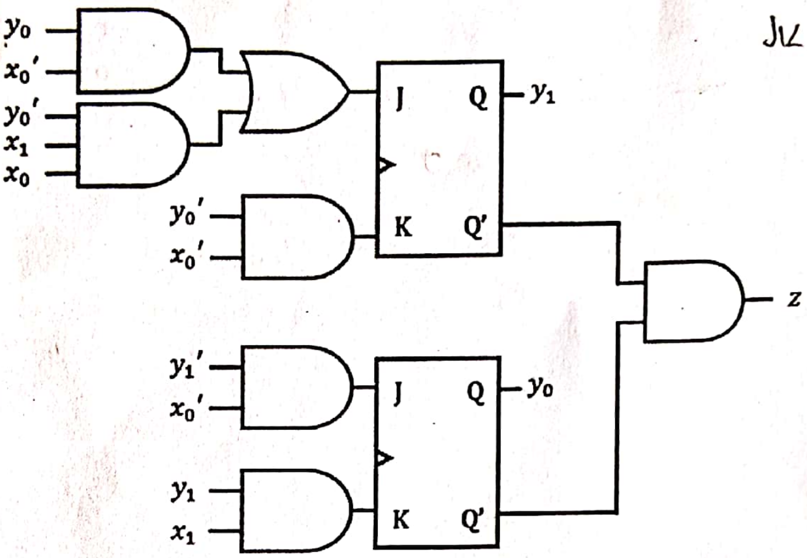
Problem 1 (20 points)

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.



JK

PS \ NS	0	1
0	0-	1-
1	-1	-0

PS $y_1 y_0$	Input $x_1 x_0$: 00	01	10	11	00	01	10	11
00	01	00	01	10	0	1	0	0
01	11	01	11	01	0	0	0	0
10	00	10	00	10	1	0	1	0
11	11	10	11	10	0	0	0	0

NS   output

$y_1 = 0 \Rightarrow J = y_0 x_0' + y_0' x_1 x_0$

$y_1 = 1 \Rightarrow K = y_0' x_0'$

$y_0 = 0 \Rightarrow J = y_1' x_0'$

$y_0 = 1 \Rightarrow K = y_1 x_1$

$S_0 = y_1 y_0 = 00$
$S_1 = 01$
$S_2 = 10$
$S_3 = 11$

PS	input x=00	01	10	11
S_0	S_1,0	S_0,1	S_1,0	S_2,0
S_1	S_3,0	S_1,0	S_3,0	S_1,0
S_2	S_0,1	S_2,0	S_0,1	S_2,0
S_3	S_3,0	S_2,0	S_3,0	S_2,0

NS, z

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.

	input x=0	x=1
A	A,0	A,0
B	A,0	A,0
C	A,0	A,0
D	A,0	A,0
E	A,0	A,0
F	A,0	A,0
G	A,0	G,1
H	G,1	H,1

NS, z

14 +1

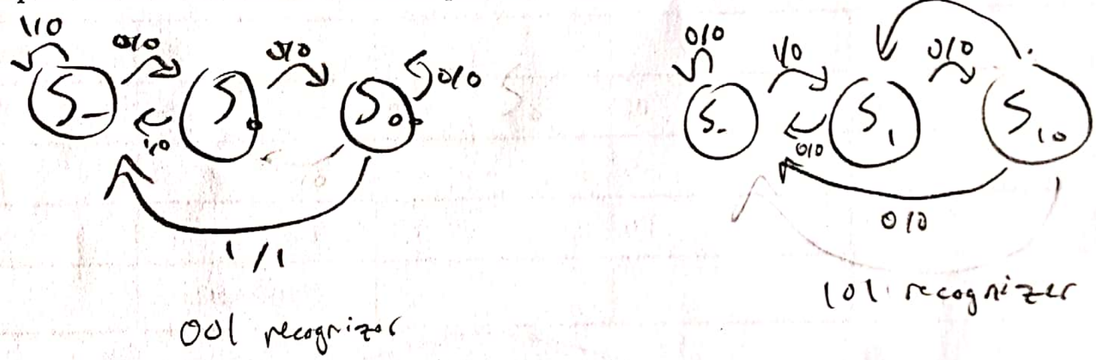
Problem 3 (20 points)

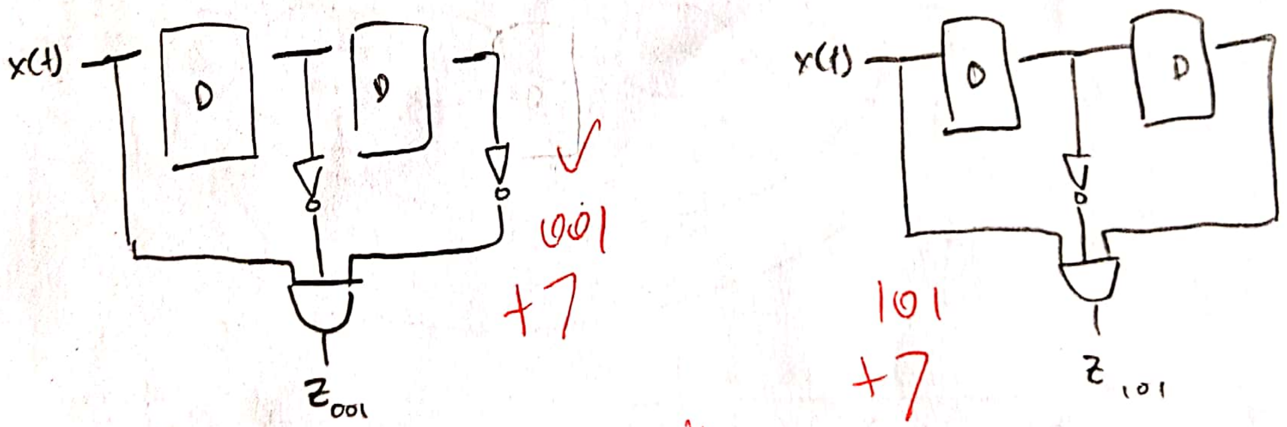
Given two 1-bit input streams A and B, output 1 if the difference between the number of times the pattern “001” appears in stream A and “101” appears in stream B is 3. If the difference between the number of their appearances is not 3, then the output is 1. You may use any type of flip flops or logical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

For example:

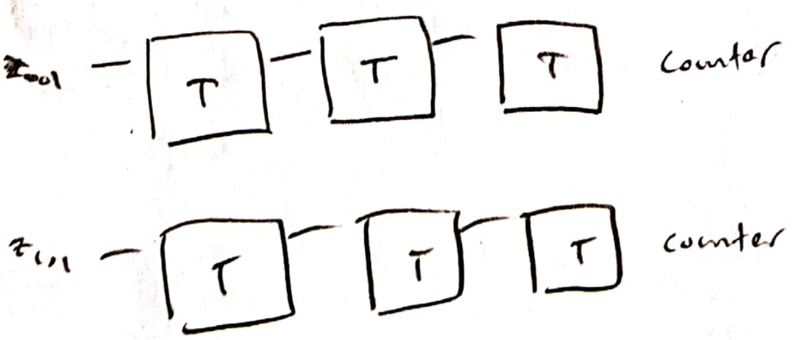
A: 001000000
B: 101010101

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





not quite



10 state transition table + diag
+7
+1
18

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}d}

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

0 otherwise

ababa

Notes:

Overlaps can occur. For example adada would output 00101

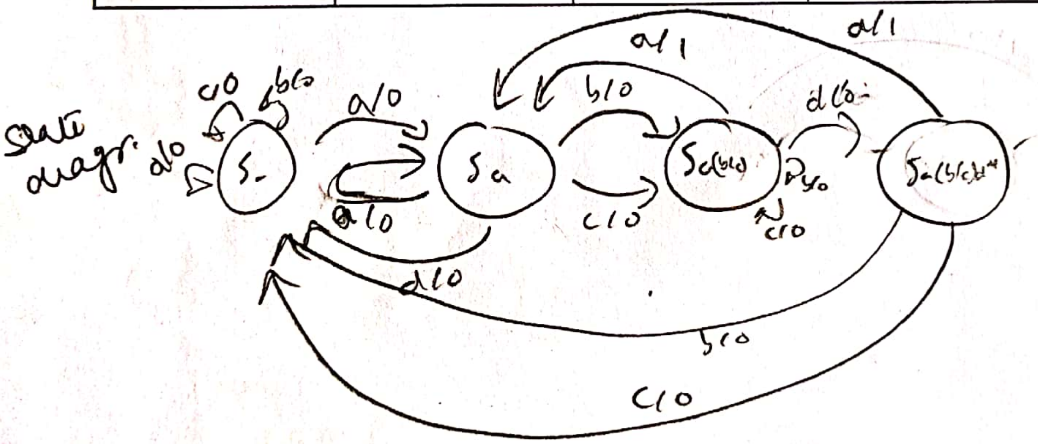
adb

| means OR

* means 0 or more of the previous character

+ means 1 or more of the previous character

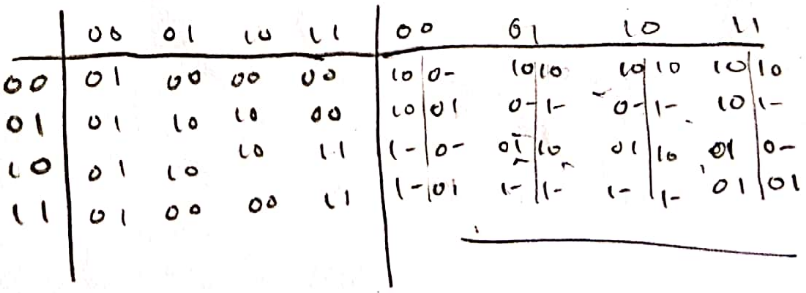
Prev State Q(t)	OK			
	00	01	10	11
0	1	1	0	-
1	-	1	0	0
	Nxt State Q(t+1)			

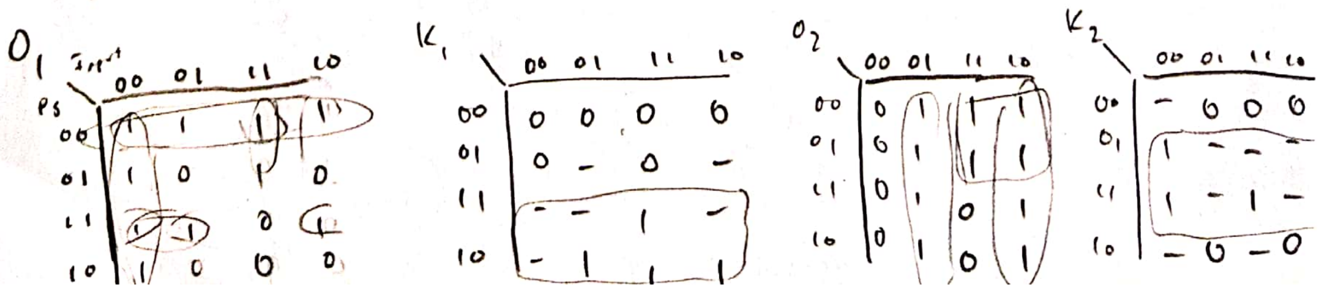


PS	NS 0	NS 1
0	10	0-
1	1-	01

PS	a	b	c	d
S_-	S_1,0	S_-,0	S_-,0	S_-,0
S_1	S_1,0	S_2,0	S_2,0	S_-,0
S_2	S_1,1	S_2,0	S_2,0	S_3,0
S_3	S_1,1	S_-,0	S_-,0	S_3,0

=)





Problem 4 Extra Page

$O_1 = (x_1' x_0') + (y_1' y_0) + (y_1 x_0 y_1') + (y_1 y_0 x_1') + (y_1 y_0 x_2')$

$K_1 = y_1$

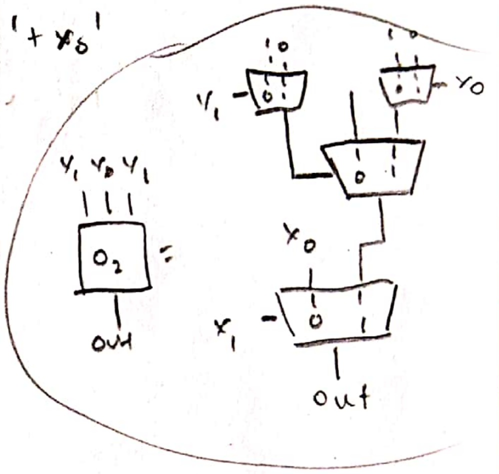
$O_2 = x_1' x_0 + x_1 y_1' + x_1 x_0'$

$O_{2x_1} = y_1' + x_0'$

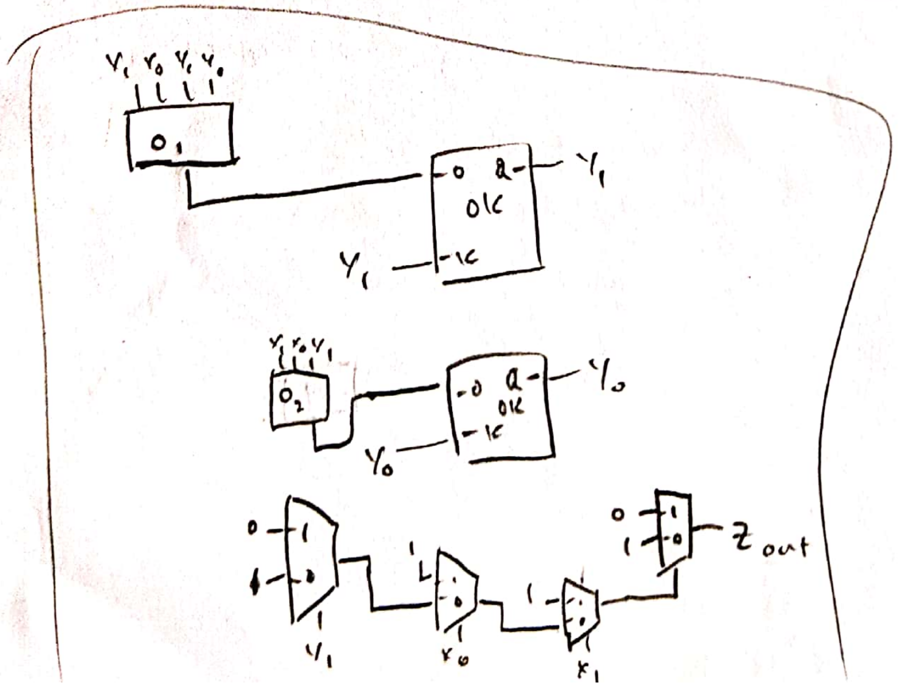
$O_{2x_1'} = x_0$

$K_2 = y_0$

$Z = x_1' y_0' y_1 = (x_1 + x_0 + y_1')'$



O_1 not implemented (would use Shannon decomposition)

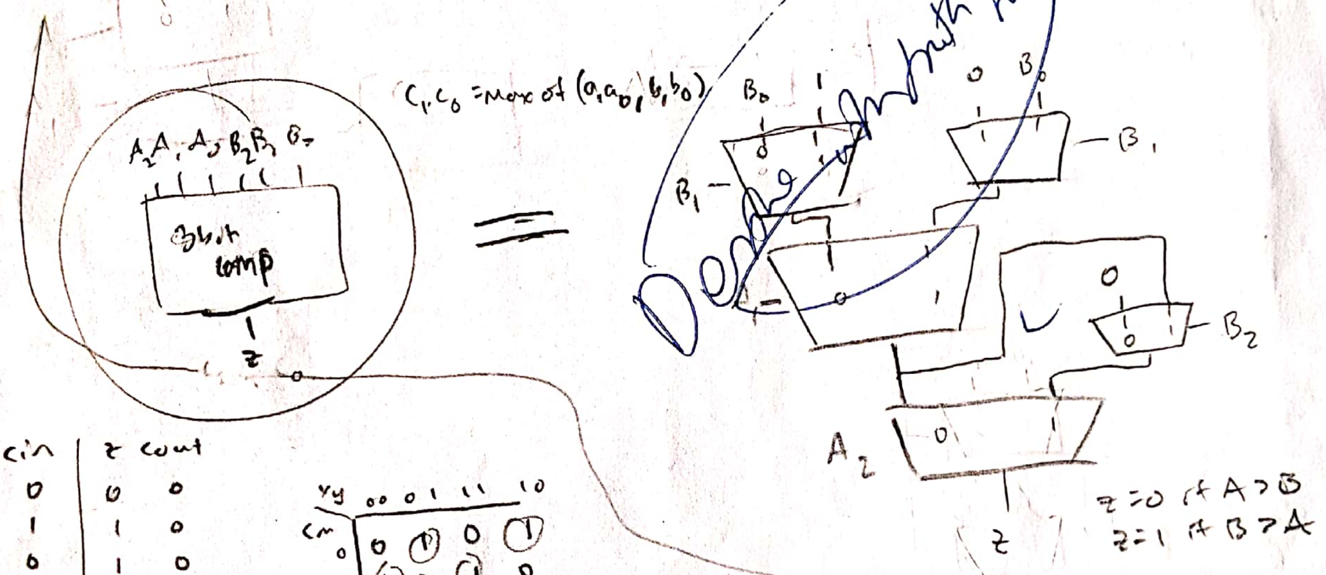


Problem 5 (20 points)

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.

Let a 3 bit mux and adder circuit be implemented as shown below



x	y	cin	z	cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

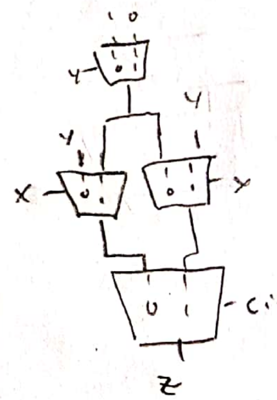
cin \ xy	00	01	11	10
0	0	1	0	1
1	1	0	1	0

cin \ xy	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$z = cin\,x'y' + cin'x'y + xy\,cin + xy'cin'$

$z_{cin} = x'y' + xy$

$z_{cin'} = x'y + xy'$

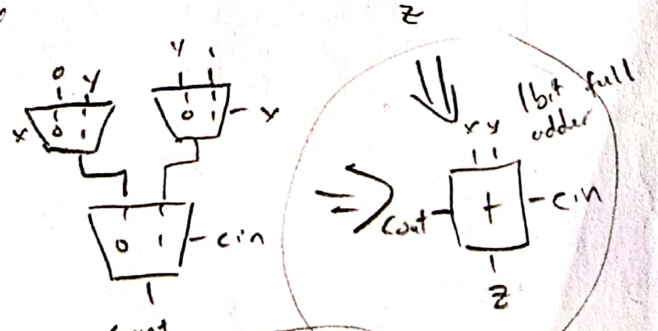


$C_{out} = xy + cin\,y + cin\,x$

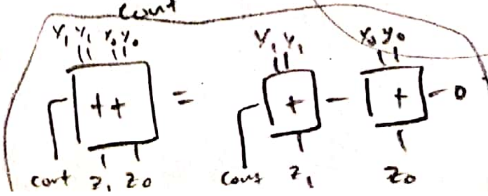
$C_{out} = cin\,xy + cin'xy + cin\,y + cin\,x$

$C_{out\,cin} = xy + x + y = x + y$

$C_{out\,cin'} = xy \qquad = xy$



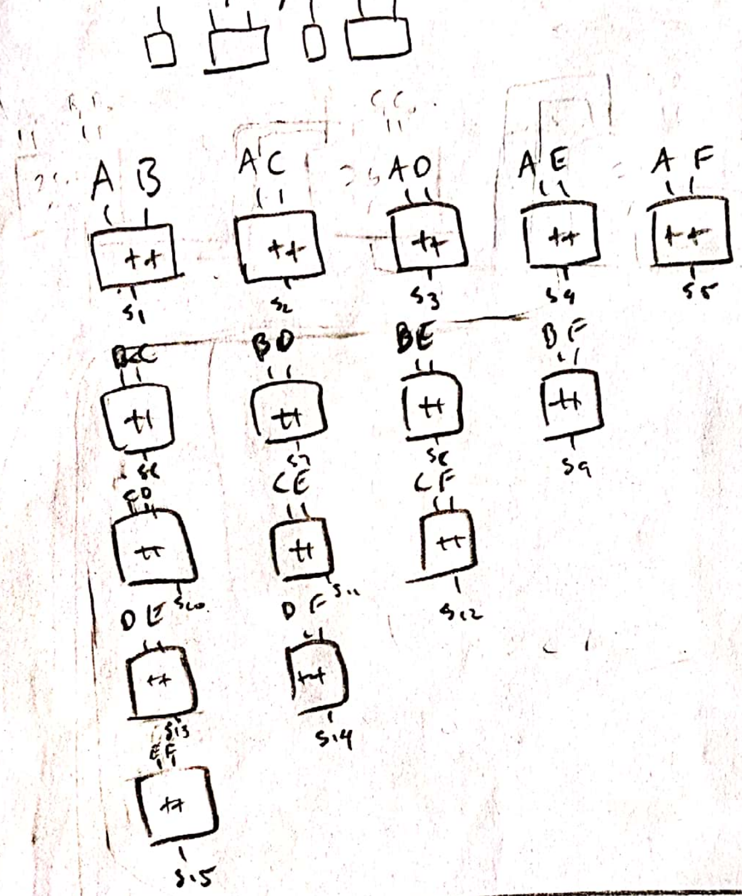
2 bit adder



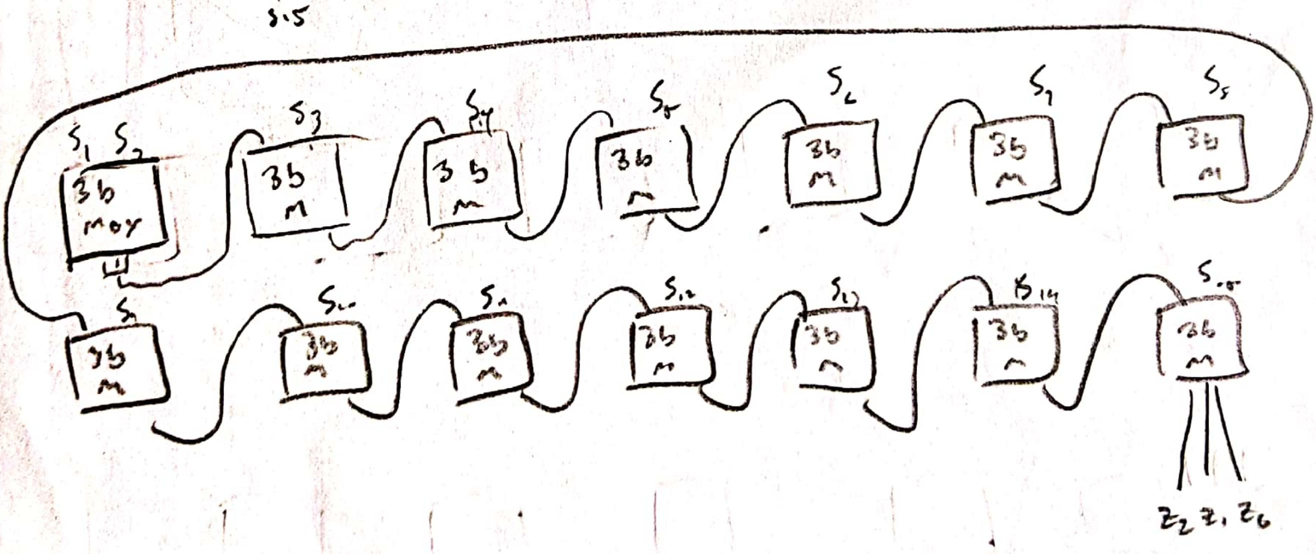
Problem 5 Extra Page

Solution

Let A = A_1A_0, B = B_1B_0 ... etc



Messy



3 bit Mux defined on back of last page

3 bit comp defined on first page of 5

and 2 bit adder defined on first page of 5