Laboratórna úloha číslo 6

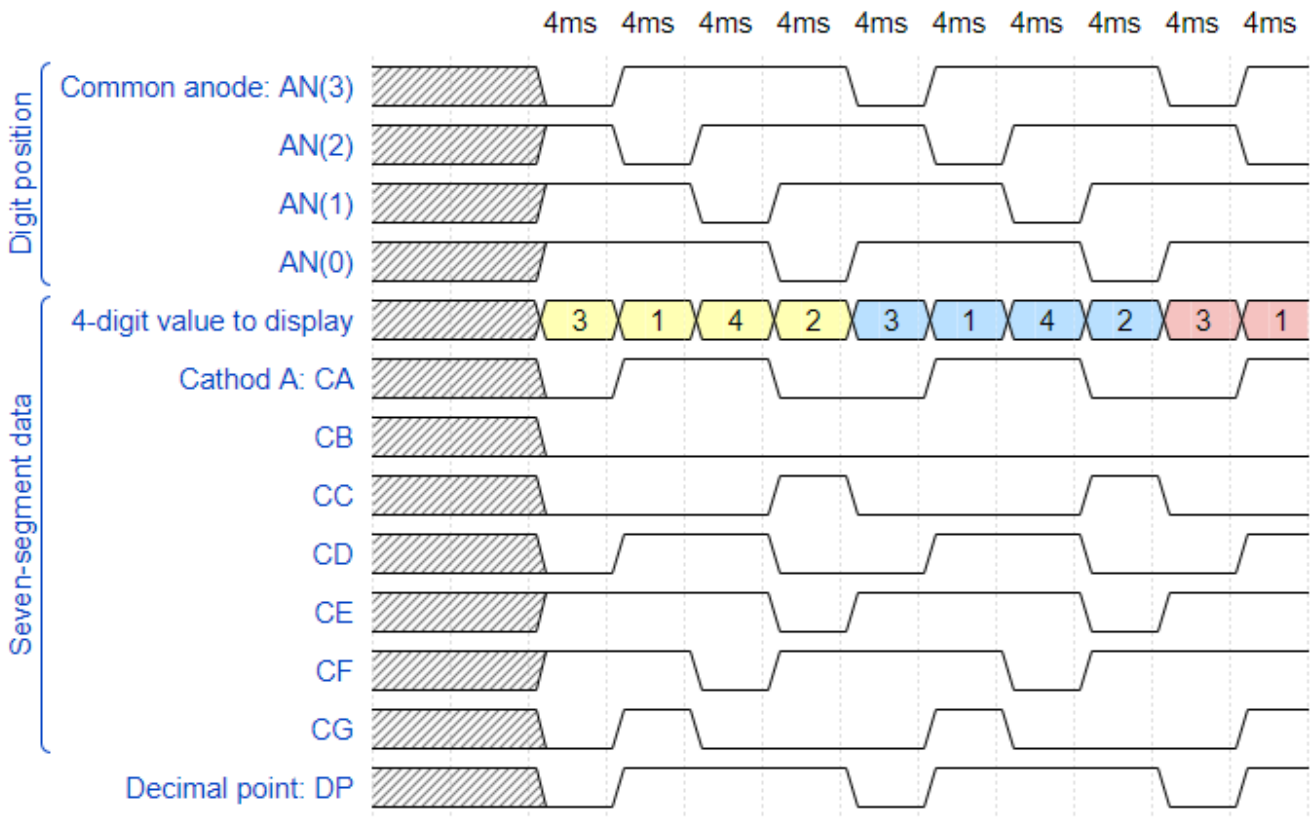
Daniel Haluška

GitHub:

Link repozitára: https://github.com/DaNNym99/Digital-electronics-1

1. Časový diagram:

1.1. Garf



1.2. Kód pre vytvorenie

```
{
  signal:
  [
    ['Digit position',
      {name: 'Common anode: AN(3)', wave: 'xx01..01..01'},
      {name: 'AN(2)', wave: 'xx101..01..0'},
      {name: 'AN(1)', wave: 'xx1.01..01..'},
      {name: 'AN(0)', wave: 'xx1..01..01.'},
    ],
    ['Seven-segment data',
      {name: '4-digit value to display', wave: 'xx3333555599', data:
['3','1','4','2','3','1','4','2','3','1']},
```

```
      {name: 'Cathod A: CA', wave: 'xx01.0.1.0.1'},
      {name: 'CB', wave: 'xx0.........'},
      {name: 'CC', wave: 'xx0..10..10.'},
      {name: 'CD', wave: 'xx01.0.1.0.1'},
      {name: 'CE', wave: 'xx1..01..01.'},
      {name: 'CF', wave: 'xx1.01..01..'},
      {name: 'CG', wave: 'xx010..10..1'},
    ],
    {name: 'Decimal point: DP', wave: 'xx01..01..01'},
  ],
  head:
  {
    text: '                    4ms   4ms   4ms   4ms   4ms   4ms   4ms   4ms   4ms
4ms',
  },
}
```

- Vytvorené v (https://wavedrom.com/editor.html)

2. Ovládač pre 4x7-segmentovky

2.1. Proces p_mux

```
p_mux : process(s_cnt, data0_i, data1_i, data2_i, data3_i, dp_i)
    begin
        case s_cnt is
            when "11" =>
                s_hex <= data3_i;
                dp_o  <= dp_i(3);
                dig_o <= "0111";

            when "10" =>
                -- WRITE YOUR CODE HERE
                s_hex <= data2_i;
                dp_o  <= dp_i(2);
                dig_o <= "1011";

            when "01" =>
                -- WRITE YOUR CODE HERE
                s_hex <= data1_i;
                dp_o  <= dp_i(1);
                dig_o <= "1101";

            when others =>
                -- WRITE YOUR CODE HERE
                s_hex <= data0_i;
                dp_o  <= dp_i(0);
                dig_o <= "1110";
        end case;
    end process p_mux;
```

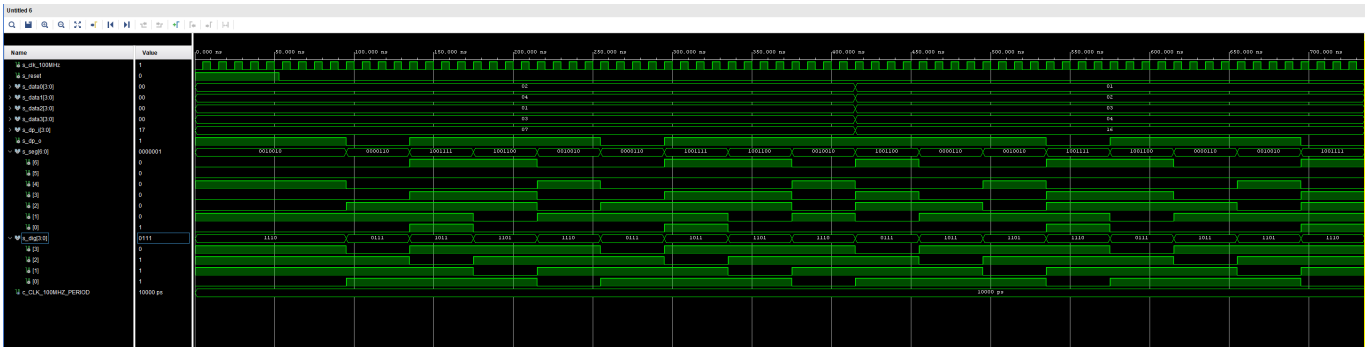
2.2. Súbor tb_driver_7seg_4digits.vhdl

```
----------------------------------------------------------------------------
--
-- Template for 4-digit 7-segment display driver testbench.
-- Nexys A7-50T, Vivado v2020.1.1, EDA Playground
--
-- Copyright (c) 2020 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_driver_7seg_4digits is
    -- Entity of testbench is always empty
end entity tb_driver_7seg_4digits;

----------------------------------------------------------------------------
-- Architecture body for testbench
----------------------------------------------------------------------------
architecture testbench of tb_driver_7seg_4digits is



    -- Local constants
    constant c_CLK_100MHZ_PERIOD : time    := 10 ns;

    --Local signals
    signal s_clk_100MHz : std_logic;

    --- WRITE YOUR CODE HERE
    signal s_reset      : std_logic;

    signal s_data0     : std_logic_vector(4 - 1 downto 0);
    signal s_data1     : std_logic_vector(4 - 1 downto 0);
    signal s_data2     : std_logic_vector(4 - 1 downto 0);
    signal s_data3     : std_logic_vector(4 - 1 downto 0);

    signal s_dp_i     : std_logic_vector(4 - 1 downto 0);
    signal s_dp_o     : std_logic;
    signal s_seg     : std_logic_vector(7 - 1 downto 0);
    signal s_dig     : std_logic_vector(4 - 1 downto 0);



begin
```

```
    -- Connecting testbench signals with driver_7seg_4digits entity
    -- (Unit Under Test)
    --- WRITE YOUR CODE HERE

    ----------------------------------------------------------------------
    -- Clock generation process
    ----------------------------------------------------------------------

    uut_driver_7seg_4digits : entity work.driver_7seg_4digits
    port map(
        clk =>s_clk_100MHz,
        reset =>s_reset,

        data0_i =>s_data0,
        data1_i =>s_data1,
        data2_i =>s_data2,
        data3_i =>s_data3,

        dp_i =>s_dp_i,

        dp_o =>s_dp_o,
        seg_o =>s_seg,
        dig_o =>s_dig
    );


    p_clk_gen : process
    begin
        while now < 750 ns loop         -- 75 periods of 100MHz clock
            s_clk_100MHz <= '0';
            wait for c_CLK_100MHZ_PERIOD / 2;
            s_clk_100MHz <= '1';
            wait for c_CLK_100MHZ_PERIOD / 2;
        end loop;
        wait;
    end process p_clk_gen;

    ----------------------------------------------------------------------
    -- Reset generation process
    ----------------------------------------------------------------------
    --- WRITE YOUR CODE HERE

    p_reset_gen : process
    begin
        s_reset <= '0';
        wait for 0 ns;
        -- Reset activated
        s_reset <= '1';
        wait for 53 ns;
        s_reset <= '0';
        wait;

    end process p_reset_gen;
```

```
    ------------------------------------------------------------------------
    -- Data generation process
    ------------------------------------------------------------------------
    --- WRITE YOUR CODE HERE

    p_stimulus : process
    begin
        report "Stimulus process started" severity note;

        s_data3 <= "0011";
        s_data2 <= "0001";
        s_data1 <= "0100";
        s_data0 <= "0010";
        s_dp_i <= "0111";
        wait for 415 ns;
        s_data3 <= "0100";
        s_data2 <= "0011";
        s_data1 <= "0010";
        s_data0 <= "0001";
        s_dp_i <= "1110";
        wait for 320 ns;
        s_data3 <= "0000";
        s_data2 <= "0000";
        s_data1 <= "0000";
        s_data0 <= "0000";
        s_dp_i <= "1111";
        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;

end architecture testbench;
```

2.3. Vystup symulácie

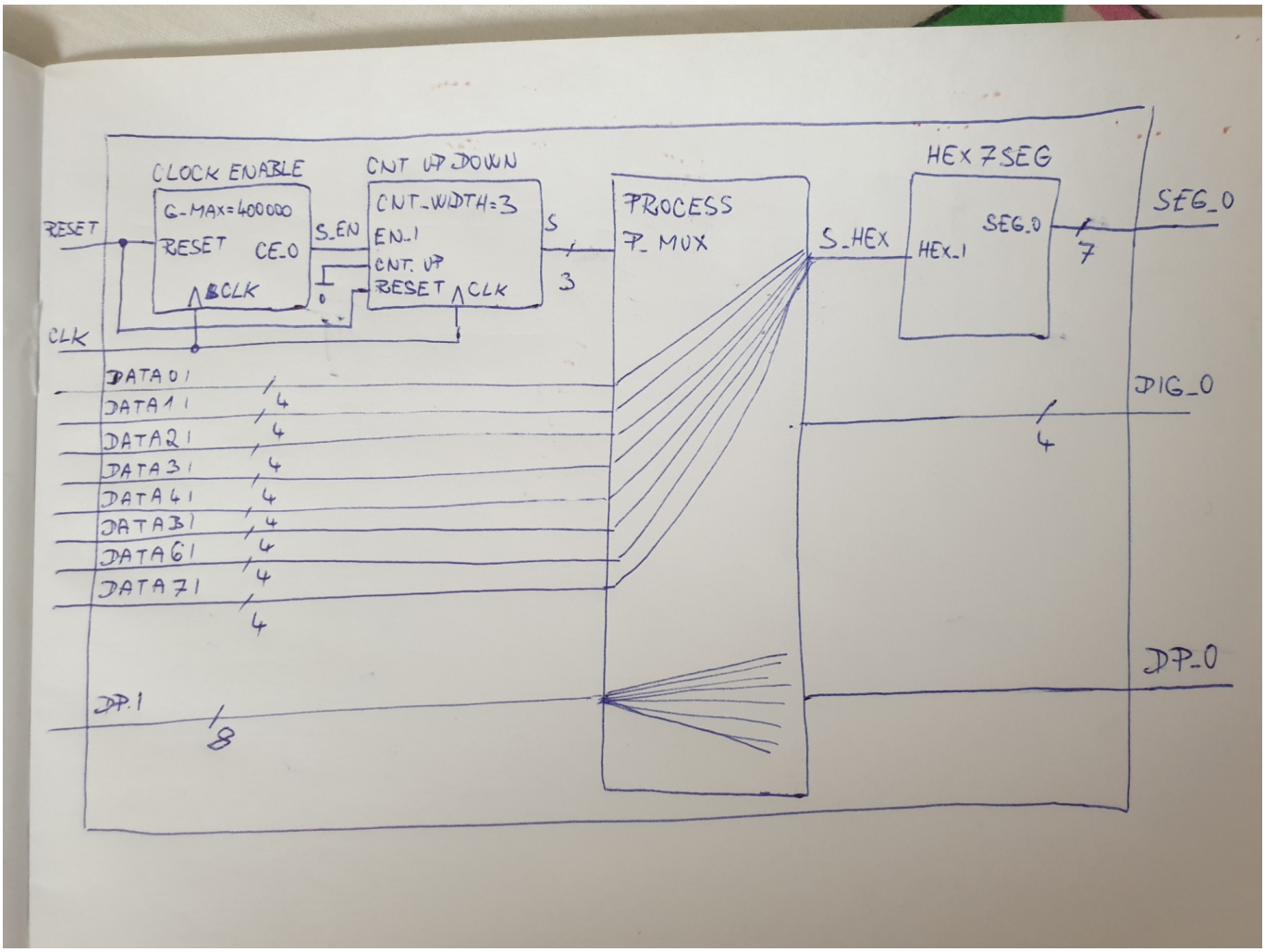


2.4. Architectura súboru top

```
    architecture Behavioral of top is
    -- No internal signals
begin

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

```
    -- Instance (copy) of driver_7seg_4digits entity
    driver_seg_4 : entity work.driver_7seg_4digits
        port map(
            clk         => CLK100MHZ,
            reset       => BTNC,
            data0_i(3) => SW(3),
            data0_i(2) => SW(2),
            data0_i(1) => SW(1),
            data0_i(0) => SW(0),
            data1_i(3) => SW(7),
            data1_i(2) => SW(6),
            data1_i(1) => SW(5),
            data1_i(0) => SW(4),

            data2_i(3) => SW(11),
            data2_i(2) => SW(10),
            data2_i(1) => SW(9),
            data2_i(0) => SW(8),

            data3_i(3) => SW(15),
            data3_i(2) => SW(14),
            data3_i(1) => SW(13),
            data3_i(0) => SW(12),
            --- WRITE YOUR CODE HERE

            dp_i => "0111",
            dp_o => DP,


            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,

            dig_o => AN(4-1 downto 0)

            --- WRITE YOUR CODE HERE

        );

    -- Disconnect the top four digits of the 7-segment display
    AN(7 downto 4) <= b"1111";

end architecture Behavioral;
```

3. Schéma radiča pre 8x7-segmentoviek


CLOCK ENABLE
G_MAX=400000
RESET
CE_O
CLK
S_EN
CNT UP DOWN
CNT_WIDTH=3
EN_I
CNT_UP
RESET
CLK
S
3
PROCESS
P_MUX
S_HEX
HEX 7SEG
HEX_I
SEG_O
7
SEG_O
DIG_O
4
DP_O
RESET
CLK
DATA 0 I
DATA 1 I
DATA 2 I
DATA 3 I
DATA 4 I
DATA 5 I
DATA 6 I
DATA 7 I
4
DP_I
8