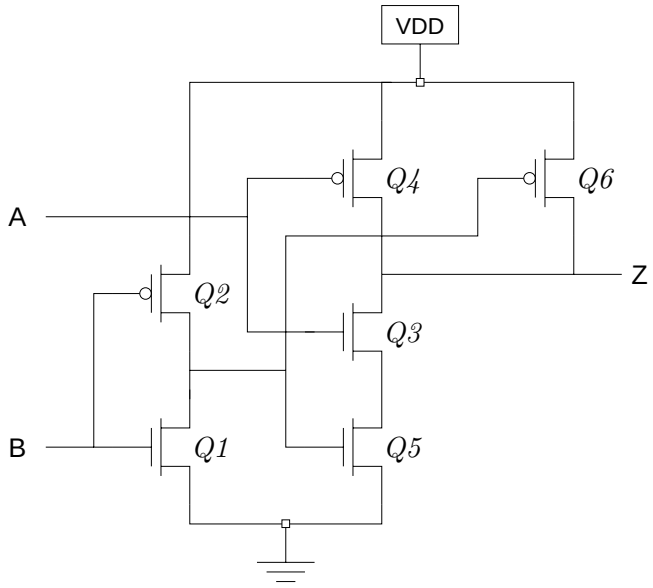
EE 121 May 7, 2002
Digital Design Laboratory Handout #21

Midterm Examination #1 Solutions

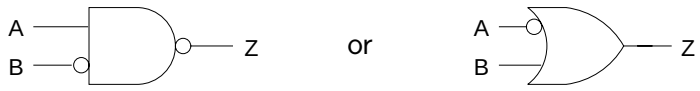
Open book, open notes. Time limit: 75 minutes

1. (20 points) *CMOS logic circuit.* Draw a circuit diagram, function table, and logic symbol in the style of *DDPP* Figure 3-19 for a CMOS gate with two inputs, A and B, and an output Z, where $Z = 0$ if $A = 1$ and $B = 0$, and $Z = 1$ otherwise. Hint: Six transistors.

   Circuit diagram: $Z = (A \cdot B')' = \mathsf{NAND}(A, B') = \mathsf{OR}(A', B)$



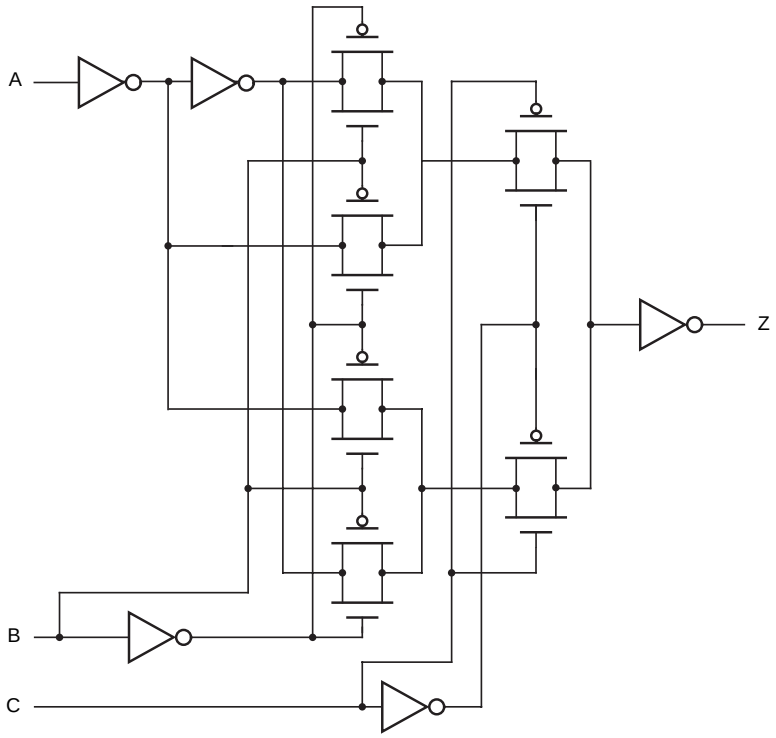
   Logic symbol:



   Function table:

A	B	*Q1*	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	Z
L	L	off	on	off	on	on	off	H
L	H	on	off	off	on	off	on	H
H	L	off	on	on	off	on	off	L
H	H	on	off	on	off	off	on	H

2. (20 points) *CMOS circuit analysis.* Write the truth table and a logic diagram for the logic function performed by the following CMOS circuit.

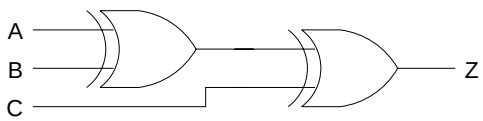


The four transmission gates on the left are pairs of multiplexers controlled by B; the upper transmission gates in each pair is enabled when B is 1. The output of the upper pair is A when $\mathsf{B} = 1$ and A' when $\mathsf{B} = 0$, in other words, $\mathrm{XNOR}(\mathsf{A}, \mathsf{B})$. The output of the lower pair is the complementary value, $\mathrm{XOR}(\mathsf{A}, \mathsf{B})$.

The rightmost two transmission gates form a multiplexer controlled by C; the lower transmission gate is enabled when C is 1. The output of these transmission gates is $\mathrm{XOR}(\mathsf{A}, \mathsf{B})$ when $\mathsf{C} = 1$ and $\mathrm{XNOR}(\mathsf{A}, \mathsf{B})$ when $\mathsf{C} = 0$. The output inverter complements this result. Thus

$$\mathsf{Z} = \mathsf{C} \cdot \mathrm{XNOR}(\mathsf{A}, \mathsf{B}) + \mathsf{C}' \cdot \mathrm{XOR}(\mathsf{A}, \mathsf{B}) = \mathrm{XOR3}(\mathsf{A}, \mathsf{B}, \mathsf{C})$$

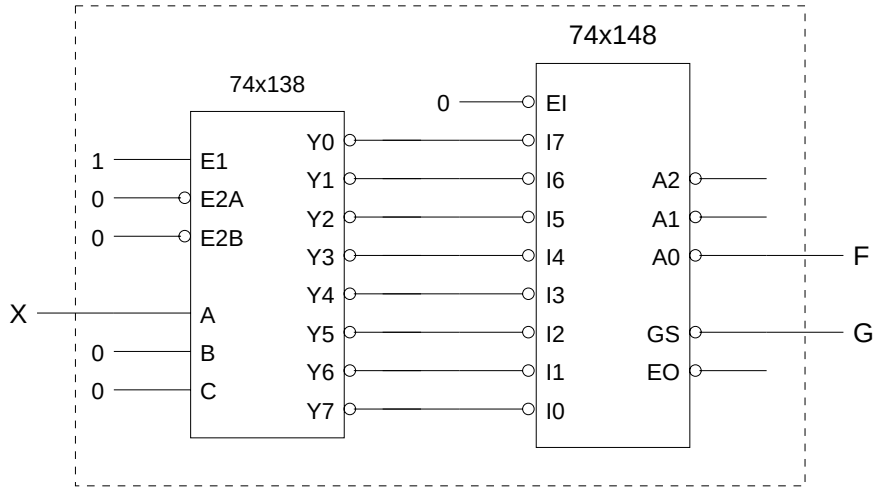
The function computed by the circuit is the exclusive-or of three inputs, $\mathrm{XOR3}(\mathsf{A}, \mathsf{B}, \mathsf{C})$. An equivalent logic circuit is shown below.



This is the actual circuit of an EO3 3-input XOR cell in LSI Logic's LCA 10000 series of CMOS gate arrays.

A	B	C	Z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

3. (15 points) *MSI components.* A logic designer with time on his hands connected a 74x138 decoder and a 74x148 priority encoder as shown below.



What logic functions are computed by F and G ?

F = X

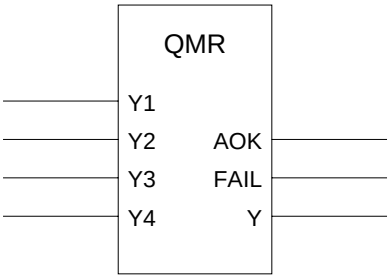
Depending on X, either Y0 or Y1 is active. Suppose X $= 0$. Then Y0 $= 0$. Since Y0 is connected to 74x148 input I7, the encoding is A2 A1 A0 $= 111$, but the output is 000 because the encoder has active low outputs. Thus F $= 0$. Similarly, if X $= 1$ then F $= 1$.

G $= 0$

One output of the 74x138 decoder is always active, hence one input of the 74x148 encoder is active, hence the group selective signal GS is always active. Since GS is active low, the constant value of G is 0.

Note: there are simpler circuits for these logic functions.

4. (30 points) *Circuit synthesis.* A common method for building fault tolerance into digital systems is *triple modular redundancy* (TMR), in which a logic function is computed by three independent circuits and the final output is chosen by majority vote of the outputs of the three circuits. the final output is incorrect when two or three of the circuits are wrong. In this problem, we consider the even more reliable *quadruple modular redundancy.*



The QMR circuit shown in the figure above analyzes the outputs $\mathsf{Y1}, \mathsf{Y2}, \mathsf{Y3}, \mathsf{Y4}$ of four independent circuits for the same function.

- If all four values are the same, then AOK is true. (One use of AOK is to measure the failure rate of the circuits that compute $\mathsf{Y1}, \mathsf{Y2}, \mathsf{Y3}, \mathsf{Y4}$.)
- If there is no clear majority value for the four inputs—that is, two of $\mathsf{Y1}, \mathsf{Y2}, \mathsf{Y3}, \mathsf{Y4}$ are 0 and two are 1—then FAIL is true, which indicates that QMR cannot make a decision.
- When there is no failure, the output Y is the majority vote of the inputs $\mathsf{Y1}, \mathsf{Y2}, \mathsf{Y3}, \mathsf{Y4}$.

Write optimized Boolean equations (minimal sums of products) for each of the outputs of the QMR circuit. Your equations for Y should be simplified by using the fact that the value of Y does not matter when FAIL is true.

$$\mathsf{AOK} = \mathsf{Y1} \cdot \mathsf{Y2} \cdot \mathsf{Y3} \cdot \mathsf{Y4} + \mathsf{Y1}' \cdot \mathsf{Y2}' \cdot \mathsf{Y3}' \cdot \mathsf{Y4}'$$

$$\mathsf{FAIL} = \mathsf{Y1}' \cdot \mathsf{Y2}' \cdot \mathsf{Y3} \cdot \mathsf{Y4} + \mathsf{Y1}' \cdot \mathsf{Y2} \cdot \mathsf{Y3}' \cdot \mathsf{Y4} + \mathsf{Y1}' \cdot \mathsf{Y2} \cdot \mathsf{Y3} \cdot \mathsf{Y4}' + \mathsf{Y1} \cdot \mathsf{Y2}' \cdot \mathsf{Y3}' \cdot \mathsf{Y4} + \mathsf{Y1} \cdot \mathsf{Y2}' \cdot \mathsf{Y3} \cdot \mathsf{Y4}' + \mathsf{Y1} \cdot \mathsf{Y2} \cdot \mathsf{Y3}' \cdot \mathsf{Y4}'$$

$$\mathsf{Y} = \mathsf{Y1} \cdot \mathsf{Y2} + \mathsf{Y3} \cdot \mathsf{Y4} = \mathsf{Y1} \cdot \mathsf{Y3} + \mathsf{Y2} \cdot \mathsf{Y4} = \mathsf{Y1} \cdot \mathsf{Y4} + \mathsf{Y2} \cdot \mathsf{Y3}$$

Karnaugh maps for AOK, FAIL, and Y are shown below.

Y3 Y4 \ Y1 Y2	00	01	11	10
00	1			
01				
11			1	
10				

Y3 Y4 \ Y1 Y2	00	01	11	10
00			1	
01		1		1
11	1			
10		1		1

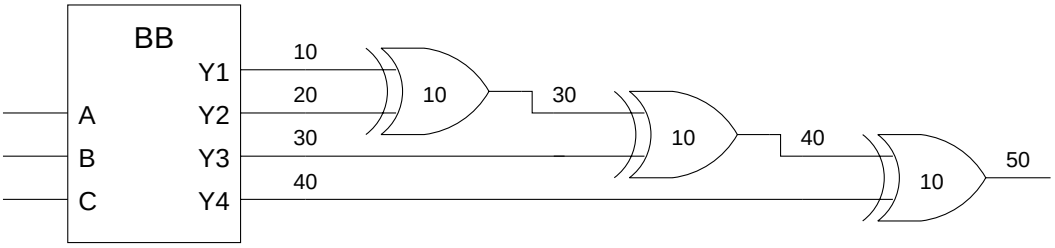
Y3 Y4 \ Y1 Y2	00	01	11	10
00			x	
01		x	1	x
11	x	1	1	1
10		x	1	x

The sum of products for Y can be simplified in several ways using the don't care squares corresponding to FAIL $= 1$. Once it is known that at most one $\mathsf{Y}i$ is incorrect, any two of the others give the correct answer. Without don't cares, four product terms are needed.

5. (20 points) XOR *trees.* The combinational logic propagation delays for the component shown below are different for the four outputs, as shown in the following table.

From (input)	To (output)	Max
Any	Y1	10 ns
Any	Y2	20 ns
Any	Y3	30 ns
Any	Y4	40 ns

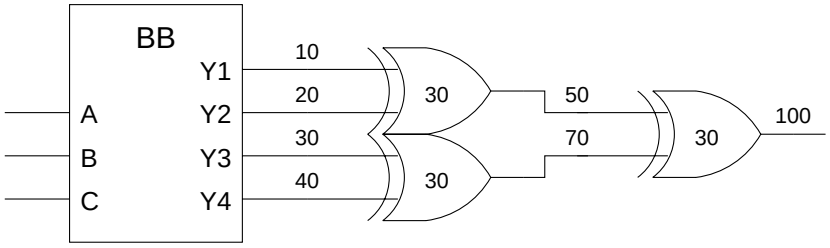
a. Suppose that the propagation delay of an XOR gate is 10 ns. Connect three 2-input XOR gates to compute Y = XOR(Y1, Y2, Y3, Y4) so that the propagation delay from inputs A, B, C to output Y is minimized. What is this minimum propagation delay?



The fastest XOR tree is obtained by combining signals as they arrive or are computed. This "greedy" algorithm can be proven to produce the fastest circuit in general. In this case, the XOR gates are fast compared to the difference in arrival times of the signals Y1 to Y4, so Y1 to Y4 are combined one at a time as they arrive.

Propagation delay (XOR delay 10 ns): 50 ns

b. Suppose that the propagation delay of an XOR gate is 30 ns. Connect three 2-input XOR gates to compute Y = XOR(Y1, Y2, Y3, Y4) so that the propagation delay from inputs A, B, C to output Y is minimized. What is this minimum propagation delay?



In this case, the XOR delay is large compared to the difference in arrival times, so a balanced XOR tree is optimal.

Propagation delay (XOR delay 30 ns): 100 ns