HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY – VNU HCMC
OFFICE FOR INTERNATIONAL STUDY PROGRAM
FACULTY OF ELECTRICAL AND ELECTRONIC ENGINEERING

*

DIGITAL SYSTEMS (LAB)
EXPERIMENTAL REPORT (Lab 1)

Lecturer : **Mr. Nguyễn Tuấn Hùng**
Subject : **Digital Systems**
Class : **TT06**
Name : **Lương Triển Thắng**
Student ID : **2051194**

Ho Chi Minh City, 31st May, 2022

Contents

I Laboratory 1:

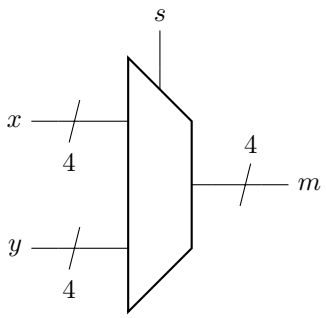
Get started with FPGA **2**

1. Know how to program n-bit wide 2-to-1 multiplexer 2
   a. Code 2
   b. Waveform 3
   c. Result of RTL viewer 3
2. Known how to program n-bit wide 4-to-1 multiplexer 4
   a. Code 4
   b. Waveform 5
   c. Result of RTL viewer 5
3. Known how to interface with 7-segment LED 6
   a. Method 1: with MUX 6
   b. Method 2: without MUX 10
   c. Waveform 12
4. Known how to interface 7-segment LED and multiplexer 13
   a. Code 13
   b. Waveform 16
   c. Result of RTL viewer 17
5. Known how to interface 7-segment LED and multiplexer 19
   a. Method 1: with MUX, comparator and “Circuit A” 19
   b. Method 2: using conditions in VHDL 24
   c. Waveform 26

I Laboratory 1

Get started with FPGA

1. Know how to program n-bit wide 2-to-1 multiplexer



Circuit diagram

a. Code

MUX.vhdl

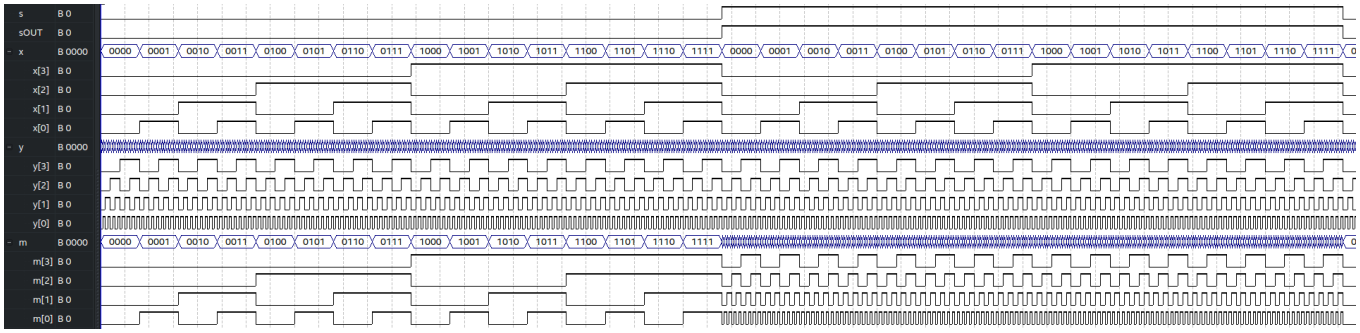
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY MUX IS
        PORT (
                a : IN std_logic;
                sel : IN std_logic;
                b : IN std_logic;
                z : OUT std_logic
        );
END ENTITY;

ARCHITECTURE arch OF MUX IS
BEGIN
        z <= (NOT(sel) AND a) OR (sel AND b);
END arch;
```

Exc1.vhdl

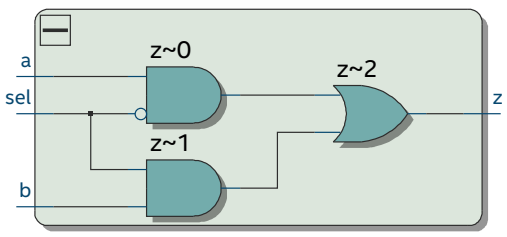
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY Exc1 IS
        PORT (
                x : IN std_logic_vector(3 DOWNTO 0);
                y : IN std_logic_vector(3 DOWNTO 0);
                s : IN std_logic;
                sOUT : OUT std_logic;
                m : OUT std_logic_vector(3 DOWNTO 0)
        );
END ENTITY;

ARCHITECTURE behave OF Exc1 IS
        COMPONENT MUX IS
                PORT (
                        a : IN std_logic;
```

```
                        sel : IN std_logic;
                        b : IN std_logic;
                        z : OUT std_logic
                );
        END COMPONENT;

BEGIN
        sOUT <= s;
        gen : FOR i IN 3 DOWNTO 0 GENERATE
                MUX4 : MUX
                PORT MAP(
                        a => x(i),
                        b => y(i),
                        sel => s,
                        z => m(i)
                );
        END GENERATE;

END ARCHITECTURE;
```

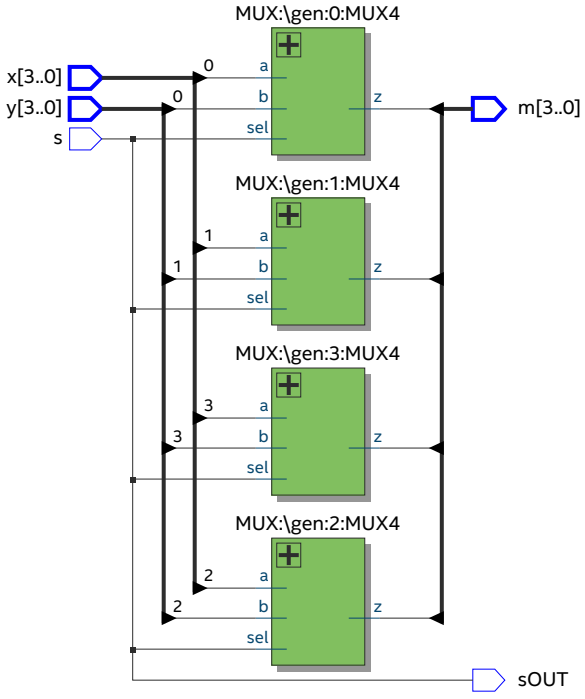
b. Waveform



c. Result of RTL viewer

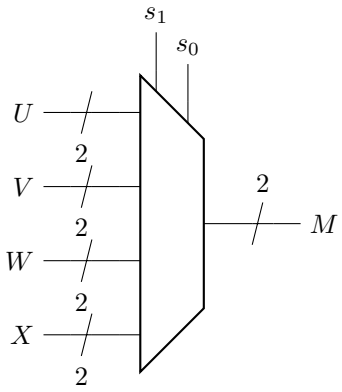


2-to-1 multiplexer



4-bit 2-to-1 multiplexer

2. Known how to program n-bit wide 4-to-1 multiplexer



Circuit diagram

a. Code

Four2OneMUX.vhdl

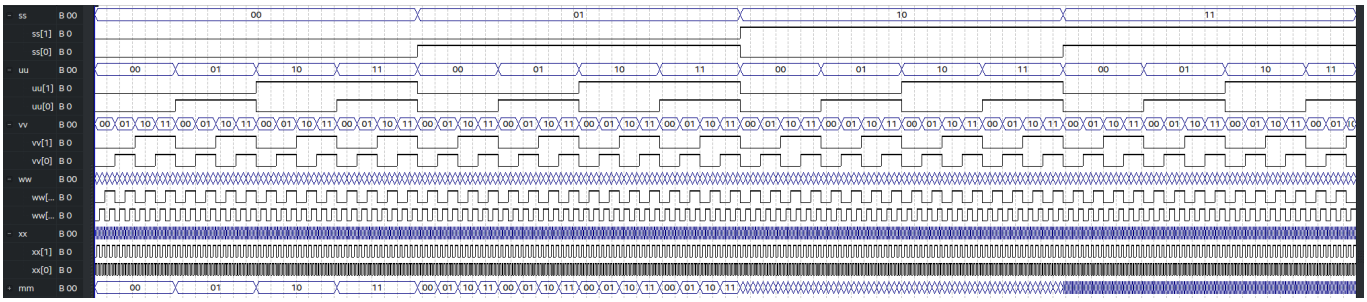
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY Four2OneMUX IS
        PORT (
                s : IN std_logic_vector(1 DOWNTO 0);
                u : IN std_logic;
                v : IN std_logic;
                w : IN std_logic;
                x : IN std_logic;
                m : OUT std_logic
        );
END ENTITY;

ARCHITECTURE behave OF Four2OneMUX IS
BEGIN
        WITH s SELECT
        m <= u WHEN "00",
             v WHEN "01",
             w WHEN "10",
             x WHEN "11";

END ARCHITECTURE;
```

Exc1.vhdl

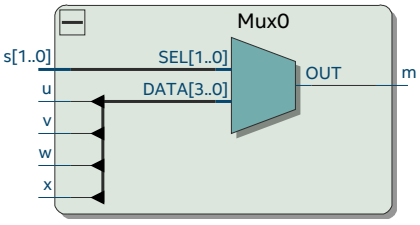
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY Exc2 IS
        PORT (
                ss : IN std_logic_vector(1 DOWNTO 0);
                uu : IN std_logic_vector(1 DOWNTO 0);
                vv : IN std_logic_vector(1 DOWNTO 0);
                ww : IN std_logic_vector(1 DOWNTO 0);
                xx : IN std_logic_vector(1 DOWNTO 0);
                mm : OUT std_logic_vector(1 DOWNTO 0)
        );
END ENTITY;

ARCHITECTURE behave OF Exc2 IS
```

```
        COMPONENT Four2OneMUX IS
                PORT (
                        s : IN std_logic_vector(1 DOWNTO 0);
                        u : IN std_logic;
                        v : IN std_logic;
                        w : IN std_logic;
                        x : IN std_logic;
                        m : OUT std_logic
                );
        END COMPONENT;

BEGIN
        gen : FOR i IN 1 DOWNTO 0 GENERATE
                MUX2 : Four2OneMUX
                PORT MAP(
                        s => ss,
                        u => uu(i),
                        v => vv(i),
                        w => ww(i),
                        x => xx(i),
                        m => mm(i)
                );
        END GENERATE;

END ARCHITECTURE;
```

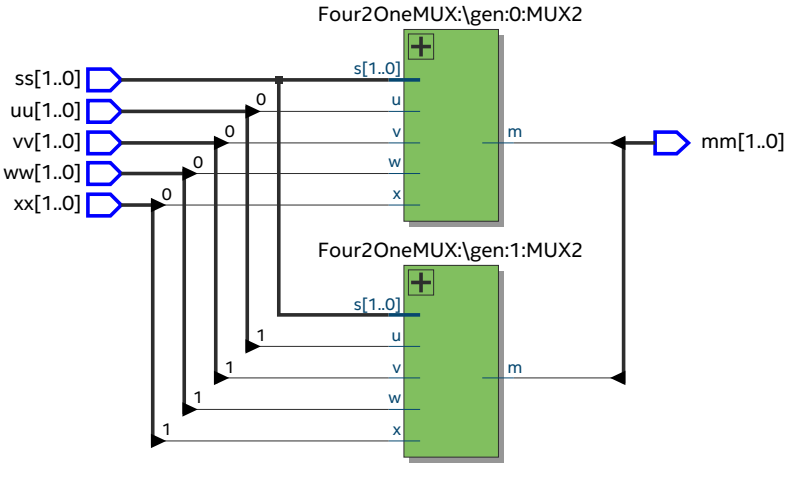
b. Waveform



c. Result of RTL viewer



4-to-1 multiplexer



Top level

3. Known how to interface with 7-segment LED

s_1	s_0	LEDs
0	0	
0	1	
1	0	
1	1	

a. Method 1: with MUX

Code

DE10.vhdl

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

ENTITY DE10 IS
        PORT (
                c : IN std_logic_vector(1 DOWNTO 0);
                HEXn : OUT std_logic_vector(0 TO 6)
        );
END DE10;

ARCHITECTURE behavior OF DE10 IS
        SIGNAL HEX : std_logic_vector(0 TO 6);
BEGIN
        HEXn <= NOT(HEX);
        WITH c SELECT
        HEX <= "0111101" WHEN "00",
               "1001111" WHEN "01",
               "0110000" WHEN "10",
               "1111110" WHEN "11",
               "0000000" WHEN OTHERS;
END behavior;
```

Four2OneMUX.vhdl

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

ENTITY Four2OneMUX IS
        PORT (
                s : IN std_logic_vector(1 DOWNTO 0);
                u : IN std_logic;
                v : IN std_logic;
                w : IN std_logic;
                x : IN std_logic;
                m : OUT std_logic
        );
END ENTITY;

ARCHITECTURE behave OF Four2OneMUX IS
BEGIN
        WITH s SELECT
        m <= u WHEN "00",
```

```
            v WHEN "01",
            w WHEN "10",
            x WHEN "11";

END ARCHITECTURE;
```

TwoBitFour2OneMUX.vhdl

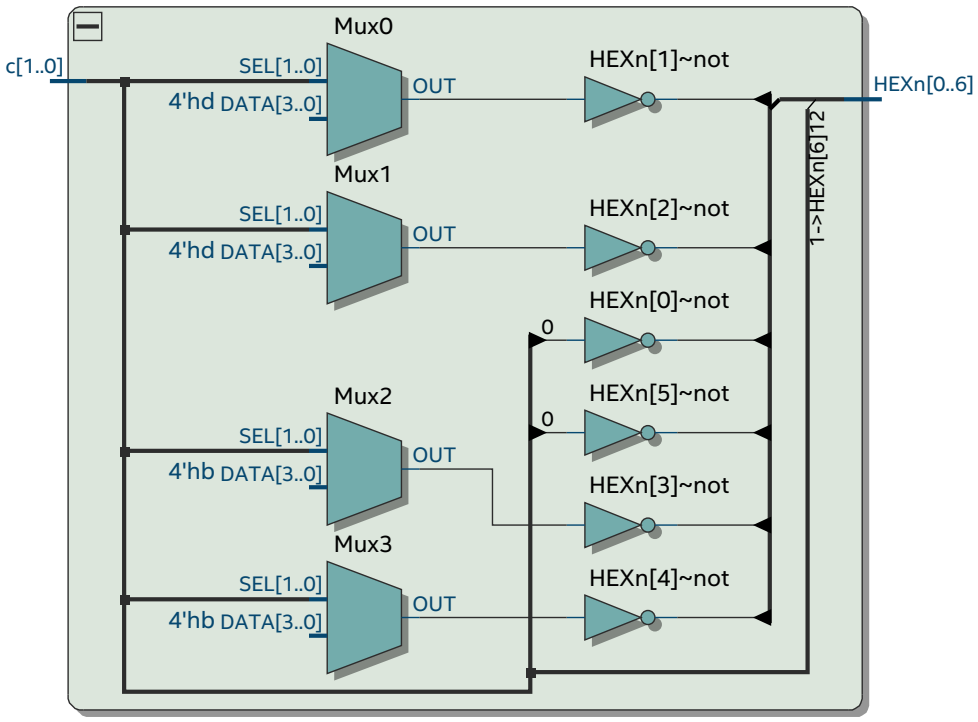
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY TwoBitFour2OneMUX IS
        PORT (
                ss : IN std_logic_vector(1 DOWNTO 0);
                uu : IN std_logic_vector(1 DOWNTO 0);
                vv : IN std_logic_vector(1 DOWNTO 0);
                ww : IN std_logic_vector(1 DOWNTO 0);
                xx : IN std_logic_vector(1 DOWNTO 0);
                mm : OUT std_logic_vector(1 DOWNTO 0)
        );
END ENTITY;

ARCHITECTURE behave OF TwoBitFour2OneMUX IS
        COMPONENT Four2OneMUX IS
                PORT (
                        s : IN std_logic_vector(1 DOWNTO 0);
                        u : IN std_logic;
                        v : IN std_logic;
                        w : IN std_logic;
                        x : IN std_logic;
                        m : OUT std_logic
                );
        END COMPONENT;

BEGIN
        gen : FOR i IN 1 DOWNTO 0 GENERATE
                MUX2 : Four2OneMUX
                PORT MAP(
                        s => ss,
                        u => uu(i),
                        v => vv(i),
                        w => ww(i),
                        x => xx(i),
                        m => mm(i)
                );
        END GENERATE;

END ARCHITECTURE;
```

Exc1.vhdl

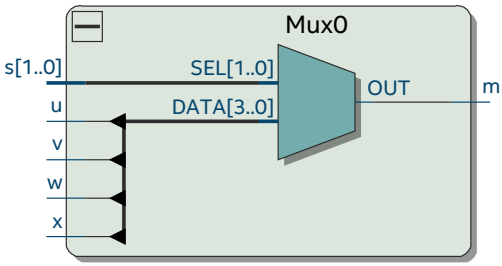
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY Exc3 IS
        PORT (
                SW : IN std_logic_vector(9 DOWNTO 0);
                HEX0 : OUT std_logic_vector(0 TO 6);
                HEX1 : OUT std_logic_vector(0 TO 6);
                HEX2 : OUT std_logic_vector(0 TO 6);
                HEX3 : OUT std_logic_vector(0 TO 6)
        );
END ENTITY;
```

```
ARCHITECTURE behave OF Exc3 IS
        SIGNAL sel3, sel2, sel1, sel0, selct : std_logic_vector(1 DOWNTO 0);
        SIGNAL outHEX3, outHEX2, outHEX1, outHEX0 : std_logic_vector(1 DOWNTO 0);

        COMPONENT DE10 IS
                PORT (
                        c : IN std_logic_vector(1 DOWNTO 0);
                        HEXn : OUT std_logic_vector(0 TO 6)
                );
        END COMPONENT;

        COMPONENT TwoBitFour2OneMUX IS
                PORT (
                        ss : IN std_logic_vector(1 DOWNTO 0);
                        uu : IN std_logic_vector(1 DOWNTO 0);
                        vv : IN std_logic_vector(1 DOWNTO 0);
                        ww : IN std_logic_vector(1 DOWNTO 0);
                        xx : IN std_logic_vector(1 DOWNTO 0);
                        mm : OUT std_logic_vector(1 DOWNTO 0)
                );
        END COMPONENT;

BEGIN
        selct <= SW(9 DOWNTO 8);
        sel3 <= SW(7 DOWNTO 6);
        sel2 <= SW(5 DOWNTO 4);
        sel1 <= SW(3 DOWNTO 2);
        sel0 <= SW(1 DOWNTO 0);
        MUX3 : TwoBitFour2OneMUX
        PORT MAP(
                uu => sel3, vv => sel2, ww => sel1, xx => sel0, ss => selct, mm => outHEX3
        );
        MUX2 : TwoBitFour2OneMUX
        PORT MAP(
                uu => sel2, vv => sel1, ww => sel0, xx => sel3, ss => selct, mm => outHEX2
        );
        MUX1 : TwoBitFour2OneMUX
        PORT MAP(
                uu => sel1, vv => sel0, ww => sel3, xx => sel2, ss => selct, mm => outHEX1
        );
        MUX0 : TwoBitFour2OneMUX
        PORT MAP(
                uu => sel0, vv => sel3, ww => sel2, xx => sel1, ss => selct, mm => outHEX0
        );

        HEX03 : DE10
        PORT MAP(c => outHEX3, HEXn => HEX3);
        HEX02 : DE10
        PORT MAP(c => outHEX2, HEXn => HEX2);
        HEX01 : DE10
        PORT MAP(c => outHEX1, HEXn => HEX1);
HEX00 : DE10
PORT MAP(c => outHEX0, HEXn => HEX0);

END ARCHITECTURE;
```

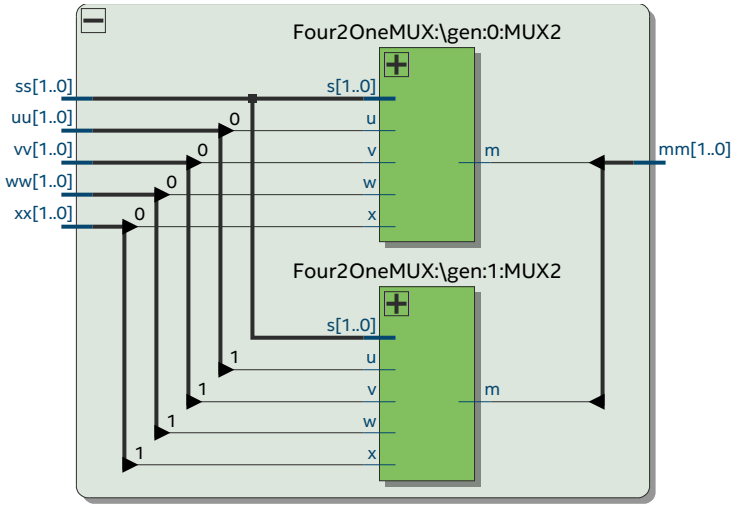
Result of RTL viewer



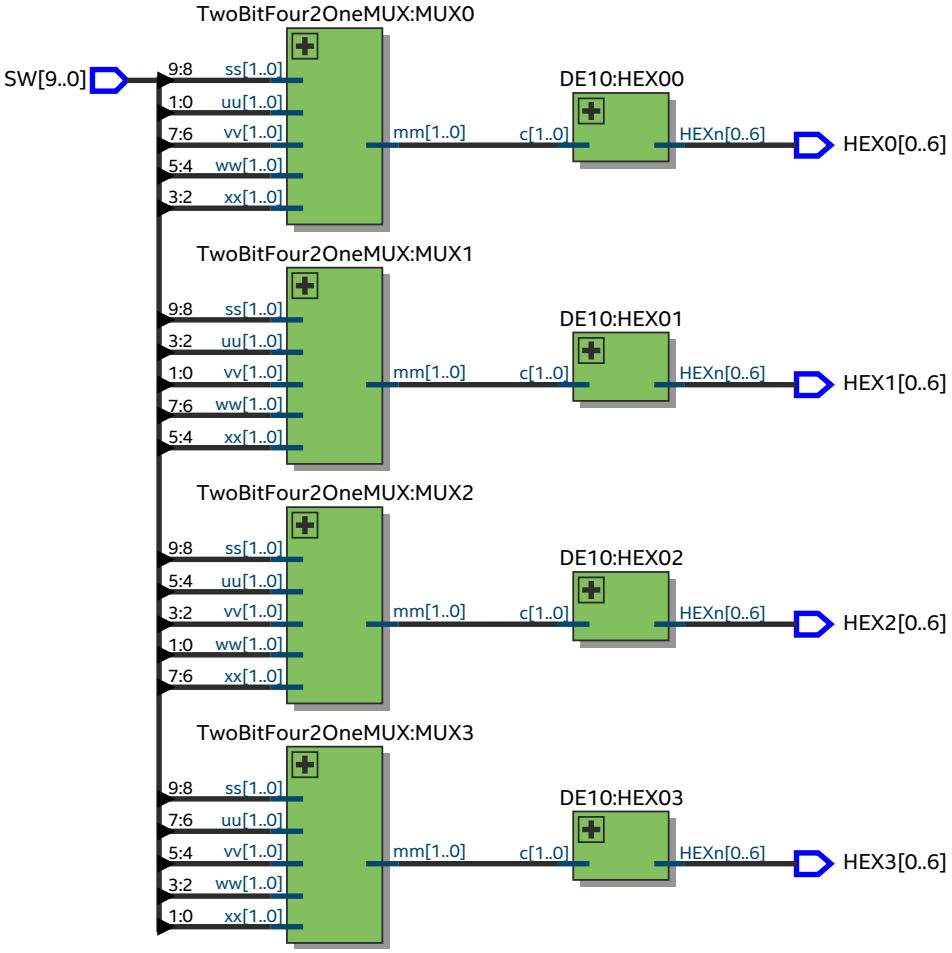
"dE10" 7-segment decoder



4-to-1 multiplexer



2-bit 4-to-1 multiplexer



Top level

b. Method 2: without MUX

Code

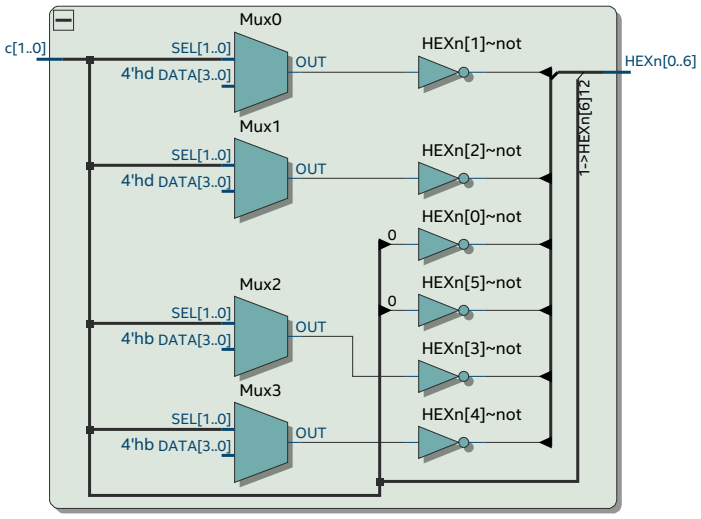
DE10.vhdl

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

ENTITY DE10 IS
        PORT (
                c : IN std_logic_vector(1 DOWNTO 0);
                HEXn : OUT std_logic_vector(0 TO 6)
        );
END DE10;

ARCHITECTURE behavior OF DE10 IS
        SIGNAL HEX : std_logic_vector(0 TO 6);
BEGIN
        HEXn <= NOT(HEX);
        WITH c SELECT
        HEX <= "0111101" WHEN "00",
               "1001111" WHEN "01",
               "0110000" WHEN "10",
               "1111110" WHEN "11",
               "0000000" WHEN OTHERS;
END behavior;
```

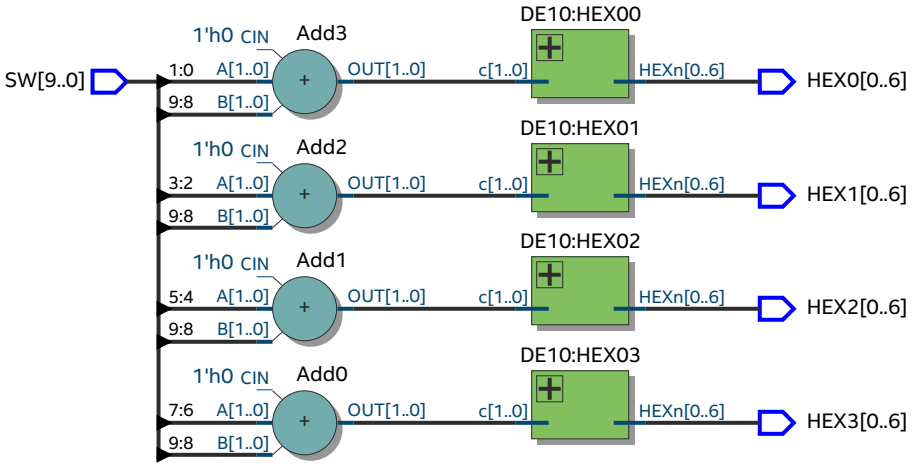
Exc1.vhdl

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

ENTITY Exc3 IS
        PORT (
                SW : IN std_logic_vector(9 DOWNTO 0);
                HEX0 : OUT std_logic_vector(0 TO 6);
                HEX1 : OUT std_logic_vector(0 TO 6);
                HEX2 : OUT std_logic_vector(0 TO 6);
                HEX3 : OUT std_logic_vector(0 TO 6)
        );
END ENTITY;

ARCHITECTURE behave OF Exc3 IS
        SIGNAL sel3, sel2, sel1, sel0 : std_logic_vector(1 DOWNTO 0);

        COMPONENT DE10 IS
                PORT (
                        c : IN std_logic_vector(1 DOWNTO 0);
                        HEXn : OUT std_logic_vector(0 TO 6)
                );
        END COMPONENT;

BEGIN
        sel3 <= std_logic_vector(unsigned(SW(7 DOWNTO 6)) + unsigned(SW(9 DOWNTO 8)));
        sel2 <= std_logic_vector(unsigned(SW(5 DOWNTO 4)) + unsigned(SW(9 DOWNTO 8)));
        sel1 <= std_logic_vector(unsigned(SW(3 DOWNTO 2)) + unsigned(SW(9 DOWNTO 8)));
        sel0 <= std_logic_vector(unsigned(SW(1 DOWNTO 0)) + unsigned(SW(9 DOWNTO 8)));

        HEX03 : DE10 PORT MAP(c => sel3, HEXn => HEX3);
        HEX02 : DE10 PORT MAP(c => sel2, HEXn => HEX2);
        HEX01 : DE10 PORT MAP(c => sel1, HEXn => HEX1);
        HEX00 : DE10 PORT MAP(c => sel0, HEXn => HEX0);

END ARCHITECTURE;
```

Result of RTL viewer

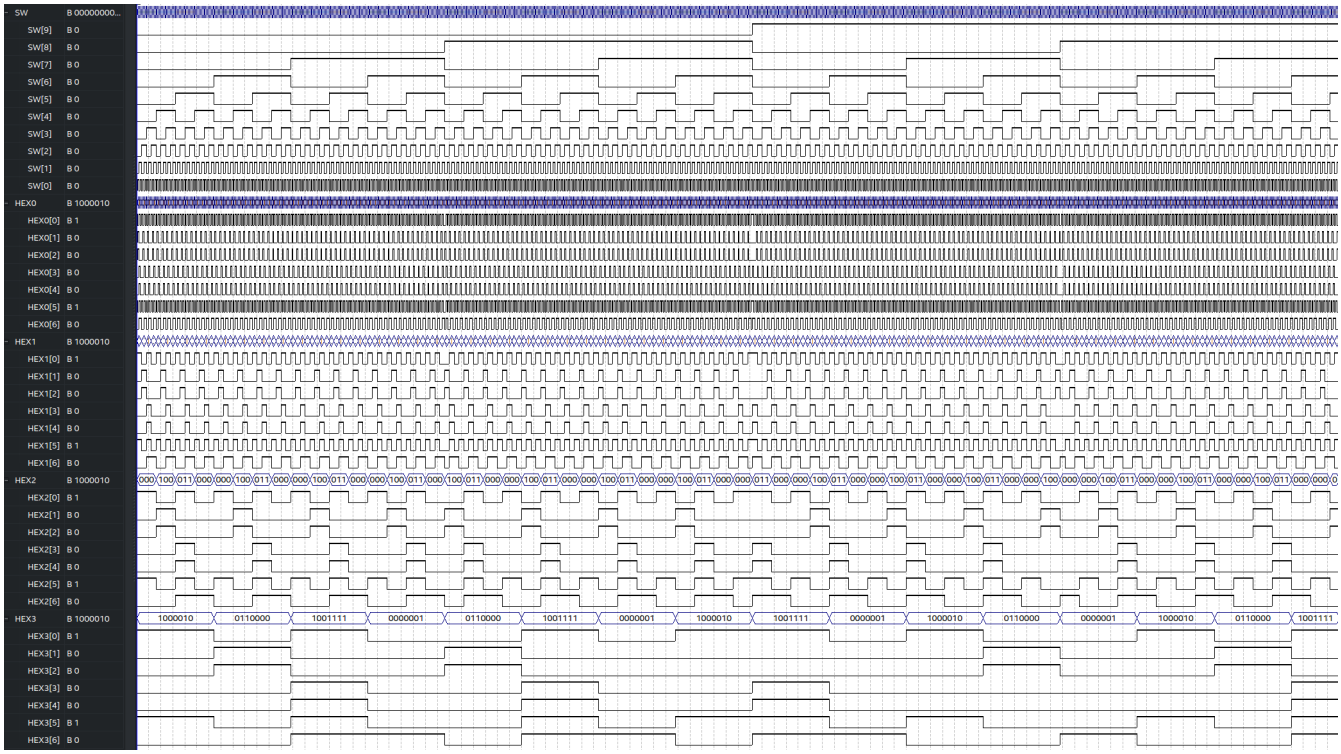


"dE10" 7-segment decoder



Top level

c. Waveform



4. Known how to interface 7-segment LED and multiplexer

s_2	s_1	s_0	LEDs
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

a. Code

HELLO.vhdl

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

ENTITY HELLO IS
        PORT (
                c : IN std_logic_vector(2 DOWNTO 0);
                HEXn : OUT std_logic_vector(0 TO 6)
        );
END HELLO;

ARCHITECTURE behavior OF HELLO IS
        SIGNAL HEX : std_logic_vector(0 TO 6);
BEGIN
        HEXn <= NOT(HEX);
        WITH c SELECT
        HEX <= "0110111" WHEN "000",
               "1001111" WHEN "001",
               "0001110" WHEN "010",
               "1111110" WHEN "011",
               "0000000" WHEN OTHERS;
END behavior;
```

Six2OneMUX.vhdl

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

ENTITY Six2OneMUX IS
        PORT (
                s : IN std_logic_vector(2 DOWNTO 0);
                u : IN std_logic;
                v : IN std_logic;
                w : IN std_logic;
                x : IN std_logic;
```

```
                y : IN std_logic;
                z : IN std_logic;
                m : OUT std_logic
        );
END ENTITY;

ARCHITECTURE behave OF Six2OneMUX IS
BEGIN
        WITH s SELECT
        m <= u WHEN "000",
             v WHEN "001",
             w WHEN "010",
             x WHEN "011",
             y WHEN "100",
             z WHEN "101",
             '1' WHEN OTHERS;

END ARCHITECTURE;
```

ThreeBitSix2OneMUX.vhdl

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

ENTITY ThreeBitSix2OneMUX IS
        PORT (
                ss : IN std_logic_vector(2 DOWNTO 0);
                uu : IN std_logic_vector(2 DOWNTO 0);
                vv : IN std_logic_vector(2 DOWNTO 0);
                ww : IN std_logic_vector(2 DOWNTO 0);
                xx : IN std_logic_vector(2 DOWNTO 0);
                yy : IN std_logic_vector(2 DOWNTO 0);
                zz : IN std_logic_vector(2 DOWNTO 0);
                mm : OUT std_logic_vector(2 DOWNTO 0)
        );
END ENTITY;

ARCHITECTURE behave OF ThreeBitSix2OneMUX IS
        COMPONENT Six2OneMUX IS
                PORT (
                        s : IN std_logic_vector(2 DOWNTO 0);
                        u : IN std_logic;
                        v : IN std_logic;
                        w : IN std_logic;
                        x : IN std_logic;
                        y : IN std_logic;
                        z : IN std_logic;
                        m : OUT std_logic
                );
        END COMPONENT;

BEGIN
        gen : FOR i IN 2 DOWNTO 0 GENERATE
                MUX2 : Six2OneMUX
                PORT MAP(
                        s => ss,
                        u => uu(i),
                        v => vv(i),
                        w => ww(i),
                        x => xx(i),
                        y => yy(i),
                        z => zz(i),
                        m => mm(i)
```

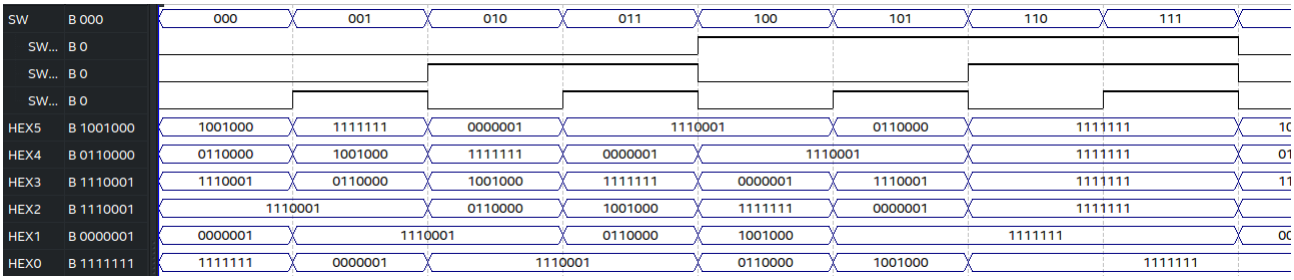
```
                );
        END GENERATE;

END ARCHITECTURE;
```

HELLO.vhdl

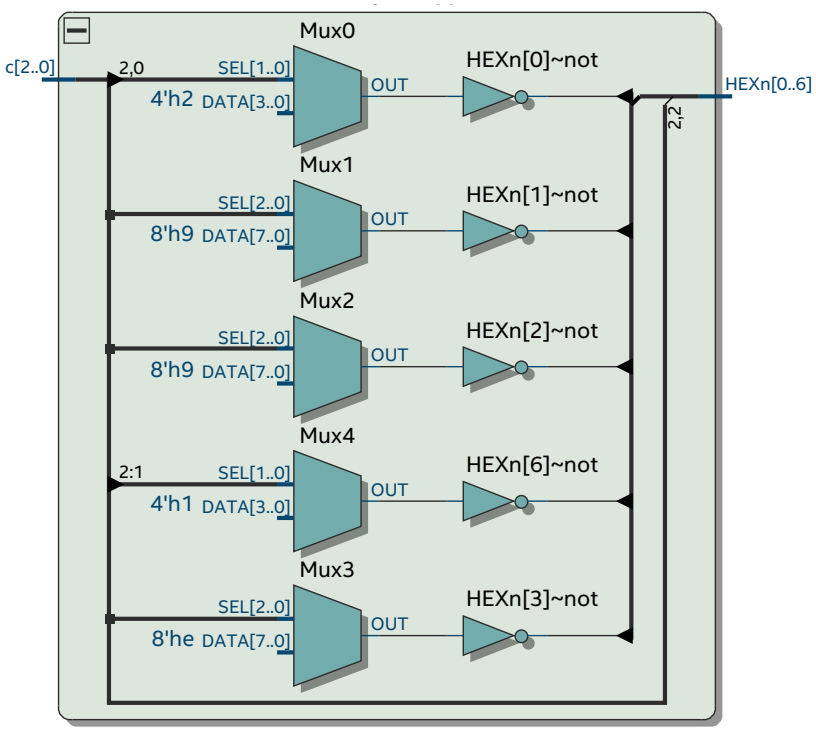
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY Exc4 IS
        PORT (
                SW : IN std_logic_vector(2 DOWNTO 0);
                HEX0 : OUT std_logic_vector(0 TO 6);
                HEX1 : OUT std_logic_vector(0 TO 6);
                HEX2 : OUT std_logic_vector(0 TO 6);
                HEX3 : OUT std_logic_vector(0 TO 6);
                HEX4 : OUT std_logic_vector(0 TO 6);
                HEX5 : OUT std_logic_vector(0 TO 6)
        );
END ENTITY;

ARCHITECTURE behave OF Exc4 IS
        SIGNAL sel5, sel4, sel3, sel2, sel1, sel0, selct : std_logic_vector(2 DOWNTO 0);
        SIGNAL outHEX5, outHEX4, outHEX3, outHEX2, outHEX1, outHEX0 : std_logic_vector(2 DOWNTO 0);

        COMPONENT HELLO IS
                PORT (
                        c : IN std_logic_vector(2 DOWNTO 0);
                        HEXn : OUT std_logic_vector(0 TO 6)
                );
        END COMPONENT;

        COMPONENT ThreeBitSix2OneMUX IS
                PORT (
                        ss : IN std_logic_vector(2 DOWNTO 0);
                        uu : IN std_logic_vector(2 DOWNTO 0);
                        vv : IN std_logic_vector(2 DOWNTO 0);
                        ww : IN std_logic_vector(2 DOWNTO 0);
                        xx : IN std_logic_vector(2 DOWNTO 0);
                        yy : IN std_logic_vector(2 DOWNTO 0);
                        zz : IN std_logic_vector(2 DOWNTO 0);
                        mm : OUT std_logic_vector(2 DOWNTO 0)
                );
        END COMPONENT;

BEGIN
        selct <= SW(2 DOWNTO 0);

        -- HELLO
        sel5 <= "000";
        sel4 <= "001";
        sel3 <= "010";
        sel2 <= "010";
        sel1 <= "011";
        sel0 <= "100";

        MUX5 : ThreeBitSix2OneMUX
        PORT MAP(
                uu => sel5, vv => sel0, ww => sel1, xx => sel2, yy => sel3,
                zz => sel4, ss => selct, mm => outHEX5
        );
        MUX4 : ThreeBitSix2OneMUX
```

```
        PORT MAP(
                uu => sel4, vv => sel5, ww => sel0, xx => sel1, yy => sel2,
                zz => sel3, ss => selct, mm => outHEX4
        );
        MUX3 : ThreeBitSix2OneMUX
        PORT MAP(
                uu => sel3, vv => sel4, ww => sel5, xx => sel0, yy => sel1,
                zz => sel2, ss => selct, mm => outHEX3
        );
        MUX2 : ThreeBitSix2OneMUX
        PORT MAP(
                uu => sel2, vv => sel3, ww => sel4, xx => sel5, yy => sel0,
                zz => sel1, ss => selct, mm => outHEX2
        );
        MUX1 : ThreeBitSix2OneMUX
        PORT MAP(
                uu => sel1, vv => sel2, ww => sel3, xx => sel4, yy => sel5,
                zz => sel0, ss => selct, mm => outHEX1
        );
        MUX0 : ThreeBitSix2OneMUX
        PORT MAP(
                uu => sel0, vv => sel1, ww => sel2, xx => sel3, yy => sel4,
                zz => sel5, ss => selct, mm => outHEX0
        );

        HEX05 : HELLO PORT MAP(c => outHEX5, HEXn => HEX5);
        HEX04 : HELLO PORT MAP(c => outHEX4, HEXn => HEX4);
        HEX03 : HELLO PORT MAP(c => outHEX3, HEXn => HEX3);
        HEX02 : HELLO PORT MAP(c => outHEX2, HEXn => HEX2);
        HEX01 : HELLO PORT MAP(c => outHEX1, HEXn => HEX1);
        HEX00 : HELLO PORT MAP(c => outHEX0, HEXn => HEX0);

END ARCHITECTURE;
```

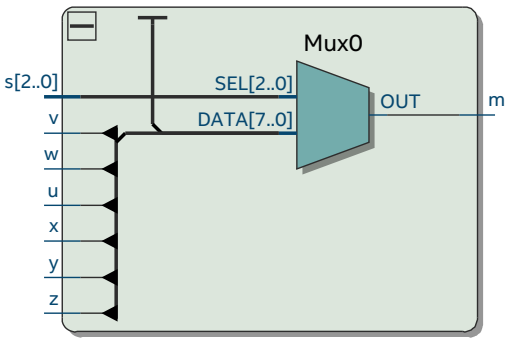
b. Waveform



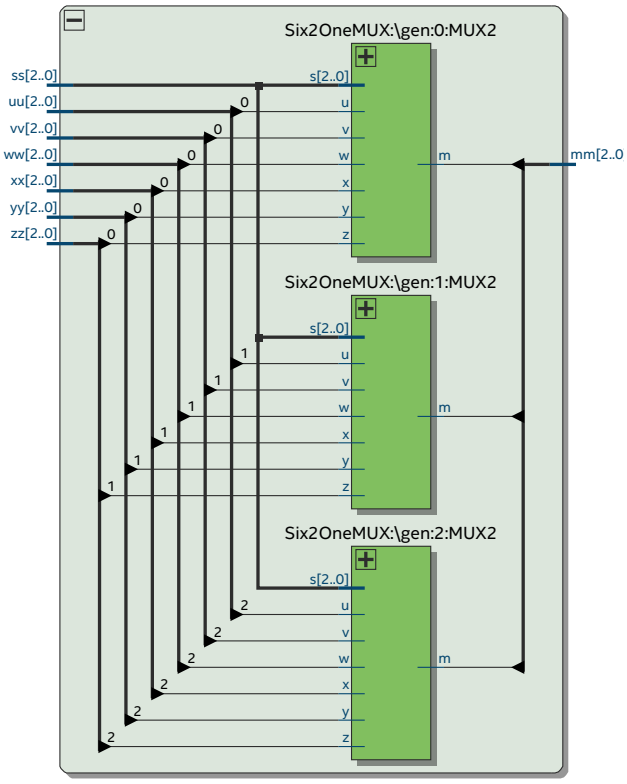
c. Result of RTL viewer



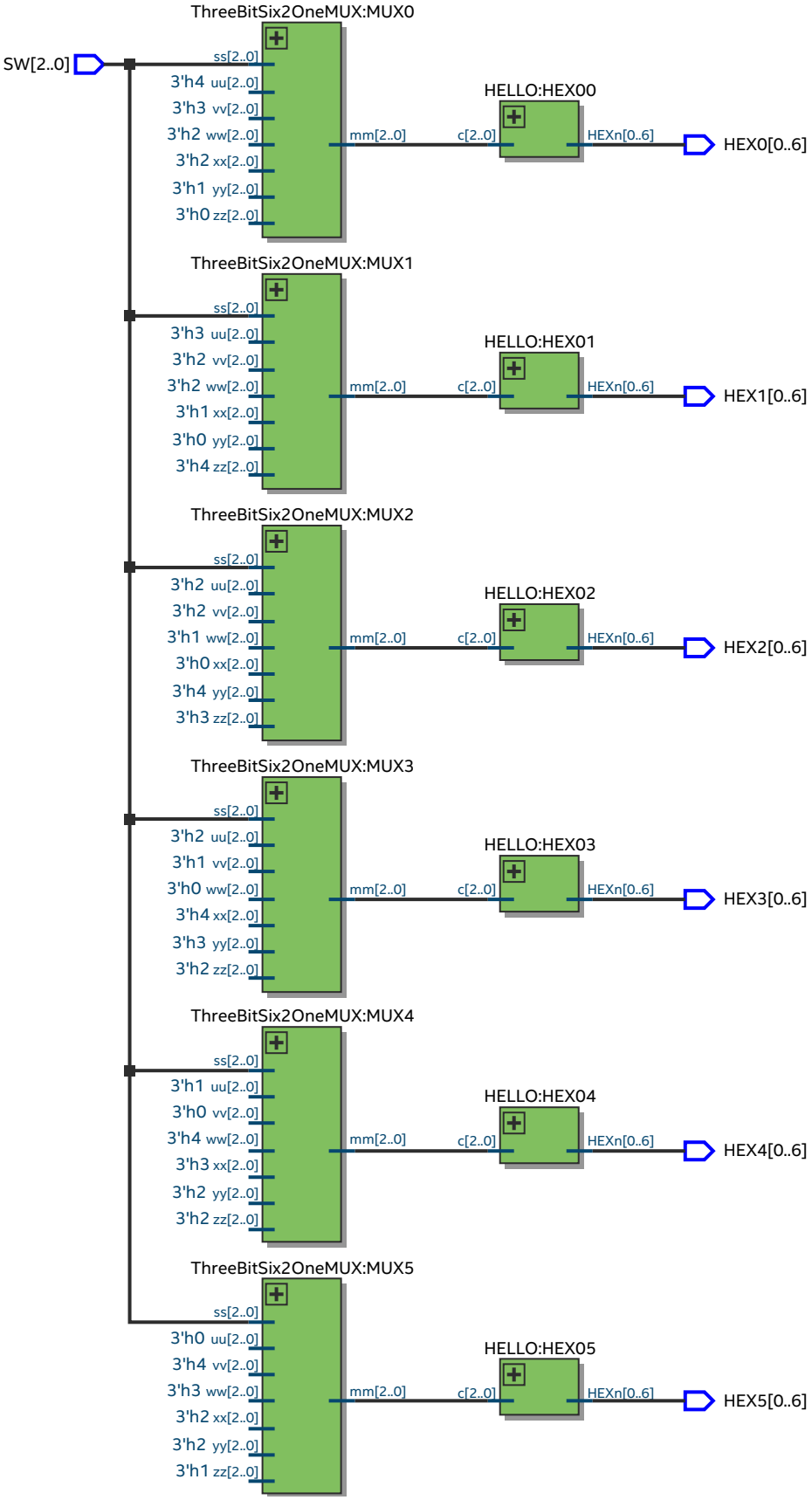
"HELLO" 7-segment decoder



6-to-1 multiplexer



3-bit 6-to-1 multiplexer


ThreeBitSix2OneMUX:MUX0
SW[2..0]
ss[2..0]
3'h4 uu[2..0]
3'h3 vv[2..0]
3'h2 ww[2..0]
3'h2 xx[2..0]
3'h1 yy[2..0]
3'h0 zz[2..0]
mm[2..0]
HELLO:HEX00
c[2..0]
HEXn[0..6]
HEX0[0..6]
ThreeBitSix2OneMUX:MUX1
ss[2..0]
3'h3 uu[2..0]
3'h2 vv[2..0]
3'h2 ww[2..0]
3'h1 xx[2..0]
3'h0 yy[2..0]
3'h4 zz[2..0]
mm[2..0]
HELLO:HEX01
c[2..0]
HEXn[0..6]
HEX1[0..6]
ThreeBitSix2OneMUX:MUX2
ss[2..0]
3'h2 uu[2..0]
3'h2 vv[2..0]
3'h1 ww[2..0]
3'h0 xx[2..0]
3'h4 yy[2..0]
3'h3 zz[2..0]
mm[2..0]
HELLO:HEX02
c[2..0]
HEXn[0..6]
HEX2[0..6]
ThreeBitSix2OneMUX:MUX3
ss[2..0]
3'h2 uu[2..0]
3'h1 vv[2..0]
3'h0 ww[2..0]
3'h4 xx[2..0]
3'h3 yy[2..0]
3'h2 zz[2..0]
mm[2..0]
HELLO:HEX03
c[2..0]
HEXn[0..6]
HEX3[0..6]
ThreeBitSix2OneMUX:MUX4
ss[2..0]
3'h1 uu[2..0]
3'h0 vv[2..0]
3'h4 ww[2..0]
3'h3 xx[2..0]
3'h2 yy[2..0]
3'h2 zz[2..0]
mm[2..0]
HELLO:HEX04
c[2..0]
HEXn[0..6]
HEX4[0..6]
ThreeBitSix2OneMUX:MUX5
ss[2..0]
3'h0 uu[2..0]
3'h4 vv[2..0]
3'h3 ww[2..0]
3'h2 xx[2..0]
3'h2 yy[2..0]
3'h1 zz[2..0]
mm[2..0]
HELLO:HEX05
c[2..0]
HEXn[0..6]
HEX5[0..6]

Top level

5. Known how to interface 7-segment LED and multiplexer

v_3	v_2	v_1	v_0	d_1d_0
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

a. Method 1: with MUX, comparator and "Circuit A"

Truth tables and functions

Comparator > 9

v_3	v_2	v_1	v_0	z
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

$\Rightarrow z = v_3(v_2 + v_1)$

Circuit A

v_3	v_2	v_1	v_0	A_3	A_2	A_1	A_0
0	0	0	0	×	×	×	×
0	0	0	1	×	×	×	×
0	0	1	0	×	×	×	×
0	0	1	1	×	×	×	×
0	1	0	0	×	×	×	×
0	1	0	1	×	×	×	×
0	1	1	0	×	×	×	×
0	1	1	1	×	×	×	×
1	0	0	0	×	×	×	×
1	0	0	1	×	×	×	×
1	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1
1	1	0	0	0	0	1	0
1	1	0	1	0	0	1	1
1	1	1	0	0	1	0	0
1	1	1	1	0	1	0	1

$$\Rightarrow \begin{array}{rcl} A_3 & = & 0 \\ A_2 & = & v_2 v_1 \\ A_1 & = & \bar{v}_1 \\ A_0 & = & v_0 \end{array}$$

Code

BCD.vhdl

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

ENTITY BCD IS
        PORT (
                c : IN std_logic_vector(3 DOWNTO 0);
                HEXn : OUT std_logic_vector(0 TO 6)
        );
END BCD;

ARCHITECTURE behavior OF BCD IS
        SIGNAL HEX : std_logic_vector(0 TO 6);
BEGIN
        HEXn <= NOT(HEX);
        WITH c SELECT
        HEX <= "1111110" WHEN "0000",
               "0110000" WHEN "0001",
               "1101101" WHEN "0010",
               "1111001" WHEN "0011",
               "0110011" WHEN "0100",
               "1011011" WHEN "0101",
               "1011111" WHEN "0110",
               "1110000" WHEN "0111",
               "1111111" WHEN "1000",
               "1111011" WHEN "1001",
               "0000000" WHEN OTHERS;
END behavior;
```

MUX.vhdl

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

ENTITY MUX IS
        PORT (
                muxIn1 : IN std_logic;
                muxSel : IN std_logic;
                muxIn2 : IN std_logic;
                muxOut : OUT std_logic
        );
END ENTITY;

ARCHITECTURE arch OF MUX IS
BEGIN
        muxOut <= (NOT(muxSel) AND muxIn1) OR (muxSel AND muxIn2);
END arch;
```

FourBitMUX.vhdl

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

ENTITY FourBitMUX IS
        PORT (
                fourBitMuxIn1 : IN std_logic_vector(3 DOWNTO 0);
                fourBitMuxIn2 : IN std_logic_vector(3 DOWNTO 0);
                fourBitMuxSel : IN std_logic;
                fourBitMuxOut : OUT std_logic_vector(3 DOWNTO 0)
        );
```

```
END ENTITY;

ARCHITECTURE arch OF FourBitMUX IS
	COMPONENT MUX
		PORT (
			muxIn1 : IN std_logic;
			muxSel : IN std_logic;
			muxIn2 : IN std_logic;
			muxOut : OUT std_logic
		);
	END COMPONENT;
BEGIN
	gen : FOR i IN 3 DOWNTO 0 GENERATE
		MUX2 : MUX
		PORT MAP(
			muxSel => fourBitMuxSel,
			muxIn1 => fourBitMuxIn1(i),
			muxIn2 => fourBitMuxIn2(i),
			muxOut => fourBitMuxOut(i)
		);
	END GENERATE;
END arch;
```

Comparator.vhdl

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

ENTITY Comparator IS
	PORT (
		compIn : IN std_logic_vector(3 DOWNTO 0);
		compOut : OUT std_logic
	);
END ENTITY;

ARCHITECTURE behave OF Comparator IS
BEGIN
	-- Z = A(B+C)
	compOut <= compIn(3) AND (compIn(2) OR compIn(1));

END ARCHITECTURE;
```

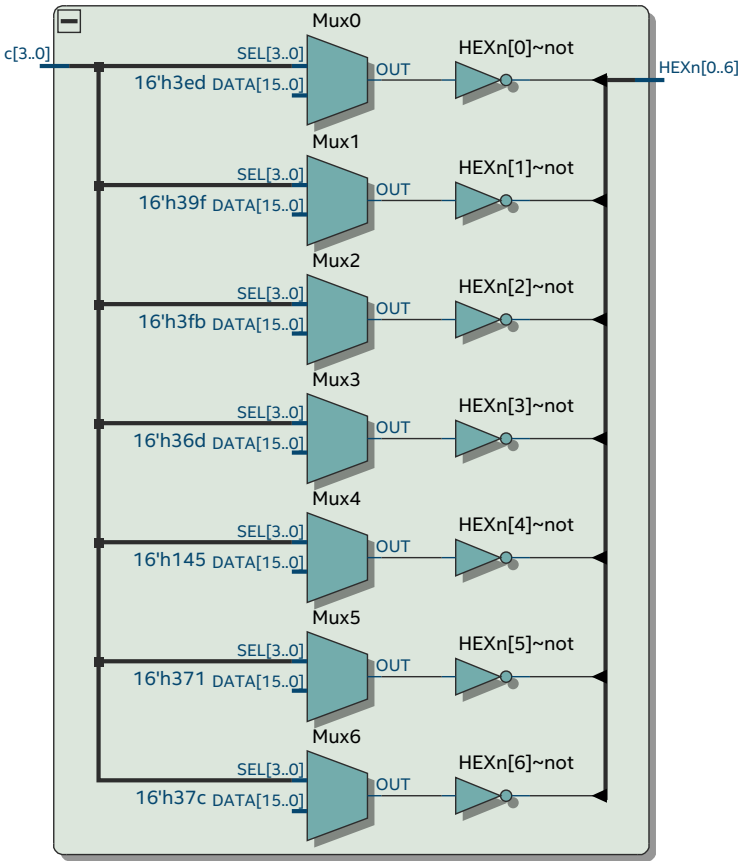
CircuitA.vhdl

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

ENTITY CircuitA IS
	PORT (
		dIn : IN std_logic_vector(3 DOWNTO 0);
		dOut : OUT std_logic_vector(3 DOWNTO 0)
	);
END ENTITY;

ARCHITECTURE behave OF CircuitA IS
BEGIN
	dOut(3) <= '0';
	dOut(2) <= dIn(2) AND dIn(1);
	dOut(1) <= NOT(dIn(1));
	dOut(0) <= dIn(0);

END ARCHITECTURE;
```

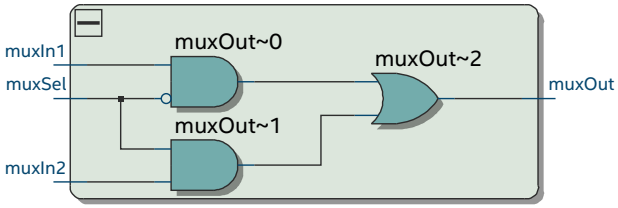
Exc5.vhdl

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

ENTITY Exc5 IS
        PORT (
                V : IN std_logic_vector(3 DOWNTO 0);
                HEX0 : OUT std_logic_vector(0 TO 6);
                HEX1 : OUT std_logic_vector(0 TO 6)
        );
END ENTITY;

ARCHITECTURE behave OF Exc5 IS

        SIGNAL z : std_logic;
        SIGNAL A, m : std_logic_vector(3 DOWNTO 0);

        COMPONENT FourBitMUX IS
                PORT (
                        fourBitMuxIn1 : IN std_logic_vector(3 DOWNTO 0);
                        fourBitMuxIn2 : IN std_logic_vector(3 DOWNTO 0);
                        fourBitMuxSel : IN std_logic;
                        fourBitMuxOut : OUT std_logic_vector(3 DOWNTO 0)
                );
        END COMPONENT;

        COMPONENT Comparator IS
                PORT (
                        compIn : IN std_logic_vector(3 DOWNTO 0);
                        compOut : OUT std_logic
                );
        END COMPONENT;

        COMPONENT CircuitA IS
                PORT (
                        dIn : IN std_logic_vector(3 DOWNTO 0);
                        dOut : OUT std_logic_vector(3 DOWNTO 0)
                );
        END COMPONENT;

        COMPONENT BCD IS
                PORT (
                        c : IN std_logic_vector(3 DOWNTO 0);
                        HEXn : OUT std_logic_vector(0 TO 6)
                );
        END COMPONENT;

BEGIN
        comp : Comparator PORT MAP( compIn => V, compOut => z);
        cirA : CircuitA PORT MAP(dIn => V, dOut => A);
        mux : FourBitMUX
        PORT MAP(
                fourBitMuxIn1 => V,
                fourBitMuxIn2 => A,
                fourBitMuxSel => z,
                fourBitMuxOut => m
        );

        HEX01 : BCD PORT MAP(c => "000" & z, HEXn => HEX1);
        HEX00 : BCD PORT MAP(c => m, HEXn => HEX0);

END ARCHITECTURE;
```

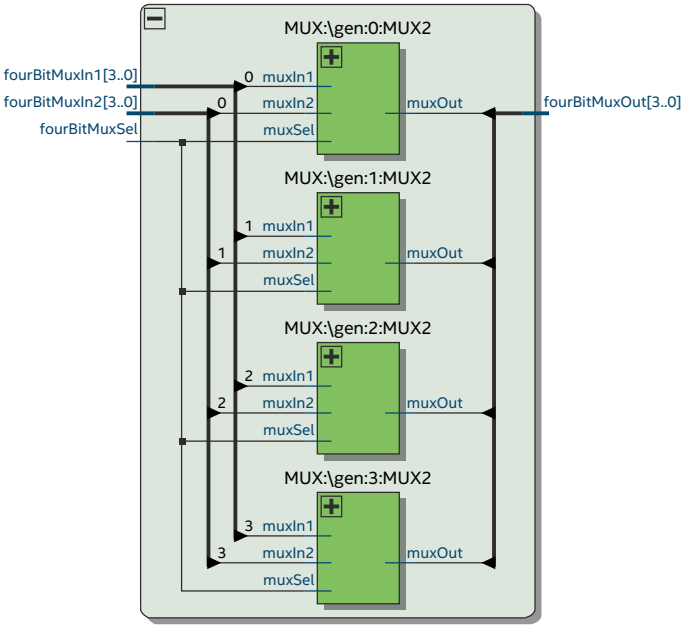
Result of RTL viewer



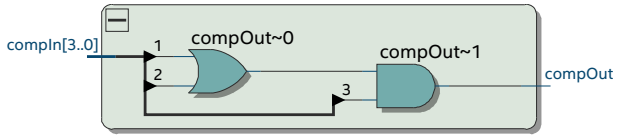
BCD 7-segment decoder



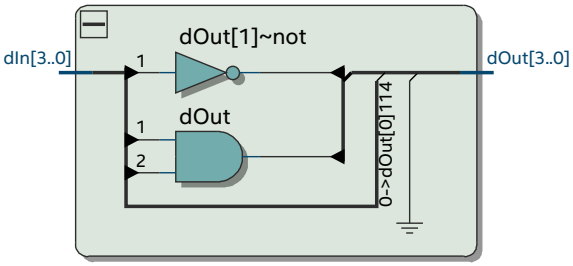
2-to-1 multiplexer



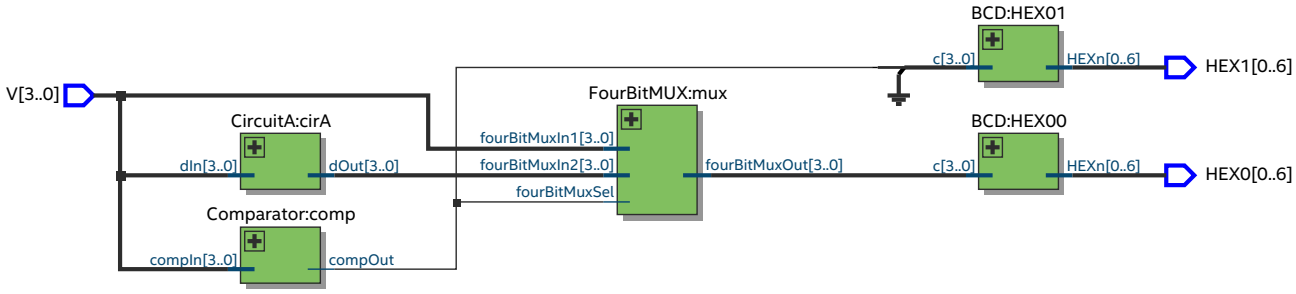
4-bit 2-to-1 multiplexer



Comparator > 9



Circuit A



Top level

b. Method 2: using conditions in VHDL

BCD.vhdl

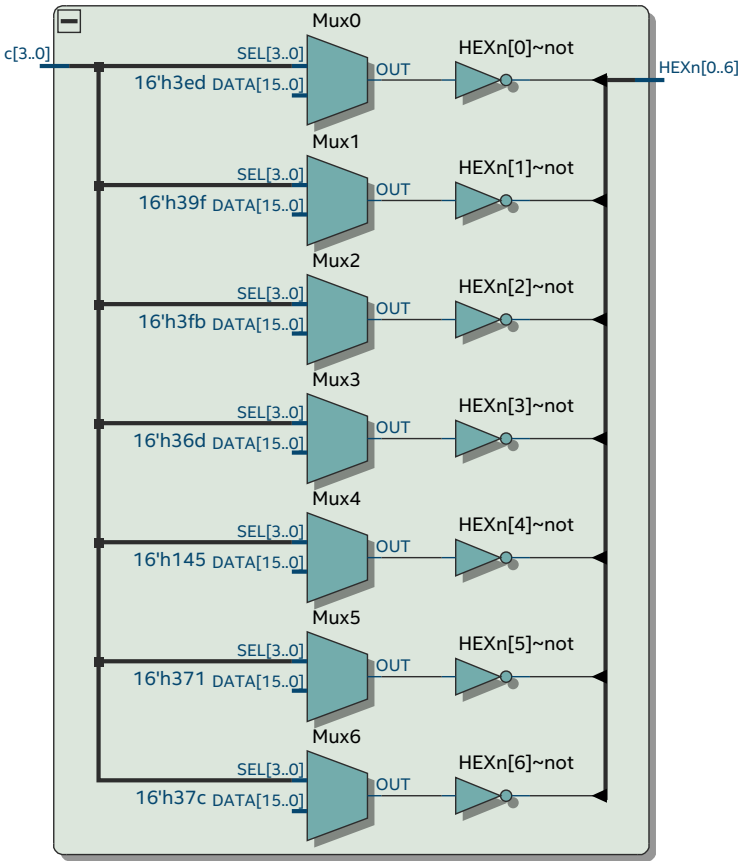
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY BCD IS
        PORT (
                c : IN std_logic_vector(3 DOWNTO 0);
                HEXn : OUT std_logic_vector(0 TO 6)
        );
END BCD;

ARCHITECTURE behavior OF BCD IS
        SIGNAL HEX : std_logic_vector(0 TO 6);
BEGIN
        HEXn <= NOT(HEX);
        WITH c SELECT
        HEX <= "1111110" WHEN "0000",
               "0110000" WHEN "0001",
               "1101101" WHEN "0010",
               "1111001" WHEN "0011",
               "0110011" WHEN "0100",
               "1011011" WHEN "0101",
               "1011111" WHEN "0110",
               "1110000" WHEN "0111",
               "1111111" WHEN "1000",
               "1111011" WHEN "1001",
               "0000000" WHEN OTHERS;
END behavior;
```

Exc5.vhdl

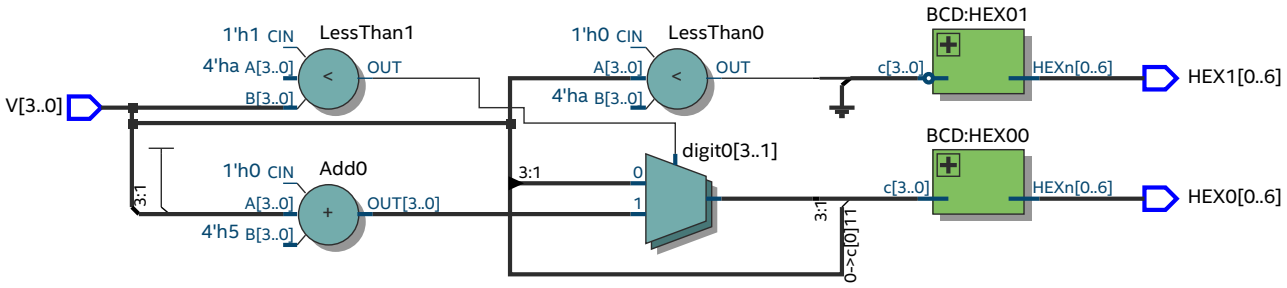
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
```

```
USE IEEE.numeric_std.ALL;

ENTITY Exc5 IS
        PORT (
                V : IN std_logic_vector(3 DOWNTO 0);
                HEX0 : OUT std_logic_vector(0 TO 6);
                HEX1 : OUT std_logic_vector(0 TO 6)
        );
END ENTITY;

ARCHITECTURE behave OF Exc5 IS

        SIGNAL digit1, digit0 : std_logic_vector(3 DOWNTO 0);

        COMPONENT BCD IS
                PORT (
                        c : IN std_logic_vector(3 DOWNTO 0);
                        HEXn : OUT std_logic_vector(0 TO 6)
                );
        END COMPONENT;


BEGIN
        digit1 <= "0000" WHEN unsigned(V) < 10 ELSE "0001";
        digit0 <= std_logic_vector(unsigned(V) - 10) WHEN unsigned(V) >= 10 ELSE
                  std_logic_vector(unsigned(V));

        HEX01 : BCD PORT MAP(c => digit1, HEXn => HEX1);
        HEX00 : BCD PORT MAP(c => digit0, HEXn => HEX0);

END ARCHITECTURE;
```

Result of RTL viewer



BCD 7-segment decoder



Top level

c. Waveform