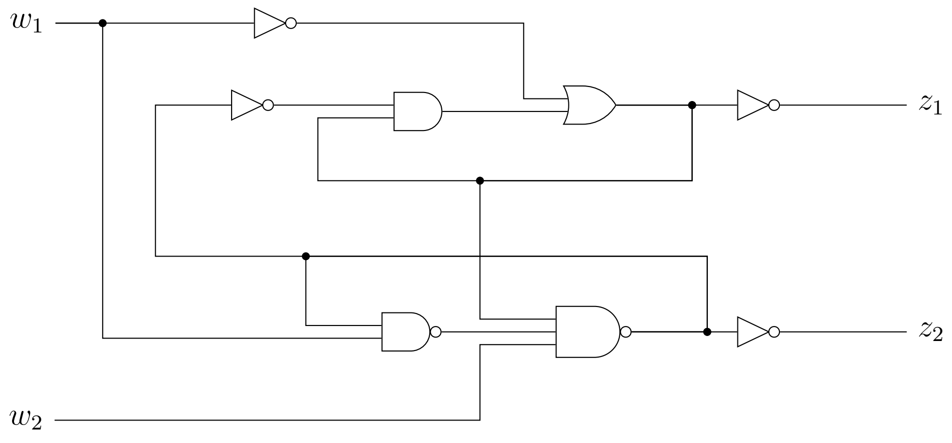
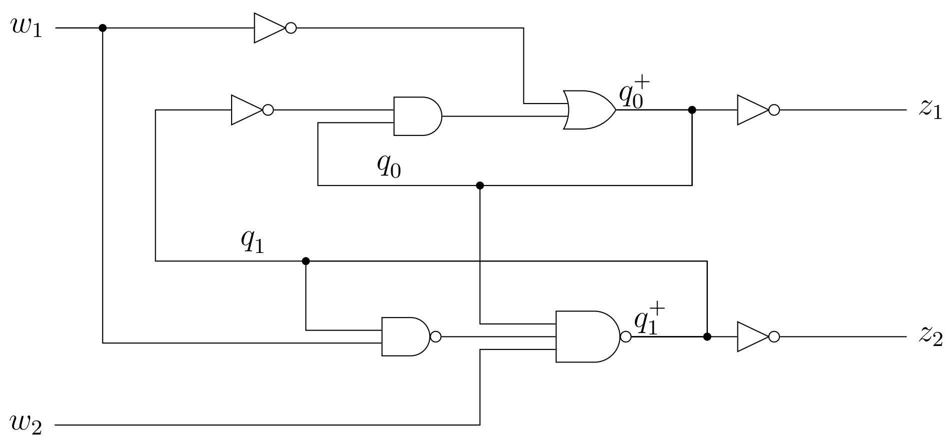
ECE 124 digital circuits and systems
Assignment #10

Q1: Consider the asynchronous circuit shown below. Derive the transition table and flow table for the circuit. Remember to circle stable states.



Solution:
Label a few things.



Here q_0 and q_1 are intended to represent the current state (secondary) variables. The values q_0^+ and q_1^+ are intended to represent the next state (excitation) variables.

The circuit outputs written in terms of the current state variables are:

$$\begin{aligned} z_1 &= \overline{q}_0 \\ z_2 &= \overline{q}_1 \end{aligned}$$

The next state equations are:

$$\begin{aligned} q_0^+ &= q_0\overline{q}_1 + \overline{w}_1 \\ q_1^+ &= \overline{q_0 w_2 \overline{q_1 w_1}} \\ &= \overline{q}_0 + \overline{w}_2 + q_1 w_1 \end{aligned}$$

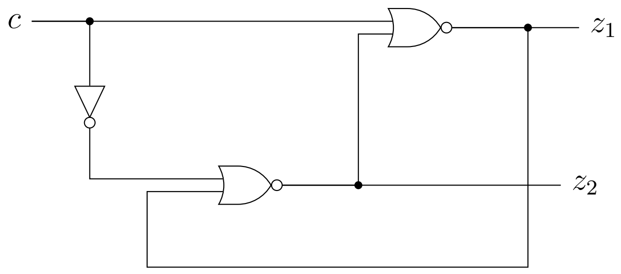
The transition table is (the outputs are only a function of the current state):

Current state	Next state ($q_1^+q_0^+$)				Outputs (z_1z_2)
q_1q_0	$w_1w_2 = 00$	$w_1w_2 = 01$	$w_1w_2 = 11$	$w_1w_2 = 10$	
00	11	11	10	10	11
01	11	$\boxed{01}$	$\boxed{01}$	11	10
10	11	11	$\boxed{10}$	$\boxed{10}$	01
11	$\boxed{11}$	01	10	10	00

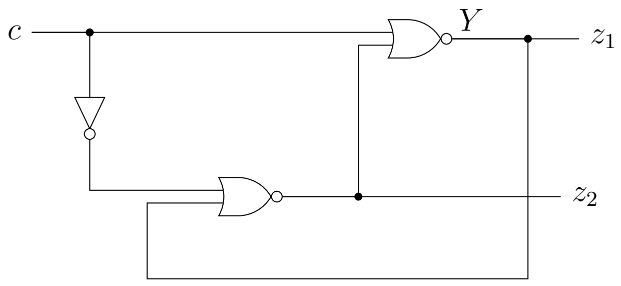
The flow table is:

Current state	Next state ($q_1^+q_0^+$)				Outputs (z_1z_2)
q_1q_0	$w_1w_2 = 00$	$w_1w_2 = 01$	$w_1w_2 = 11$	$w_1w_2 = 10$	
A	D	D	C	C	11
B	D	$\boxed{B}$	$\boxed{B}$	D	10
C	D	D	$\boxed{C}$	$\boxed{C}$	01
D	$\boxed{D}$	B	C	C	00

Q2: Consider the asychronous circuit shown below. Derive a transition table and flow table for the circuit. Draw waveforms for signals c, z_1 and z_2 assuming that c is a square wave clock signal and that each gate has a propagation delay of Δ units of time.



Solution:
The circuit really only has a single loop so the state only consists of a single bit:



We need to write the next state (excitation) variables in terms of the current state (secondary variables). This gives

$$Y = \overline{c + z_2} = \overline{c + \overline{\overline{c} + y}} = \overline{c}(\overline{c} + y) = \overline{c} + \overline{c}y = \overline{c}$$

We need to write the output equations in terms of the current state (secondary variables). This gives

$$\begin{aligned} z_1 &= y \\ z_2 &= \overline{\overline{c} + y} = c\overline{y} \end{aligned}$$

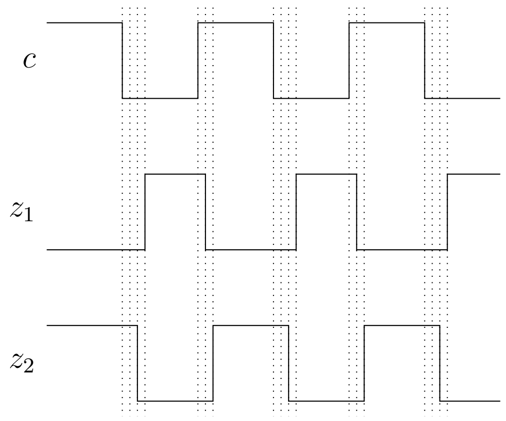
The transition table:

Current state	Next state (Y)		Outputs ($z_1 z_2$)	
y	$c = 0$	$c = 1$	$c = 0$	$c = 1$
0	1	$\boxed{0}$	00	01
1	$\boxed{1}$	0	10	10

The flow table:

Current state	Next state (Y)		Outputs ($z_1 z_2$)	
y	$c = 0$	$c = 1$	$c = 0$	$c = 1$
A	B	$\boxed{A}$	00	01
B	$\boxed{B}$	A	10	10

Drawing the waveforms...

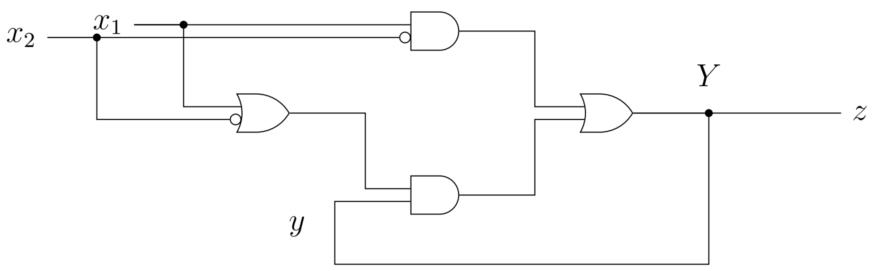


The distance between pairs of dotted vertical lines is Δ.

Q3: An asynchronous sequential circuit with two inputs x_1 and x_2 and one output z is described by the following equations and output functions: $Y = x_1 x_2' + (x_1 + x_2')y$ and $z = y$. Here, Y's represent the excitation variables (next state variables) and y's represent the secondary variables (current state variables).

Draw the logic diagram of the circuit. Derive the transition table and flow table for the circuit.

Solution:
The circuit...



The transition table:

Current state	Next state (Y)				Outputs (z)
y	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$	
0	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	1	0
1	$\boxed{1}$	0	$\boxed{1}$	$\boxed{1}$	1

The flow table:

Current state	Next state (Y)				Outputs (z)
y	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$	
A	$\boxed{A}$	$\boxed{A}$	$\boxed{A}$	B	0
B	$\boxed{B}$	A	$\boxed{B}$	$\boxed{B}$	1

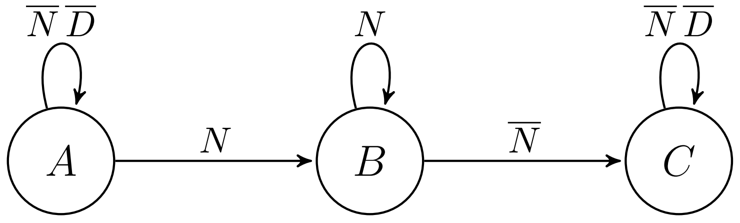
Q4: A control mechanism for a vending machine accepts nickels and dimes. It dispenses merchandise when 20 cents is deposited; it does not give change if more than 20 cents is deposited. Design a state diagram assuming that the control mechanism will be implemented via an asynchronous sequential circuit (the state diagram might have a large number of states). When deriving your state diagram, assume that the circuit operates in fundamental mode (only one input changes at a time).

Solution:
It is best to assume fundamental mode. When doing an synchronous state diagram, assume you start in a stable state, you change a single input and you go to another stable state.

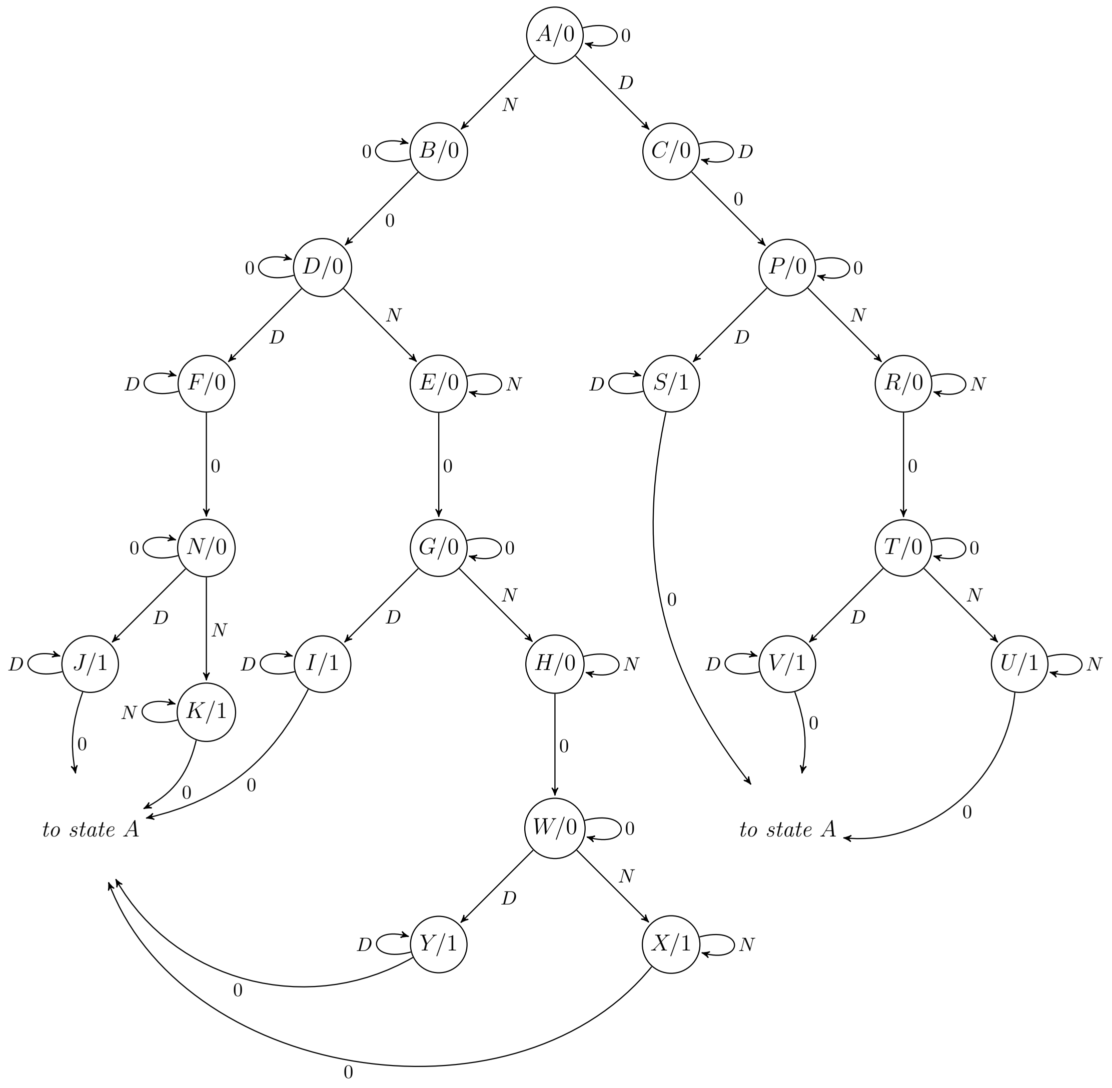
This circuit has two inputs N and D to indicate whether a nickel or dime is deposited, respectively. The input patterns that can happen are:

1. $DN = 00$. This means no coins are deposited $\rightarrow$ we will represent this in the diagram by a 0.
2. $DN = 10$. This means a dime is deposited $\rightarrow$ we will represent this in the diagram by a D.
3. $DN = 01$. This means a nickel is deposited $\rightarrow$ we will represent this in the diagram by a N.
4. $DN = 11$. This means a nickel and a dime are deposited $\rightarrow$ we will this is impossible since only one coin deposited at a time.

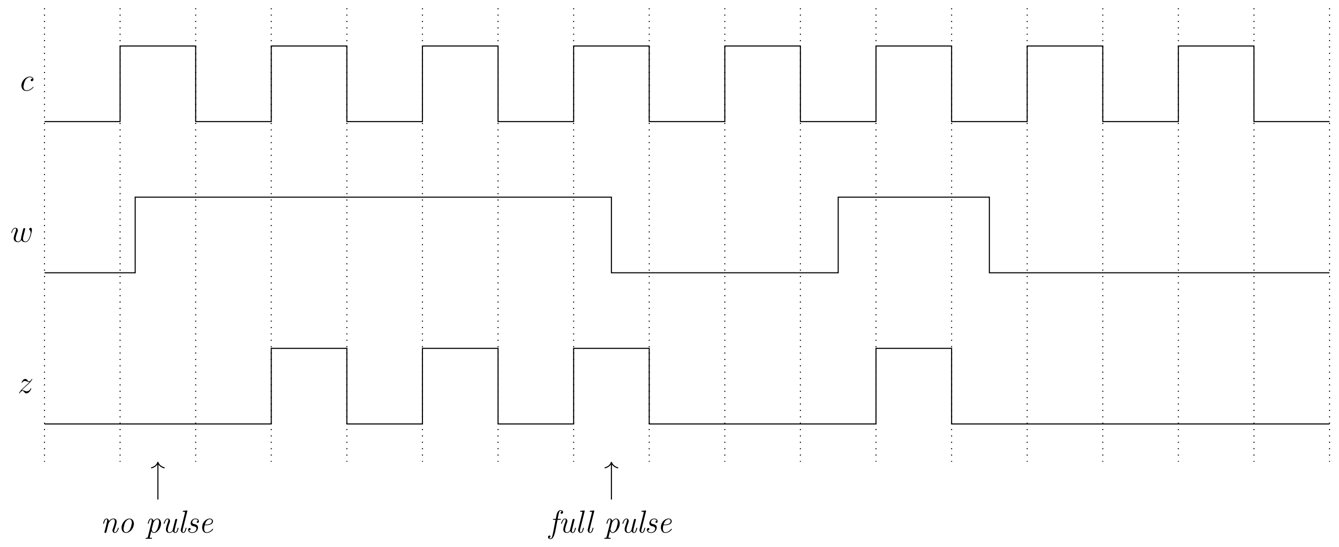
Firstly, in the following state diagram we will see things like the following ... Assume we are in stable state A and $N = 0$ and $D = 0$. The diagram shows that we now have $N = 1$ which means a single input has changed. We therefore move to a new stable state B. Notice in B we have a self loop as long as $N = 1$ and B is a stable state. Finally, notice that as soon as $N = 0$ (a single input has changed) we immediately move to another stable state C in which $N = 0$ and $D = 0$. This pattern is repeated a ton of times since we need to detect a coin **once and only once** and then move on. The interpretation of the pattern is waiting for a coin (state A), got a coin and waiting for it to pass through the coin dispenser and don't want to count it twice (state B), and the coin has passed and waiting for another coin (state C).


$$\begin{tikzcd} A \arrow[loop above, "\overline{N}\,\overline{D}"] \arrow[r, "N"] & B \arrow[loop above, "N"] \arrow[r, "\overline{N}"] & C \arrow[loop above, "\overline{N}\,\overline{D}"] \end{tikzcd}$$

Here is the entire diagram.

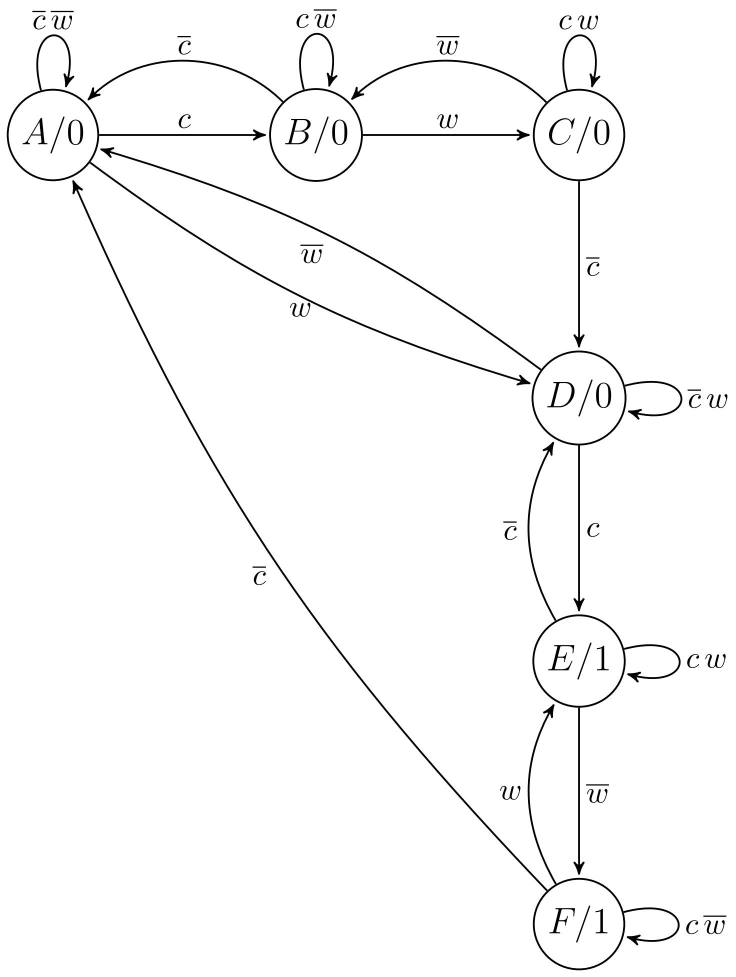

A/0
B/0
C/0
D/0
P/0
F/0
E/0
S/1
R/0
N/0
G/0
T/0
J/1
K/1
I/1
H/0
V/1
U/1
W/0
Y/1
X/1
to state A
to state A
0
N
D

Q5: Assume that you are to design an asynchronous sequential circuit with the following specifications. The circuit has two inputs c and w. The input c is a series of pulses. The output z replicates the input c when $w = 1$, otherwise $z = 0$. The pulses on z must be full pulses. That is, if $c = 1$ when w changes from $0 \to 1$, then the circuit will not produce a partial pulse on z but will wait until the next pulse to generate $z = 1$. If $c = 1$ when w changes from $1 \to 0$, then a full pulse must be generated. The desired operation is illustrated below.



Derive a primitive flow table for this problem assuming fundamental mode operation. Perform state minimization to obtain a reduced flow table. Finally, perform state assignment and derive a circuit.

Solution:
Assume some initial state (following the timing diagram, we could assume $c = 0$ and $w = 0$ which produce and output of $z = 0$). Then, consider changing a single input and go to a new stable state. Before adding another new stable state, ask whether or not such a state already exists. If yes, then go there, otherwise create another new stable state. Make sure all transitions are accounted for. This will lead to a primitive flow table (I will draw a state diagram first). Note that with 2 inputs, assuming fundamental mode there should be two edges leaving each state (these edges might loop back to the same state) — this is because each input (and there's 2 of them) might change.



Note that in the state diagram, the output has been shown as a Moore output. However, the output is only meaningful in situations where the state is **a stable state**.

So the primitive flow table is (outputs shown only in stable states!):

Current State	**Next state**				**Output** (z)			
	$cw = 00$	$cw = 01$	$cw = 11$	$cw = 10$	$cw = 00$	$cw = 01$	$cw = 11$	$cw = 10$
A	$\boxed{A}$	D	–	B	0	–	–	–
B	A	–	C	$\boxed{B}$	–	–	–	0
C	–	D	$\boxed{C}$	B	–	–	0	–
D	A	$\boxed{D}$	E	–	–	0	–	–
E	–	D	$\boxed{E}$	F	–	–	1	–
F	A	–	E	$\boxed{F}$	–	–	–	1

The implication chart... Notice that with all the don't cares at the outputs, certain states are not prevented from being merged.

B	✓				
C	✓	✓			
D	✓	(C,E)X	(C,E)X		
E	(B,F)X	(C,E)X,(B,F)X	X	✓	
F	(B,F)X	X	(C,E)X,(B,F)X	✓	✓
	A	B	C	D	E

The solution is to merge A, B and C into one state and D, E, and F into another state. The reduced flow table is:

Current State	**Next state**				**Output** (z)			
	$cw = 00$	$cw = 01$	$cw = 11$	$cw = 10$	$cw = 00$	$cw = 01$	$cw = 11$	$cw = 10$
A	$\boxed{A}$	F	$\boxed{A}$	$\boxed{A}$	0	–	0	0
F	A	$\boxed{F}$	$\boxed{F}$	$\boxed{F}$	–	0	1	1

The transition table is:

Current State	**Next state** (Y)				**Output** (z)			
y	$cw = 00$	$cw = 01$	$cw = 11$	$cw = 10$	$cw = 00$	$cw = 01$	$cw = 11$	$cw = 10$
0	$\boxed{0}$	1	$\boxed{0}$	$\boxed{0}$	0	–	0	0
1	0	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	–	0	1	1

The equations required to draw the circuit are:

$$\begin{aligned} Y &= \bar{c}w + cy \\ z &= cy \end{aligned}$$

You can draw this circuit.

Q6: Obtain a state diagram and a primitive flow table for an asynchronous sequential circuit with two inputs x_1 and x_2, and two outputs z_1 and z_2. The circuit must have the following behaviour:

(a) When $x_1x_2 = 00$, the output is $z_1z_2 = 00$;

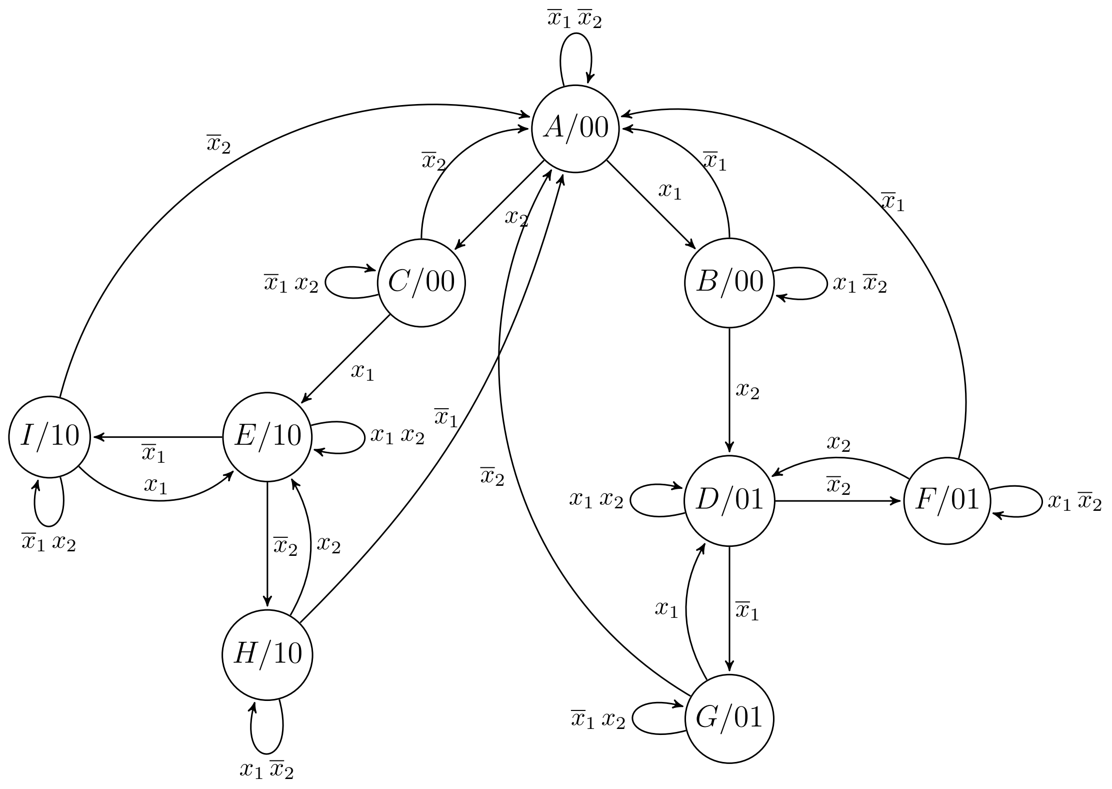
(b) When $x_1 = 1$ and x_2 changes from $0 \to 1$, the output is $z_1z_2 = 01$;

(c) When $x_2 = 1$ and x_1 changes from $0 \to 1$, the output is $z_1z_2 = 10$;

(d) Otherwise, then output does not change.

Solution:

Assume fundamental mode. Pick a start state in which inputs and outputs are know (and therefore stable). Change one input at a time generating new states as required. Since the circuit has 2 inputs, there should be two edges (ignoring self loops) leaving each state since each input can change. I'll start with the initial state $x_1x_2 = 00$ and the outputs are $z_1z_2 = 00$ which I know is stable.



The primitive flow table...

Current State	Next state $x_1x_2=00$	Next state $x_1x_2=01$	Next state $x_1x_2=11$	Next state $x_1x_2=10$	Output (z_1z_2) $x_1x_2=00$	Output $x_1x_2=01$	Output $x_1x_2=11$	Output $x_1x_2=10$
A	$\boxed{A}$	C	–	B	00	–	–	–
B	A	–	D	$\boxed{B}$	–	–	–	00
C	A	$\boxed{C}$	E	–	–	00	–	–
D	–	G	$\boxed{D}$	F	–	–	01	–
E	–	I	$\boxed{E}$	H	–	–	10	–
F	A	–	D	$\boxed{F}$	–	–	–	01
G	A	$\boxed{G}$	D	–	–	01	–	–
H	A	–	E	$\boxed{H}$	–	–	–	10
I	A	$\boxed{I}$	E	–	–	01	–	–

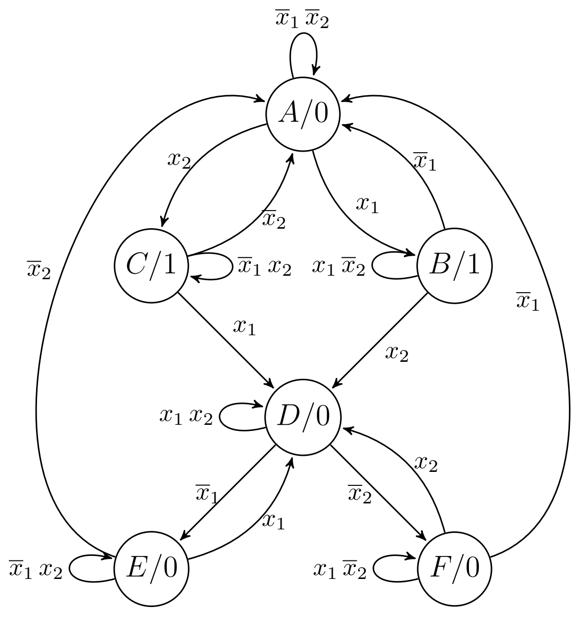
Q7: Consider an asynchronous sequential circuit with two inputs x_1 and x_2, and one output z. Initially, both inputs are equal to 0. When either x_1 or x_2 becomes 1, the output z becomes 1. When the other input becomes 1, the output z changes back to 0. The output z remains 0 until both inputs return to 0 and the process repeats itself.

Assume fundamental mode operation. Derive a state diagram and a primitive flow table. Perform state minimization to show that the state table can be reduced to the following flow table.

Current State	Next State $x_1x_2 = 00$	01	11	10	Output (z) $x_1x_2 = 00$	01	11	10
a	$\boxed{a}$	$\boxed{a}$	b	$\boxed{a}$	0	1	-	1
b	a	$\boxed{b}$	$\boxed{b}$	$\boxed{b}$	-	0	0	0

Solution:

The state diagram assuming fundamental mode operation (output is shown as a Moore output but it only assumes that value if the state is stable):



The primitive flow table:

Current State	**Next state** $x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$	**Output** (z) $x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$
A	$\boxed{A}$	C	–	B	0	–	–	–
B	A	–	D	$\boxed{B}$	–	–	–	1
C	A	$\boxed{C}$	D	–	–	1	–	–
D	–	E	$\boxed{D}$	F	–	–	0	–
E	A	$\boxed{E}$	D	–	–	0	–	–
F	A	–	D	$\boxed{F}$	–	–	–	0

The implication chart...

	A	B	C	D	E
B	✓				
C	✓	✓			
D	(C,E),(B,F)	(B,F)	(C,E)		
E	(C,E)	✓	X	✓	
F	(B,F)	X	✓	✓	✓

None of the conditions are true. The solution is to merge A, B and C into one state and D, E, and F into another state. The yields the desired reduced table.

Q8: Find a hazard-free minimum cost SOP implementations for the following functions:

(a) $f(a,b,c,d) = \sum(0,4,11,13,15) + D(2,3,5,10)$.

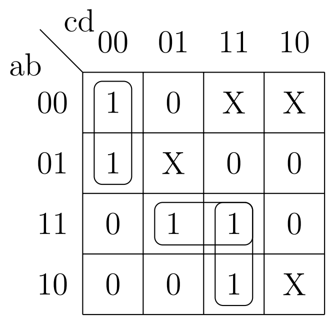
(b) $f(a,b,c,d,e) = \sum(0,4,5,24,25,29) + D(8,13,16,21)$.

Solution:
We need to write down Karnaugh maps to obtain both a minimum cost SOP as well as any redundant terms required to avoid hazards (due to single input variable changes).

(a) $f(a,b,c,d) = \sum(0,4,11,13,15) + D(2,3,5,10)$.

ab \ cd	00	01	11	10
00	1	0	X	X
01	1	X	0	0
11	0	1	1	0
10	0	0	1	X

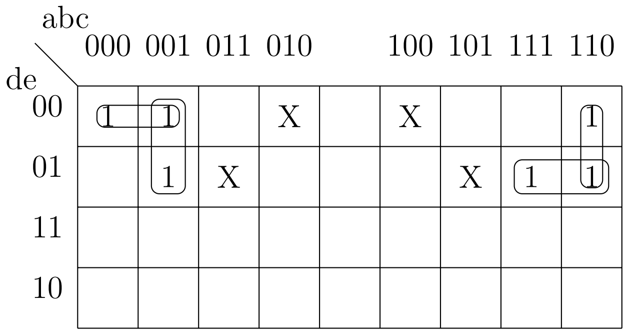
From this Karnaugh map, it actually appears the best way to implement the function is to **not** try and minimize it. Consider the following selection of product terms:



ab \ cd	00	01	11	10
00	1	0	X	X
01	1	X	0	0
11	0	1	1	0
10	0	0	1	X

Here $f = a'c'd + abd + acd$ and is hazard free. If we tried to replace the term acd with $b'c$ we would have a cheaper implementation of f, but we would then be required to add additional product terms (including acd) to remove hazards.

(b) $f(a,b,c,d,e) = \sum(0,4,5,24,25,29) + D(8,13,16,21)$.



de \ abc	000	001	011	010		100	101	111	110
00	1	1		X		X			1
01		1	X				X	1	1
11									
10									

The implementation $f = a'b'd'e' + a'b'cd' + abc'd' + abd'e$ has no hazards.

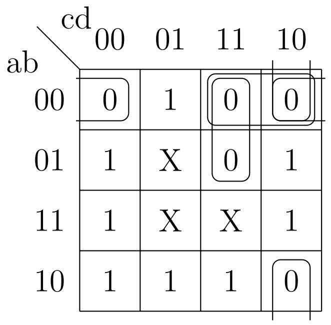
Q9: Finda hazard-free minimum cost POS implementations for the following functions:

(a) $f(a,b,c,d) = \Pi(0,2,3,7,10) + D(5,13,15)$.

(b) $f(a,b,c,d,e) = \Pi(2,6,7,25,28,29) + D(0,8,9,10,11,21,24,26,27,30)$.

Solution:

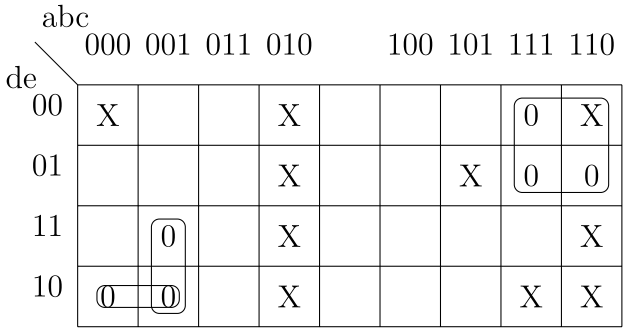
(a) $f(a,b,c,d) = \Pi(0,2,3,7,10) + D(5,13,15)$.



ab \ cd	00	01	11	10
00	0	1	0	0
01	1	X	0	1
11	1	X	X	1
10	1	1	1	0

The hazard free implementation is $f = (a+b+d)(a+c'+d')(b+c'+d)(a+b+c')$ and the last sum term is the required redundant sum term to avoid hazards.

(b) $f(a,b,c,d,e) = \Pi(2,6,7,25,28,29) + D(0,8,9,10,11,21,24,26,27,30)$.



de \ abc	000	001	011	010		100	101	111	110
00	X			X				0	X
01				X			X	0	0
11		0		X					X
10	0	0		X				X	X

The hazard free implementation is $f = (a+b+d'+e)(a+b+c'+d')(a'+b'+d)$.

Q10: Perform state minimization to reduce the following flow table to one with fewer states which exhibits the same functional behaviour.

Current State	Next State $x_1x_2 = 00$	01	11	10	Output z
a	$\boxed{a}$	b	c	-	0
b	k	$\boxed{b}$	-	h	0
c	f	-	$\boxed{c}$	m	0
d	$\boxed{d}$	e	j	-	1
e	a	$\boxed{e}$	-	m	0
f	$\boxed{f}$	l	j	-	0
g	d	$\boxed{g}$	-	h	0
h	-	g	j	$\boxed{h}$	1
j	f	-	$\boxed{j}$	h	0
k	$\boxed{k}$	l	c	-	1
l	a	$\boxed{l}$	-	h	0
m	-	g	c	$\boxed{m}$	1

Solution:
The implication chart is large so I am not going to show it. First, however, we need to take into account that the outputs (while shown as depending only on the state) have a lot of don't cares. The state table is really like this:

Current State	Next State $x_1x_2 = 00$	01	11	10	Output (z) $x_1x_2 = 00$	01	11	10
a	$\boxed{a}$	b	c	-	0	-	-	-
b	k	$\boxed{b}$	-	h	-	0	-	-
c	f	-	$\boxed{c}$	m	-	-	0	-
d	$\boxed{d}$	e	j	-	1	-	-	-
e	a	$\boxed{e}$	-	m	-	0	-	-
f	$\boxed{f}$	l	j	-	0	-	-	-
g	d	$\boxed{g}$	-	h	-	0	-	-
h	-	g	j	$\boxed{h}$	-	-	-	1
j	f	-	$\boxed{j}$	h	-	-	0	-
k	$\boxed{k}$	l	c	-	1	-	-	-
l	a	$\boxed{l}$	-	h	-	0	-	-
m	-	g	c	$\boxed{m}$	-	-	-	1

I believe if you do the implication chart, you will find that you can merge (c, m), (g, h) and (f, j). This gives the reduced chart:

Current	Next State				Output (z)			
State	$x_1x_2 = 00$	01	11	10	$x_1x_2 = 00$	01	11	10
a	$\boxed{a}$	b	c	-	0	-	-	-
b	k	$\boxed{b}$	-	g	-	0	-	-
c	f	g	$\boxed{c}$	$\boxed{c}$	-	-	0	1
d	$\boxed{d}$	e	f	-	1	-	-	-
e	a	$\boxed{e}$	-	c	-	0	-	-
f	$\boxed{f}$	l	$\boxed{f}$	g	0	-	0	-
g	d	$\boxed{g}$	f	$\boxed{g}$	-	0	-	1
k	$\boxed{k}$	l	c	-	1	-	-	-
l	a	$\boxed{l}$	-	g	-	0	-	-

Q11: Consider the following flow table.

Current	Next State				Output (z)			
State	$x_1x_2 = 00$	01	11	10	$x_1x_2 = 00$	01	11	10
a	$\boxed{a}$	b	d	$\boxed{a}$	0	-	1	1
b	c	$\boxed{b}$	$\boxed{b}$	d	0	1	0	0
c	$\boxed{c}$	$\boxed{c}$	b	a	0	1	0	1
d	a	c	$\boxed{d}$	$\boxed{d}$	-	-	1	0

Find a suitable race-free state assignment which

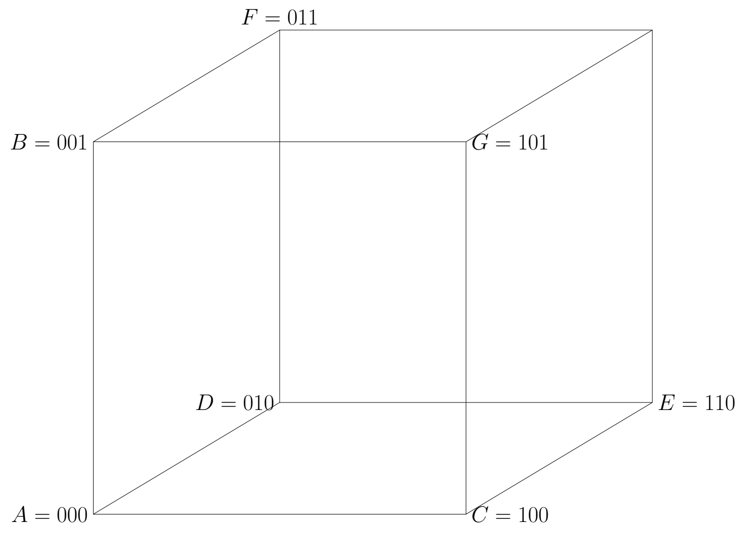
(a) Uses as few states as possible.

(b) Uses the method of state duplication.

(c) Uses the method of one-hot encoding.

Solution:

Using as few states as possible (this requires some trial and error)...



This embedding and assignment leads to...

Current	Next State				Output (z)			
State	$x_1x_2 = 00$	01	11	10	$x_1x_2 = 00$	01	11	10
a	$\boxed{a}$	b	d	$\boxed{a}$	0	-	1	1
b	g	$\boxed{b}$	$\boxed{b}$	f	0	1	0	0
c	$\boxed{c}$	$\boxed{c}$	g	a	0	1	0	1
d	a	e	$\boxed{d}$	$\boxed{d}$	-	-	1	0
e	-	c	-	-	-	1	-	-
f	-	-	-	d	-	-	-	0
g	c	-	b	-	0	-	0	-

Using state duplication...

Current	Next State				Output (z)			
State	$x_1x_2 = 00$	01	11	10	$x_1x_2 = 00$	01	11	10
a0	$\boxed{a0}$	b0	*a1*	$\boxed{a0}$	0	-	1	1
a1	$\boxed{a1}$	b1	d1	$\boxed{a1}$	0	-	1	1
b0	c0	$\boxed{b0}$	$\boxed{b0}$	*b1*	0	1	0	0
b1	*c0*	$\boxed{b1}$	$\boxed{b1}$	*d0*	0	1	0	0
c0	$\boxed{c0}$	$\boxed{c0}$	b0	*c1*	0	1	0	1
c1	$\boxed{c1}$	$\boxed{c1}$	*c0*	*a0*	0	1	0	1
d0	*d1*	c0	$\boxed{d0}$	$\boxed{d0}$	-	-	1	0
d1	a1	c1	$\boxed{d1}$	$\boxed{d1}$	-	-	1	0

Using one hot encoding...

Assignment	Current State	Next State $x_1x_2 = 00$	01	11	10	Output (z) $x_1x_2 = 00$	01	11	10
0001	a	$\boxed{a}$	e	f	$\boxed{a}$	0	-	1	1
0010	b	g	$\boxed{b}$	$\boxed{b}$	h	0	1	0	0
0100	c	$\boxed{c}$	$\boxed{c}$	g	i	0	1	0	1
1000	d	f	j	$\boxed{d}$	$\boxed{d}$	-	-	1	0
0011	e	-	b	-	-	-	1	-	-
1001	f	a	-	d	-	0	-	1	-
0110	g	c	-	b	-	0	-	0	-
1010	h	-	-	-	d	-	-	-	0
0101	i	-	-	-	a	-	-	-	1
1100	j	-	c	-	-	-	1	-	-

Q12: Consider an asynchronous sequential circuit defined by the equations $Y = x_1x_2 + (x_1 + x_2)y$ and $z = y$. Implement this circuit using a NOR SR latch. Implement this circuit using a NAND $S'R'$ latch.

Solution:

Using a **NOR** latch...

Write down the transition table and figure out what the SR inputs need to be to get the desired output transitions. Should recall how a **NOR** latch output *changes* based on the inputs:

$y \to Y$	S	R	
$0 \to 0$	0	X	$\leftarrow$ *reset or hold*
$0 \to 1$	1	0	$\leftarrow$ *set*
$1 \to 0$	0	1	$\leftarrow$ *reset*
$1 \to 1$	X	0	$\leftarrow$ *set or hold*

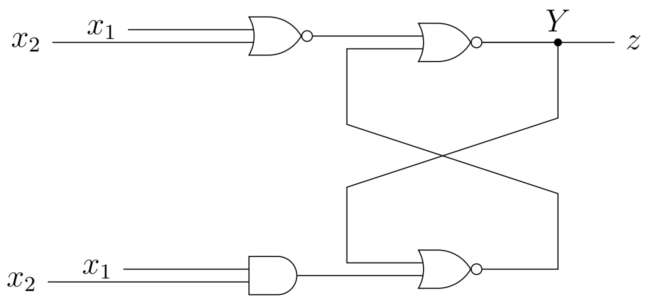
Also, it never illustrates the $SR = 11$ restricted case.

The transition table and the SR inputs...

Current state	Next state (Y)				Latch inputs (SR)			
y	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$
0	0	0	1	0	$0X$	$0X$	10	$0X$
1	0	1	1	1	01	$X0$	$X0$	$X0$

We can now write down optimized logic equations for S and R. Since there are don't cares, *we should try to avoid the situation where $SR = 11$ can happen to avoid the restricted case.* We find $S = x_1x_2$ and $R = \overline{x}_1\overline{x}_2 = \overline{x_1 + x_2}$. Note that $SR = x_1x_2\overline{x}_1\overline{x}_2 = 0$ and therefore our circuit will never encounter the case $SR = 11$.

The circuit:



Using a **NOR** latch...

Write down the transition table and figure out what the SR inputs need to be to get the desired output transitions. Should recall how a **NAND** latch output *changes* based on the inputs:

$y \to Y$	S	R	
$0 \to 0$	1	X	$\leftarrow$ *reset or hold*
$0 \to 1$	0	1	$\leftarrow$ *set*
$1 \to 0$	1	0	$\leftarrow$ *reset*
$1 \to 1$	X	1	$\leftarrow$ *set or hold*

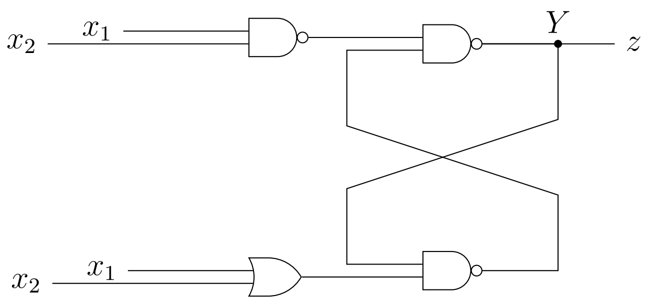
Also, it never illustrates the $SR = 00$ restricted case.

The transition table and the SR inputs...

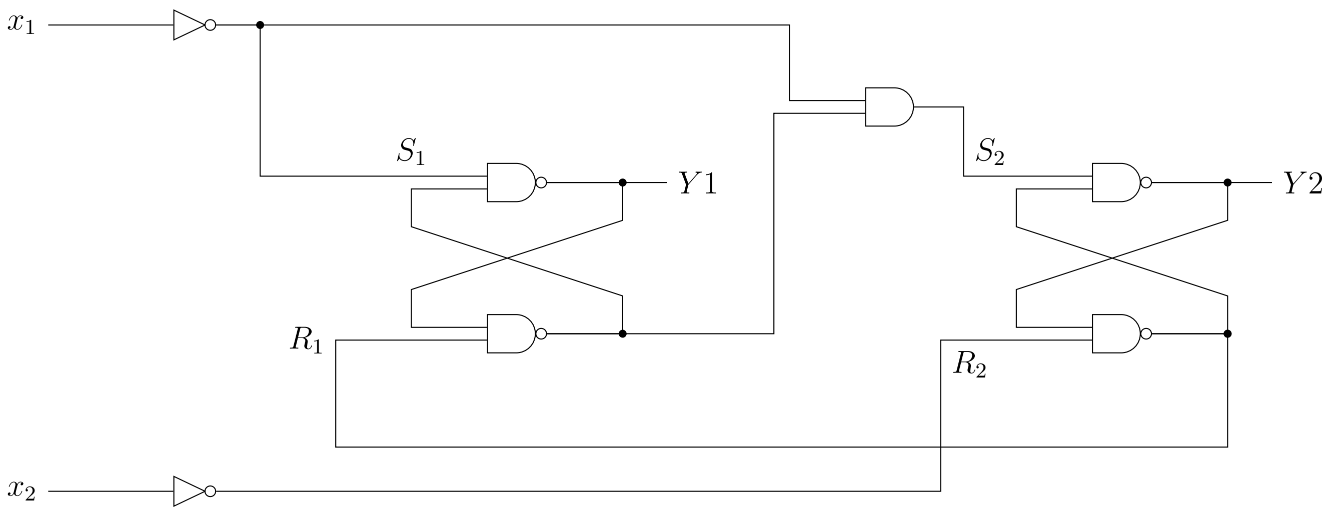
Current state	Next state (Y)				Latch inputs (SR)			
y	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$
0	0	0	1	0	$1X$	$1X$	01	$1X$
1	0	1	1	1	10	$X1$	$X1$	$X1$

We can now write down optimized logic equations for S and R. We find $S = \overline{x}_1 + \overline{x}_2 = \overline{x_1x_2}$ and $R = x_1 + x_2$ Since there are don't cares, *we should try to avoid the situation where* $SR = 00$ *can happen to avoid the restricted case.* Note that $S + R = x_1 + x_2 + \overline{x}_1 + \overline{x}_2 = 1$ and therefore our circuit will never encounter the case $SR = 00$.

The circuit:



Q13: Consider the circuit shown below which consists of two $S'R'$ latches.



Write down equations for the latch inputs S_1, R_1, S_2 and R_2. Derive equations for the latch outputs Y_1 and Y_2. Derive a transition table for the circuit.

Solution:
Equations for the latch inputs:

$$\begin{aligned} S_1 &= \overline{x}_1 \\ R_1 &= \overline{y_2\overline{x}_2} = \overline{y}_2 + x_2 \\ S_2 &= \overline{x}_1\overline{y_1R_1} = \overline{x}_1(\overline{y}_1 + \overline{R}_1 = \overline{x}_1(\overline{y}_1 + y_2\overline{x}_2) = \overline{x}_1\overline{y}_1 + \overline{x}_1\overline{x}_2y_2 \\ R_2 &= \overline{x}_2 \end{aligned}$$

Notice that these equations were **not** written using the complementary output from each latch. We can't do this because of the restricted case of the latch — we must write everything in terms of inputs and secondary (current state) variables.

The logic equations for the excitation (next state) variables are

$$\begin{aligned} Y_1 &= \overline{S_1\overline{R_1y_1}} = \overline{S}_1 + R_1y_1 = x_1 + x_2y_1 + y_1\overline{y}_2 \\ Y_2 &= \overline{S_2\overline{R_2y_2}} = \overline{S}_2 + R_2y_2 = x_1 + x_2y_1 + y_1\overline{y}_2 + \overline{x}_2y_2 \end{aligned}$$

Using the excitation variable equations we can write down the following transition table:

Current state	Next state (Y_1Y_2)			
y_1y_2	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$
00	$\boxed{00}$	$\boxed{00}$	11	11
01	$\boxed{01}$	00	11	11
10	11	11	11	11
11	01	$\boxed{11}$	$\boxed{11}$	$\boxed{11}$