Programowalne układy cyfrowe **2014/2015**

Prowadzący: mgr. Tomaszewski Grzegorz poniedziałek, 14:00

Data oddania: _____ Ocena: _____

Witold Olechowski 127517
Tomasz Marecik 127374

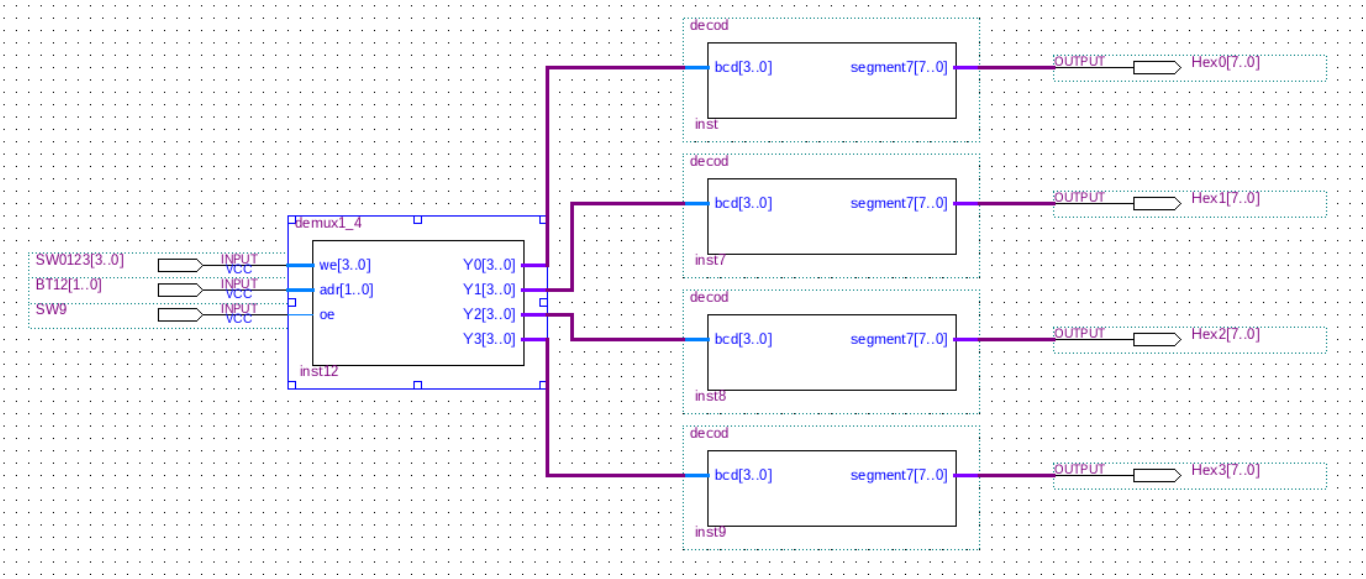
Zadanie 3,4: Projektowanie prostych układów kombinacyjnych z użyciem języka VHDL

1. Cel:

Używając języka VHDL zrealizować:

— transkoder kodu BCD 8421 na kod wskaźnika 7-segmentowego
— demultiplekser umożliwiający wybór 1 z 4 wyświetlaczy LED

2. Pierwszy wariant realizacji zadania



Rysunek 1. Schemat blokowy układu z transkodowaniem po demultipleksacji

2.1. Listing kodu źródłowego dla demultipleksera:

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

entity demux1_4 is
port (
```

```
        -- wejscia
        we      : in std_logic_vector(3 downto 0);
        adr     : in std_logic_vector(1 downto 0);
        oe      : in std_logic ;
        -- wyjscia
        Y0,Y1,Y2,Y3 : out std_logic_vector (3 downto 0)
    );
end demux1_4;

architecture Behavioral of demux1_4 is

begin
process(oe,adr) is
begin
--gdy '1' na wejsciu oe, demltiplexera
-- we zostaje przepisane na Yx w zaleznosci
-- od wejscia adresowego
-- '0' wygasza w 4 segmenty
if oe = '1' then
        if adr="00" then
        Y0<=we; Y1<="1111"; Y2<="1111"; Y3<="1111";
        elsif adr="01" then
        Y0<="1111"; Y1<=we; Y2<="1111"; Y3<="1111";
        elsif adr="10" then
        Y0<="1111"; Y1<="1111"; Y2<=we; Y3<="1111";
        elsif adr="11" then
        Y0<="1111"; Y1<="1111"; Y2<="1111"; Y3<=we;
end if;
elsif oe='0' then
Y0<="1111"; Y1<="1111"; Y2<="1111"; Y3<="1111";
end if;
end process;
end Behavioral;
```

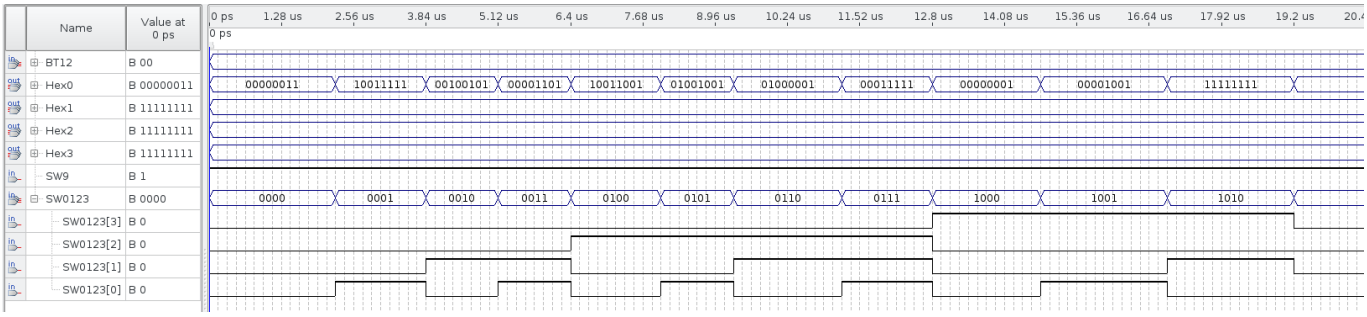
2.2. Listing kodu źródłowego dla dekodera BCD:

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

entity decod is
port (
        --wejscie BCD.
        bcd : in std_logic_vector(3 downto 0);
        -- wyjscie dekodera 8 bit.
        segment7 : out std_logic_vector(7 downto 0)
    );
end decod;

architecture Behavioral of decod is

begin
```

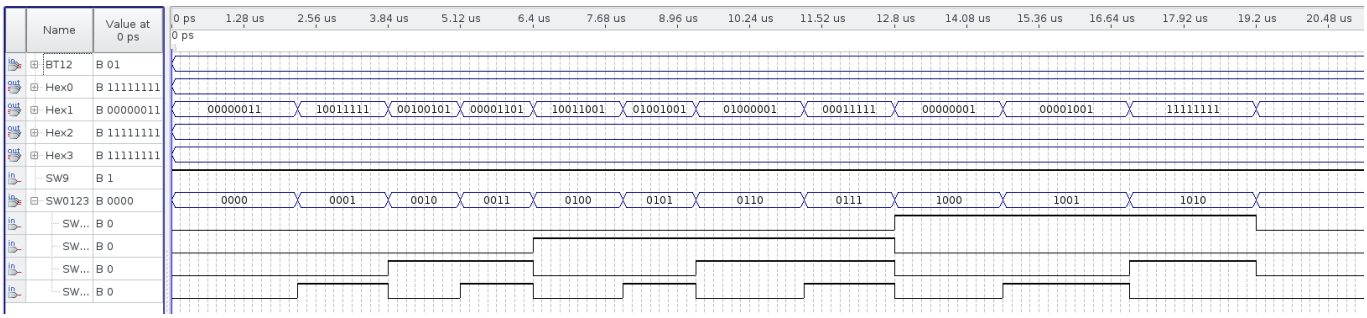
```
process (bcd)
BEGIN
case  bcd is
        when "0000"=> segment7 <="00000011";   -- '0'
        when "0001"=> segment7 <="10011111";   -- '1'
        when "0010"=> segment7 <="00100101";   -- '2'
        when "0011"=> segment7 <="00001101";   -- '3'
        when "0100"=> segment7 <="10011001";   -- '4'
        when "0101"=> segment7 <="01001001";   -- '5'
        when "0110"=> segment7 <="01000001";   -- '6'
        when "0111"=> segment7 <="00011111";   -- '7'
        when "1000"=> segment7 <="00000001";   -- '8'
        when "1001"=> segment7 <="00001001";   -- '9'
        -- stany wyzsze od 9 wygaczaja segement
        when others=> segment7 <="11111111";
end case;
end process;
end Behavioral;
```

2.3. Przebiegi czasowe układu:

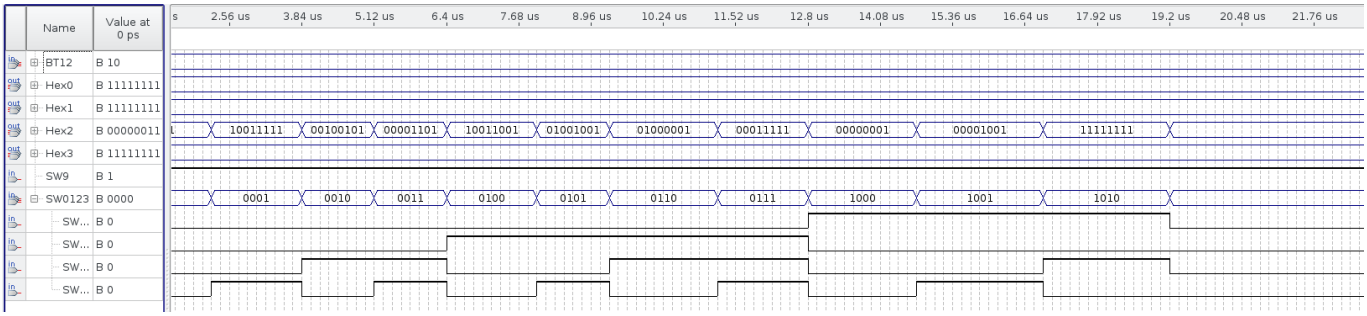
Na rysunkach: 2, 3, 4, 5, przedstawione zostały przebiegi czasowe, na wejscie bcd ustawiono stany od 0 do 10. Symulacje kolejno przedstawiaja działanie demultipleksera dla adresowania(BT12) 0-3. Wygaszenia wszystkich segmentów dokonuje sie prze kasowanie bitu oe(SW9) na wejściu demultipleksera [rys. 6].



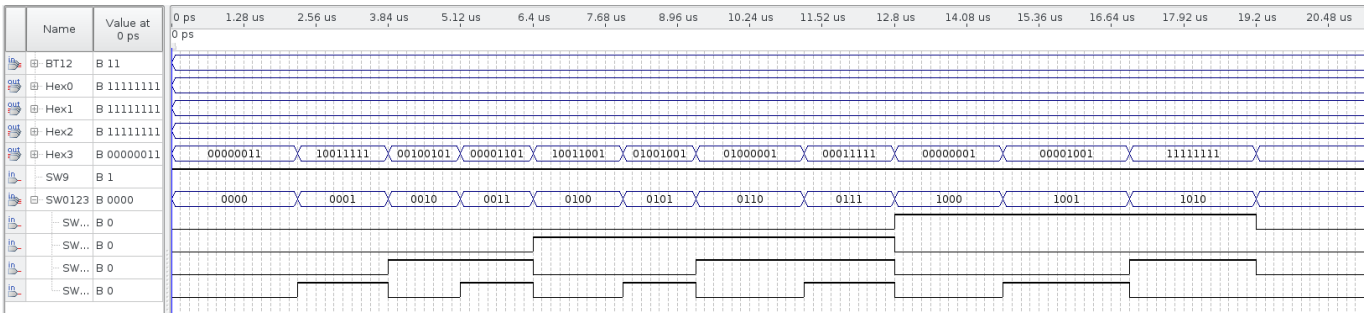
Rysunek 2. SW9='1', BT12='00'



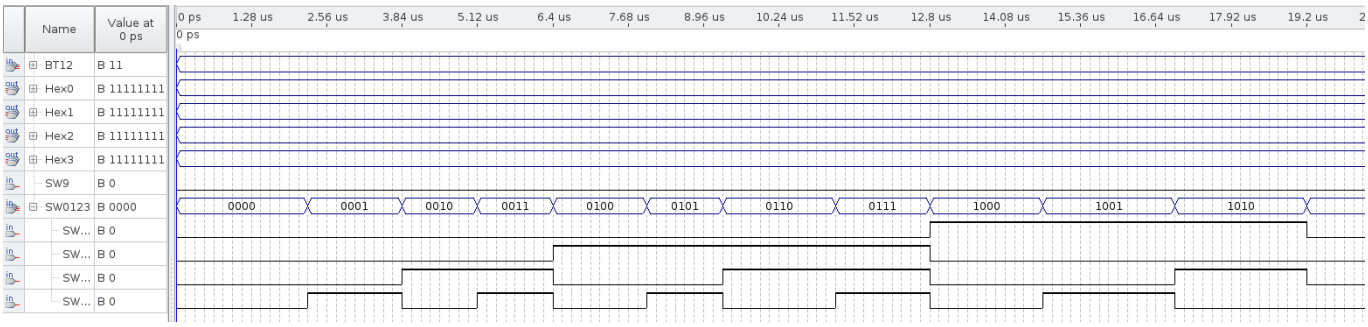
Rysunek 3. SW9='1', BT12='01'



Rysunek 4. SW9=’1’, BT12=’10’



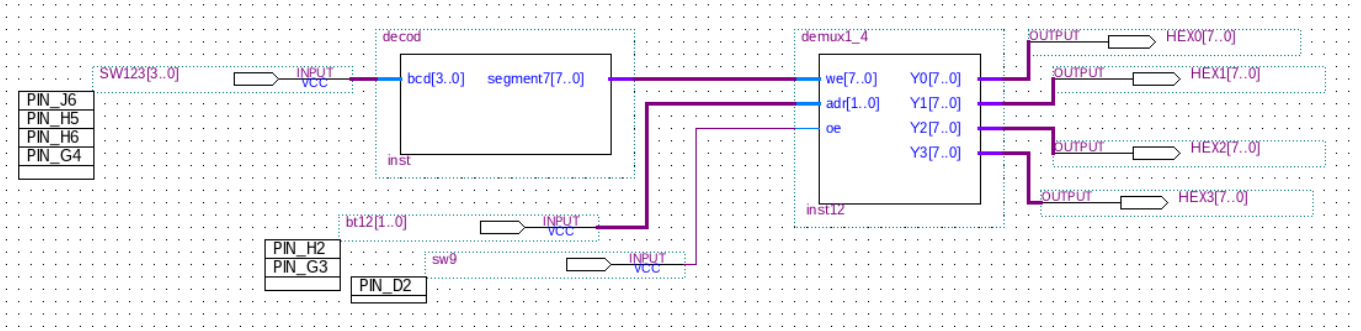
Rysunek 5. SW9=’1’, BT12=’11’



Rysunek 6. SW9=’0’, BT12=’11’

3. Drugi wariant realizacji zadania z transkodowaniem

Logicznie działanie układów jest identyczne, w porównaniu do układu 2 różni sie modyfikacją demultpleksera, na we/wy 8-bitowe oraz usunięciu 3 dekoderów. Symulacje czasowe są tożsame dla wariantu 2-giego z 1-szym i zostaly pominięte.



Rysunek 7. Schemat blokowy układu z transkodowaniem przed demultipleksacją

3.1. Listing kodu źródłowego dla demultipleksera po modyfikacji:

Kod został omówiny na podstawie listingu: 2.1.

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

entity demux1_4 is
port (
        we      : in std_logic_vector(7 downto 0);
        adr     : in std_logic_vector(1 downto 0);
        oe      : in std_logic ;
        Y0,Y1,Y2,Y3 : out std_logic_vector (7 downto 0)
     );
end demux1_4;

architecture Behavioral of demux1_4 is

begin
process(oe,adr) is
begin
if oe = '1' then
        if adr="00" then
        Y0<=we; Y1<="11111111"; Y2<="11111111"; Y3<="11111111";
        elsif adr="01" then
        Y0<="11111111"; Y1<=we; Y2<="11111111"; Y3<="11111111";
        elsif adr="10" then
        Y0<="11111111"; Y1<="11111111"; Y2<=we; Y3<="11111111";
        elsif adr="11" then
        Y0<="11111111"; Y1<="11111111"; Y2<="11111111"; Y3<=we;
end if;
elsif oe='0' then
Y0<="11111111"; Y1<="11111111"; Y2<="11111111"; Y3<="11111111";
end if;
end process;
end Behavioral;
```

4. Wnioski

Cele wykonywanego ćwiczenia zostały zrealizowane w sposób poprawny, zaprojektowany układ transkodera kodu BCD odzwierciedla jego właściwą zasadę działania co potwierdzają przeprowadzone symulacje, zarówno dla pierwszego jak i drugiego wariantu realizowanego układu. Podczas realizacji założeń projektowych doskonaliliśmy również posługiwanie się językiem programowania VHDL, ponadto zaprojektowane układy poddawaliśmy testom na poprawność działania.

Literatura

[1] John Wiley and Sons Publishers. *Digital Design*, University of California, Riverside, 2007