Digital-electronics-1

Labs/04-segment

Dominik Grenčík, 220815

Digital-electronics-1

1. Preparation tasks

- **Table with connection of 7-segment displays on Nexys A7 board**

Anode	Board	7-seg display	Board
AN0	J17	CA	T10
AN1	J18	CB	R10
AN2	T9	CC	K16
AN3	J14	CD	K13
AN4	P14	CE	P15
AN5	T14	CF	T11
AN6	K2	CG	L18
AN7	U13	DP	H15

- **Decoder truth table for common anode 7-segment display**

Hex	Inputs	A	B	C	D	E	F	G
0	0000	0	0	0	0	0	0	1
1	0001	1	0	0	1	1	1	1
2	0010	0	0	1	0	0	1	0
3	0011	0	0	0	0	1	1	0
4	0100	1	0	0	1	1	0	0
5	0101	0	1	0	0	1	0	0
6	0110	0	1	0	0	0	0	0
7	0111	0	0	0	1	1	1	1
8	1000	0	0	0	0	0	0	0
9	1001	0	0	0	0	1	0	0

Hex	Inputs	A	B	C	D	E	F	G
A	1010	0	0	0	1	0	0	0
b	1011	1	1	0	0	0	0	0
C	1100	0	1	1	0	0	0	1
d	1101	1	0	0	0	0	1	0
E	1110	0	1	1	0	0	0	0
F	1111	0	1	1	1	0	0	0

2. Seven-segment display decoder

- **Listing of VHDL architecture from source file hex_7seg.vhd**

```
architecture Behavioral of hex_7seg is

begin

p_7seg_decoder : process(hex_i)
    begin
        case hex_i is
            when "0000" =>
                seg_o <= "0000001";     -- 0
            when "0001" =>
                seg_o <= "1001111";     -- 1
            when "0010" =>
                seg_o <= "0010010";     -- 2
            when "0011" =>
                seg_o <= "0000110";     -- 3
            when "0100" =>
                seg_o <= "1001100";     -- 4
            when "0101" =>
                seg_o <= "0100100";     -- 5
            when "0110" =>
                seg_o <= "0100000";     -- 6
            when "0111" =>
                seg_o <= "0001111";     -- 7
            when "1000" =>
                seg_o <= "0000000";     -- 8
            when "1001" =>
                seg_o <= "0000100";     -- 9
            when "1010" =>
                seg_o <= "0001000";     -- A
            when "1011" =>
                seg_o <= "1100000";     -- b
            when "1100" =>
                seg_o <= "0110001";     -- C
            when "1101" =>
                seg_o <= "1000010";     -- d
```

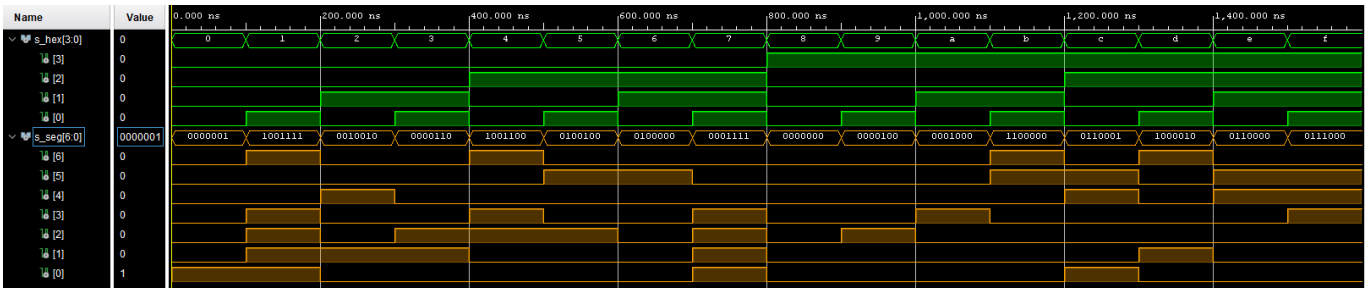
```
            when "1110" =>
                seg_o <= "0110000";     -- E
            when others =>
                seg_o <= "0111000";     -- F
        end case;
    end process p_7seg_decoder;


end Behavioral;
```

- **Listing of VHDL stimulus process from testbench file tb_hex_7seg.vhd**

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

        s_hex     <= "0000";  wait for 100 ns;

        s_hex     <= "0001";  wait for 100 ns;

        s_hex     <= "0010";  wait for 100 ns;

        s_hex     <= "0011";  wait for 100 ns;

        s_hex     <= "0100";  wait for 100 ns;

        s_hex     <= "0101";  wait for 100 ns;

        s_hex     <= "0110";  wait for 100 ns;

        s_hex     <= "0111";  wait for 100 ns;

        s_hex     <= "1000";  wait for 100 ns;

        s_hex     <= "1001";  wait for 100 ns;

        s_hex     <= "1010";  wait for 100 ns;

        s_hex     <= "1011";  wait for 100 ns;

        s_hex     <= "1100";  wait for 100 ns;

        s_hex     <= "1101";  wait for 100 ns;

        s_hex     <= "1110";  wait for 100 ns;

        s_hex     <= "1111";  wait for 100 ns;

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

```
        wait;
    end process p_stimulus;
```

- **Screenshot with simulated time waveforms**



- **Listing of VHDL code from source file top.vhd**

```
entity top is
    Port
    (
        SW : in  STD_LOGIC_VECTOR (3 downto 0);
        CA : out STD_LOGIC;
        CB : out STD_LOGIC;
        CC : out STD_LOGIC;
        CD : out STD_LOGIC;
        CE : out STD_LOGIC;
        CF : out STD_LOGIC;
        CG : out STD_LOGIC;

        LED : out STD_LOGIC_VECTOR (8 - 1 downto 0);
        AN  : out STD_LOGIC_VECTOR (8 - 1 downto 0)
    );
end top;

architecture Behavioral of top is

begin

-- Instance (copy) of hex_7seg entity
    hex2seg : entity work.hex_7seg
        port map(
            hex_i    => SW,


            seg_o(6) => CA,
            seg_o(5) => CB,
            seg_o(4) => CC,
            seg_o(3) => CD,
            seg_o(2) => CE,
            seg_o(1) => CF,
            seg_o(0) => CG
        );
```

```
        AN <= b"1111_0111";

        LED(3 downto 0) <= SW;

        LED(4)  <= '1' when (SW = "0000") else '0';
        LED(5)  <= '1' when (SW > "1001") else '0';
        LED(6)  <= '1' when (SW = "0001" or SW = "0011" or SW = "0101" or SW =
"0111" or SW = "1001" or SW = "1011" or SW = "1101" or SW = "1111") else '0';
        LED(7)  <= '1' when (SW = "0001" or SW = "0010" or SW = "0100" or SW =
"1000") else '0';

end Behavioral;
```

---

3. LED(7:4) indicators

- **Truth table for LEDs(7:4)**

Hex	Inputs	LED4	LED5	LED6	LED7
0	0000	1	0	0	0
1	0001	0	0	1	1
2	0010	0	0	0	1
3	0011	0	0	1	0
4	0100	0	0	0	1
5	0101	0	0	1	0
6	0110	0	0	0	0
7	0111	0	0	1	0
8	1000	0	0	0	1
9	1001	0	0	1	0
A	1010	0	1	0	0
b	1011	0	1	1	0
C	1100	0	1	0	0
d	1101	0	1	1	0
E	1110	0	1	0	0
F	1111	0	1	1	0

- **Listing of VHDL code for LEDs(7:4) from source file top.vhd**

```
entity top is
    Port
    (
        SW : in  STD_LOGIC_VECTOR (3 downto 0);
        CA : out STD_LOGIC;
        CB : out STD_LOGIC;
        CC : out STD_LOGIC;
        CD : out STD_LOGIC;
        CE : out STD_LOGIC;
        CF : out STD_LOGIC;
        CG : out STD_LOGIC;

        LED : out STD_LOGIC_VECTOR (8 - 1 downto 0);
        AN  : out STD_LOGIC_VECTOR (8 - 1 downto 0)
    );
end top;

architecture Behavioral of top is

begin

-- Instance (copy) of hex_7seg entity
    hex2seg : entity work.hex_7seg
        port map(
            hex_i    => SW,

            seg_o(6) => CA,
            seg_o(5) => CB,
            seg_o(4) => CC,
            seg_o(3) => CD,
            seg_o(2) => CE,
            seg_o(1) => CF,
            seg_o(0) => CG
        );

        AN <= b"1111_0111";

        LED(3 downto 0) <= SW;

        LED(4)  <= '1' when (SW = "0000") else '0';
        LED(5)  <= '1' when (SW > "1001") else '0';
        LED(6)  <= '1' when (SW = "0001" or SW = "0011" or SW = "0101" or SW =
"0111" or SW = "1001" or SW = "1011" or SW = "1101" or SW = "1111") else '0';
        LED(7)  <= '1' when (SW = "0001" or SW = "0010" or SW = "0100" or SW =
"1000") else '0';

end Behavioral;
```

- **Listing of VHDL code for LEDs(7:4) from testbench file tb_top.vhd**

```
architecture Behavioral of tb_top is

    signal s_hex : std_logic_vector (4 - 1 downto 0);
    signal s_LED : std_logic_vector (8 - 1 downto 0);
    signal s_AN  : std_logic_vector (8 - 1 downto 0);


begin

    uut_top : entity work.top
        port map(
            SW          => s_hex,
            LED         => s_LED,
            AN          => s_AN
        );

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

        s_hex     <= "0000";  wait for 100 ns;

        s_hex     <= "0001";  wait for 100 ns;

        s_hex     <= "0010";  wait for 100 ns;

        s_hex     <= "0011";  wait for 100 ns;

        s_hex     <= "0100";  wait for 100 ns;

        s_hex     <= "0101";  wait for 100 ns;

        s_hex     <= "0110";  wait for 100 ns;

        s_hex     <= "0111";  wait for 100 ns;

        s_hex     <= "1000";  wait for 100 ns;

        s_hex     <= "1001";  wait for 100 ns;

        s_hex     <= "1010";  wait for 100 ns;

        s_hex     <= "1011";  wait for 100 ns;

        s_hex     <= "1100";  wait for 100 ns;

        s_hex     <= "1101";  wait for 100 ns;

        s_hex     <= "1110";  wait for 100 ns;

        s_hex     <= "1111";  wait for 100 ns;
```

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

end Behavioral;
```

- **Screenshot with simulated time waveforms**