PRÁCTICA 5: CONTADORES

Edgar A. Ramos Mesas
ESCOM DSD

Códigos Implementados

Para el desarrollo de esta práctica se requirió de dos códigos de implementación. El primero de ellos empleado en un contador ascendente de 3 bits con enable.

Código del contador de 3 bits.

```
library ieee;
use ieee.std_logic_1164.all;
entity contador is port(
       clk, clr, en: in std_logic;
       q: inout std_logic_vector(2 downto 0)
);
attribute pin_numbers of contador : entity is
       "clk:1 clr:2"
       & " q(0):14 q(1):15 q(2):16";
end entity;
architecture a_contador of contador is
begin
process(clk, clr)
variable acc : std_logic;
begin
              if(clr = '1') then
                      q <= "000";
              elsif(rising_edge(clk)) then
                      if (en = '0') then
                             q <= q;
                      else
                             for i in 0 to 2 loop
                                    acc := '1';
                                    for j in 0 to i-1 loop
                                           acc := acc and q(j);
                                    end loop;
                                    q(i) <= q(i) xor (en and acc);
                             end loop;
                      end if;
              end if;
 end process;
 end architecture;
```

Código de implementación contador de 7 bits.

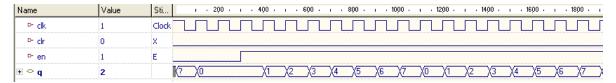
Para el contador de 7 bits se agregaron algunas cosas, como la opción de carga, es decir que se brinda la posibilidad de establecer un estado inicial, así como la oportunidad de escoger un conteo ascendente o descendente a través de un botón de control. El enable funciona del mismo modo que el establecido en el contador de 3 bits, ya que de existir un 1 el contador seguirá avanzando en la dirección deseada, mientras que cuando exista un 0, el contador retendrá el último número contado

```
library ieee;
use ieee.std logic 1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity contadorb is port(
clk, clr, en, ud, 1 : in std_logic;
                                     : in std_logic_vector(6 downto 0);
                                     : out std logic;
У
                                     : inout std_logic_vector(6 downto 0)
q
);
attribute pin numbers of contadorb : entity is
       "clk:1 clr:2"
       & " d(0):6 d(1):7 d(2):8 d(3):9 d(4):10 d(5):11 d(6):13"
       & " q(0):14 q(1):15 q(2):16 q(3):17 q(4):18 q(5):19 q(6):20 y:21";
end entity;
architecture a contadorb of contadorb is
begin
process(clk, clr)
Begin
       if(clr = '1') then
              q <= (others =>'0');
       elsif (rising_edge(clk)) then
              if(en = '0') then
                      q <= q;
              else
                      if(1 = '1') then
                             q \leftarrow d;
                      else
                             if(ud = '1') then
                                     q <= q + 1;
                             else
```

Simulaciones

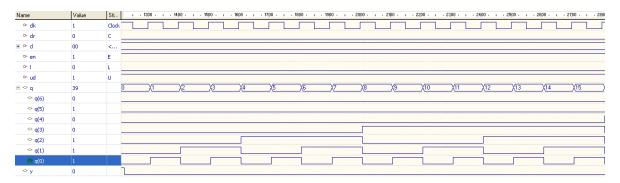
Contador de 3 bits

Al ser un contador de solo 3 bits, el número máximo al que se puede contar es el 7. Es decir que ,al igual que se muestra en la ilustración, el conteo va de 0 a 7.



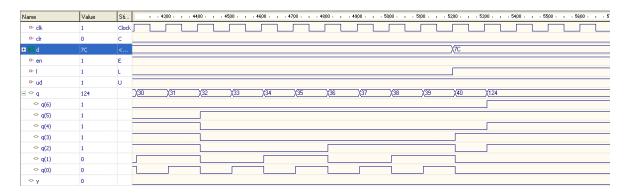
Contador de 7 bits

Conteo Ascendente

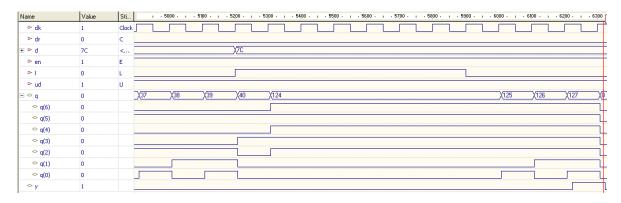


Observamos que el conteo se realiza de forma ascendente como se tenía previsto.

Estableciendo un estado con load.



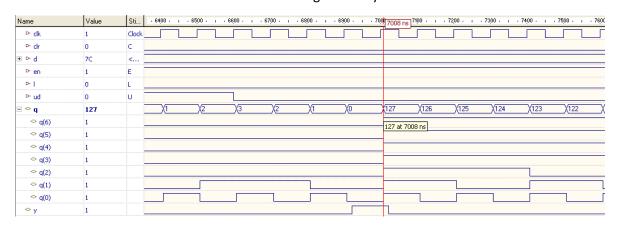
Se observa que el valor 1111100 o 7C en hexadecimal ha sido cargado. Para mostrar el desborde incrementaremos el contador.



Observamos que, al alcanzar el estado de desbordamiento, la salida "y" toma el valor de 1.

Conteo descendente.

Para el conteo descendente la entrada ud se configura en 0 y enable se mantiene.



Observamos que el conteo descendiente se realiza como estaba previsto y que, además, al igual que en el conteo ascendente, al llegar al estado "final" se activa la salida de desbordamiento.