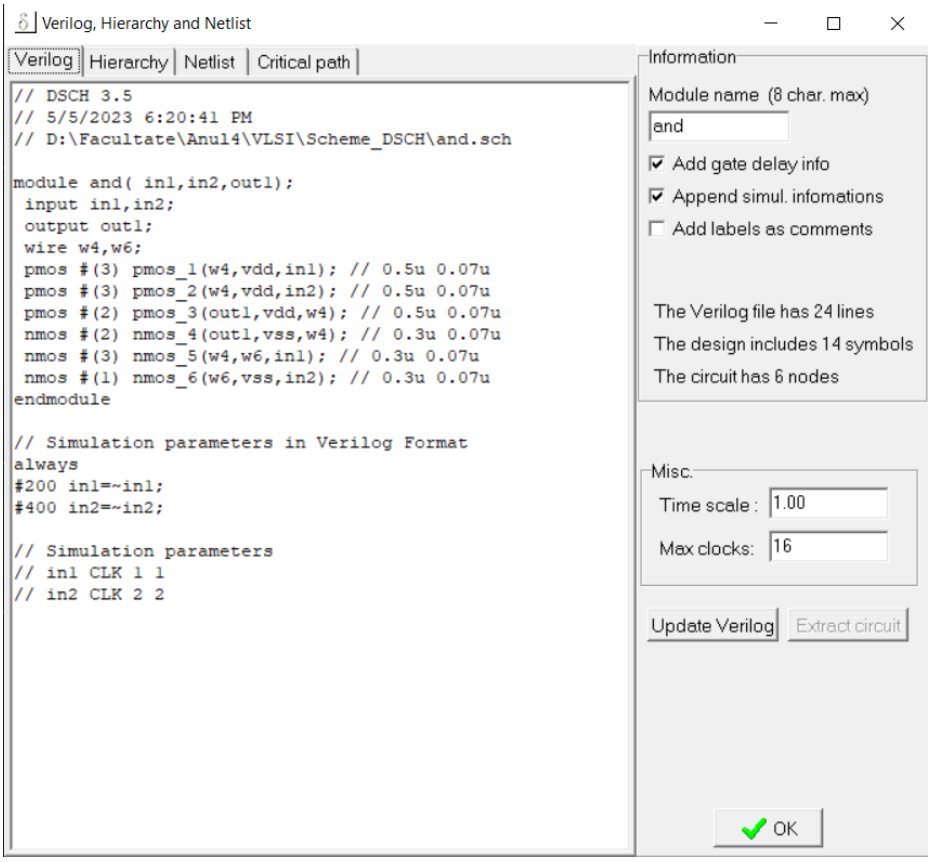
A	B	C	D
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

b. Implementarea în DSCH

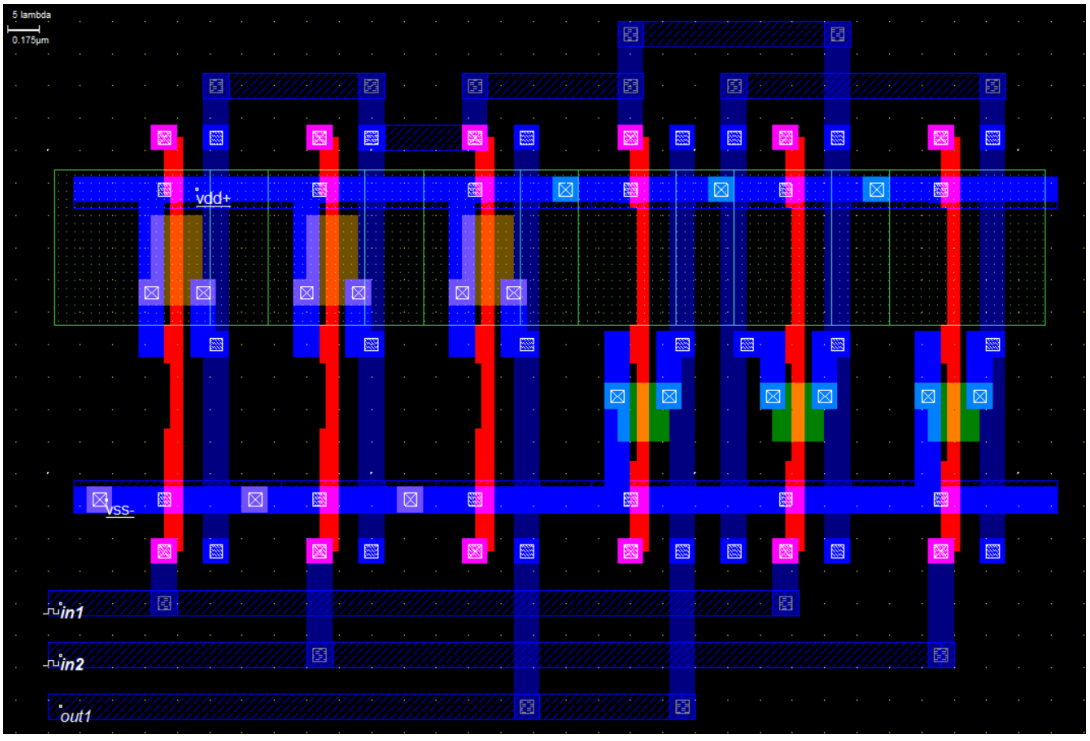


c. Simularea în DSCH

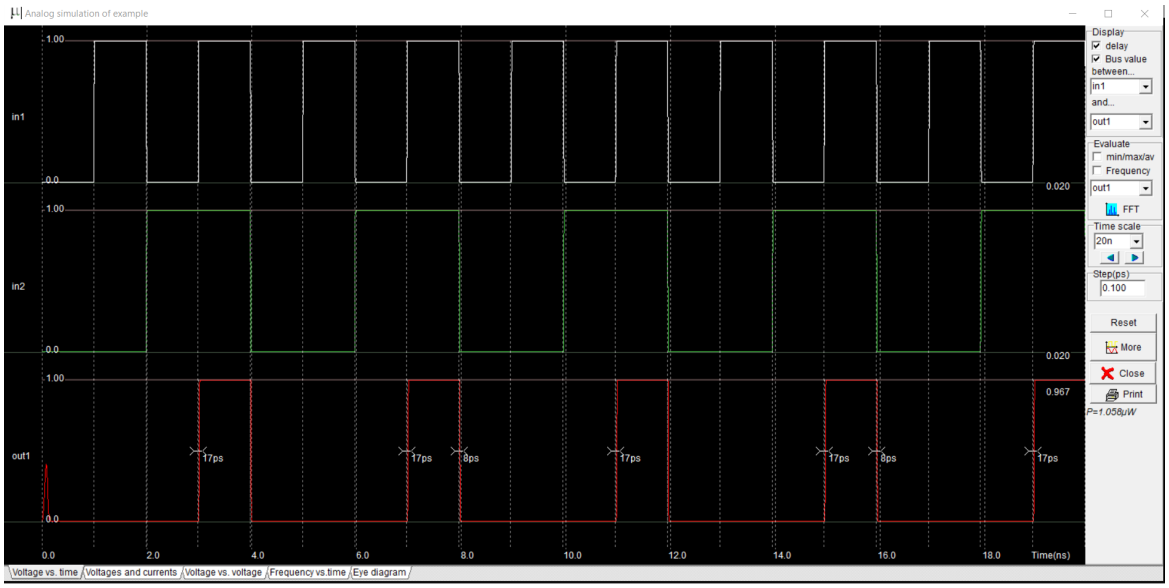
d. codul Verilog obținut



e. Implementarea în Microwind

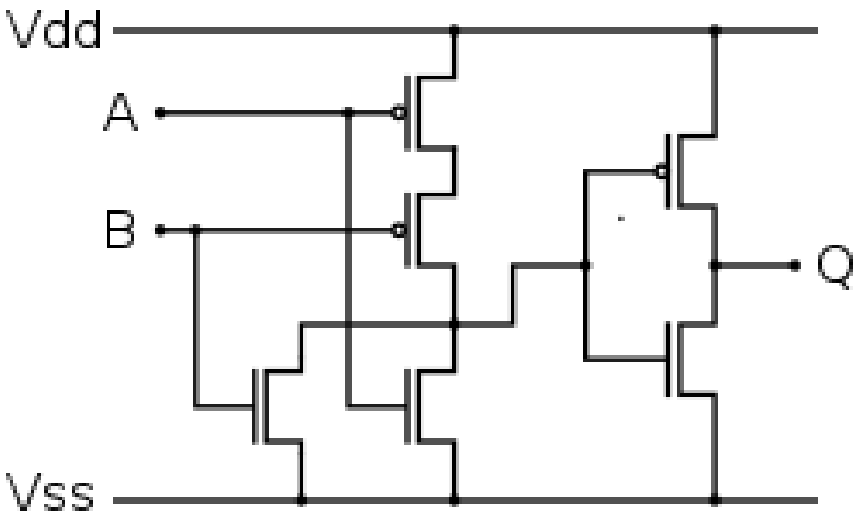


f. Simularea obținută în Microwind

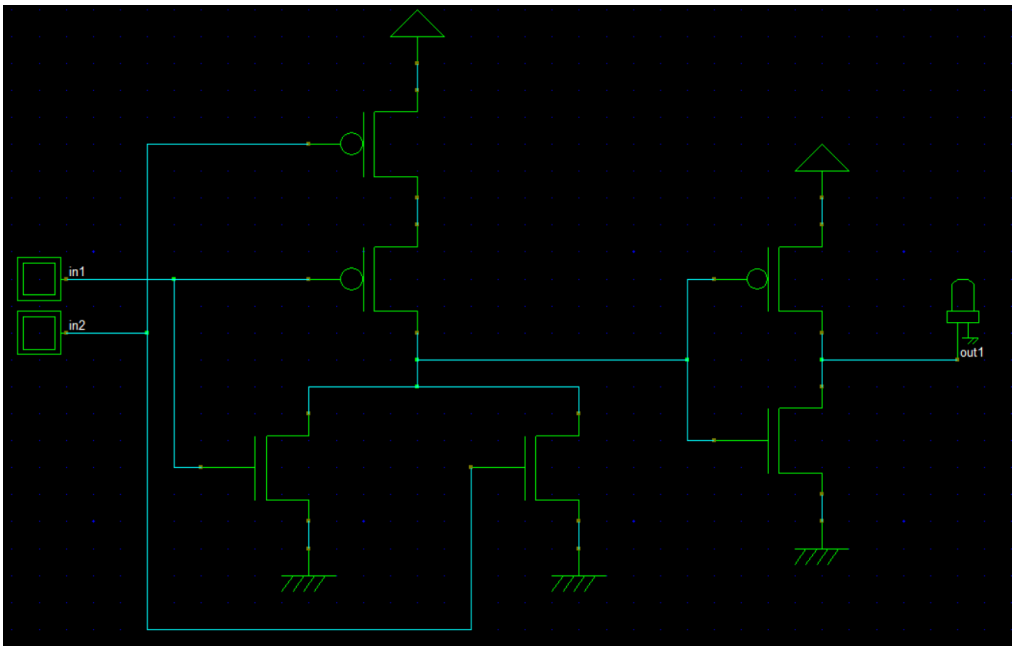


3. OR

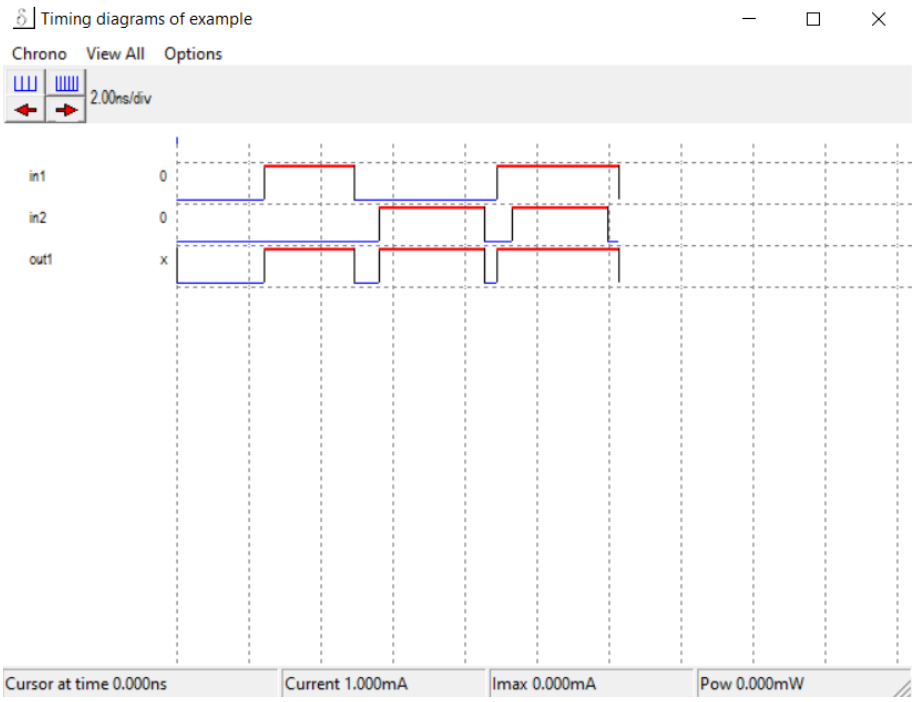
a. Schema circuitului (CMOS/PMOS)
Am preluat schema circuitului de la adresa:
https://en.wikipedia.org/wiki/OR_gate



b. Implementarea în DSCH



c. Simularea în DSCH



d. codul Verilog obținut

Verilog, Hierarchy and Netlist

Verilog | Hierarchy | Netlist | Critical path

```
// DSCH 3.5
// 5/5/2023 7:42:42 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH\or.sch

module or( in1,in2,out1);
 input in1,in2;
 output out1;
 wire w4,w5,;
 pmos #(1) pmos_1(w4,vdd,in2); // 0.5u 0.07u
 pmos #(3) pmos_2(w5,w4,in1); // 0.5u 0.07u
 nmos #(3) nmos_3(w5,vss,in1); // 0.3u 0.07u
 nmos #(3) nmos_4(w5,vss,in2); // 0.3u 0.07u
 pmos #(2) pmos_5(out1,vdd,w5); // 0.5u 0.07u
 nmos #(2) nmos_6(out1,vss,w5); // 0.3u 0.07u
endmodule

// Simulation parameters in Verilog Format
always
#200 in1=~in1;
#400 in2=~in2;

// Simulation parameters
// in1 CLK 1 1
// in2 CLK 2 2
```

Information

Module name (8 char. max)

or

Add gate delay info

Append simul. infomations

Add labels as comments

The Verilog file has 24 lines

The design includes 14 symbols

The circuit has 6 nodes

Misc.

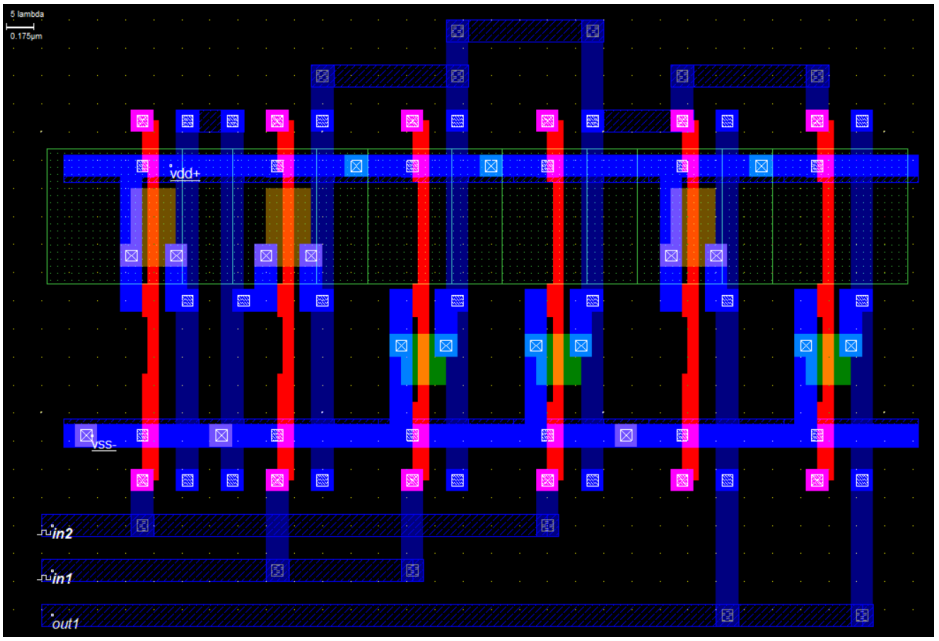
Time scale : 1.00

Max clocks: 16

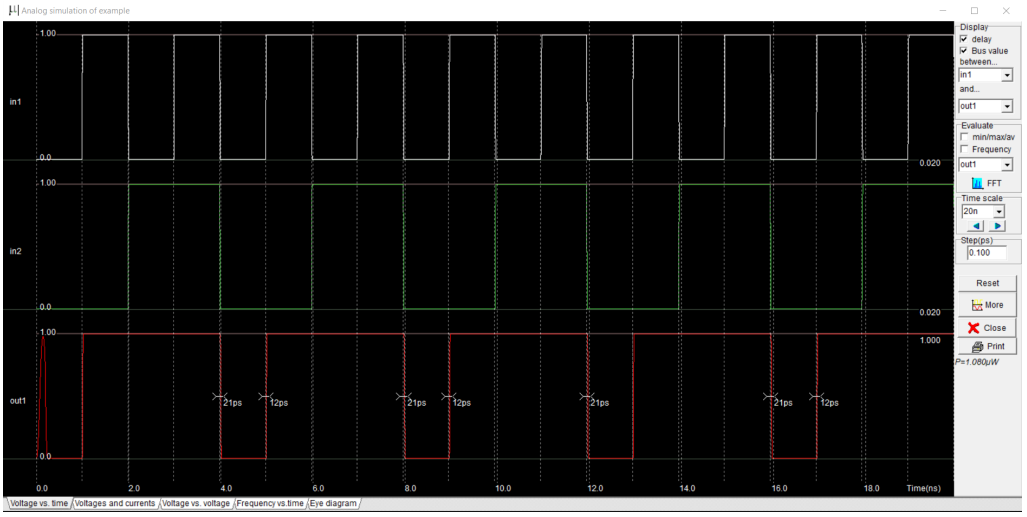
Update Verilog | Extract circuit

OK

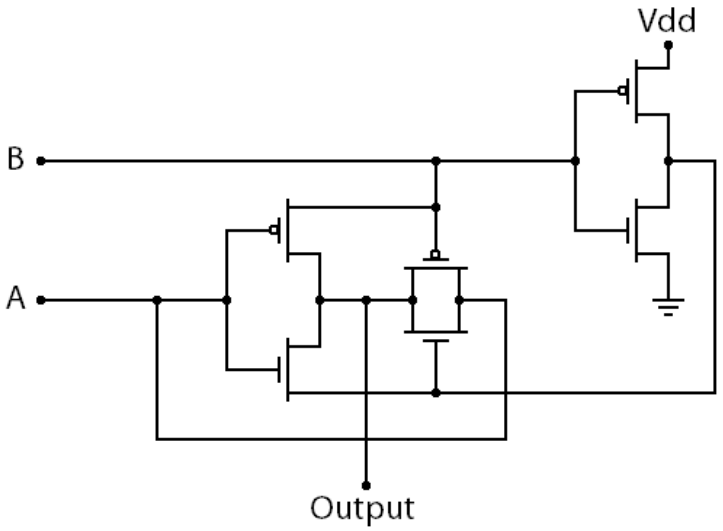
e. Implementarea în Microwind



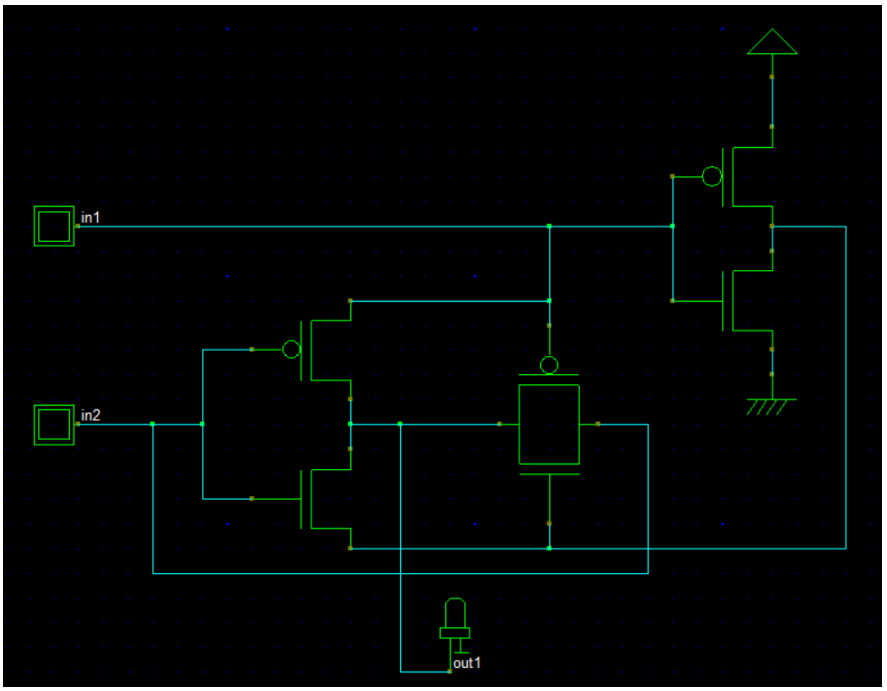
f. Simularea obținută în Microwind



4. XOR
   a. Schema circuitului (CMOS/PMOS)
   Am preluat schema circuitului de la adresa:
   https://en.wikipedia.org/wiki/XOR_gate



b. Implementarea în DSCH



c. Simularea în DSCH

d. codul Verilog obținut

Verilog, Hierarchy and Netlist

Verilog | Hierarchy | Netlist | Critical path

```
// DSCH 3.5
// 5/5/2023 10:51:26 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH\xor.sch

module xor( in1,in2,out1);
 input in1,in2;
 output out1;
 wire w5;
 pmos #(3) pmos_1(out1,in1,in2); // 0.5u 0.07u
 nmos #(3) nmos_2(out1,w5,in2); // 0.3u 0.07u
 pmos #(3) pmos_3(out1,in2,in1); // 0.5u 0.07u
 nmos #(3) nmos_4(out1,in2,w5); // 0.3u 0.07u
 pmos #(2) pmos_5(w5,vdd,in1); // 0.5u 0.07u
 nmos #(2) nmos_6(w5,vss,in1); // 0.3u 0.07u
endmodule

// Simulation parameters in Verilog Format
always
#200 in1=~in1;
#400 in2=~in2;

// Simulation parameters
// in1 CLK 1 1
// in2 CLK 2 2
```

Information

Module name (8 char. max)

xor

☑ Add gate delay info

☑ Append simul. infomations

☐ Add labels as comments

The Verilog file has 24 lines

The design includes 11 symbols

The circuit has 5 nodes

Misc.

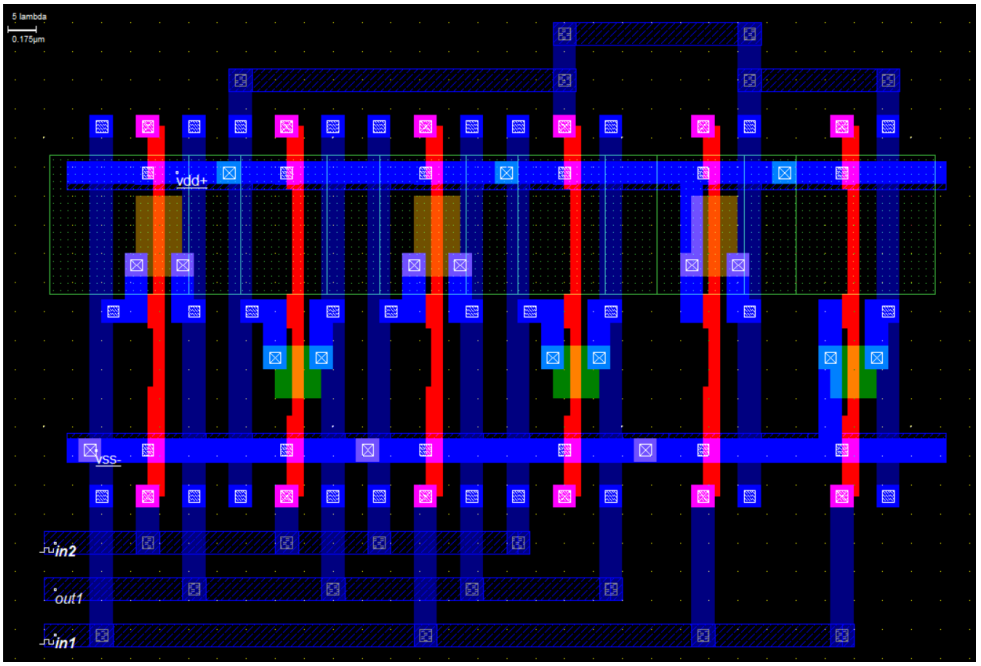
Time scale : 1.00

Max clocks: 16

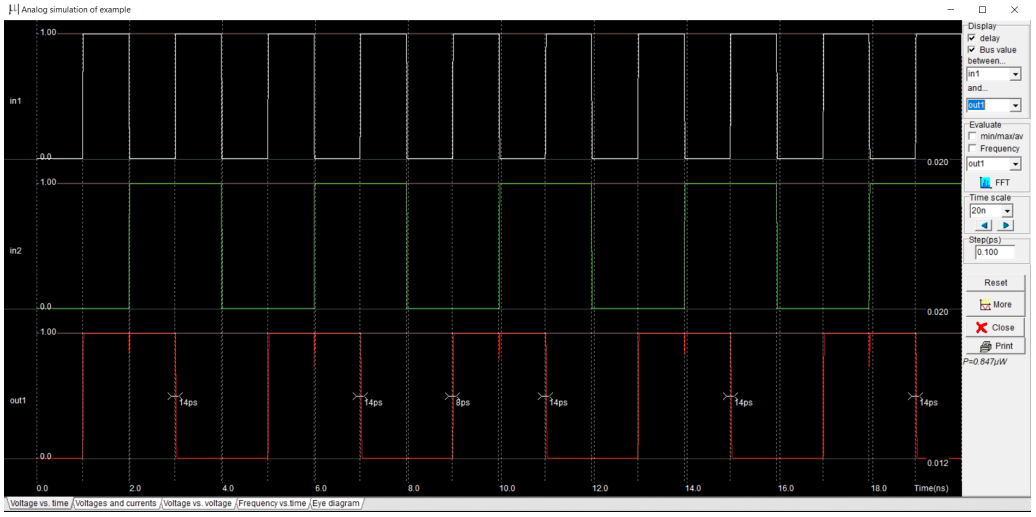
Update Verilog | Extract circuit

OK

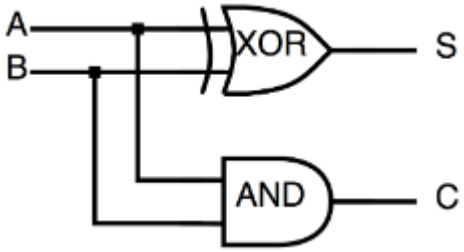
e. Implementarea în Microwind



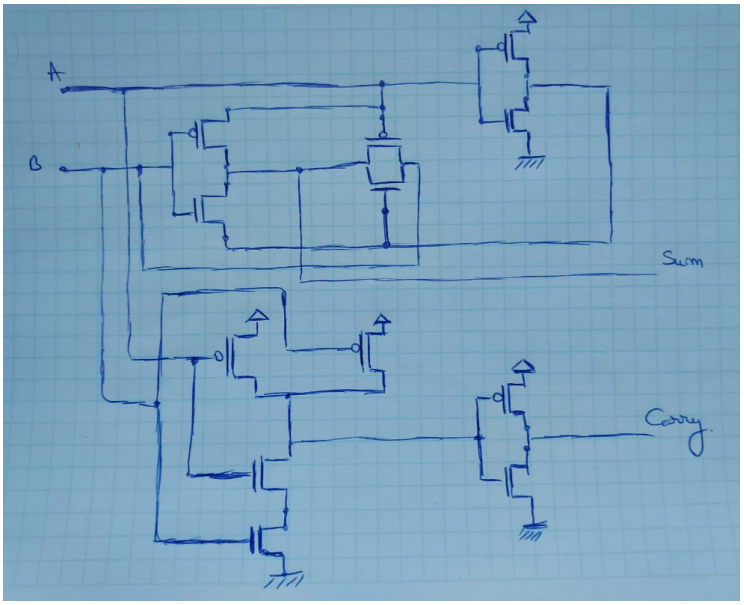
f. Simularea obținută în Microwind



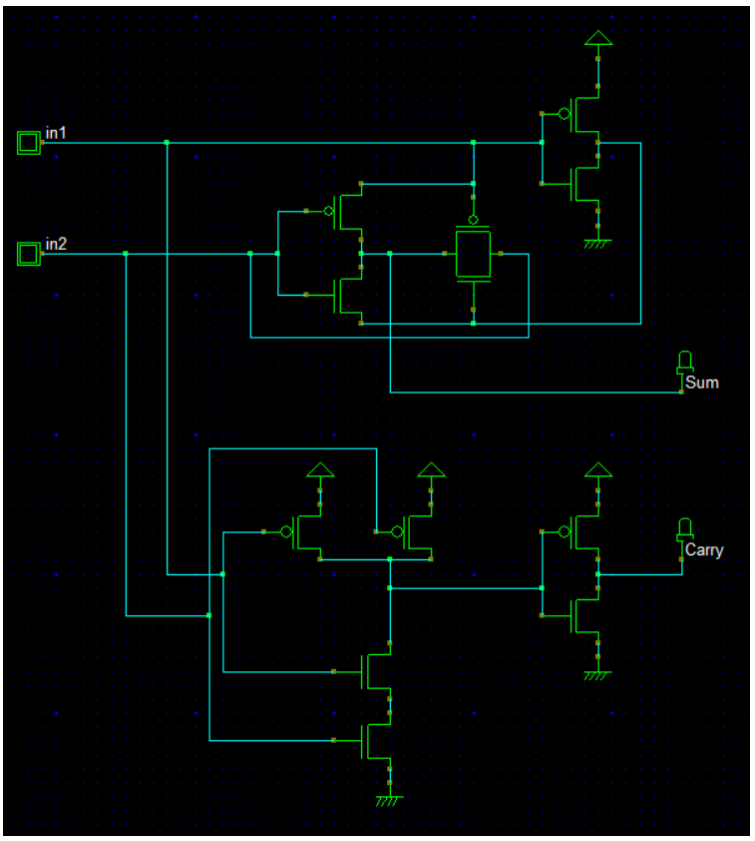
5. Sumator
   a. Schema circuitului (CMOS/PMOS)
   Am preluat schema circuitului de aici:
   https://ocw.cs.pub.ro/courses/cn1/laboratoare/06



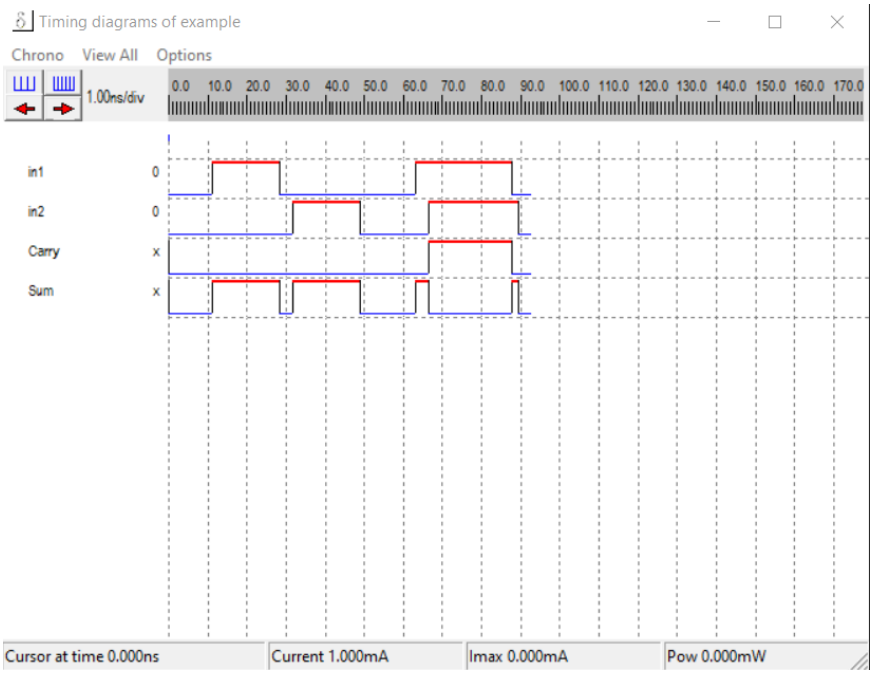
Unde am înlocuit XOR și AND cu schemele făcute la punctele anterioare.
A rezultat următoarea schemă:



b. Implementarea în DSCH



c. Simularea în DSCH



d. codul Verilog obținut

Verilog, Hierarchy and Netlist

Verilog | Hierarchy | Netlist | Critical path

```
// DSCH 3.5
// 5/6/2023 1:16:30 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH\adder.sch

module adder( in1,in2,Carry,Sum);
 input in1,in2;
 output Carry,Sum;
 wire w5,w7,w8;
 pmos #(3) pmos_1(Sum,in1,in2); // 0.5u 0.07u
 pmos #(3) pmos_2(w5,vdd,in1); // 0.5u 0.07u
 pmos #(3) pmos_3(w5,vdd,in2); // 0.5u 0.07u
 pmos #(2) pmos_4(Carry,vdd,w5); // 0.5u 0.07u
 nmos #(2) nmos_5(Carry,vss,w5); // 0.3u 0.07u
 nmos #(3) nmos_6(w5,w7,in1); // 0.3u 0.07u
 nmos #(1) nmos_7(w7,vss,in2); // 0.3u 0.07u
 nmos #(2) nmos_8(w8,vss,in1); // 0.3u 0.07u
 pmos #(2) pmos_9(w8,vdd,in1); // 0.5u 0.07u
 nmos #(3) nmos_10(Sum,in2,w8); // 0.3u 0.07u
 pmos #(3) pmos_11(Sum,in2,in1); // 0.5u 0.07u
 nmos #(3) nmos_12(Sum,w8,in2); // 0.3u 0.07u
endmodule

// Simulation parameters in Verilog Format
always
#200 in1=~in1;
#400 in2=~in2;

// Simulation parameters
// in1 CLK 1 1
// in2 CLK 2 2
```

Information

Module name (8 char. max)

adder

☑ Add gate delay info

☑ Append simul. infomations

☐ Add labels as comments

The Verilog file has 30 lines

The design includes 23 symbols

The circuit has 8 nodes

Misc.

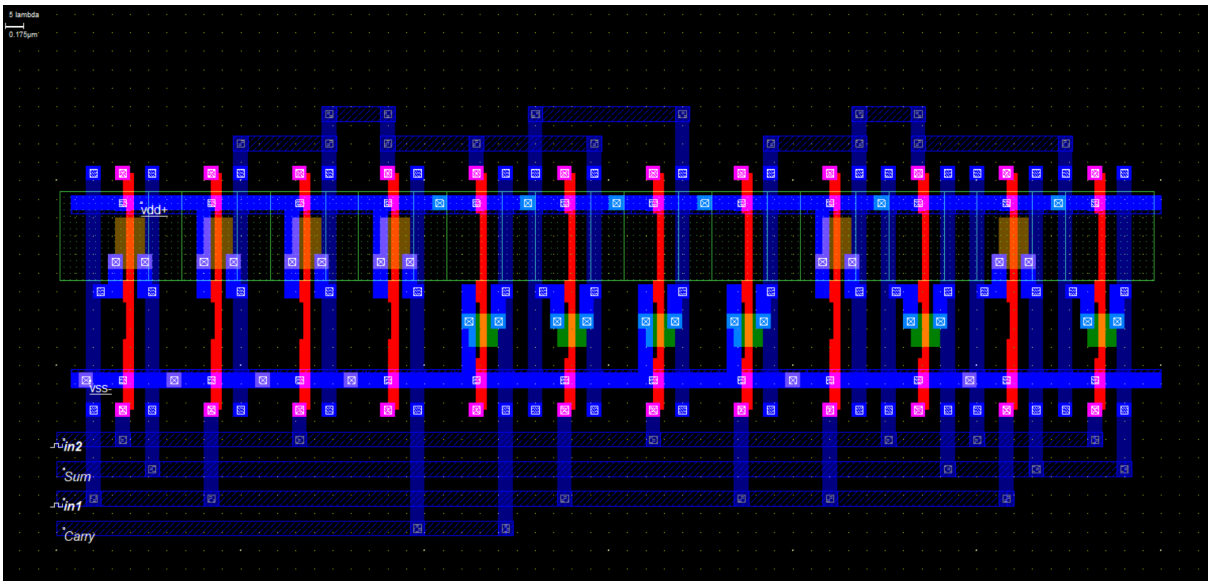
Time scale : 1.00

Max clocks: 16

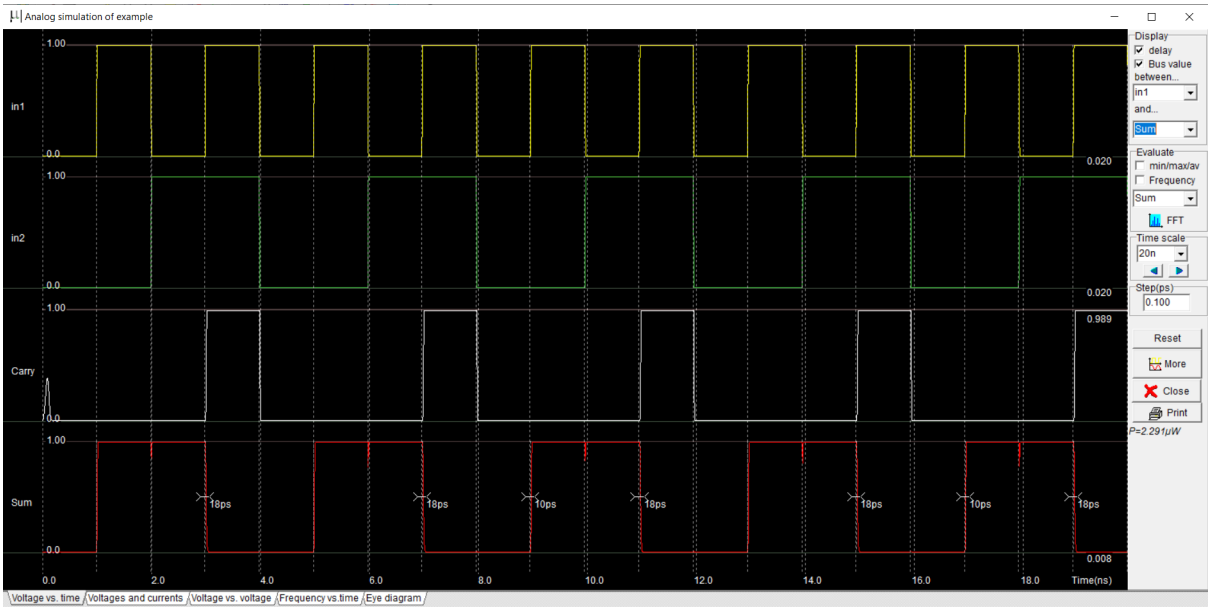
Update Verilog | Extract circuit

OK

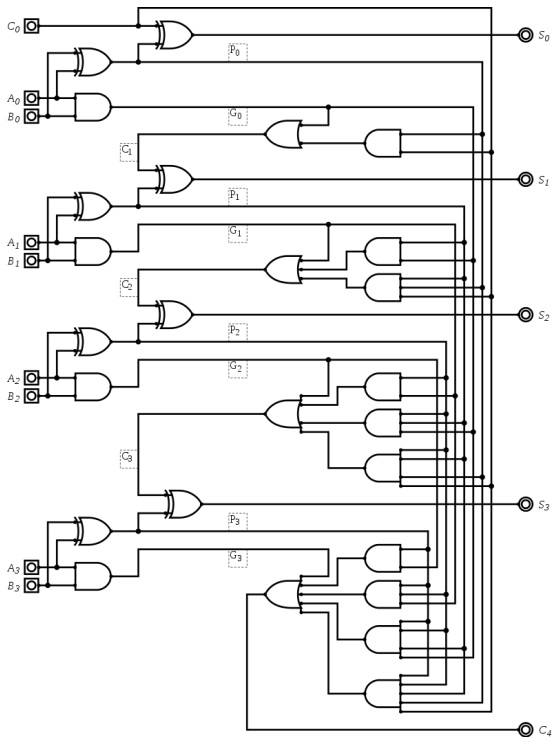
e. Implementarea în Microwind



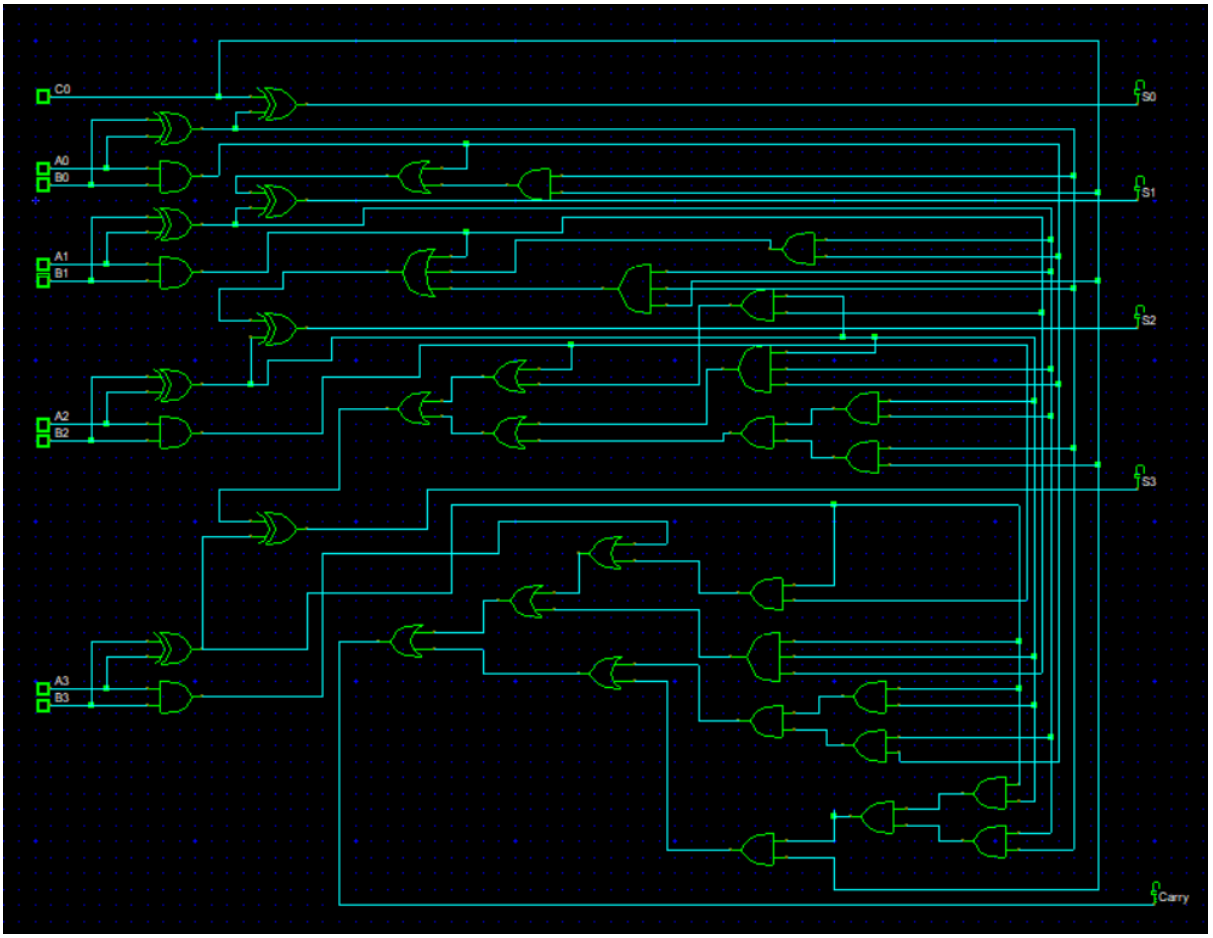
f. Simularea obținută în Microwind



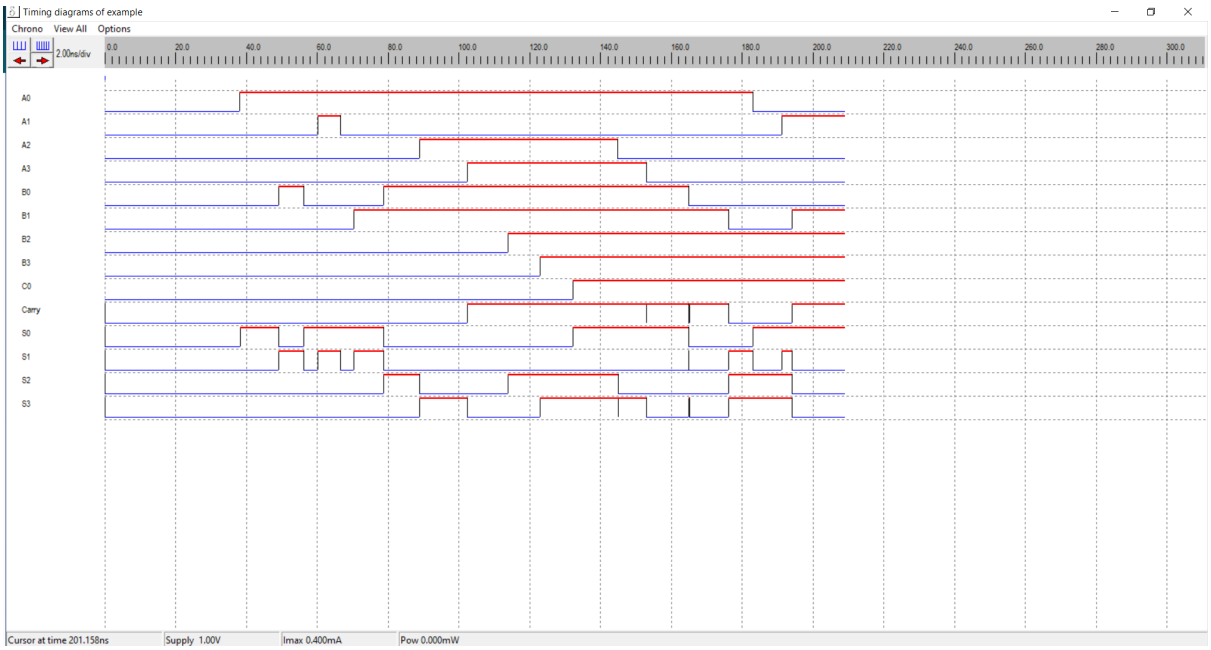
6. Sumator 4 biți Carry Look Ahead
   a. Schema circuitului (porți logice)
      Am preluat schema circuitului de la această adresă:
      https://en.wikipedia.org/wiki/Carry-lookahead_adder



b. Implementarea în DSCH



c. Simularea în DSCH



d. codul Verilog obținut

Verilog, Hierarchy and Netlist

Verilog | Hierarchy | Netlist | Critical path

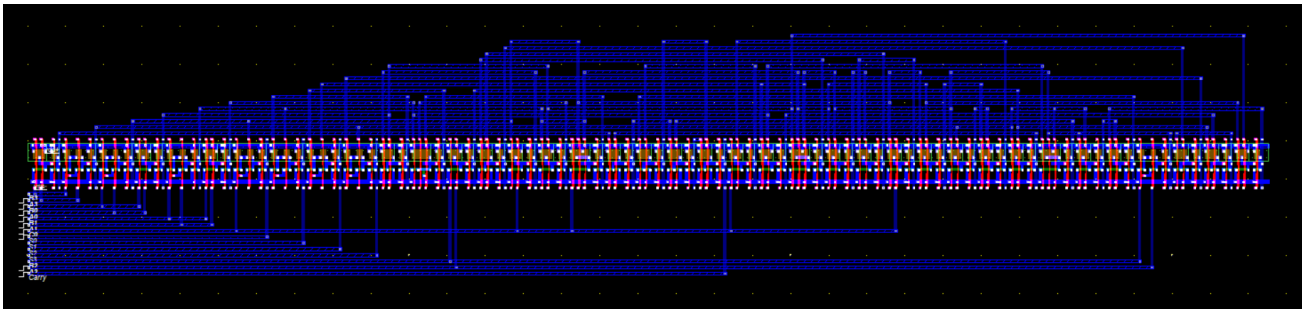
```
// DSCH 3.5
// 5/6/2023 4:08:58 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH\cla.sch

module cla( C0,A0,B0,A1,B1,B3,A3,A2,
 B2,Carry,S3,S2,S1,S0);
 input C0,A0,B0,A1,B1,B3,A3,A2;
 input B2;
 output Carry,S3,S2,S1,S0;
 wire w9,w10,w13,w14,w15,w16,w20,w21;
 wire w22,w23,w24,w25,w26,w29,w30,w31;
 wire w32,w33,w34,w35,w36,w37,w38,w39;
 wire w40,w41,w42,w43,w44,w45,w46,w47;
 wire w48;
 and #(3) and2_1(w9,B3,A3);
 xor #(5) xor2_2(w10,B3,A3);
 xor #(5) xor2_3(w13,B0,A0);
 and #(5) and2_4(w14,B0,A0);
 xor #(6) xor2_5(w15,B1,A1);
 and #(4) and2_6(w16,B1,A1);
 xor #(3) xor2_7(S0,C0,w13);
 xor #(3) xor2_8(S1,w20,w15);
 xor #(3) xor2_9(S2,w21,w22);
 xor #(3) xor2_10(S3,w23,w10);
 or #(3) or2_11(w20,w24,w14);
 or #(4) or3_12(w21,w16,w25,w26);
 and #(4) and2_13(w29,B2,A2);
 or #(3) or2_14(w32,w30,w31);
 and #(3) and2_15(w24,w13,C0);
 and #(3) and2_16(w25,w15,w14);
 and #(3) and3_17(w26,C0,w13,w15);
 or #(3) or2_18(w34,w9,w33);
 or #(3) or2_19(w37,w35,w36);
 or #(3) or2_20(w39,w34,w38);
 or #(3) or2_21(Carry,w39,w37);
 and #(3) and2_22(w36,C0,w40);
 and #(3) and2_23(w41,w22,w16);
 and #(3) and3_24(w30,w14,w15,w22);
 and #(3) and2_25(w42,w15,w22);
 and #(3) and2_26(w31,w43,w42);
 and #(3) and2_27(w43,w13,C0);
 and #(3) and2_28(w44,w13,w15);
 and #(3) and2_29(w45,w22,w10);
 and #(3) and2_30(w40,w44,w45);
 and #(3) and2_31(w33,w10,w29);
 and #(3) and3_32(w38,w16,w22,w10);
```

Verilog, Hierarchy and Netlist

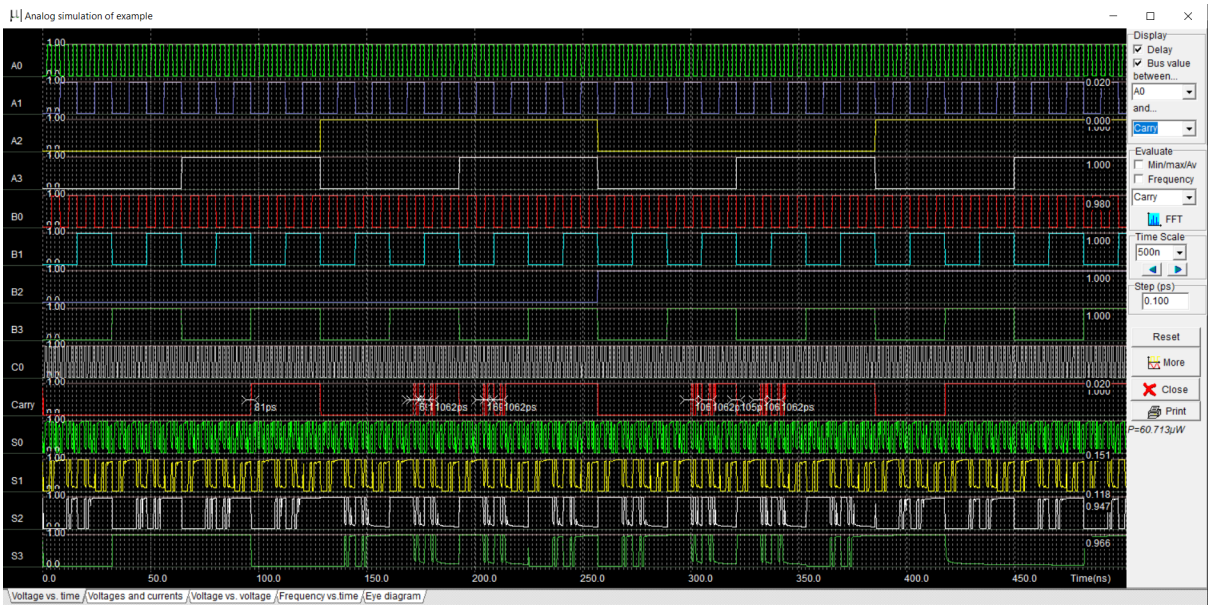
Verilog | Hierarchy | Netlist | Critical path

```
 and #(3) and2_16(w25,w15,w14);
 and #(3) and3_17(w26,C0,w13,w15);
 or #(3) or2_18(w34,w9,w33);
 or #(3) or2_19(w37,w35,w36);
 or #(3) or2_20(w39,w34,w38);
 or #(3) or2_21(Carry,w39,w37);
 and #(3) and2_22(w36,C0,w40);
 and #(3) and2_23(w41,w22,w16);
 and #(3) and3_24(w30,w14,w15,w22);
 and #(3) and2_25(w42,w15,w22);
 and #(3) and2_26(w31,w43,w42);
 and #(3) and2_27(w43,w13,C0);
 and #(3) and2_28(w44,w13,w15);
 and #(3) and2_29(w45,w22,w10);
 and #(3) and2_30(w40,w44,w45);
 and #(3) and2_31(w33,w10,w29);
 and #(3) and3_32(w38,w16,w22,w10);
 and #(3) and2_33(w46,w22,w10);
 and #(3) and2_34(w35,w47,w46);
 xor #(6) xor2_35(w22,B2,A2);
 or #(3) or2_36(w23,w48,w32);
 or #(3) or2_37(w48,w29,w41);
 and #(3) and2_38(w47,w15,w14);
endmodule

// Simulation parameters in Verilog Format
always
#200 C0=~C0;
#400 A0=~A0;
#800 B0=~B0;
#1600 A1=~A1;
#3200 B1=~B1;
#6400 B3=~B3;
#12800 A3=~A3;
#25600 A2=~A2;
#51200 B2=~B2;

// Simulation parameters
// C0 CLK 1 1
// A0 CLK 2 2
// B0 CLK 4 4
// A1 CLK 8 8
// B1 CLK 16 16
// B3 CLK 32 32
// A3 CLK 64 64
// A2 CLK 128 128
// B2 CLK 256 256
```

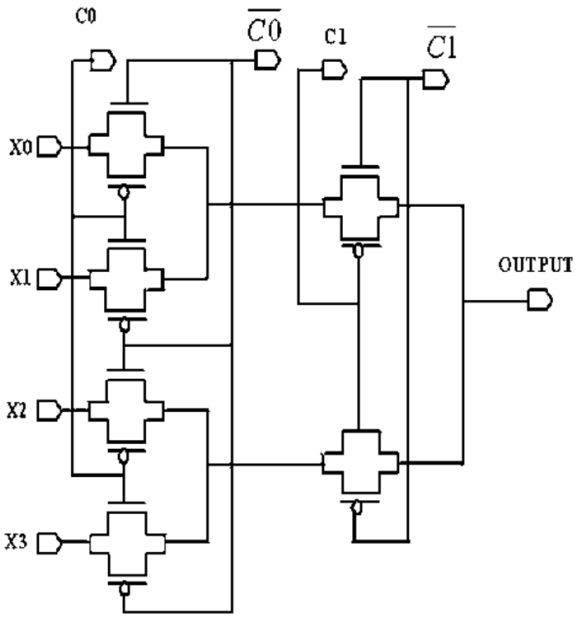
e. Implementarea în Microwind



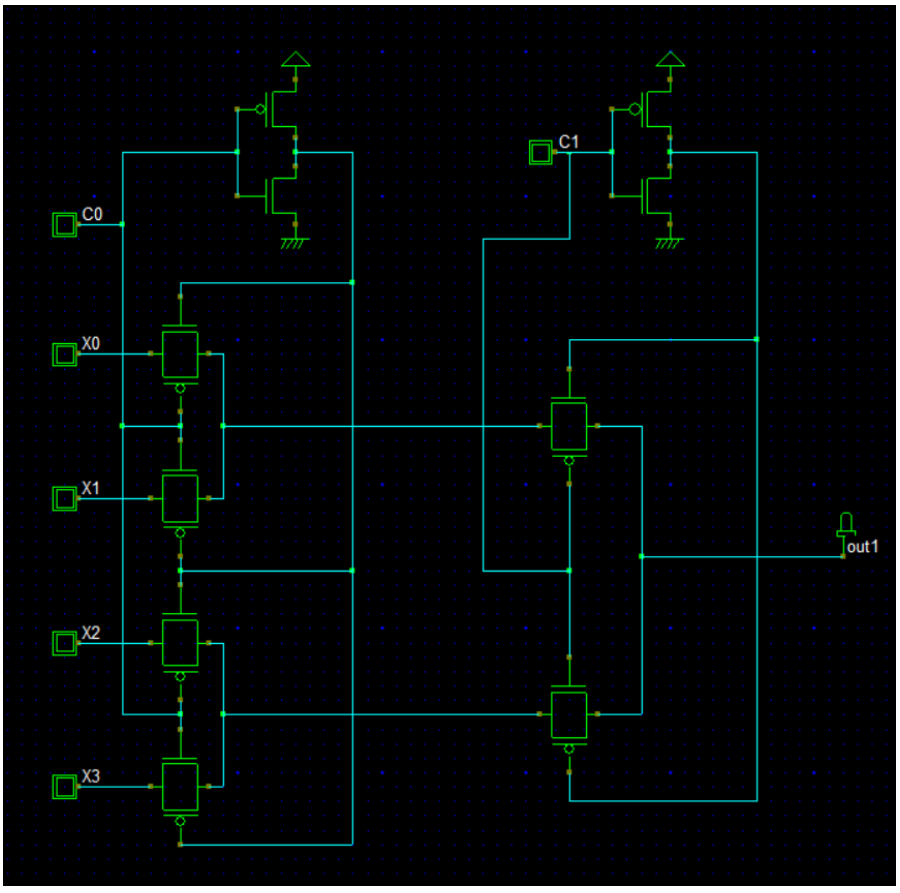
f. Simularea obținută în Microwind



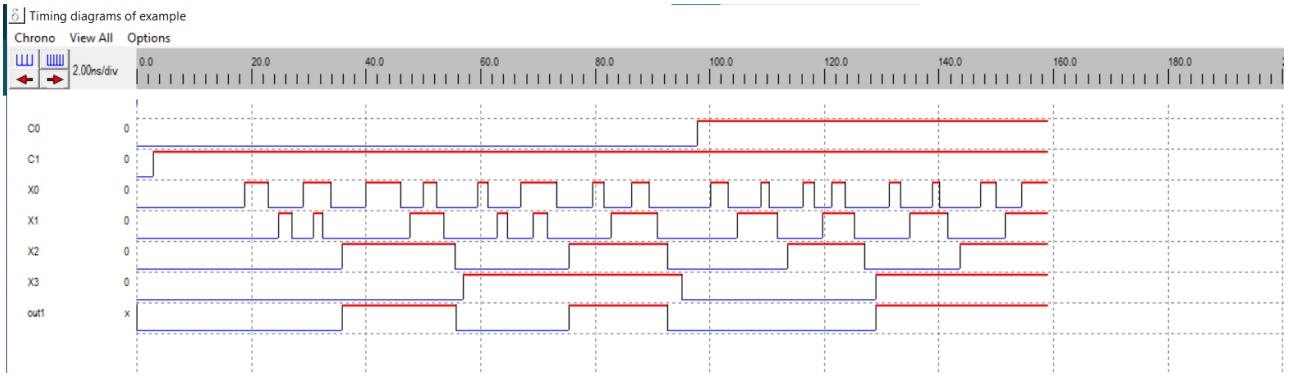
7. MUX 4:1
    a. Schema circuitului (CMOS/PMOS)
    Schema circuitului am preluat-o aici:
    https://www.researchgate.net/figure/Transmission-gate-based-41-MUX_fig5_257799438

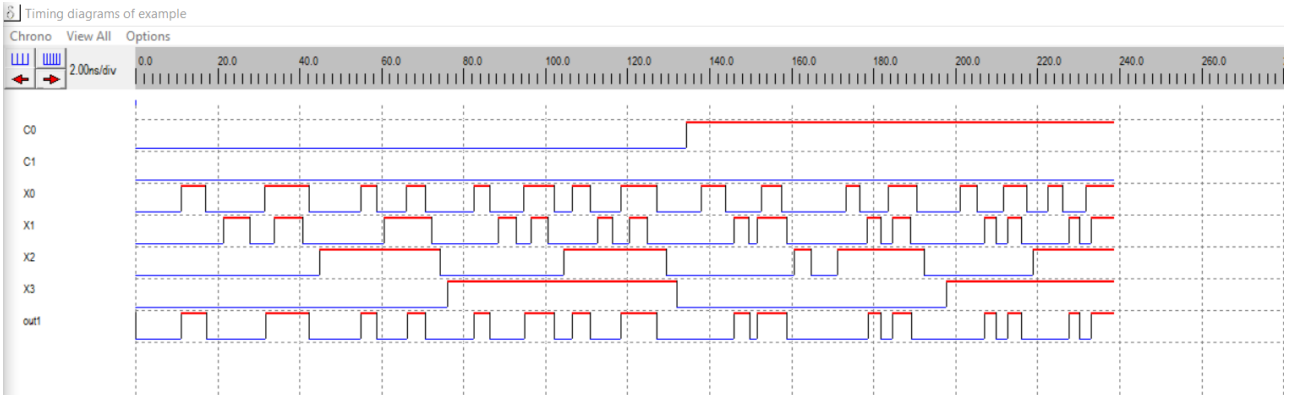


    b. Implementarea în DSCH



c. Simularea în DSCH





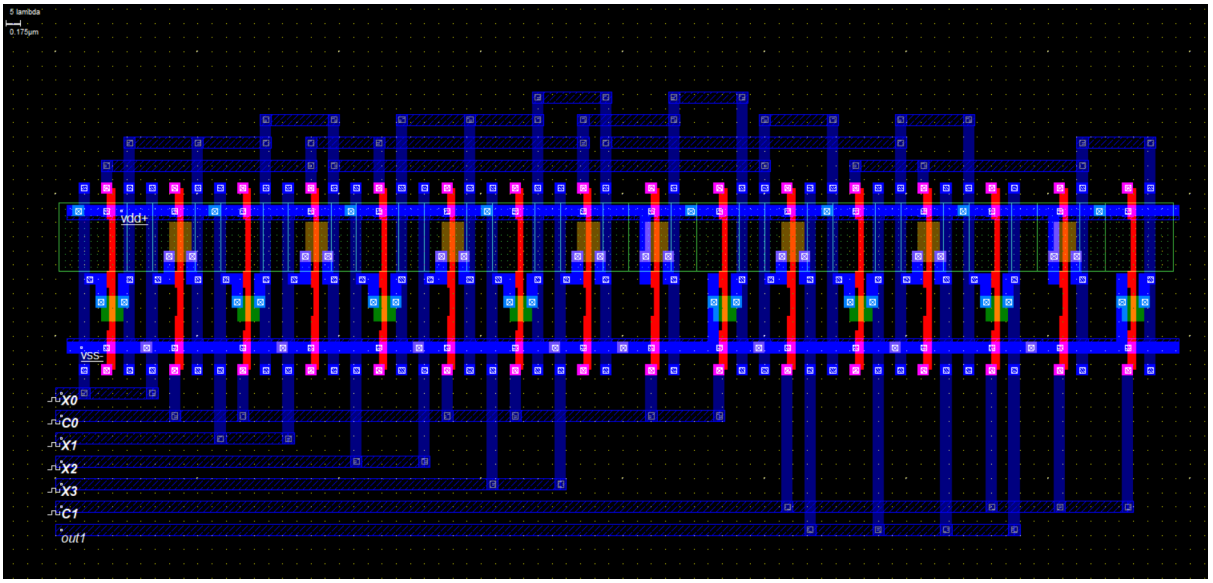
d. codul Verilog obținut

Verilog, Hierarchy and Netlist

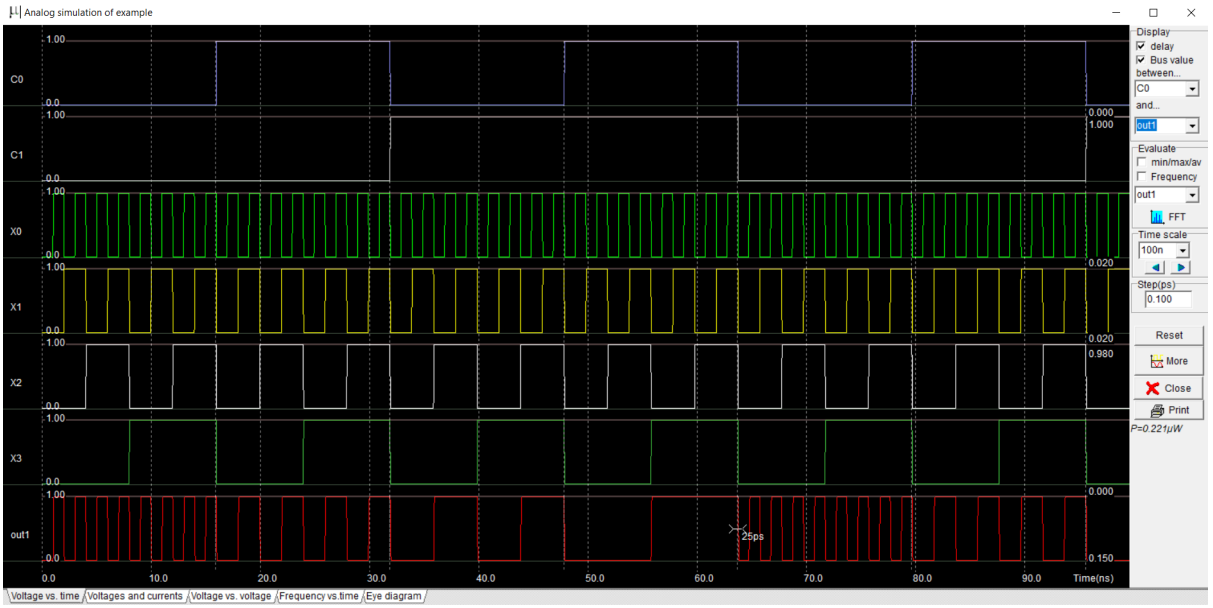
Verilog | Hierarchy | Netlist | Critical path

```
// DSCH 3.5
// 5/6/2023 5:49:03 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH\mux.sch

module mux( X0,X1,X2,X3,C0,C1,out1);
 input X0,X1,X2,X3,C0,C1;
 output out1;
 wire w6,w7,w9,w12;
 nmos #(3) nmos_1(w7,X0,w6); // 0.3u 0.07u
 pmos #(3) pmos_2(w7,X0,C0); // 0.5u 0.07u
 nmos #(3) nmos_3(w7,X1,C0); // 0.3u 0.07u
 pmos #(3) pmos_4(w7,X1,w6); // 0.5u 0.07u
 nmos #(3) nmos_5(w9,X2,w6); // 0.3u 0.07u
 pmos #(3) pmos_6(w9,X2,C0); // 0.5u 0.07u
 nmos #(3) nmos_7(w9,X3,C0); // 0.3u 0.07u
 pmos #(3) pmos_8(w9,X3,w6); // 0.5u 0.07u
 pmos #(3) pmos_9(w6,vdd,C0); // 0.5u 0.07u
 nmos #(3) nmos_10(w6,vss,C0); // 0.3u 0.07u
 pmos #(3) pmos_11(out1,w7,C1); // 0.5u 0.07u
 nmos #(3) nmos_12(out1,w7,w12); // 0.3u 0.07u
 pmos #(3) pmos_13(out1,w9,w12); // 0.5u 0.07u
 nmos #(3) nmos_14(out1,w9,C1); // 0.3u 0.07u
 pmos #(2) pmos_15(w12,vdd,C1); // 0.5u 0.07u
 nmos #(2) nmos_16(w12,vss,C1); // 0.3u 0.07u
endmodule

// Simulation parameters in Verilog Format
always
#200 X0=~X0;
#400 X1=~X1;
#800 X2=~X2;
#1600 X3=~X3;
#3200 C0=~C0;
#6400 C1=~C1;

// Simulation parameters
// X0 CLK 1 1
// X1 CLK 2 2
// X2 CLK 4 4
// X3 CLK 8 8
// C0 CLK 16 16
// C1 CLK 32 32
```

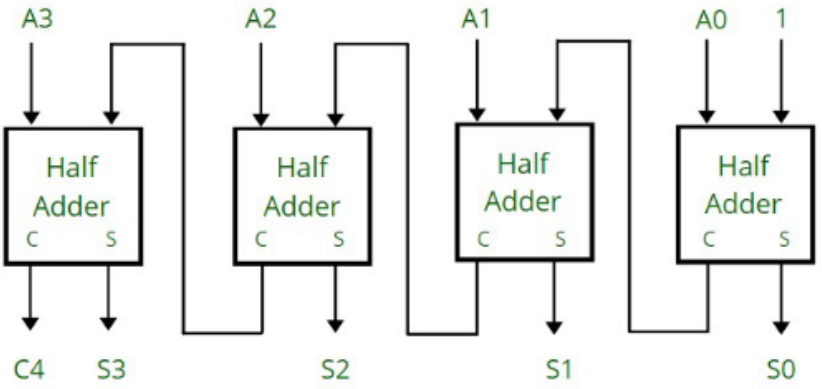
e. Implementarea în Microwind



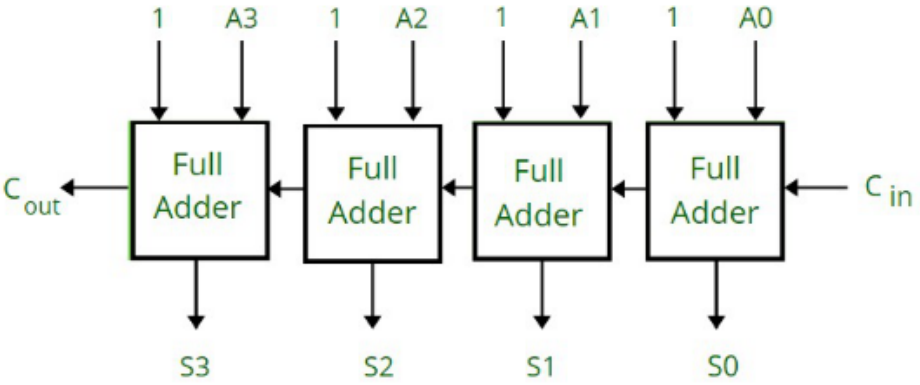
f. Simularea obținută în Microwind



8. Incrementator / decrementator 4 biți
    a. Schema circuitului (porți logice)
    Am preluat ideea de la aceste adrese:
    https://www.geeksforgeeks.org/4-bit-binary-incrementer/
    https://www.geeksforgeeks.org/4-bit-binary-decrementer/
    Practic voi folosi schema de la adder CLA, unde voi aduna 0001 dacă e incrementer sau voi aduna 1111 dacă e decrementer.

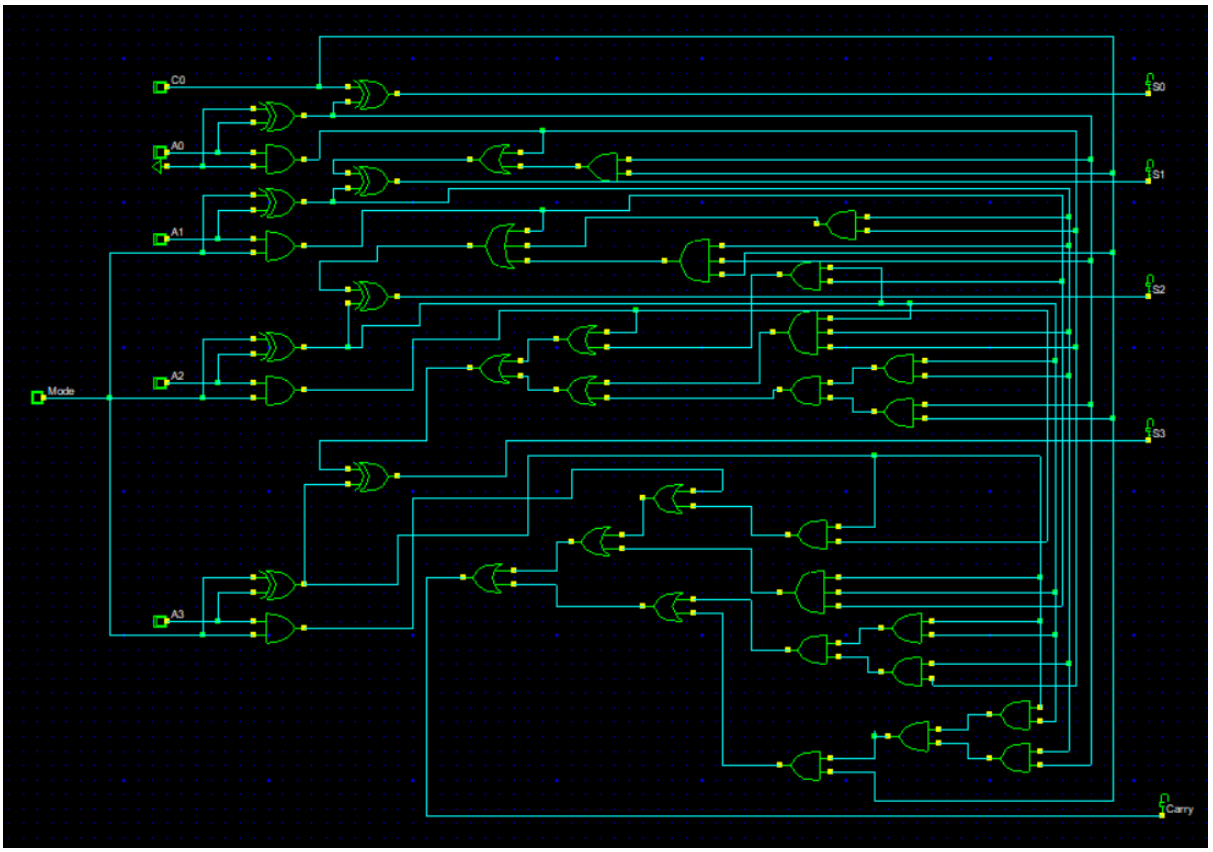


4- Bit Binary Incrementer



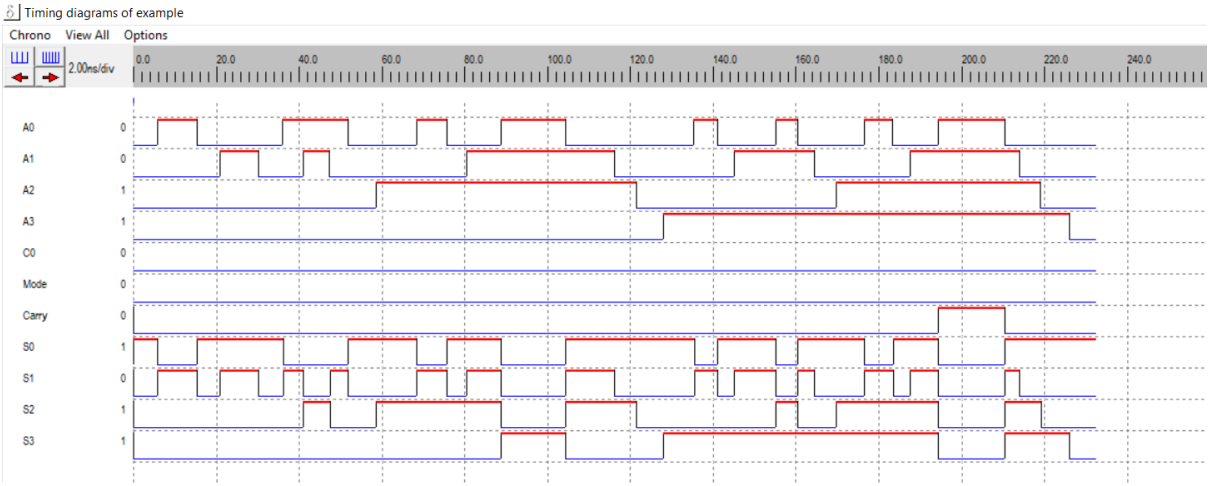
4- Bit Binary Decrementer

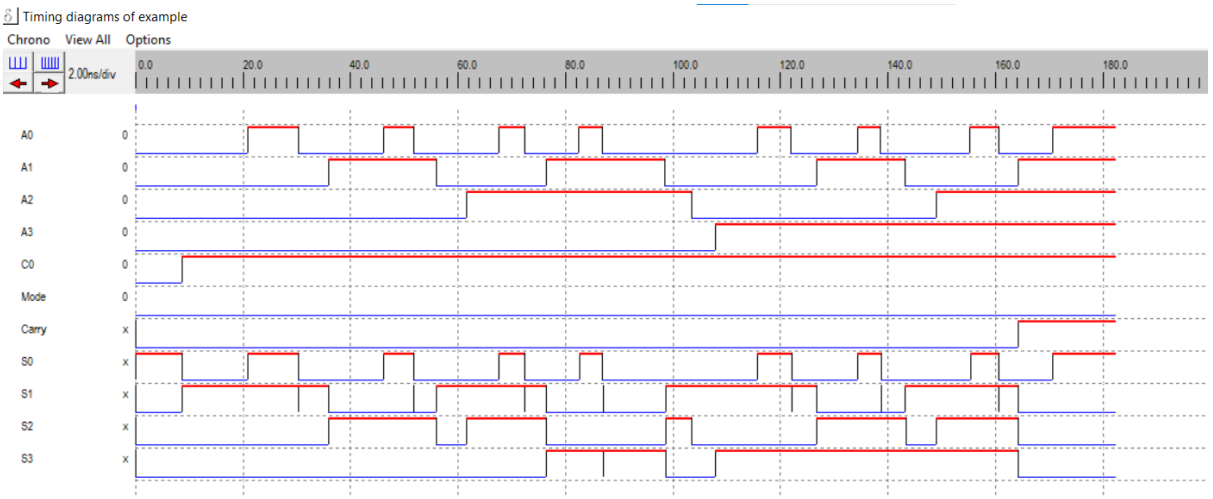
b. Implementarea în DSCH



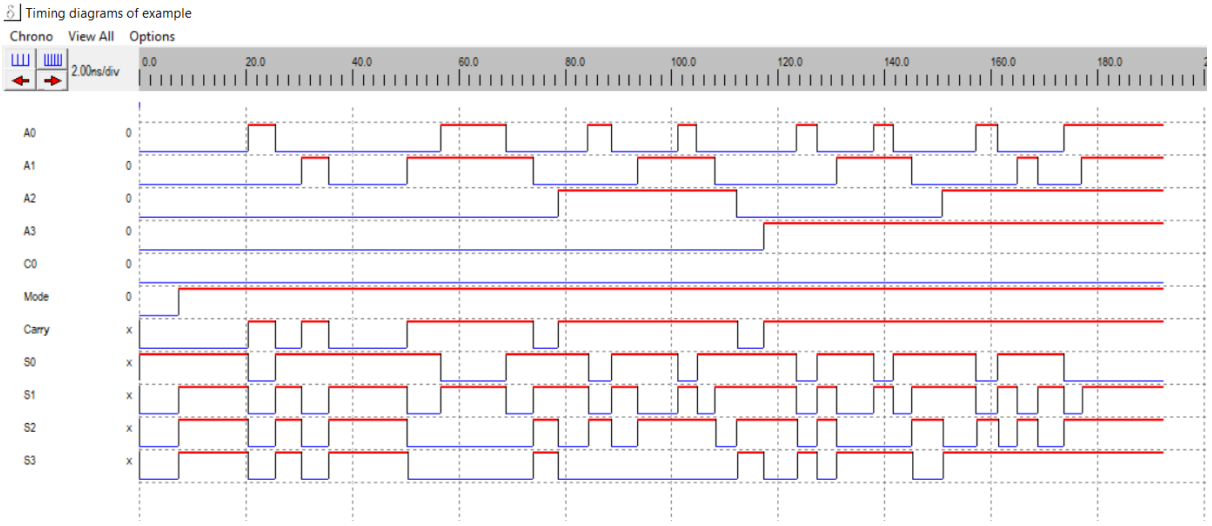
c. Simularea în DSCH

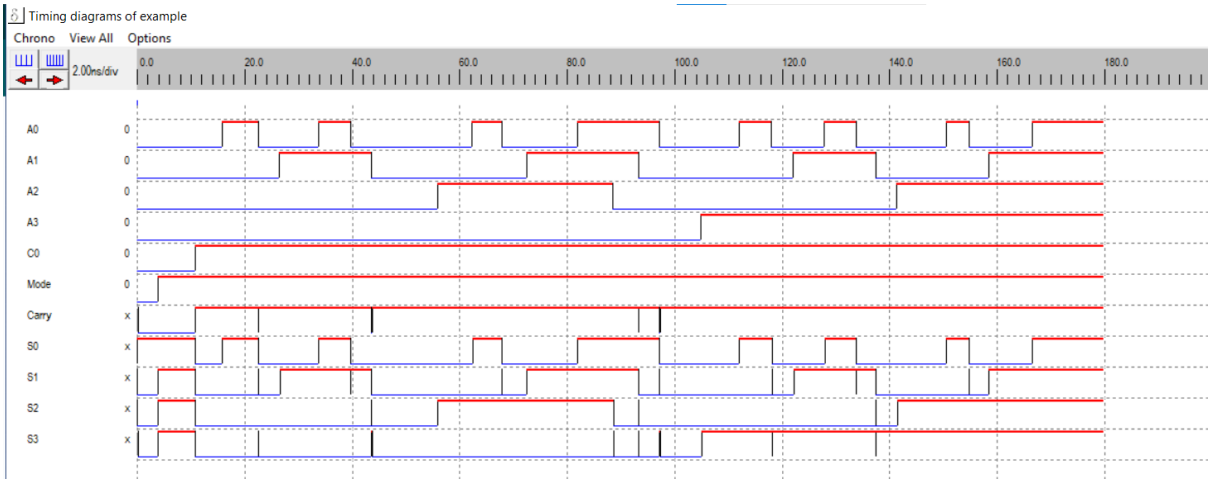
i. incrementer





ii. decrementer





d. codul Verilog obținut

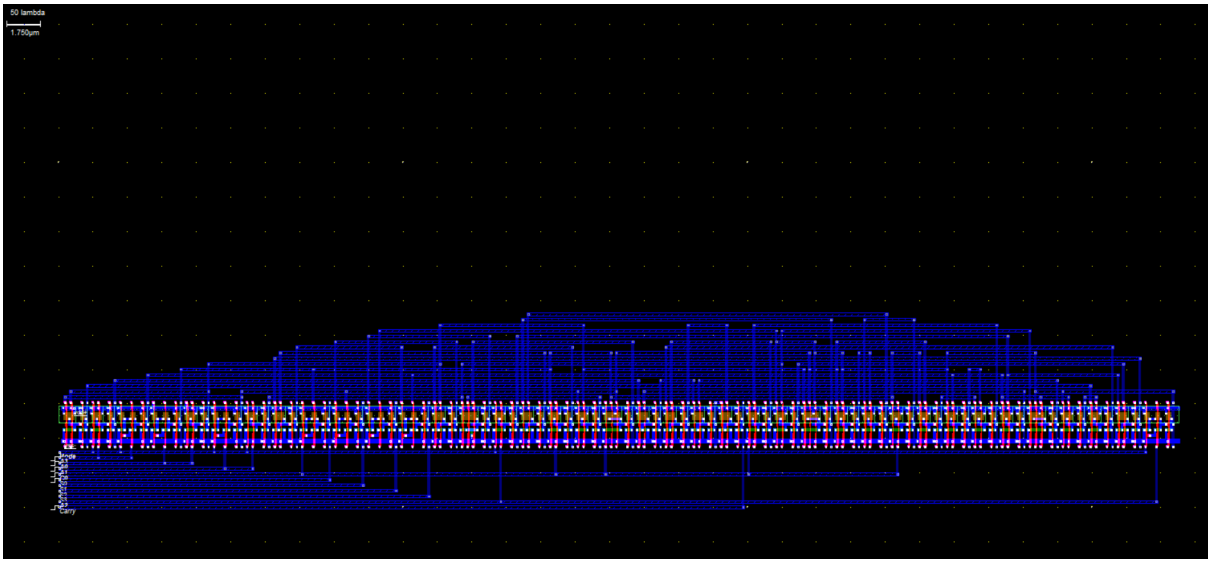
Verilog, Hierarchy and Netlist

Verilog | Hierarchy | Netlist | Critical path

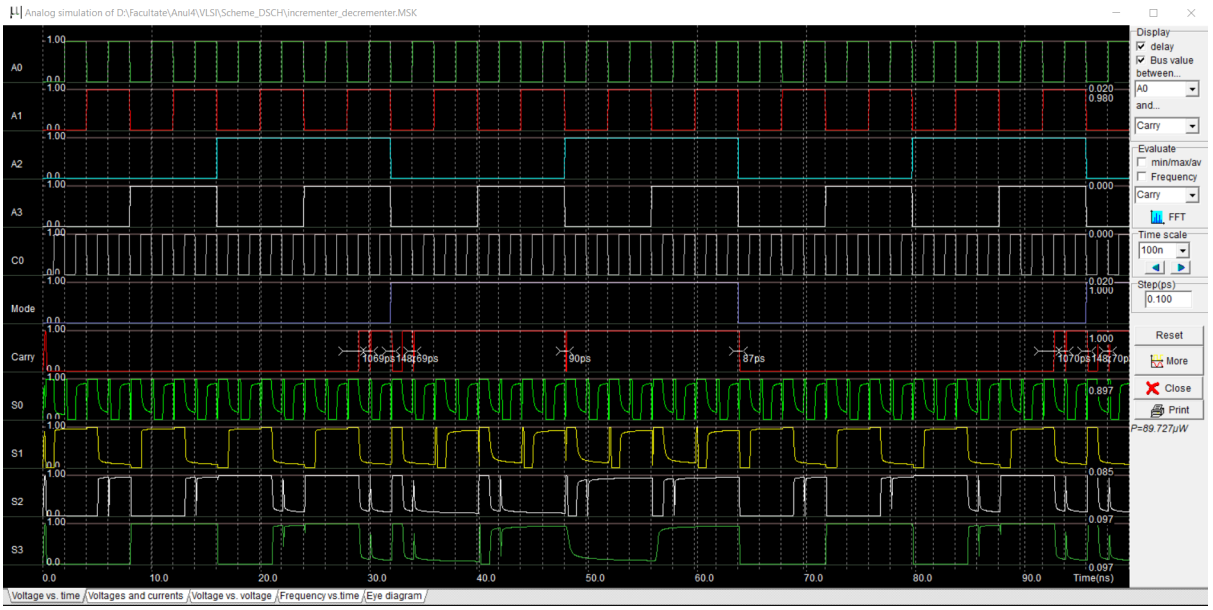
```
// DSCH 3.5
// 5/6/2023 9:06:30 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH\incrementer_decrementer.sch

module incrementer_decrementer( C0,A0,A1,A3,A2,Mode,Carry,S3,
 S2,S1,S0);
 input C0,A0,A1,A3,A2,Mode;
 output Carry,S3,S2,S1,S0;
 wire w5,w6,w7,w10,w11,w14,w15,w16;
 wire w17,w18,w22,w23,w24,w25,w26,w27;
 wire w29,w30,w31,w32,w33,w34,w35,w36;
 wire w37,w38,w39,w40,w41,w42,w43,w44;
 wire w45;
 and #(3) and2_1(w7,w5,w6);
 and #(3) and2_2(w10,Mode,A3);
 xor #(5) xor2_3(w11,Mode,A3);
 xor #(5) xor2_4(w14,vdd,A0);
 and #(5) and2_5(w6,vdd,A0);
 xor #(6) xor2_6(w5,Mode,A1);
 and #(4) and2_7(w15,Mode,A1);
 or #(3) or2_8(w18,w16,w17);
 xor #(3) xor2_9(S0,C0,w14);
 xor #(3) xor2_10(S1,w22,w5);
 xor #(3) xor2_11(S2,w23,w24);
 xor #(3) xor2_12(S3,w18,w11);
 or #(3) or2_13(w22,w25,w6);
 or #(4) or3_14(w23,w15,w26,w27);
 and #(4) and2_15(w29,Mode,A2);
 or #(3) or2_16(w17,w30,w31);
 and #(3) and2_17(w25,w14,C0);
 and #(3) and2_18(w26,w5,w6);
 and #(3) and3_19(w27,C0,w14,w5);
 or #(3) or2_20(w33,w10,w32);
 or #(3) or2_21(w36,w34,w35);
 or #(3) or2_22(w38,w33,w37);
 or #(3) or2_23(Carry,w38,w36);
 and #(3) and2_24(w35,C0,w39);
 and #(3) and2_25(w40,w24,w15);
 and #(3) and3_26(w30,w6,w5,w24);
 and #(3) and2_27(w41,w5,w24);
 and #(3) and2_28(w31,w42,w41);
 and #(3) and2_29(w42,w14,C0);
 and #(3) and2_30(w43,w14,w5);
 and #(3) and2_31(w44,w24,w11);
 and #(3) and2_32(w39,w43,w44);
 and #(3) and2_33(w32,w11,w29);
 and #(3) and3_34(w37,w15,w24,w11);
```

```
 and #(3) and3_34(w37,w15,w24,w11);
 and #(3) and2_35(w45,w24,w11);
 and #(3) and2_36(w34,w7,w45);
 or #(3) or2_37(w16,w29,w40);
 xor #(6) xor2_38(w24,Mode,A2);
endmodule

// Simulation parameters in Verilog Format
always
#200 C0=~C0;
#400 A0=~A0;
#800 A1=~A1;
#1600 A3=~A3;
#3200 A2=~A2;
#6400 Mode=~Mode;

// Simulation parameters
// C0 CLK 1 1
// A0 CLK 2 2
// A1 CLK 4 4
// A3 CLK 8 8
// A2 CLK 16 16
// Mode CLK 32 32
```

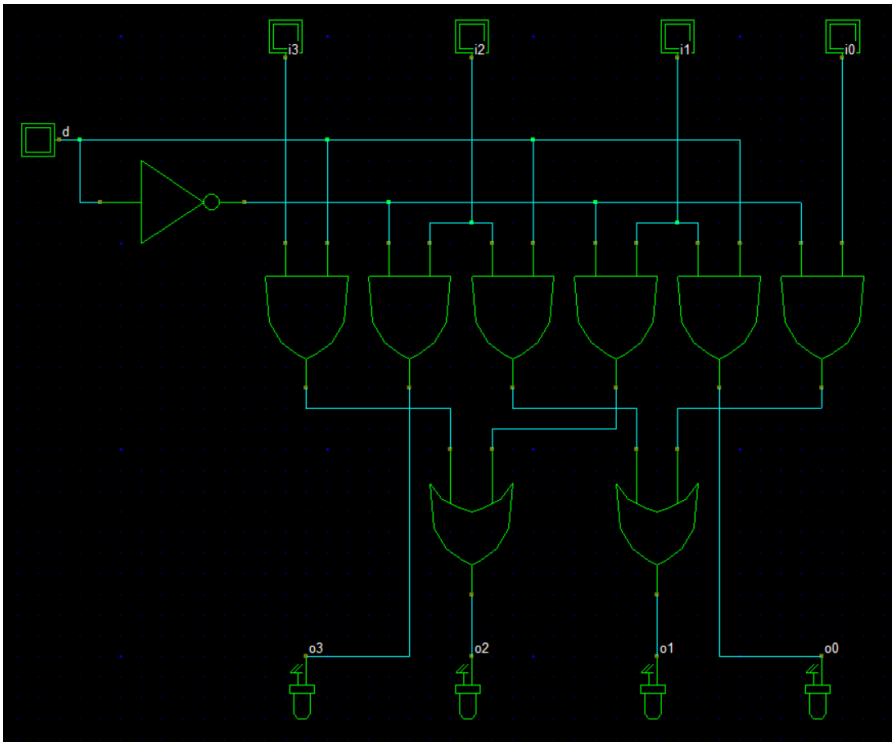
e. Implementarea în Microwind



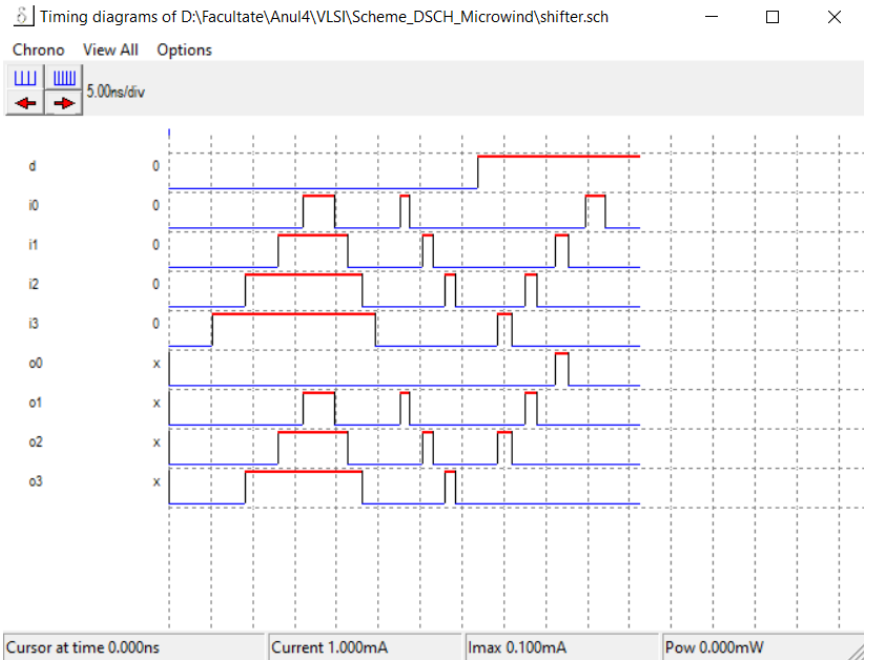
f. Simularea obținută în Microwind



9. Registru deplasare stânga / dreapta 4 biți
   a. Schema circuitului (porți logice)
      Am preluat schema circuitului de la adresa:
      https://electronicspost.com/the-shift-register/
   b. Implementarea în DSCH



c. Simularea în DSCH



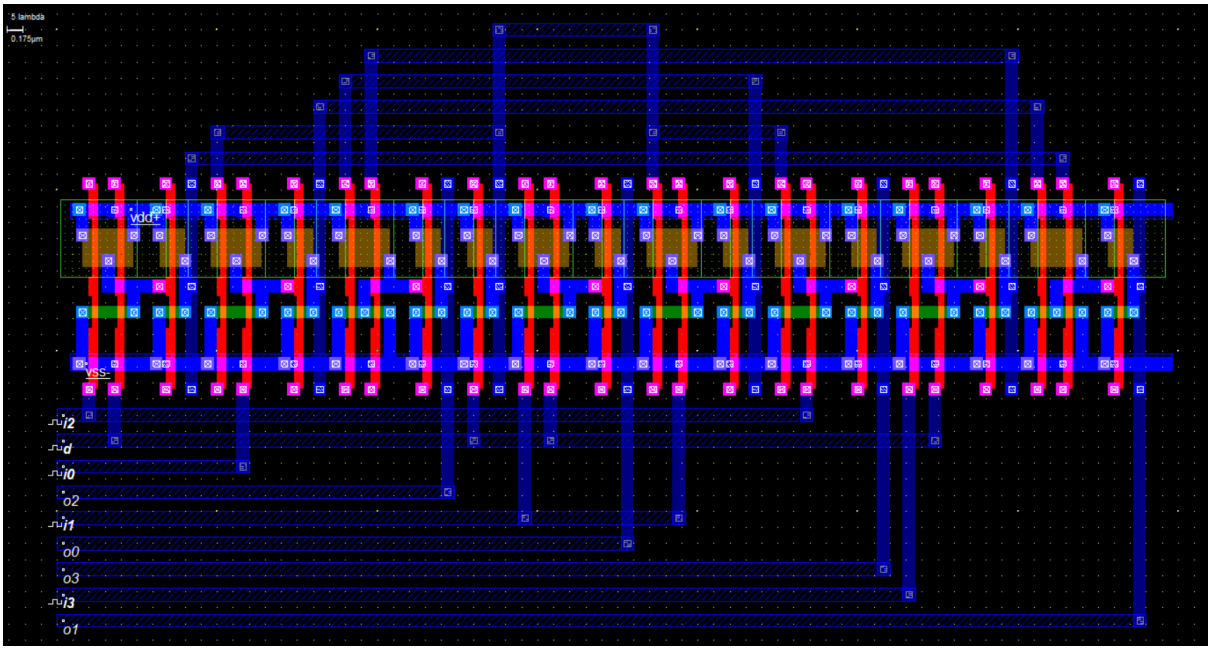
d. codul Verilog obținut

Verilog, Hierarchy and Netlist

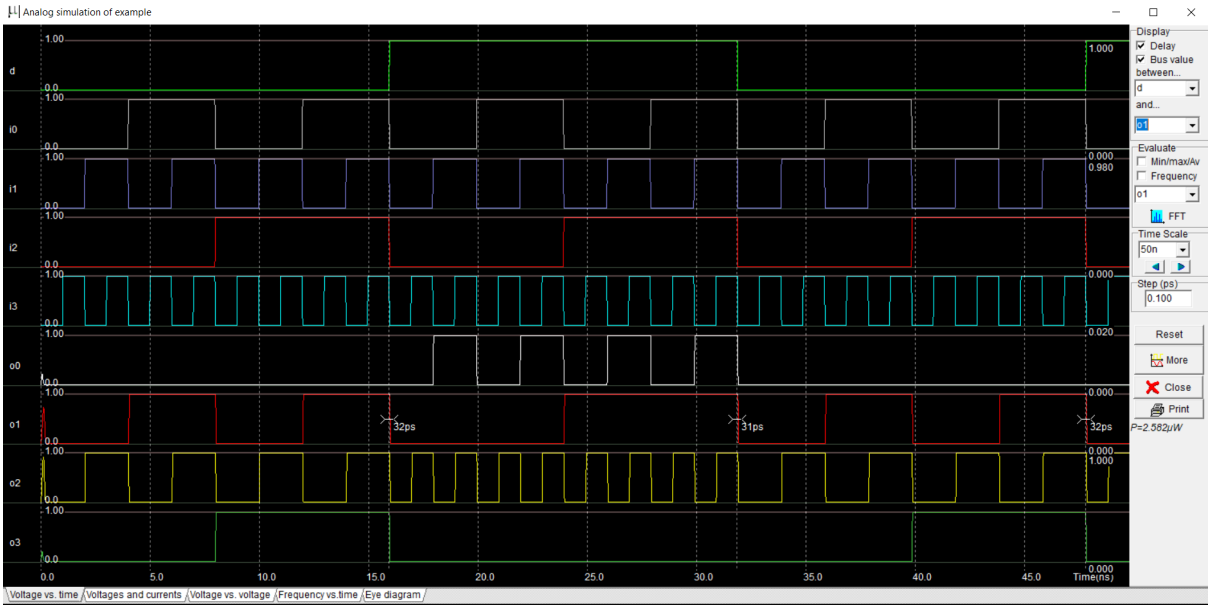
Verilog | Hierarchy | Netlist | Critical path

```
// DSCH 3.5
// 5/12/2023 12:49:53 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH_Microwind\shifter.sch

module shifter( i3,il,i0,i2,d,o3,o0,ol,
 o2);
 input i3,il,i0,i2,d;
 output o3,o0,ol,o2;
 wire w4,w6,w8,w9,wl0,;
 and #(3) and2_1(w4,i2,d);
 and #(3) and2_2(w8,w6,i0);
 or #(3) or2_3(o2,w9,wl0);
 not #(2) inv_4(w6,d);
 and #(3) and2_5(o0,il,d);
 and #(3) and2_6(w9,w6,il);
 and #(3) and2_7(o3,w6,i2);
 and #(3) and2_8(wl0,i3,d);
 or #(3) or2_9(ol,w8,w4);
endmodule

// Simulation parameters in Verilog Format
always
#200 i3=~i3;
#400 il=~il;
#800 i0=~i0;
#1600 i2=~i2;
#3200 d=~d;

// Simulation parameters
// i3 CLK 1 1
// il CLK 2 2
// i0 CLK 4 4
// i2 CLK 8 8
// d CLK 16 16
```

e. Implementarea în Microwind



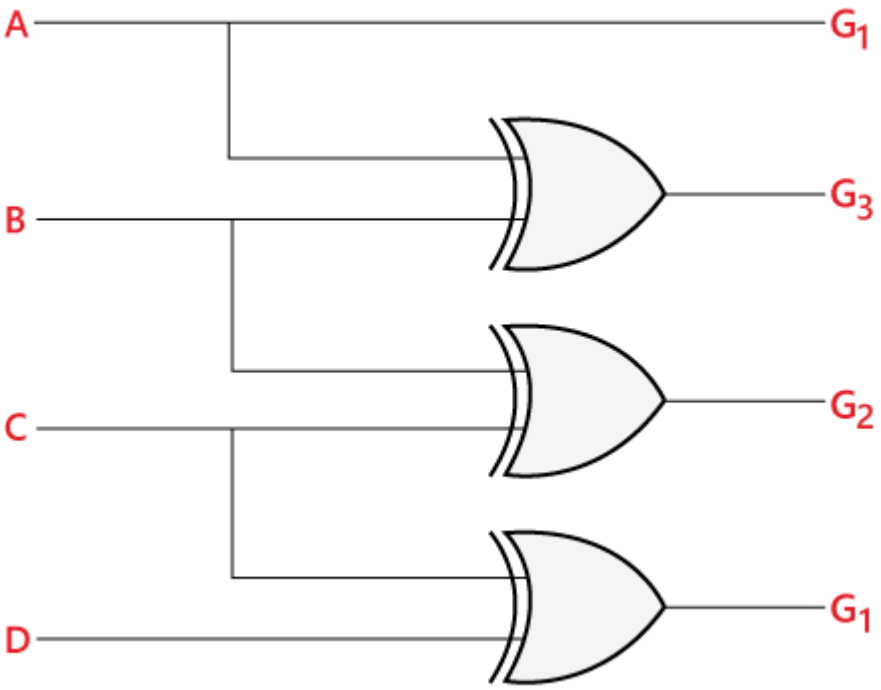
f. Simularea obținută în Microwind



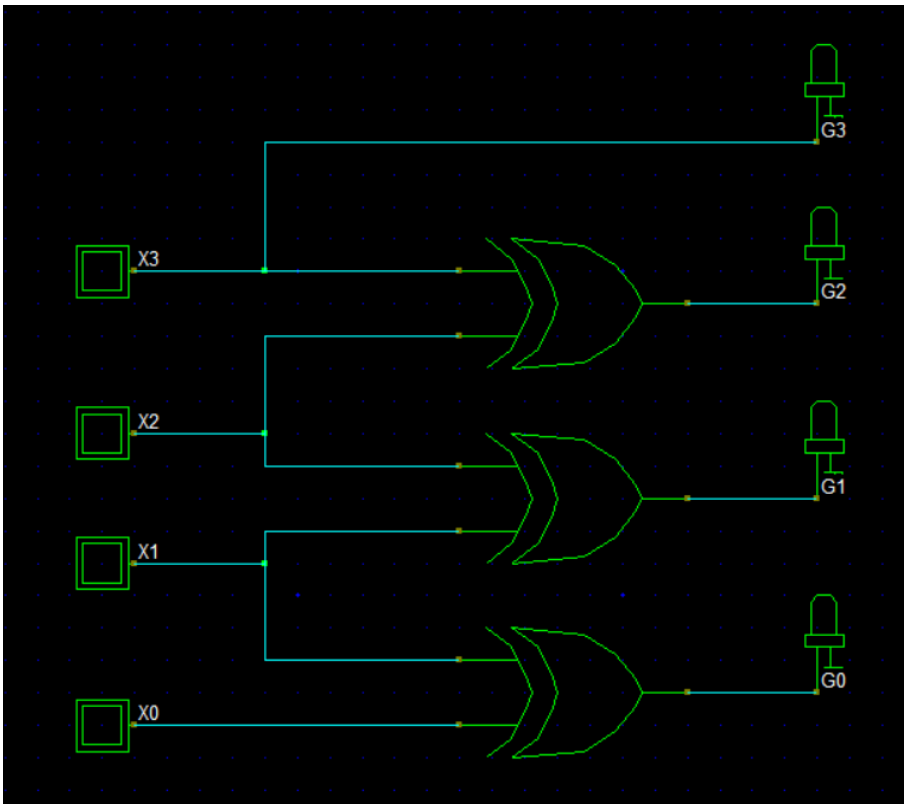
10. Convertor cod Gray 4 biți

a. Schema circuitului (porți logice)

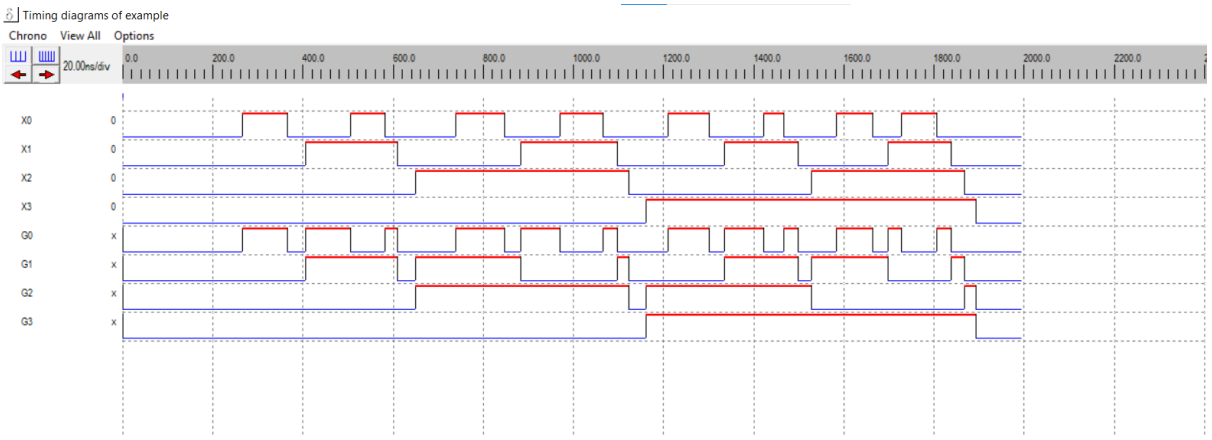
Am preluat schema circuitului de la adresa următoare:
https://www.javatpoint.com/binary-to-gray-code-cconversion-in-digital-electronics



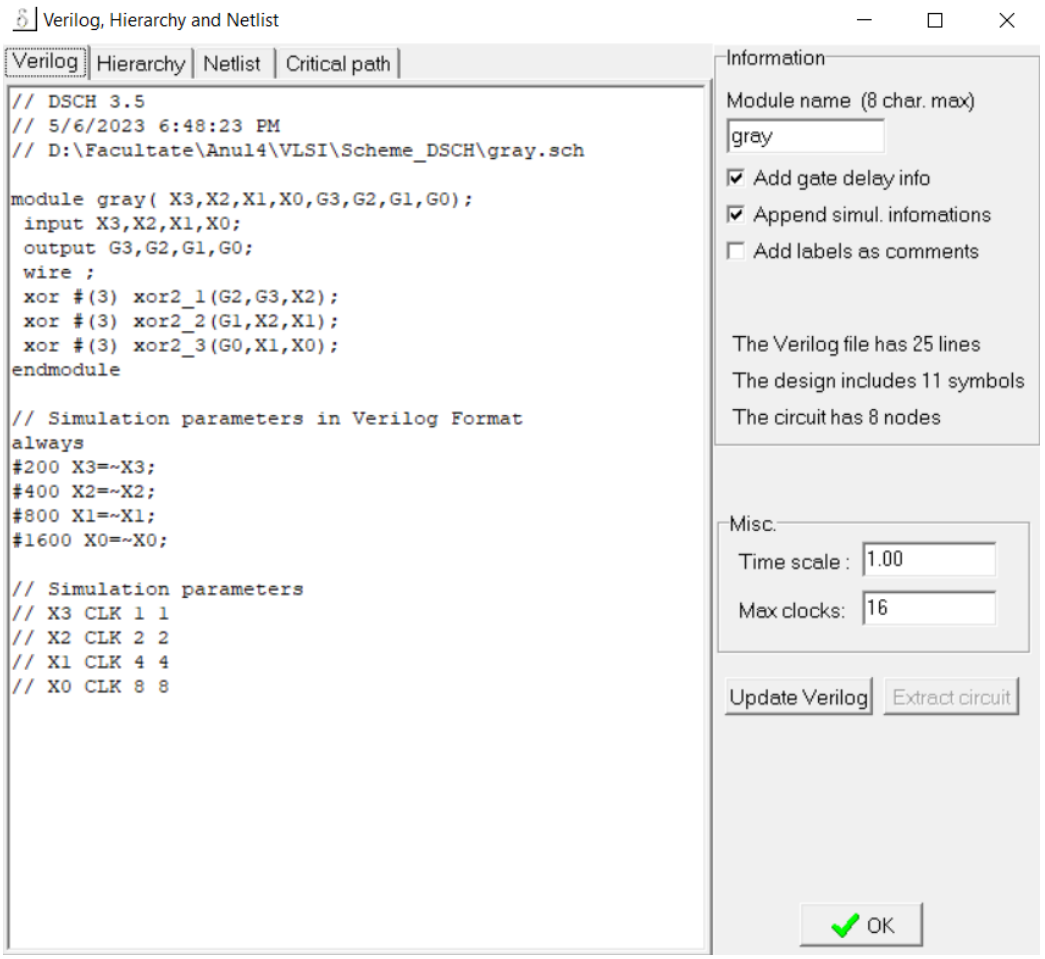
b. Implementarea în DSCH



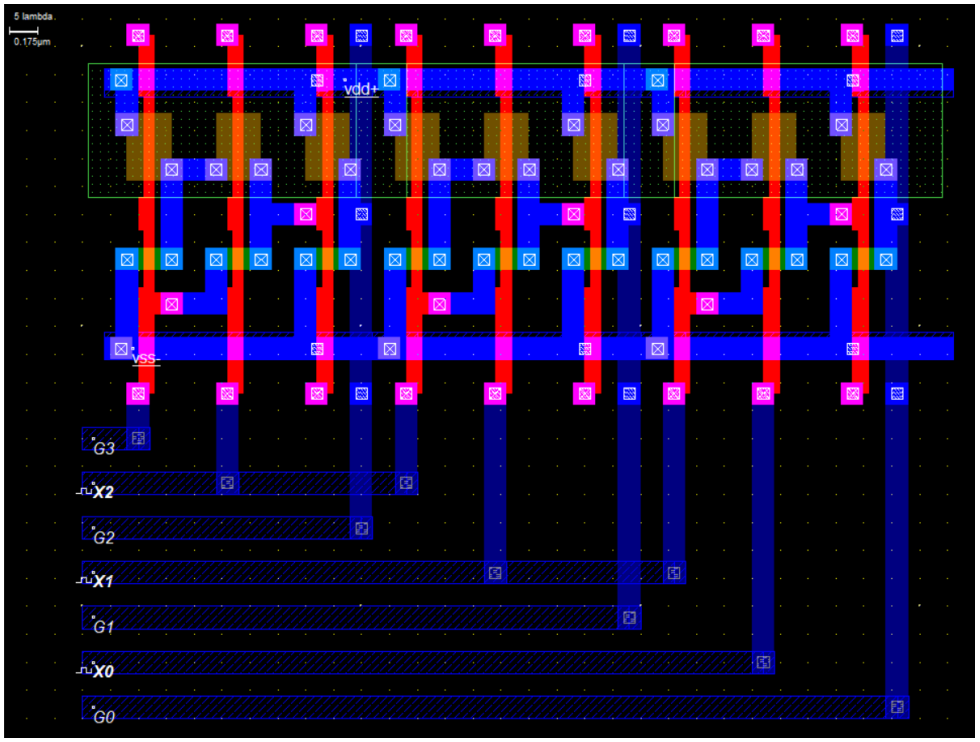
c. Simularea în DSCH



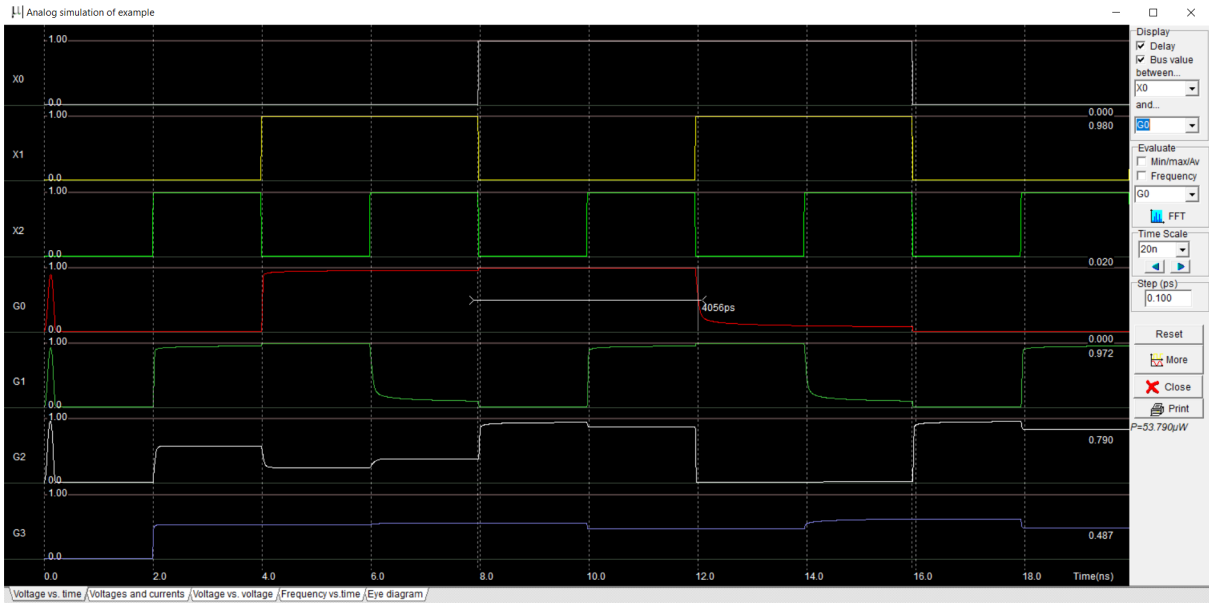
d. codul Verilog obținut



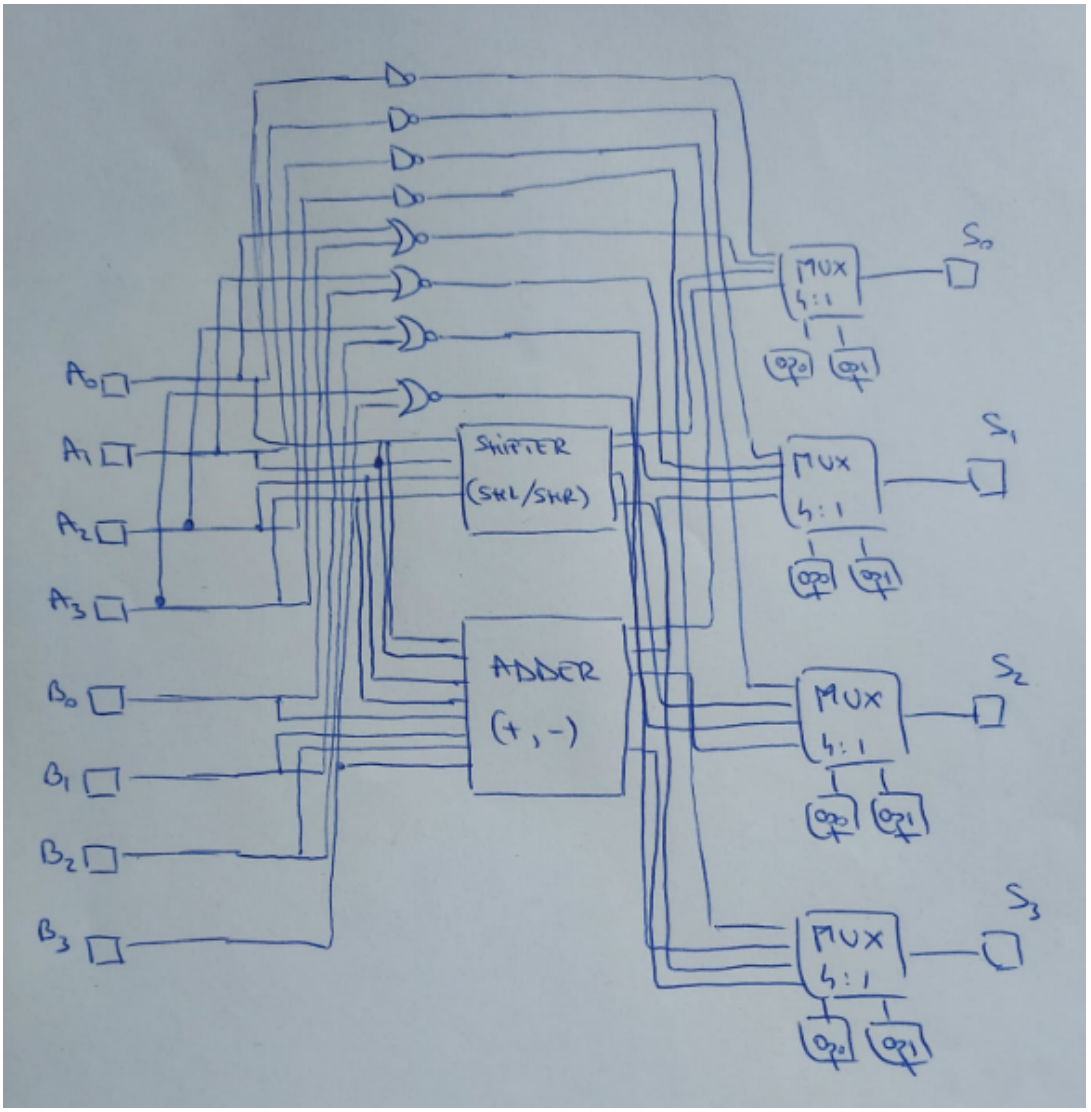
e. Implementarea în Microwind



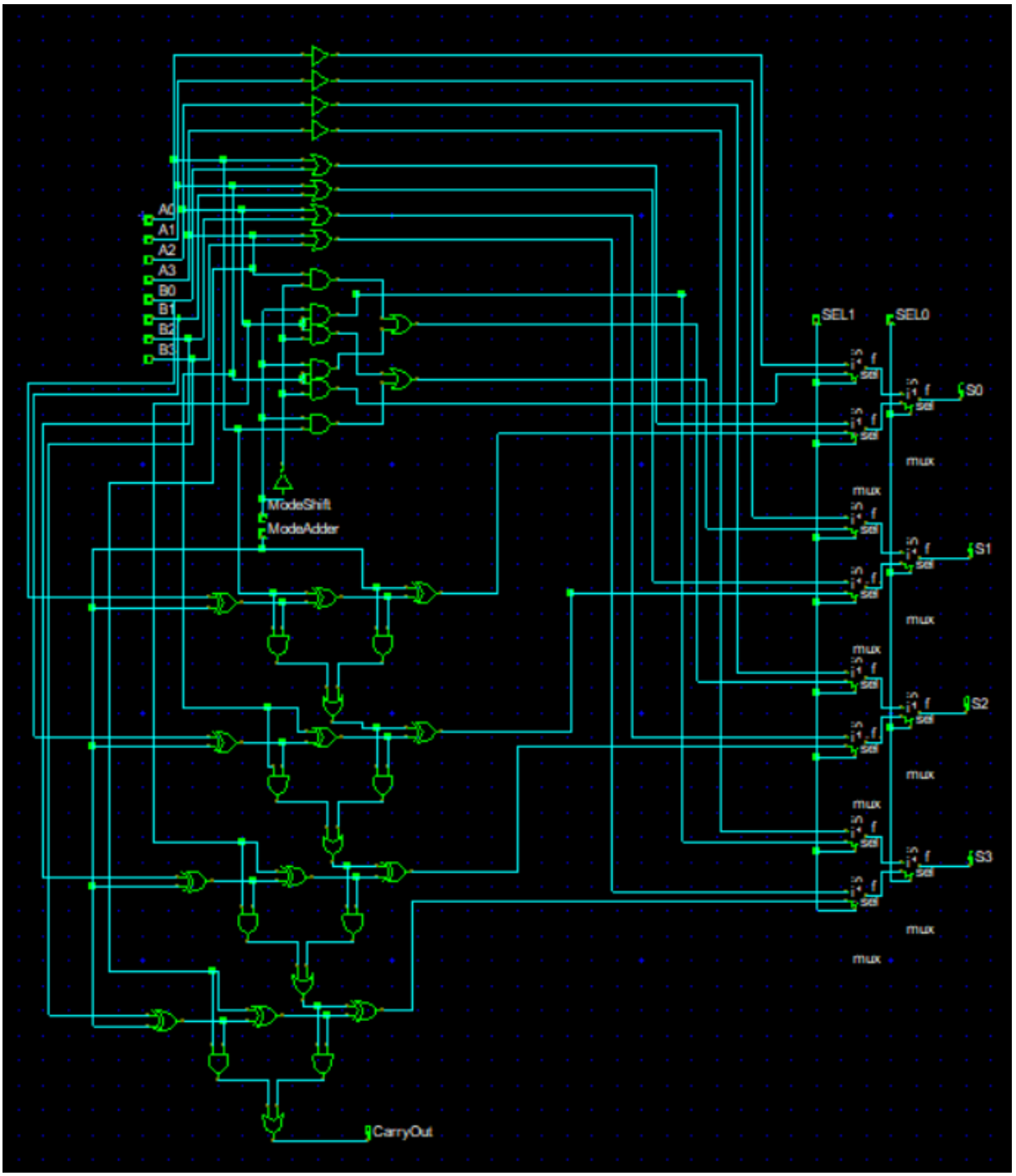
f. Simularea obținută în Microwind



11. ALU pe 4 biți (not,+,-,nor,shr,shl)
    a. Schema circuitului (porți logice)
       Link inspirație:
       https://www.geeksforgeeks.org/full-adder-in-digital-logic/
       https://esim.fossee.in/circuit-simulation-project/download/project-file/267
       http://www.csc.villanova.edu/~mdamian/Past/csc2400fa13/assign/ALU.html
       Am folosit SEL1 și SEL0 pentru a alege operația făcută:
       - SEL1 = 0, SEL0 = 0 - not (se face doar pe A)
       - SEL1 = 0, SEL0 = 1 - nor
       - SEL1 = 1, SEL0 = 0 - shiftare (se alege tipul din ModeShift)
       - SEL1 = 1, SEL0 = 1 - +/- (se alege tipul din ModeAdder)

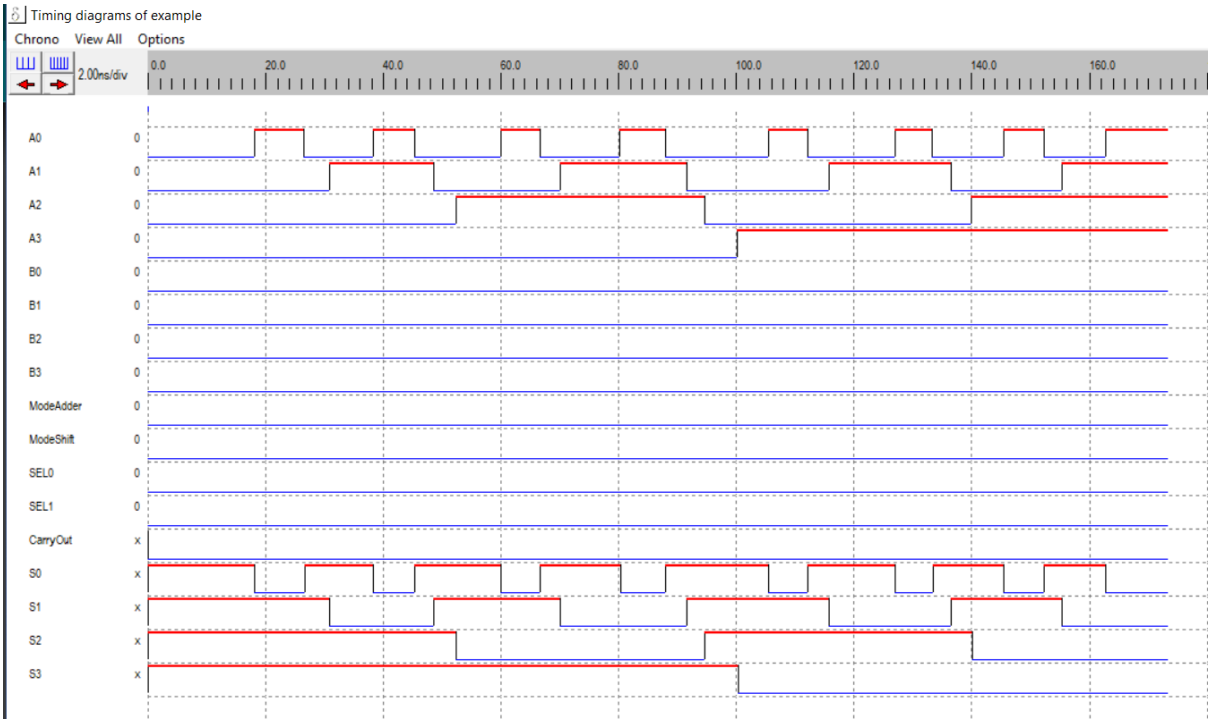


b. Implementarea în DSCH

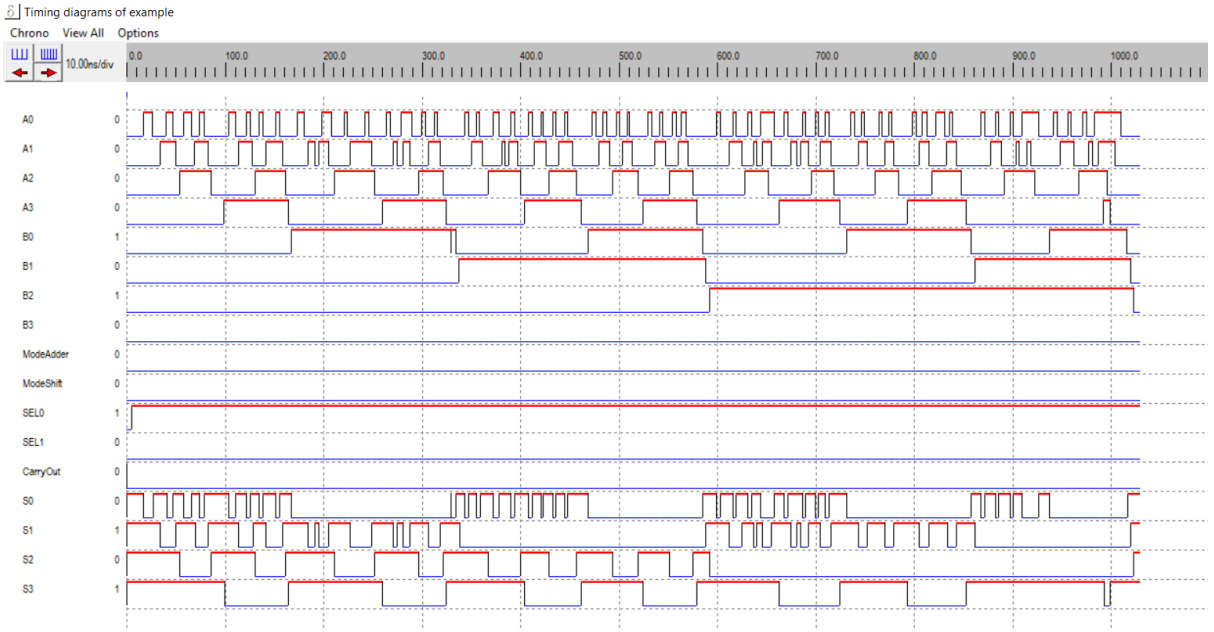


c. Simularea în DSCH

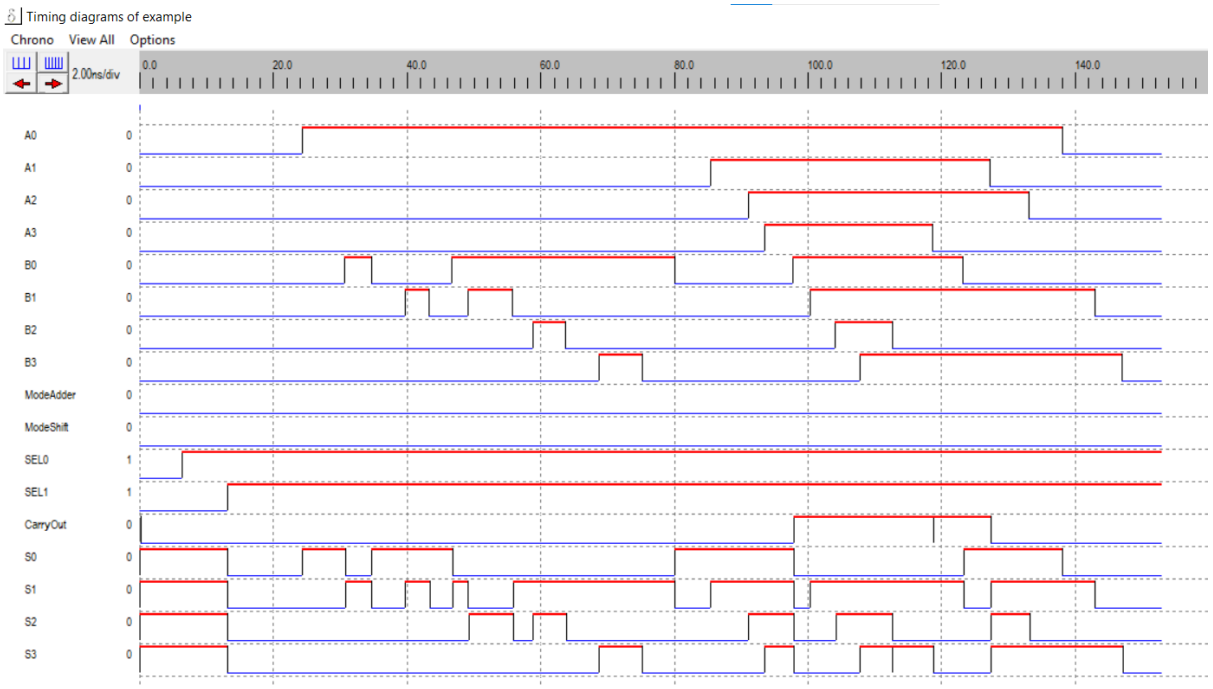
i. not



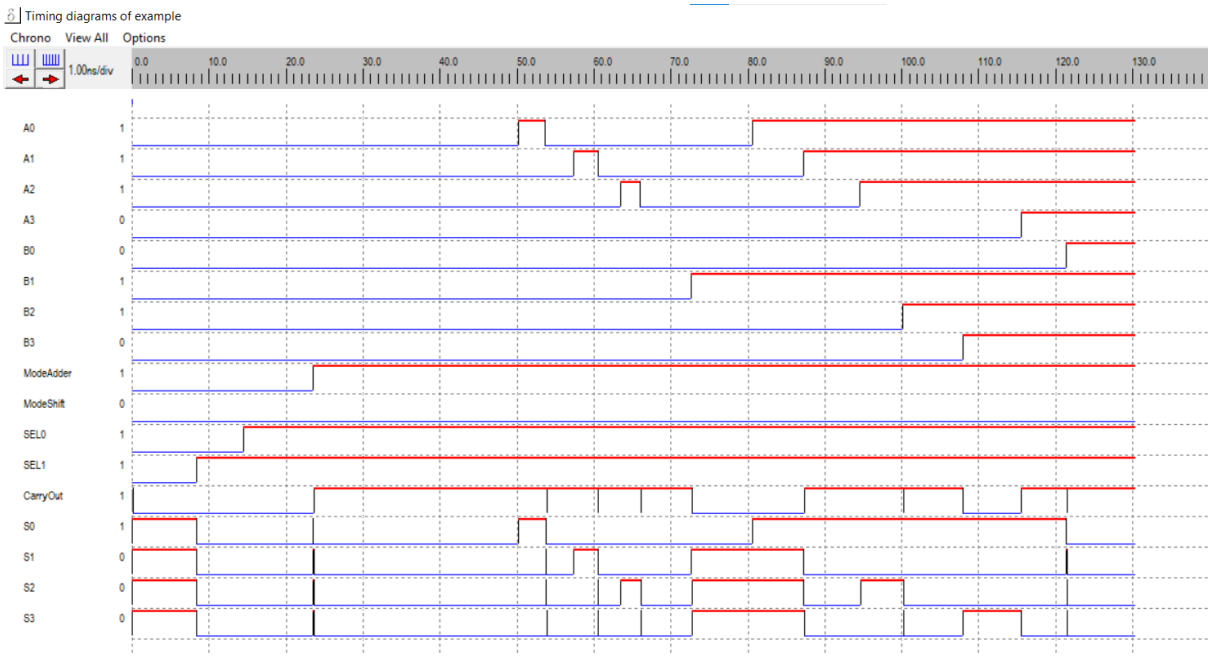
ii. nor



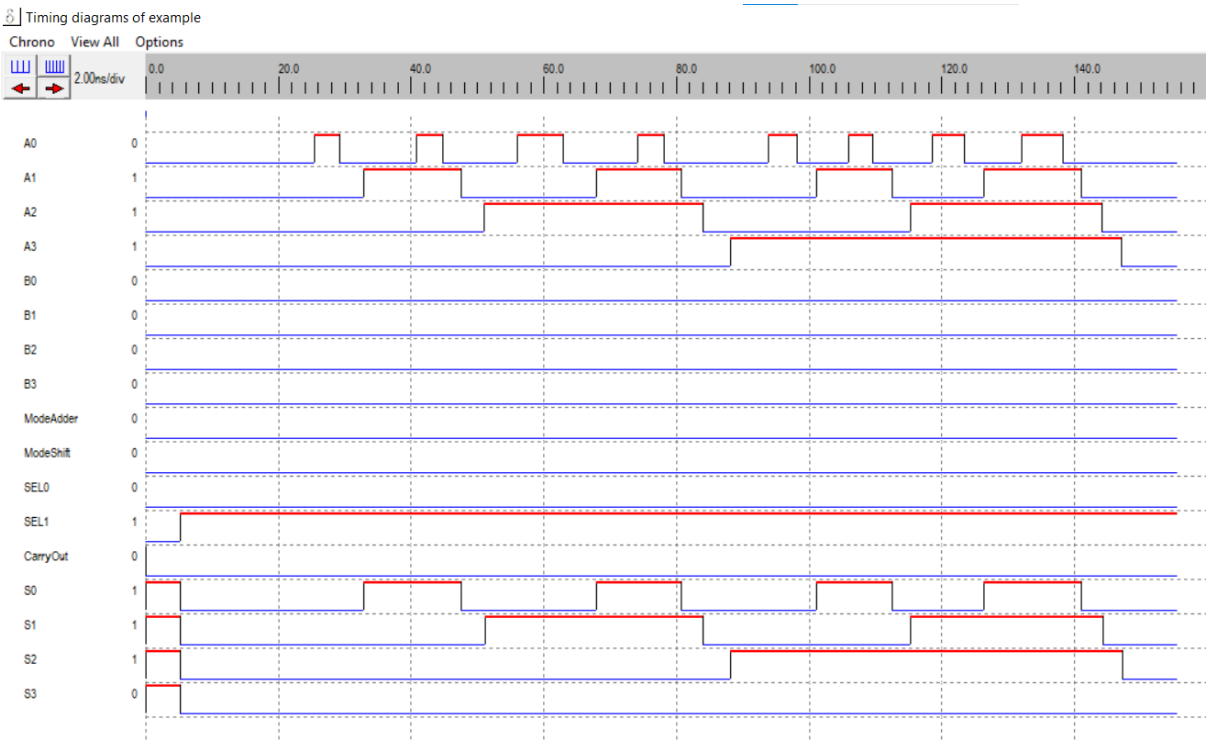
iii. adder(+)



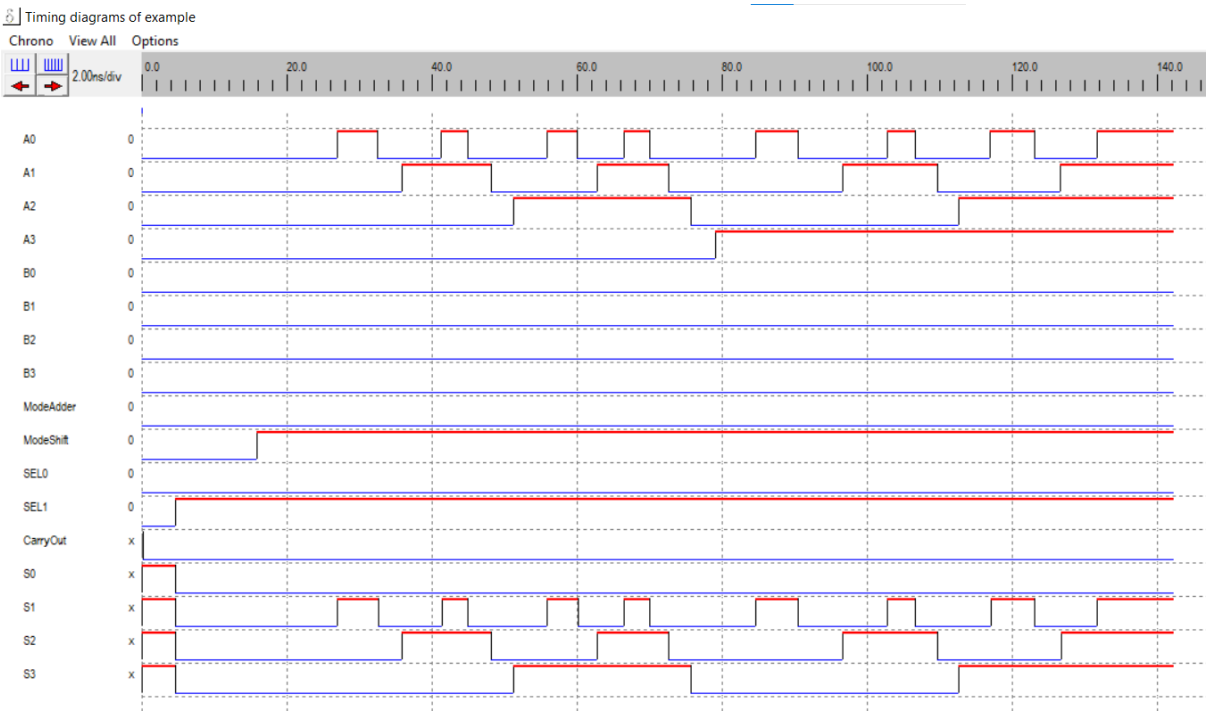
iv. adder(-)



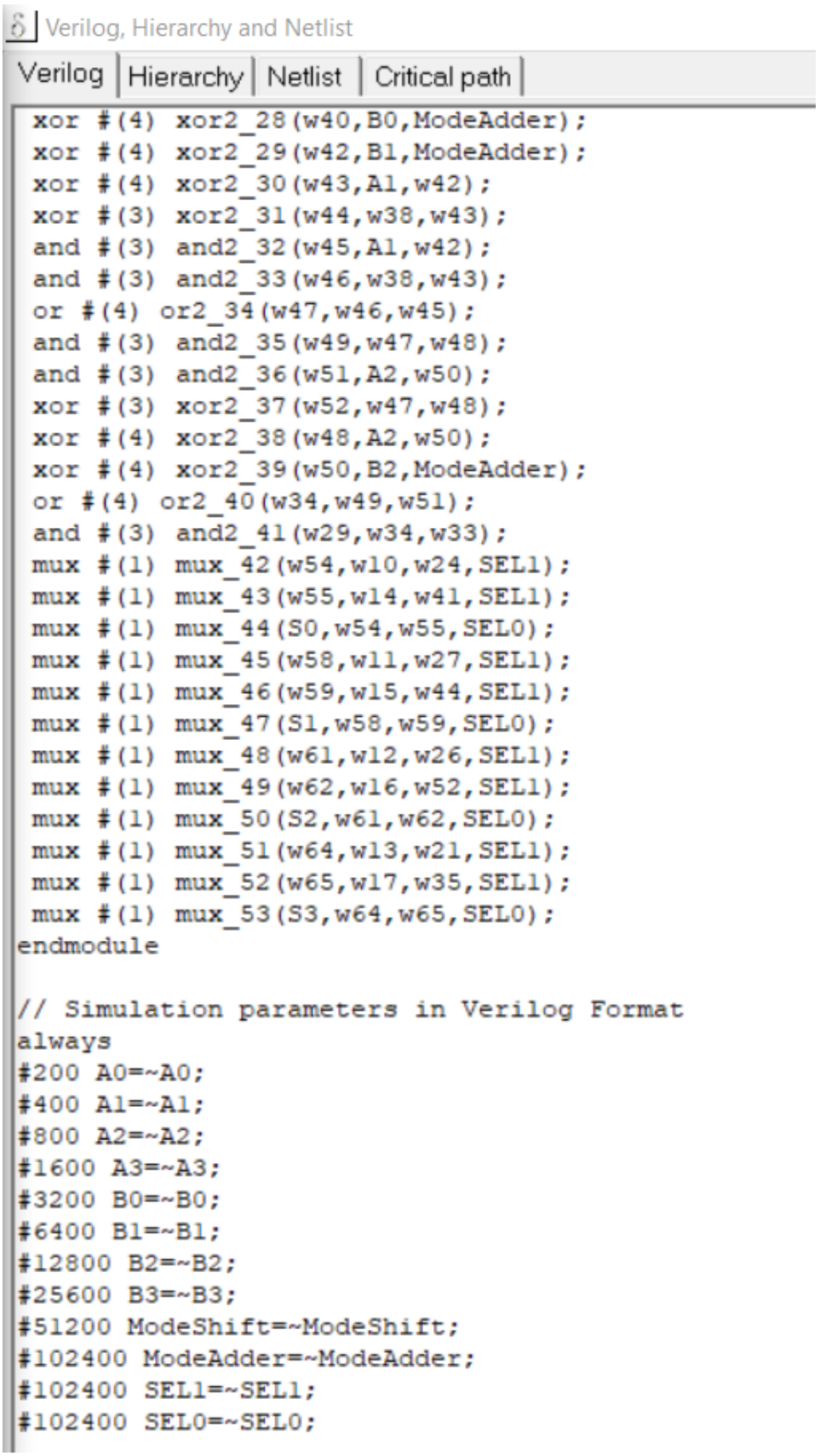
v. shl


Timing diagrams of example
Chrono View All Options
2.00ns/div
0.0 20.0 40.0 60.0 80.0 100.0 120.0 140.0
A0 0
A1 1
A2 1
A3 1
B0 0
B1 0
B2 0
B3 0
ModeAdder 0
ModeShift 0
SEL0 0
SEL1 1
CarryOut 0
S0 1
S1 1
S2 1
S3 0

vi. shr


Timing diagrams of example
Chrono View All Options
2.00ns/div
0.0 20.0 40.0 60.0 80.0 100.0 120.0 140.0
A0 0
A1 0
A2 0
A3 0
B0 0
B1 0
B2 0
B3 0
ModeAdder 0
ModeShift 0
SEL0 0
SEL1 0
CarryOut x
S0 x
S1 x
S2 x
S3 x

d. codul Verilog obținut




```
 xor #(4) xor2_28(w40,B0,ModeAdder);
 xor #(4) xor2_29(w42,B1,ModeAdder);
 xor #(4) xor2_30(w43,A1,w42);
 xor #(3) xor2_31(w44,w38,w43);
 and #(3) and2_32(w45,A1,w42);
 and #(3) and2_33(w46,w38,w43);
 or #(4) or2_34(w47,w46,w45);
 and #(3) and2_35(w49,w47,w48);
 and #(3) and2_36(w51,A2,w50);
 xor #(3) xor2_37(w52,w47,w48);
 xor #(4) xor2_38(w48,A2,w50);
 xor #(4) xor2_39(w50,B2,ModeAdder);
 or #(4) or2_40(w34,w49,w51);
 and #(3) and2_41(w29,w34,w33);
 mux #(1) mux_42(w54,w10,w24,SEL1);
 mux #(1) mux_43(w55,w14,w41,SEL1);
 mux #(1) mux_44(S0,w54,w55,SEL0);
 mux #(1) mux_45(w58,w11,w27,SEL1);
 mux #(1) mux_46(w59,w15,w44,SEL1);
 mux #(1) mux_47(S1,w58,w59,SEL0);
 mux #(1) mux_48(w61,w12,w26,SEL1);
 mux #(1) mux_49(w62,w16,w52,SEL1);
 mux #(1) mux_50(S2,w61,w62,SEL0);
 mux #(1) mux_51(w64,w13,w21,SEL1);
 mux #(1) mux_52(w65,w17,w35,SEL1);
 mux #(1) mux_53(S3,w64,w65,SEL0);
endmodule

// Simulation parameters in Verilog Format
always
#200 A0=~A0;
#400 A1=~A1;
#800 A2=~A2;
#1600 A3=~A3;
#3200 B0=~B0;
#6400 B1=~B1;
#12800 B2=~B2;
#25600 B3=~B3;
#51200 ModeShift=~ModeShift;
#102400 ModeAdder=~ModeAdder;
#102400 SEL1=~SEL1;
#102400 SEL0=~SEL0;
```

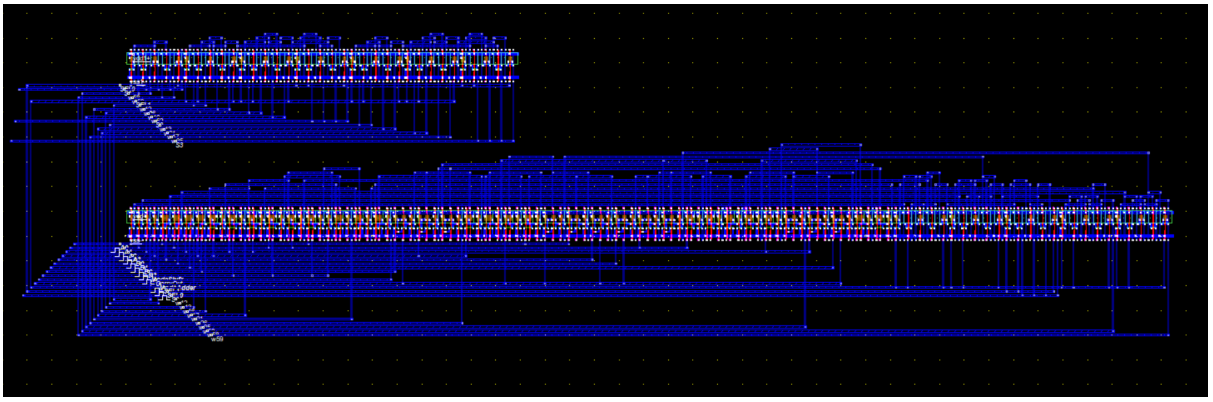
8 Verilog, Hierarchy and Netlist

Verilog | Hierarchy | Netlist | Critical path

```
// DSCH 3.5
// 5/8/2023 3:11:18 PM
// D:\Facultate\Anul4\VLSI\Scheme_DSCH\alu.sch

module alu( A0,A1,A2,A3,B0,B1,B2,B3,
 ModeShift,ModeAdder,SEL1,SEL0,CarryOut,S0,S1,S2,
 S3);
 input A0,A1,A2,A3,B0,B1,B2,B3;
 input ModeShift,ModeAdder,SEL1,SEL0;
 output CarryOut,S0,S1,S2,S3;
 wire w10,w11,w12,w13,w14,w15,w16,w17;
 wire w18,w19,w21,w22,w23,w24,w25,w26;
 wire w27,w29,w30,w32,w33,w34,w35,w36;
 wire w37,w38,w39,w40,w41,w42,w43,w44;
 wire w45,w46,w47,w48,w49,w50,w51,w52;
 wire w54,w55,w58,w59,w61,w62,w64,w65;
 wire ;
 not #(1) inv_1(w10,A0);
 not #(1) inv_2(w11,A1);
 not #(1) inv_3(w12,A2);
 not #(1) inv_4(w13,A3);
 nor #(3) nor2_5(w14,A0,B0);
 nor #(3) nor2_6(w15,A1,B1);
 nor #(3) nor2_7(w16,A2,B2);
 nor #(3) nor2_8(w17,A3,B3);
 and #(3) and2_9(w19,w18,A3);
 and #(3) and2_10(w21,A2,ModeShift);
 and #(3) and2_11(w22,w18,A2);
 and #(3) and2_12(w23,A1,ModeShift);
 and #(3) and2_13(w24,w18,A1);
 and #(3) and2_14(w25,A0,ModeShift);
 not #(2) inv_15(w18,ModeShift);
 or #(3) or2_16(w26,w19,w23);
 or #(3) or2_17(w27,w22,w25);
 or #(3) or2_18(CarryOut,w29,w30);
 xor #(4) xor2_19(w32,B3,ModeAdder);
 xor #(4) xor2_20(w33,A3,w32);
 xor #(3) xor2_21(w35,w34,w33);
 and #(3) and2_22(w30,A3,w32);
 or #(4) or2_23(w38,w36,w37);
 and #(3) and2_24(w36,ModeAdder,w39);
 and #(3) and2_25(w37,A0,w40);
 xor #(3) xor2_26(w41,ModeAdder,w39);
 xor #(4) xor2_27(w39,A0,w40);
 xor #(4) xor2_28(w40,B0,ModeAdder);
 xor #(4) xor2_29(w42,B1,ModeAdder);
```

```
// Simulation parameters
// A0 CLK 1 1
// A1 CLK 2 2
// A2 CLK 4 4
// A3 CLK 8 8
// B0 CLK 16 16
// B1 CLK 32 32
// B2 CLK 64 64
// B3 CLK 128 128
// ModeShift CLK 256 256
// ModeAdder CLK 512 512
// SEL1 CLK 1024 1024
// SEL0 CLK 2048 2048
```

e. Implementarea în Microwind



f. Simularea obținută în Microwind