Układ komparatora dwóch liczb n-bitowych (n=3,4) A i B realizującego funkcje: A>B, A=B, A<B. Finalnie zaprojektować z parametrem n.

1. Kod źródłowy (dla liczb 4-bitowych) (zał. kompar.vhd).

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

entity kompar is
        generic (n: positive :=3);
   Port ( V1 : in  STD_LOGIC_VECTOR (n-1 downto 0);
          V2 : in  STD_LOGIC_VECTOR (n-1 downto 0);
          V1bigger : out  STD_LOGIC;
          Vequal : out  STD_LOGIC;
          V1smaller : out  STD_LOGIC);
end kompar;

architecture Behavioral of kompar is
        signal bigger :  STD_LOGIC;
        signal equal :  STD_LOGIC;
        signal smaller : STD_LOGIC;

begin
                process (V1, V2)
                        begin
                                if V1>V2 then
                                        bigger <= '1';
                                        equal <= '0';
                                        smaller <= '0';
                                elsif V1<V2 then
                                        bigger <= '0';
                                        equal <= '0';
                                        smaller <= '1';
                                elsif V1=V2 then
                                        bigger <= '0';
                                        equal <= '1';
                                        smaller <= '0';
                                end if;
                end process;
        V1bigger <= bigger;
        Vequal  <= equal;
        V1smaller <= smaller;

end Behavioral;
```

2. Test bench (dla liczb 4-bitowych) (zał. test3.vhd).

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

```
ENTITY test2 IS
END test2;

ARCHITECTURE behavior OF test2 IS

   -- Component Declaration for the Unit Under Test (UUT)

   COMPONENT kompar
   PORT(
        V1 : IN  std_logic_vector(2 downto 0);
        V2 : IN  std_logic_vector(2 downto 0);
        V1bigger : OUT  std_logic;
        Vequal : OUT  std_logic;
        V1smaller : OUT  std_logic
       );
   END COMPONENT;

  --Inputs
  signal V1 : std_logic_vector(2 downto 0) := (others => '0');
  signal V2 : std_logic_vector(2 downto 0) := (others => '0');

          --Outputs
  signal V1bigger : std_logic;
  signal Vequal : std_logic;
  signal V1smaller : std_logic;
  -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name

BEGIN

          -- Instantiate the Unit Under Test (UUT)
  uut: kompar PORT MAP (
         V1 => V1,
         V2 => V2,
         V1bigger => V1bigger,
         Vequal => Vequal,
         V1smaller => V1smaller
       );

  -- Stimulus process
  stim_proc: process
  begin

     wait for 10 ns;
                  V1<="010";
                  wait for 10 ns;
                  V2<="100";
                  wait for 10 ns;
                  V1<="111";
                  wait for 10 ns;
```

```
                V2<="111";

    wait;
  end process;

END;
```

3. Opis projektu.

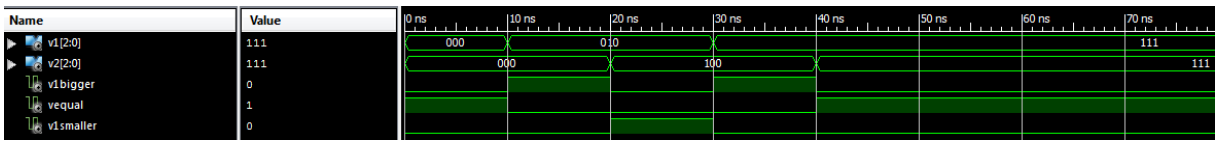
Projekt posiada:

- n-bitowe wejścia V1 i V2 zależne od deklarowanej stałej n, na które podawane są liczby do porównania,
- wyjście V1bigger, na które podawany jest stan wysoki w przypadku gdy liczba V1 jest większa, w pozostałych przypadkach podawany jest stan niski,
- wyjście Vequal , na które podawany jest stan wysoki w przypadku gdy liczby są równe, w pozostałych przypadkach podawany jest stan niski,
- wyjście V1smaller, na które podawany jest stan wysoki w przypadku gdy liczba V1 jest mniejsza, w pozostałych przypadkach podawany jest stan niski,

4. Symulacja.

- Dla liczb 3-bitowych (n=3).

Na początku, oba wejścia mają wartość 0, więc przez 10ns stan wysoki jest na wyjściu Vequal. W 10ns wejście V1 przyjmuje wartość 2, więc przez kolejne 10ns stan wysoki jest na wyjściu V1bigger. W 20ns wejście V2 przyjmuje wartość 4, więc przez kolejne 10ns stan wysoki jest na wyjściu V1smaller. W 30ns wejście V1 przyjmuje wartość 7, więc przez kolejne 10ns, ponownie stan wysoki jest na wyjściu V1bigger. W 40ns, również wejście V2 przyjmuje wartość 7 i do końca symulacji stan wysoki jest na wyjściu Vequal.



- Dla liczb 4-bitowych (n=4).

Na początku, oba wejścia mają wartość 0, więc przez 10ns stan wysoki jest na wyjściu Vequal. W 10ns wejście V1 przyjmuje wartość 6, więc przez kolejne 10ns stan wysoki jest na wyjściu V1bigger. W 20ns wejście V2 przyjmuje wartość 10, więc przez kolejne 10ns stan wysoki jest na wyjściu V1smaller. W 30ns wejście V1 przyjmuje wartość 13, więc przez kolejne 10ns, ponownie stan wysoki jest na wyjściu V1bigger. W 40ns, również wejście V2 przyjmuje wartość 13 i do końca symulacji stan wysoki jest na wyjściu Vequal.