DESCRIPCION EN VHDL – 2 -

1. DESCRIPCION EN VHDL DE UN CONTADOR

a. TIPO BINARIO ASCENDENTE

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_arith.ALL;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity contadorbinas is
    Port ( reloj : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           salida : out  STD_LOGIC_VECTOR (3 downto 0));
end contadorbinas;

architecture Behavioral of contadorbinas is
signal Qp,Dp : STD_LOGIC_VECTOR (3 downto 0) :=(others =>'0');

begin

--- Registro de estado
process (reloj)
begin
   if reloj'event and reloj='1' then
      if reset='1' then
         Qp <= (others =>'0');
      else
         Qp <= Dp;
      end if;
   end if;
end process;
--- Lógica del estado siguiente

Dp <=  Qp + '1';

--- Lógica de salida

salida <= Qp;

end Behavioral;
```

b. BINARIO ASCENDENTE EN MODULO M

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_arith.ALL;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity contadorbinas is
    Port ( reloj : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           salida : out  STD_LOGIC_VECTOR (3 downto 0));
end contadorbinas;

architecture Behavioral of contadorbinas is
signal Qp,Dp : STD_LOGIC_VECTOR (3 downto 0) :=(others =>'0');

begin

--- Registro de estado
process (reloj)
begin
  if reloj'event and reloj='1' then
    if reset='1' then
      Qp <= (others =>'0');
    else
      Qp <= Dp;
    end if;
  end if;
end process;
--- Lógica del estado siguiente

Dp <=  (others =>'0') when  Qp= "1010" else

Qp + '1';

--- Lógica de salida

salida <= Qp;

end Behavioral;
```

2. DESCRIPCION EN VHDL DE UN REGISTRO DE DESPLAZAMIENTO

a. ENTRADA SERIE SALIDA PARALELO

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_arith.ALL;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity registroserie_paralelo is
    Port ( reloj : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           entrada : in  STD_LOGIC;
           salida : out  STD_LOGIC_VECTOR (3 downto 0));
end registroserie_paralelo;

architecture Behavioral of registroserie_paralelo is
signal Qp,Dp : STD_LOGIC_VECTOR (3 downto 0) :=(others =>'0');

begin

--- Registro de estado
process (reloj)
begin
  if reloj'event and reloj='1' then
    if reset='1' then
      Qp <= (others =>'0');
    else
      Qp <= Dp;
    end if;
  end if;
end process;
--- Lógica del estado siguiente

Dp <=  entrada & Qp (3 downto 1 ) ;

--- Lógica de salida

salida <= Qp;

end Behavioral;
```

b. ENTRADA SERIE SALIDA SERIE

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_arith.ALL;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity registroserie_serie is
    Port ( reloj : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           salida : out  STD_LOGIC;
          entrada : in STD_LOGIC);

end registroserie_serie;

architecture Behavioral of registroserie_serie is
signal Qp,Dp : STD_LOGIC_VECTOR (3 downto 0) :=(others =>'0');

begin

--- Registro de estado
process (reloj)
begin
   if reloj'event and reloj='1' then
       if reset = '1' then
        Qp <= (others =>'0');
      else
         Qp <= Dp;
       end if ;
       end if;
end process;
--- Lógica del estado siguiente

Dp <=  entrada & Qp (3 downto 1 ) ;

--- Lógica de salida

salida <= Qp (0) ;

end Behavioral;
```

o o o O o o o