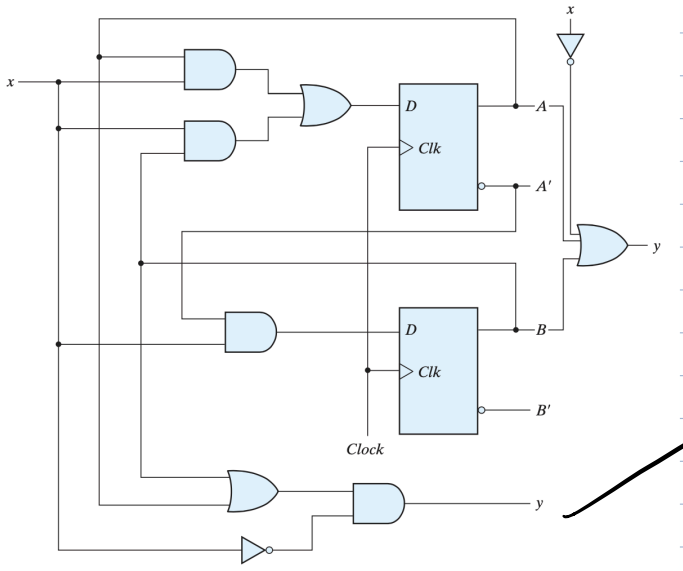
ANALYSIS OF SEQUENTIAL CIRCUITS

- Analysis consists of obtaining table/diagram for the sequence of inputs, outputs, & internal states.

State equations

Consider:



Since A will be set to D,

$A(t+1) = A(t) \cdot x(t) + B(t) \cdot x(t) = (A+B)x$

$B(t+1) = A'(t) \cdot x(t)$

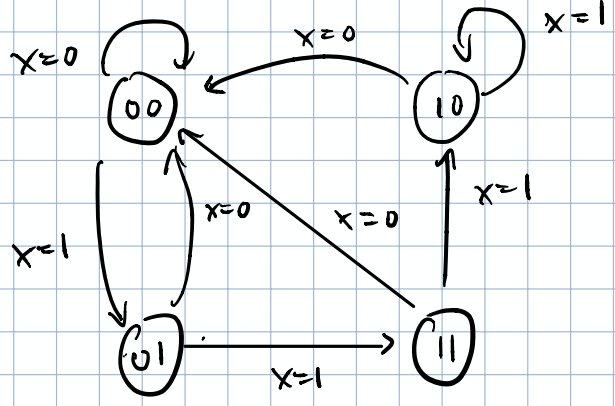
↳ these are derived from combinational logic

$Y(t) = (A+B)x'$

P.S. A	P.S. B	input x	N.S. A	N.S. B	output Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	0

or

P.S.	N.S. x=0	N.S. x=1	output x=0	output x=1
AB	AB	AB	Y	Y
0 0	0 0	0 1	0	0
0 1	0 0	1 1	1	0
1 0	0 0	1 0	1	0
1 1	0 0	1 0	1	0



if x=0, reset to 00

if x=1, advance to 10.

→ output 0 as long as next input is 1.

→ output 1 when next input 0 and prev state was 00.

→ this detects 0s in sequence.

Circuit diagram ↔ equations ↔ state table ↔ state diagram

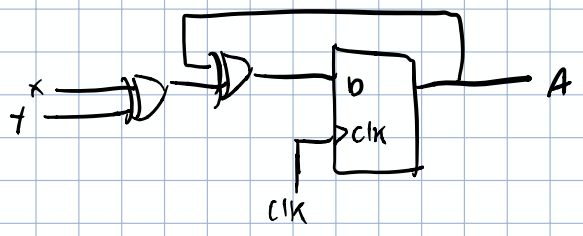
we can designate the logic of each flip flop:

$$D_A = Ax + Bx$$
$$D_B = A'x$$
$$y = (A+B)x$$

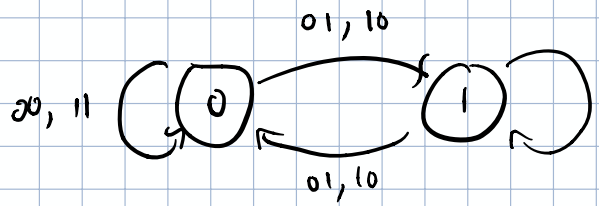
This allows us to simplify the analysis of flip flops.

ex.

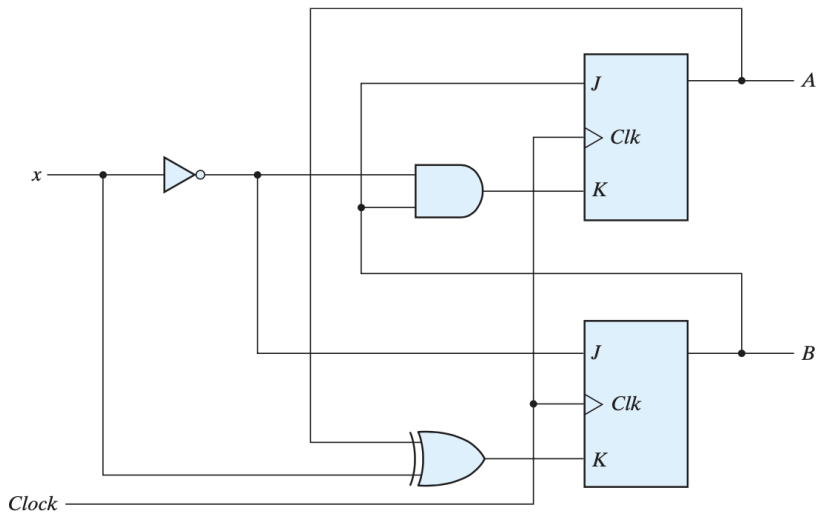
$$D_A = A \oplus x \oplus y$$



A	x	y	A(t+1)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



we can also analyze with JK flip flops:



$J_A = B$ $\quad K_A = Bx'$

$J_B = x'$ $\quad K_B = A \oplus x$

P.S.		input	N.S.		flip flop			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	1
1	0	1	1	0	0	0	0	0
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	0

- next state determined by substituting JK flip flop values into char. eqn.

$$A(t+1) = JA' + K'A$$

$$B(t+1) = JB' + K'B$$

$$A(t+1) = BA' + (B' + x)A$$

$$= A'B + AB' + Ax$$

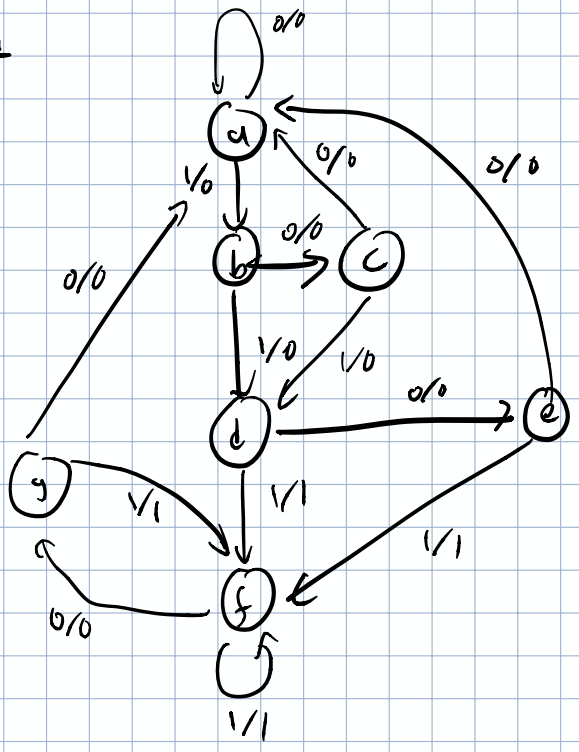
$$B(t+1) = x'B' + (A \oplus x)'B$$

$$= B'x' + ABx + A'Bx'$$

STATE REDUCTION / ASSIGNMENT

- m flip-flops → 2^m states
- reduction may or may not reduce # of flip flops
  → if # FF reduced, may need more complex combinational logic.

ex.



state	a	a	b	c	d	e	f	f	g	f	g	a
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

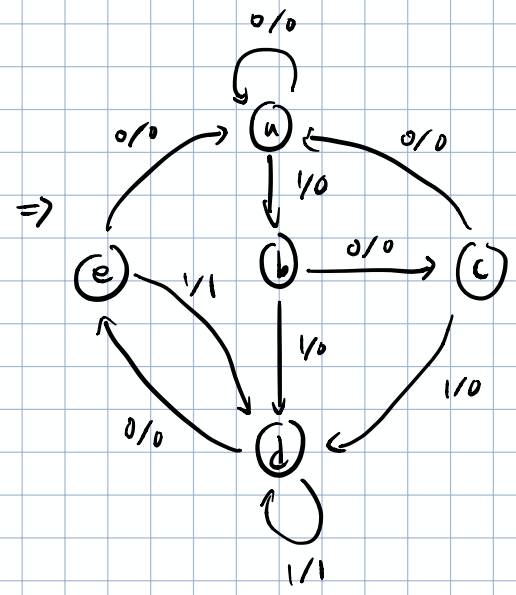
- to reduce states, make state table first.

P.S	N.S. x=0	N.S. x=1	Output x=0	Output x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	~~f~~ d	0	1
e	a	~~f~~ d	0	1
f	~~g~~ e	~~f~~ d	0	1
g	a	~~f~~ d	0	1

- when 2 states are equivalent (if ∀ inputs → same/equivalent states)

ex.
- e and g. we now replace g with e.
- f and d are now equivalent. Replace f w/ d.

P.S	n.s. x=0	n.s. x=1	Output x=0	Output x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1



STATE ASSIGNMENT

- m states require n bits to represent, where $2^n \geq m$ or $n \geq \log_2 m$

3 possible ways of assigning states:

1) Binary
2) Grey code
3) 1-hot

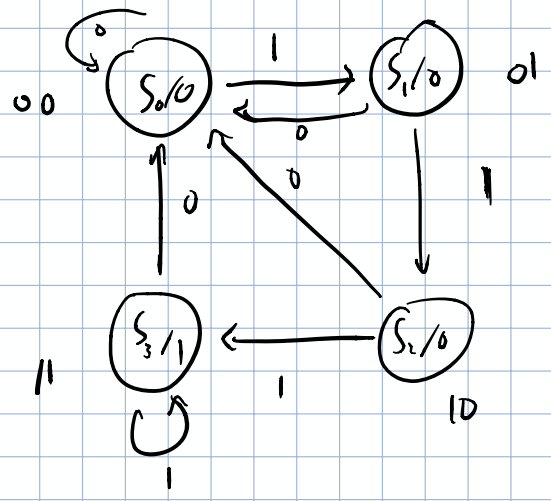
State	Binary	Grey	One-hot
a	000	000	00001
b	001	001	00010
c	010	011	00100
d	011	010	01000
e	100	110	10000

- One hot usually leads to easier decoding logic

Design Procedure

1) derive state diagram
2) reduce # states if necessary
3) assign binary values
4) obtain state table
5) choose flip-flops
6) derive flip flop input/output equations
7) draw the diagram

ex: Design circuit for detecting sequence of 3+ consecutive 1s.



4 states ↔ 2 bits → 2 flip flops

State	input 0	input 1	output 0	output 1
S_0	S_0	S_1	0	0
S_1	S_0	S_2	0	0
S_2	S_0	S_3	0	0
S_3	S_0	S_3	1	1

we wanna do K-map, so maybe this is better:

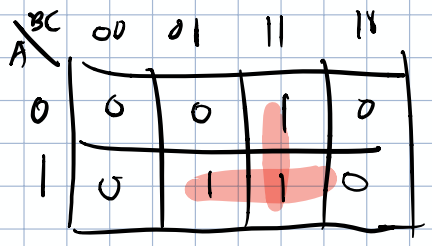
	State	input	n.s.		output
00	S_0	0	S_0	00	0
00	S_0	1	S_1	01	0
01	S_1	0	S_0	00	0
01	S_1	1	S_2	10	0
10	S_2	0	S_0	00	0
10	S_2	1	S_3	11	0
11	S_3	0	S_0	00	1
11	S_3	1	S_3	11	1

let $S_0 = 00$, $S_1 = 01$, $S_2 = 10$, $S_3 = 11$ (A B)

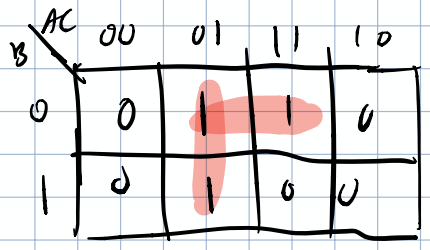
$$A(t+1) = \Sigma\,(3, 5, 7) = A'BC + AB'C + ABC$$

$$B(t+1) = \Sigma\,(1, 5, 7) = A'B'C + AB'C + ABC$$

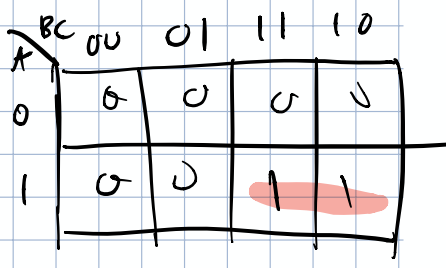
$$Y = \Sigma\,(6, 7) = ABC' + ABC$$



$A = AC + BC = (A+B)C$



$B = A'B + B'C$



$Y = AB$

$D_A = (A+B)x$

$D_B = Ax + B'x$

$Y = AB$

- using other types of flip flops other than D is complicated
  - → to determine next state, we need to derive relationship btw. state tables & input equations

excitation tables

Q(t)	Q(t+1)	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

JK Flip Flop

Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

T Flip Flop

ex:

p.s. A	p.s. B	input x	n.s. A	n.s. B	flip flop inputs → derived from excitation table. J_A	K_A	J_B	K_B
0	0	0	0	0	0	x	0	x
0	0	1	0	1	0	x	1	x
0	1	0	1	0	1	x	x	1
0	1	1	0	1	0	x	x	0
1	0	0	1	0	x	0	0	x
1	0	1	1	1	x	0	0	x
1	1	0	1	1	x	0	x	0
1	1	1	0	0	x	1	x	1