Sprawozdanie

Inżynieria Układów Programowalnych

Nazwa ćwiczenia	Układ kombinacyjny
Numer ćwiczenia	2
Imię	Mateusz
Nazwisko	Gabryel
Numer indeksu	181329
Kierunek	Elektronika i Telekomunikacja
Numer grupy dziekańskiej	3

Opis działania zadania:

Układ po zaprogramowaniu, służy jako licznik jedynek ustawionych na wejściu. Na podstawie wprowadzonego słowa 4-bitowego, wyświetla informacje, na siedmiosegmentowym wyświetlaczu znajdującego się na płytce FPGA, liczbie „jedynek” w słowie wejściowym.

W sytuacji gdy wprowadzone słowo nie zawiera żadnej „jedynki”, układ sygnalizuje to na wyświetlaczu siedmiosegmentowym za pomocą wyświetlanej cyfry „0”.

W sytuacji gdy wprowadzone słowo zawiera jedną „jedynkę”, układ sygnalizuje to na wyświetlaczu siedmiosegmentowym za pomocą wyświetlanej cyfry „1”.

W sytuacji gdy wprowadzone słowo zawiera dwie „jedynki”, układ sygnalizuje to na wyświetlaczu siedmiosegmentowym za pomocą wyświetlanej cyfry „2”.

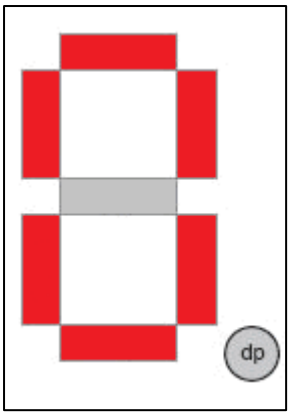
W sytuacji gdy wprowadzone słowo zawiera trzy „jedynki”, układ sygnalizuje to na wyświetlaczu siedmiosegmentowym za pomocą wyświetlanej cyfry „3”.

W sytuacji gdy wprowadzone słowo zawiera cztery „jedynki”, układ sygnalizuje to na wyświetlaczu siedmiosegmentowym za pomocą wyświetlanej cyfry „4”.

W przeprowadzonej symulacji układu przełączniki są zainicjalizowane na OFF, a następnie co 100 ms przełączane na ON, tak aby zostały wypróbowane wszystkie 16 możliwych kombinacji.

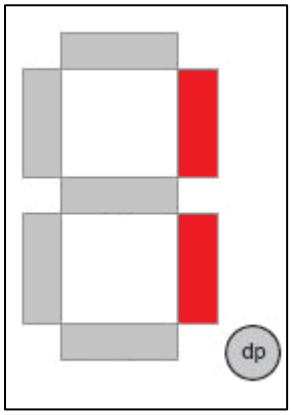
Aktywowanie poszczególnych segmentów wyświetlacza odbywa się poprzez podanie stanu niskiego „0” na wyprowadzenie AN0 - AN3.

Aktywowanie poszczególnych segmentów pojedynczego wyświetlacza odbywa się poprzez podanie stanu niskiego „0” na wyprowadzeniach A-G.



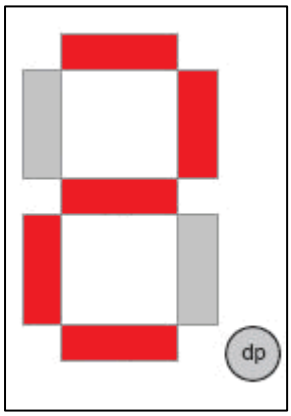
Do uzyskania na wyświetlaczu cyfry „0” potrzebne jest podanie następujących sygnałów: A=’0’, B=’0’, C=’0’ D=’0’, E=’0’, F=’0’, G=’1’, DP=’1’ („00000011”). Podanie sygnałów w takiej kolejności spowoduje zmianę wyświetlacza na sytuację tak jak na zdjęciu obok.

Kolor czerwony na zdjęciu symbolizuje świeceniu się segmentu, a kolor szary symbolizuje wygaszenie segmentu.



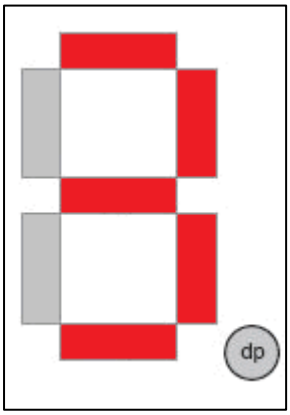
Do uzyskania na wyświetlaczu cyfry „1” potrzebne jest podanie następujących sygnałów: A=’1’, B=’0’, C=’0’, D=’1’, E=’1’, F=’1’, G=’1’, DP=’1’ („10011111”). Podanie sygnałów w takiej kolejności spowoduje zmianę wyświetlacza na sytuację tak jak na zdjęciu obok.

Kolor czerwony na zdjęciu symbolizuje świeceniu się segmentu, a kolor szary symbolizuje wygaszenie segmentu.



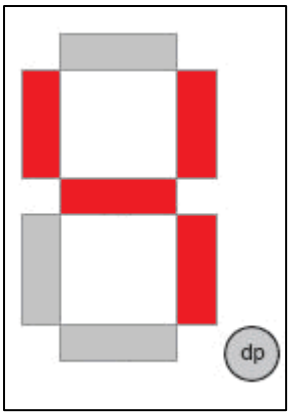
Do uzyskania na wyświetlaczu cyfry „2” potrzebne jest podanie następujących sygnałów: A=’0’, B=’0’, C=’1’, D=’0’, E=’0’, F=’1’, G=’0’, DP=’1’ („00100101”). Podanie sygnałów w takiej kolejności spowoduje zmianę wyświetlacza na sytuację tak jak na zdjęciu obok.

Kolor czerwony na zdjęciu symbolizuje świeceniu się segmentu, a kolor szary symbolizuje wygaszenie segmentu.



Do uzyskania na wyświetlaczu cyfry „3” potrzebne jest podanie następujących sygnałów: A=’0’, B=’0’, C=’0’, D=’0’, E=’1’, F=’1’, G=’0’, DP=’1’ („00001101). Podanie sygnałów w takiej kolejności spowoduje zmianę wyświetlacza na sytuację tak jak na zdjęciu obok.

Kolor czerwony na zdjęciu symbolizuje świeceniu się segmentu, a kolor szary symbolizuje wygaszenie segmentu.



Do uzyskania na wyświetlaczu cyfry „4” potrzebne jest podanie następujących sygnałów: A=’1’, B=’0’, C=’0’, D=’1’, E=’1’, F=’0’, G=’0’, DP=’1’ („10011001). Podanie sygnałów w takiej kolejności spowoduje zmianę wyświetlacza na sytuację tak jak na zdjęciu obok.

Kolor czerwony na zdjęciu symbolizuje świeceniu się segmentu, a kolor szary symbolizuje wygaszenie segmentu.

Opis pliku top.vhd:

1.

2.

3.

4.

5.

6.

```
1  -- Mateusz Gabryel 181329 EiT 3
2  library IEEE;
3  use IEEE.STD_LOGIC_1164.ALL;
4  use IEEE.std_logic_signed.all;
5  use IEEE.std_logic_unsigned.all;
6  use IEEE.std_logic_arith.all;
7  use std.standard.all;
8
9  ENTITY top IS
10     PORT ( sw_i : in STD_LOGIC_VECTOR (3 downto 0);
11            led7_an_o : out STD_LOGIC_VECTOR (3 downto 0);
12            led7_seg_o : out STD_LOGIC_VECTOR (7 downto 0)
13            );
14 END top;
15
16 ARCHITECTURE Behavioral OF top IS
17
18 BEGIN
19     -- podanie 1 skutkuje wygaszeniem (AN3 -1, AN2 -1, AN1 -1)
20     -- podanie 0 skutkuje dzialaniem (AN0 -0)
21     led7_an_o(3) <= '1';
22     led7_an_o(2) <= '1';
23     led7_an_o(1) <= '1';
24     led7_an_o(0) <= '0';
25
26 PROCESS(sw_i)
27 variable a: integer:=0;
28 BEGIN
29
30 a:=0;
31     for b in 0 to 3 loop
32         if(sw_i(b)='1')then
33             a:=a+1;
34         END if;
35     END loop;
36
37    -- podanie 1 skutkuje wygaszeniem
38    -- podanie 0 skutkuje dzialaniem
39 if   (a=0) then led7_seg_o <="00000011"; -- cyfra: 0
40 elsif(a=1) then led7_seg_o <="10011111"; -- cyfra: 1
41 elsif(a=2) then led7_seg_o <="00100101"; -- cyfra: 2
42 elsif(a=3) then led7_seg_o <="00001101"; -- cyfra: 3
43            else led7_seg_o <="10011001"; -- cyfra: 4
44
45 END if;
46 END PROCESS;
47 END Behavioral;
```

1. Deklaracja użycia biblioteki IEEE oraz deklaracja użycia pakietów i ich składników.

2. Deklaracja przydziału portów:

- sw_i: słowo wprowadzane, wejściowe
- led7_an_o: pojedyncze wyświetlacze
- led7_seg_o: segmenty pojedynczego wyświetlacza

3. Podanie odpowiednich stanów na wyprowadzeniach AN0 - AN3 do ich aktywacji lub wygaszenia. W zadaniu wynik ma być wyświetlany na wyświetlaczu AN0, czyli podanie „0" uaktywnia go. Pozostałe wyświetlacze trzeba wygasić, podając sygnały „1".

4. Deklaracja zmiennej „a" typu integer, która posłuży jako licznik jedynek występujących w słowie wejściowym.

5. Utworzenie pętli odpowiadającej za zliczanie jedynek w słowie wejściowym.

6. Przypisanie odpowiednich wartości sygnałów na poszczególne segmenty pojedynczych wyświetlaczy w zależności od wartości zmiennej, która zliczy liczbę jedynek w słowie wejściowym.

Do prawidłowego działania układu na płytce FPGA jest potrzebny plik constraint iup2.xdc:

```
#virtual DIP Switches
set_property -dict { PACKAGE_PIN H14   IOSTANDARD LVCMOS33 } [get_ports { sw_i[0] }];
set_property -dict { PACKAGE_PIN G16   IOSTANDARD LVCMOS33 } [get_ports { sw_i[1] }];
set_property -dict { PACKAGE_PIN F16   IOSTANDARD LVCMOS33 } [get_ports { sw_i[2] }];
set_property -dict { PACKAGE_PIN D14   IOSTANDARD LVCMOS33 } [get_ports { sw_i[3] }];
#7-segment display
#segments (active low)
set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[7] }]; #IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[6] }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[5] }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[4] }]; #IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[3] }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[2] }]; #IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[1] }]; #IO_L4P_T0_D04_14 Sch=cg
set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { led7_seg_o[0] }]; #IO_L19N_T3_A21_VREF_15 Sch=dp
#digits (active low)
set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { led7_an_o[0] }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { led7_an_o[1] }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { led7_an_o[2] }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { led7_an_o[3] }]; #IO_L19P_T3_A22_15 Sch=an[3]
## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]
```

W pliku tb.vhd została wykonana symulacja poprzez podanie odpowiednich wartości słowa wejściowego:

```
-- Mateusz Gabryel 181329 EiT 3
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

ENTITY tb IS
END tb;

ARCHITECTURE Behavioral OF tb IS

COMPONENT top IS
    PORT ( sw_i : in STD_LOGIC_VECTOR (3 downto 0);
           led7_an_o : out STD_LOGIC_VECTOR (3 downto 0);
           led7_seg_o : out STD_LOGIC_VECTOR (7 downto 0)
           );
END COMPONENT top;

   signal sw_i : STD_LOGIC_VECTOR (3 downto 0);
   signal led7_an_o : STD_LOGIC_VECTOR (3 downto 0);
   signal led7_seg_o : STD_LOGIC_VECTOR (7 downto 0);

BEGIN
    uut: top PORT MAP (
          sw_i => sw_i,
          led7_seg_o => led7_seg_o,
          led7_an_o => led7_an_o
        );

tb: PROCESS
   BEGIN
        sw_i <= "0000"; -- I kombinacja - 0
        wait for 100 ms;
        sw_i <= "0001"; -- II kombinacja - 1
        wait for 100 ms;
        sw_i <= "0010"; -- III kombinacja - 2
        wait for 100 ms;
        sw_i <= "0011"; -- IV kombinacja - 3
        wait for 100 ms;
        sw_i <= "0100"; -- V kombinacja - 4
        wait for 100 ms;
        sw_i <= "0101"; -- VI kombinacja - 5
        wait for 100 ms;
        sw_i <= "0110"; -- VII kombinacja - 6
        wait for 100 ms;
        sw_i <= "0111"; -- VIII kombinacja - 7
        wait for 100 ms;
        sw_i <= "1000"; -- IX kombinacja - 8
        wait for 100 ms;
        sw_i <= "1001"; -- X kombinacja - 9
        wait for 100 ms;
        sw_i <= "1010"; -- XI kombinacja - 10
        wait for 100 ms;
        sw_i <= "1011"; -- XII kombinacja - 11
        wait for 100 ms;
        sw_i <= "1100"; -- XIII kombinacja - 12
        wait for 100 ms;
        sw_i <= "1101"; -- XIV kombinacja - 13
        wait for 100 ms;
        sw_i <= "1110"; -- XV kombinacja - 14
        wait for 100 ms;
        sw_i <= "1111"; -- XVI kombinacja - 15
        wait for 100 ms;
      wait;
   END PROCESS;
END Behavioral;
```

Wyniki symulacji: