a.

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

entity SUMADOR_1bit is
        port(a, b, ci : in std_logic;
                s, co: out std_logic);
end SUMADOR_1bit;

architecture SUMADOR_1bit of SUMADOR_1bit is
begin
        s <= ci xor b xor a;
        co <= (ci and (b xor a)) or (b and a);
end SUMADOR_1bit;
```

b.

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

entity COMP_4bit is
        port(a, b : in std_logic_vector (3 downto 0);
                s : out std_logic);
end COMP_4bit;

architecture COMP_4bit of COMP_4bit is
begin
        s <= '0' when a = b else
             '1';
end COMP_4bit;
```

c.

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

entity FlipFlopD is
        port(res,D,clk : in std_logic;
                Q : buffer std_logic);
end FlipFlopD;

architecture FlipFlopD of FlipFlopD is
begin
        process(clk, res)
        begin
                if (rising_edge(clk)) then
                        Q <= '1' when and (res = '0') else
                              '0' when and (D = '1') else
                              Q;
                else    Q <= Q;
                end if;
        end process;
end FlipFlopD;
```

d.

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

entity Contador_3bit is
        port(res,alpha,clk : in std_logic;
                Q : out std_logic_vector (2 downto 0));
end Contador_3bit;

architecture Contador_3bit of Contador_3bit is
begin
        process
        variable temp : unsigned (2 downto 0);
        begin
                wait until (rising_edge(clk));
                if (res = '0') then temp := "000";
                elsif(alpha = '0') then temp := temp + 1;
                else temp := temp;
                end if;
                Q <= std_logic_vector(temp);
        end process;
end Contador_3bit;
```

e.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity Doble_Input is
        port(A : in std_logic_vector (7 downto 0);
                Q : out std_logic_vector (4 downto 0));
end Doble_Input;

architecture Doble_Input of Doble_Input is
begin
        process(A)
        variable num : unsigned (4 downto 0) := "00000";
        begin
                for i in 0 to 7 loop
                        if (A(i) = '1') then num := num + 2;
                        end if;
                end loop;
                Q <= std_logic_vector(num);
        end process;
end Doble_Input;
```

f.

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

entity Mayor_4num2bit is
        port(A,B,C,D : in std_logic_vector (1 downto 0);
                Q : out std_logic_vector (1 downto 0));
end Mayor_4num2bit;

architecture Mayor_4num2bit of Mayor_4num2bit is
begin
        process(A,B,C,D)
        variable max : std_logic_vector (1 downto 0);
        begin
                max := "00";
                if (A > max) then max := A;
                end if;
                if (B > max) then max := B;
                end if;
                if (C > max) then max := C;
                end if;
                if (D > max) then max := D;
                end if;
                Q <= max;
        end process;
end Mayor_4num2bit;
```

g.

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

entity Flanco_Bajada is
        port(clk, E : in std_logic;
                S : out std_logic);
end Flanco_Bajada;

architecture Flanco_Bajada of Flanco_Bajada is
        signal state : std_logic := '0';
begin
        process
        begin
                wait on E, clk;
                if (falling_edge(E)) then state <= '1';
                end if;
                if (rising_edge(clk) and (state = '1')) then
                        state <= '0';
                        S <= '1';
                        wait until rising_edge(clk);
                        S <= '0';
                end if;
        end process;
end Flanco_Bajada;
```