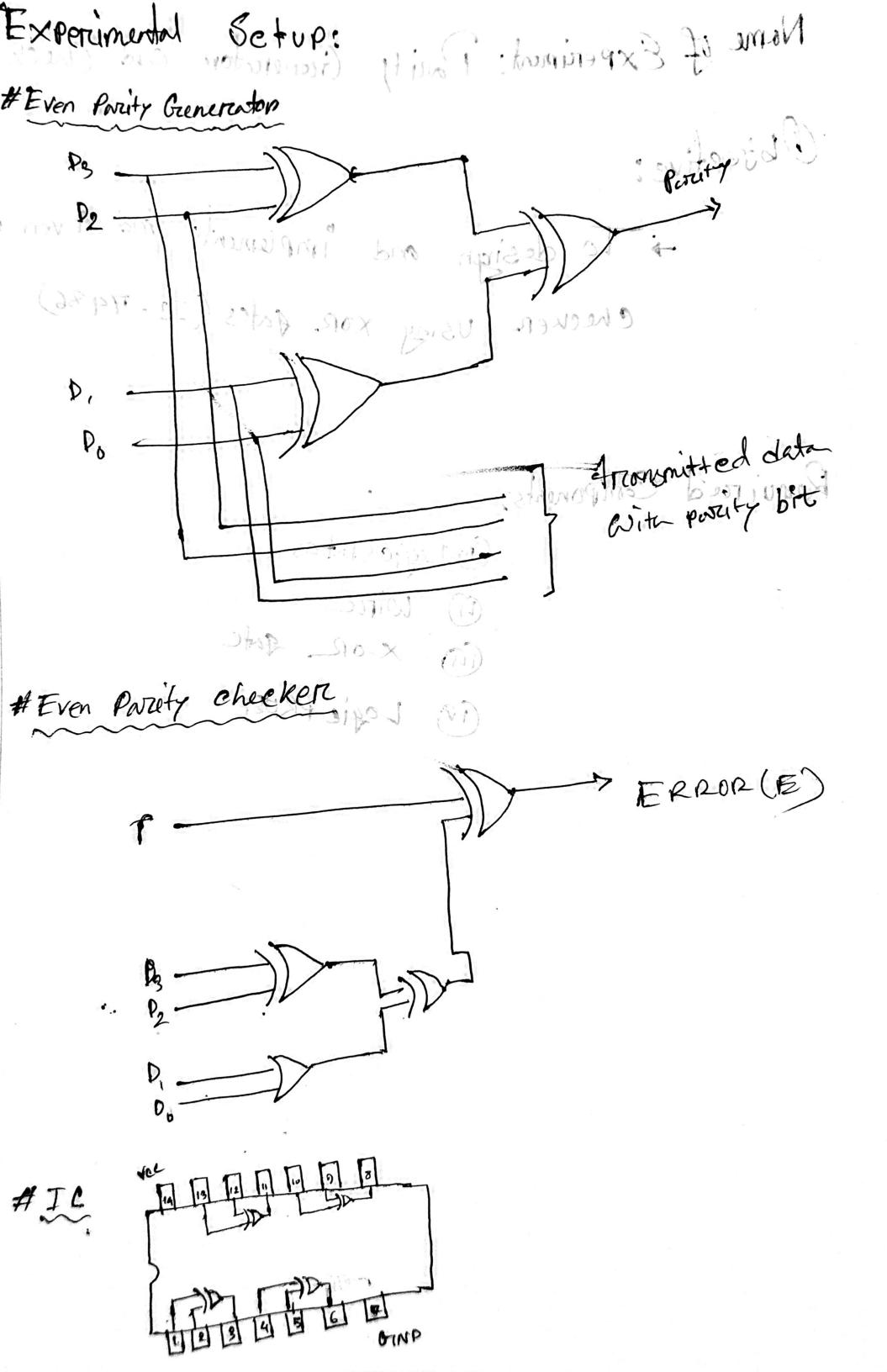
Parcity Generator And Checker

Si has 3 ahorina 20101402

Section: 11

Name of Experiment:	Parity	Granerator and Cheeker
Objective:	gen and	implement and Even Povity
checken	Using	implement and Even Portity KOR Antes (IC-7486)
Required Components	(I) (oirre x-or gute Logie PROBE
	The same of the sa	



Gunerator: I have take 4 logic states, denoted as Po, P1, D2, D3. Then Do R D1 is connected to a XOR gate and Dolly is Connected to another XOR gate. The both of the output is connected to onothe XOR gate, which is Parity bit.

Even Parity Checker: Herre Houre legicstates have Connected with two unique XOP Ante. Po, Pr. Pr. Pr. wa those. The two output is connected to mother XOR gate Where P was and mitial input. Lene 0=1+0=,000

We get E. 0=0+0=00000

Results is Tagolented form:

From Even priory Generation

1 1	- \ b. \ 6	Parity !	transmitted data
$\begin{bmatrix} D_0 \end{bmatrix} D_2 \begin{bmatrix} V_1 \end{bmatrix}$	0	The Control	10111
1 1	1.67		01001
0		6	00000
1 0	0 0	0	10 100
0 0	2 0	1 1	10 100
0 1 1			

01111 P 9Not $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0' D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0 + 0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0$ $D_0 \oplus D_1 = D_0' D_1 + D_1 D_0 = 0$ $D_0 \oplus D_0 = D_0' D_1 + D_0' D_0 = 0$ $D_0 \oplus D_0 = D_0' D_0 = 0$ $D_0 \oplus$ Joseph is consisted in some consisted in consister moder xor forte, which is & Prairy bit. $D_0 \oplus D_1 = D_0 \times D_3 = D_2 \times D_3 \times D_3 = 0 + 1 = 1$ in two output is = I & It e gote Whorze P was mittient in $D_0 \oplus D_1 = 0 + 1 = 0$ 0000 e tet E. D2 (D3 = 0+0= 6 : 000 = 0 :mod botton: 0 = 000: com Even prisery conserver 0100. Do (f) Di = 9+1 = 1 1000 D29 D3 = 0+0 = 0 1101 .. 100=1 000

Parity Checken:

D	Do	02	D.	Do	Gunerated	Ermon	OUTPUT
<u> </u>	1	0	1	0	0	0	0
	1			0		0	0
1	y			1	0		The second secon
1	1		4 2		,		A resolution of the second of
- 1	0	0	0	6	0	1	

Discussion: From the privity checker, we con notice that the colevlation is correct because the executation of generated parity and given parity, the error and the output from the circuit is similar.

From this experiment, we can learn about Party bit generation using logic godes and con-

implement.