EXPERIMENT No-8

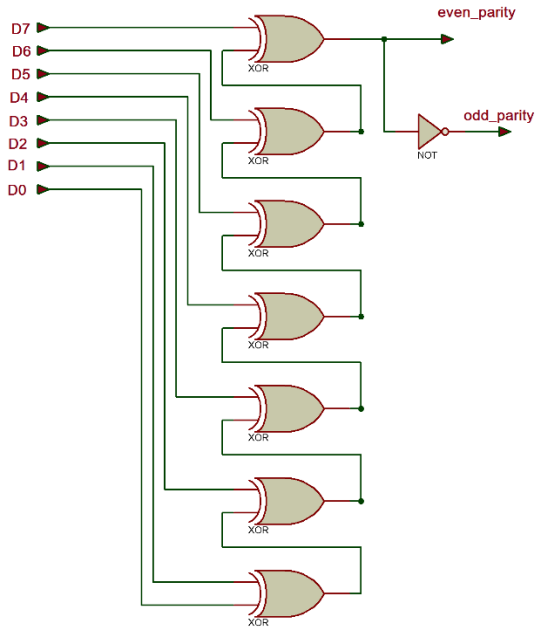
Aim: To Design an 8-bit parity generator and checker circuits using VHDL

Description:

A parity bit is an extra bit in any binary message to make the total number of 1's either odd or even. We need to add the parity bit to a signal, done by the Parity generator. This parity inclusive binary message then transmits from transmitter to receiver end.

The Parity Checker matches the number of 1's at the receiver's end with that of the transmitter's end to check for errors. If there is a change in the number of 1s at the receiving end, then that detects the presence of an error.

The 8-bit parity generator



Truth table

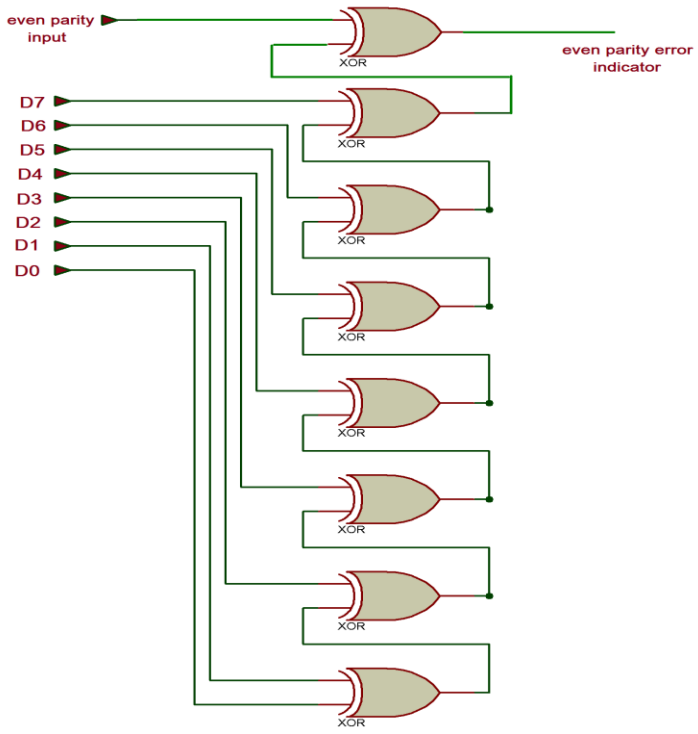
D7	D6	D5	D4	D3	D2	D1	D0	Even_parity	Odd_parity
1	0	1	1	0	0	1	0	0	1
1	1	0	0	1	0	0	0	1	0
1	1	1	1	1	0	1	1	1	0
1	0	1	1	1	1	1	0	0	1
0	0	1	0	1	0	1	0	1	0
0	1	1	1	0	1	0	1	1	0
0	1	0	1	0	0	1	1	0	1

VHDL Program:

```
library ieee;
use ieee.std_logic_1164.all;
entity ParityGenerator is
      port( data: in bit_vector(7 downto 0);
                  even_parity ,odd_parity: out bit);
end ParityGenerator;
architecture parity_generator of ParityGenerator is
signal temp : bit_vector (5 downto 0);
  begin
    temp(0)<=data(0) xor data(1);
    temp(1)<=temp(0) xor data(2);
    temp(2)<=temp(1) xor data(3);
    temp(3)<=temp(2) xor data(4);
    temp(4)<=temp(3) xor data(5);
    temp(5)<=temp(4) xor data(6);
    even_parity <= temp(5) xor data(7);
    odd_parity <= not(temp(5) xor data(7));
end parity_generator;
```

Output:

The 8-bit parity checker



Truth table

D7	D6	D5	D4	D3	D2	D1	D0	Parity input	Error
1	0	1	1	0	0	1	0	1	1
1	1	0	0	1	0	0	0	1	0
1	0	1	1	1	0	1	1	1	1
1	0	1	1	1	1	1	0	0	0
0	0	1	0	1	0	1	0	1	0
0	1	1	1	0	1	0	1	0	1
0	1	0	1	0	0	1	1	1	1

Note: here not all of the 256 combinations of the D0-D7 are displayed. Just a few are taken as examples.

VHDL program:

```
library ieee;
use ieee.std_logic_1164.all;
entity ParityChecker is
      port( data: in bit_vector(7 downto 0);
                  p: in bit; e: out bit);
end ParityChecker;
architecture parity_checker of ParityChecker  is
```

```
signal temp : bit_vector(6 downto 0);
  begin
    temp(0)<=data(0) xor data(1);
    temp(1)<=temp(0) xor data(2);
    temp(2)<=temp(1) xor data(3);
    temp(3)<=temp(2) xor data(4);
    temp(4)<=temp(3) xor data(5);
    temp(5)<=temp(4) xor data(6);
    temp(6) <= temp(5) xor data(7);
    e <= p xor temp(6);
end parity_checker;
```

Output: