Q2 - 3-bit odd parity generator.

For odd parity, the parity bit 1 is inserted if and only if the number of 1's in the preceding string of three symbols are even.

We define 7 states:

000 or A - The sequence has not yet started.

001 or B - sequence detects first zero.

010 or C - sequence detects second zero/second one.

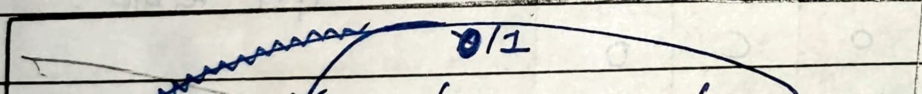
011 or D - sequence detects ~~third zero~~ even one's in three bits.

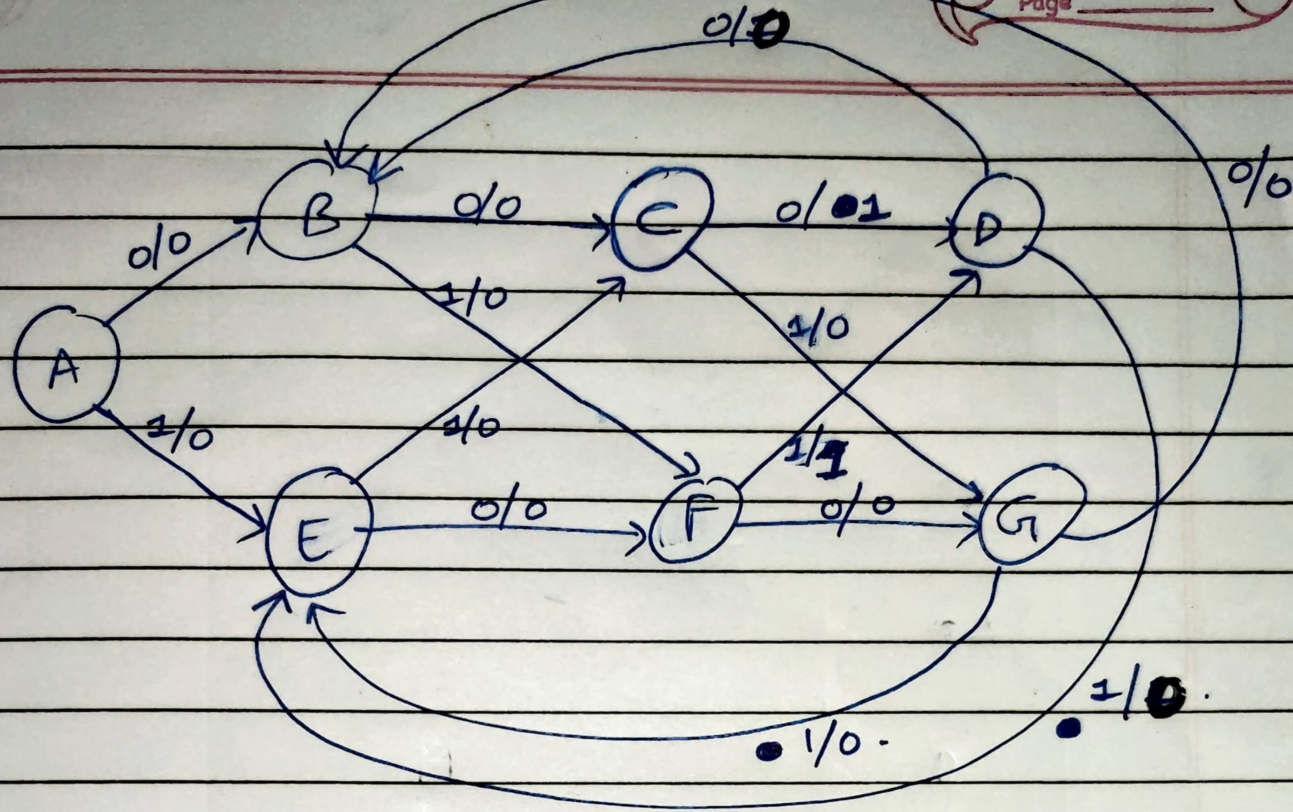
100 or E - sequence detects first one.

101 or F - sequence detects 1 one and 1 zero.

110 or G - sequence detects odd one's in three bits

A → ? ? ?	B → 0 ? ?	C → 0 0 ? / 1 1 ?	D → 000, 011, 110, 101
	E → 1 ? ?	F → 0 1 ? / 1 0 ?	G → 111, 100, 010, 001
0 bits read	1 bit read.	2 bits read	three bits read, return to A.





STATE DIAGRAM.

output is decided at C and F.

we return back to B or E after reading first bit of new input at D or G.

Present State \ Input	(NS, O/P) 0	(NS, O/P) 1
A	(B, 0)	(E, 0)
B	(C, 0)	(F, 0)
E	(F, 0)	(C, 0)
C	(D, 1)	(G, 0)
F	(G, 0)	(D, 1)
D	(B, 0)	(E, 0)
G	(B, 0)	(E, 0)

State table for odd parity generator.

Present State \ Input	0		1	
	Next State	output	Next State	output
A	B	0	E	0
B	C	0	F	0
E	F	0	C	0
C	D	1	G	0
F	G	0	D	1
D	B	0	E	0
G	B	0	E	0

Transition and Output table.

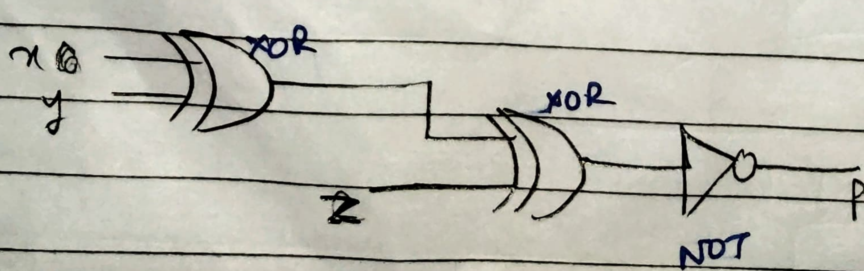
If we consider the three input bits as x, y, z

K-map.

z \ xy	00	01	11	10
0	1	0	1	0
1	0	1	0	1

$P = \overline{x \oplus y \oplus z}$

(XNOR)



Excitation table:

Present State	Next State	Input	Output
A	B	0	0
A	E	1	0
B	C	0	0
B	F	1	0
C	D	0	1
C	G	1	0
D	B	0	0
D	E	1	0
E	F	0	0
E	C	1	0
F	G	0	0
F	D	1	1
G	B	0	0
G	E	1	0

Logic: If input is 1 or 0 at state D, then p output is 1
(where D = 1 if currently at state D)

~~p → D~~

~~otherwise $\boxed{p=0}$~~

~~$A = D + G$.~~

$B = A \cdot \overline{in} + (D+G)\overline{in}$

$E = A \cdot in + (D+G)\, in$

$C = B \cdot \overline{in} + E \cdot in$

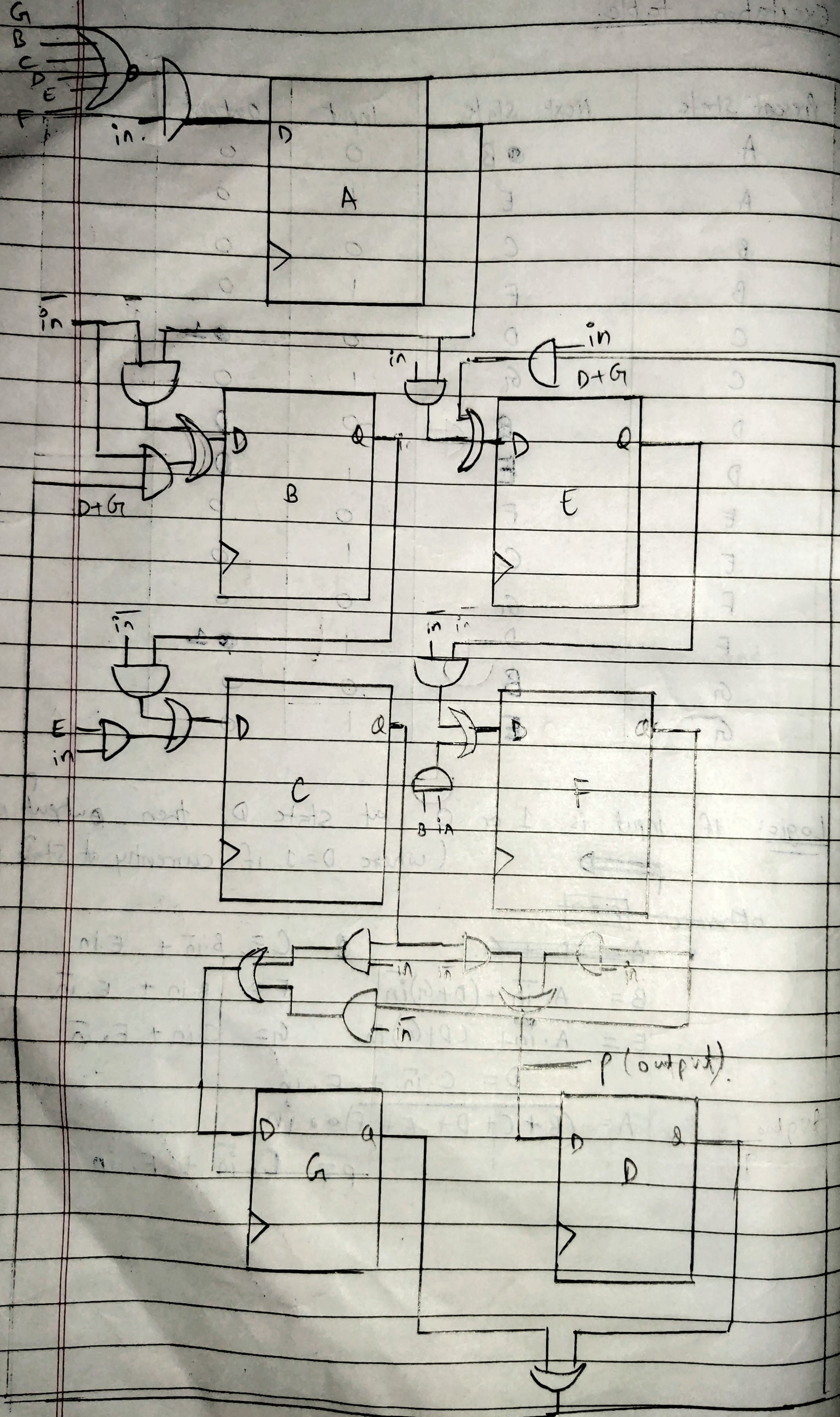
$F = B \cdot in + E \cdot \overline{in}$

$G = C \cdot in + F \cdot \overline{in}$

$D = C \cdot \overline{in} + F \cdot in$

$A = \overline{(B + C + D + E + F)} \cdot in$

$p = C \cdot \overline{in} + F \cdot in$


G
B
C
D
E
F
in.
D
A
in
in
D+G
Q
B
D+G
D
Q
E
in
in
D
Q
C
E
in
D
F
B
in
in
in
in
in
p (output).
D
Q
G
D
Q
D