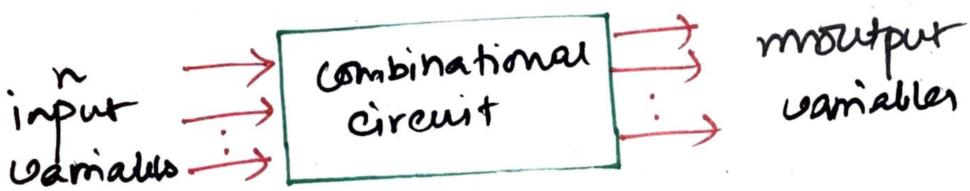
WEEK ~~13~~ 14

Combinational Circuits

a connected arrangement of logic gates with a ~~lot~~ set of inputs & outputs.

a) n binary input variables
b) m binary output variables
c) interconnection of logic gates



- generate binary control decisions & provides digital components required for data processing.

- ~~can~~ Can be specified with Boolean functions, one for each output variable.

Combinational circuit design

- Problem Def.
- input, output variable
- Truth table that defines input/output relationship.
- Simplified Boolean fcn. for each output
- logic diagram drawn.

Example: Half adder, full adder →
two simple arithmatic circuit
→ basic building blocks for more complex arithmatic circuit design.

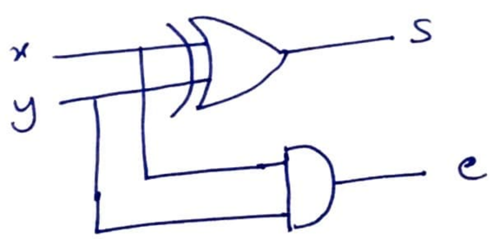
Half Adder & Full Adder:

- addition of two bits → half-adder
- addition of 3 bits (two significant bits & a precious carry)
  → Full-adder.
  - 2 half-adders are needed to implement a full adder.

Half-adder:

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Truth table of HA



Logic diagram of HA

$S = x'y + xy' = x \oplus y$ (XOR gate)

$c = xy$ (AND gate)

Full-Adder:

inputs			outputs	
x	y	z	c	s
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

Truth table of FA

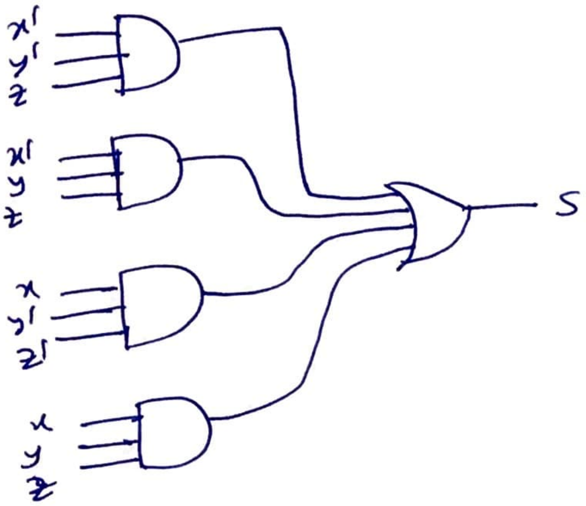
x, y → two significant bits to be added

z → carry from the previous lower significant position.

$S = x'y'z + x'yz' + xy'z' + xyz$
$= x \oplus y \oplus z.$

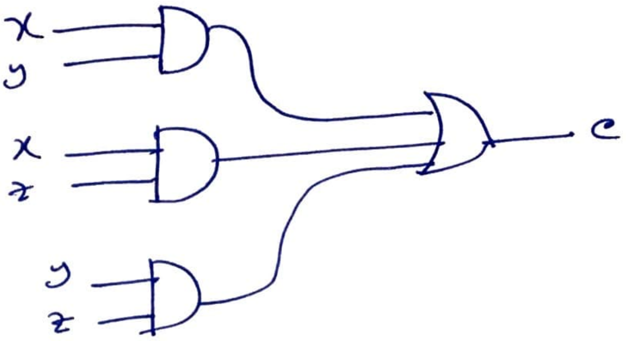
Implementation of Full Adder in SOPs

$S = x'y'z + x'yz' + xy'z' + xyz$, $C = xy + xz + yz$



$x \backslash yz$	00	01	11	10
0			1	
x { 1		1	1	1

$\underbrace{11}^{y}$, xz, z, xz



Subtractors

Half - Subtractor

$D = x'y + xy'$ → same as S in half adder

$B = x'y$

x	y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Full Subtractor

$D = x'y'z + x'yz' + xy'z' + xyz$ → same as S in full adder

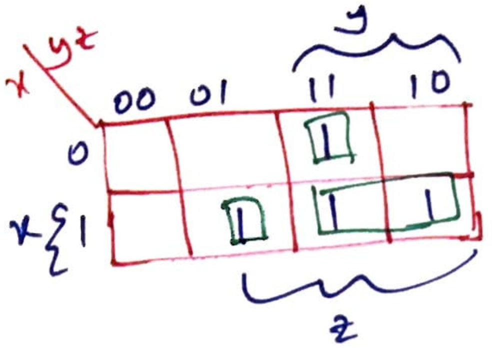
$B = x'y + x'z + yz$

↳ same as C in full adder except that x is complemented.

x	y	z	B	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

B

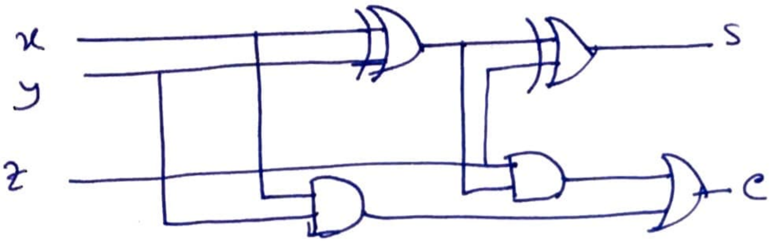
$x \backslash yz$	00	01	11	10
0		1	1	1
x { 1			1	



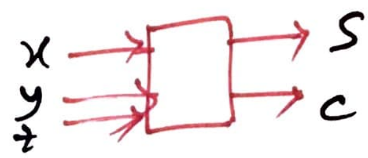
$S = x \oplus y \oplus z$

$$c = x'yz + xy'z + xy$$
$$= xy + (x'y + xy')z$$
$$= xy + (x \oplus y)z$$

Several ways to combine 1's for c.



logic diagram for FA



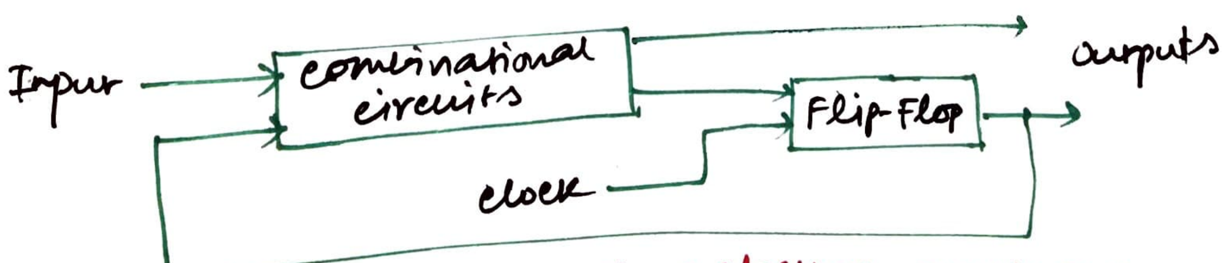
Block diagram for FA

Flip-Flops

- Digital Circuits
  1. Combinational circuit
     - output at any given time are entirely dependent on the inputs present at the time
  2. Sequential circuit
     - include storage elements (Flip-flops)

interconnection of flip-flop & $\boxed{\text{gates}}$ → combinational circuits.

- specified by a time sequence of external inputs, external outputs, and internal flip-flops binary states



Block diagram of a clocked synchronous sequential circuit.

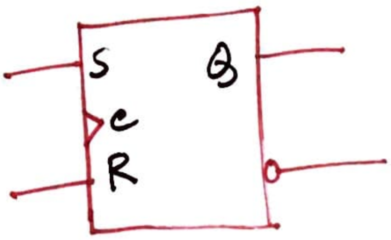
Clocked Synchronous sequential Circuits

- empty signals that effect the storage element only at a discrete instance of time.
- Synchronization is achieved by a timing device, called clock pulse generator that produces a train of clock pulses
- clock pulses are distributed througout the system in such a way that storage elements are affected only with the arrival of the synchronization pulse.
- Seldome manifest instability problems, and timing is easily broken down into independent discrete steps, each of which may be considered seperately.

Flip-Flops

- Storage elements employed in clocked sequential circuits.
- a binary cell capable of storing one bit of information.
- two outputs, one for the normal value and one for the complement value of the bit stored in it.
- maintains a binary state until directed by a clock pulse to switch states.

SR Flip-Flop (Set-Reset Flip-Flop)



Graphic Symbol

when clock signal changes from 0 to 1

S (→ set)	R (→ Reset)	Q (t+1)	
0	0	Q (t)	No change
0	1	0	clear to 0
1	0	1	set to 1
1	1	?	Indeterminate

characteristic table

may be 0 or 1 depending on internal delays that occur within the circuit.

3 inputs: S (for set), R (for reset), C (for clock)

Output: Q, sometime has complemented output

> in front of C : indicates dynamic input.

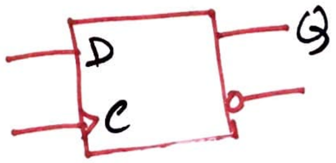
↓

flip-flop responds to a positive transition (from 0 to 1) of the input clock signal.

- no signal to the clock input C → no change of output of the circuit irrespective of the values S & R.
- clock signal changes from 0 to 1 → output of the circuit is affected according to the values of S & R.
- Binary State of Q output → Q(t) : present state
  Q(t+1) : next state
- SR-flipflop is seldom used in practice as indeterminate condition makes SR flip flop difficult to manage.

D Flip-Flop (Date Flip Flop)

- Slight modification of the SR flip flop.
- SR flipflop connected to D flipflop by inserting an inverter between S & R and assigning the D to the ~~signal~~ single input.



Graphic Symbol

when clock symbol changes from 0 to 1

D	Q(t+1)	
0	0	clear to 0
1	1	set to 1

Characteristic table

- characteristic equation: $\boxed{Q(t+1) = D}$

i.e., Q output of the D flipflop receives its value from the D input every time ~~value~~ that the clock signal goes through a transition from 0 to 1.

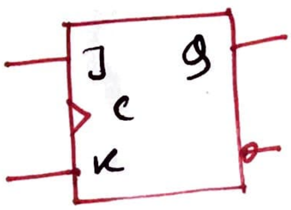
Advantage → only one input D (excluding c)

Disadvantage → No input condition exists that will leave the state of the D flip-flop unchanged.

→ No change condition $Q(t+1) = Q(t)$ can be achieved
- either by disabling the clock signal
- or by feeding the output back into the input so that clock pulses keep the state of the flip-flop unchanged.

JK Flip Flop

- A refinement of SR flip flop, ~~independent~~ indeterminate condition of SR flip flop is defined in JK flip flop.



Graphic symbol

Set → J, Reset → K

when clock signal changes from 0 to 1

J	K	Q(t+1)	
0	0	Q(t)	No change
0	1	0	clear to 0
1	0	1	set to 1
1	1	Q'(t)	complement

Characteristic table

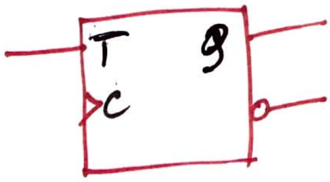
- J input equivalent to the S (set) input of SR flip flop.
- K input equivalent to the R (Reset) input of SR flip flop
- JK flip flop has a complement condition $\boxed{Q(t+1) = Q'(t)}$ when both J & K are equal to 1.

T Flip flop (Toggle flip flop)

characteristic eq.

$$\boxed{Q(t+1) = Q(t) \oplus T}$$

- obtained from JK flipflop when inputs J and K are connected to provide a single input designated by T.



Graphic symbol

clock transition from 0 to 1

T	Q(t+1)
0	Q(t)
1	Q'(t)

Characteristic table

- T=0 (J=K=0) — Clock trans does not change the state
- T=1 (J=K=1) Clock trans. complement the state

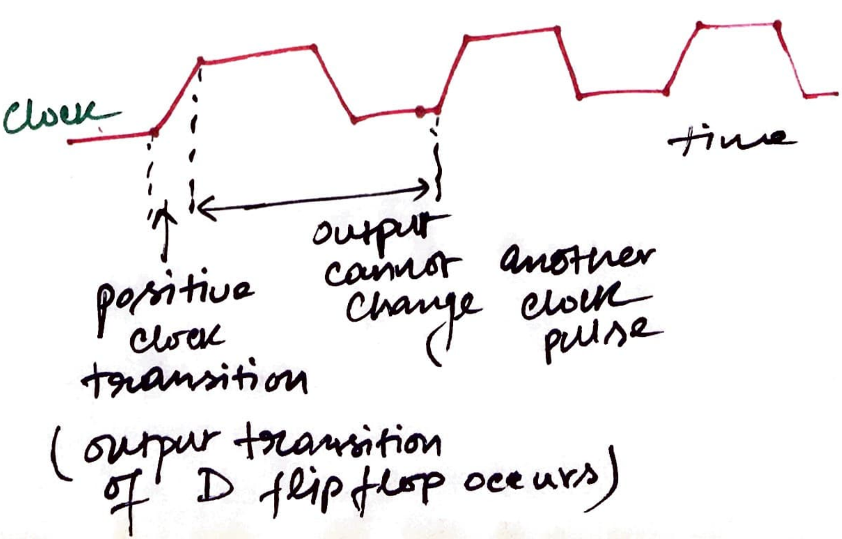
Edge Triggered Flip Flop

- Most common type of flip flop used to synchronise the state change during a clock pulse transition.
- Output transition occur at a specific level of the clock pulse.
- When pulse input level exceeds this threshold level, the inputs are locked out → flip flop is unresponsive to further changes in ~~the~~ inputs until the clock pulse returns to 0 and another clock pulse occurs.

transition
- → positive edge transition (flip flop causes a transition on the rising edge of the clock signal)
- → negative edge transition (flip-flop causes a transition on the falling edge of the clock signal)

Positive edge-triggered D flip-flop

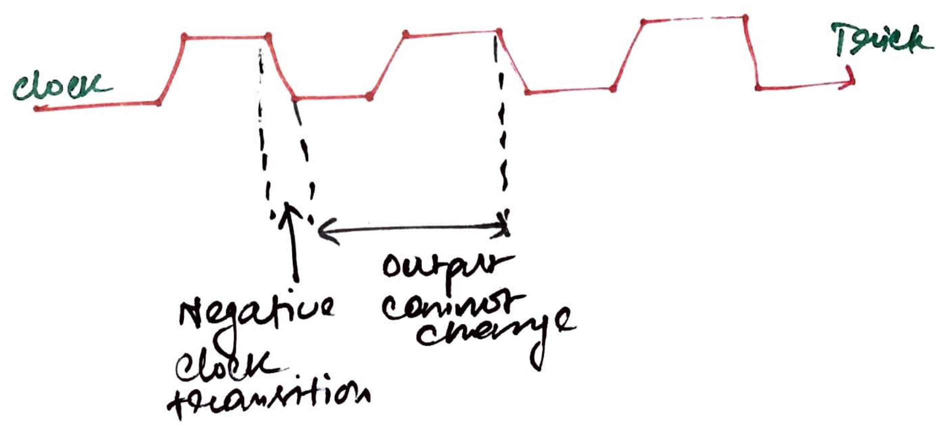
Graphic Symbol



- a value of the D input is transferred to the Q output when the clock makes a positive transition
- The output cannot change when the clock is in the level 1, level 0 or in a transition from the level 1 to level 0.
- effective positive clock transition ⟶ usually a very small fraction of the total period of the clock pulse.

includes
- → setup time (D input must remain at a const. value before the transition)
- → Hold time (D input must not change after the positive transition)

Negative edge triggered D flipflop



(flip flop responds to a transition from level 1 to level 0 of the clock signal)

(output transition of D flip flop)

Master-Slave Flip Flop

- Circuit consists of two flipflops
  - master, which responds to the +ve level of the clock
  - slave, which responds to the -ve level of the clock
- Output changes during the 1-to-0 transition of the clock signal.
- Flipflops sometime provide special input/terminal for setting or clearing the flipflop asynchronously → preset and clear input
  - → useful for bringing the flipflop to an initial state prior to its clocked operation
  - → affect the flipflop on a -ve level of the input signal without the need of clock pulse

Excitation Tables

- input, present state → find next state (characteristic table)
- present state, next state → find the flipflop input conditions that cause the required transition (excitation table)

SR FlipFlop

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

$Q(t) = Q(t+1) = 1$

when
- → no change: $\boxed{S=0}$, $R=0$
- → set to 1: $\boxed{S=1}$, $R=0$

$\boxed{S=X, R=0}$

$Q(t) = Q(t+1) = 0$

when
- → no change: $S=0$, $\boxed{R=0}$
- → clear to 0: $S=0$, $\boxed{R=1}$

$\boxed{S=0, R=X}$

D flipflop

Q(t)	Q(t+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

$$Q(t+1) = Q'(t)D + Q(t)D$$
$$= D$$

JK flipflop

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

$Q(t) = Q(t+1) = 0$ { if no change $J=0, K=0$; if clear to 0 $J=0, K=1$

$Q(t) = 0, Q(t+1) = 1$ { if $J=0, K=0$; if $J=1, K=1$

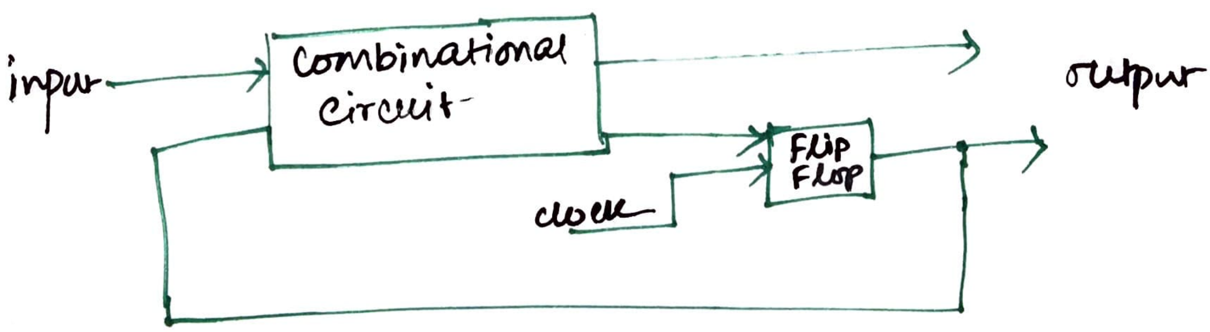
T flipflop

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

$$Q(t+1) = Q'(t)T + Q(t)T'$$
$$= Q(t) \oplus T$$

Sequential Circuits

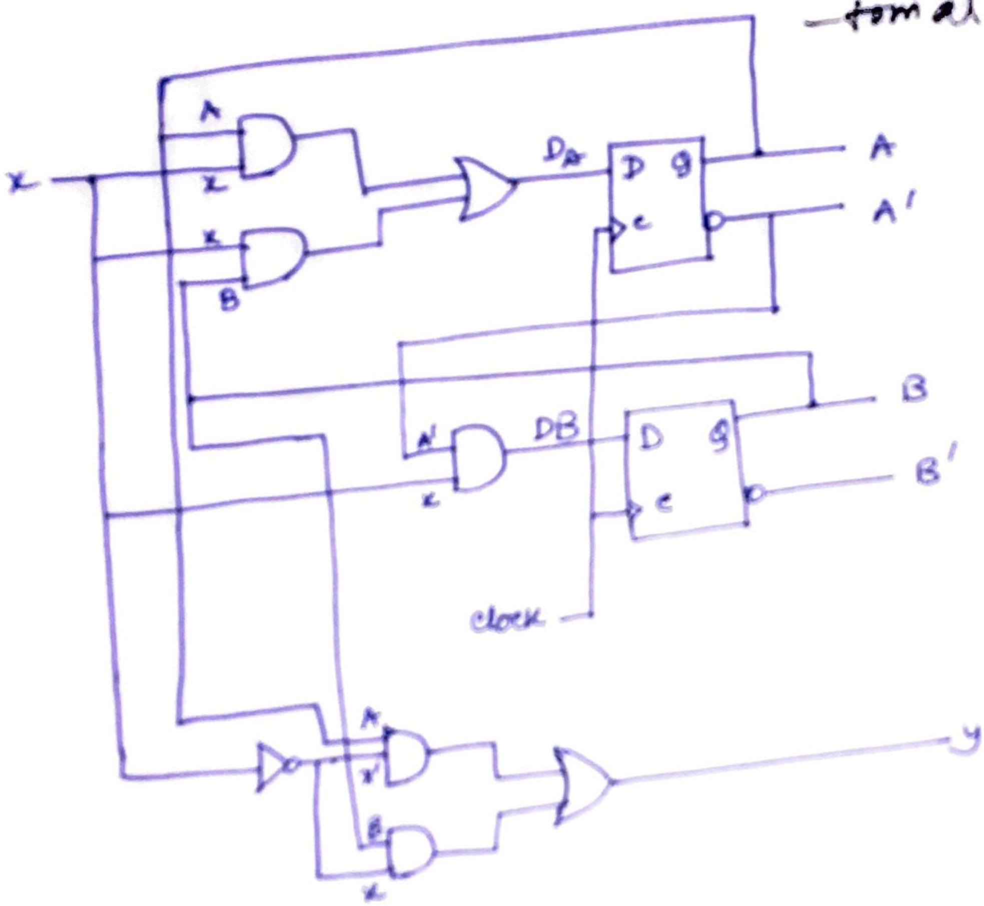
- external outputs are functions of both external inputs and present state of the flipflop.
- next state of flipflops is also functions of their present state & external inputs.



Example (Sequential Circuit)

- part of combinatorial circuit that generates the inputs to flip flops → expressed boolean expression called flip flop input equations

- Set of Boolean expressions → interconnection among the gates in the combinatorial circuit.



Input : x

Output : y

$D_A = Ax + Bx$
$D_B = A'x$ } flip flop input equations

$y = Ax' + Bx'$ → external output, a fun. of external input & the state of the flip flop.

State Table : $D_A = Ax + Bx$, $D_B = A'x$, $y = Ax' + Bx'$

present state		input	next state		output
A	B	x	A	B	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	0	0
1	1	0	0	0	1
1	1	1	1	0	0

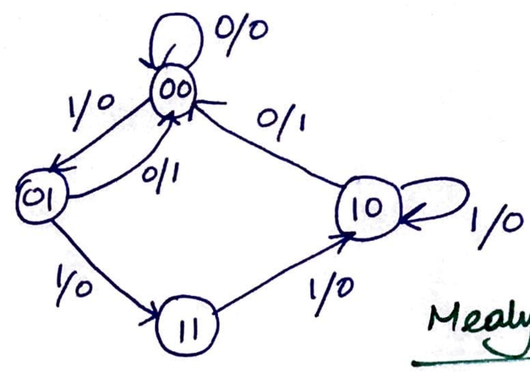
- present state → states of flipflops A & B at any given time t
- input → a value x for each possible present state.
- next state → States of flipflops A & B are clocked period later at time (t+1)
- output → a value of y for each present state and input condition.

State table for sequential circuit with m flipflops, n input variables & p output variables →

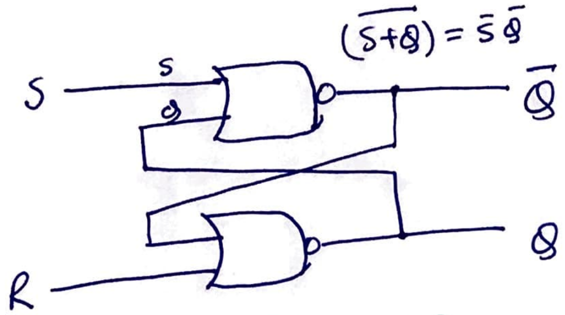
→ m columns for present states
n columns for inputs } 2^{m+n} binary combinations

→ m columns for next state
p columns for outputs } function of present state and inputs. derives from the circuit.

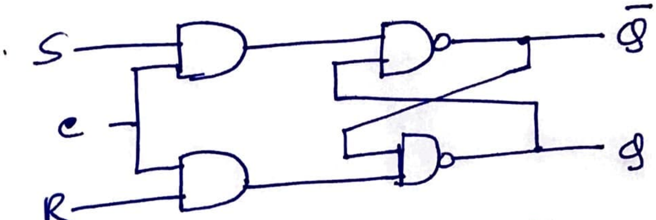
State Diagram :



Mealy machine

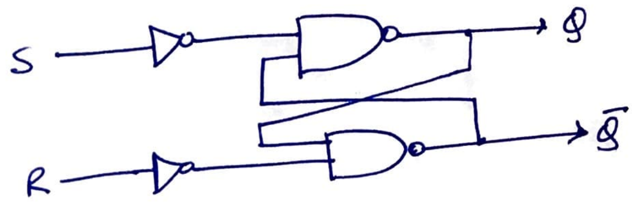


SR-FlipFlop-NOR latch



Clocked SR FlipFlop

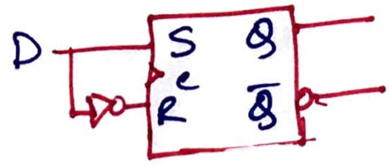
S	R	Q(t+1)
0	0	No change
0	1	0
1	0	1
1	1	undefined



SR-FlipFlop NAND latch

Present state			Next state
S	R	Q(t)	Q(t+1)
0	0	0	0 } no change
0	0	1	1
0	1	0	0 } clear to 0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	undefined
1	1	1	undefined

D FlipFlop as a Modified SR FlipFlop

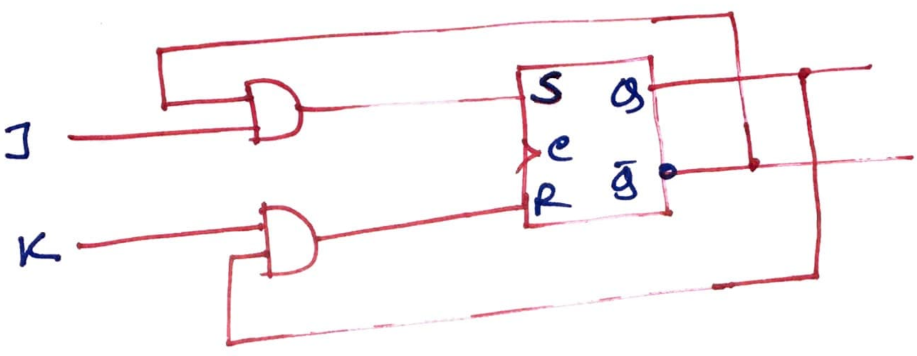


D	Q (t+1)
0	0
1	1

char. eq.

$$\boxed{Q(t+1) = D}$$

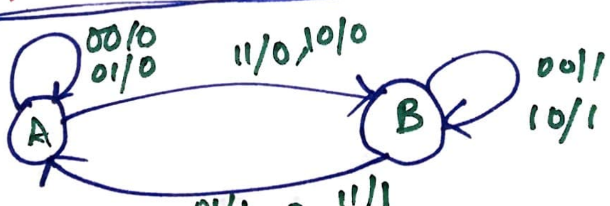
JK FlipFlop as a modified SR FlipFlop



Characteristic table

J	K	Q (t+1)	
0	0	Q (t)	No change
0	1	0	clear to 0
1	0	1	set to 1
1	1	Q'(t)	complement

JK FlipFlop as a Mealy machine



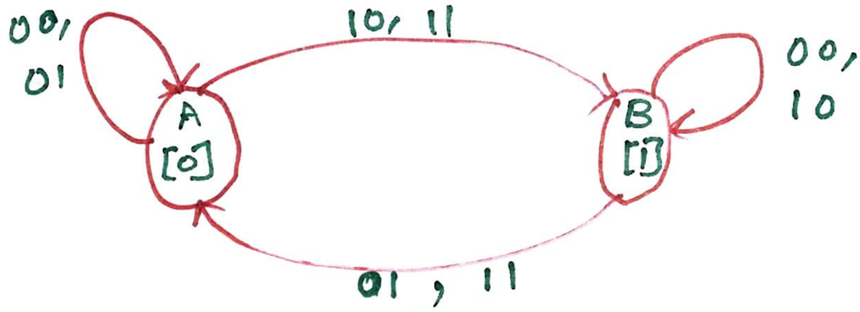
- output is associated with each arc
- output is a fcn of current state & its input

Truth table for JK FlipFlop

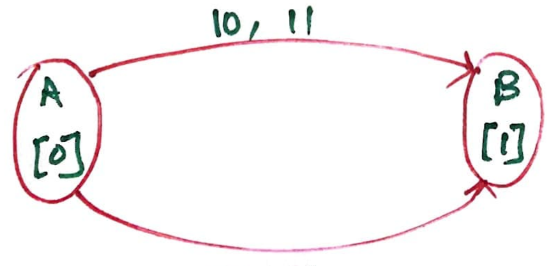
Present State			Next State	
J	K	Q(t)	Q(t+1)	
0	0	0	0	no change
0	0	1	1	
0	1	0	0	~~clr~~ clear too
0	1	1	0	
1	0	0	1	set to 1
1	0	1	1	
1	1	0	1	complement
1	1	1	0	

JK FlipFlop represented as a Moore Machine

- output is a fun of only current state
- output associated with Nodes / states

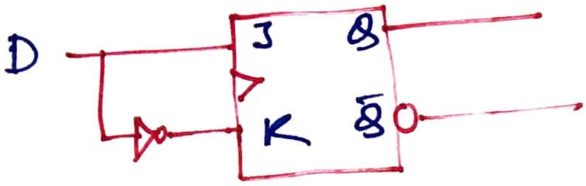


OR

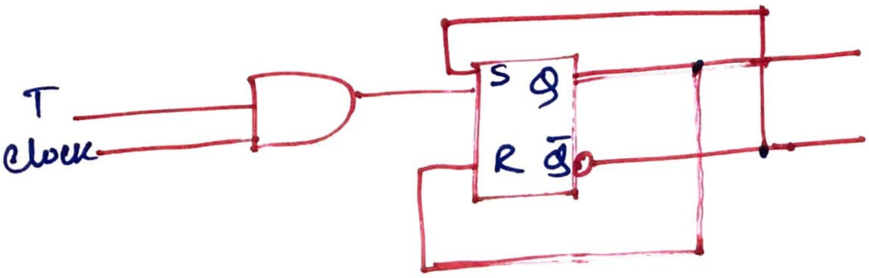


- ~~#~~ Dropping reflexive arcs as output of the machine changes only when state changes
- State does not change through a reflexive arc.

Transforming JK to D FlipFlop



T FlipFlop as a modified SR FlipFlop (Toggle)



char. eq.

$$\boxed{Q(t+1) = Q(t) \oplus T}$$

characteristic tables

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

D	Q(t+1)
0	0
1	1

$$\boxed{Q(t+1) = D}$$

J	K	Q (t+1)
0	0	Q (t)
0	1	0
1	0	1
1	1	Q' (t)

T	Q (t+1)	
0	Q (t)	→ no change
1	Q' (t)	complement

$$\boxed{Q(t+1) = Q(t) \oplus T}$$

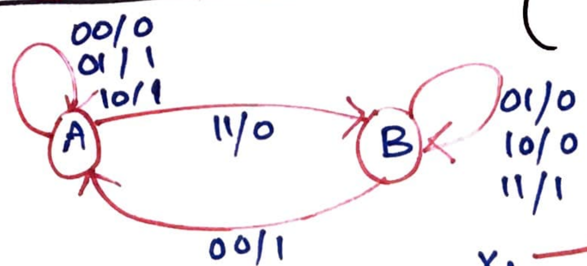
Sequential Circuits

Example (Social Binary Adder):

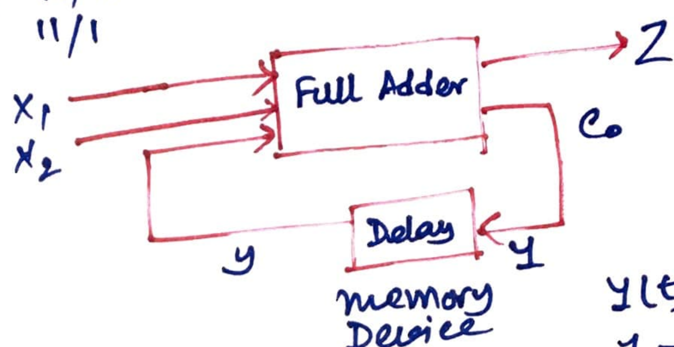
carry →	1	1				
	0	1	1	0	0	$= x_1$
	0	1	1	1	0	$= x_2$
	1	1	0	1	0	$= Z$

Mealy machine

(State A → carry 0
State B → carry 1)



State diagram of a Serial Adder



$y(t) = Y(t+1)$
$Y \rightarrow$ Delay
y = state variable
x_1, x_2 = primary input variable

State table:

Present State	Next State, Z			
	$x_1x_2=00$	$x_1x_2=01$	$x_1x_2=11$	$x_1x_2=10$
A	A, 0	A, 1	B, 0	A, 1
B	A, 1	B, 0	B, 1	B, 0

State Assignment

(Present state) carry → y		(Next State) Y				(output) Z			
		$x_1x_2=00$	$x_1x_2=01$	$x_1x_2=11$	$x_1x_2=10$	00	01	10	11
A	0	0	0	1	0	0	1	0	1
B	1	0	1	1	1	1	0	1	0

Z-map

$y \backslash x_1x_2$	00	01	11	10
0		1		1
1	1		1	

Y-map

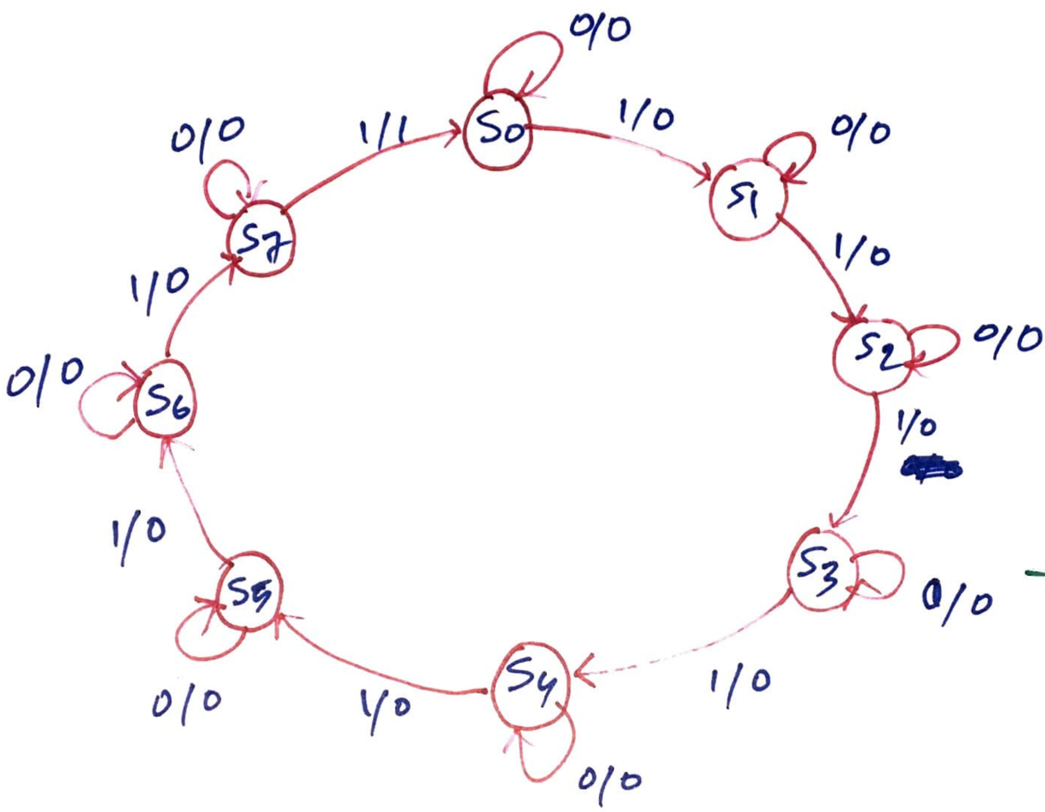
$y \backslash x_1x_2$	00	01	11	10
0			1	
1		1	1	1

$Y = x_1x_2 + yx_2 + yx_1$

logic eq.: $Z = x_1' x_2 y' + x_1 x_2' y' + x_1' x_2' y + x_1 x_2 y$
$= x_1 \oplus x_2 \oplus y$

Binary Counter

- one input ; one output
- counts binary no system upto 7 on input 1
- produces output value 1 for every 8 input 1 value
- if a count of 7 reached with next input 1 then reset counter to its initial state, i.e., to a count of 0.



Mealy machine

Present State $y_3\ y_2\ y_1$	Next State $x=0$	Next State $x=1$	Z $x=0$	Z $x=1$
0 0 0	000	001	0	0
0 0 1	001	010	0	0
0 1 0	010	011	0	0
0 1 1	011	100	0	0
1 0 0	100	101	0	0
1 0 1	101	110	0	0
1 1 0	110	111	0	0
1 1 1	111	000	0	1

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

Excitation table for T FlipFlop

Excitation table for T Flip Flop for Binary Counter

$y_3 y_2 y_1$	$T_3 T_2 T_1$ ($x=0$)	$T_3 T_2 T_1$ ($x=1$)
0 0 0	0 0 0	0 0 1
0 0 1	0 0 0	0 1 1
0 1 0	0 0 0	0 0 1
0 1 1	0 0 0	1 1 1
1 0 0	0 0 0	0 0 1
1 0 1	0 0 0	0 1 1
1 1 0	0 0 0	0 0 1
1 1 0	0 0 0	1 1 1

Schematic diagram of a modulo 8 binary counter with T FlipFlop:

T_1-map

$T_1 = x$

$y_3 y_2 y_1$ \ x	0	1
000		
001		1
010		
011		1
100		
101		1
110		
111		1

T_2-map

$y_3 y_2 y_1$ \ x	0	1
000		
001		
010		
011		1
100		
101		
110		
111		1

T_3-map

$$T_2 = y_3 y_2 y_1 x + y_3' y_2 y_1 x + y_3 y_2' y_1 x + y_3 y_2 y_1 x$$

$$= y_3' y_1 x + y_3 y_1 x = y_1 x$$

$$T_3 = y_3' y_2 y_1 x + y_3 y_2 y_1 x = y_2 y_1 x$$

z-map $\quad z = y_3 y_2 y_1 x$

logic equations :

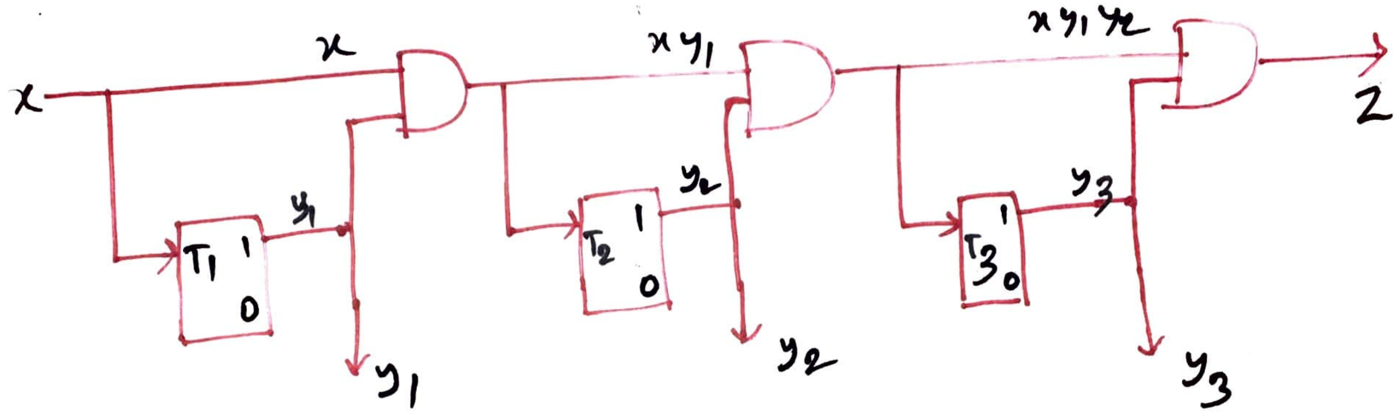
$T_1 = x$

$T_2 = x y_1$

$T_3 = x y_1 y_2$

$Z = x y_1 y_2 y_3$

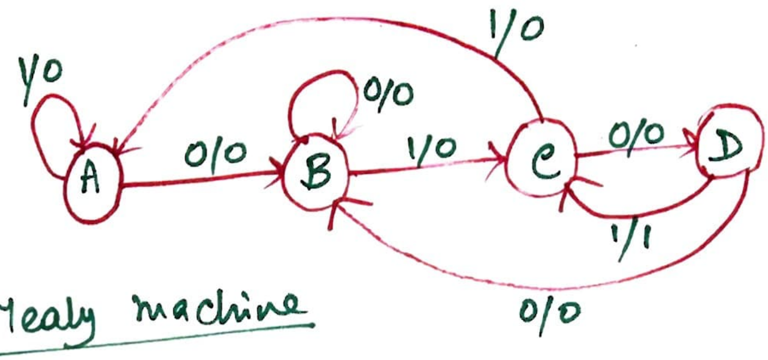
logic diagram :



Using SR Flip Flop :

- → Excitation table for SR Flip Flop
- → Excitation table for SR FlipFlop in modulo 8 binary ~~cou~~ counter
- → logic eq.
- → logic diagram.

The Sequence Detector



Mealy machine

Present State	Next State, output	
	$x=0$	$x=1$
A	B, 0	A, 0
B	B, 0	C, 0
C	D, 0	A, 0
D	B, 0	C, 1

A State Diagram

(Present State) $y_1 y_2$		(Next State) $Y_1 Y_2$ $x=0$	$x=1$	(Output) z $x=0$	$x=1$
A	00	01	00	0	0
B	01	01	11	0	0
C	10	10	00	0	0
D	11	01	11	0	1

Z-map

y_1y_2 \ x	0	1
00		
01		
11		
10		1

$$Z = y_1 y_2' x$$

Y_1-map

y_1y_2 \ x	0	1
00		
01		1
11	1	
10		1

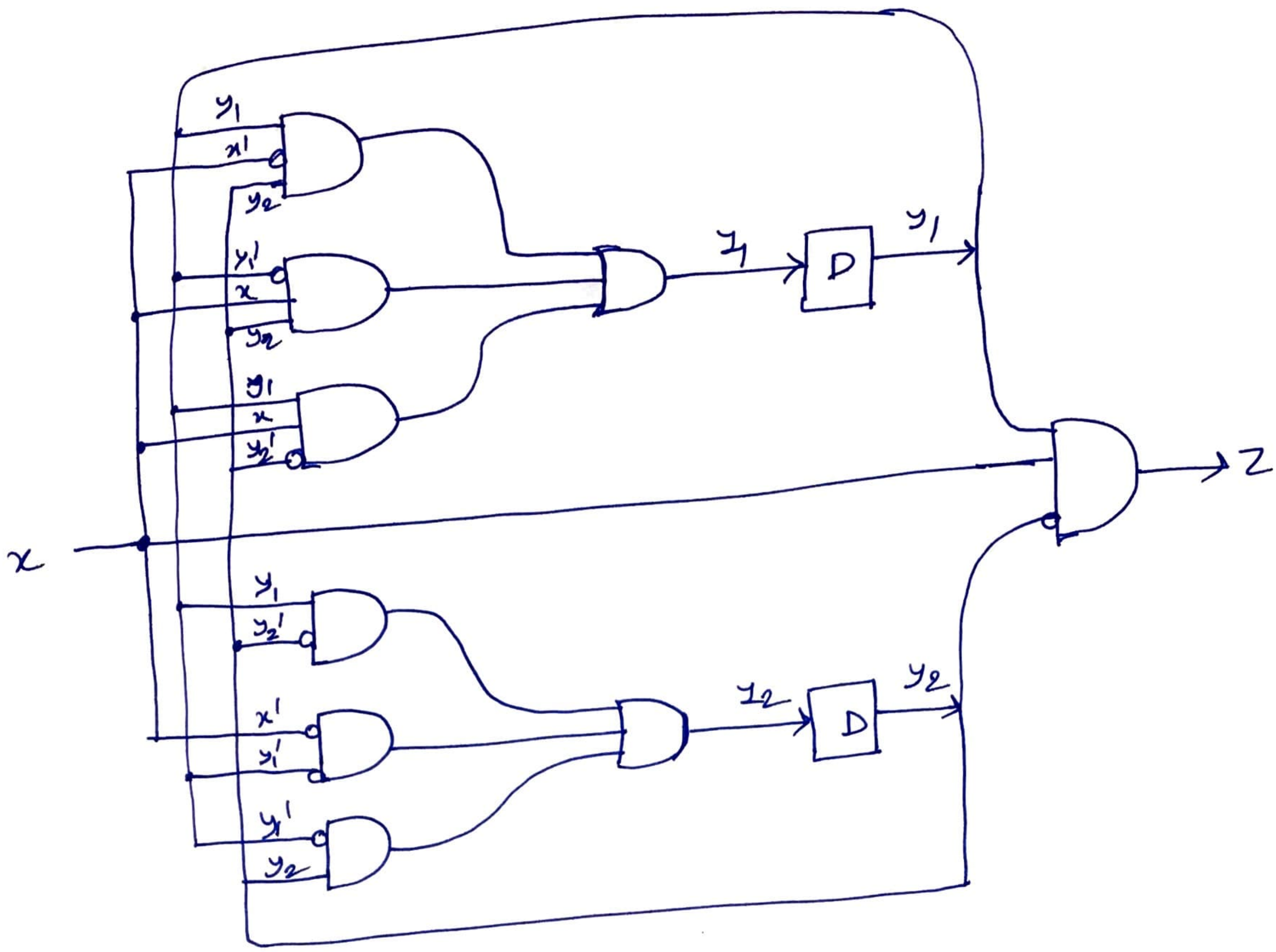
$$Y_1 = y_1 y_2 x' + y_1' y_2 x + y_1 y_2' x$$

Y_2-map

y_1y_2 \ x	0	1
00	1	
01	1	1
11		
10	1	1

$$Y_2 = y_1' y_2 + y_1 y_2' + x' y_1'$$

Logic Diagram



Another State Diagram

	$y_1 y_2$	$Y_1 Y_2$ ($x=0$)	$Y_1 Y_2$ ($x=1$)	Z ($x=0$)	Z ($x=1$)
A	00	01	00	0	0
B	01	01	10	0	0
C	10	11	00	0	0
D	11	01	10	0	1

Z-map

$y_1y_2 \backslash x$	0	1
00		
01		
11		1
10		

$Z = y_1 y_2 x$

Y_1-map

$y_1y_2 \backslash x$	0	1
00		
01		1
11		1
10	1	

$Y_1 = y_2 x + y_1 y_2' x'$

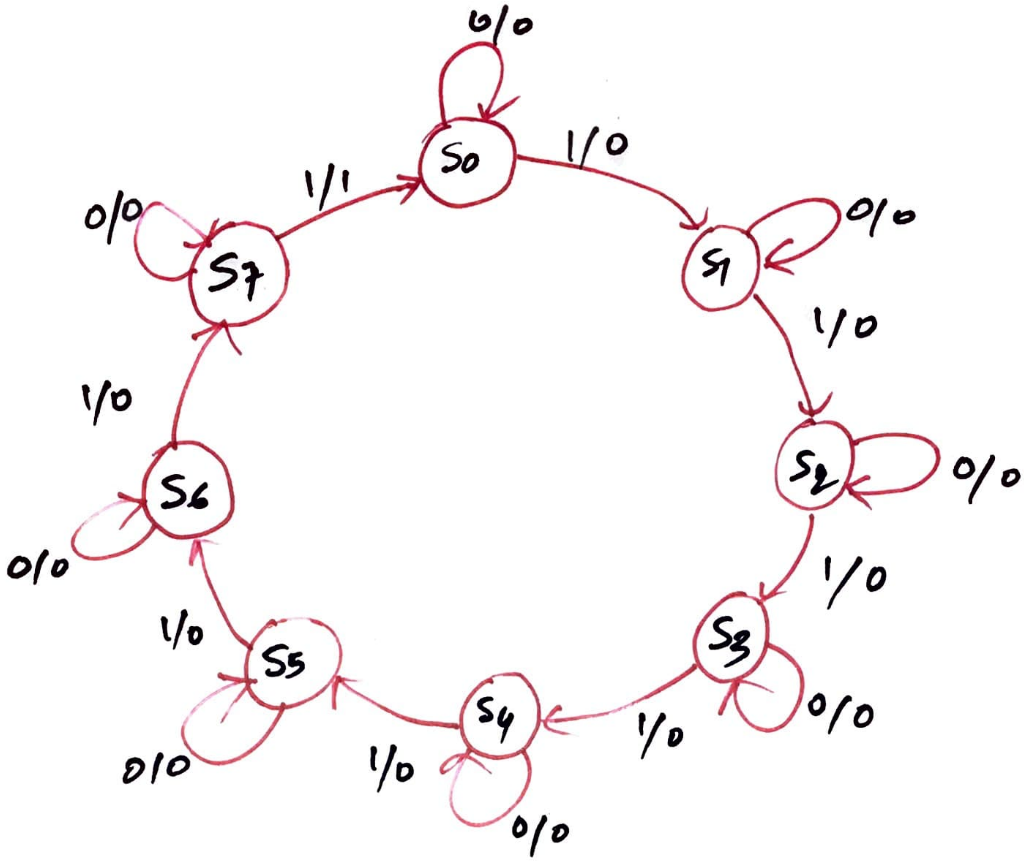
Y_2-map

$y_1y_2 \backslash x$	0	1
00	1	
01	1	
11	1	
10	1	

$Y_2 = x'$

Modulo-8 binary Counter

Implementing with SR flip flop



Transition Table:

	Present State $y_3\,y_2\,y_1$	Next State $x=0$	Next State $x=1$	Z $x=0$	Z $x=1$
S_0	0 0 0	0 0 0	0 0 1	0	0
S_1	0 0 1	0 0 1	0 1 0	0	0
S_2	0 1 0	0 1 0	0 1 1	0	0
S_3	0 1 1	0 1 1	1 0 0	0	0
S_4	1 0 0	1 0 0	1 0 1	0	0
S_5	1 0 1	1 0 1	1 1 0	0	0
S_6	1 1 0	1 1 0	1 1 1	0	0
S_7	1 1 1	1 1 1	0 0 0	0	1

Excitation Table for SR flip flops for the transition table in the previous page:

$y_3 y_2 y_1$	$x=0$: S_3R_3	S_2R_2	S_1R_1	$x=1$: S_3R_3	S_2R_2	S_1R_1
0 0 0	0 –	0 –	0 –	0 –	0 –	1 0
0 0 1	0 –	0 –	– 0	0 –	1 0	0 1
0 1 0	0 –	– 0	0 –	0 –	– 0	1 0
0 1 1	0 –	– 0	– 0	1 0	0 1	0 1
1 0 0	– 0	0 –	0 –	– 0	0 –	1 0
1 0 1	– 0	0 –	– 0	– 0	1 0	0 1
1 1 0	– 0	– 0	0 –	– 0	– 0	1 0
1 1 1	– 0	– 0	– 0	0 1	0 1	0 1

$x=0$, $S_0 = 000$ goes to $S_0 = 000$

$y_1 = 0$, does not change $\rightarrow$ $S_1R_1 = 0-$

$y_2 = 0$, does not change $\rightarrow$ $S_2R_2 = 0-$

$y_3 = 0$, does not change $\rightarrow$ $S_3R_3 = 0-$

$x=1$, $S_0 = 000$ goes to $S_0 = 001$

y_1 changes from 0 to 1 $\rightarrow$ $S_1R_1 = 10$

$y_2 = 0$, does not change $\rightarrow$ $S_2R_2 = 0-$

$y_3 = 0$, does not change $\rightarrow$ $S_3R_3 = 0-$

change in y from	to	S	R
0	0	0	–
0	1	1	0
1	0	0	1
1	1	–	0

S_1-map:

$$S_1 = x(y_3'y_2'y_1' + y_3'y_2y_1' + y_3y_2'y_1' + y_3y_2y_1')$$
$$= x(y_3'y_1' + y_3y_1') = xy_1'$$

R_1-map:

$$R_1 = x(y_3'y_1 + y_3y_1)$$
$$= xy_1$$

S_2-map:

$$S_2 = x(y_3'y_2'y_1 + y_3y_2'y_1)$$
$$= xy_2'y_1$$

R_2-map:

$$R_2 = x(y_3'y_2y_1 + y_3y_2y_1)$$
$$= xy_2y_1$$

S_3-map:

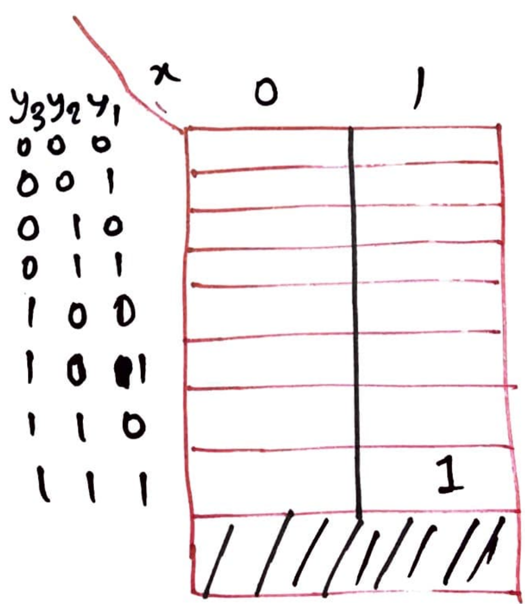
$$S_3 = xy_3'y_2y_1$$

R_3-map:

$$R_3 = xy_3y_2y_1$$

Z-map:

$$Z = xy_1y_2y_3$$

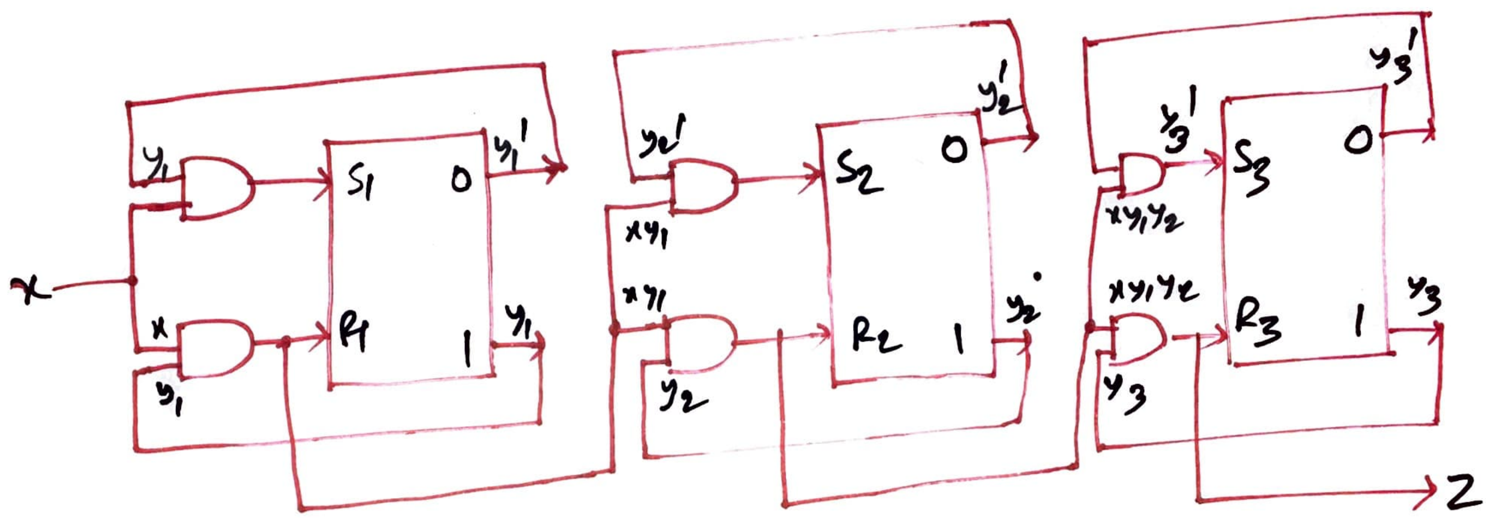


Logic Equation :

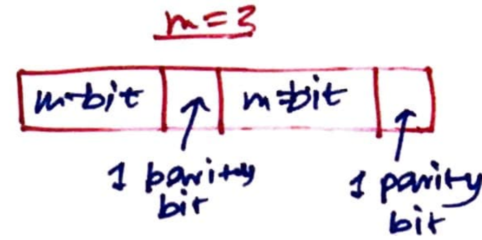
$S_1 = xy_1'$, $S_2 = xy_1y_2'$, $S_3 = xy_1y_2y_3'$

$R_1 = xy_1$, $R_2 = xy_1y_2$, $R_3 = xy_1y_2y_3$

$Z = xy_1y_2y_3$

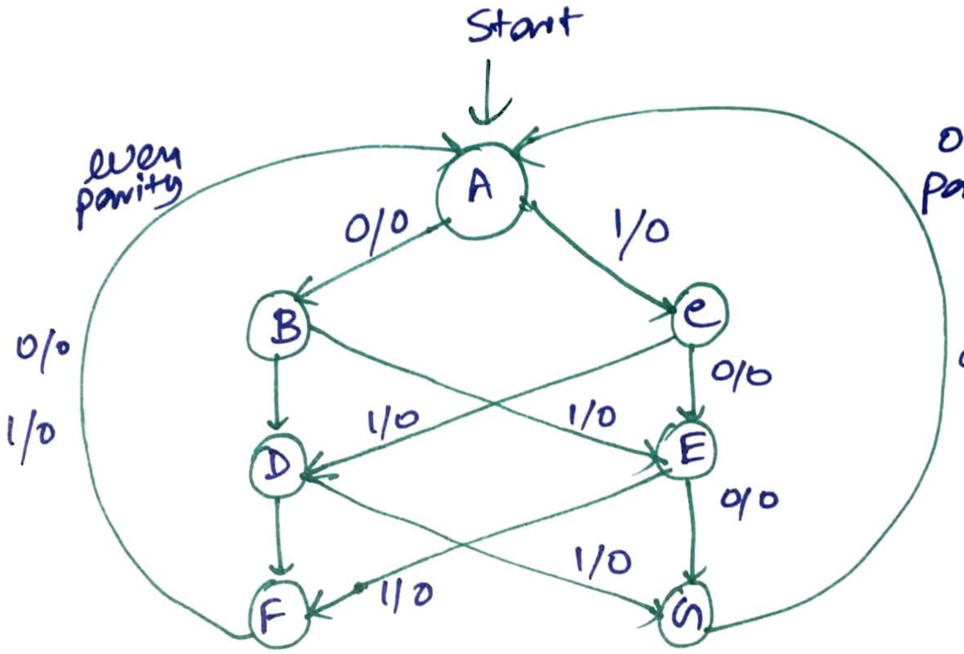


A Serial parity-bit generator :



- 3 bit message
- even parity used
- parity bit 1 to be inserted iff the number of 1's in the preceeding string of three symbols is odd.

State diagram



→ B, D, F → correspond to even no. of 1's out of one, two & three incoming input symbols resp.

→ C, E, G → that to odd no. of 1's

→ from either F or G state, the machine goes to state A, regardless of the input symbol.

7 states → 3 state variables needed
→ one of the 8 states will not be assigned, considered as don't care

A State Assignment table :

	Present State $y_1 y_2 y_3$	Next State $x=0$	Next State $x=1$	Z $x=0$	Z $x=1$
A	0 0 0	B	C	0	0
B	0 1 0	D	E	0	0
C	0 1 1	E	D	0	0
D	1 1 0	F	G	0	0
E	1 1 1	G	F	0	0
F	1 0 0	A	A	0	0
G	1 0 1	A	A	1	1

- Use JK flip flop as memory element

logic equations :

$J_1 = y_2$, $J_2 = y_1'$, $J_3 = x y_1' + x y_2$

$K_1 = y_2'$, $K_2 = y_1$, $K_3 = y_2' + x$

$Z = y_2' y_3$.