Digital-Electronics-1

Laboratory N.2

Link text itself

Exercise 1: The truth table for 2-bit

```
**Dec.equivalent** | **B[1:0]** | **A[1:0]** | B is greater than A | B equals A |
--- | --- | --- | --- | --- | ---
0 | 00 | 00 | 0 | 1 | 0
1 | 00 | 01 | 0 | 0 | 1
2 | 00 | 10 | 0 | 0 | 1
3 | 00 | 11 | 0 | 0 | 1
4 | 01 | 00 | 1 | 0 | 0
5 | 01 | 01 | 0 | 1 | 0
6 | 01 | 10 | 0 | 0 | 1
7 | 01 | 11 | 0 | 0 | 1
8 | 10 | 00 | 1 | 0 | 0
9 | 10 | 01 | 1 | 0 | 0
10 | 10 | 10 | 0 | 1 | 0 |
11 | 10 | 11 | 0 | 0 | 1 |
12 | 11 | 00 | 1 | 0 | 0 |
13 | 11 | 01 | 1 | 0 | 0 |
14 | 11 | 10 | 1 | 0 | 0 |
15 | 11 | 11 | 0 | 1 | 0 |
```

Exercise 2: Comparator for 2-bit

```
### **Karnaugh maps**

#### **Greater than function Karnaugh map**
![Karnaugh map 1](https://github.com/AnaSampaio13/Digital-Electronics-1/blob/mair

### **Equals to function Karnaugh map**
![Karnaugh map 2](https://github.com/AnaSampaio13/Digital-Electronics-1/blob/mair

### **Less than function Karnaugh map**
![Karnaugh map 3](https://github.com/AnaSampaio13/Digital-Electronics-1/blob/mair

### **Simplified SoP form of the "greater than" function**
Greater_Sop = m4 + m8 + m9 + m12 + m13 + m14 = (/b1.b0./a1./a0) + (b1./b0./a1./a0

### **Simplified PoS form of the "less than" function**
Greater_PoS = m0 . m4 . m5 . m8 . m9. m10 . m12 . m13 . m14 . m15 = (b1+b0+a1+a0)

### **EDA Playground example: **
![Link text itself](https://www.edaplayground.com/x/hQ7j)
```

```
------------------------------------------------------------------------
--
-- Example of 2-bit binary comparator using the when/else assignment.
-- EDA Playground
--
-- Copyright (c) 2020-2021 Tomas Fryza
-- Dept. of Radio Electronics, Brno University of Technology, Czechia
-- This work is licensed under the terms of the MIT license.
--
------------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;

------------------------------------------------------------------------
-- Entity declaration for 2-bit binary comparator
------------------------------------------------------------------------
entity comparator_2bit is
    port(
       a_i           : in  std_logic_vector(2 - 1 downto 0);
       b_i                       : in  std_logic_vector(2 - 1 downto 0);
       B_less_A_o    : out std_logic;     -- B is less than A
       B_greater_A_o : out std_logic;     -- B is greater than A
       B_equals_A_o  : out std_logic      -- B is equal to A
    );
end entity comparator_2bit;

------------------------------------------------------------------------
-- Architecture body for 2-bit binary comparator
------------------------------------------------------------------------
architecture Behavioral of comparator_2bit is
begin
    B_less_A_o    <= '1' when (b_i < a_i) else '0';
    B_greater_A_o <= '1' when (b_i > a_i) else '0';
    B_equals_A_o  <= '1' when (b_i = a_i) else '0';


end architecture Behavioral;



------------------------------------------------------------------------
--
-- Testbench for 2-bit binary comparator.
-- EDA Playground
--
-- Copyright (c) 2020-2021 Tomas Fryza
-- Dept. of Radio Electronics, Brno University of Technology, Czechia
-- This work is licensed under the terms of the MIT license.
--
------------------------------------------------------------------------

library ieee;
```

```
use ieee.std_logic_1164.all;

--------------------------------------------------------------------------
-- Entity declaration for testbench
--------------------------------------------------------------------------
entity tb_comparator_2bit is
    -- Entity of testbench is always empty
end entity tb_comparator_2bit;

--------------------------------------------------------------------------
-- Architecture body for testbench
--------------------------------------------------------------------------
architecture testbench of tb_comparator_2bit is

    -- Local signals
    signal s_a       : std_logic_vector(2 - 1 downto 0);
    signal s_b       : std_logic_vector(2 - 1 downto 0);
    signal s_B_greater_A : std_logic;
    signal s_B_equals_A  : std_logic;
    signal s_B_less_A    : std_logic;

begin
    -- Connecting testbench signals with comparator_2bit entity (Unit Under Test)
    uut_comparator_2bit : entity work.comparator_2bit
        port map(
            a_i           => s_a,
            b_i           => s_b,
            B_greater_A_o => s_B_greater_A,
            B_equals_A_o  => s_B_equals_A,
            B_less_A_o    => s_B_less_A
        );

    --------------------------------------------------------------------------
    -- Data generation process
    --------------------------------------------------------------------------
    p_stimulus : process
    begin
        -- Report a note at the begining of stimulus process
        report "Stimulus process started" severity note;


        -- First test values
        s_b <= "00"; s_a <= "00"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '0') and (s_B_equals_A = '1') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 00, 00" severity error;

        -- Second test values
         s_b <= "01"; s_a <= "00"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 01, 00" severity error;
```

```
        -- Third test values
        s_b <= "11"; s_a <= "00"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 11, 00" severity error;

        -- Fourth test values
        s_b <= "01"; s_a <= "11"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 01, 11" severity error;

        -- Fifth test values
        s_b <= "00"; s_a <= "10"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 00, 10" severity error;



        -- Report a note at the end of stimulus process
        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;

end architecture testbench;
```

Exercise 3: Comparator for 4-bit

```
### **EDA Playground example: **

![Link text itself](https://www.edaplayground.com/x/FbCd)

--------------------------------------------------------------------------
--
-- Example of 4-bit binary comparator using the when/else assignment.
-- EDA Playground
--
-- Copyright (c) 2020-2021 Tomas Fryza
-- Dept. of Radio Electronics, Brno University of Technology, Czechia
-- This work is licensed under the terms of the MIT license.
--
--------------------------------------------------------------------------

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

```
----------------------------------------------------------------------------
-- Entity declaration for 4-bit binary comparator
----------------------------------------------------------------------------
entity comparator_4bit is
    port(
        a_i           : in  std_logic_vector(4 - 1 downto 0);
        b_i           : in  std_logic_vector(4 - 1 downto 0);

        B_less_A_o    : out std_logic;      -- B is less than A
        B_equals_A_o  : out std_logic;      -- B is equal than A
        B_greater_A_o : out std_logic       -- B is greater than A
    );

end entity comparator_4bit;

----------------------------------------------------------------------------
-- Architecture body for 4-bit binary comparator
----------------------------------------------------------------------------
architecture Behavioral of comparator_4bit is
begin
    B_less_A_o   <= '1' when (b_i < a_i) else '0';
    B_equals_A_o <= '1' when (b_i = a_i) else '0';
    B_greater_A_o<= '1' when (b_i > a_i) else '0';



end architecture Behavioral;



----------------------------------------------------------------------------
--
-- Testbench for 4-bit binary comparator.
-- EDA Playground
--
-- Copyright (c) 2020-2021 Tomas Fryza
-- Dept. of Radio Electronics, Brno University of Technology, Czechia
-- This work is licensed under the terms of the MIT license.
--
----------------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;

----------------------------------------------------------------------------
-- Entity declaration for testbench
----------------------------------------------------------------------------
entity tb_comparator_4bit is
    -- Entity of testbench is always empty
end entity tb_comparator_4bit;

----------------------------------------------------------------------------
-- Architecture body for testbench
----------------------------------------------------------------------------
```

```
    architecture testbench of tb_comparator_4bit is

        -- Local signals
        signal s_a       : std_logic_vector(4 - 1 downto 0);
        signal s_b       : std_logic_vector(4 - 1 downto 0);
        signal s_B_greater_A : std_logic;
        signal s_B_equals_A  : std_logic;
        signal s_B_less_A    : std_logic;

    begin
        -- Connecting testbench signals with comparator_4bit entity (Unit Under Test)
        uut_comparator_4bit : entity work.comparator_4bit
            port map(
                a_i           => s_a,
                b_i           => s_b,
                B_greater_A_o => s_B_greater_A,
                B_equals_A_o  => s_B_equals_A,
                B_less_A_o    => s_B_less_A
            );

        ----------------------------------------------------------------------
        -- Data generation process
        ----------------------------------------------------------------------
        p_stimulus : process
        begin
            -- Report a note at the beginning of stimulus process
            report "Stimulus process started" severity note;


            -- First test values
            s_b <= "0000"; s_a <= "0000"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '1') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 0000, 1100" severity error;

             -- Second test values
             s_b <= "0010"; s_a <= "0000"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 0001, 0000" severity error;

            -- Third test values
            s_b <= "0011"; s_a <= "0000"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 0011, 0000" severity error;

            -- Fourth test values
            s_b <= "0000"; s_a <= "0011"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
```

```
            report "Test failed for input combination: 0001, 0011" severity error;

            -- Fifth test values
            s_b <= "0000"; s_a <= "0010"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 0000, 0010" severity error;

                      -- Sixth test values
            s_b <= "1010"; s_a <= "1010"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '1') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 1010, 1010" severity error;

            -- Seventh test values
            s_b <= "1100"; s_a <= "1000"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 1100, 1000" severity error;

            -- Eight test values
            s_b <= "1100"; s_a <= "0000"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 1100, 0000" severity error;

            -- Ninth test values
            s_b <= "0000"; s_a <= "1111"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 0000, 1111" severity error;

            -- Tenth test values
            s_b <= "1111"; s_a <= "1111"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '1') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 1111, 1111" severity error;



            -- Report a note at the end of stimulus process
            report "Stimulus process finished" severity note;
            wait;
        end process p_stimulus;

    end architecture testbench;
```

**EDA Playground example with one reported error: **

```
------------------------------------------------------------------------

![Link text itself](https://www.edaplayground.com/x/AjCA)

------------------------------------------------------------------------
--
-- Example of 4-bit binary comparator using the when/else assignment.
-- EDA Playground
--
-- Copyright (c) 2020-2021 Tomas Fryza
-- Dept. of Radio Electronics, Brno University of Technology, Czechia
-- This work is licensed under the terms of the MIT license.
--
------------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;

------------------------------------------------------------------------
-- Entity declaration for 4-bit binary comparator
------------------------------------------------------------------------
entity comparator_4bit is
    port(
        a_i           : in  std_logic_vector(4 - 1 downto 0);
        b_i           : in  std_logic_vector(4 - 1 downto 0);

        B_less_A_o    : out std_logic;      -- B is less than A
        B_equals_A_o  : out std_logic;      -- B is equal than A
        B_greater_A_o : out std_logic       -- B is greater than A
    );

end entity comparator_4bit;

------------------------------------------------------------------------
-- Architecture body for 4-bit binary comparator
------------------------------------------------------------------------
architecture Behavioral of comparator_4bit is
begin
    B_less_A_o   <= '1' when (b_i < a_i) else '0';
    B_equals_A_o <= '1' when (b_i = a_i) else '0';
    B_greater_A_o<= '1' when (b_i > a_i) else '0';



end architecture Behavioral;



------------------------------------------------------------------------
--
-- Testbench for 4-bit binary comparator.
```

```
-- EDA Playground
--
-- Copyright (c) 2020-2021 Tomas Fryza
-- Dept. of Radio Electronics, Brno University of Technology, Czechia
-- This work is licensed under the terms of the MIT license.
--
------------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;

------------------------------------------------------------------------
-- Entity declaration for testbench
------------------------------------------------------------------------
entity tb_comparator_4bit is
    -- Entity of testbench is always empty
end entity tb_comparator_4bit;

------------------------------------------------------------------------
-- Architecture body for testbench
------------------------------------------------------------------------
architecture testbench of tb_comparator_4bit is

    -- Local signals
    signal s_a       : std_logic_vector(4 - 1 downto 0);
    signal s_b       : std_logic_vector(4 - 1 downto 0);
    signal s_B_greater_A : std_logic;
    signal s_B_equals_A  : std_logic;
    signal s_B_less_A    : std_logic;

begin
    -- Connecting testbench signals with comparator_4bit entity (Unit Under Test)
    uut_comparator_4bit : entity work.comparator_4bit
        port map(
            a_i           => s_a,
            b_i           => s_b,
            B_greater_A_o => s_B_greater_A,
            B_equals_A_o  => s_B_equals_A,
            B_less_A_o    => s_B_less_A
        );

    ------------------------------------------------------------------------
    -- Data generation process
    ------------------------------------------------------------------------
    p_stimulus : process
    begin
        -- Report a note at the beginning of stimulus process
        report "Stimulus process started" severity note;


        -- First test values
        s_b <= "0000"; s_a <= "0000"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '0') and (s_B_equals_A = '1') and (s_B_less_A =
        -- If false, then report an error
```

```
        report "Test failed for input combination: 0000, 1100" severity error;

         -- Second test values
         s_b <= "0010"; s_a <= "0000"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 0001, 0000" severity error;

        -- Third test values
        s_b <= "0011"; s_a <= "0100"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 0011, 0000" severity error;

        -- Fourth test values
        s_b <= "0000"; s_a <= "0011"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 0001, 0011" severity error;

        -- Fifth test values
        s_b <= "0000"; s_a <= "0010"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 0000, 0010" severity error;


    -- Sixth test values
        s_b <= "1010"; s_a <= "1010"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '0') and (s_B_equals_A = '1') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 1010, 1010" severity error;

        -- Seventh test values
        s_b <= "1100"; s_a <= "1000"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 1100, 1000" severity error;

        -- Eight test values
        s_b <= "1100"; s_a <= "0000"; wait for 100 ns;
        -- Expected output
        assert ((s_B_greater_A = '1') and (s_B_equals_A = '0') and (s_B_less_A =
        -- If false, then report an error
        report "Test failed for input combination: 1100, 0000" severity error;

        -- Ninth test values
        s_b <= "0000"; s_a <= "1111"; wait for 100 ns;
        -- Expected output
```

```
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '0') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 0000, 1111" severity error;

            -- Tenth test values
            s_b <= "1111"; s_a <= "1111"; wait for 100 ns;
            -- Expected output
            assert ((s_B_greater_A = '0') and (s_B_equals_A = '1') and (s_B_less_A =
            -- If false, then report an error
            report "Test failed for input combination: 1111, 1111" severity error;


            -- Report a note at the end of stimulus process
            report "Stimulus process finished" severity note;
            wait;
        end process p_stimulus;

    end architecture testbench;
```

[2021-02-23 03:16:26 EST] ghdl -i design.vhd testbench.vhd && ghdl -m tb_comparator_4bit && ghdl -r tb_comparator_4bit
analyze design.vhd analyze testbench.vhd elaborate tb_comparator_4bit testbench.vhd:51:9:@0ms:(report note): Stimulus process started testbench.vhd:71:9:@300ns:(assertion error): Test failed for input combination: 0011, 0000 testbench.vhd:127:9:@1us:(report note): Stimulus process finished Done