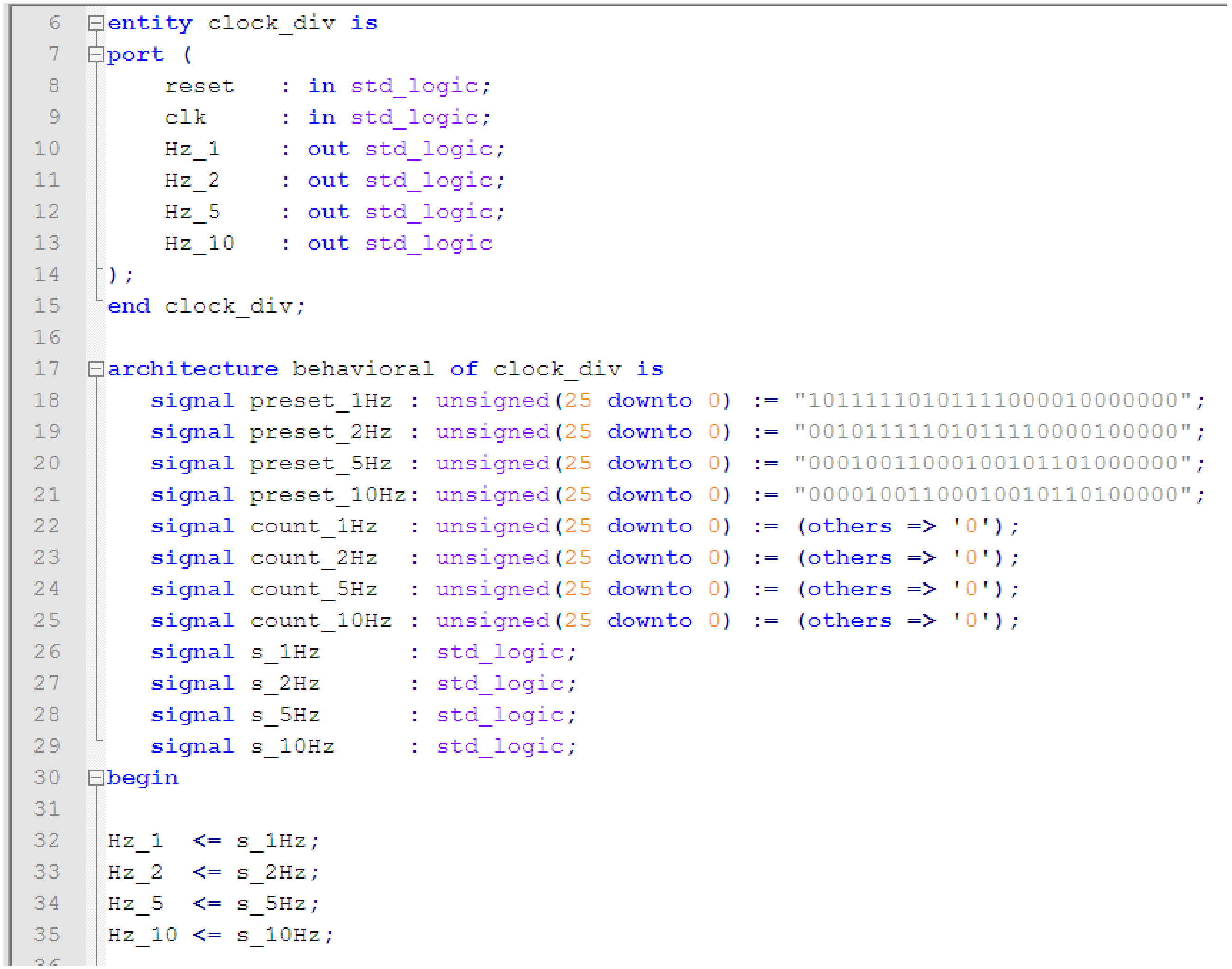
Projeto com Circuitos Reconfiguráveis

Projeto lógico sequencial
divisor de *clock*, contadores e *shift registers*

Prof. Daniel M. Muñoz Arboleda

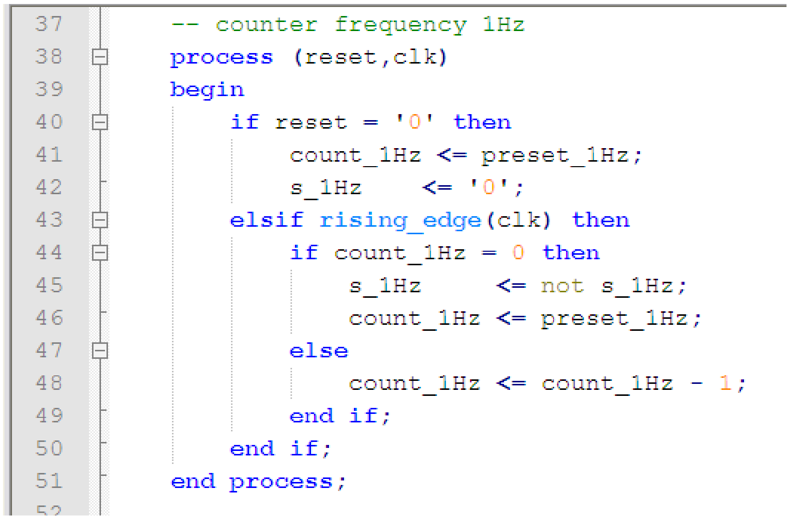
FGA - UnB

Descrição em VHDL de um divisor de *clock*

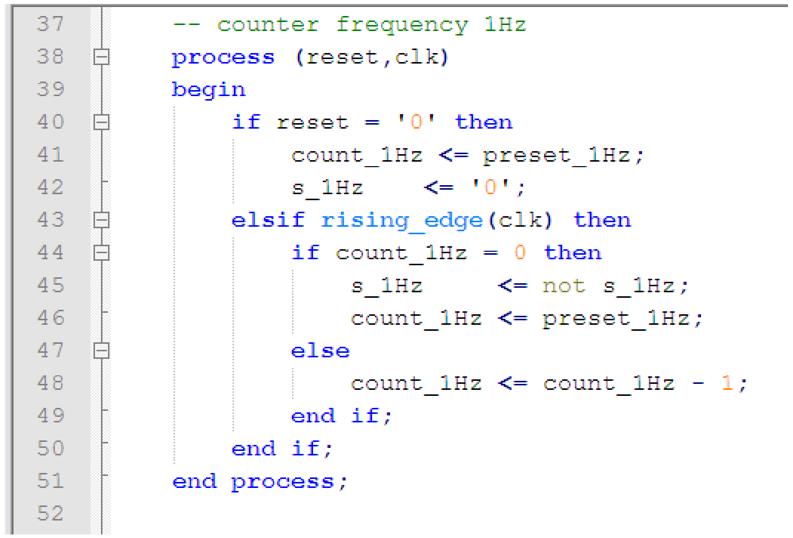


```vhdl
entity clock_div is
port (
        reset    : in std_logic;
        clk      : in std_logic;
        Hz_1     : out std_logic;
        Hz_2     : out std_logic;
        Hz_5     : out std_logic;
        Hz_10    : out std_logic
);
end clock_div;

architecture behavioral of clock_div is
    signal preset_1Hz : unsigned(25 downto 0) := "10111110101111000010000000";
    signal preset_2Hz : unsigned(25 downto 0) := "00101111101011110000100000";
    signal preset_5Hz : unsigned(25 downto 0) := "00010011000100101101000000";
    signal preset_10Hz: unsigned(25 downto 0) := "00001001100010010110100000";
    signal count_1Hz  : unsigned(25 downto 0) := (others => '0');
    signal count_2Hz  : unsigned(25 downto 0) := (others => '0');
    signal count_5Hz  : unsigned(25 downto 0) := (others => '0');
    signal count_10Hz : unsigned(25 downto 0) := (others => '0');
    signal s_1Hz      : std_logic;
    signal s_2Hz      : std_logic;
    signal s_5Hz      : std_logic;
    signal s_10Hz     : std_logic;
begin

Hz_1  <= s_1Hz;
Hz_2  <= s_2Hz;
Hz_5  <= s_5Hz;
Hz_10 <= s_10Hz;
```

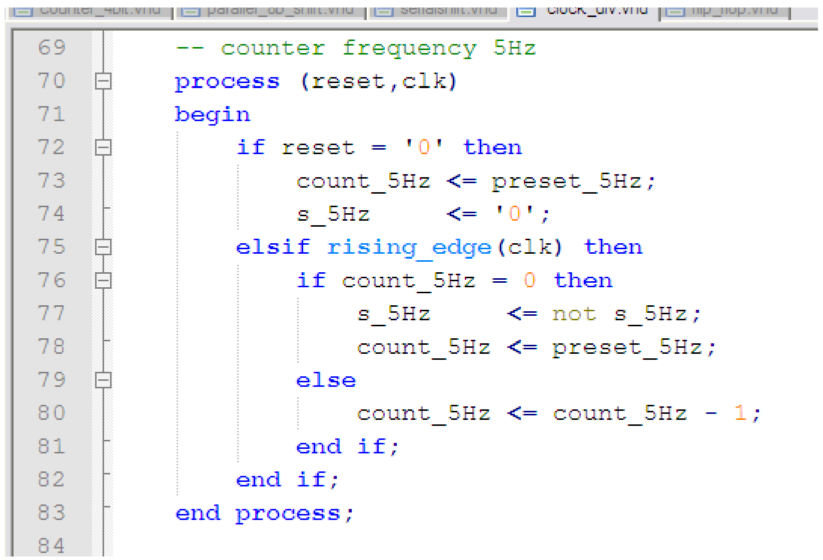
Descrição em VHDL de um divisor de *clock*



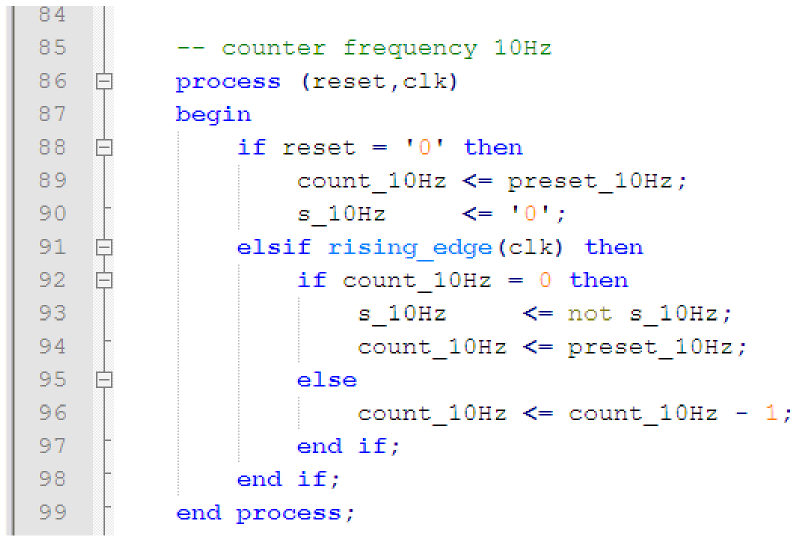
```vhdl
-- counter frequency 1Hz
process (reset,clk)
begin
    if reset = '0' then
        count_1Hz <= preset_1Hz;
        s_1Hz    <= '0';
    elsif rising_edge(clk) then
        if count_1Hz = 0 then
            s_1Hz     <= not s_1Hz;
            count_1Hz <= preset_1Hz;
        else
            count_1Hz <= count_1Hz - 1;
        end if;
    end if;
end process;
```



```vhdl
-- counter frequency 1Hz
process (reset,clk)
begin
    if reset = '0' then
        count_1Hz <= preset_1Hz;
        s_1Hz    <= '0';
    elsif rising_edge(clk) then
        if count_1Hz = 0 then
            s_1Hz     <= not s_1Hz;
            count_1Hz <= preset_1Hz;
        else
            count_1Hz <= count_1Hz - 1;
        end if;
    end if;
end process;
```

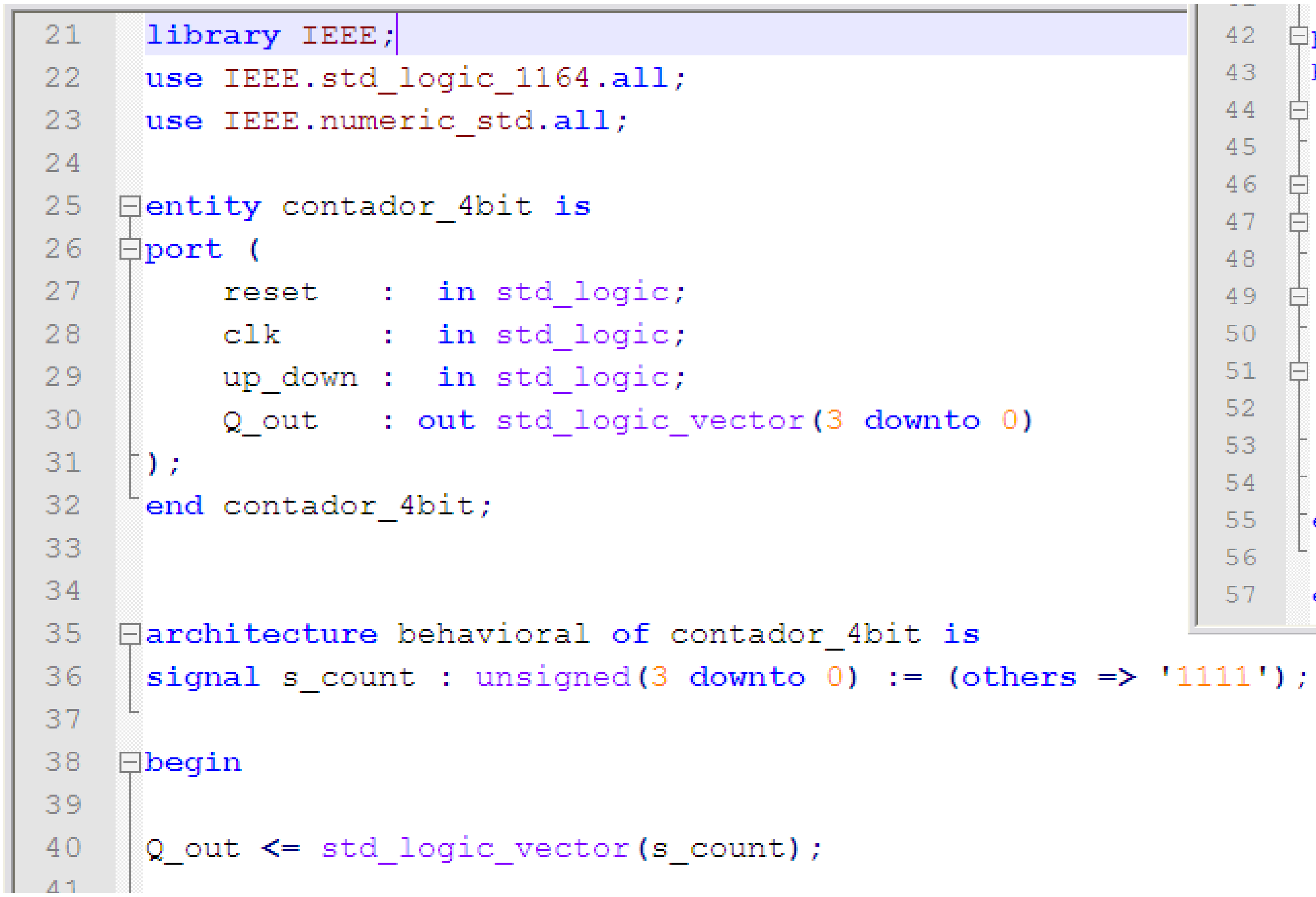


```vhdl
-- counter frequency 5Hz
process (reset,clk)
begin
    if reset = '0' then
        count_5Hz <= preset_5Hz;
        s_5Hz    <= '0';
    elsif rising_edge(clk) then
        if count_5Hz = 0 then
            s_5Hz     <= not s_5Hz;
            count_5Hz <= preset_5Hz;
        else
            count_5Hz <= count_5Hz - 1;
        end if;
    end if;
end process;
```

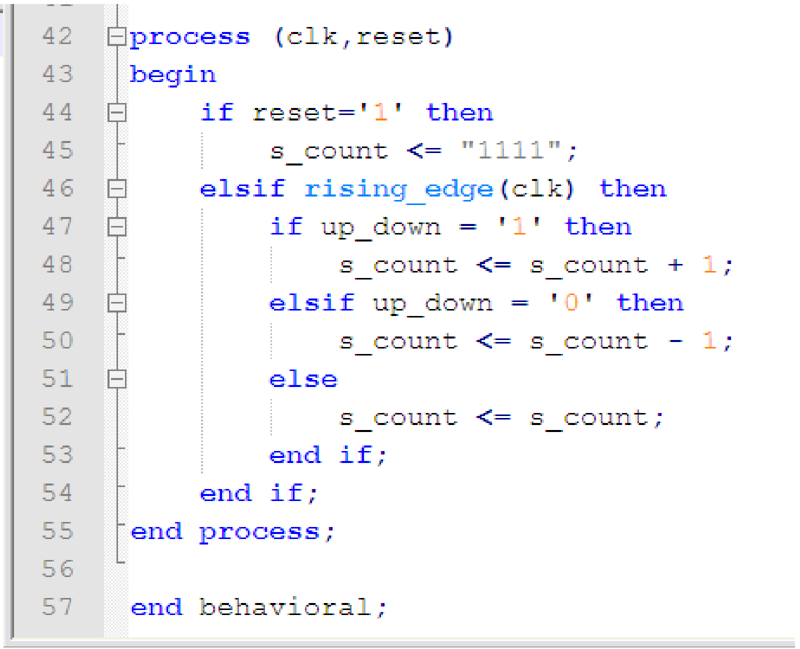


```vhdl
-- counter frequency 10Hz
process (reset,clk)
begin
    if reset = '0' then
        count_10Hz <= preset_10Hz;
        s_10Hz     <= '0';
    elsif rising_edge(clk) then
        if count_10Hz = 0 then
            s_10Hz     <= not s_10Hz;
            count_10Hz <= preset_10Hz;
        else
            count_10Hz <= count_10Hz - 1;
        end if;
    end if;
end process;
```

Descrição em VHDL de um contador de 4bits (exemplo 1)



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

entity contador_4bit is
port (
     reset    :  in std_logic;
     clk      :  in std_logic;
     up_down  :  in std_logic;
     Q_out    : out std_logic_vector(3 downto 0)
);
end contador_4bit;


architecture behavioral of contador_4bit is
signal s_count : unsigned(3 downto 0) := (others => '1111');

begin

Q_out <= std_logic_vector(s_count);
```

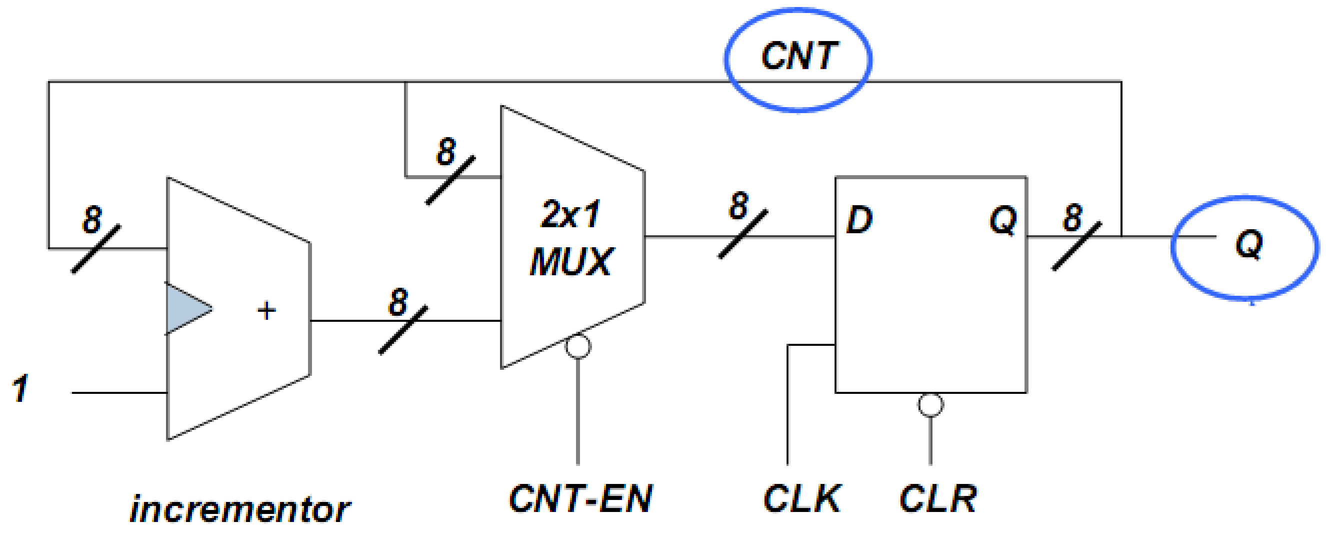


```
process (clk,reset)
begin
    if reset='1' then
        s_count <= "1111";
    elsif rising_edge(clk) then
        if up_down = '1' then
            s_count <= s_count + 1;
        elsif up_down = '0' then
            s_count <= s_count - 1;
        else
            s_count <= s_count;
        end if;
    end if;
end process;

end behavioral;
```

Descrição em VHDL de um contador de 4bits (exemplo 2)

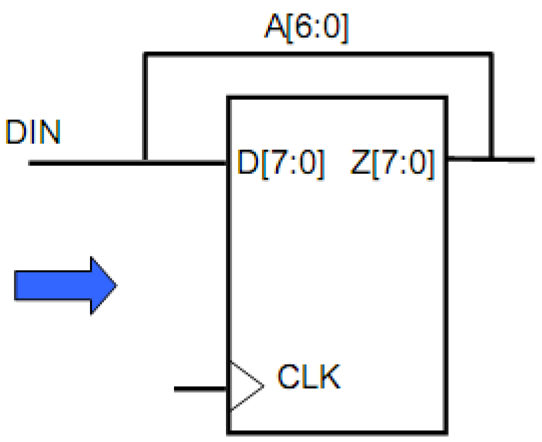
```
library ieee; use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity COUNTER is
 port (CLK,CNT_EN,CLR:in std_logic;
       Q              :out std_logic_vector(7 downto 0));
end COUNTER;
architecture BEHAVE of COUNTER is
  signal CNT:std_logic_vector(7 downto 0);
begin
 FIRST: process (CLK, CLR)
 begin
   if (CLR = '0') then
     CNT <= "00000000";
   elsif (CLK'event and CLK = '1') then
     if (CNT_EN = '0') then
        CNT <= CNT + '1';
     end if ;
   end if ;
 end process FIRST;
 Q <= CNT;
end BEHAVE;
```

Descrição em VHDL de um contador de 4bits (exemplo 2 cont.)

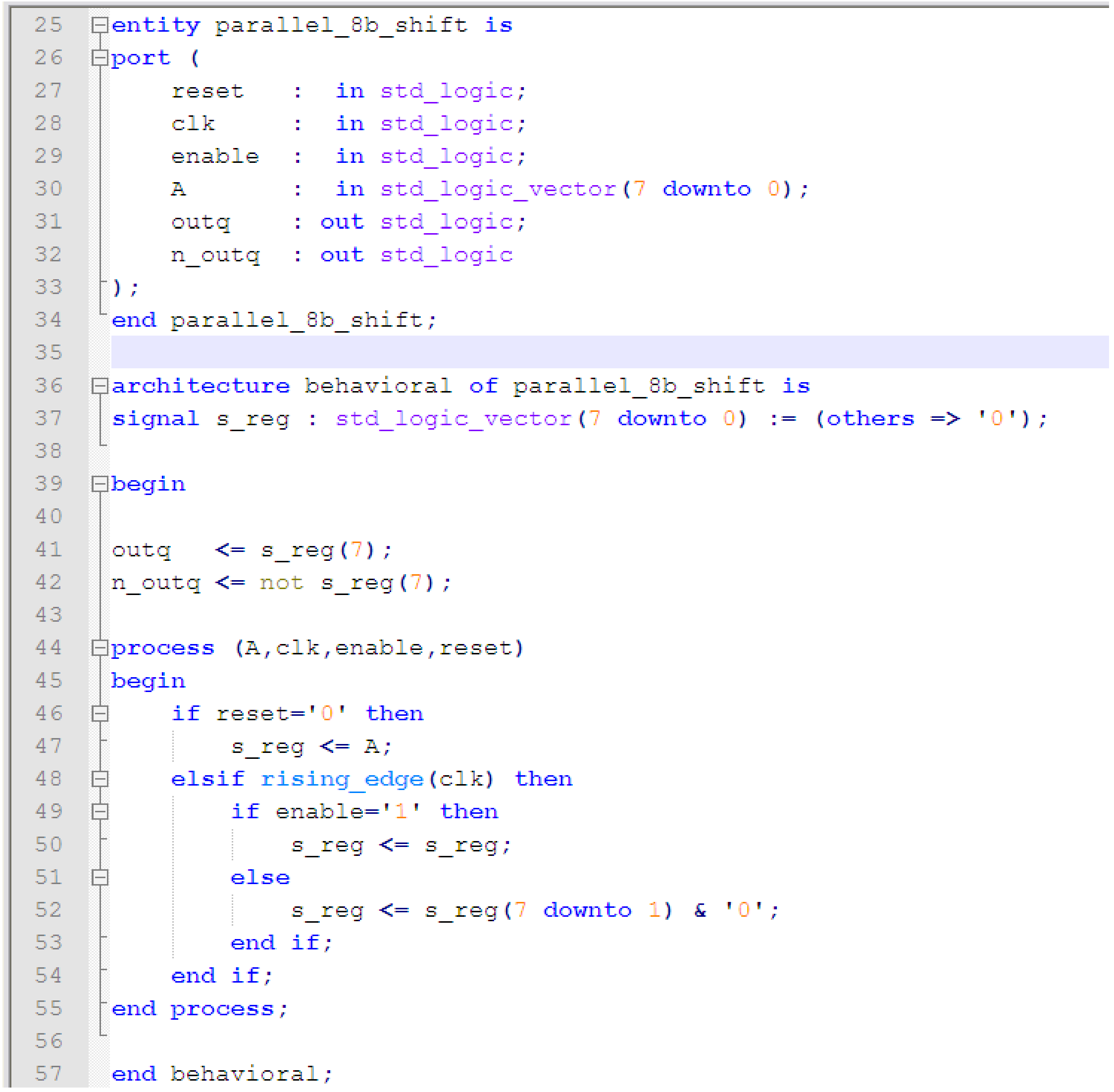


Descrição em VHDL de um *shift register* de 8 bits serial

```
library ieee;
use ieee.std_logic_1164.all;
entity SHIFTER is
port(CLK, DIN: in std_logic;
     Z:  out std_logic_vector(7 downto 0));
end SHIFTER;
architecture RTL of SHIFTER is
signal A: std_logic_vector(7 downto 0);
begin
process (CLK)
begin
if (CLK'event and CLK='1') then
   A <= A (6 downto 0) & DIN; -- shift left
end if;
end process;
Z <= A;
end RTL
```



Descrição em VHDL de um *shift register* de 8 bits paralelo



```
entity parallel_8b_shift is
port (
    reset   :  in std_logic;
    clk     :  in std_logic;
    enable  :  in std_logic;
    A       :  in std_logic_vector(7 downto 0);
    outq    : out std_logic;
    n_outq  : out std_logic
);
end parallel_8b_shift;

architecture behavioral of parallel_8b_shift is
signal s_reg : std_logic_vector(7 downto 0) := (others => '0');

begin

outq   <= s_reg(7);
n_outq <= not s_reg(7);

process (A,clk,enable,reset)
begin
    if reset='0' then
        s_reg <= A;
    elsif rising_edge(clk) then
        if enable='1' then
            s_reg <= s_reg;
        else
            s_reg <= s_reg(7 downto 1) & '0';
        end if;
    end if;
end process;

end behavioral;
```