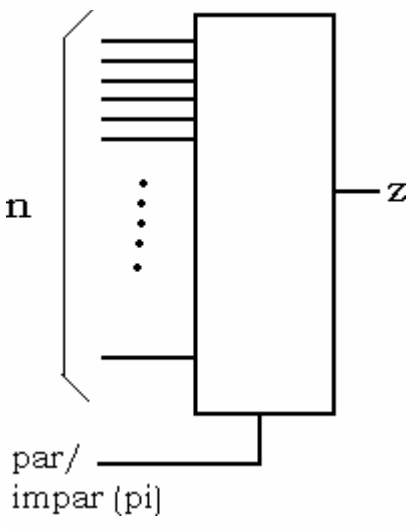
CIRCUITO DETECTOR DE PARIDAD CONFIGURABLE.

Usaremos el siguiente esquema tentativo




```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity problema3_paridad is
     generic (n: positive);
    Port (entrada: in std_logic_vector ((n-1) downto 0);
      pi:in bit;
      z:out bit);
end problema3_paridad;
architecture Behavioral of problema3_paridad is
begin
process (entrada,pi)
variable cuenta,residuo: integer :=0;
begin
     paso1:     for i in 0 to n-1 loop
                     if entrada(i)='1' then
                     cuenta:=cuenta+1;
                     end if;
                     end loop paso1;
     --para chequear si es par o impar
     residuo:=cuenta mod 2;
     if pi='1' then
          if residuo=0 then z<='1';
          else z<='0';
          end if;
     else
          if residuo=0 then z<='0';
          else z<='1';
          end if;
```

```
     end if;
end process;
end Behavioral;
```

Para probar esto creamos el siguiente archivo de prueba

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity test_problema3 is
    Port ( pi : in bit;
           z : out bit;
           ent : in std_logic_vector(15 downto 0));
end test_problema3;

architecture Behavioral of test_problema3 is
component problema3_paridad is
     generic (n: positive);
    Port (entrada :in std_logic_vector ((n-1) downto 0);
      pi:in bit;
      z:out bit);
end component;

begin
U: problema3_paridad generic map (15) port map (ent,pi,z);

end Behavioral;
```