Design Procedure

Example: Design a gated latch circuit with two inputs G (gate) and D (Data), and one output Q. The value of D is transferred to Q when G=1. when G=0 the value of Q will not charge.

Solution

Step 1: get the primitive flow table (flow table with only one stable total state in each row), and the total state consists of the internal state combined with inputs ⇒ in this example we may have up to 8 ~~total stable~~ stable total states

D	G	Q	Stable ??
0	0	0	yes
0	0	1	yes
0	1	0	yes
0	1	1	no
1	0	0	yes
1	0	1	yes
1	1	0	no
1	1	1	yes

⇒ In this example we have 6 stable total states, let us now rearrange them

	Inputs		output		after state b or e
state	D	G	Q	~~b,c,f~~	comments
a	0	1	0	b,c,f	Q = D ~~D = Q~~ because G = 1
b	1	1	1	after a,d,e	Q = D ~~D = Q~~ // G = 1
c	0	0	0		after state a or d
d	1	0	0		after state c
e	1	0	1		after state b or f
f	0	0	1		after state e

⇒ now we can draw the flow table which has one row for each state and one column for each input combination ⇒ the flow table for this example will have 6 rows & 4 columns

DG	00	01	11	10
a	c, –	ⓐ, 0	b, –	–, –
b	–, –	a, –	ⓑ, 1	e, –
c	ⓒ, 0	a, –	–, –	d, –
d	c, –	–, –	b, –	ⓓ, 0
e	f, –	–, –	b, –	ⓔ, 1
f	ⓕ, 1	a, –	–, –	e, –

a. fill in one square in each row belongin to the stable state in that row

b. because both inputs are not allowed to change simultaneously we can enter dash marks (don't care) in each row that differs in two or more variable from the input variable associated with stable states.

e. to fill the empty squares, we will use the comments in the previous table to see how we can reach a stable total state

Step 2: use the implication table to merge the flow table by finding the maximal compatibles of the total states

- any two states may be
  - a. incompatible
  - b. compatible
  - c. equivalent

any equivalent pair of states are compatible.
any incompatible pair of states are inequivalent.
two compatible states are not necessarly to be equivalent.

~~Example of using Implication table to find equivalent states.~~ Two states are said to be equivalent if they have the same output and go to the same or equivalent next states

Ex:

Present state	next state $x=0$	next state $x=1$	output $x=0$	output $x=1$
a	c	b	0	1
b	d	a	0	1
c	a	d	1	0
d	b	d	1	0

b	c,d		
c	X	X	
d	X	X	a,b
	a	b	c

this means that (a,b) are equivalent imply (c,d) and (c,d) imply (a,b)

⟹ a equivalent to b and
c equivalent to d

$(a,b),(c,d)$
$\downarrow \quad\quad \downarrow$
$a \quad , \quad c$

Ex.

Present state	next state $x=0$	next state $x=1$	output $x=0$	output $x=1$
a	d	a	0	0
b	e	a	0	0
c	g	f	0	1
d	a	d	1	0
e	a	d	1	0
f	c	b	0	0
g	a	e	1	0

b	d,e ✓					
c	X	X				
d	X	X	X			
e	X	X	X	✓		
f	c,d X	c,e a,b X	X	X	X	
g	X	X	X	d,e ✓	d,e ✓	X
	a	b	c	d	e	f

(a, b), (d, e), (d, g), (e, g)

⇒ (a, b), (c), (d, e, g), (f)

↓

(a), (c), (d), (f)

⊛ compatible states

Two states are said to be compatible if for each possible input they have the same output whenever specified and their next state are compatible whenever they are specified. All don't care conditions marked with dashes have no effect when searching for compatibible states because they represent unspecified condition

b	✓				
c	✓	d, e ✗			
d	✓	d, e ✗	✓		
e	c, f ✗	✓	c, f ✗ d, e	✗	
f	c, f ✗	✓	✗	✗	✓
	a	b	c	d	e

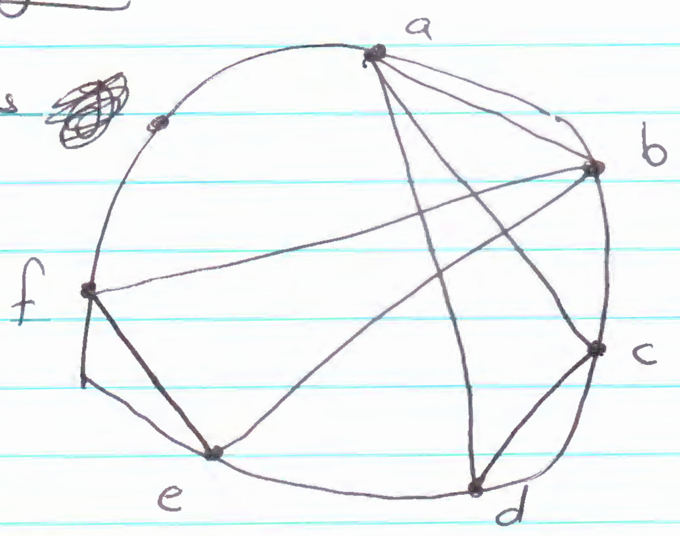
⇒ the compatible pairs are

(a, b) (a, c) (a, d) (b, e) (b, f) (c, d) (e, f)

- Having found all compatible pairs, the next step is to find larger sets of states that are compatible. The maximal compatible is a group of compatibles that contains all the possible combinations of compatible states.

using <u>merger-diagram</u>

⇒ maximal compatibles (a, c, d), (b, e, f), (a, b).



- after finding the maximal compatibles, we must find a minimal collection of compatibles that covers all the states and is closed.

The set will cover all the states if it inclu all the states of the original table.

The closure condition is satisfied if there are no implied states or if the implied states are included within the set

⇒ in our example, minimal collection of compatibles are (a, c, d) and (b, e, f)

this collection includes all states and it is closed because there is no implied states.

(a, c, d) ⇓ can be merged in one state (one row)

(b, e, f) ⇓ can be merged in one state (one row)

	DG 00	01	11	10
a, c, d	ⓒ, 0	ⓐ, 0	b, −	ⓓ, 0
b, e, f	ⓕ, 1	a, −	ⓑ, 1	ⓔ, 1

=

	DG 00	01	11	10
a	ⓐ, 0	ⓐ, 0	b, −	ⓐ, 0
b	ⓑ, 1	a, −	ⓑ, 1	ⓑ, 1

④ It is now the time for state assignment, in state assignment we should try to avoid changes of more than one internal state variable when there is a change in the input (This is in order to avoid critical race).

In this example, there is no critical race because we have only 2 states (1 internal state variable y).

- assign a = 0 , b = 1

⇒ Transition table and output maps are

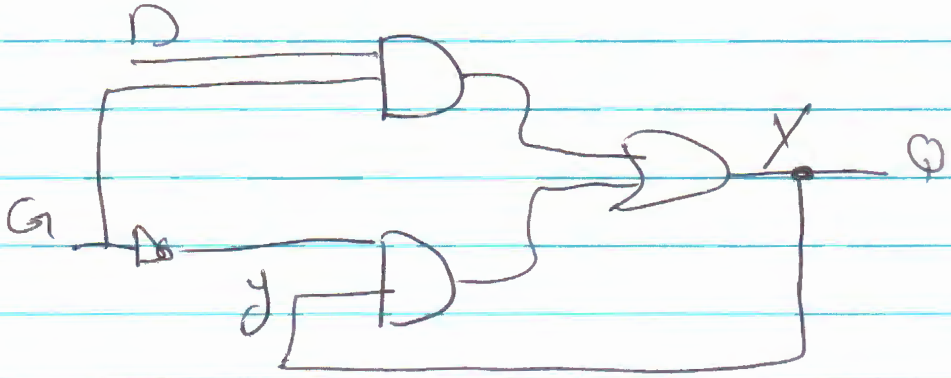
y \ DG	00	01	11	10
0	0	0	1	0
1	1	0	1	1

$Y = DG + G'y$

y \ DG	00	01	11	10
0	0	0	x	0
1	1	x	1	1

$Q = y$

or $Q = Y$



④ map for S and R (using and gate

y \ DG	00	01	11	10
0	0	0	1	0
1	X	0	X	X

$S = DG$

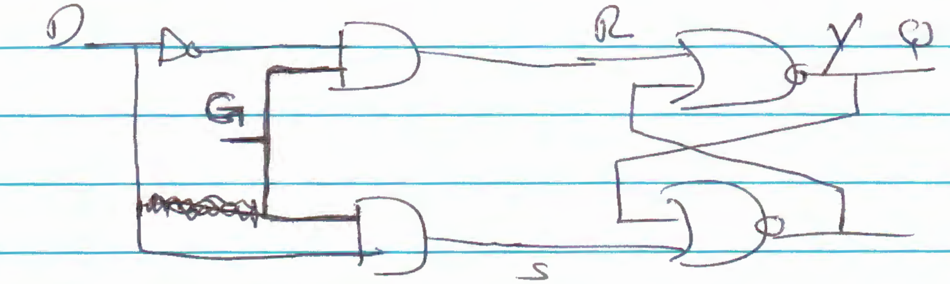
y \ DG	00	01	11	10
0	X	X	0	X
1	0	1	0	0

$R = D'G$

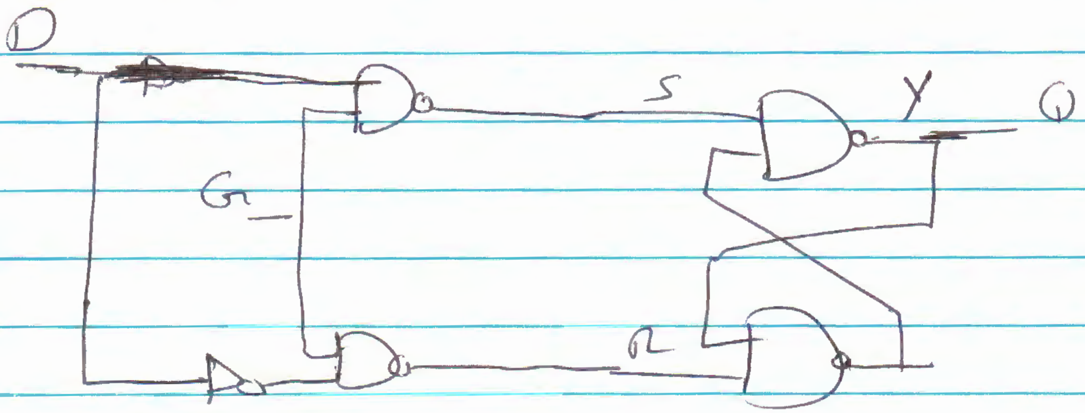
Q(t)	Q(t+1)	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

~~$Y = DG + D'G$~~

$Y = S + R'Y$ (2 nor gates)



$Y = S' + RY$ (2 nand gate



⊛ Assigning Outputs to Unstable states

→ The stable states in a flow table have specific output entries associated with them

- The unstable states have unspecified output entries designated by dash (-)
- The output values for the unstable states must be chosen so that no momentary false outputs occur when the circuit switches between stable states ⇒ use the following rules

① Assign the output 0 when the transition occurs between two states with 0 outputs

② Assign the output 1 when the transition occurs between two states with 1 outputs

③ Assign ~~don't~~ the output don't care (X) when the transition between two states with different outputs

Ex. given the following flow table

	x = 0	x = 1
a	ⓐ 0	b, -
b	c, -	ⓑ 0
c	ⓒ 1	d, -
d	a, -	ⓓ 1

⇒

0	0
X	0
1	1
X	1

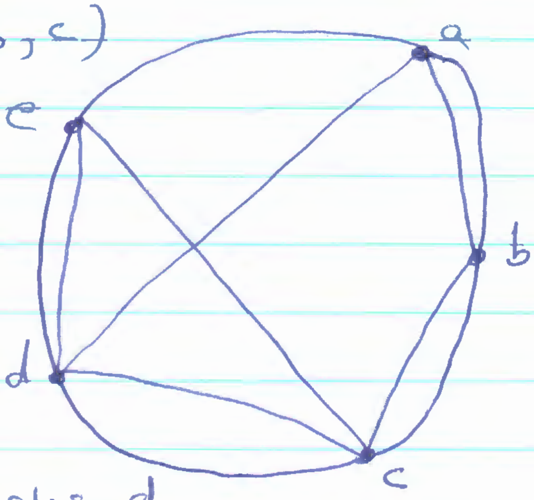
⊛ closed covering condition

Example with implied States:

b	b, c ✓			
c	X	d, e ✓		
d	b, c ✓	X	a, d ✓	
e	X	X	✓	b, c ✓
	a	b	c	d

⇒ compatible pairs are
(a, b), (a, d), (b, c), (c, d), (c, e), (d, e)

⇒ maximal compatibles
(c, d, e), (a, b), (a, d), (b, c)



now we want to find the ~~minimum~~ minimum set of compatibles that covers all the state and is closed

- (a, b), (c, d, e) cover all states but it is not closed

closure table

compatibles	(a, b)	(a, d)	(b, c)	(c, d, e)
Implied states	b, c	b, c	d, e	a, d b, c

⇒ A set of compatibles that will satisfy the closed covering condition is
(a, d) (b, c) (c, d, e)

⇒ we reduce the table from 5 rows to 3 rows.

- note that another closed-covered compatibles would be (a, b), (b, c), (d, e)
(In general, there may be more than one possible way of merging rows).

⊛ Race-Free state Assignment

- in our assignment for the states in a flow table we should prevent the occurance of critical races.
- critical races can be avoided if we assign the states such that only one change occur when we go from an internal state to another.

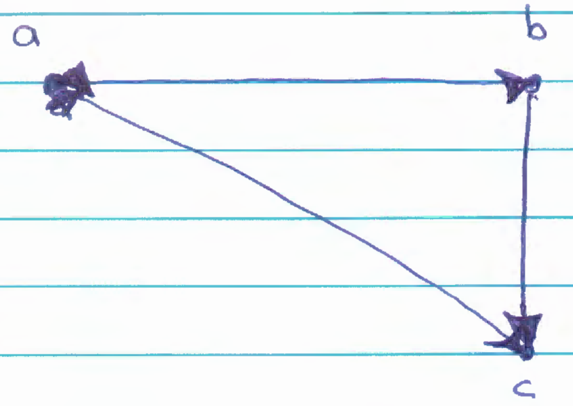
- In the Two-Row flow table we don't have any problem with assignment, but for 3-rows and more we may have some problems.

Example of 3-row flow table

x_1x_2

	00	01	11	10
a	ⓐ	b	c	ⓐ
b	a	ⓑ	ⓑ	c
c	a	ⓒ	ⓒ	ⓒ

- The information inside flow-table about the transitions between states is transferred to a transition diagram



transition diagram

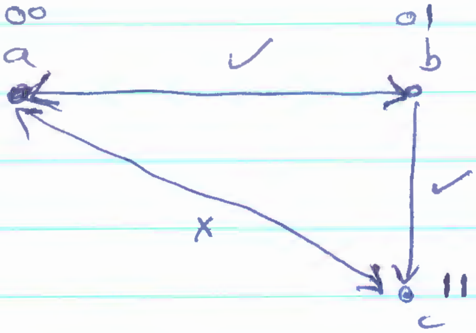
assignmt $a = 00$ $b = 01$ $c = 10$

x

$a = 00$ $b = 01$ $c = 11$

x

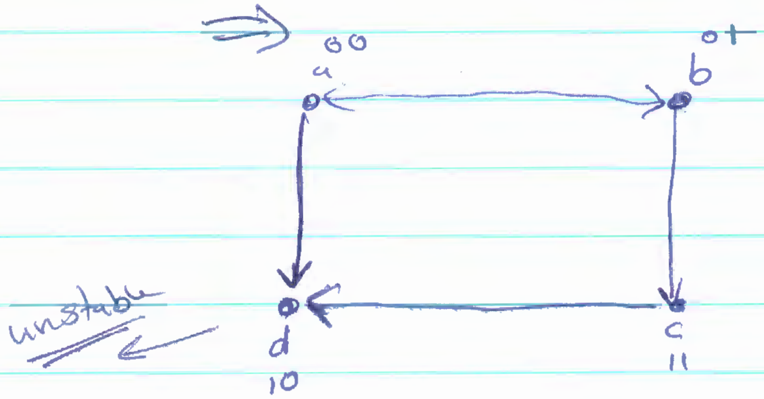
- This problem can be solved by adding a fourth row (d) to the flow table in order to form a cycle between stable states



from a to c critical

from c to a non-critical

show transition table



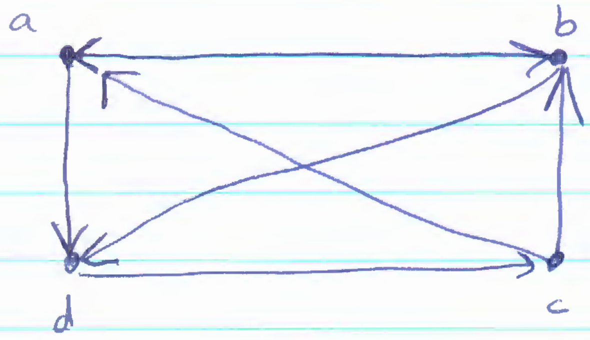
	x_1x_2			
	00	01	11	10
a	ⓐ	b	d	ⓐ
b	a	ⓑ	ⓑ	c
c	d	ⓒ	ⓒ	ⓒ
d	a	-	c	-

- the dashes in row d can be considered as don't care <u>BUT</u> must not be assigned a value of <u>10</u> because the state will be stable in this case.

Example of Four-Row flow table

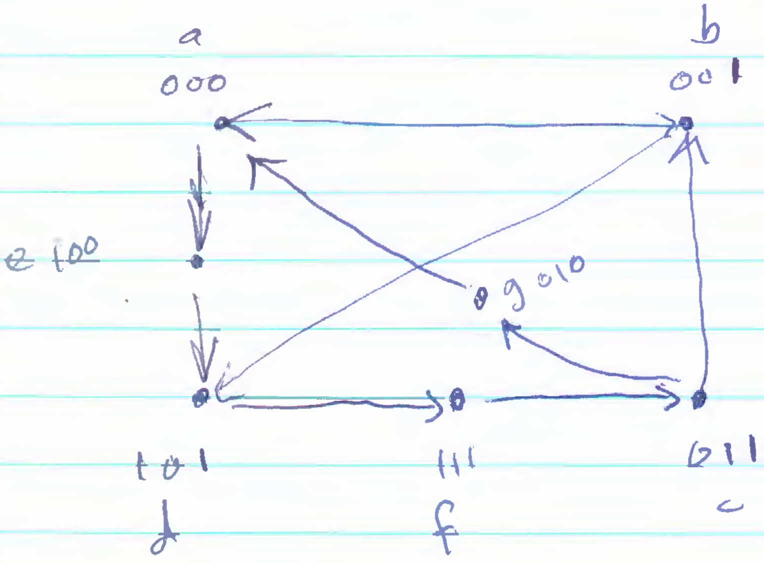
	x_1x_2			
	00	01	11	10
a	b	ⓐ	d	ⓐ
b	ⓑ	d	ⓑ	a
c	ⓒ	a	b	ⓒ
d	c	ⓓ	ⓓ	c

⇒ transition table



00 01 11 10

000 001 | 011 010 | 110 111 101 100



		x_1x_2			
		00	01	11	10
a	000	b	ⓐ	e	ⓐ
b	001	ⓑ	d	ⓑ	a
c	011	ⓒ	g	b	ⓒ
g	010	–	a	–	–
–	110	–	–	–	–
f	111	c	–	–	c
d	101	f	ⓓ	ⓓ	f
e	100	–	–	d	–

H.W

9-2 , 9-4 , 9-6 , 9-9 , 9-12 , 9-14 ,
9-15 , 9-18 , 9-19 , 9-22