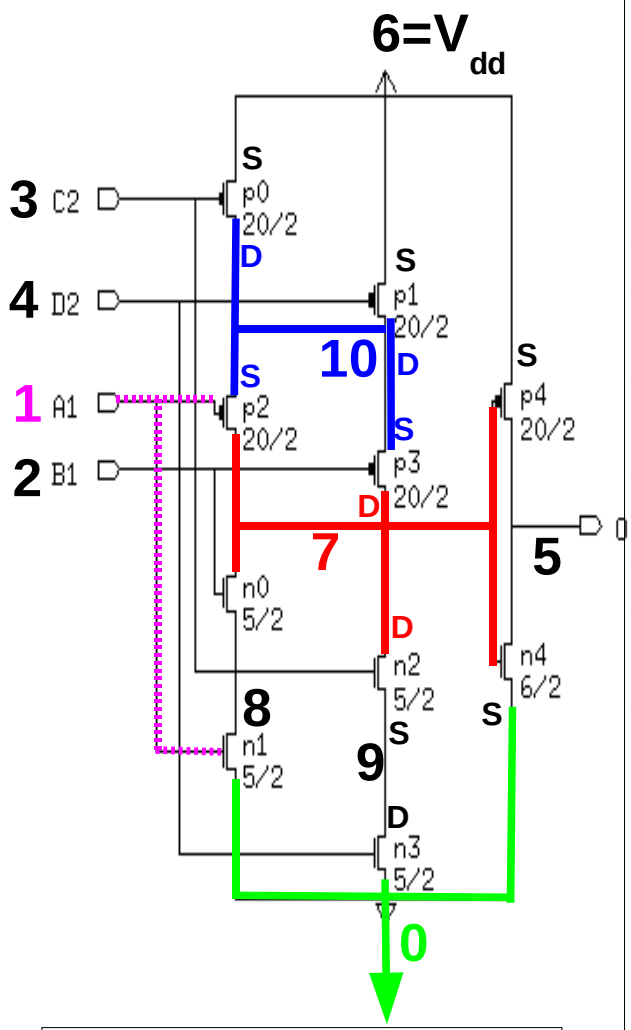
EECS 281: Homework #1 Problem 1 (1a SPICE)

Convert the following schematic to (a) SPICE, (b) truth table, (c) logic gates (d) logic expression.

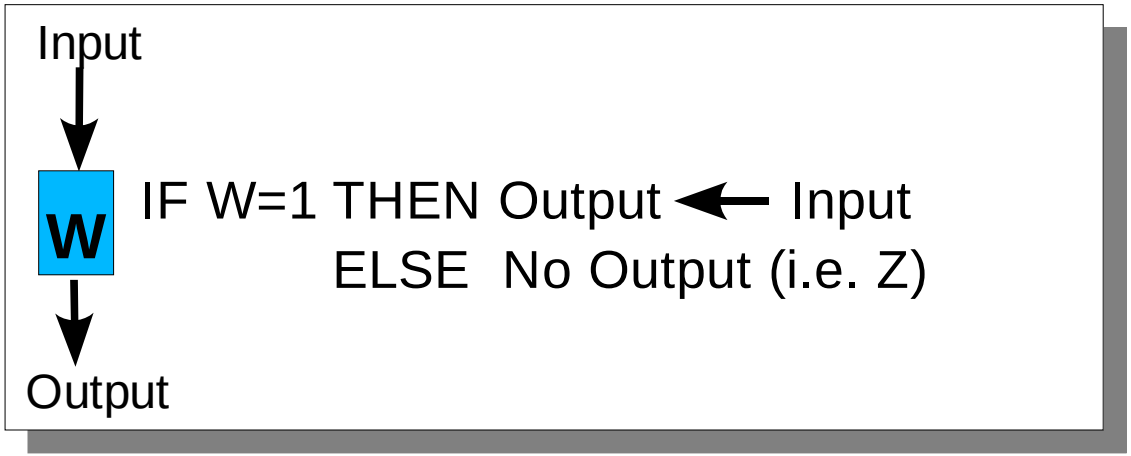


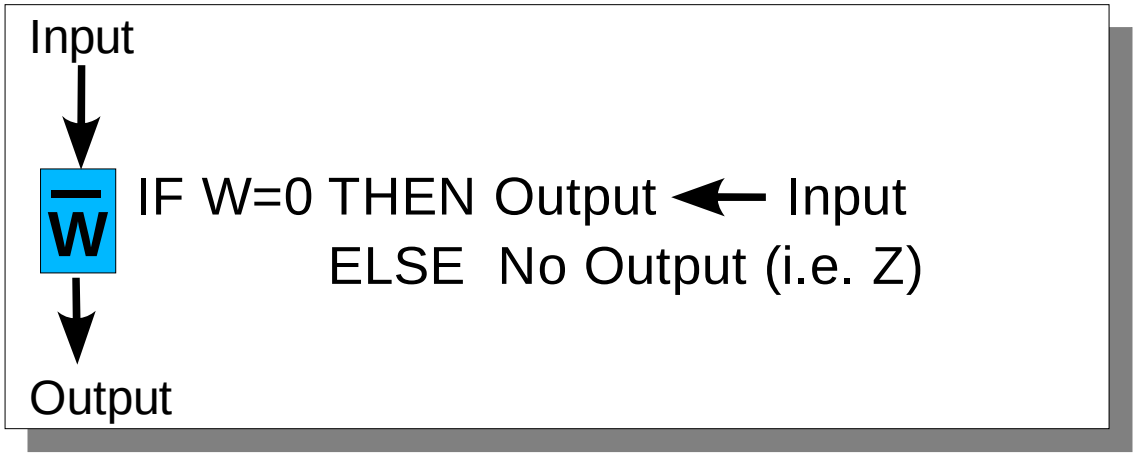
Compare this with with the CMOS AND-OR-INVERT gate Fig. 3-20 in Wakerly's book.

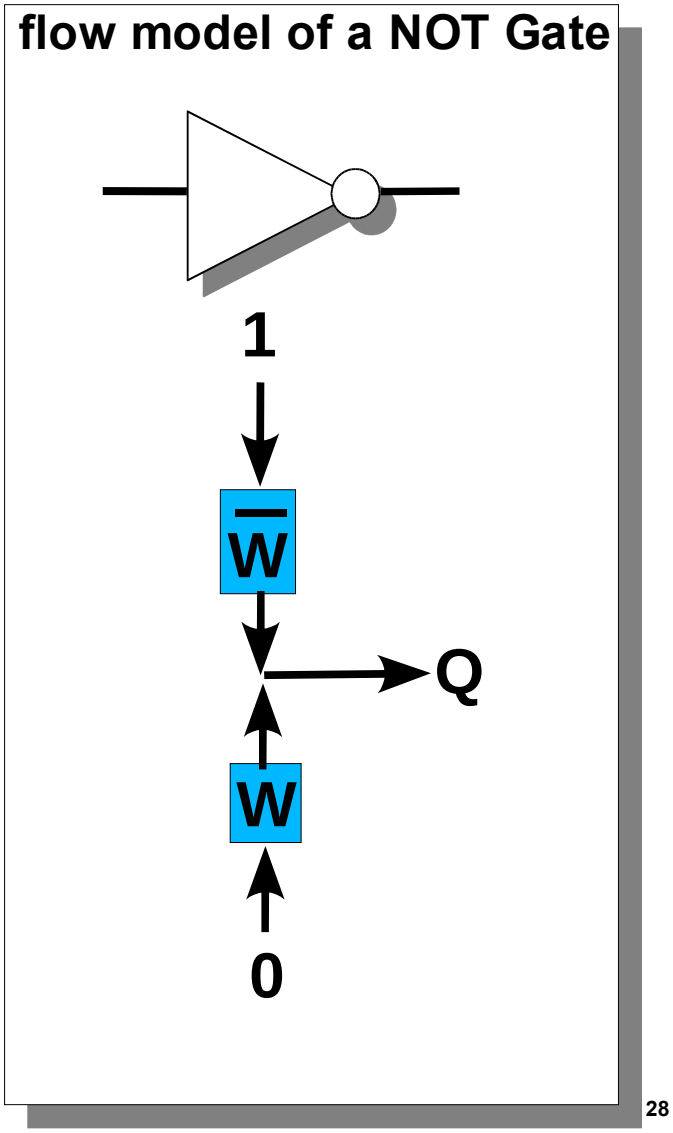
```
.SUBCKT GATE1X  1 2 3 4 5 6
* .SUBCKT GATE1X A1=1 B1=2 C1=3 D1=4 Q1=5 VDD=6
*
*Mname DRAIN GATE SOURCE SUBSTRATE MODEL WIDTH   LENGTH
*      NODE  NODE NODE   NODE      NAME  MICRONS MICRONS
*----- ----- ---- ------ --------- ----  ------- ------
MP0      10     3     6        6          PCH   W=20U   L=2U
MP2      7      1     10       6          PCH   W=20U   L=2U
MN0      7      2     8        0          NCH   W=5U    L=2U
MN1      8      1     0        0          NCH   W=5U    L=2U

MP1      10     4     6        6          PCH   W=20U   L=2U
MP3      7      2     10       6          PCH   W=20U   L=2U
MN2      7      3     9        0          NCH   W=5U    L=2U
MN3      9      4     0        0          NCH   W=5U    L=2U

MP4      5      7     6        6          PCH   W=20U   L=2U
MN4      5      7     0        0          NCH   W=6U    L=2U
.ENDS GATE1X

*MODEL NAME TYPE; 2 Micron process technology
*----- ---- ----
.model NCH nmos LEVEL=1 KP=48E-6 LAMBDA=0.032
          VTO=0.88 GAMMA=0.66 PHI=0.7
.model PCH pmos LEVEL=1 KP=16E-6 LAMBDA=0.044
          VTO=-0.85 GAMMA=0.69 PHI=0.7
```

1b: flow model

Another way to view CMOS logic is the box flow model



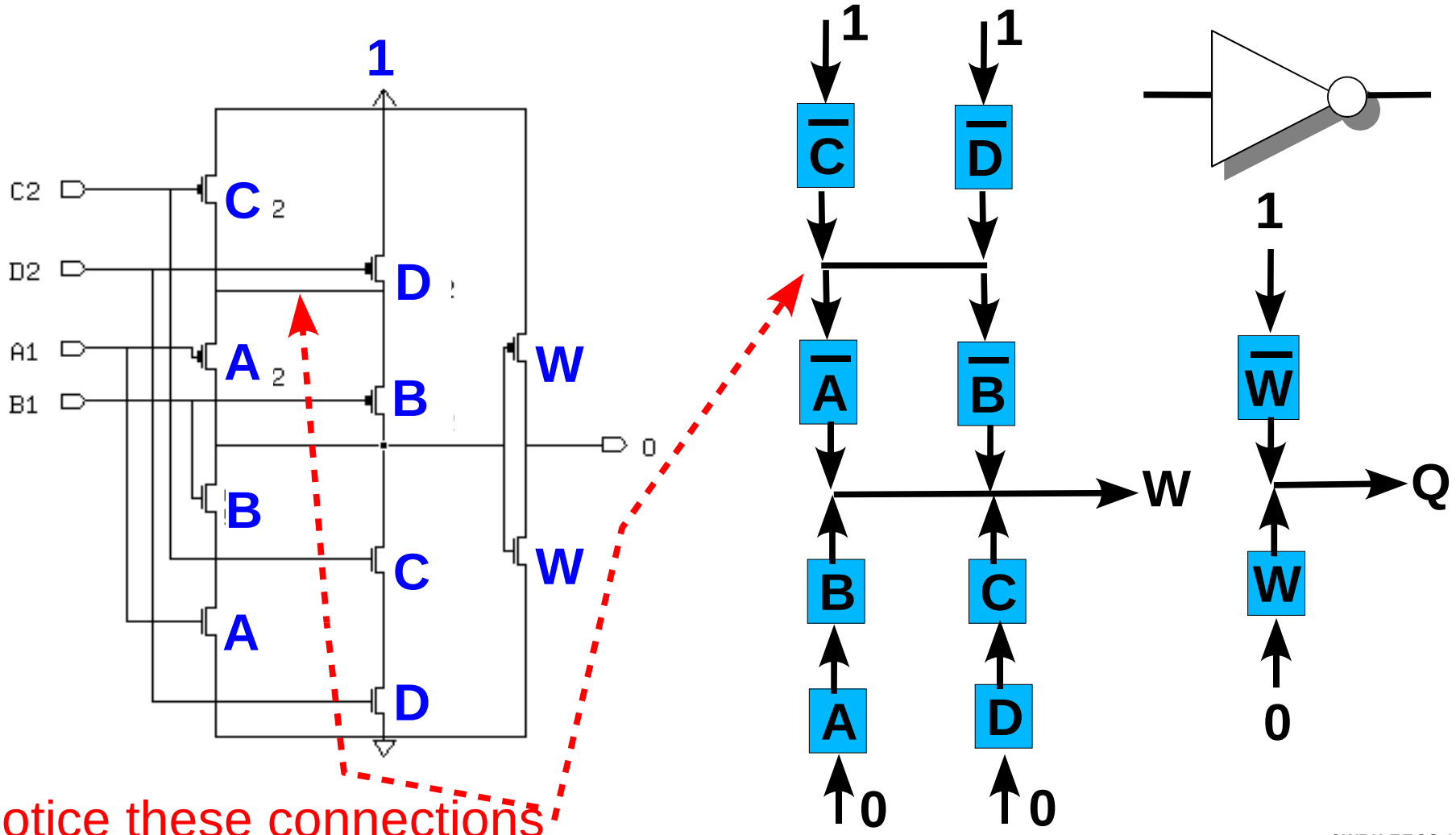




1b: flow model

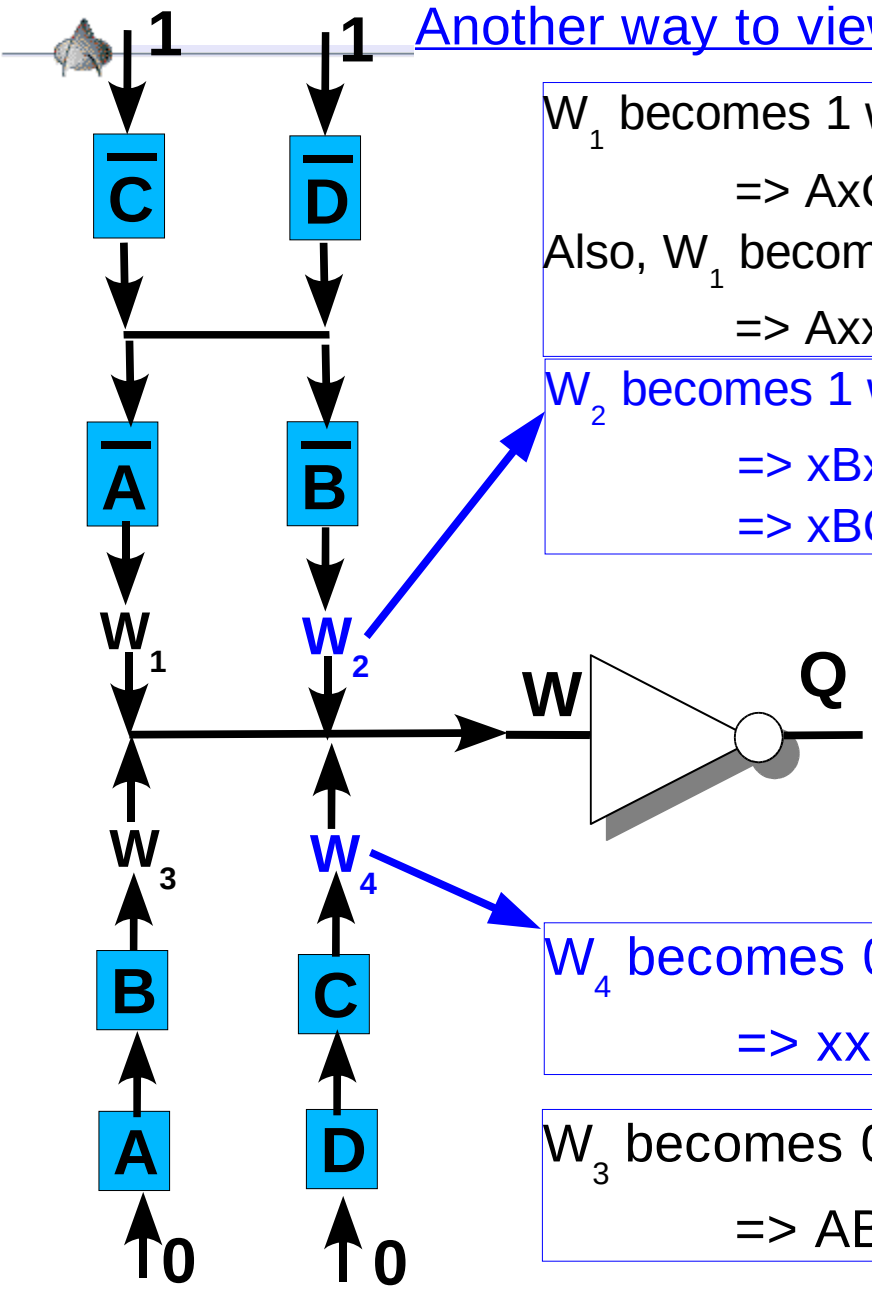
Convert the following schematic to (a) SPICE, (b) truth table, (c) logic gates (e) logic expression.

Rewrite the circuit as a flow model



Notice these connections

1b: flow model



Another way to view this (where x=1 or 0=Don't Care):

W_1 becomes 1 when C=0 & A=0 otherwise Z

=> AxCx => 0x0x => **0**0**0**0, **0**0**0**1, **0**1**0**0, **0**1**0**0

Also, W_1 becomes 1 when D=0 & A=0 & otherwise Z

=> AxxD => 0xx0 => 0000, 0010, 0100, 0110

W_2 becomes 1 when (D=0 & B=0) OR (C=0 & B=0) otherwise Z

=> xBxD => x0x0 => 0000, 0010, 1000, 1010

=> xBCx => x00x => 0000, 0001, 1000, 1001

OVERLAP: notice W_3 and W_4 overlaps when ABCD is equal to 1111 that is ok because the result W_4 =1 W_4 =1 it's when the differ, it's bad!

W_4 becomes 0 only when D=1 & C=1

=> xxCD => xx11 => 0011, 0111, 1011, 1011

W_3 becomes 0 only when A=1 & B=1

=> ABxx => 11xx => 1100, 1101, 1110, 1111

1b: Truth Table (behavior)

W_1 is Z otherwise is 1 when

=> AxCx => 0x0x => 0000, 0001, 0100, 0101

=> AxxD => 0xx0 => 0000, 0010, 0100, 0110

W_2 is Z otherwise is 1 when

=> xBxD => x0x0 => 0000, 0010, 1000, 1010

=> xBCx => x00x => 0000, 0001, 1000, 1001

W_3 becomes 0 when A=1 & B=1 otherwise Z

=> ABxx => 11xx => 1100, 1101, 1110, 1111

W_4 becomes 0 when C=1 & D=1 otherwise Z

=> xxCD => xx11 => 0011, 0111, 1011, 1111

$W = W_1$ OR W_2 OR W_3 OR W_4

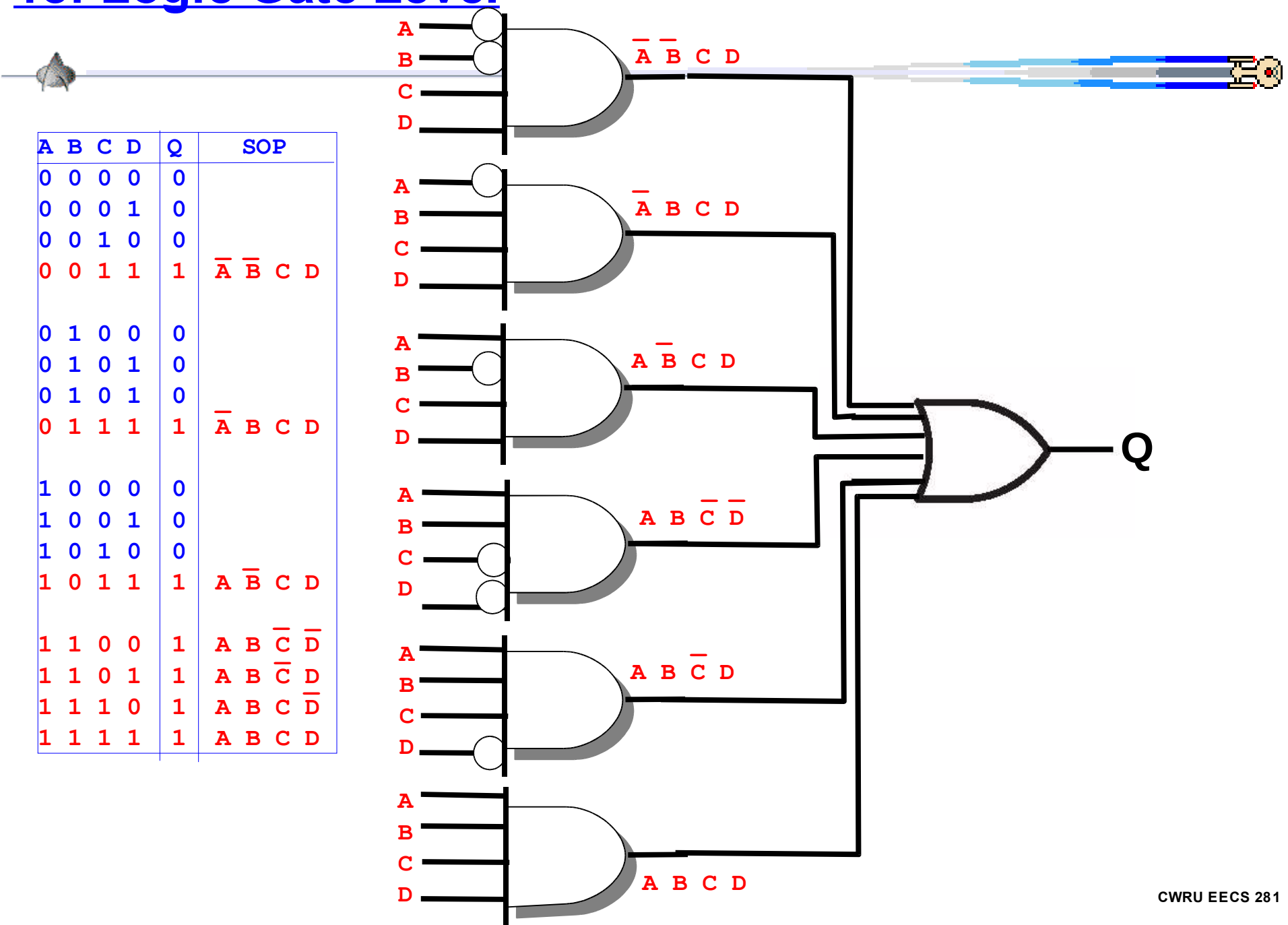
$Q = NOT(W) = NOT(W_1$ OR W_2 OR W_3 OR $W_4)$

Cases of ABCD	
W1,W1,W2,W2:	0x0x,0xx0,x0x0,x00x
W1,W2:	0x0x,x00x
W1,W2:	0xx0,x0x0
W4:	xx11
W1,W1:	0x0x,0xx0
W2:	0x0x
W1:	0xx0
W4:	xx11
W2:	x0x0,x00x
W2:	x00x
W2:	x0x0
W4:	xx11
W3:	11xx
W3:	11xx
W3:	11xx
W3,W4:	xx11,11xx

A	B	C	D	W_1	W_2	W_3	W_4	W	Q
0	0	0	0	1	1	Z	Z	1	0
0	0	0	1	1	1	Z	Z	1	0
0	0	1	0	1	1	Z	Z	1	0
0	0	1	1	Z	Z	Z	0	0	1
0	1	0	0	1	Z	Z	Z	1	0
0	1	0	1	1	Z	Z	Z	1	0
0	1	0	1	1	Z	Z	Z	1	0
0	1	1	1	Z	Z	Z	0	0	1
1	0	0	0	Z	1	Z	Z	1	0
1	0	0	1	Z	1	Z	Z	1	0
1	0	1	0	Z	1	Z	Z	1	0
1	0	1	1	Z	Z	Z	0	0	1
1	1	0	0	Z	Z	0	Z	0	1
1	1	0	1	Z	Z	0	Z	0	1
1	1	1	0	Z	Z	0	Z	0	1
1	1	1	1	Z	Z	0	0	0	1

1c: Logic Gate Level

A	B	C	D	Q	SOP
0	0	0	0	0	
0	0	0	1	0	
0	0	1	0	0	
0	0	1	1	1	$\bar{A}\ \bar{B}\ C\ D$
0	1	0	0	0	
0	1	0	1	0	
0	1	0	1	0	
0	1	1	1	1	$\bar{A}\ B\ C\ D$
1	0	0	0	0	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	$A\ \bar{B}\ C\ D$
1	1	0	0	1	$A\ B\ \bar{C}\ \bar{D}$
1	1	0	1	1	$A\ B\ \bar{C}\ D$
1	1	1	0	1	$A\ B\ C\ \bar{D}$
1	1	1	1	1	$A\ B\ C\ D$



1d: Logic Expression

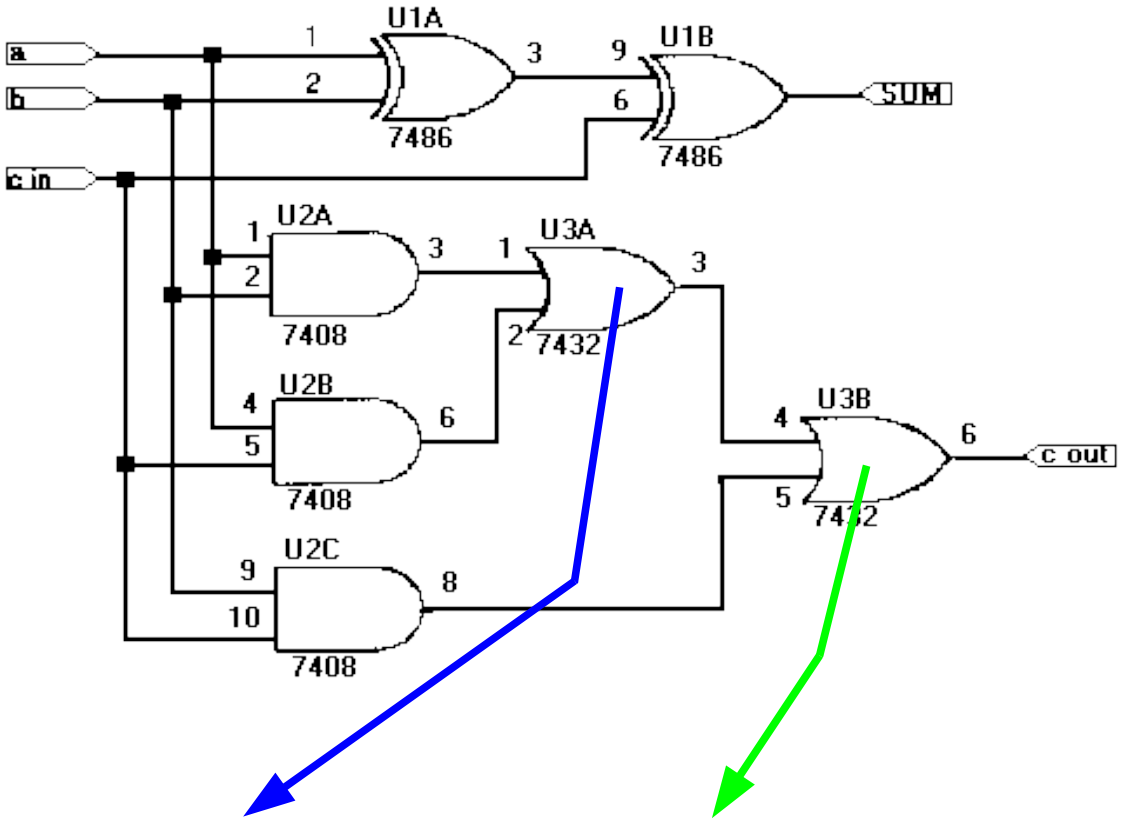
A	B	C	D	Q	SOP
0	0	0	0	0	
0	0	0	1	0	
0	0	1	0	0	
0	0	1	1	1	$\overline{A}\ \overline{B}\ C\ D$
0	1	0	0	0	
0	1	0	1	0	
0	1	0	1	0	
0	1	1	1	1	$\overline{A}\ B\ C\ D$
1	0	0	0	0	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	$A\ \overline{B}\ C\ D$
1	1	0	0	1	$A\ B\ \overline{C}\ \overline{D}$
1	1	0	1	1	$A\ B\ \overline{C}\ D$
1	1	1	0	1	$A\ B\ C\ \overline{D}$
1	1	1	1	1	$A\ B\ C\ D$

```
Q =    (~A & ~B &  C &  D)
     | (~A &  B &  C &  D)
     | ( A & ~B &  C &  D)
     | ( A &  B & ~C & ~D)
     | ( A &  B & ~C &  D)
     | ( A &  B &  C & ~D)
     | ( A &  B &  C &  D);
```

2a: Logic Expression

Re-write the following schematic as two logic expressions, sum=? And cout=? (b) as VHDL (c) and convert schematic using only NORs.

sum = cin ^ (a ^ b) = cin XOR (a XOR b);

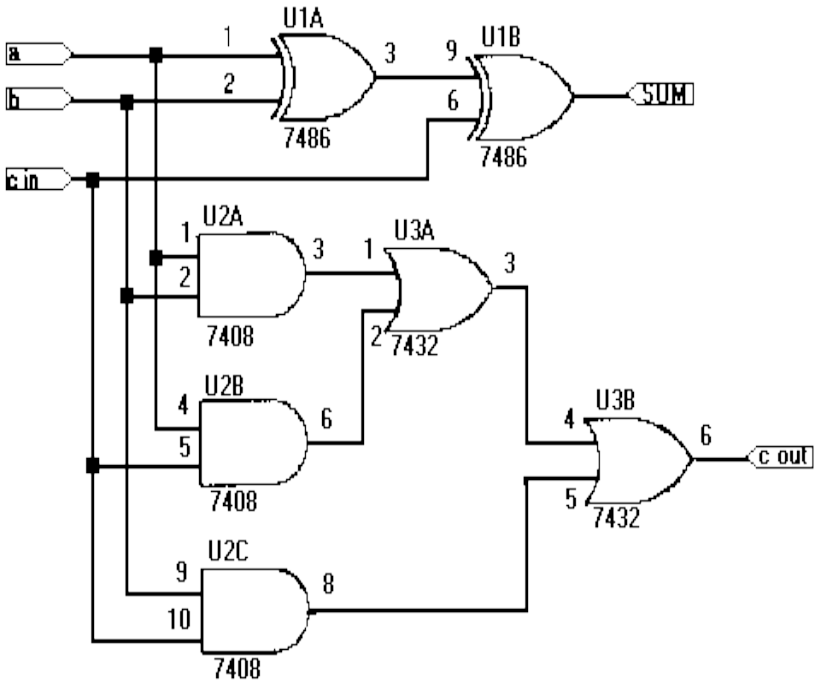


VHDL: cout = (((a AND b) OR (a AND cin)) OR (b AND cin));

C/C++: cout = (((a & b) | (a & cin)) | (b & cin));

2b: VHDL

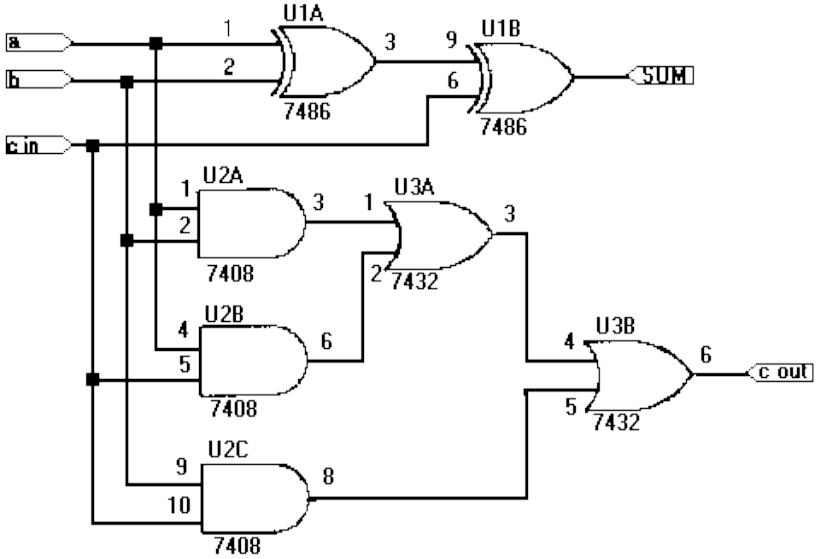
Re-write the following schematic as two logic expressions, sum=?
And cout=? (b) as VHDL (c) and convert schematic using only NORs.



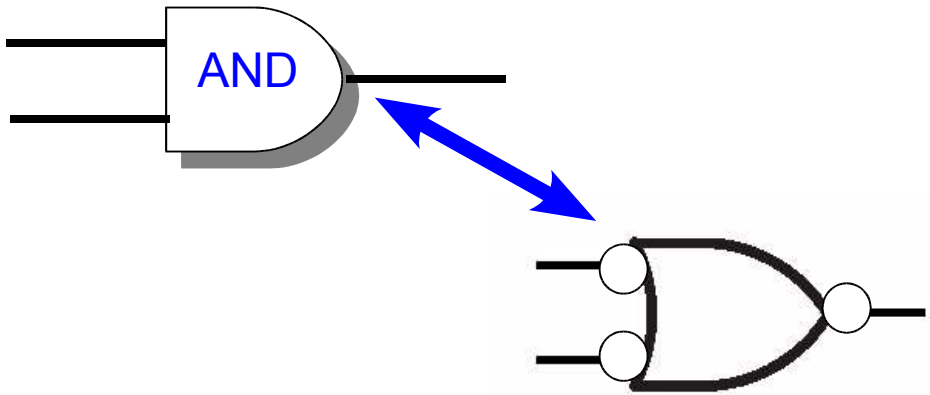

```
ENTITY P2 IS
  PORT ( a, b, cin: IN std_logic;
         Sum, Cout: OUT std_logic
  );
END P2;

ARCHITECTURE P2A OF  P2  IS

BEGIN

  sum <= cin XOR (a XOR b);

  cout <= ((a AND b) OR (a AND cin))

          OR (b AND cin);

END P2A;
```

2c: AND-to-NOR, NOT-to-NOR, OR-to-NOR Transforms

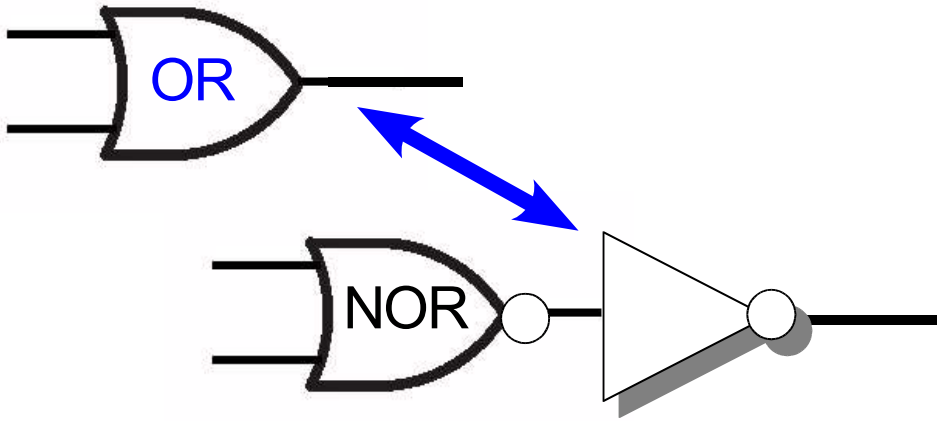
Re-write the following schematic as two logic expressions, sum=?
And cout=? (b) as VHDL (c) and convert schematic using only NORs.



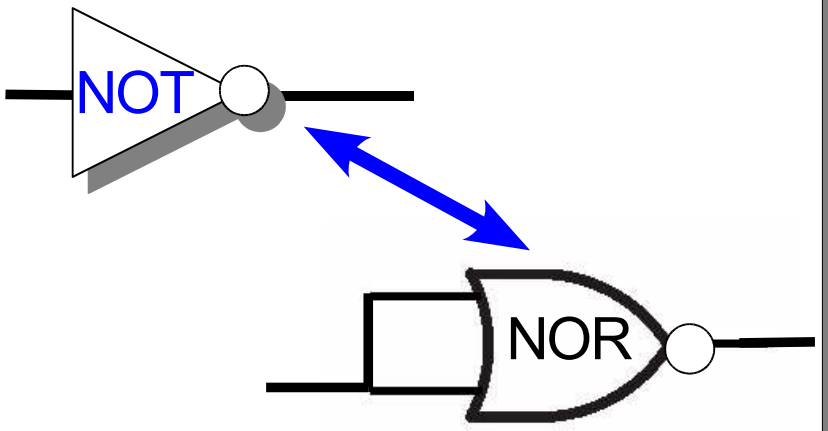
DeMorgans Law:



X = NOT(NOT x)):



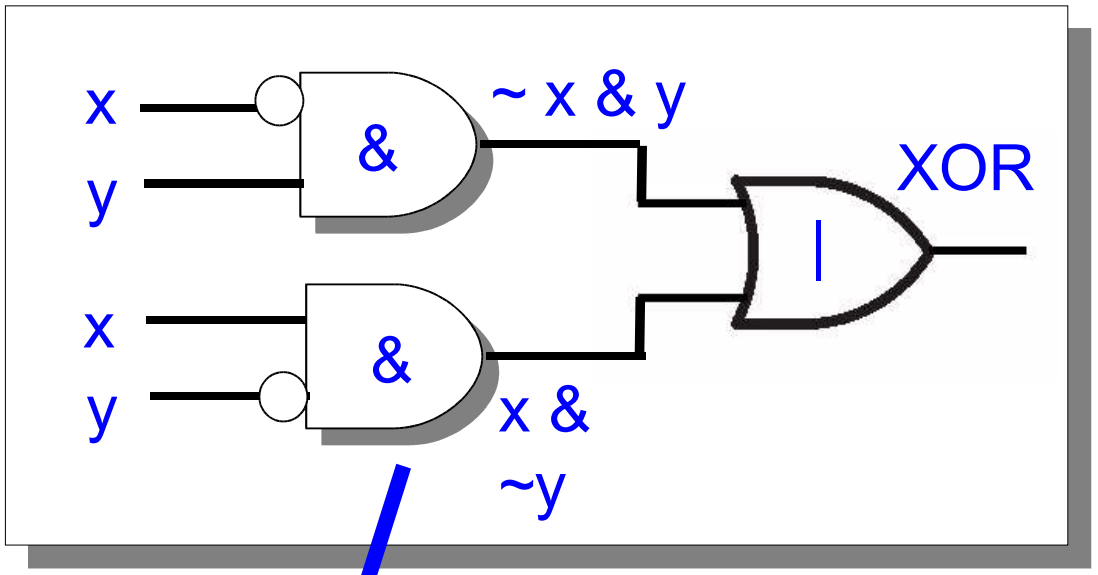
NOT(X) = NOT(X OR X) =X NOR X;



2c: XOR Transform

Truth Table: behavior

x	y	f_{xor}	XOR
0	0	0	
0	1	1	~x & y
1	0	1	x & ~y
1	1	0	

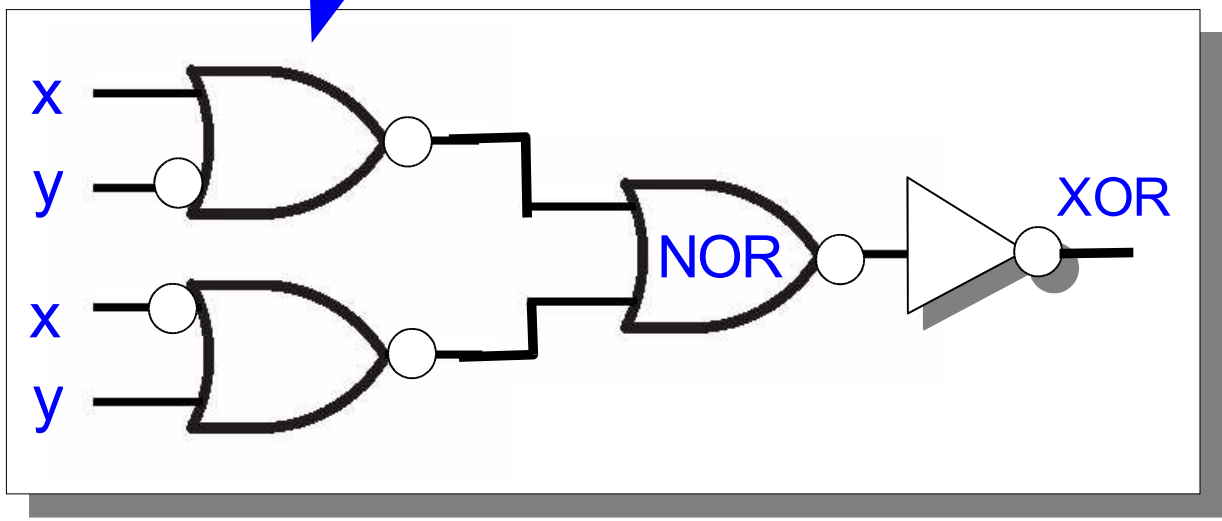


SOP

Gate Level

DeMorgan's Law

XOR = (~x & y) | (x & ~y)



2c: XOR Transform

Replace every XOR with the following structural circuit: