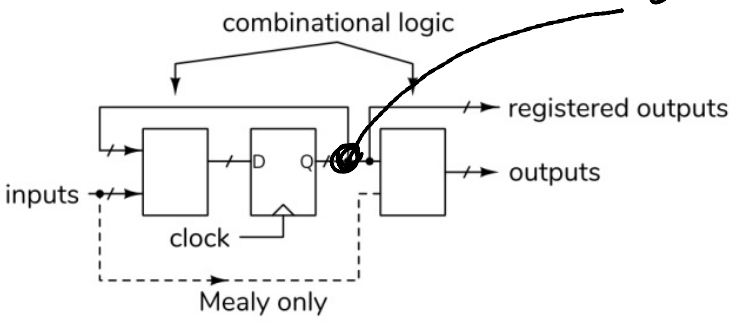
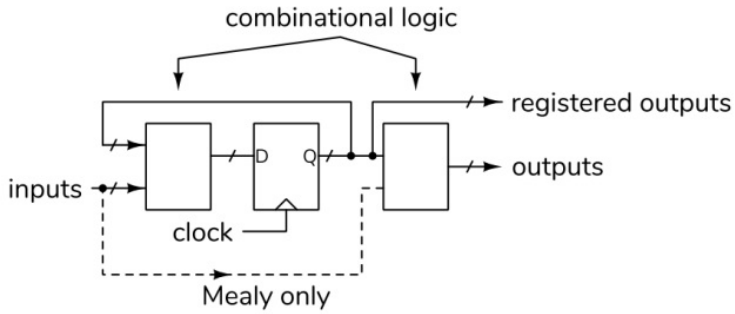
State Machines

Exercise 1: Which signals in the above diagrams indicate the current state?

output of register is the "state".

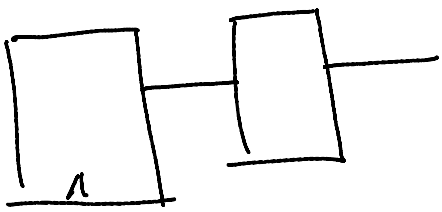


Exercise 2: Which outputs are registered? Which outputs could change whenever the input changes?



only SM can output change when input changes.

Exercise 3: Why?



output = f (state)

State	output
0	A
1	B
2	C
3	C
4	D

because the output is only a function of the state.

Exercise 4: If we used 8-bits of state information, how many states could be represented? What if we used 8 bits of state but used a "one-hot" encoding?

$2^k = 2^8 = 256$

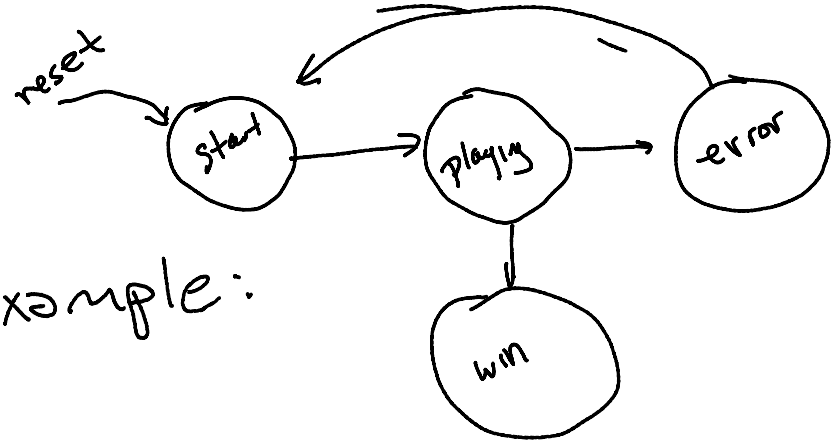
$k = 8$ states

0000 0000 → 1111 1111

1000 0000, 0100 0000,

... 0000 0001

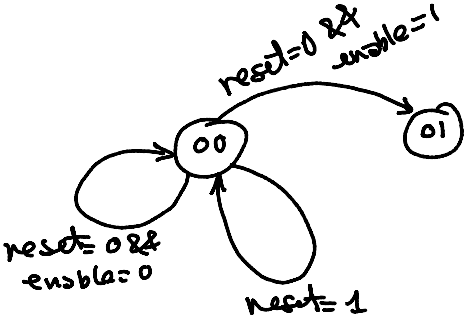
Exercise 5: The link below describes a game. List the top-level game states. Decompose each of these into multiple states. Repeat.



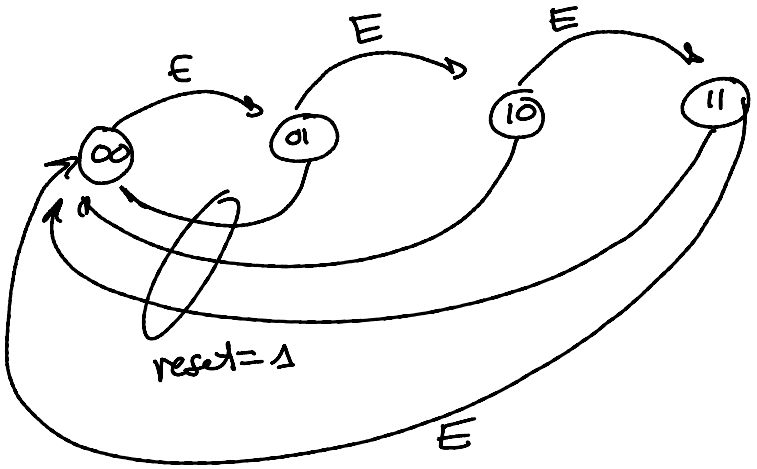
Exercise 6: What happens if both reset and enable are asserted? reset

Exercise 7: Draw the state transition diagram.

count				next count	
[1]	[0]	reset	enable	[1]	[0]
0	0	0	1	0	1
0	1	0	1	1	0
1	0	0	1	1	1
1	1	0	1	0	0
a	b	0	0	a	b
X	X	1	X	0	0

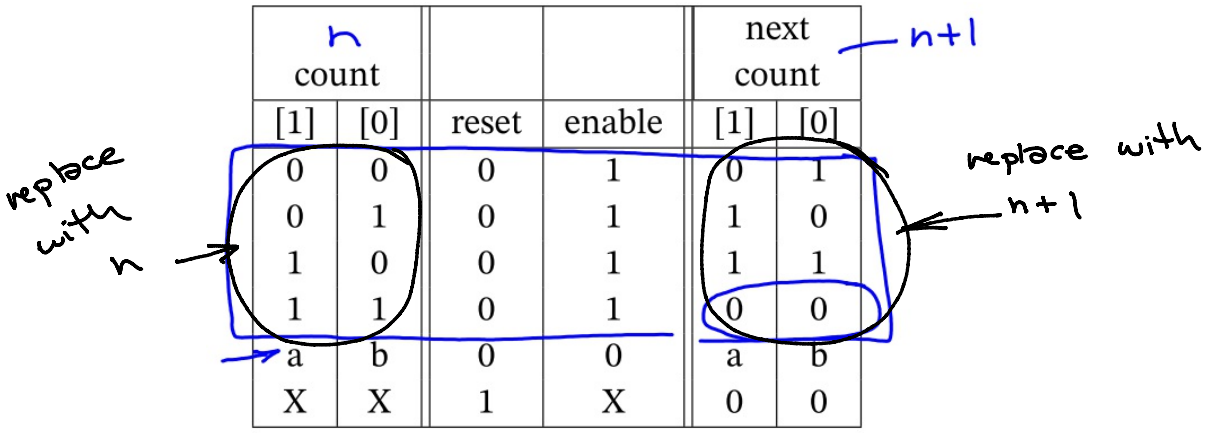


10 11



E: enable=1 && reset==0

Exercise 8: Rewrite the state transition table and the code using u□nsigned. signals ň. and ň+1..



n count				next count		n+1
[1]	[0]	reset	enable	[1]	[0]	
0	0	0	1	0	1	
0	1	0	1	1	0	
1	0	0	1	1	1	
1	1	0	1	0	0	
a	b	0	0	a	b	
X	X	1	X	0	0	

Exercise 9: Write the state transition tables for the counter and light sequencer.

state	count	state_next
rg	0	rg
.	.	.
.	.	.
.	.	.

count	State	count_next
n	X	n-1
0	rg llgr	4
0	rg llyr	2 9