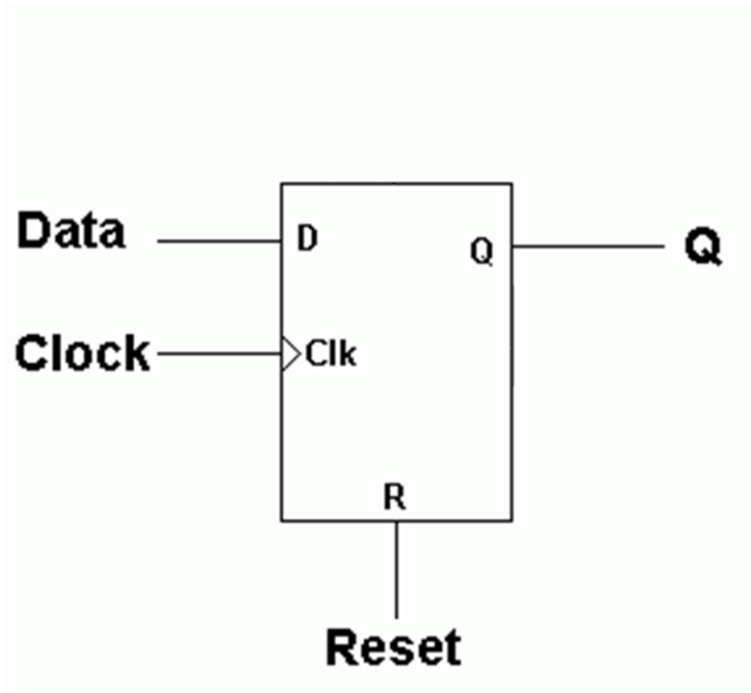
FPGA & VHDL

Aula 2

VHDL: CKTs sequenciais



Flip-Flop D

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

entity d_ff_reset is
    Port ( clk, reset : in  STD_LOGIC;
           d : in  STD_LOGIC;
           q : out  STD_LOGIC);
end d_ff_reset;

architecture arch of d_ff_reset is
begin
   process(clk, reset)
   begin
      if(reset = '1') then
         q <= '0';
      elsif (clk' event and clk='1') then
         q <= d;
      end if;
   end process;
end arch;
```

VHDL: CKTs sequenciais

Contador binário

```
library ieee;
use ieee.std_logic_1164.ALL;
use ieee.numeric_std.ALL;

entity counter is
    generic(N: integer := 8);
    Port (
         clk,reset : in  std_logic;
         max_tick : out  std_logic;
         q : out  std_logic_vector (N-1 downto 0)
         );
end counter;

architecture arch of counter is
   signal r_reg: unsigned (N-1 downto 0);
   signal r_next: unsigned (N-1 downto 0);
begin
   process(clk,reset)
   begin
      if(reset='1') then
         r_reg <= (others => '0');
      elsif (clk' event and clk='1') then
         r_reg <= r_next;
      end if;
   end process;
   r_next <= r_reg + 1;
   q <= std_logic_vector(r_reg);
   max_tick <= '1' when r_reg=(2**N-1) else '0';
end arch;
```

Testbench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

entity counter_tb is
end counter_tb;

architecture arch of counter_tb is
   -- constant
   constant N: integer :=4;
   constant T: time := 20 ns;
   --Inputs
   signal clk,reset,max_tick : std_logic;
   signal q : std_logic_vector(N-1 downto 0);

begin

   -- Instantiate the Unit Under Test (UUT)
   uut: entity work.counter(arch)
      generic map(N=>N)
      port map( clk => clk, reset => reset,
      max_tick => max_tick, q => q);

   -- Clock process definitions
   clk_process :process
   begin
      clk <= '0';
      wait for T/2;
      clk <= '1';
      wait for T/2;
   end process;
   reset <='1', '0' after T/4;
end arch;
```

VHDL: CKTs sequenciais

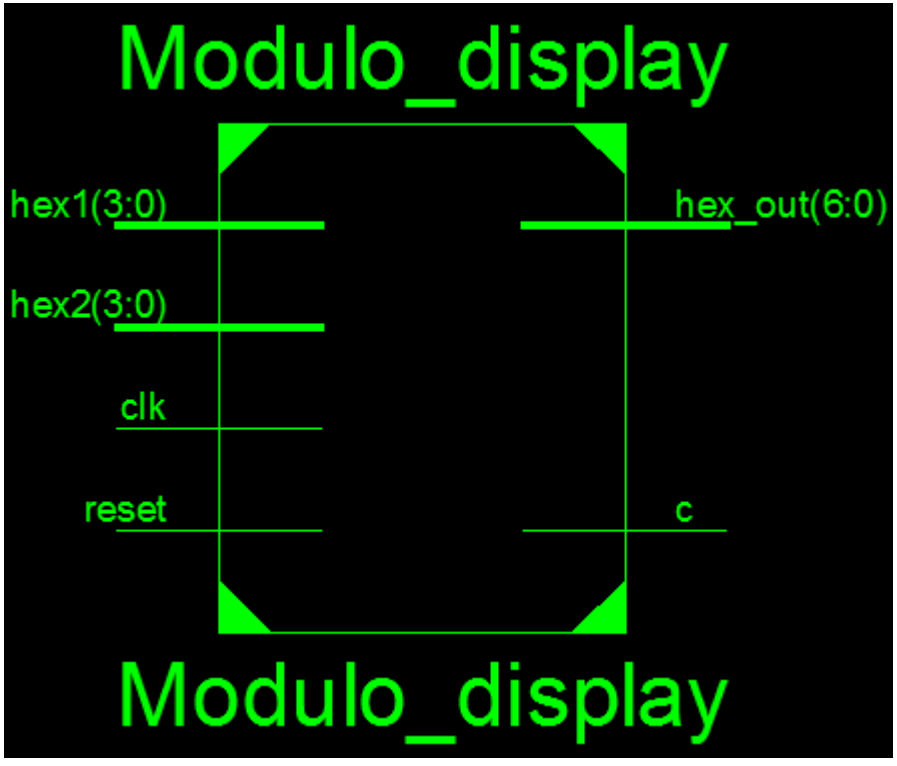
Contador binário up/down

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity counter_up_down is
   generic(N: integer:=8);
   port(
         clk, reset, up: in std_logic;
         max_tick: out std_logic;
         q: out std_logic_vector(N-1 downto 0)
         );
end counter_up_down;

architecture arch of counter_up_down is
   signal r_reg : unsigned (N-1 downto 0);
begin
   process(clk, reset)
   begin
      if(reset='1') then
         r_reg <= (others=>'0');
      elsif (clk' event and clk='1') then
         if(up='1') then
            r_reg <= r_reg +1;
         else
            r_reg <= r_reg-1;
         end if;
      end if;
   end process;
   q <= std_logic_vector(r_reg);
   max_tick <= '1' when r_reg=(2**N-1) else '0';
end arch;
```

Testbench

```
library ieee;
use ieee.std_logic_1164.ALL;

--USE ieee.numeric_std.ALL;

entity counter_up_down_tb is
end counter_up_down_tb;

architecture arch of counter_up_down_tb is
   -- constants
   constant N: integer :=4;
   constant T: time := 20 ns;
   -- inputs
   signal clk,reset,up,max_tick: std_logic;
   signal q : std_logic_vector(N-1 downto 0);
begin
   -- Instantiate the Unit Under Test (UUT)
   uut: entity work.counter_up_down(arch)
      generic map( N=> N)
      port map( clk => clk, reset => reset, up => up,
      max_tick => max_tick, q => q );

   -- Clock process definitions
   clk_process :process
   begin
      clk <= '0';
      wait for T/2;
      clk <= '1';
      wait for T/2;
   end process;

   reset <= '1', '0' after T/4;
   up <= '1', '0' after T*(2**N);
end arch;
```

Display Spartan3E




```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity Modulo_display is
    Port ( clk : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           hex1 : in  STD_LOGIC_VECTOR (3 downto 0);
           hex2 : in  STD_LOGIC_VECTOR (3 downto 0);
           hex_out : out  STD_LOGIC_VECTOR (6 downto 0);
           c : out  STD_LOGIC);
end Modulo_display;

architecture arch of Modulo_display is
   signal clk_aux: std_logic :='0';
   signal tick: std_logic :='0';
begin
counter: entity work.mod_m_counter(arch)
   GENERIC MAP(
    N =>19,M =>500000)
   PORT MAP(
      clk => clk,
      reset => reset,
      max_tick => tick
   );

dis : entity work.display_7_seg(arch)
     PORT MAP(
      clk => clk_aux,
      c => c,
      hex1 => hex1,
      hex2 => hex2,
      hex7seg => hex_out
   );
process (tick)
begin
   if (tick'event and tick='1') then
      clk_aux <= not clk_aux;
   end if;
end process;
end arch;
```

VHDL: CKTs sequenciais

Contador módulo-m

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity mod_m_counter is
   generic (
      N: integer := 4; -- no. of bits
      M: integer := 10 -- mod_M
      );
      port (
         clk, reset: in std_logic;
         max_tick: out std_logic;
         q: out std_logic_vector(N-1 downto 0)
      );
end mod_m_counter;

architecture arch of mod_m_counter is
   signal r_reg: unsigned(N-1 downto 0);
   signal r_next: unsigned(N-1 downto 0);
begin
   process(clk, reset)
   begin
      if(reset='1') then
         r_reg <= (others => '0');
      elsif (clk' event and clk='1') then
         r_reg <= r_next;
      end if;
   end process;
   r_next <= (others=>'0') when r_reg=M-1 else r_reg +1;
   q <= std_logic_vector(r_reg);
   max_tick <= '1' when r_reg=M-1 else '0';
end arch;
```

Testbench

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity mod_m_counter_tb is
end mod_m_counter_tb;

architecture arch of mod_m_counter_tb is
   -- cosntant
   constant N: integer := 6;
   constant M: integer := 4;
   constant T: time := 20 ns;
   signal clk, reset, max_tick: std_logic;
   signal q: std_logic_vector (N-1 downto 0);
begin
   -- Instantiate the Unit Under Test (UUT)
   uut: entity work.mod_m_counter(arch)
   generic map (N => N, M => M)
   port map( clk => clk, reset => reset,
   max_tick => max_tick, q => q );

   process
   begin
      clk <= '0';
      wait for T/2;
      clk <= '1';
      wait for T/2;
   end process;
   reset <= '1', '0' after T/2;
end arch;
```

Timer

- *Clock* da placa:

$$f = 50\ MHz$$
$$T = 20 \times 10^{-9} s$$

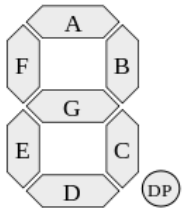
- Para : $\mathrm{t} = 1s$

$$\mathrm{t} = M \times T$$
$$M = 50000000$$

Cálculo de N:

$$2^N > M$$
$$N > 25.57$$
$$N = 26$$

Módulo Display 7 segmentos




```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity display_7_seg is
    Port ( clk : in  STD_LOGIC;
           c : out  STD_LOGIC;
           hex1 : in  STD_LOGIC_VECTOR (3 downto 0);
           hex2 : in  STD_LOGIC_VECTOR (3 downto 0);
           hex7seg : out  STD_LOGIC_VECTOR (6 downto 0));
end display_7_seg;

architecture arch of display_7_seg is
   CONSTANT ds0 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="0111111";
   CONSTANT ds1 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="0000110";
   CONSTANT ds2 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1011011";
   CONSTANT ds3 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1001111";
   CONSTANT ds4 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1100110";
   CONSTANT ds5 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1101101";
   CONSTANT ds6 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1111101";
   CONSTANT ds7 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="0000111";
   CONSTANT ds8 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1111111";
   CONSTANT ds9 : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1100111";
   CONSTANT dsA : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1110111";
   CONSTANT dsB : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1111100";
   CONSTANT dsC : STD_LOGIC_VECTOR (6 DOWNTO 0) :="0111001";
   CONSTANT dsD : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1011110";
   CONSTANT dsE : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1111001";
   CONSTANT dsF : STD_LOGIC_VECTOR (6 DOWNTO 0) :="1110001";
   CONSTANT dsn : STD_LOGIC_VECTOR (6 DOWNTO 0) :="0000000";
   SIGNAL seg_h : STD_LOGIC_VECTOR(6 DOWNTO 0):= (others=>'0');
   SIGNAL seg_l : STD_LOGIC_VECTOR(6 DOWNTO 0):= (others=>'0');
   SIGNAL hex1Int : INTEGER RANGE 15 DOWNTO 0;
   SIGNAL hex2Int : INTEGER RANGE 15 DOWNTO 0;
begin
```

Módulo Display 7 segmentos

```
table_f: BLOCK
         BEGIN
            hex7seg     <= seg_h WHEN clk = '1' ELSE
                           seg_l WHEN clk = '0';
            c <=  clk;
         END BLOCK table_f;
table_h: BLOCK
         BEGIN
            hex1Int <= to_integer(unsigned(hex1));
            seg_h <= ds0 WHEN hex1Int = 0 ELSE
            ds1 WHEN hex1Int = 1 ELSE
            ds2 WHEN hex1Int = 2 ELSE
            ds3 WHEN hex1Int = 3 ELSE
            ds4 WHEN hex1Int = 4 ELSE
            ds5 WHEN hex1Int = 5 ELSE
            ds6 WHEN hex1Int = 6 ELSE
            ds7 WHEN hex1Int = 7 ELSE
            ds8 WHEN hex1Int = 8 ELSE
            ds9 WHEN hex1Int = 9 ELSE
            dsA WHEN hex1Int = 10 ELSE
            dsB WHEN hex1Int = 11 ELSE
            dsC WHEN hex1Int = 12 ELSE
            dsD WHEN hex1Int = 13 ELSE
            dsE WHEN hex1Int = 14 ELSE
            dsF WHEN hex1Int = 15 and hex1 = "1111" ELSE
            dsn;
            END BLOCK table_h;
```

```
table_l: BLOCK
            BEGIN
            hex2Int <= to_integer(unsigned(hex2));
            seg_l <= ds0 WHEN hex2Int = 0 ELSE
               ds1 WHEN hex2Int = 1 ELSE
               ds2 WHEN hex2Int = 2 ELSE
               ds3 WHEN hex2Int = 3 ELSE
               ds4 WHEN hex2Int = 4 ELSE
               ds5 WHEN hex2Int = 5 ELSE
               ds6 WHEN hex2Int = 6 ELSE
               ds7 WHEN hex2Int = 7 ELSE
               ds8 WHEN hex2Int = 8 ELSE
               ds9 WHEN hex2Int = 9 ELSE
               dsA WHEN hex2Int = 10 ELSE
               dsB WHEN hex2Int = 11 ELSE
               dsC WHEN hex2Int = 12 ELSE
               dsD WHEN hex2Int = 13 ELSE
               dsE WHEN hex2Int = 14 ELSE
               dsF WHEN hex2Int = 15 and hex2 = "1111" ELSE
               dsn;
            END BLOCK table_l;

end arch;
```