Problem 1 (3 points) Design a sequential network that implements a finite-state machine with states A B C D, input X, 1-bit output, and the following transition table. Use *only* D flip-flops (DFFs) and NAND gates.

Present state	Next state / Output	
	X=0	X=1
A	B/0	A/1
B	C/1	C/0
C	D/1	B/1
D	A/0	D/1

State encoding

	Q_1	Q_0
A	0	0
B	0	1
C	1	0
D	1	1

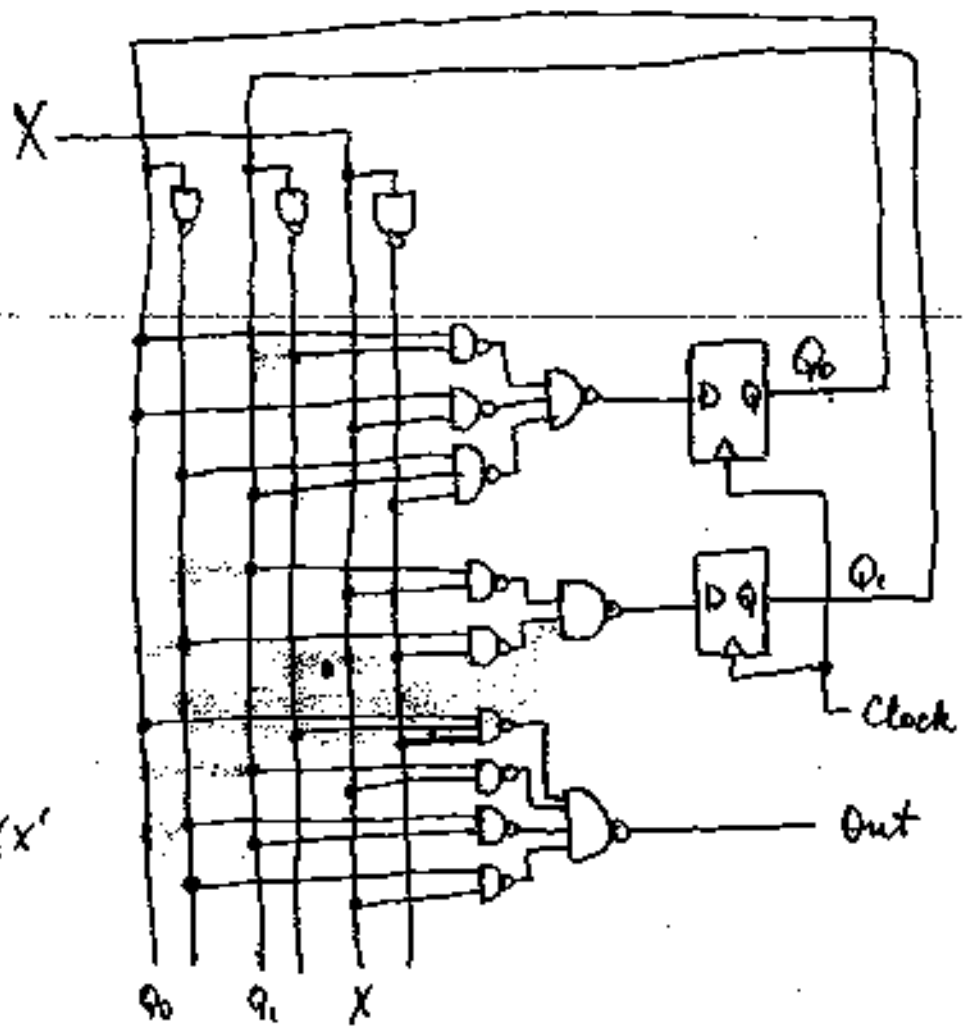
Q_1 Q_0 / X	00	01	11	10
0	01/0	10/1	00/0	11/1
1	00/1	10/0	11/1	01/1

(Q_1 next Q_0 next/Out)

$Q_{0\,next} = Q_0Q_1' + Q_0X + Q_0'Q_1X'$

$Q_{1\,next} = Q_1X + Q_0'X'$

$Out = Q_1X + Q_0'Q_1 + Q_0'X + Q_0Q_1'X'$



Problem 2 (4 points) Implement the following function:

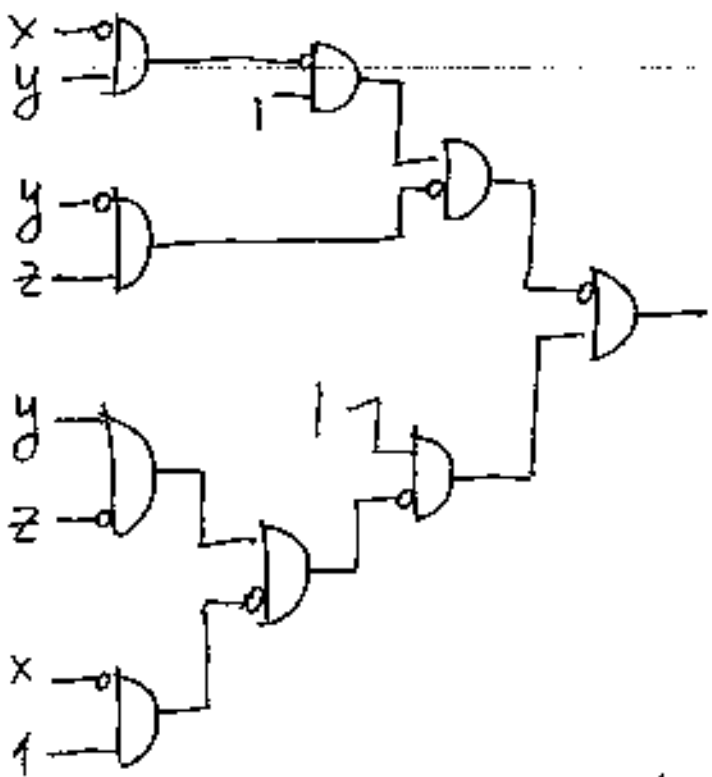
$$f(X,Y,Z)=X'Y+Y'Z+XYZ'$$

using *only* BILL gates, where a BILL gate is defined by the following truth table:

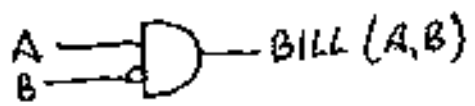
A	B	BILL(A,B)
0	0	0
0	1	0
1	0	1
1	1	0

Adopt an arbitrary reasonable symbol for the BILL gate. You are allowed to use constants 0 and 1 as inputs to your gates as needed.

$f(x,y,z)=(x'y+y'z)''+xyz'=((x'y)'\cdot(y'z)')'+xyz'=$
$(x'y)'\cdot(y'z)'\cdot(xyz')'$



where $BILL(A,B)=AB'$

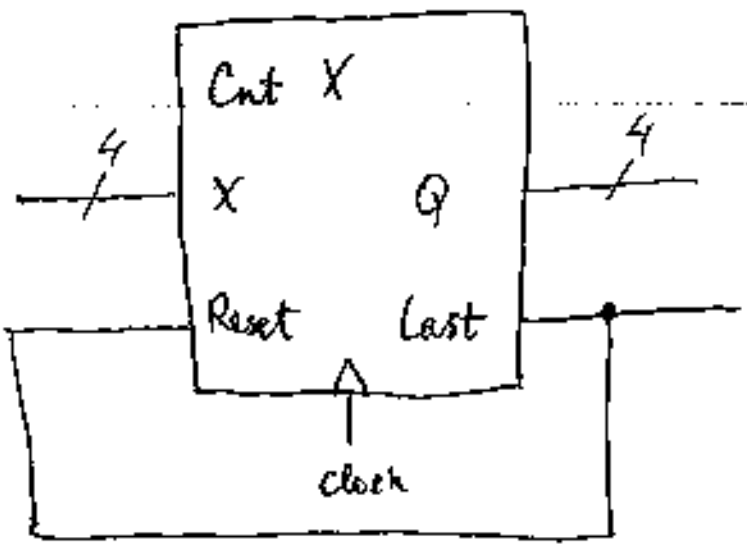


Problem 3 (5 points total) Design an Iterative Fibonacci Number Counter. It is a 4-bit counter that counts iteratively from 0 to A=fib(i), i=1,..,7, where for every iteration i the "counter deadline" A is the i^{th} Fibonacci number fib(i). (Recall that fib(i)=fib(i-1)+fib(i-2), with fib(1)=fib(2)=1.) Reset the counter deadline A back to fib(1) after counting from 0 to fib(7)=13. In other words, your counter should produce the following sequence of outputs:

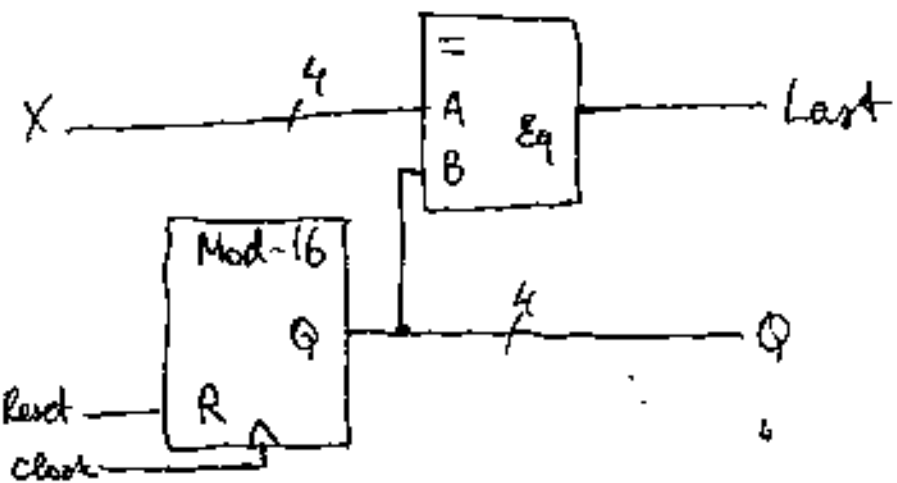
```
0 1
0 1
0 1 2
0 1 2 3
0 1 2 3 4 5
0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 1
0 1
0 1 2
etc.
```

You are allowed to use a Modulo-16 counter (i.e., a 4-bit binary counter) with a reset signal. Follow these three steps:

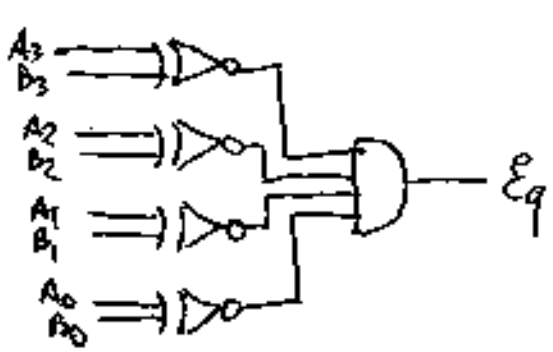
a) (1 point) Design a 4-bit counter that counts from zero to a predetermined value X, where X is a 4-bit input to the logic and 0<X<15.



where "Cnt X" is:

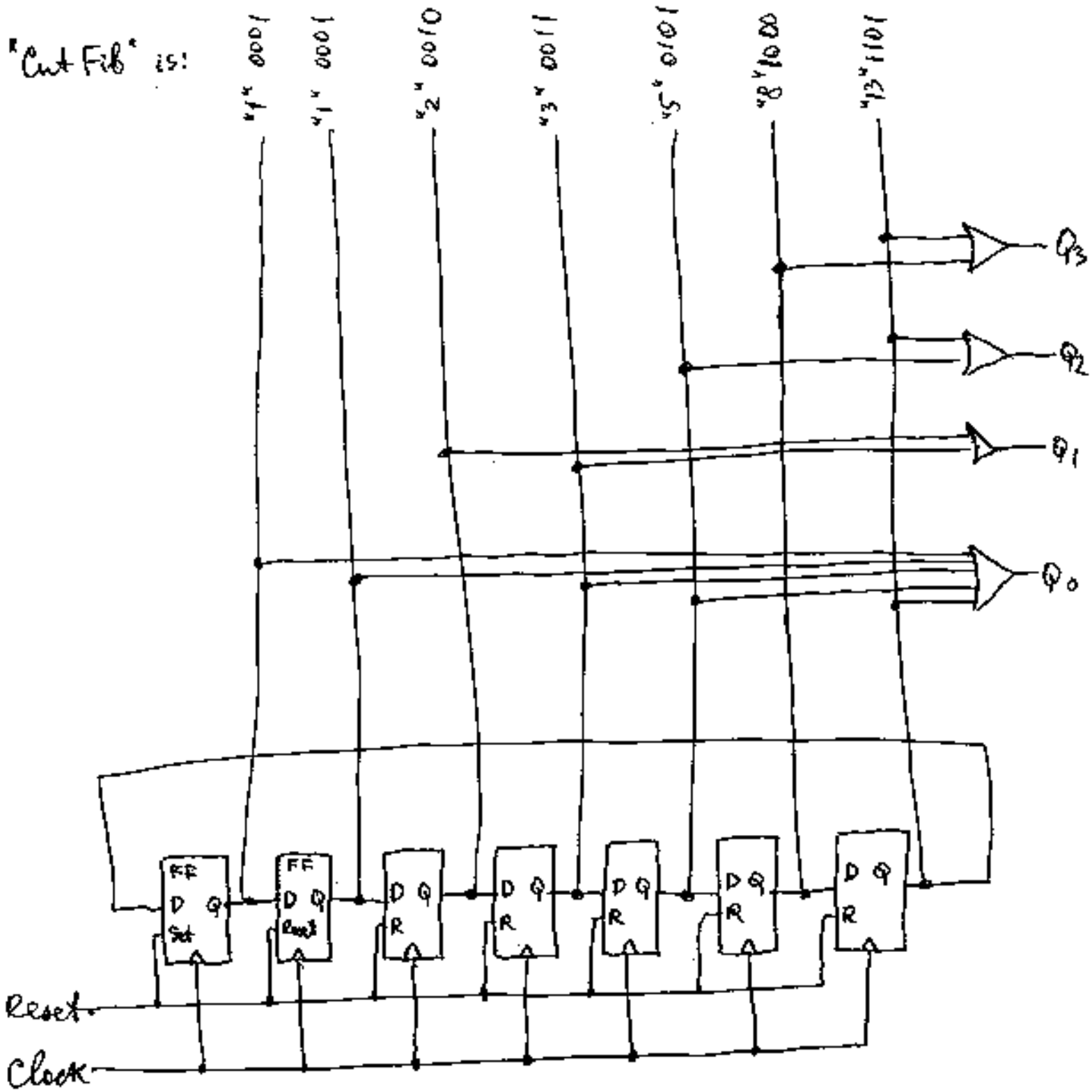


"=" is:

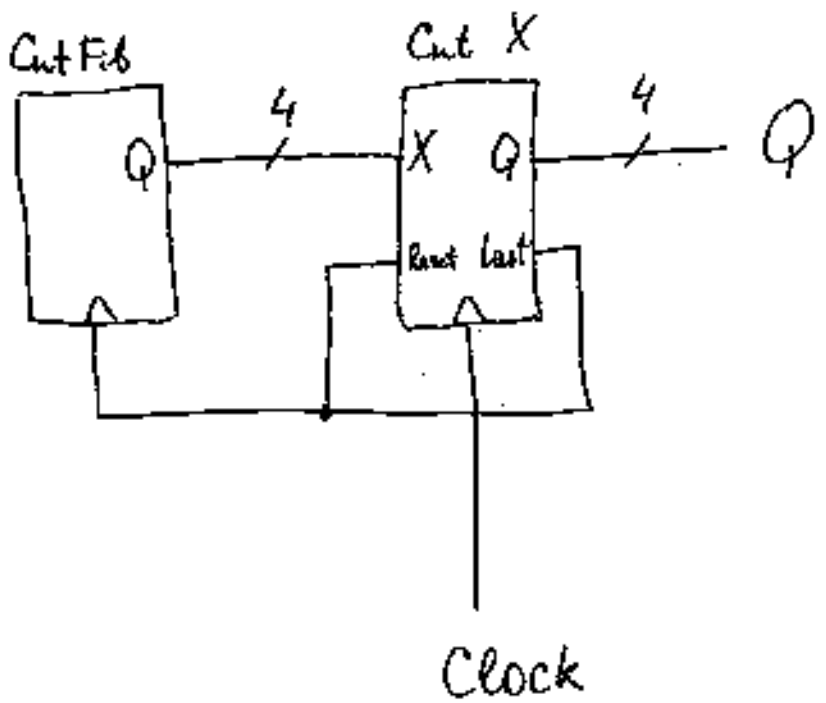


b) **(2 points)** Design an FSM that loops through the first 7 Fibonacci numbers.

The first seven Fibonacci numbers are: 1 1 2 3 5 8 13.

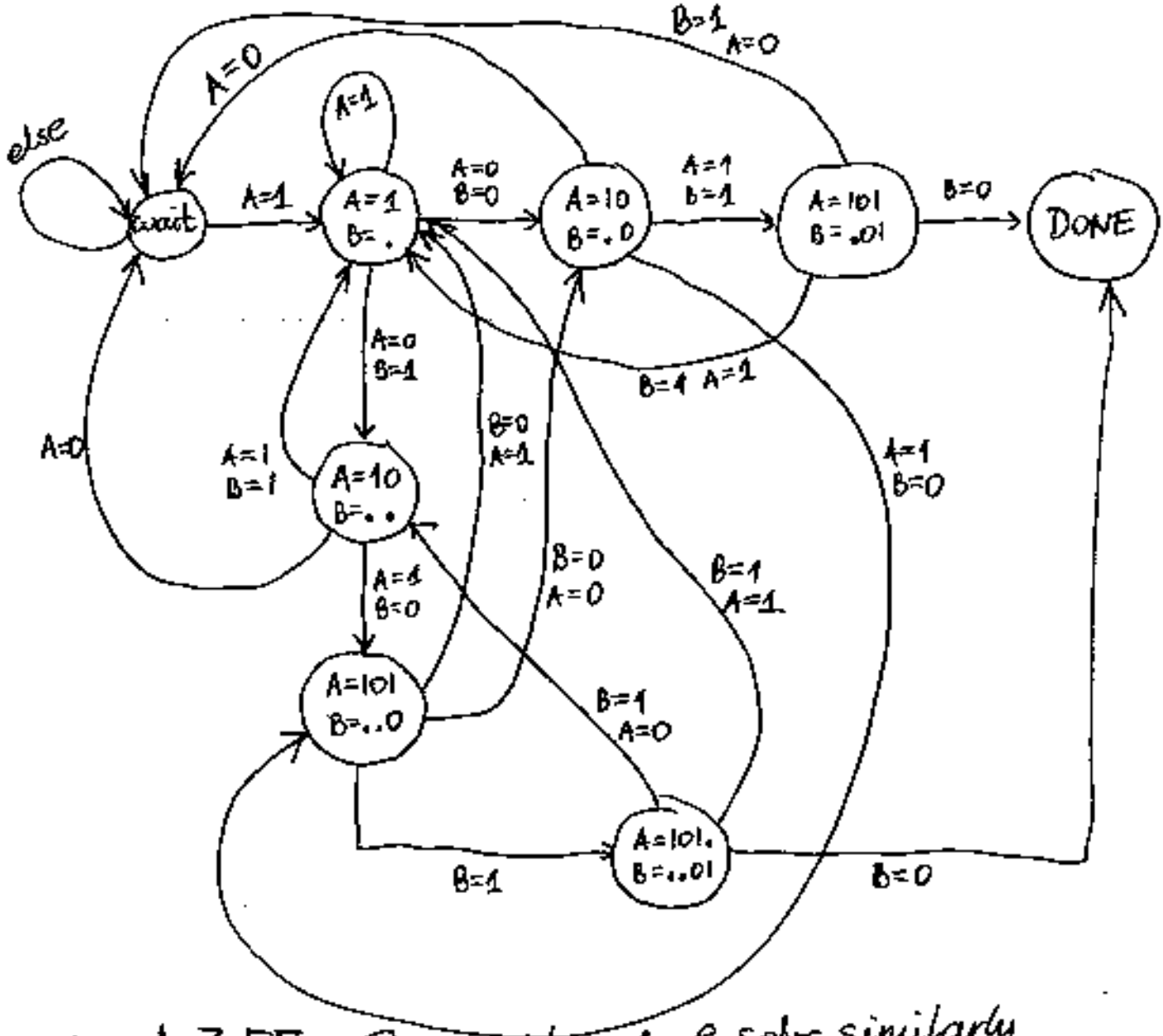


c) (2 points) Design the final product, the Iterative Fibonacci Number Counter, by combining the circuitry from sub-problems a) and b).



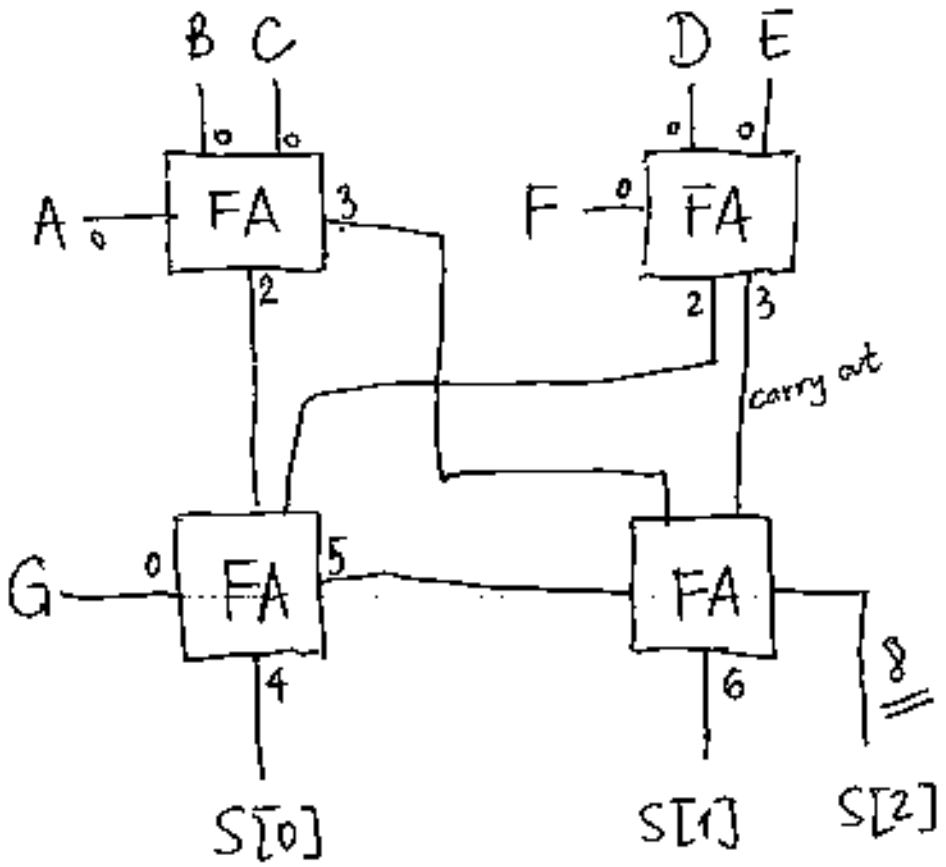
Problem 4 (7 points) Imagine that your investment portfolio is not doing that great, thus you are fairly depressed. However one night in your dreams your guardian angel pops up and tells you that if the least significant bit of the price of stock A equals 101 during three consecutive price ticks (t1,t2,t3) respectively, and the least significant bit of the price of stock B equals 010 in three consecutive price ticks starting either at t2 or t3 (so B(t2,t3,t4)=010 or B(t3,t4,t5)=010), you should buy call options of company C which is the first company after time t5 to announce news. This should make you as of tomorrow a devastatingly wealthy person. Make the money (generate the detection signal that will trigger you to seek for news on any company) using a minimal number of flip flops. Your guardian angel is not an insider to companies ABC nor is it affiliated with them in any other way.

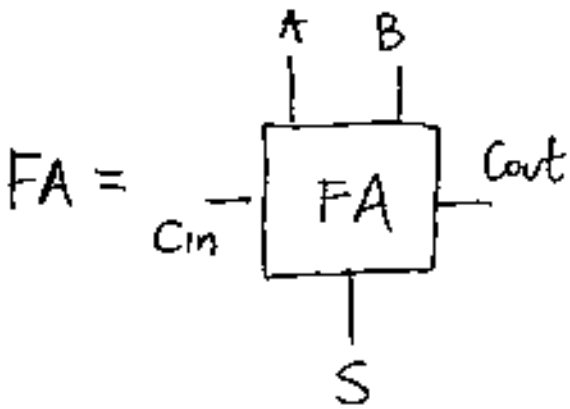
Assumption: you have a clock signal that ticks at times t1, t2, ... when price signals for company A and B are arriving.



You need 3 FFs. See problem 1 & solve similarly.
no need to optimize, just plug in the functionality of each FF as a MUX.

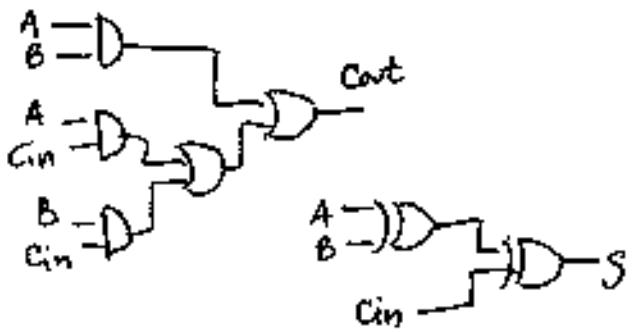
Problem 5 (4 points) Design an adder that takes as input seven 1-bit numbers A,B,C,D,E,F,G and produces a 3-bit count of 1s among the inputs, in other words, S=sum(A,B,C,D,E,F,G). Use FOUR one-bit full adders. What is the gate delay of this adder? Assume all gates have the same delay. Use only 2-input gates.





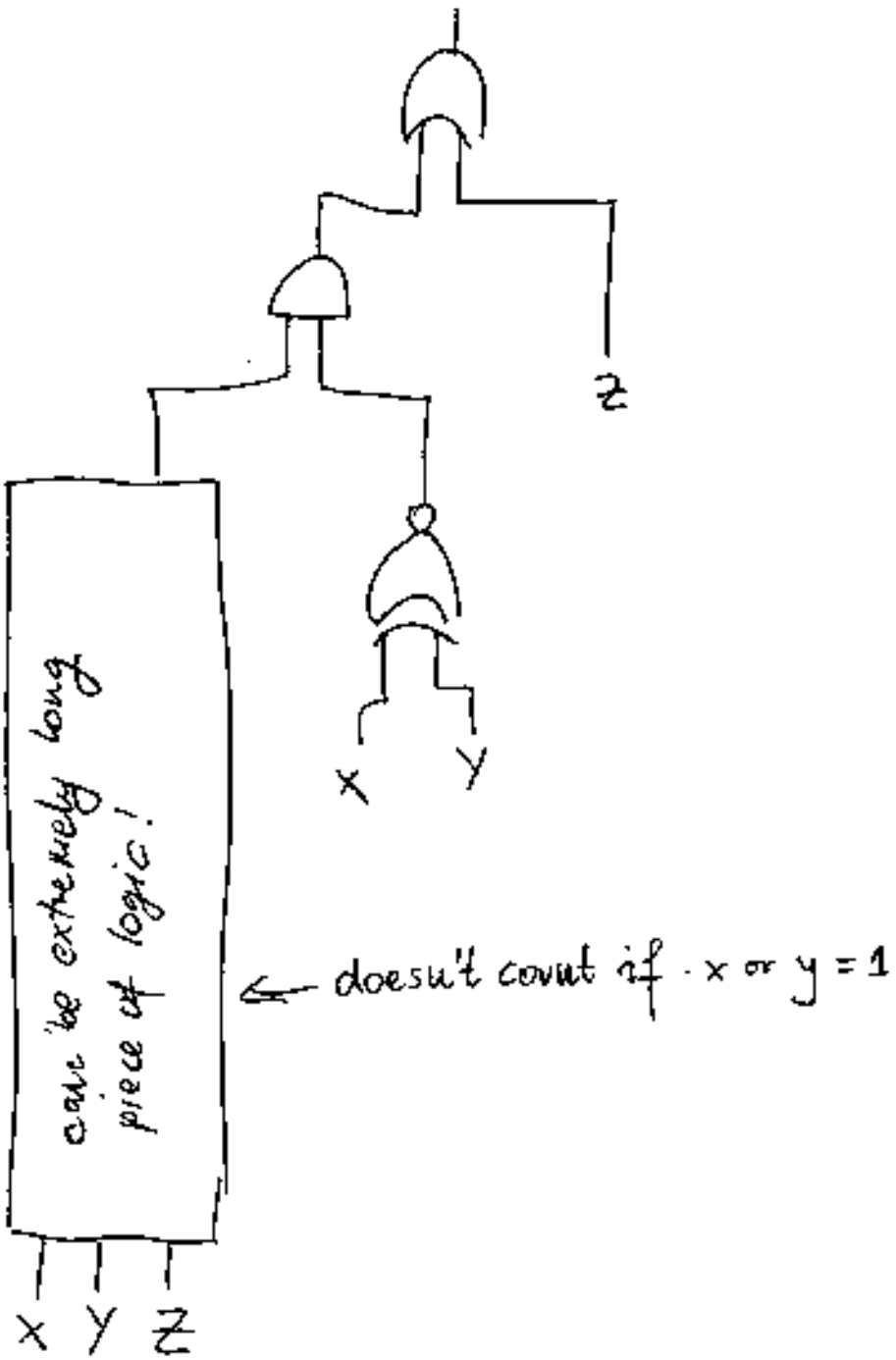
$S = A \oplus B \oplus Cin$

$Cout = AB + ACin + BCin$



Problem 6 (4 points) Demonstrate on an example how an implementation of a Boolean function f(X,Y,Z) can have dramatically different critical paths for the following two cases:

- when all possible values for X Y Z are allowed, and
- when the case X xor Y = 1 is not allowed as an input combination.



← doesn't count if x or y = 1

Problem 7 (7 points) Three 1-bit inputs P, Q, and R are given. Using *as few as possible* **2-1** multiplexers, create a combinational circuit that creates output ABC as one of the 6 possible permutations of PQR based on a 3-bit selection input XYZ. For example, if XYZ=000, pick ABC=PQR; if XYZ=001, pick ABC=PRQ, etc. Use any (most convenient) encoding XYZ for a particular permutation, as long as all 6 permutations are implemented.

Grading note: a full credit solution uses seven 2-1 multiplexers.