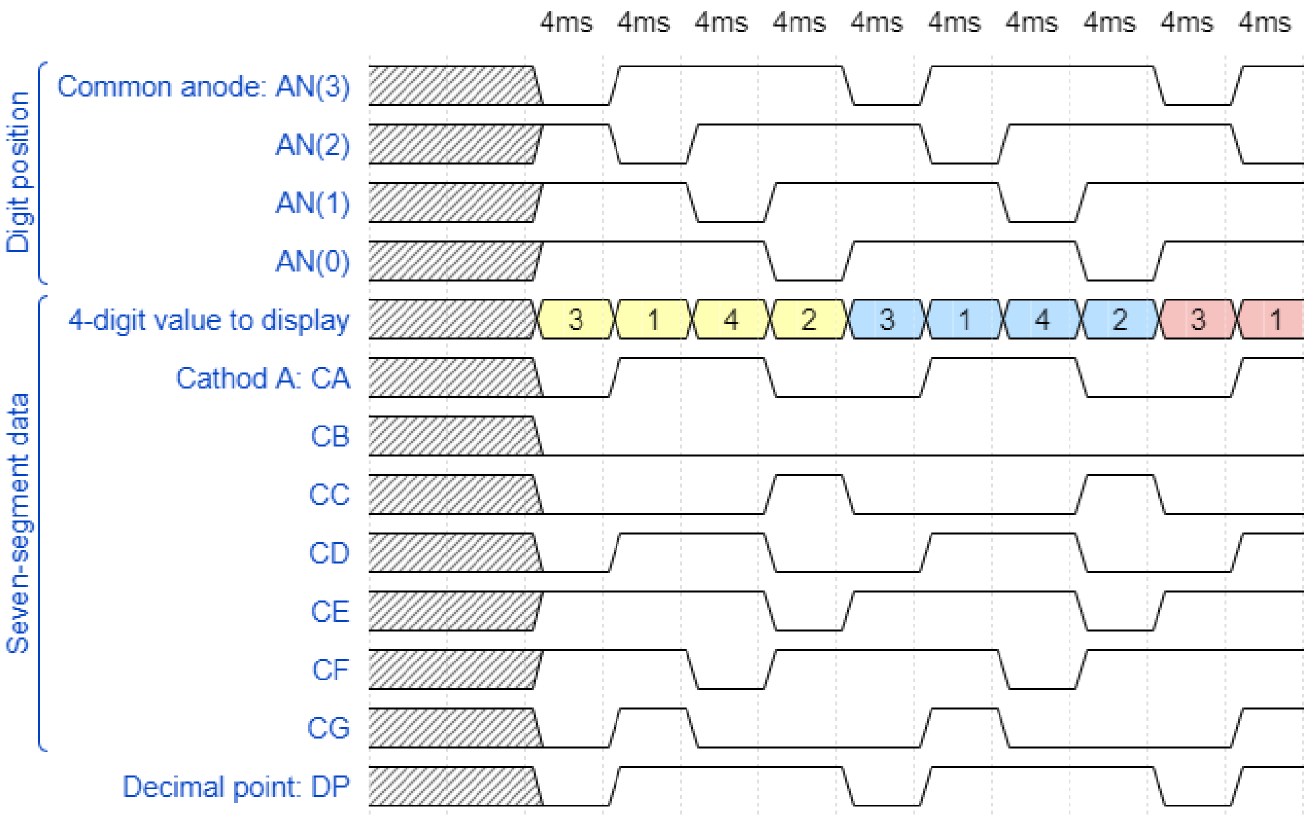
06-display_driver

My repository

My git - Tomáš Kříčka, 223283

1. Timing diagram figure for displaying value 3.142



2. Display driver

VHDL code 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" =>
```

```
                s_hex <= data2_i;
                dp_o  <= dp_i(2);
                dig_o <= "1011";

            when "01" =>
                s_hex <= data1_i;
                dp_o  <= dp_i(1);
                dig_o <= "1101";

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

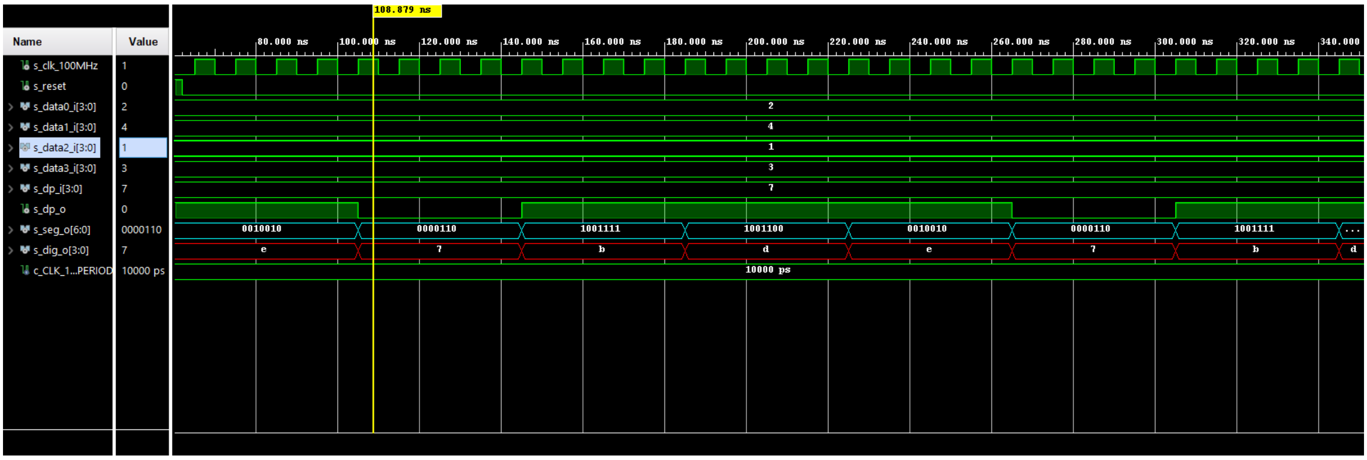
VHDL testbench

```
------------------------------------------------------------------------
--
-- 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;
    signal s_clk        : std_logic;
    signal s_reset      : std_logic;

    signal s_data0_i    : std_logic_vector (4 - 1 downto 0);
    signal s_data1_i    : std_logic_vector (4 - 1 downto 0);
    signal s_data2_i    : std_logic_vector (4 - 1 downto 0);
    signal s_data3_i    : 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_o      : std_logic_vector (7 - 1 downto 0);
    signal s_dig_o      : std_logic_vector (4 - 1 downto 0);

begin
    -- Connecting testbench signals with driver_7seg_4digits entity
    -- (Unit Under Test)
    uut_driver_7seg : entity work.driver_7seg_4digits
        port map(
            clk     => s_clk_100MHz,
            reset   => s_reset,

            data0_i => s_data0_i,
            data1_i => s_data1_i,
            data2_i => s_data2_i,
            data3_i => s_data3_i,

            dp_i    => s_dp_i,
            dp_o    => s_dp_o,
            seg_o   => s_seg_o,
            dig_o   => s_dig_o
    );

    --------------------------------------------------------------------
    -- Clock generation process
    --------------------------------------------------------------------
    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
    --------------------------------------------------------------------
    p_reset_gen : process
    begin
```

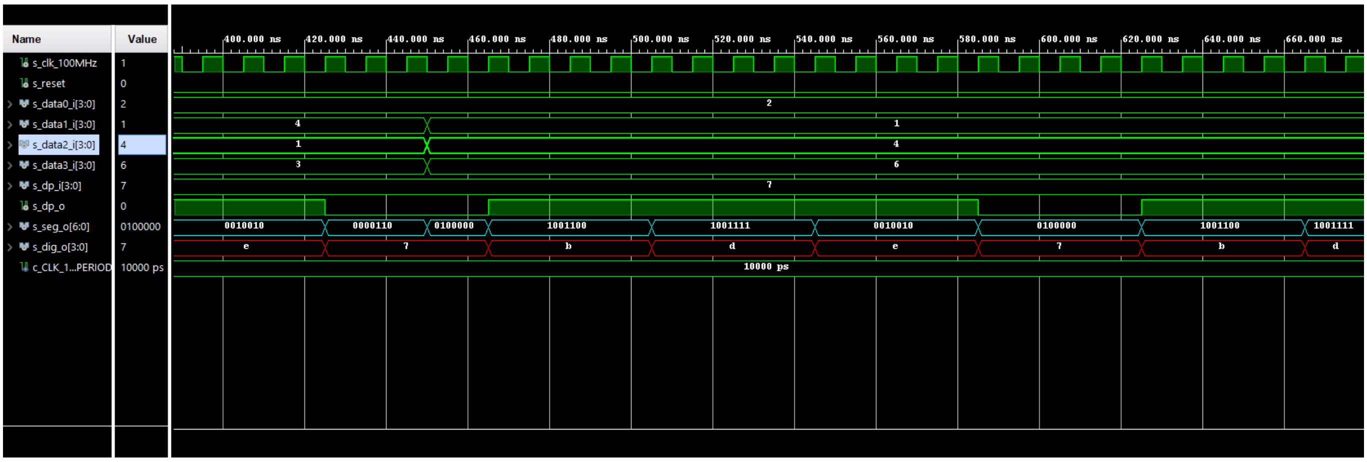
```
        s_reset <= '0';
        wait for 12 ns;

        -- Reset activated
        s_reset <= '1';
        wait for 50 ns;

        s_reset <= '0';
        wait;
    end process p_reset_gen;

    --------------------------------------------------------------------
    -- Data generation process
    --------------------------------------------------------------------
    p_stimulus : process
    begin
        report "Start simulation" severity note;

        s_data3_i <= "0011";
        s_data2_i <= "0001";
        s_data1_i <= "0100";
        s_data0_i <= "0010";
        s_dp_i    <= "0111";

        wait for 450ns;

        s_data3_i <= "0110";
        s_data2_i <= "0100";
        s_data1_i <= "0001";
        s_data0_i <= "0010";
        s_dp_i    <= "0111";

        report "End simulation" severity note;
        wait;
    end process p_stimulus;

end architecture testbench;
```

Simulation

Wavefoms for 3.142



Wavefoms for 641.2

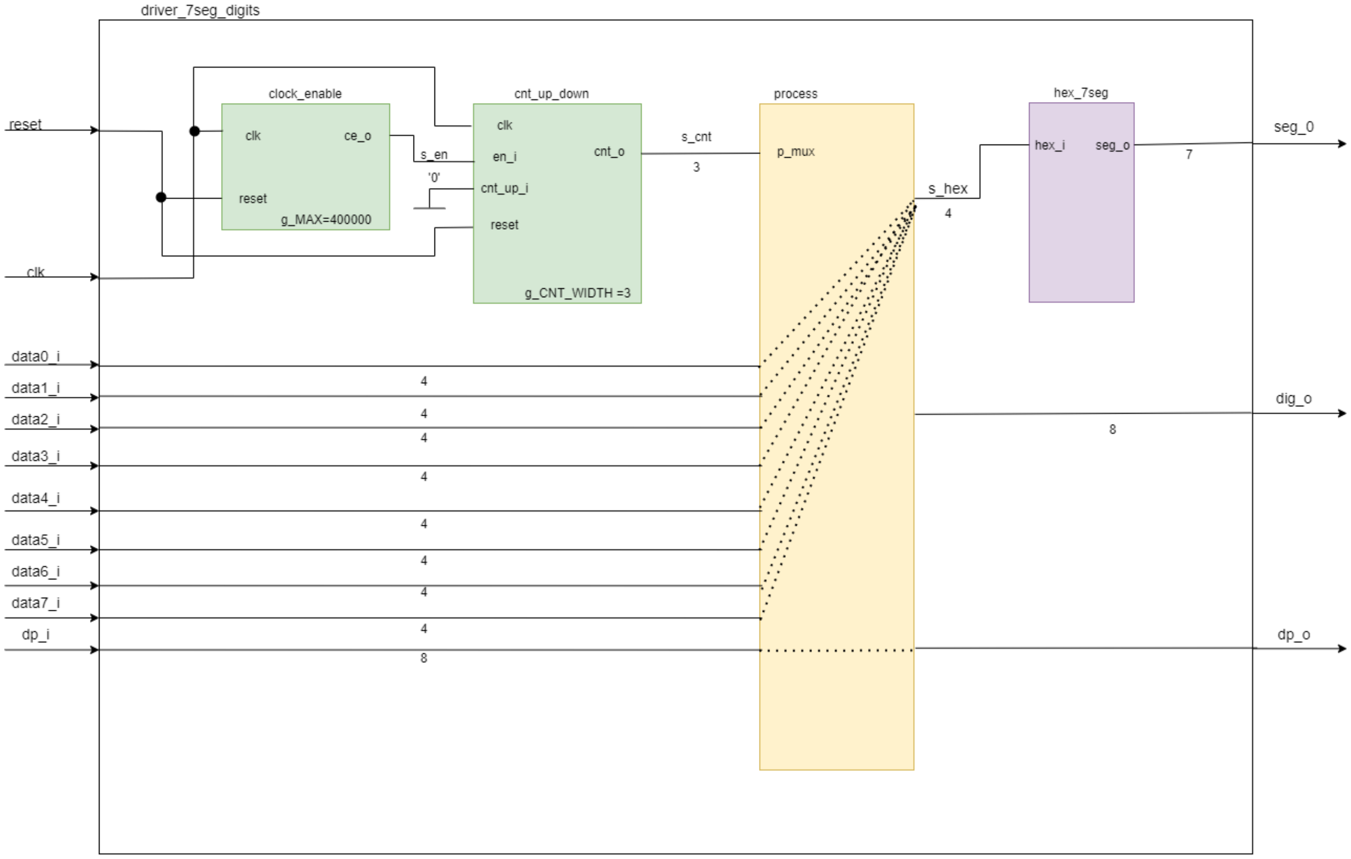


VHDL code top

```
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",
            dig_o   => AN (4 - 1 downto 0),
            seg_o(0)    => CA,
            dp_o    => DP
            --- WRITE YOUR CODE HERE
        );

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

end architecture Behavioral;
```

8-digit driver

Schema


driver_7seg_digits
reset
clk
clock_enable
clk
ce_o
reset
g_MAX=400000
s_en
'0'
cnt_up_down
clk
en_i
cnt_up_i
reset
cnt_o
g_CNT_WIDTH =3
s_cnt
3
process
p_mux
s_hex
4
hex_7seg
hex_i
seg_o
7
seg_0
data0_i
data1_i
data2_i
data3_i
data4_i
data5_i
data6_i
data7_i
dp_i
4
4
4
4
4
4
4
4
8
dig_o
8
dp_o