```
-- Divisor de frecuencia de modulo programable entre 2 y 100


library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity div_freq_prog is
port(
      clk,nrst: in std_logic;
      MODULO      : in std_logic_vector(6 downto 0);
      CICLO : buffer std_logic
);

end entity;

architecture rtl of div_freq_prog is

      signal Q    : std_logic_vector(6 downto 0);


begin

process(clk,nrst)
begin
   if nrst = '0' then
      Q <= (0 => '1', others => '0');
   elsif clk'event and clk = '1' then
      if Q = MODULO or MODULO > 100 then
         Q <= (0 => '1', others => '0');
      else
         Q <= Q + 1;
      end if;
   end if;
end process;

CICLO <= '1' when (Q > '0'&MODULO(6 downto 1)) else '0';

end rtl;
```

```
-- Test Bench del divisor de frecuencia de modulo programable entre 2 y
100


library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity tb_div_freq_prog is
end entity;

architecture test of tb_div_freq_prog is
```

```
        signal clk: std_logic;
        signal nrst: std_logic;
        signal MODULO: std_logic_vector(6 downto 0);
        signal CICLO: std_logic;

        constant T_CLK : time := 1 us;
        signal final: boolean := false;

begin

dut: entity Work.div_freq_prog(rtl)
port map(
        clk => clk,
        nrst => nrst,
        modulo => modulo,
        ciclo => ciclo
        );

process
begin

        clk <= '0';
        wait for T_CLK/2;
        clk <= '1';
        wait for T_CLK/2;
          if final then
             wait;
          end if;
end process;

process
begin
        -- Inicializacion asincrona
        nrst <= '0';
        MODULO <= (others => '1');
        wait until clk'event and clk = '1';
        wait until clk'event and clk = '1';
        nrst <= '1';
        wait until clk'event and clk = '1';

        -- Probamos con el modulo maximo
        MODULO <= "1100100";    -- 100
        wait for 115*T_CLK;
        wait until clk'event and clk = '1';

        -- Probamos con modulo minimo
        MODULO <= "0000010";    -- 2
        wait for 10*T_CLK;
        wait until clk'event and clk = '1';

        -- Probamos con modulo intermedio
        MODULO <= "0110111";    -- 55
        wait for 60*T_CLK;
        wait until clk'event and clk = '1';

        -- Probamos con modulo intermedio
        MODULO <= "0011100";    -- 28
```

```
        wait for 35*T_CLK;
        wait until clk'event and clk = '1';

        -- Probamos con modulo intermedio
        MODULO <= "1011001";    -- 89
        wait for 100*T_CLK;
        wait until clk'event and clk = '1';

        final <= true;
        wait;
end process;
end test;
```