Hacettepe University

Department of Electrical and Electronics Engineering

Ele419

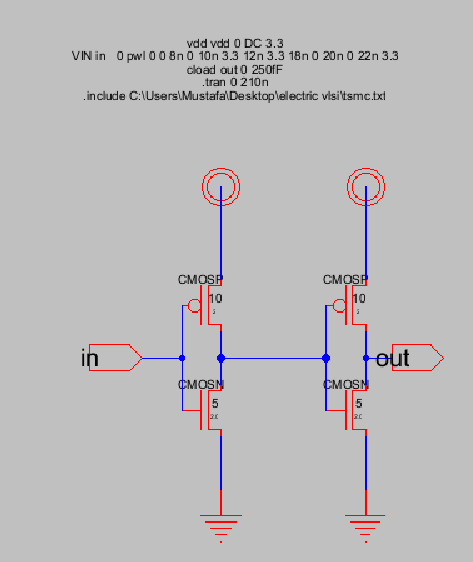
Term Project Report

M.Mustafa KOYUNCU

21986750

**Transistor Level Function Implementations:**

**1-) Transfer A (F=A):**

****

Input A inverted two times so that input can be transfered to output. F=(A’)’

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Input wave for transfer function.

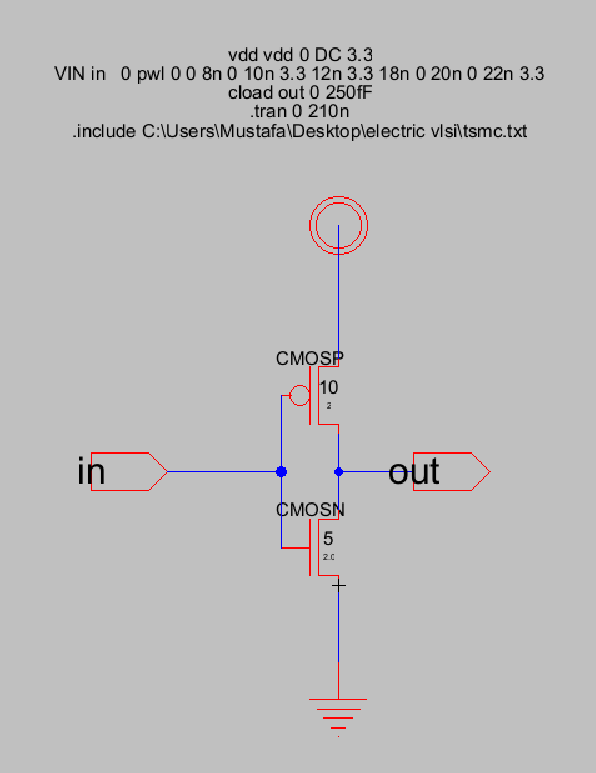
metin, ekran görüntüsü, elektronik eşyalar, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

Output wave for transfer function.

**2-) Not Function (F=A’):**

This function consist of just 1 inverter.



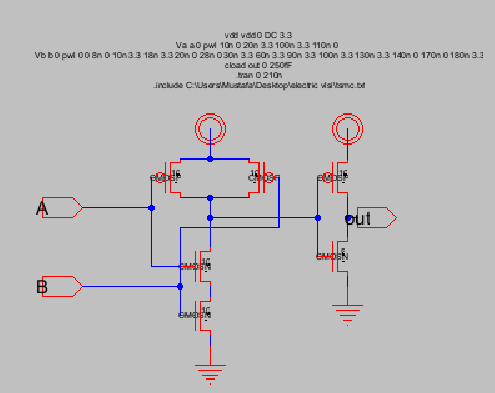
metin, elektronik eşyalar, vitrin, ekran görüntüsü içeren bir resim

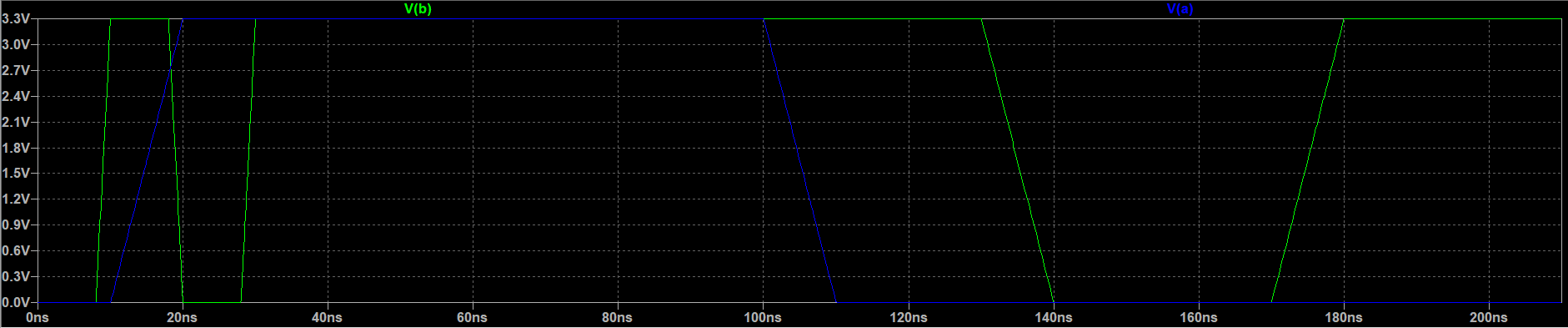
Açıklama otomatik olarak oluşturuldu

Input (blue ) and output (green) waves for inverter function.

**3-) And function (F= A and B):**

To use least number of transistor, it is more logical to invert the output of a NAND function.Otherwise we would need extra inverters for each input.This is an important gate because it is used in half adder which will be used in alu design and further functions.





Input wave for and function.

metin, elektronik eşyalar, bilgisayar, ekran görüntüsü içeren bir resim

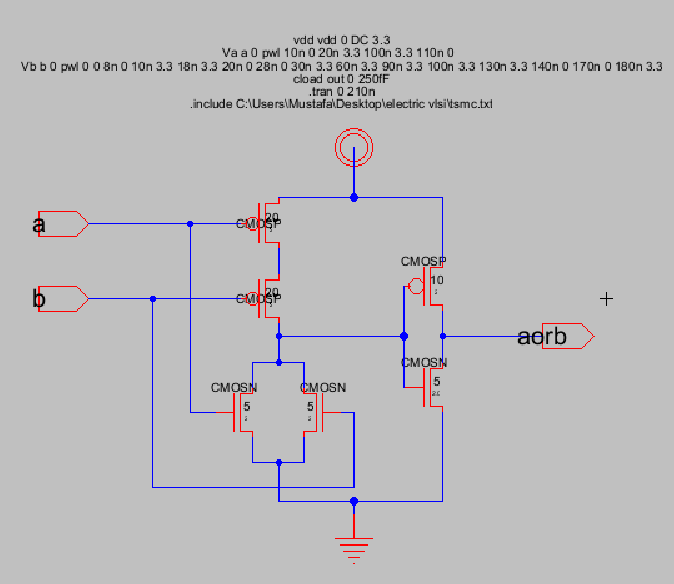
Açıklama otomatik olarak oluşturuldu

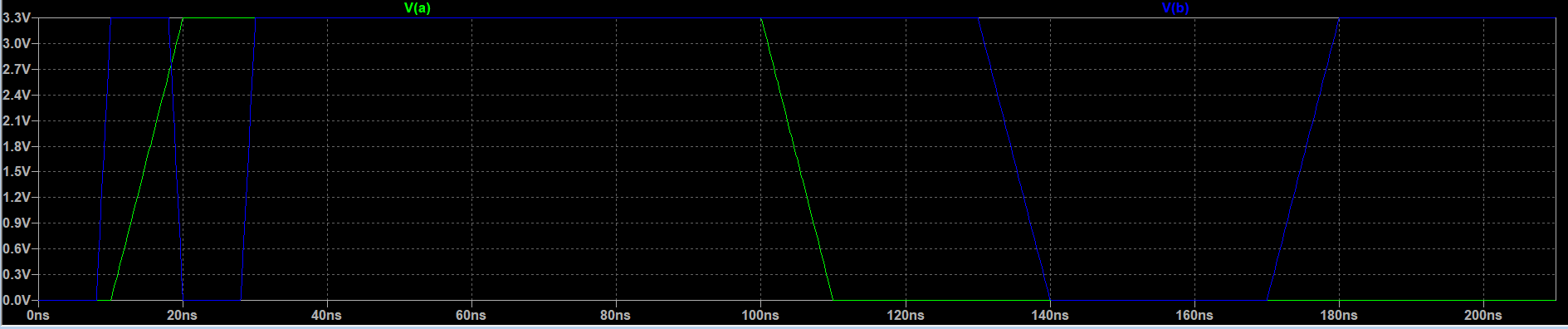
Output wave for and function.

As you can see , wave is 1 only when both inputs are 1.

**4-) Or Function (F=AorB):**

Again , to use least number of transistor it is more logical to invert the output of nor function.

****

****

Input wave for or function.

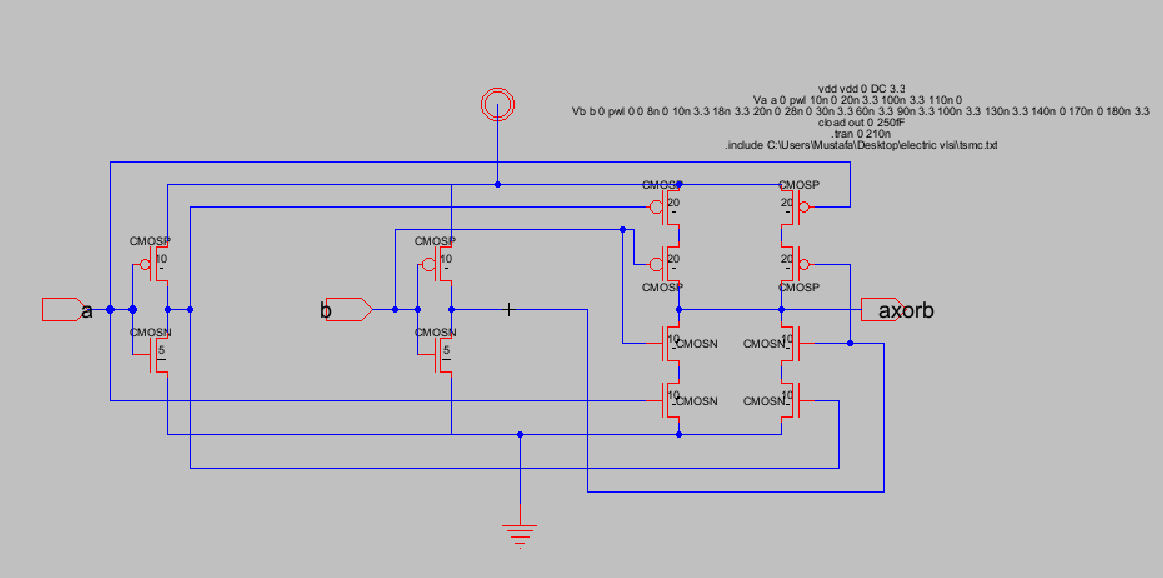
metin, elektronik eşyalar, vitrin, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

Output wave for or function.

As you can see wave is 0 only at ‘both inputs are 0 ’ case.

**5-) XOR Function (F=AxorB):**

****

This design is important because this will be used in half adder design as well as and gate.We can realize incrementer or full adder with half adder design.This function also is a odd function,means that it will give output 1 if number of 1’s in input is an odd number.

elektronik eşyalar, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

Input wave for xor function.

metin, elektronik eşyalar, bilgisayar, vitrin içeren bir resim

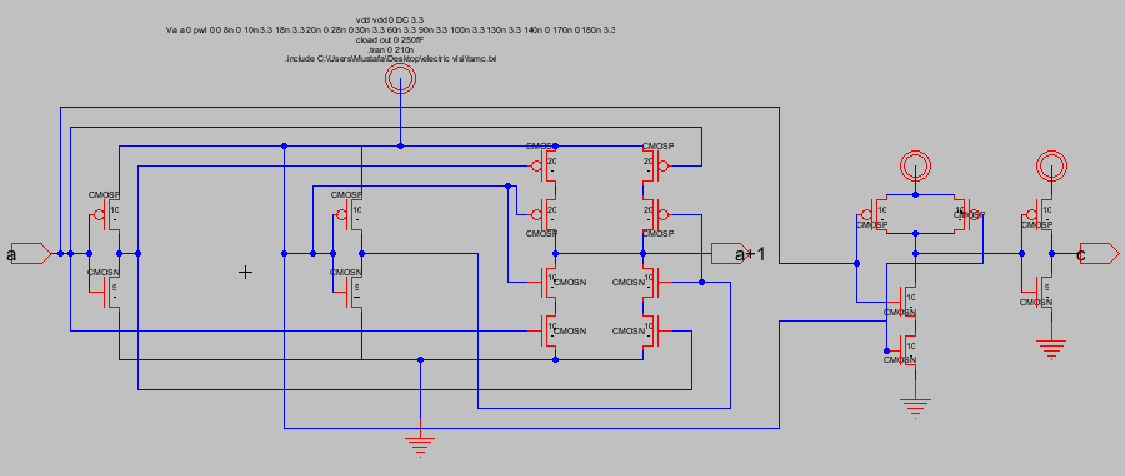
Açıklama otomatik olarak oluşturuldu

Output wave for xor function.

Output is 1 only at 1 , 0 or 0 , 1 case.

**6-) Increment A (F=A+1):**

The incrementer can be realized by using a half adder.The carry in is 0 but B input is 1 so that A will be incremented.This function will result in carry out.If we were to use multiple bit incrementer this carry out has to be other half adders B input.



****

Input wave for increment function.

metin, elektronik eşyalar içeren bir resim

Açıklama otomatik olarak oluşturuldu

Incremented A.

metin, elektronik eşyalar, bilgisayar, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

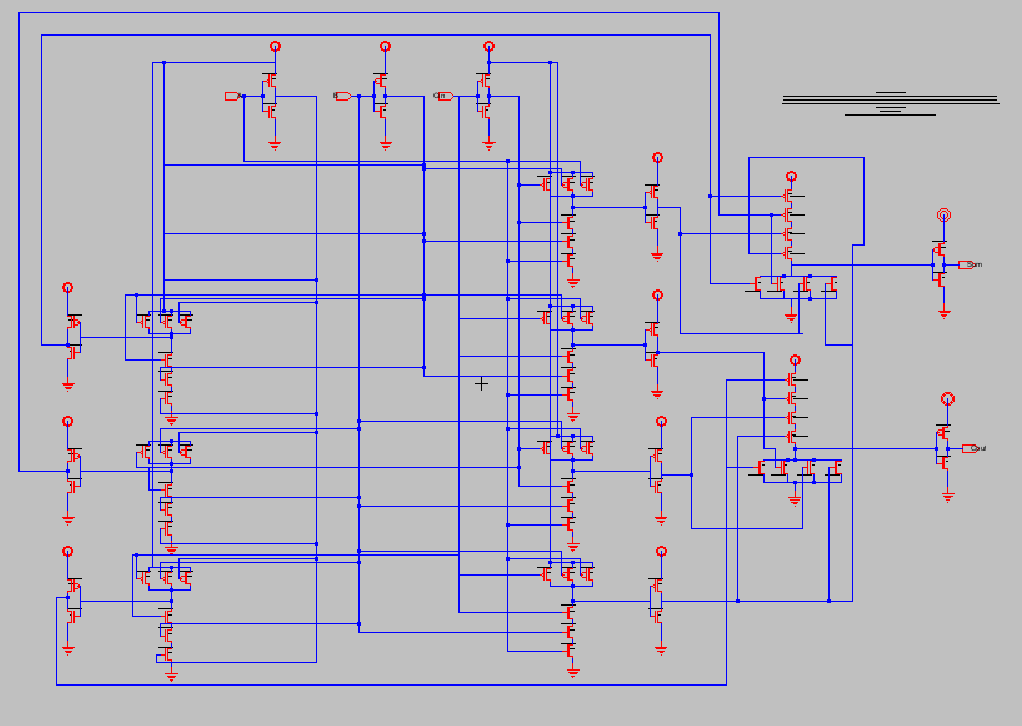
Carry out wave.

Half adder generates a carry out when A is 1 as expected.

**7-) Addition (F=A+B):**

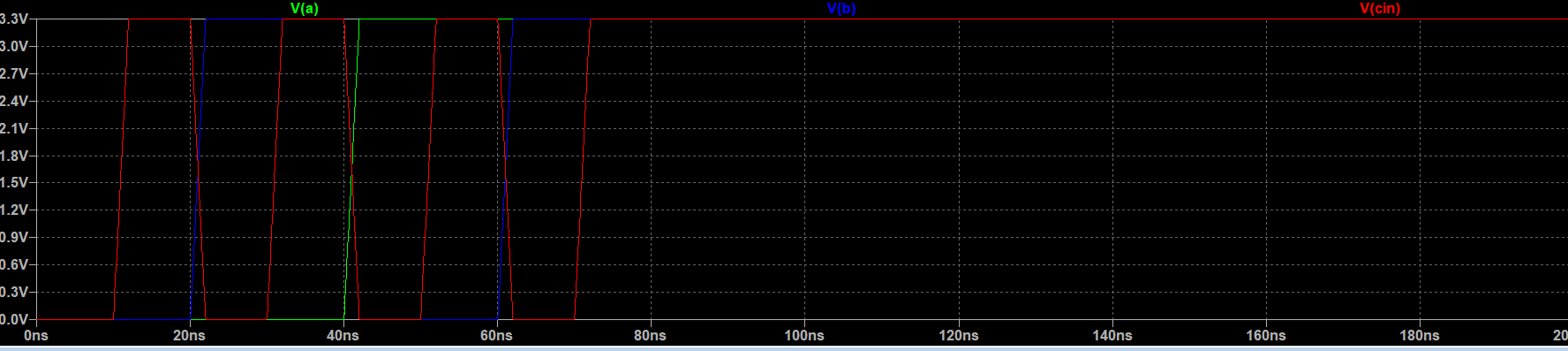
An adder can be realized with many ways.One of them is using decoder.Decoder takes combinations of A , B ,C and their not’s as input and generates q0,q1,q1…q7 as output

Summation of q1,q2,q4,q7 is equal tos um and summation of q3,q5,q6,q7 is equal to carry out.We dont need q0 so that we can reduce number of transistor.



metin içeren bir resim

Açıklama otomatik olarak oluşturuldu



Input wave for full adder.(Combination of A,B and Cin).

metin içeren bir resim

Açıklama otomatik olarak oluşturulduCarry out for full adder.

metin, iç mekan, yeşil, vitrin içeren bir resim

Açıklama otomatik olarak oluşturuldu

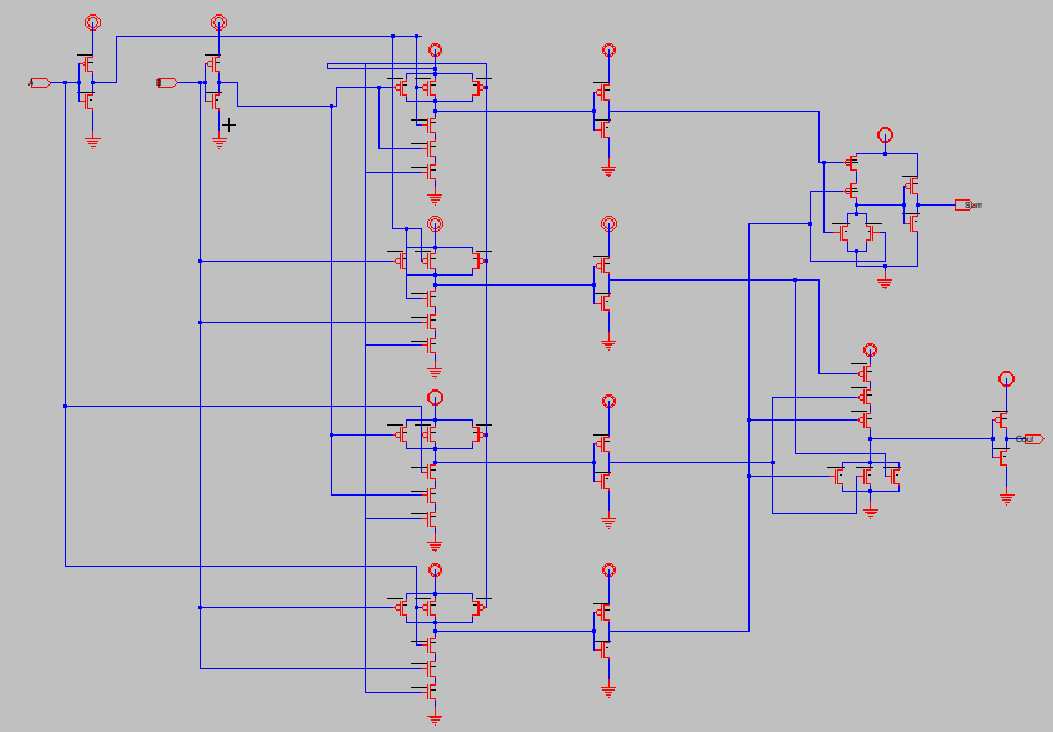
Sum for full adder.

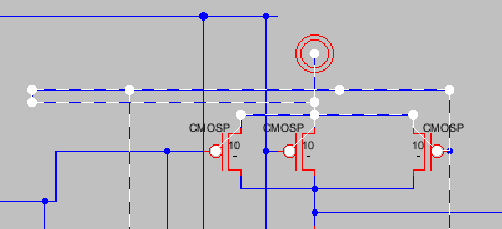
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Cin | Sum | Carry out |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

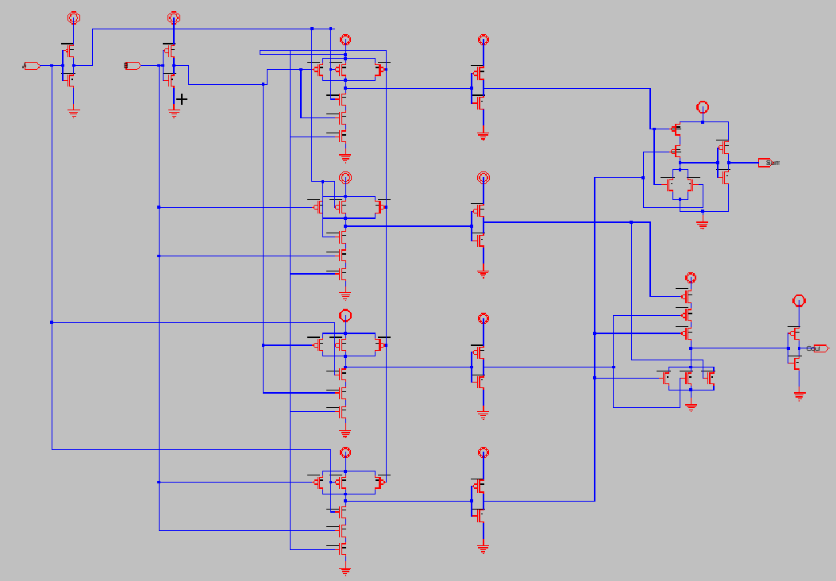
Truth table for full adder confirms the output wave for Sum and Cout.

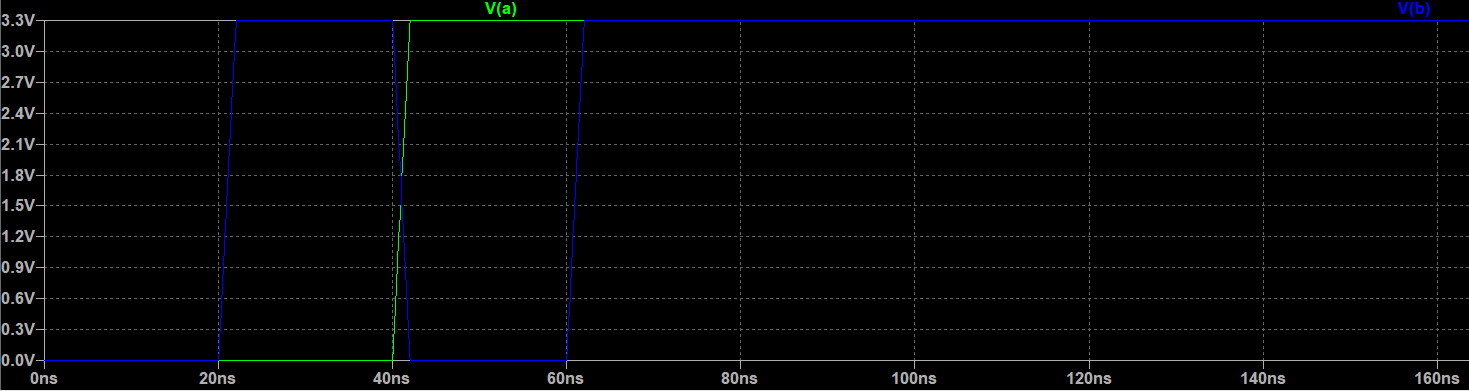
**8-)Add with Carry (F=A+B+1):**

This also can be done by using a decoder.We only need to consider cases where Cin is equal to 1 .

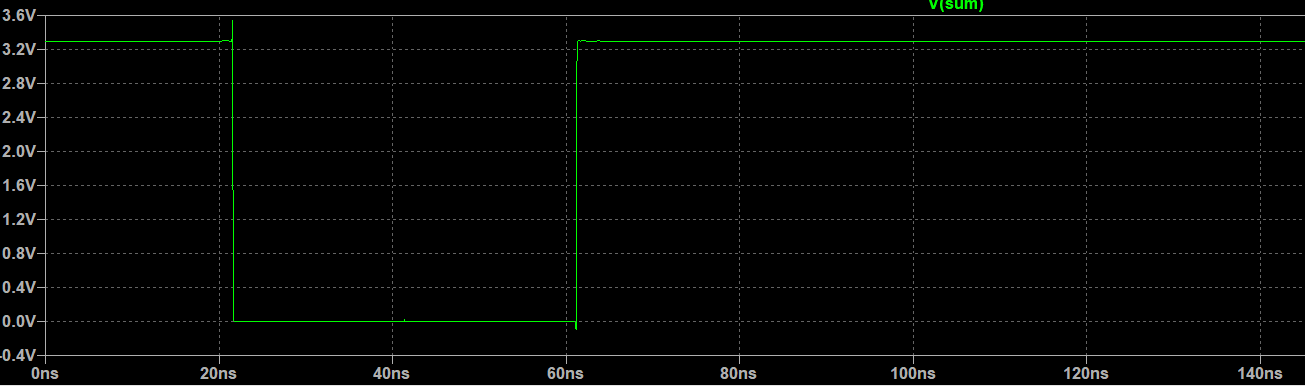




Cin is connected to Vdd provide constant 1 .



Input wave for add with carry function.



Sum for add with carry function.

metin, ekran görüntüsü, bilgisayar, vitrin içeren bir resim

Açıklama otomatik olarak oluşturuldu

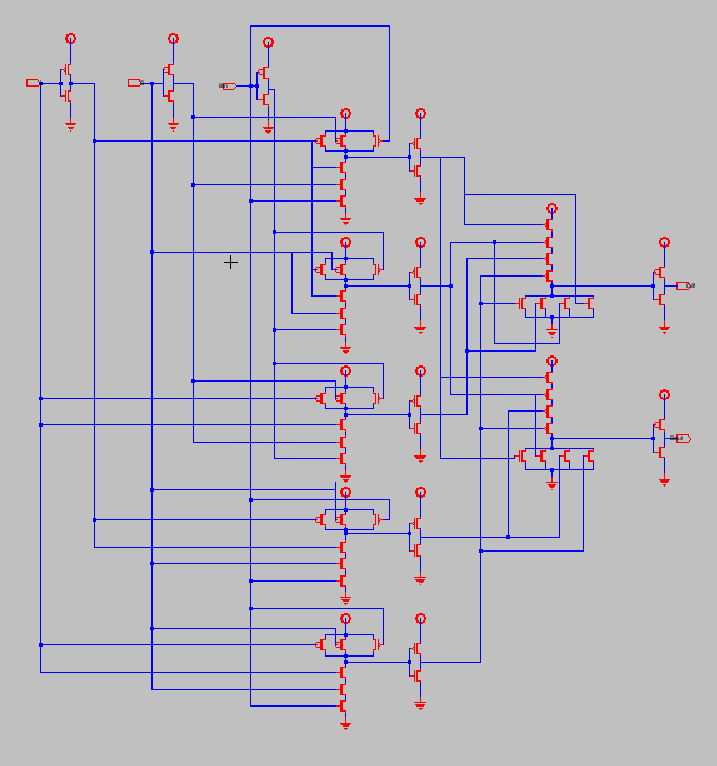
Carry out for add with carry function.

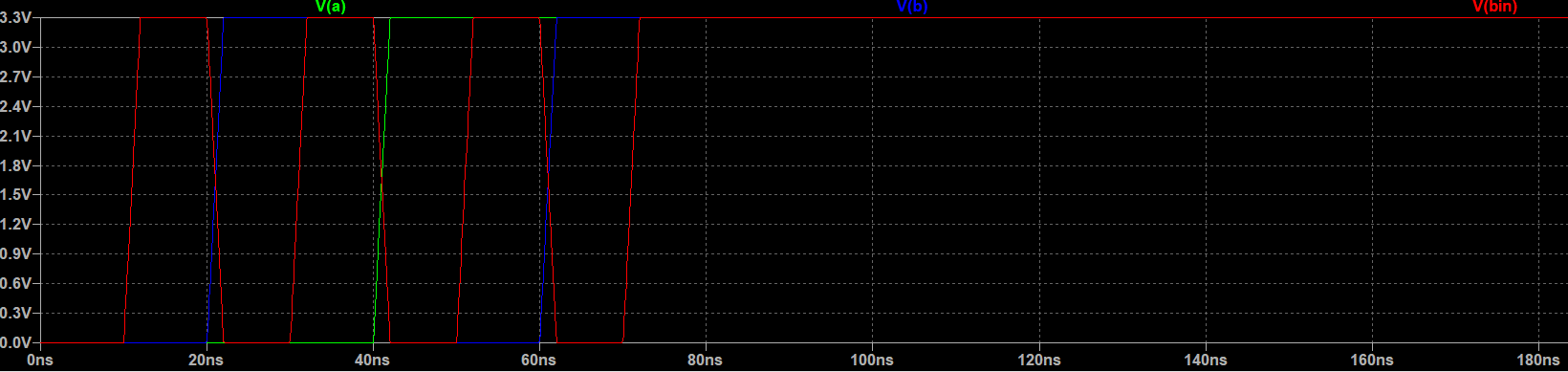
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Cin | Sum | Cout |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Truth table for add with carry function.

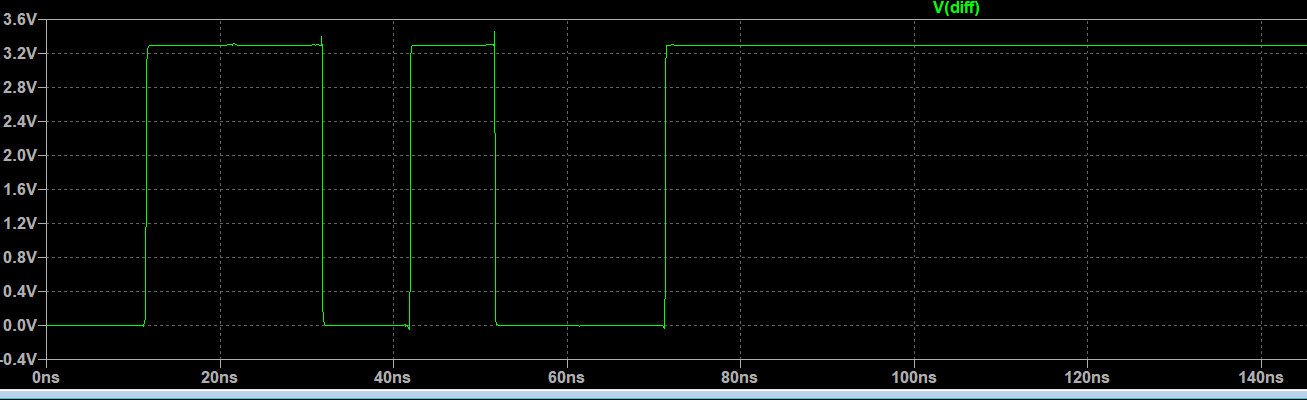
**9-) Subtraction (F=A-B):**

Decoders can be used for subtractor also.This we need summation q1,q2,q4,q7 for difference and summation of q1,q2,q3 and q7 for borrow.So wee need only five output.

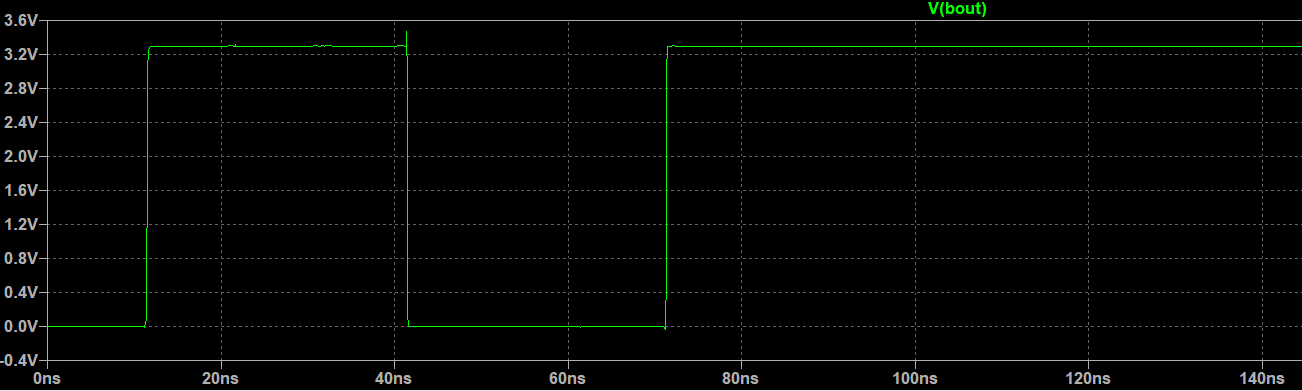




Input wave for subtractor.



Difference output wave.



Borrow out wave.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Bin | Diff | Bout |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

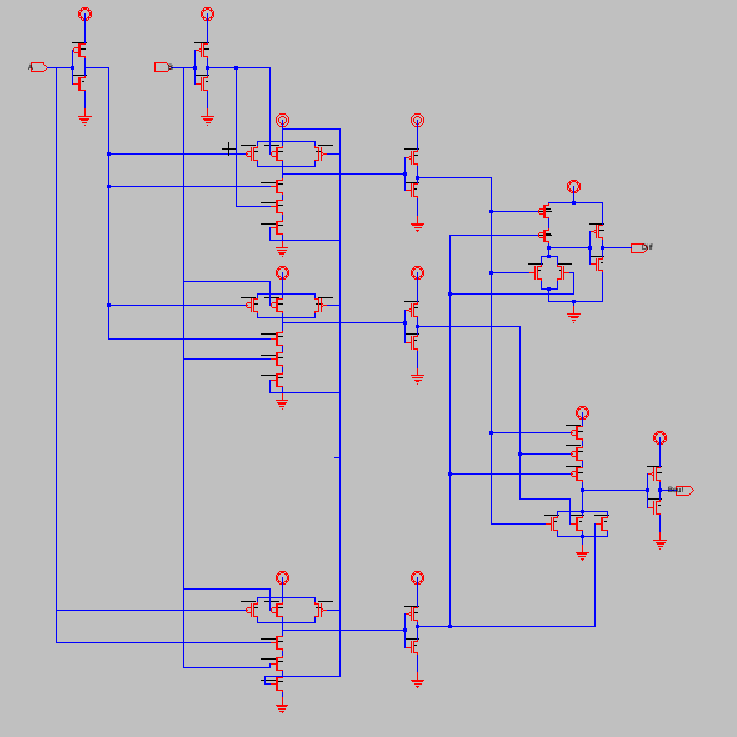
Truth table for full subtractor confirs the output waves.

**10-) Subtraction with Borrow (F=A-B-1):**

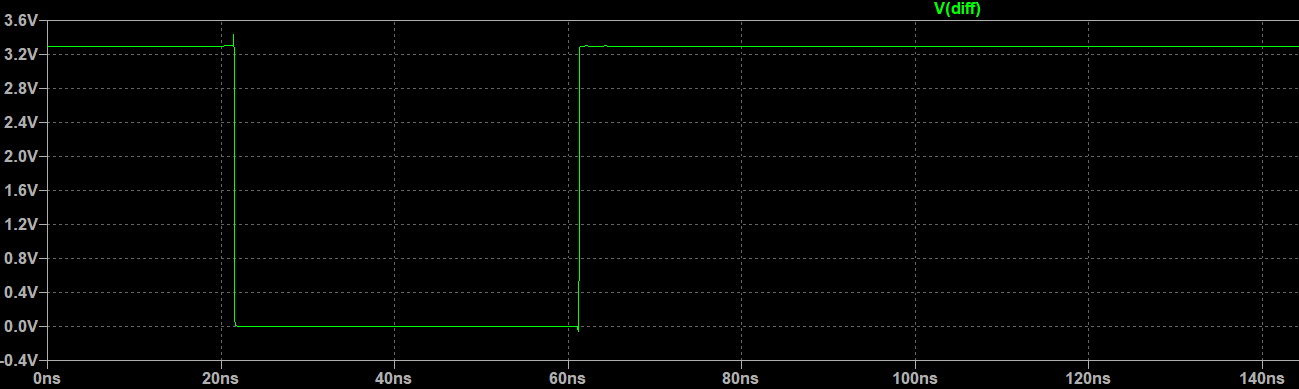
Similar to add with carry function , we can realize this by just taking the cases where Bin input is equal to 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Bin | Diff | Bout |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

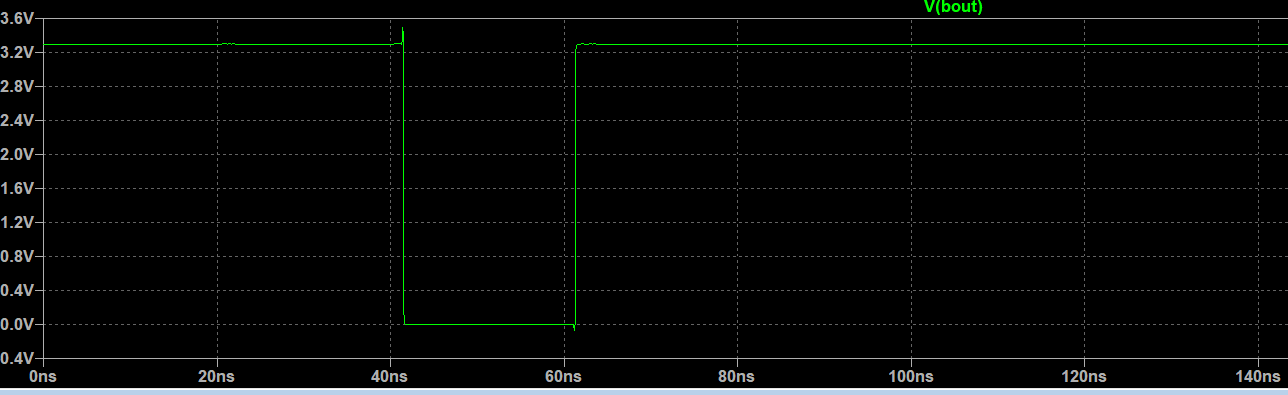
As you can see Diff only has 2 inputs and Bout only has 3 inputs.



Bin is changed with a Vdd input to give constant 1.



Diff output for Subtraction with borrow.



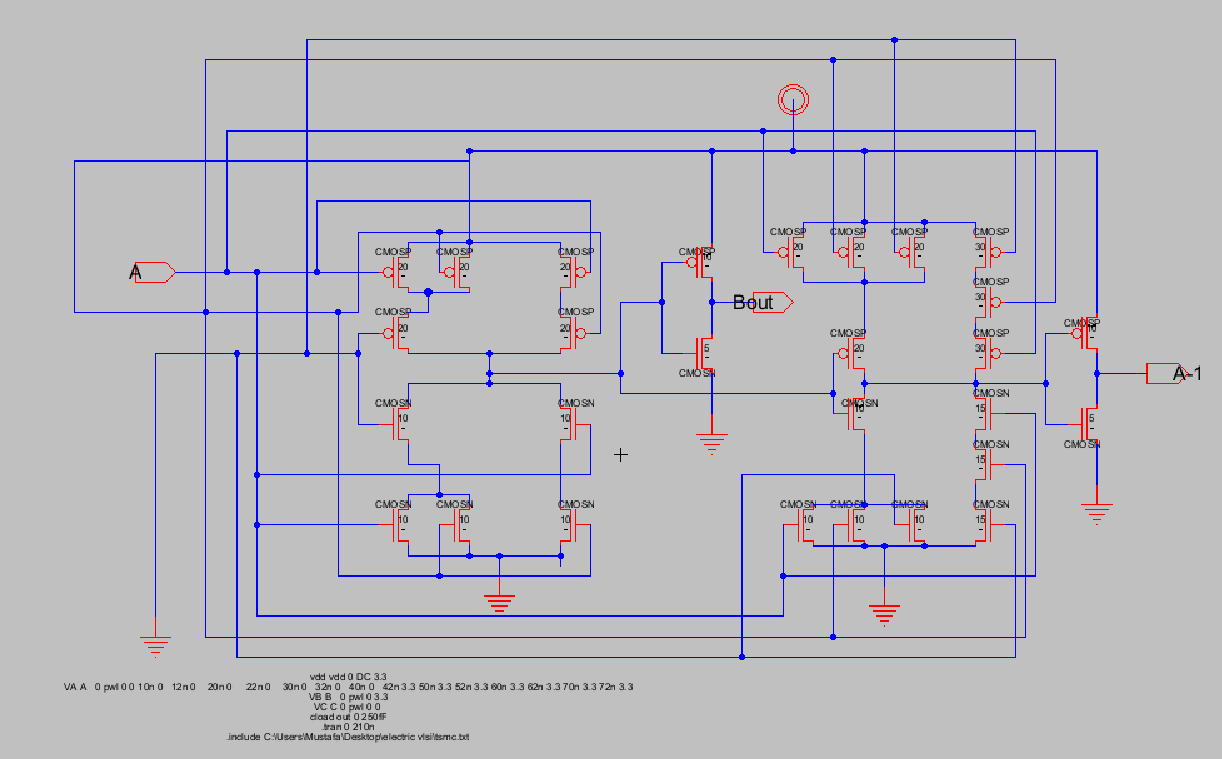
Bout output for subtraction with borrow function.

Truth table given before confirms the outputs.

**11-) Decrementer(F=A-1):**

To subtrac 1 from existing value , we are taking 2’s complement of the existing value.So we are using a full adder and adding 1 to existing value.For n bit we have to add n amount of 1’s.

Initial Cin is 0 and B input is 1.



metin, elektronik eşyalar, iç mekan, vitrin içeren bir resim

Açıklama otomatik olarak oluşturuldu

A and A-1.

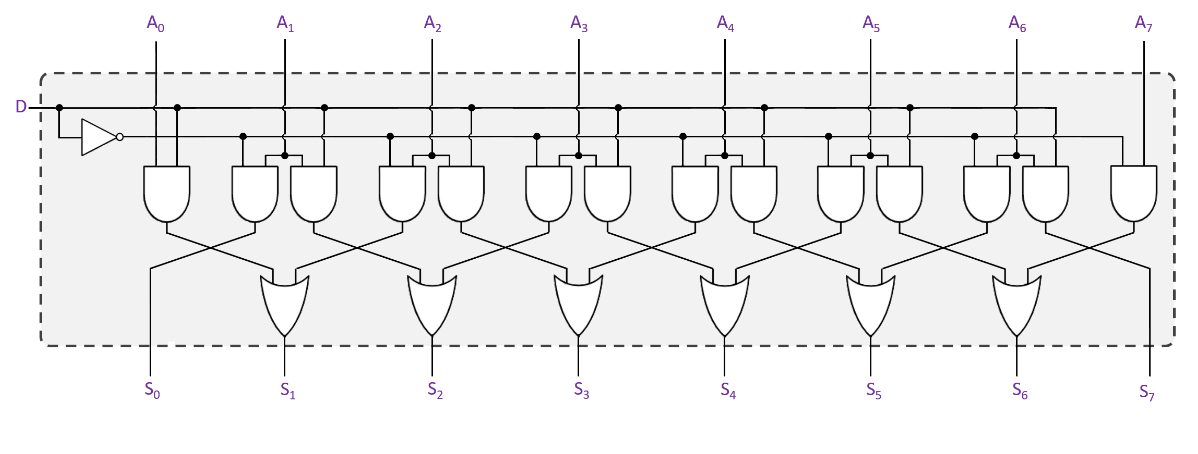


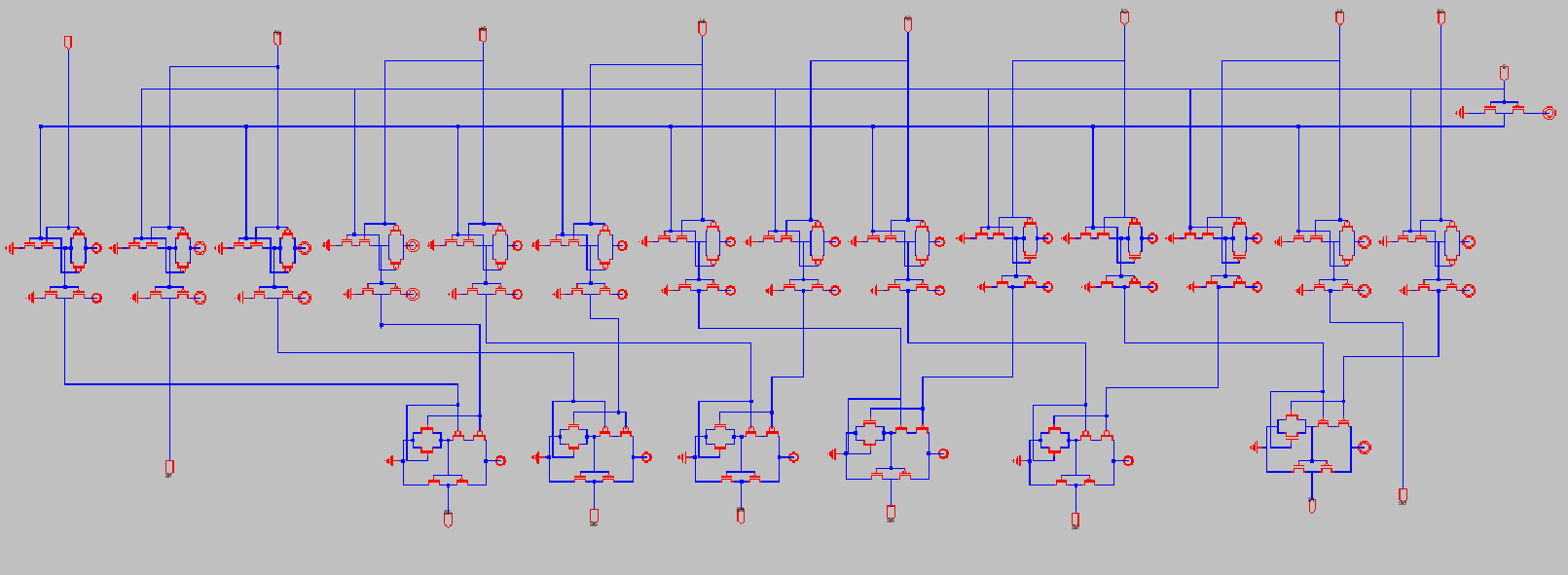
Bout is not 1 when a is 1 because this is a full adder and we are adding 1 to a bit so that carry goes to next bits Carry in so that we can take 2’s complement of given number.

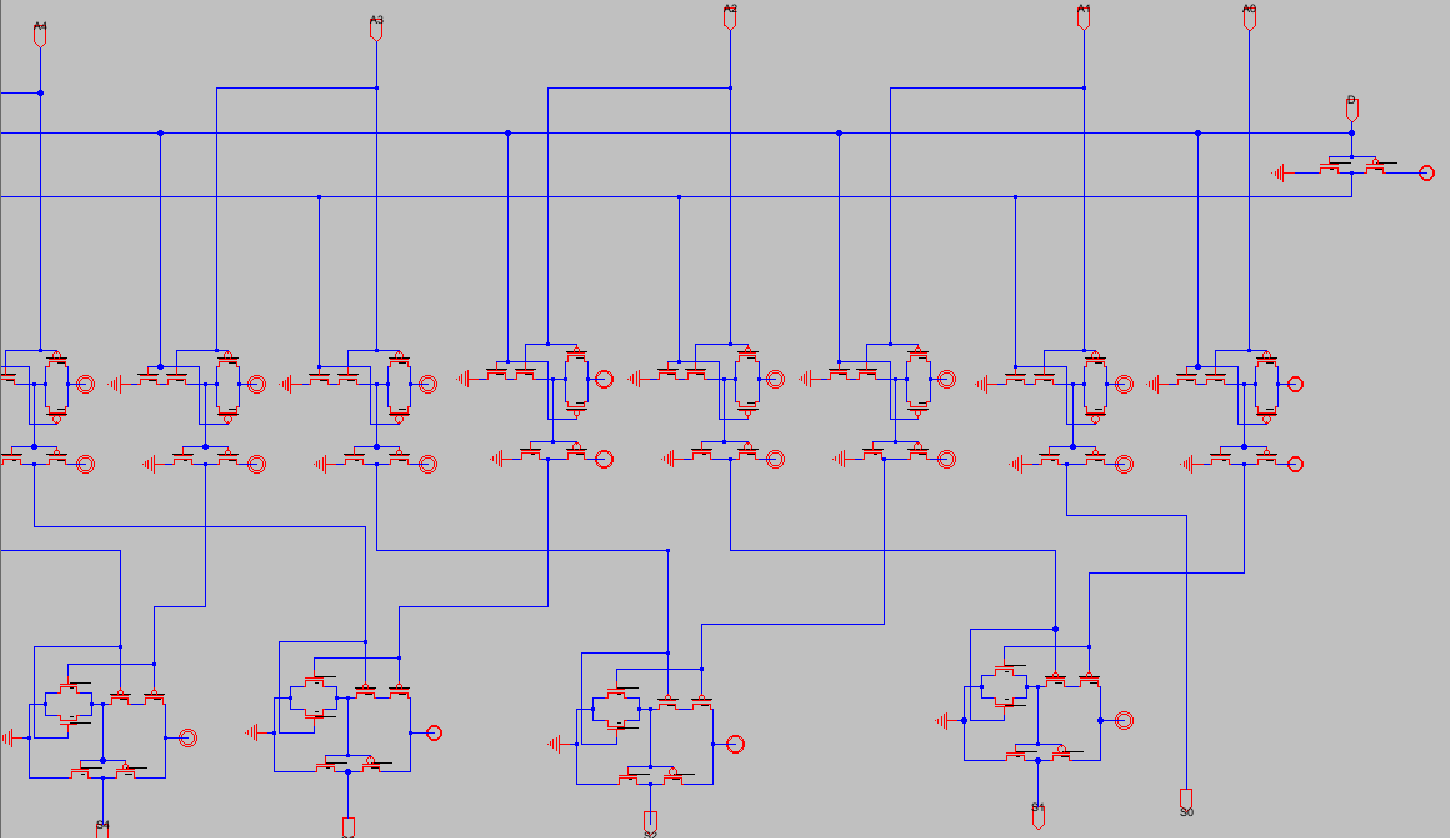
**12-) Shift Left and Shift Right (Universal Shift Register):**

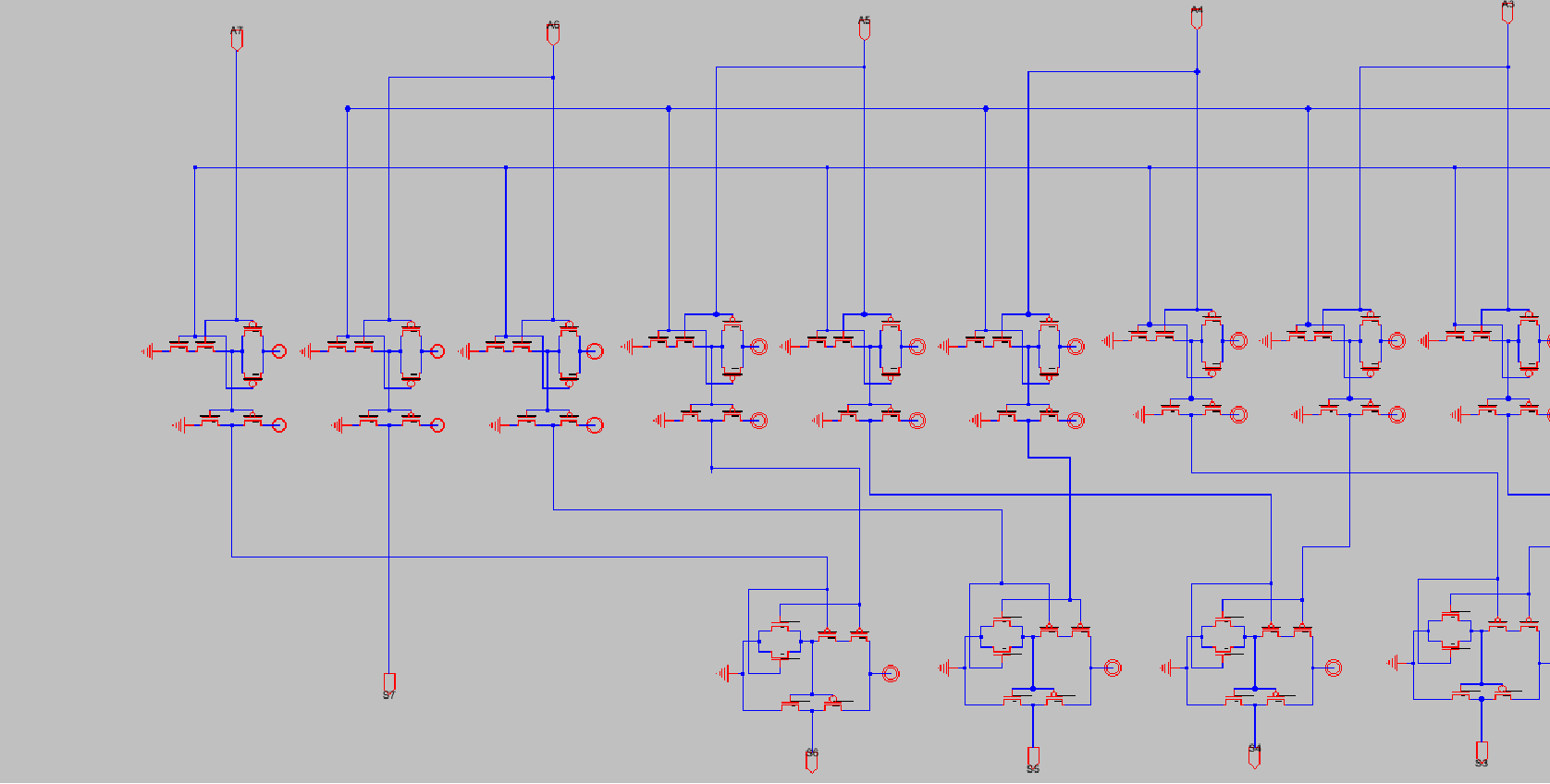
Shift left and shift right can be done in same circuitry.We just need a mod input.

Universal shift register implemented on 8 bit ALU to show its working principle better.







Inputs and outputs from pin 0 to pin 4.

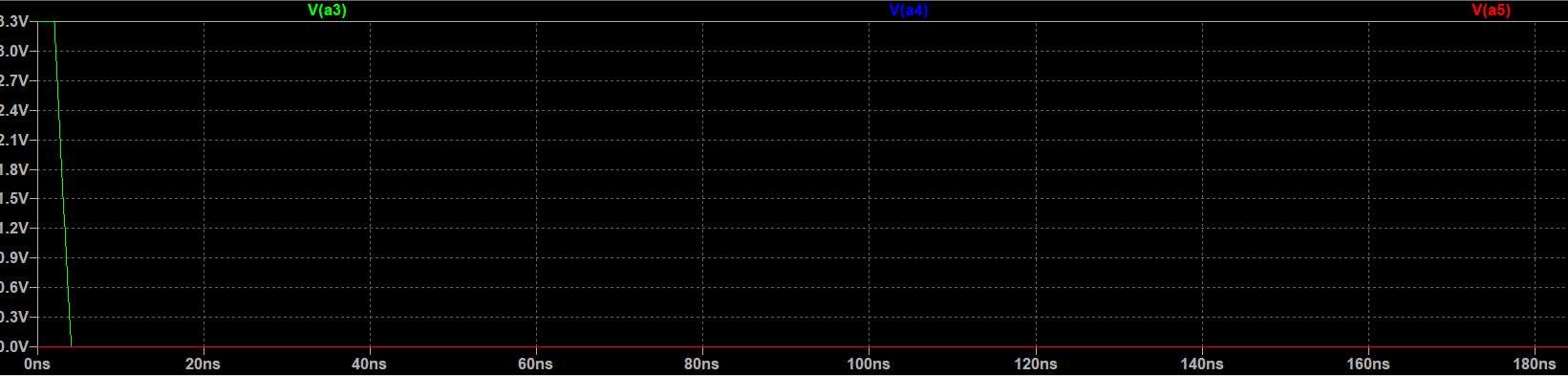
Inputs and outputs from pin 3 to pin 7.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu



First three input , only a0 is 1.



A3 , a4,a5 pins ; only a3 is 1

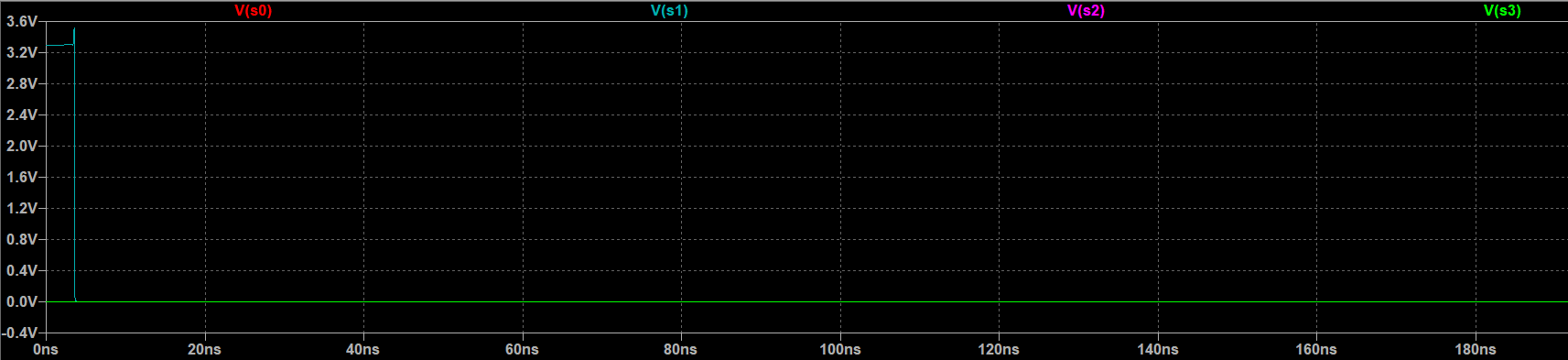
metin, elektronik eşyalar, vitrin, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

A6 and a7; only a6 is 1

Input is 10010010

When D=1 Output must be 01001001



Only S1 is 1

metin, oturma, iç mekan içeren bir resim

Açıklama otomatik olarak oluşturuldu

Only S4 is 1

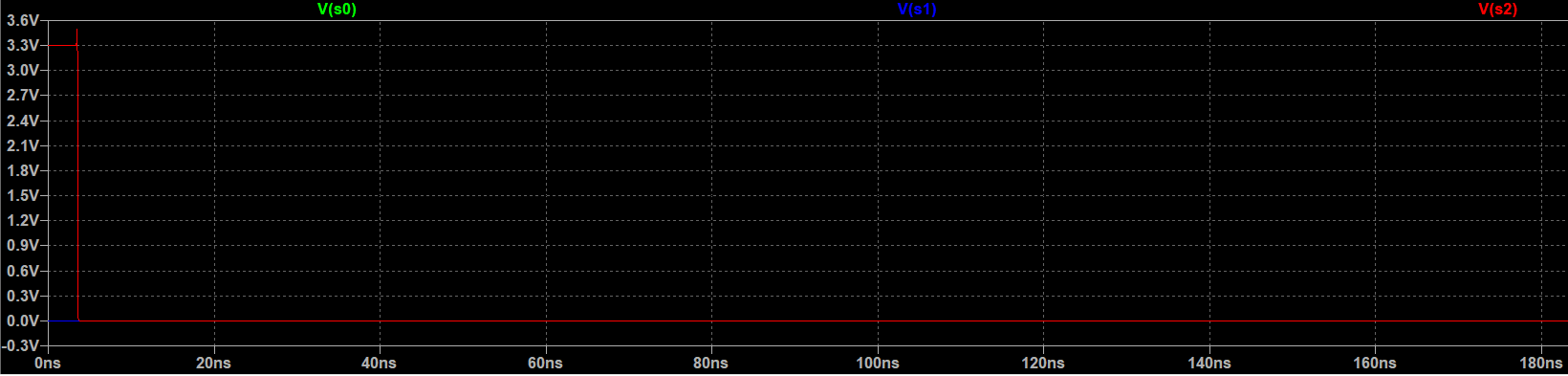
metin, bakarken içeren bir resim

Açıklama otomatik olarak oluşturuldu

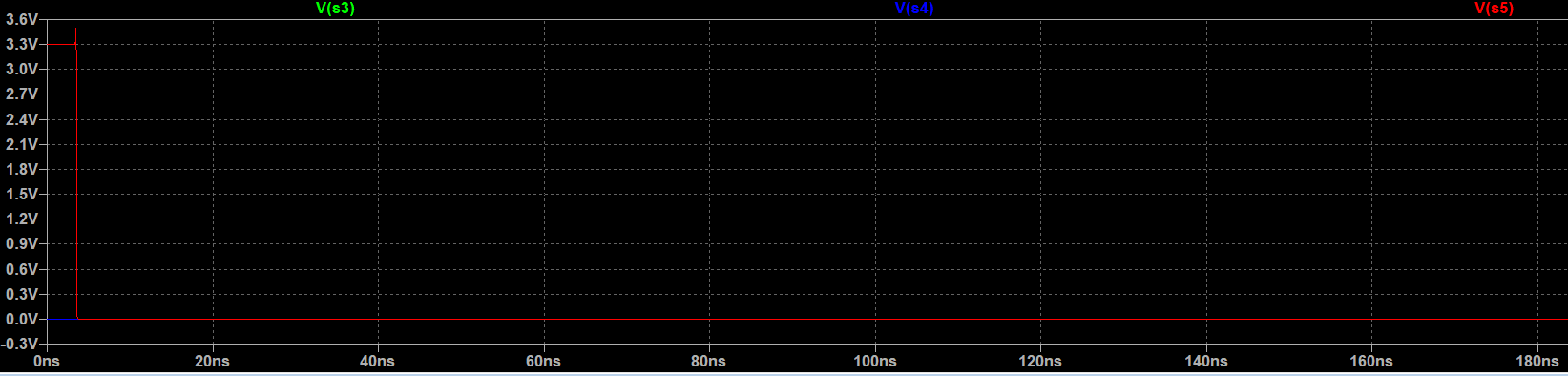
And S7 is 1

So our output is 01001001

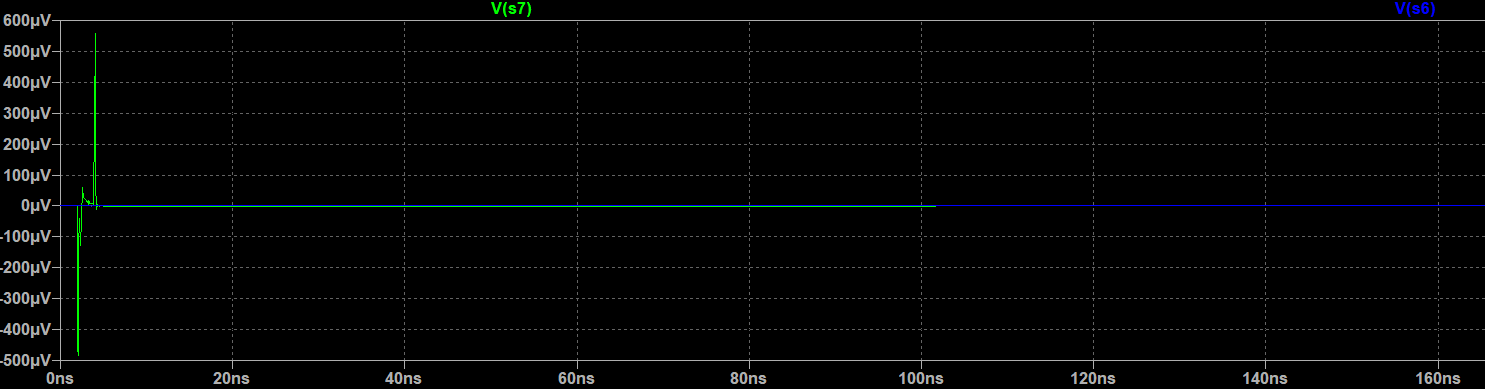
Same input is applied when D is 0.



S0 = 0 , S1=0, S2=1



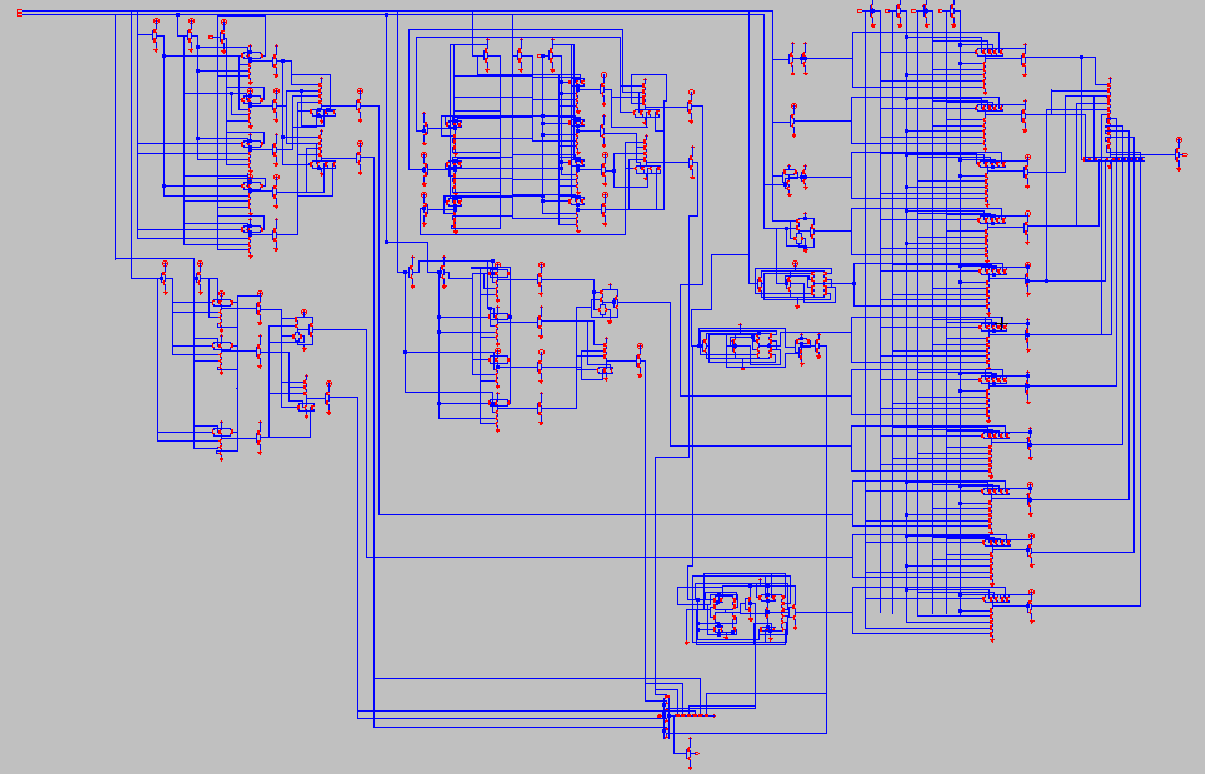
S3=0,S4=0,S5=1

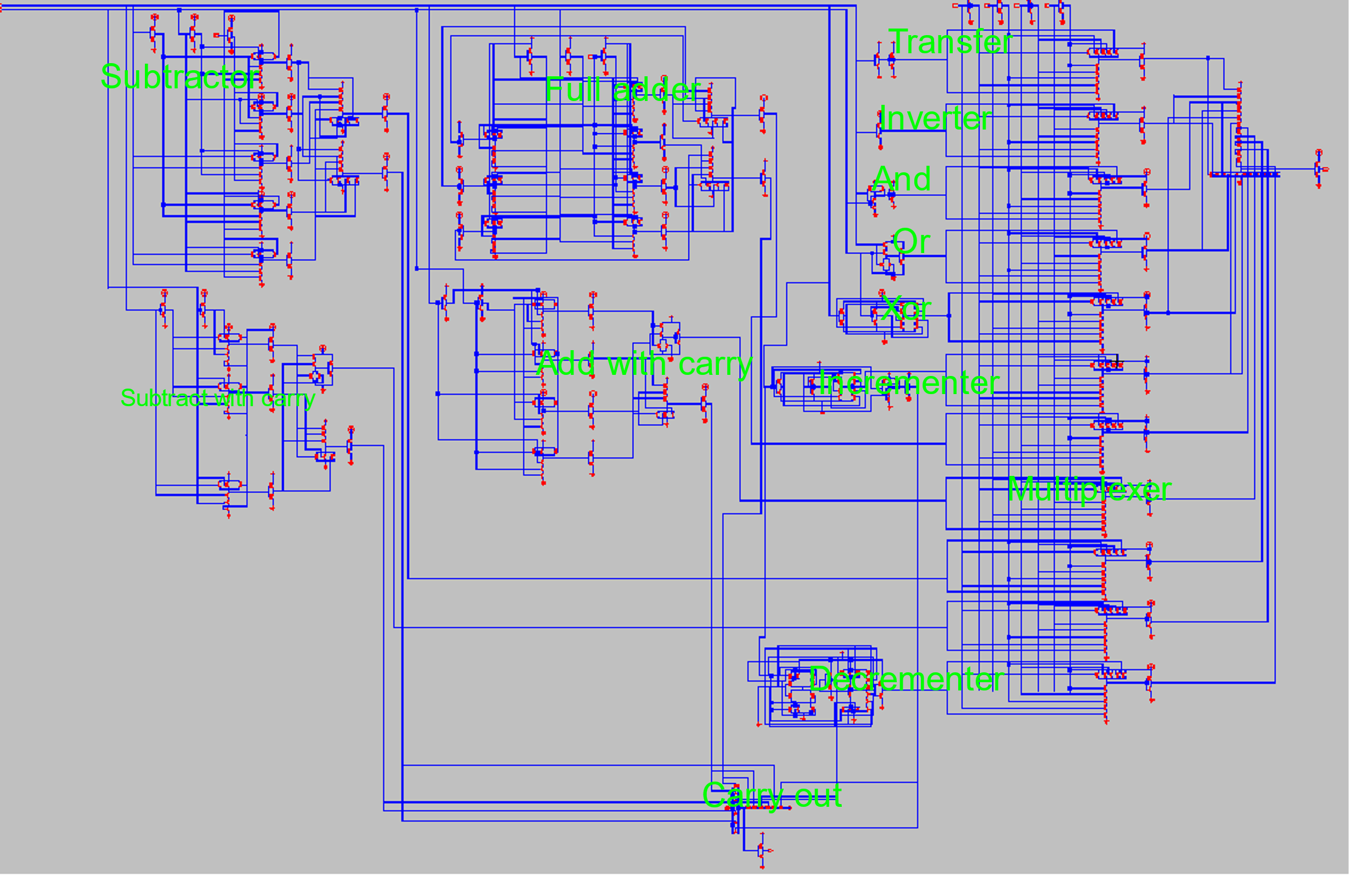


S6=0 and S7 is in micro Volts so bot hare 0.

10010010 becomes 00100100 when D=0.

**1 Bit Alu:**

****



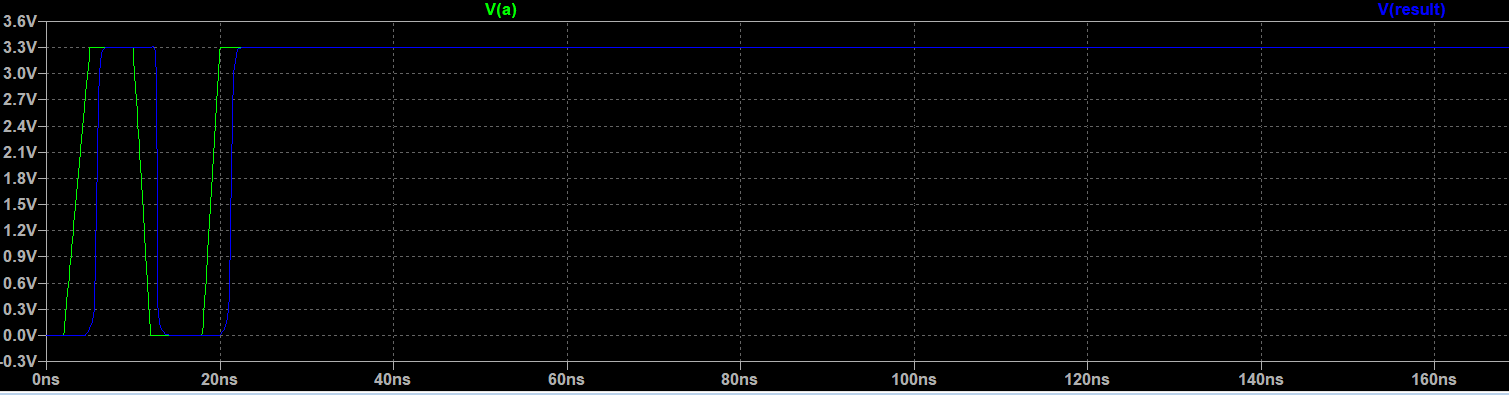
metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Such example for selecting function is like this..

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S3 | S2 | S1 | S0 | Operation | Function |
| 0 | 0 | 0 | 0 | F=A | Transfer A |
| 0 | 0 | 0 | 1 | F=A’ | Not |
| 0 | 0 | 1 | 0 | F= A and B | And |
| 0 | 0 | 1 | 1 | F= A or B | Or |
| 0 | 1 | 0 | 0 | F=A xor B | Xor |
| 0 | 1 | 0 | 1 | F=A+1 | Increment A |
| 0 | 1 | 1 | 0 | F=A+B | Addition |
| 0 | 1 | 1 | 1 | F=A+B+1 | Addition with carry |
| 1 | 0 | 0 | 0 | F=A-B | Subtraction |
| 1 | 0 | 0 | 1 | F=A-B-1 | Subtraction with carry |
| 1 | 0 | 1 | 0 | F=A-1 | Decrement |

Transfer (0000):

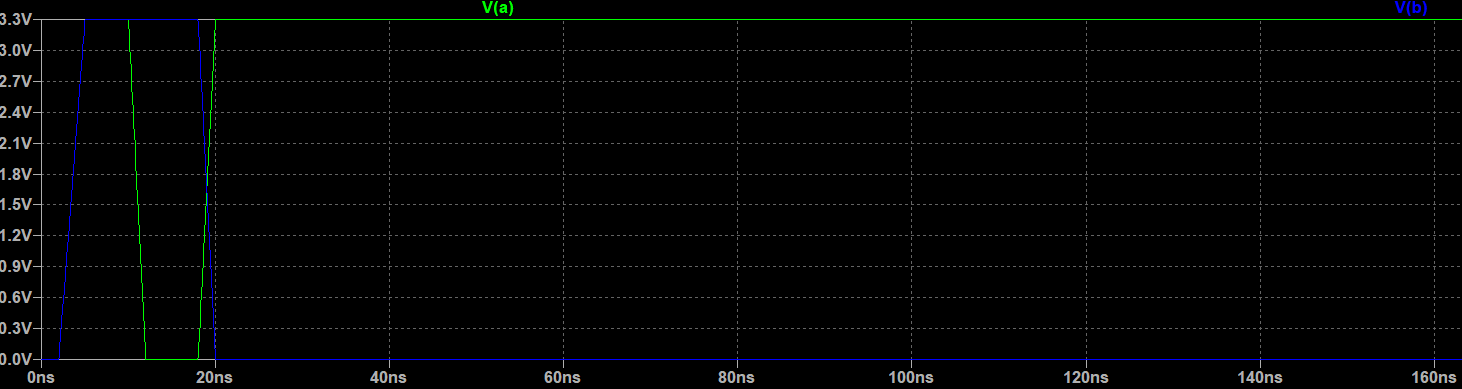


Inverter (0001):

metin, elektronik eşyalar, ekran görüntüsü, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

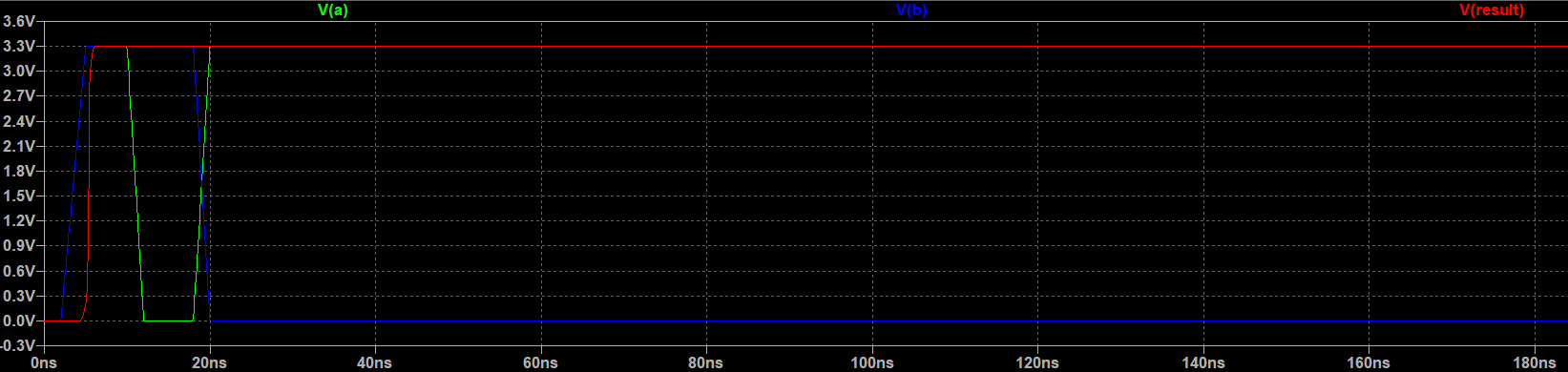
And (0010):



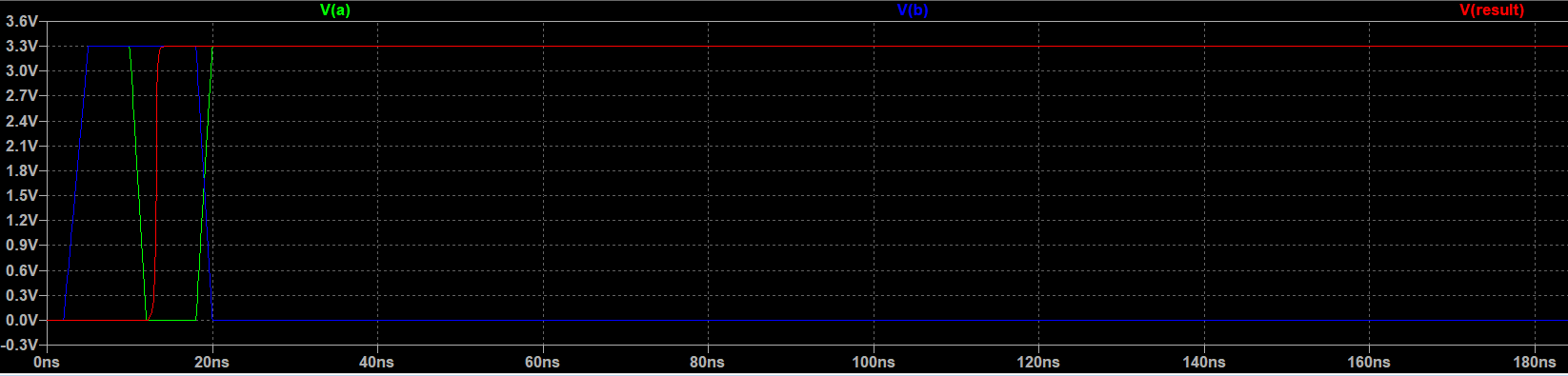
tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Or (0011):



Xor (0100):



Inc (0101):

metin, elektronik eşyalar, ekran görüntüsü, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

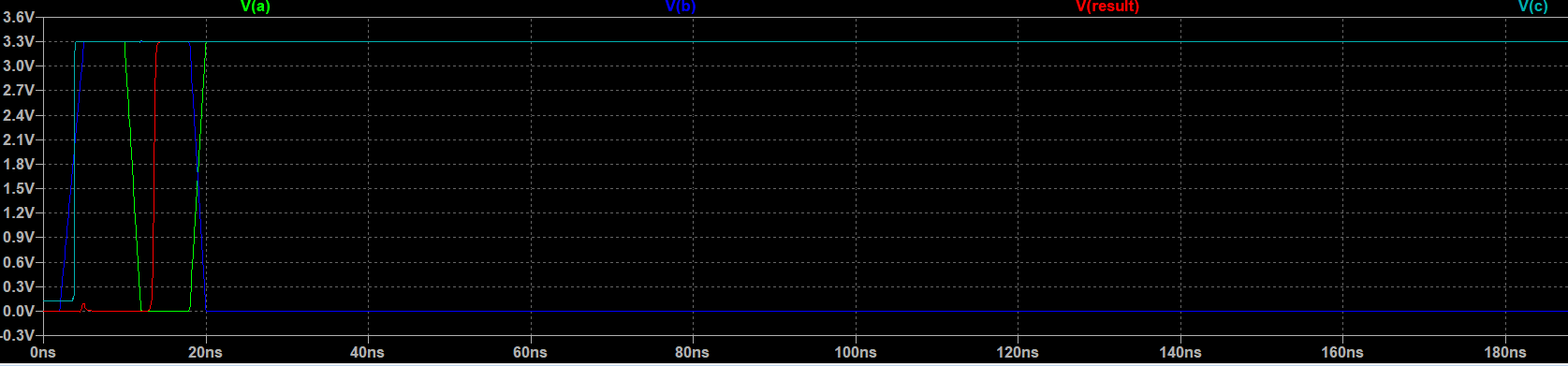
tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Add (0110):

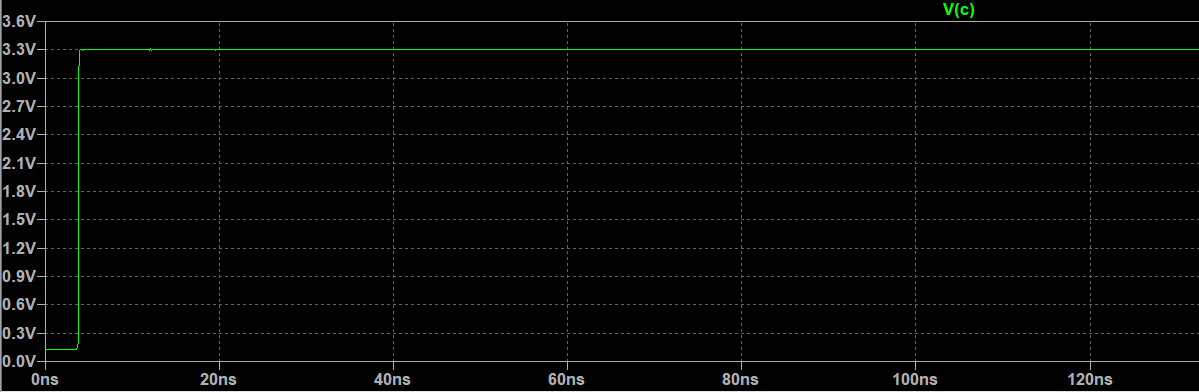
metin, elektronik eşyalar, vitrin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

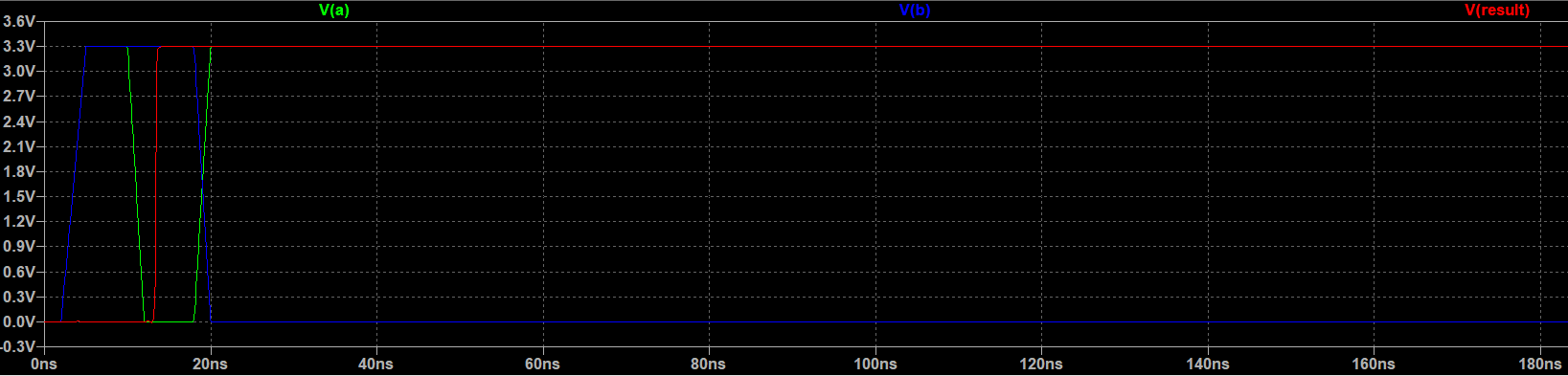


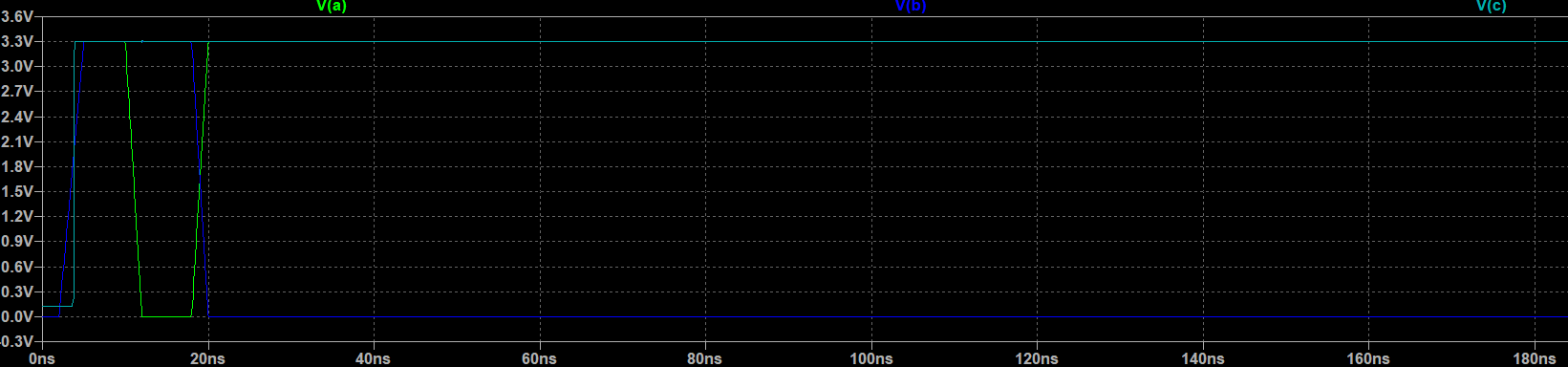
Add with carry (0111):



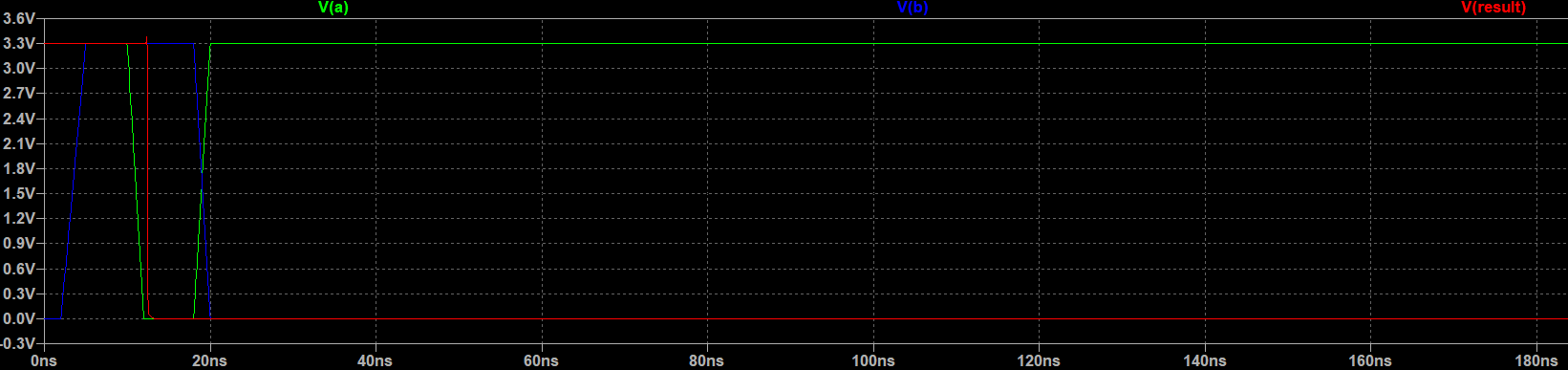


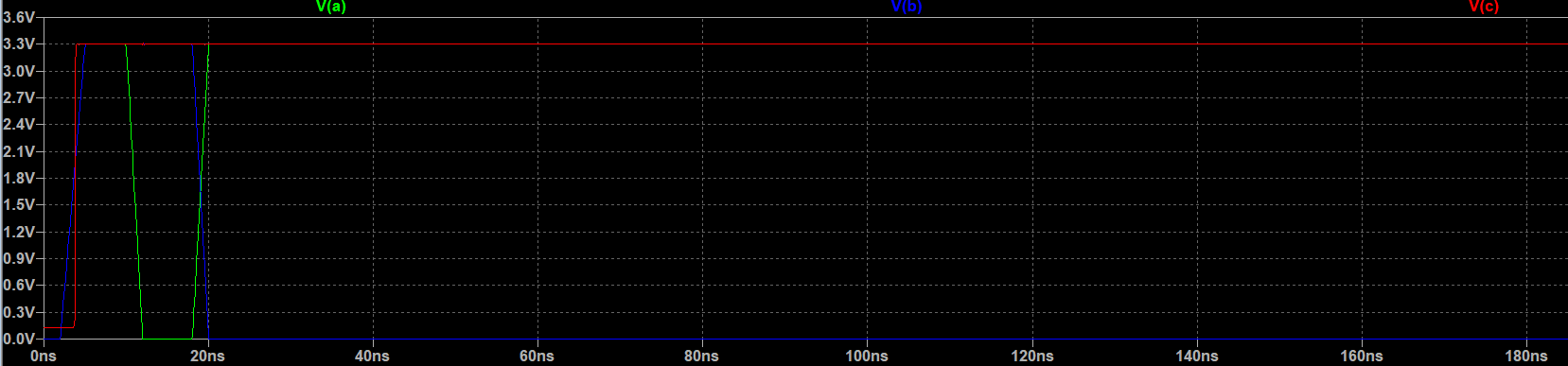
Subtraction (1000):





Subtract with borrorw (1001):





Decrement (1010):

metin, elektronik eşyalar, vitrin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, elektronik eşyalar, ekran görüntüsü içeren bir resim

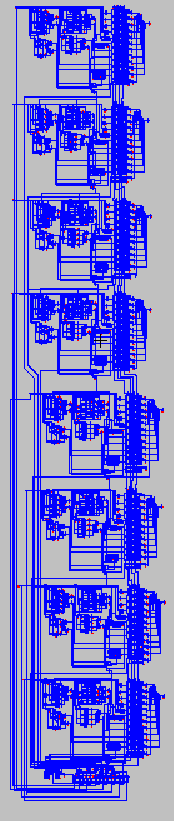
Açıklama otomatik olarak oluşturuldu

Comments (1 Bit Alu):

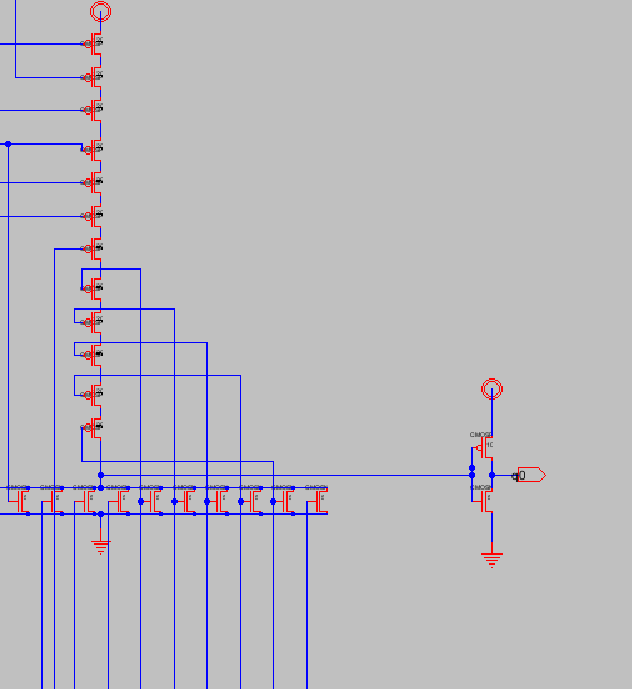
Functions showed individually before implemented on 1 bit alu except for shift registers since it is a 1 bit alu.While testing the each functions carry out output before summing them in an or gate the carry outs were correct , but after summing them and forming a single C output it always become 1 after getting activated.Besides that result output gives correct output and multiplexer working correctly as well. The whole circuitry is as compact as possible the .The smallest parts (inverter transfer and or xor) are close to multiplexer while bigger parts (Full adder ,full subtractor..) are placed away from mux to form a compact shape.

8 bit alu:

8 bit alu consist of 8 1-bit alu’s with correct input and output connections for each function and same selection pins for mux’s.

8 bit alu look like this zoomed out.

Selection pins are same for all multiplexers as well as .Each alu takes individual a and b inputs.And gives q output.

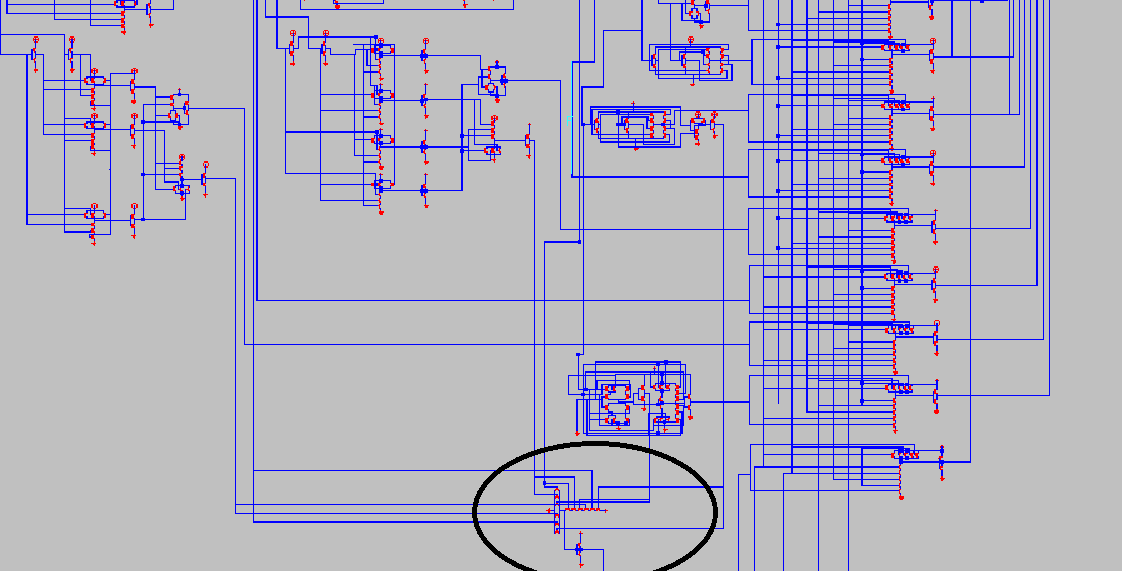


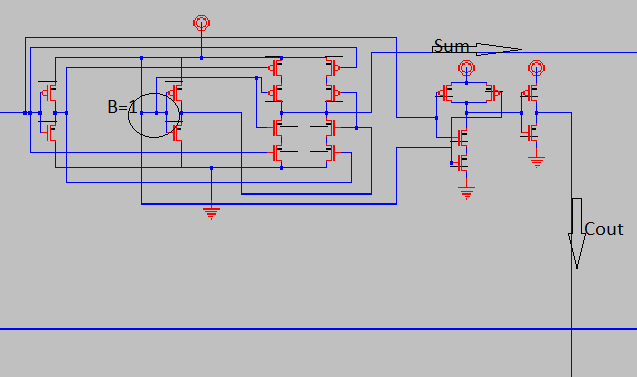
0th bit output

Each 1-bit alu’s configuration are same except at last bit there is a universal shift register.

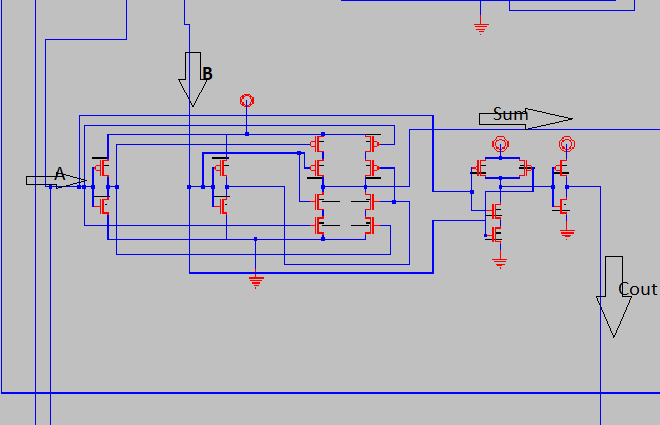
**Carry-out connections:**

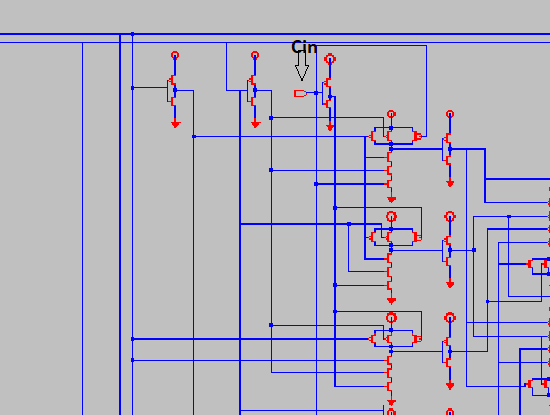
A or gate added to each 1-bit alu which basically sums all the carry out from each function that generates a carry or borrow.This or gates output goes to next alu’s functions that requires a carry-in or borrow-in.





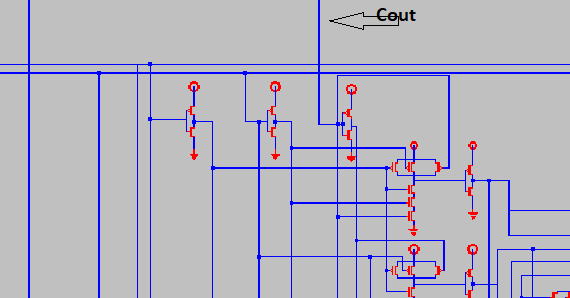
0th bits incrementer.

Here is the 1st bit incrementer.Its B input is previous incrementers Cout.



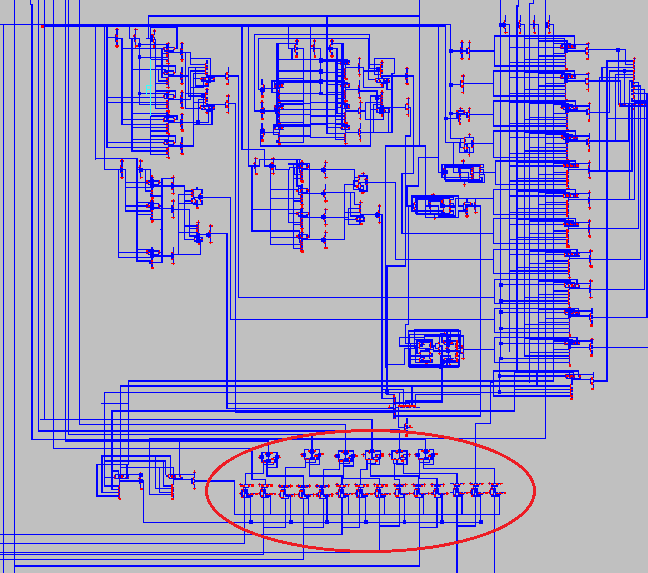
Here 0th bits Cin is 0.

But other full adders cin is previous alu’s Cout.

Cout goes into Cin of next full adder.

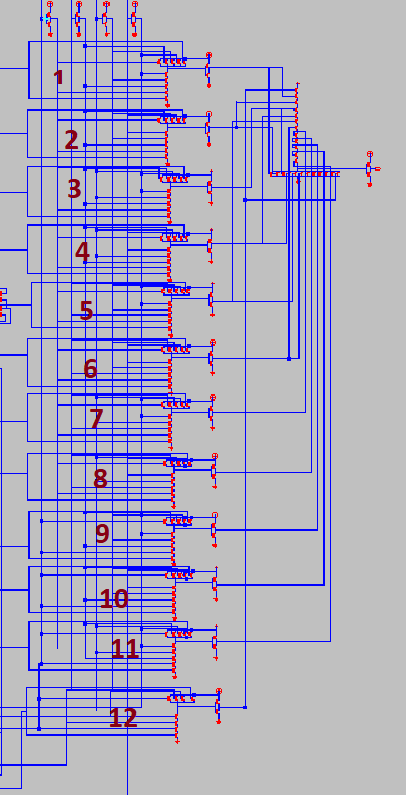
Same connections are done for Add+1,Sub , Sub -1 and Decrementer (2’s complementer which adds 11111111).

**Shift Register:**

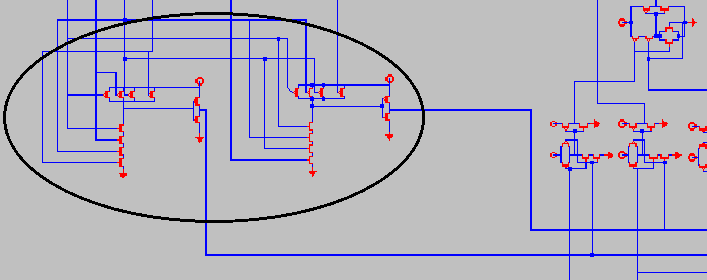
****

Universal shift register placed at the bottom horizontaly.All 8 bit is generated at the bottom but goes to each bits multiplexer respectively.

Although there is 13 function , there is 12 and gates in each multiplexer.12th and gate’s inputs is 110X. Which means that last when selection is 1100 or 1101 result is universal shift

registers output.

But to determine the shift lef tor shift right operation we need two and gates.



When and gate on left is 1 (selection pins 1100) it is shift right operation.When and gate on right is 1 (selection pins 1101) it is left shift operation.

**Comments on 8 bit Alu:**

8 bit alu consist of 8 1-bit alu , but its connection must be done.First bit and last bits alu’s are more important because its logic requires slightly different conection .Functions which requires Cin(Carry in) or Bin (Borrow in) Cout from previous alu distributed to next Cin and Bin’s . Universal shift register is placed at bottom horizontally to form a more compact design.

Transistor sizing I tried to implement is 10:5.I was able to see the error in output when it is not done properly . For example at one stage of design output was 1 at where it shouldnt be .But when sizing is done again in an inverter error is corrected.(3.3 volts is decreased to micro volts.) .

I tried to do avoid common designs in full adder, subtractor , subtract with carry , add with carry in which i used decoder.And for the shift operations.I combined right and left operations together and made a universal shift register so that it only gives 1 output and all mux’s have 12 and gates instead of 13.

**Layout design:**

**Inverter**

**metin, saat, vektör grafikler içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

No erc or drc error in inverter layout design.

Sizing is 10:5.