CONTADOR CON CLEAR Y PRESET

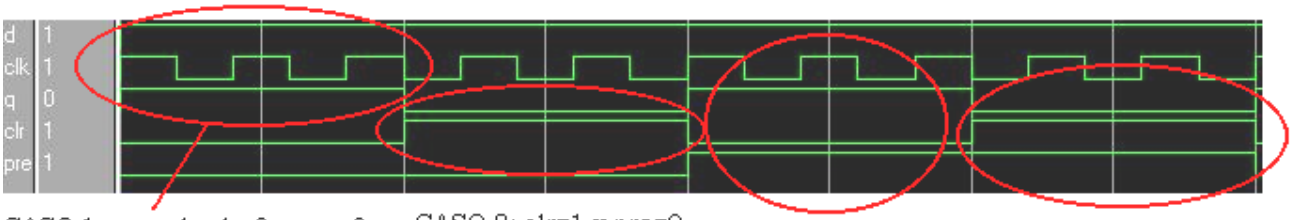
```
ENTITY registro IS
PORT (d,clk,clr,pre : IN BIT;
          q                          : OUT BIT);
END registro;
ARCHITECTURE algoritmo OF registro IS
BEGIN
     PROCESS (clk,clr,pre)
          BEGIN
               IF clr='1' THEN
               q<='0';
               ELSIF pre='1' THEN
               q<='1';
               ELSIF clk='1' THEN
               q<=d;
               END IF;
               END PROCESS;
END algoritmo;
```

Simulando:

```
ENTITY testbench IS
END testbench;

ARCHITECTURE behavior OF testbench IS

     COMPONENT registro
     PORT(
          d : IN bit;
          clk : IN bit;
          clr : IN bit;
          pre : IN bit;
          q : OUT bit);
     END COMPONENT;

     SIGNAL d :  bit;
     SIGNAL clk :  bit;
     SIGNAL clr :  bit;
     SIGNAL pre :  bit;
     SIGNAL q :  bit;

BEGIN

     uut: registro PORT MAP(
          d => d,
```

```
            clk => clk,
            clr => clr,
            pre => pre,
            q => q
      );


-- *** Test Bench - User Defined Section ***
   tb : PROCESS
   BEGIN
     clk<= (not clk );   wait for 4 ns;
   END PROCESS;
   tb1 : PROCESS
   BEGIN
     clr<= (not clr ) after 20 ns;
     wait for 20 ns;
   END PROCESS;
   tb2 : PROCESS
   BEGIN
     pre<= (not pre ) after 40 ns;
     wait for 40 ns;
   END PROCESS;
   tb3 : PROCESS
   BEGIN
     d<= '1';
     wait;
   END PROCESS;
END;
```

Y interpretando la gráfica:



CASO 1: cuando clr=0 y pre=0, tenemos que el flip flop funciona normalmente, es decir por cada pulso de reloj, el valor de d pasa hacia q.

CASO 2: clr=1 y pre=0, aqui la salida q se pondrá a 0 asincronamente.

CASO 3: clr=0 y pre=1, aqui la salida q se pondrá a 1 asincronamente.

CASO 4: clr=1 y pre=1, aqui la salida q se pondrá a 0 asincronamente, ya que el clear tiene más peso.