UID:

University of California Los Angeles Computer Science Department

CSM51A Midterm 2

Winter Quarter 2019

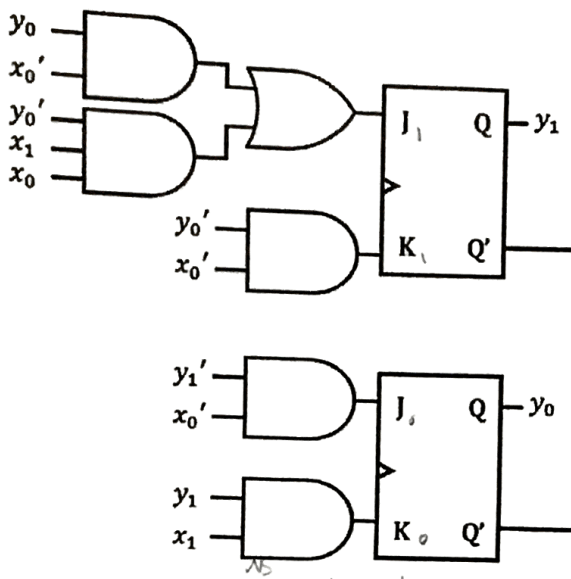
This is a closed book exam. Absolutely nothing is permitted except pen, pencil and eraser to write your solutions. Any academic dishonesty will be prosecuted to the full extent permissible by university regulations.

Time Allowed: 100 Minutes

Problem(Possible Points)	Points
1 (20)	18
2 (20)	20
3 (20)	17
4 (20)	18
5 (20)	7
Total (100)	80

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.



$J_1 = x_0'y_0 + y_0'x_1x_0$

$K_1 = x_0'y_0'$

$z = y_1'y_0'$

$J_0 = x_0'y_1'$

$K_0 = y_1x_1$

PS

x_1	x_0	y_1	y_0	J_1	K_1	J_0	K_0	Y_1	Y_0	z
0	0	0	0	0	1	0	1	0	0	0
0	0	0	1	0	1	1	0	0	1	0
0	0	1	0	0	1	0	1	0	0	0
0	0	1	1	0	1	0	1	0	0	0
0	1	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0
0	1	1	0	0	1	0	1	0	0	0
0	1	1	1	0	1	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	1	0
1	0	1	0	0	1	0	1	0	0	0
1	0	1	1	0	1	0	1	0	0	0
1	1	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	1	0
1	1	1	0	0	1	0	1	0	0	0
1	1	1	1	0	1	0	1	0	0	0

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

input = x_1x_0

	00	01	10	11
S_0	S_1, 0	S_0, 1	S_2, 0	S_2, 0
S_1	S_2, 0	S_1, 0	S_3, 0	S_1, 0
S_2	S_0, 0	S_2, 0	S_0, 1	S_2, 0
S_3	S_3, 0	S_3, 0	S_2, 0	S_3, 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.

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

	A	B C	D E	F	G	H
0	2	2 2	3 3	3	1	1
1	2	2 2	1 1	1	2	2

1 2 3

✓ →

$S_1 = \{A, B, C\}$, $S_2 = \{D E F\}$, $S_3 = \{G, H\}$

	x=0	x=1
S_1	S_2, 0	S_2, 0
S_2	S_3, 0	S_1, 1
S_3	S_1, 1	S_2, 1

Problem 3 (20 points) 17

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 0. You may use any type of flip flops or logical units of your choosing.

For example:

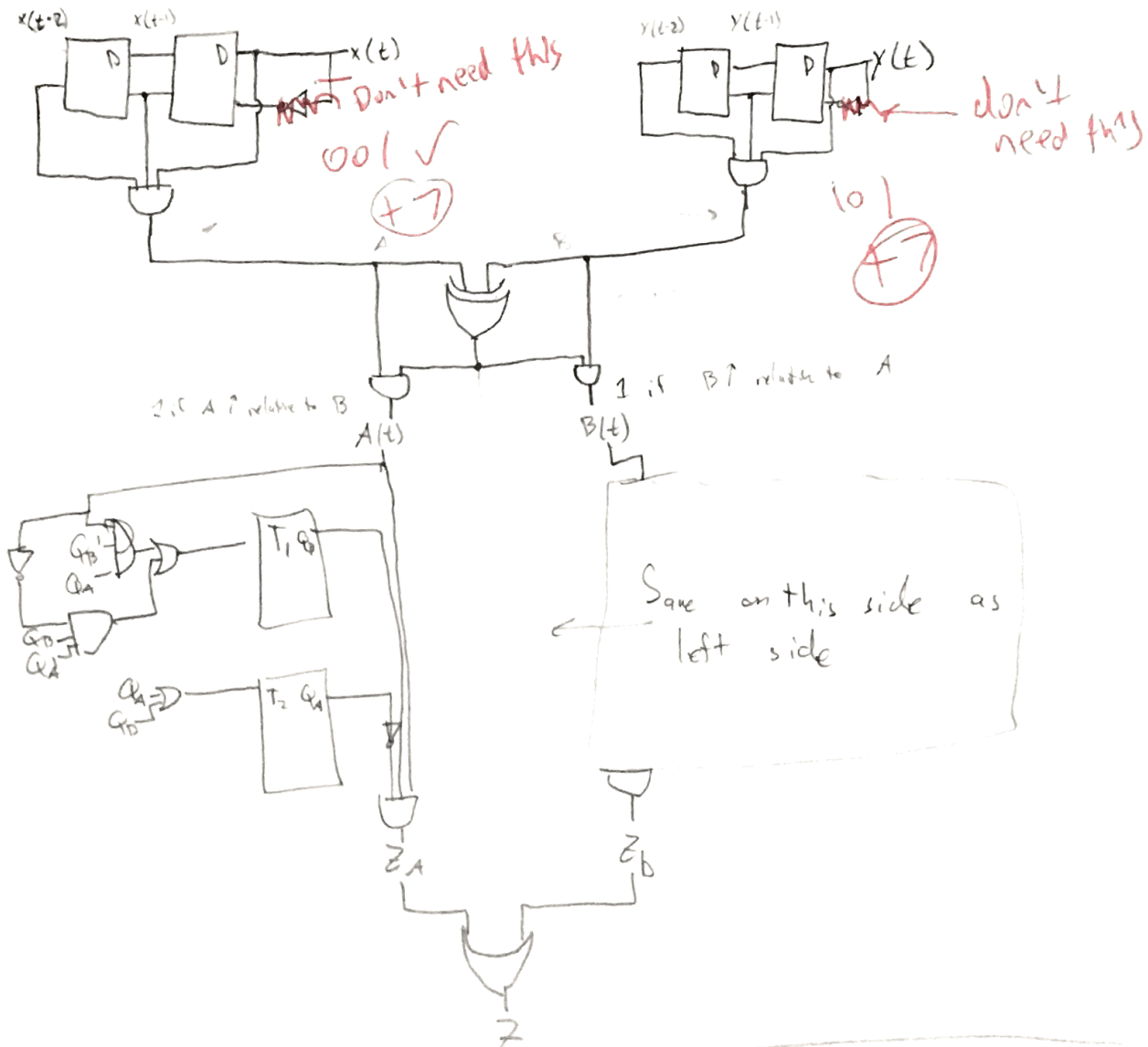
A stream = x(t)
B stream = y(t)

A: 001000000
B: 101010101

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

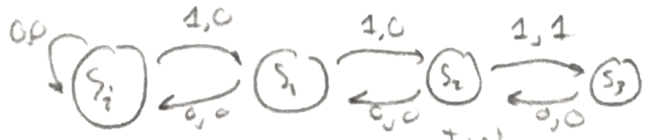
output 1 when 001 is found

output 1 when 101 is found



State machine on next page, pass both A(t) & B(t) into their own machines & OR results

Problem 3 Extra Page



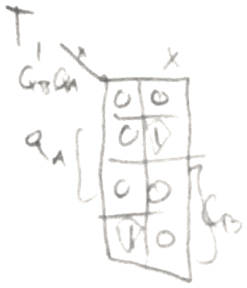
increase "count" to 3 & return 1

	x=0	x=1	enc	x=0	x=1		
S_0	S_0,0	S_1,0	00	00	00	00	00
S_1	S_1,0	S_2,0	01	00	10	01	11
S_2	S_2,0	S_3,1	10	01	11	11	01
S_3	S_3,0	—	11	10	-	01	01
			$Q_B Q_A$			T_1 T_2	T_1 T_2

Input

101

$Z = Q_B Q_A' x$



$T_1 = xQ_B'Q_A + x'Q_BQ_A'$

Needs 7 states, but nice try.

+3

$T_2 = Q_A + Q_B$

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

Notes:

Overlaps can occur. For example ~~adada~~ ababa would output 00101

| means OR

* means 0 or more of the previous character

\+ means 1 or more of the previous character

State diag – 7

table – 3

flip flop. – 6

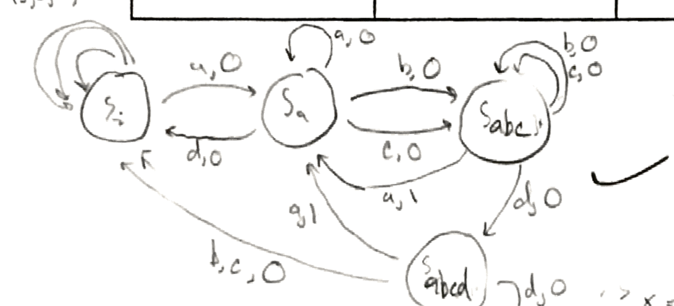
reduction 2

18

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

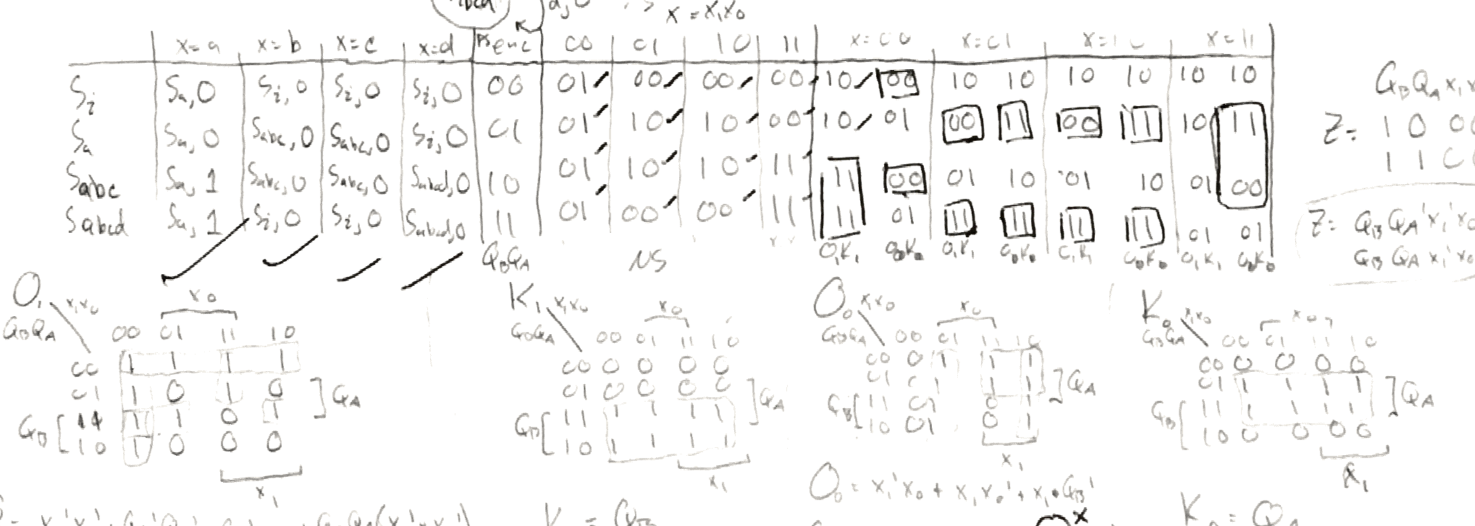
-1 for not dont cares.

-1 for not reducing expression



define input a = 00, b = 01, c = 10, d = 11

Part 1.



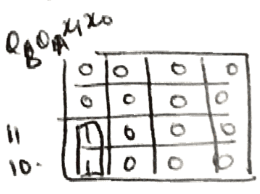
$Q = x_0'x_1' + Q_B'Q_A' + Q_B'x_1x_0 + Q_BQ_A(x_1'+x_0')$

$K_1 = Q_B$

$O_0 = x_1'x_0 + x_1x_0' + x_1 \cdot Q_B'$

$O_0 = x_1'x_0 + x_1x_0' + x_1(x_0' + Q_B')$

$K_0 = Q_A$



$Q_Bx_1x_0'$

Problem 4 Extra Page

$$D_1 = x_0'x_1' + Q_D'Q_A' + Q_B'x_1x_0 + Q_DQ_A(x_1' + x_0')$$

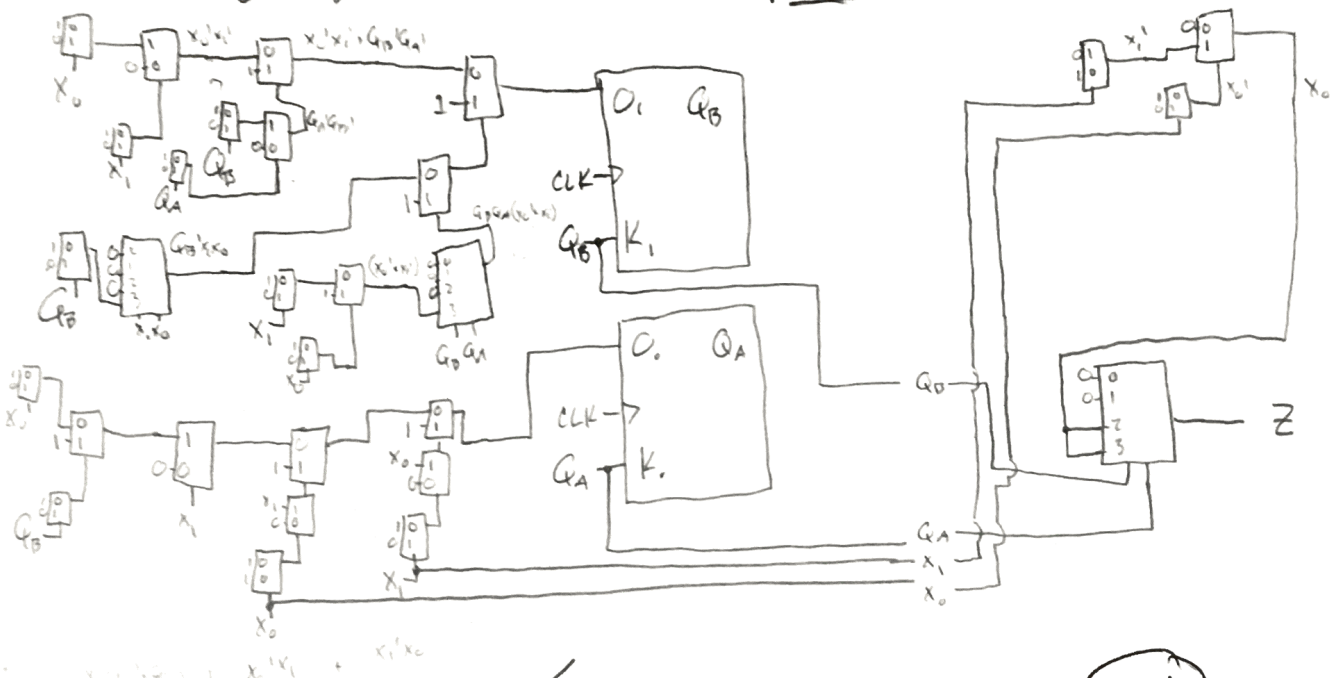
$$K_1 = Q_B$$

why do this?

$$D_0 = x_1'x_0 + x_1x_0' + x_1(x_0' + Q_D')$$

$$K_0 = Q_A$$

$$Z = Q_DQ_A'x_1'x_0' + Q_BQ_Ax_1'x_0' \quad = Q_Bx_1'x_0' \text{ can be reduced}$$



−1
for not reducing

−1
for not using don't cares

Problem 5 (20 points)

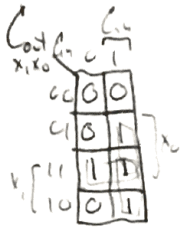
15 combos

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.

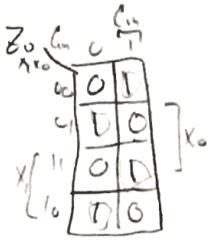
For example, if all the inputs are 01, then the maximum sum output should be 010.

1-bit adder

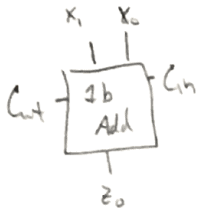
X_1	X_0	C_{in}	C_{out}	Z_0
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

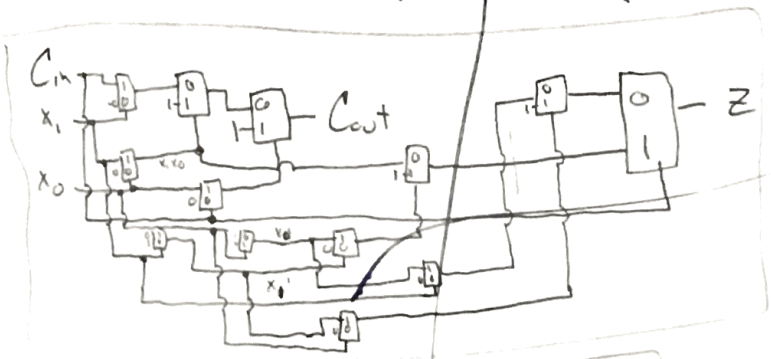


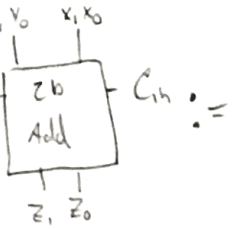
$C_{out} = C_{in}x_1 + x_1x_0 + x_0C_{in}$

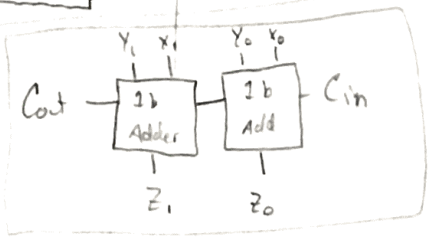


$Z_0 = C_{in}(x_1'x_0' + x_1x_0) + C_{in}'(x_1'x_0 + x_1x_0')$

define 

:= 

define 

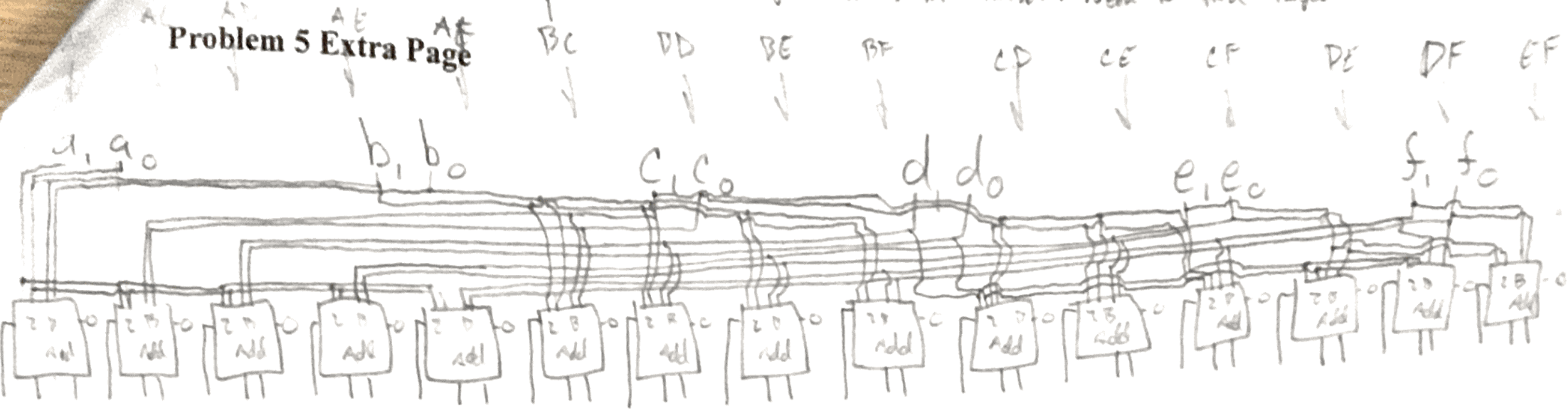
:= 

Inputs: a_1a_0, b_1b_0, c_1c_0, d_1d_0, e_1e_0, f_1f_0

Max out = 6 → 110 → 3 bits

1 adder per combo: gives 15 3-bit numbers. Need to find largest of them

Problem 5 Extra Page



← .. out of time