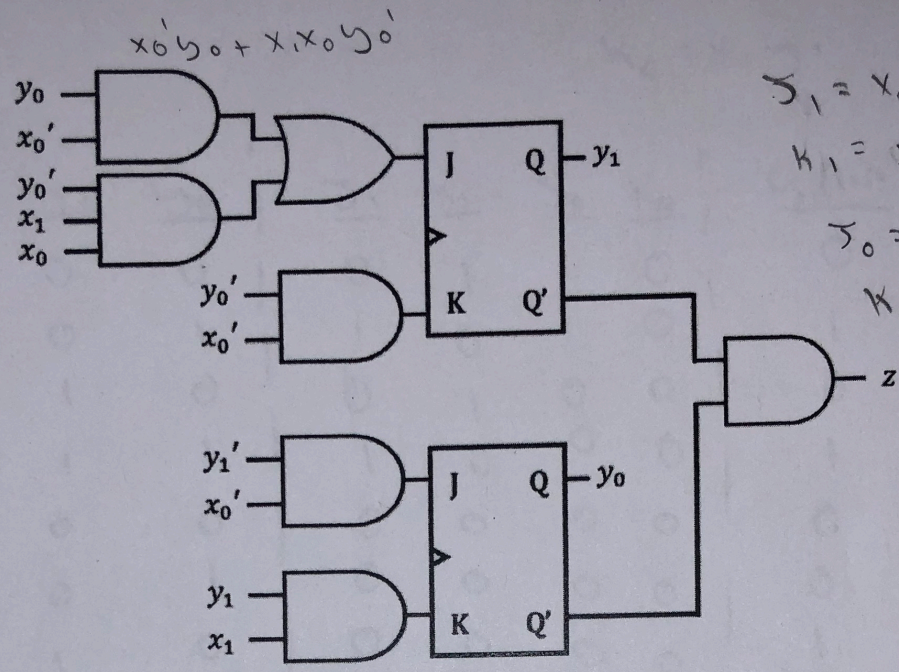
Problem 1 (20 points)

Obtain a high level description (state transition table) of the network shown in the figure below. The system has two input bits x1 and x0 with output bit z.



$J_1 = x_0'y_0 + x_1x_0y_0'$

$K_1 = x_0'y_0'$

$J_0 = x_0'y_1'$

$K_0 = x_1y_1$

(y_1y_0) PS	(input x_1x_0) 00	01	10	11
00	01/1	00/1	01/1	10/1
01	11/0	01/0	11/0	01/0
10	00/0	10/0	00/0	10/0
11	11/0	11/0	10/0	10/0

(NS $y_1(t+1)y_0(t+1)$, z)

* The output depends on y_1 and y_0, not $y_1(t+1)$ and $y_0(t+1)$, as described by TA before exam

Problem 1 Extra Page

$J_1 = x_0' y_0 + x_1 x_0 y_0'$

$K_1 = x_0' y_0'$

$J_0 = x_0' y_1'$

$K_0 = x_1 y_1$

x_1	x_0	y_1	y_0	J_1	K_1	J_0	K_0	$y_1(t+1)$	$y_0(t+1)$	z
0	0	0	0	0	1	1	0	0	1	1
0	0	0	1	1	0	1	0	1	1	0
0	0	1	0	0	1	0	0	0	0	0
0	0	1	1	1	0	0	0	1	1	0
0	1	0	0	0	0	0	0	0	0	1
0	1	0	1	0	0	0	0	0	1	0
0	1	1	0	0	0	0	0	1	0	0
0	1	1	1	0	0	0	0	1	1	0
1	0	0	0	0	1	1	0	0	1	1
1	0	0	1	1	0	1	0	1	1	0
1	0	1	0	0	1	0	1	0	0	0
1	0	1	1	1	0	0	1	1	0	0
1	1	0	0	1	0	0	0	1	0	1
1	1	0	1	0	0	0	0	0	1	0
1	1	1	0	1	0	0	1	1	0	0
1	1	1	1	0	0	0	1	1	0	0

Problem 2 (20 points) 20

Design a state transition table such that it initially has 8 states, and after minimization, reduces down to 3 states.

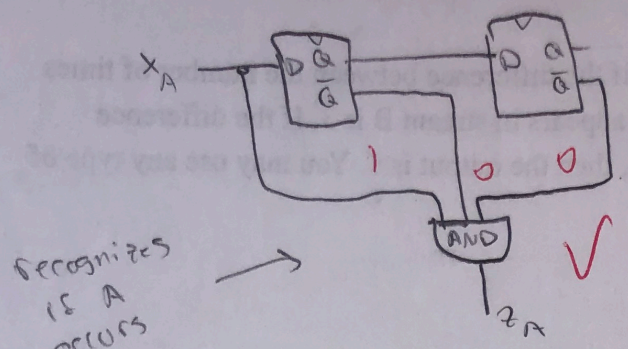
PS	0	1
A	A, 0	A, 0
B	A, 0	A, 0
C	A, 0	A, 0
D	A, 0	A, 1
E	A, 0	A, 1
F	A, 1	A, 0
G	A, 1	A, 0
H	A, 1	A, 0

nice

reduces to

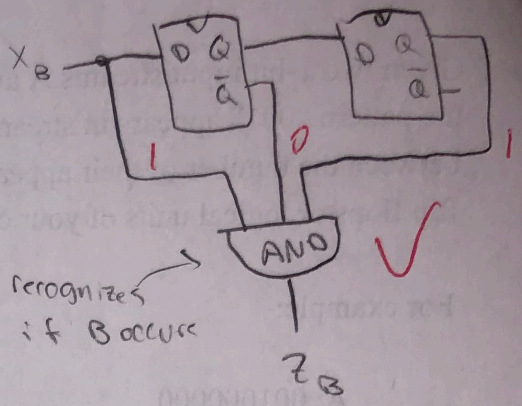
PS	0	1
A	A, 0	A, 0
D	A, 0	A, 1
F	A, 1	A, 0

A:



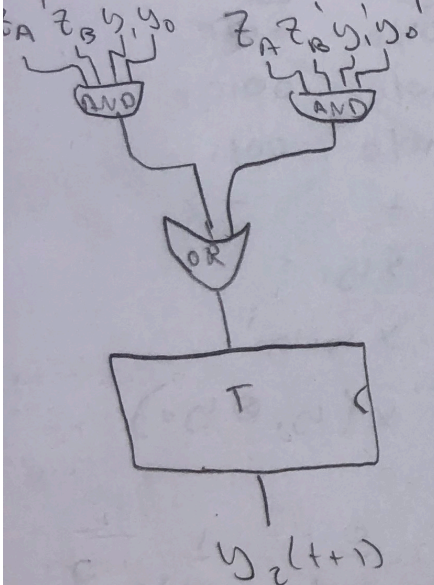
recognizes if A occurs

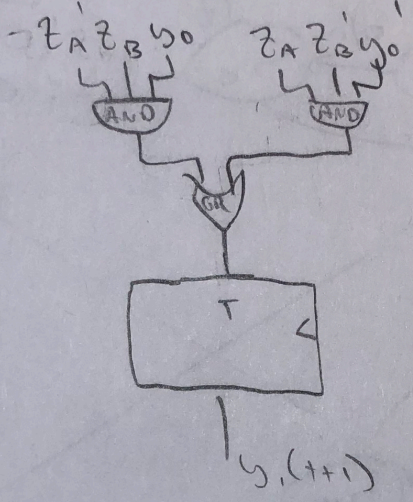
B:

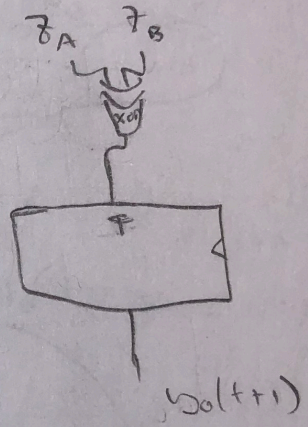


recognizes if B occurs

what are these from?

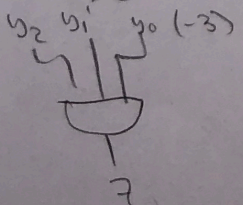






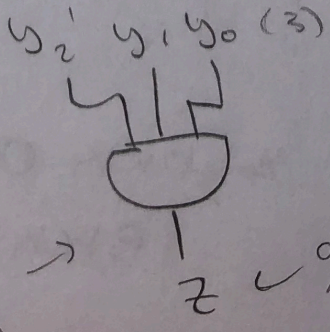
*Assumes difference not greater than 8.

Some arbitary limit must be imposed and I chose 8.



Works for A>B

Works for B>A



outputs 1 if difference is three (for B>A)

9 —— for op. missing
+7 not optimized
16 Kmaps not correct

Problem 4 (20 points)

Using **OK flip-flops** as designed below and **multiplexers** for logic, design a **minimum** system which has the following behaviors:

Input set: {a, b, c} {a, b, c, d}

Output: 1 if x(t-n, t) = a[b|c]+d*a

0 otherwise

$Y_2(t+1) = x_1' x_0 y_1' y_0 + x_1 x_0' y_1' y_0 + x_0 y_1 y_0' + x_1 y_1 y_0' + x_1 x_0 y_1$

Notes:

Overlaps can occur. For example adada would output 00101

$Y_1(t+1) = x_1' x_0' + x_1 x_0 y_1$

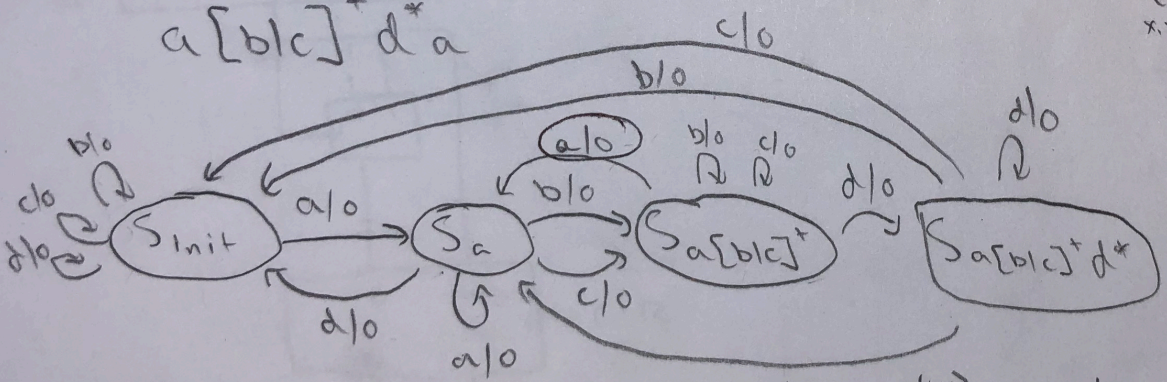
| means OR

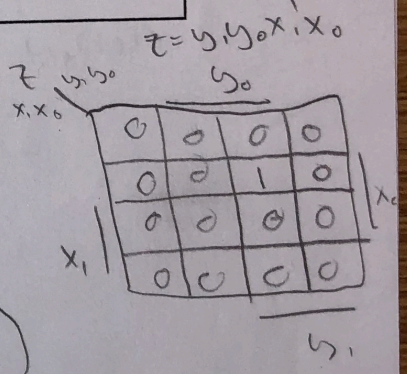
* means 0 or more of the previous character

+ means 1 or more of the previous character

$D(x) = OK(x', x)$

Prev State Q(t)	OK			
	00	**01**	**10**	**11**
0	1	1	0	-
1	-	1	0	0
	Nxt State Q(t+1)			



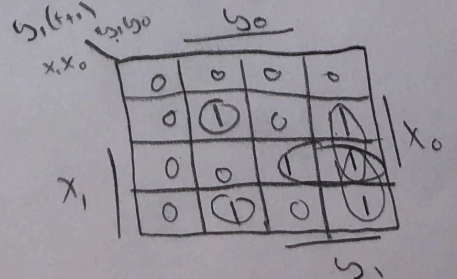


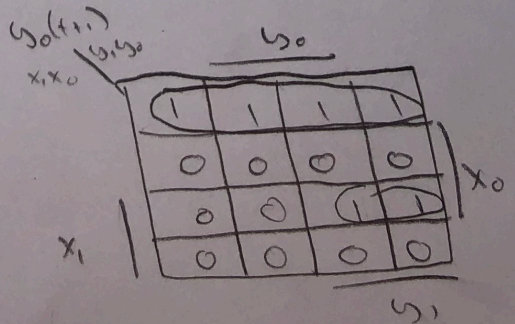
don't cares

(y_1, y_0) PS	$\overset{a}{00}$	$\overset{b}{01}$	$\overset{c}{10}$	$\overset{d}{11}$
00	01/0	00/0	00/0	00/0
01	01/0	10/0	10/0	00/0
10	01/0	10/0	10/0	11/0
11	01/1	00/0	00/0	11/0

(x_1, x_0)

(NS, Z)





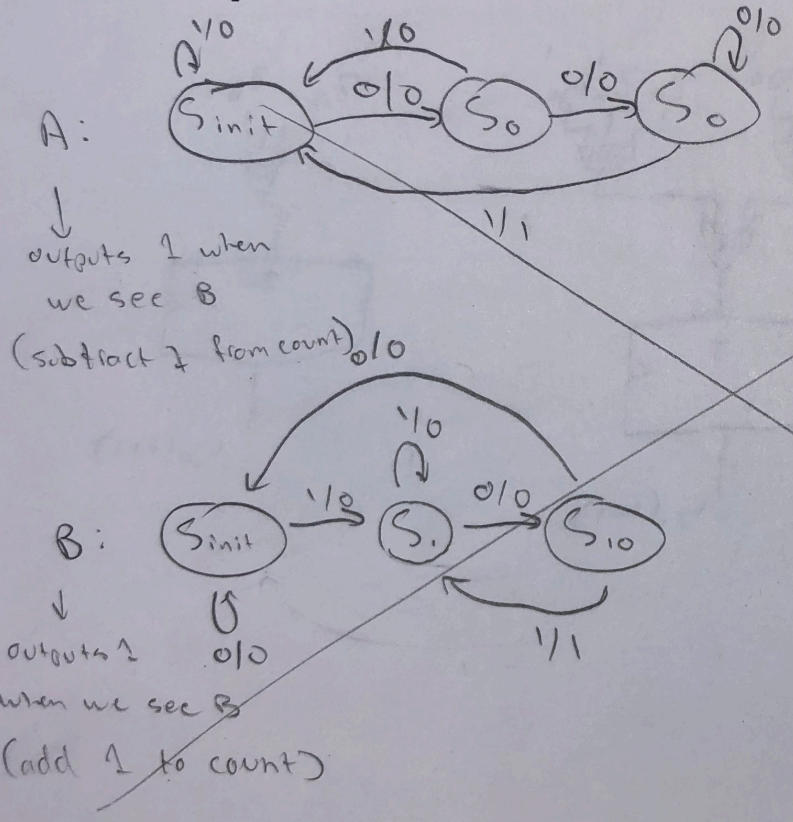
Problem 3 (20 points) ~~14~~ 20 any logical unit BUT adders

Given two 1-bit input streams A and B, output 1 if the difference between the number of times the pattern "001" appears in stream A and "101" appears in stream B is 3. If the difference between the number of their appearances is not 3, then the output is ~~1~~ 0. You may use any type of flip flops or logical units of your choosing.

For example:

A: 001000000
B: 101010101

Would output: 000000001. Notice that the B pattern overlaps.



PS	0	1
00	01/0	00/0
01	10/0	00/0
10	10/0	00/1
11	–	–

$z_A = xy_1$

$y_0^{(t+1)} = x'y_1y_0'$

$y_1^{(t+1)} = x'(y_1 \otimes y_0)$

PS	0	1
00	00/0	01/0
01	10/0	01/0
10	00/0	01/1
11	–	–

$z_B = xy_1$

$y_1^{(t+1)} = x'y_1y_0$

$y_2^{(t+1)} = x$

*WORK ON BACK OF PAGE

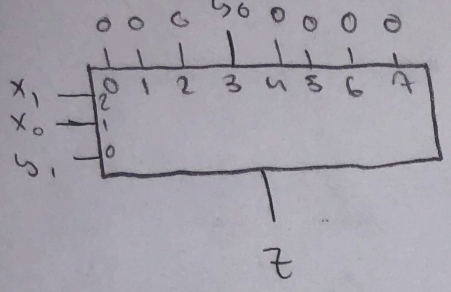
Problem 4 Extra Page

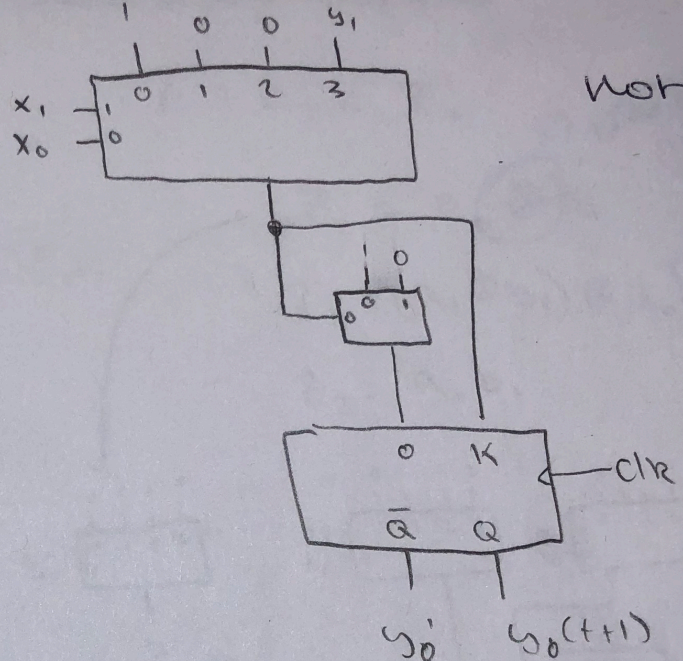
$$D(x) = OK(x, x')$$

$$z = x_1' x_0 y_1 y_0$$

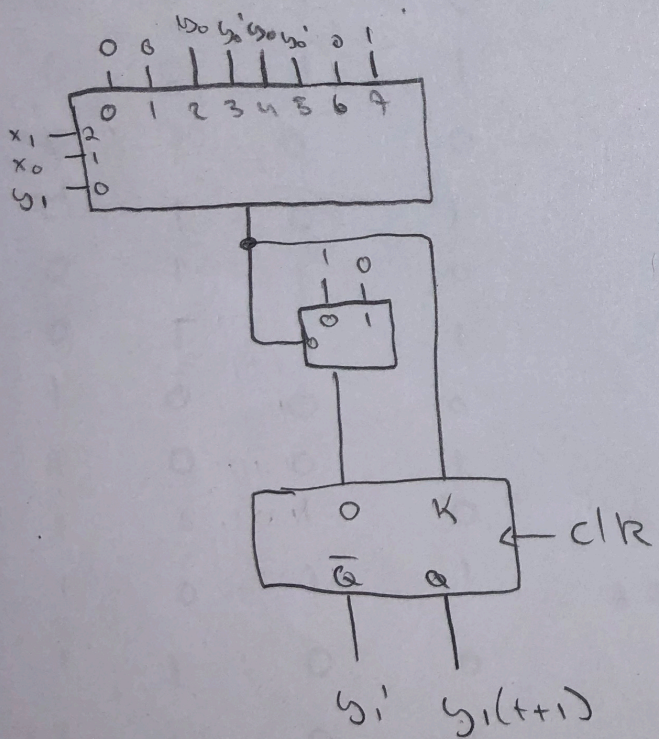
$$Y_1^{(t+1)} = x_1' x_0 y_1' y_0 + x_1 x_0' y_1' y_0 + x_0 y_1 y_0' + x_1 y_1 y_0' + x_1 x_0 y_1$$

$$Y_0^{(t+1)} = x_1' x_0' + x_1 x_0 y_1$$





Not optimized



Problem 5 (20 points)

Given 6 2-bit numbers as input, {A, B, C, D, E, F}, design a system such that the system finds the maximum sum between any of the 2 inputs. You may only use multiplexers to implement this system.

For example, if all the inputs are 01, then the maximum sum output should be 010.

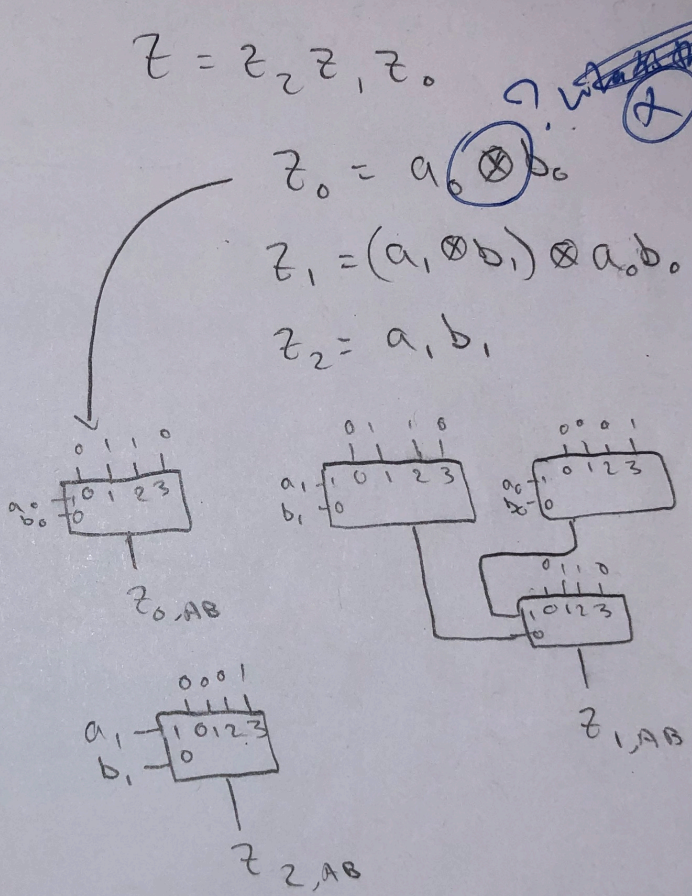
Sum of two numbers: $z = z_2 z_1 z_0$

$z_0 = a_0 \otimes b_0$

$z_1 = (a_1 \otimes b_1) \otimes a_0 b_0$

$z_2 = a_1 b_1$

a_1	a_0	b_1	b_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1



+ Implement the above for all possible combos:

AB, AC, AD, AE, AF
BC, BD, BE, BF
CD, CE, CF
DE, DF
EF

Run each z_2, z_1, z_0 into a priority decoder (with higher # = higher priority), take the ors of all resulting A's, b's, ..., 2's, 1's, 0's, run those into their respective places in a priority decoder, and the output of that is the maximum sum output.