Q1: 15 points

First, begin with the truth table.

	A	B	C	Y
$\bar{A}\bar{B}\bar{C}$ ←	0	0	0	1
	0	0	1	0
	0	1	0	0
$\bar{A}BC$ ←	0	1	1	1
	1	0	0	0
$A\bar{B}C$ ←	1	0	1	1
$AB\bar{C}$ ←	1	1	0	1
	1	1	1	0

6 points (truth table and logic)

Write out your logic equation,

$$Y = AB\bar{C} + A\bar{B}C + \bar{A}BC + \bar{A}\bar{B}\bar{C}$$

We could go straight to implementing this into CMOS, but we want to simplify the expression to reduce and optimize our implementation.

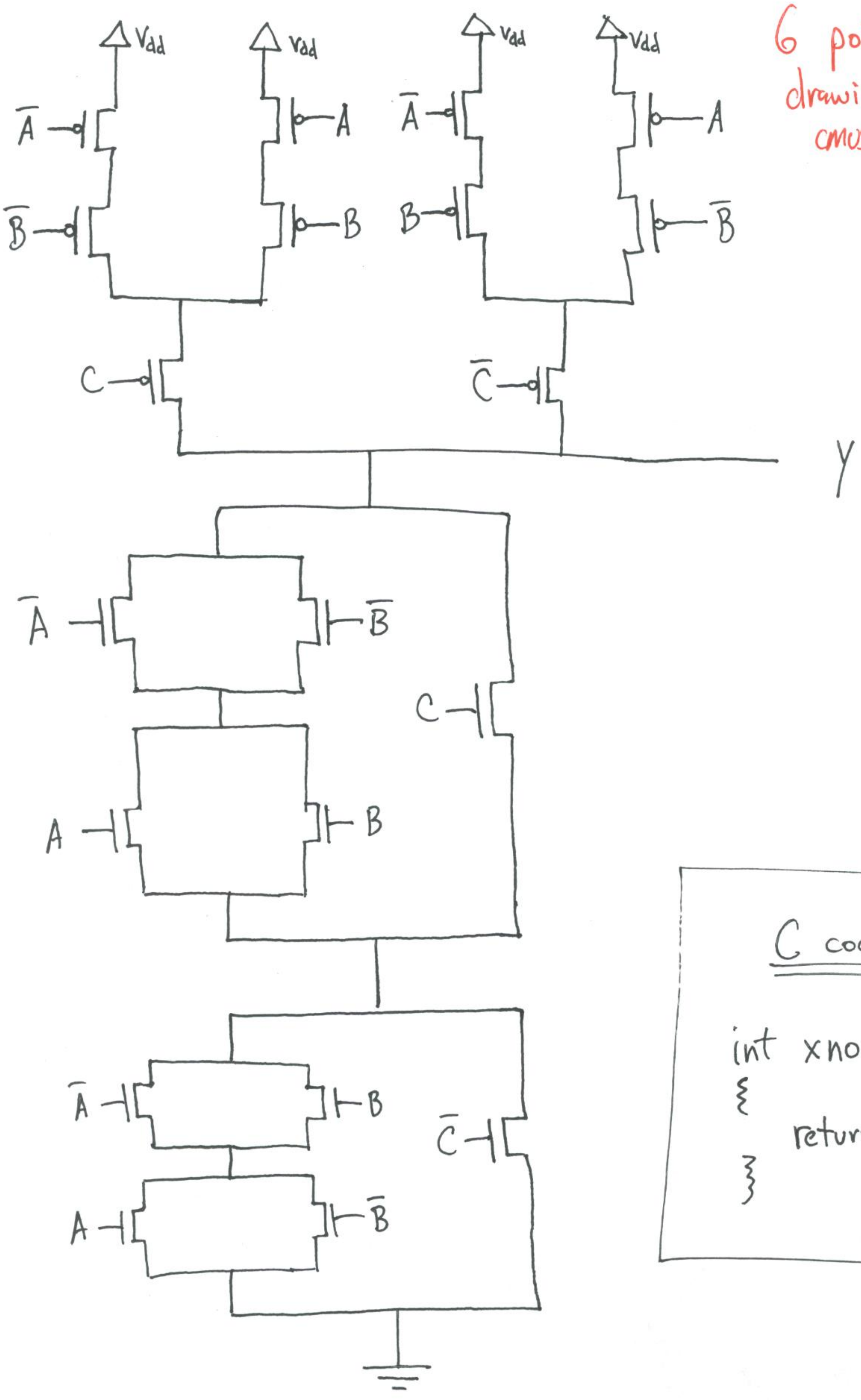
Using De Morgan's law:

$$Y = \overline{\overline{(AB+\bar{A}\bar{B})\cdot\bar{C} + (A\bar{B}+\bar{A}B)\cdot C}} = \overline{((AB+\bar{A}\bar{B})\cdot\bar{C})}\cdot\overline{((A\bar{B}+\bar{A}B)\cdot C)}$$

$$Y = \overline{(\overline{(AB+\bar{A}\bar{B})}+C)\cdot(\overline{(A\bar{B}+\bar{A}B)}+\bar{C})} = \overline{(\overline{(AB)}\circ\overline{(\bar{A}\bar{B})}+C)\cdot((\overline{A\bar{B}}\circ\overline{\bar{A}B})+C)}$$

$$Y = \overline{(((\bar{A}+\bar{B})\cdot(A+B))+C)\cdot(((\bar{A}+B)\cdot(A+\bar{B}))+\bar{C})}$$

Now everything is in terms of two terms and building up the transistor gates follows, replace ANDs with serial connections
ORs with parallel connections.



6 points
drawing out cmos

C code 3 points

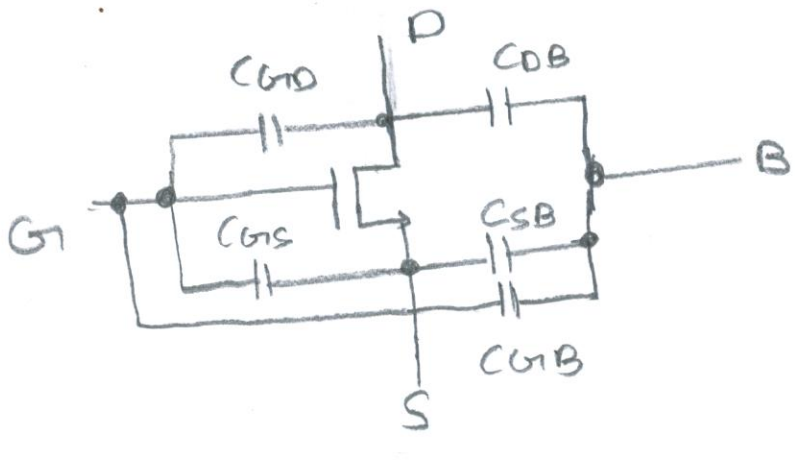
```
int xnor3(int A, int B, int C)
{
    return !(A^B^C);
}
```

2).

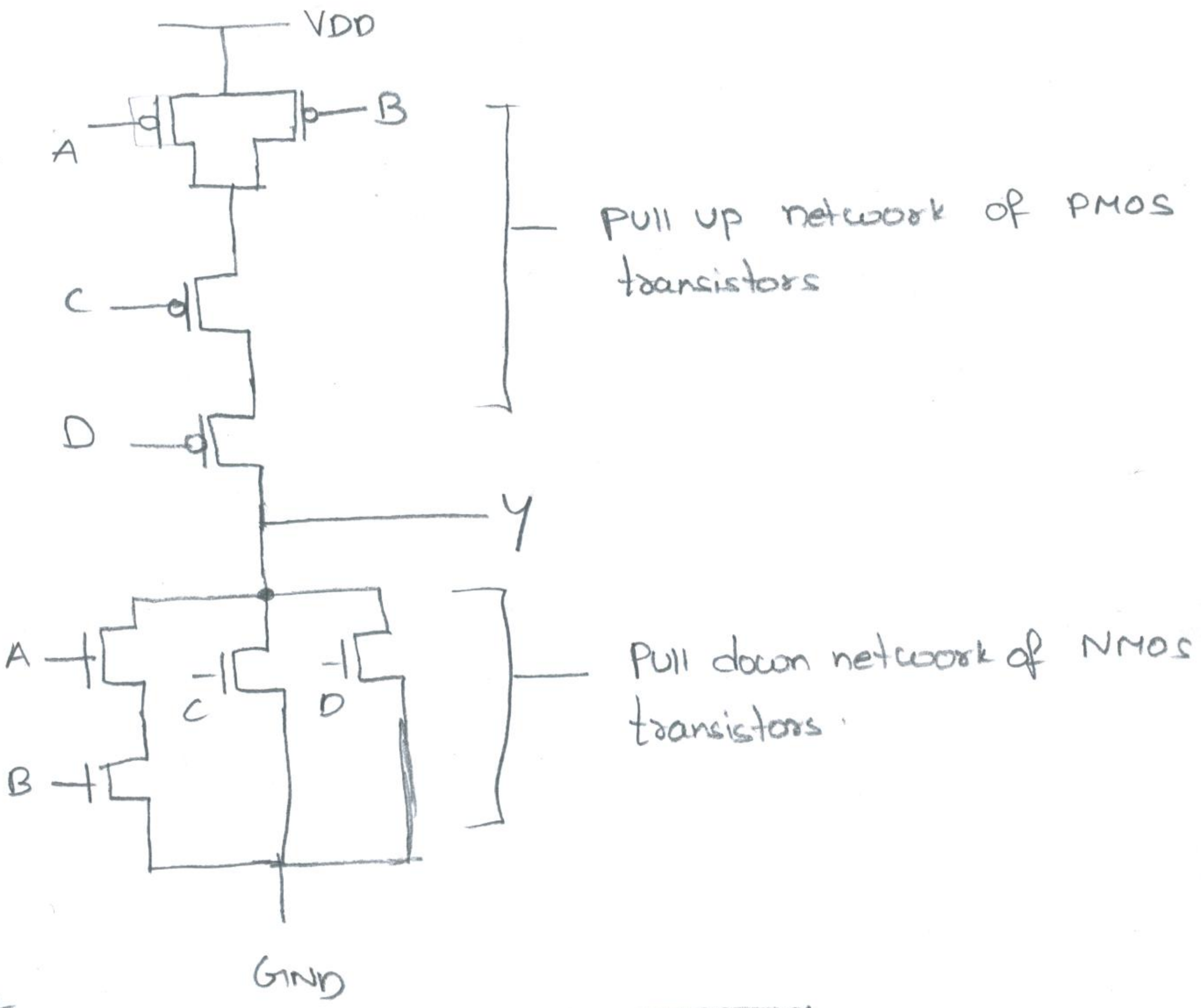
Delay in combinational circuits:- (5 points).

Delay in combinational Circuits is caused by the parasitic capacitance in Mos transistors.
The charging and discharging of capacitances in the internal nodes of the transistor due to the changes in the inputs and outputs is responsible for delay.
The W/L values for a transistor are factors in the individual capacitance values along with the oxide capacitance

A capacitive model for a mosfet



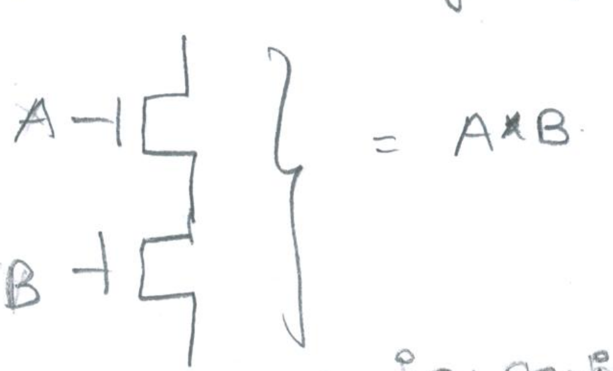
3).

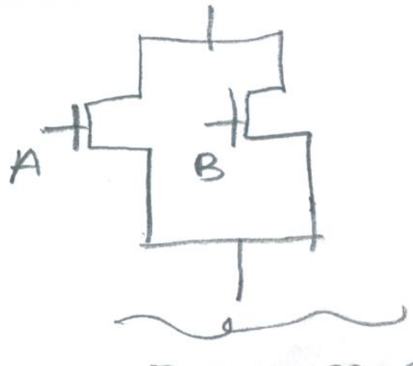

VDD
A
B
C
D
Y
Pull up network of PMOS transistors
A
B
C
D
Pull down network of NMOS transistors
GND

ANSWER:-

$\boxed{Y = \overline{(A*B)+C+D}}$ $\longrightarrow$ $\boxed{8 \text{points}}$

Pullup network and pull down network complement each other. Analyze pull down network





Two nmos connected ~~in~~ in series (cascode) implements "and" function

Two nmos connected in parallel implements "OR" A+B

Therefore, the pull down network implements.

$A \cdot B + C + D$. But, CMOS logic is Complimentary

So the final logic function implemented is

$$Y = \overline{(A \cdot B) + C + D}$$

C. Code:- $\boxed{2 \text{ points}}$

```
// Assuming A, B, C, D are either 0 or 1.
int booleanFunction (int A, int B, int C, int D){
    return !(A&B|C|D);
}
```