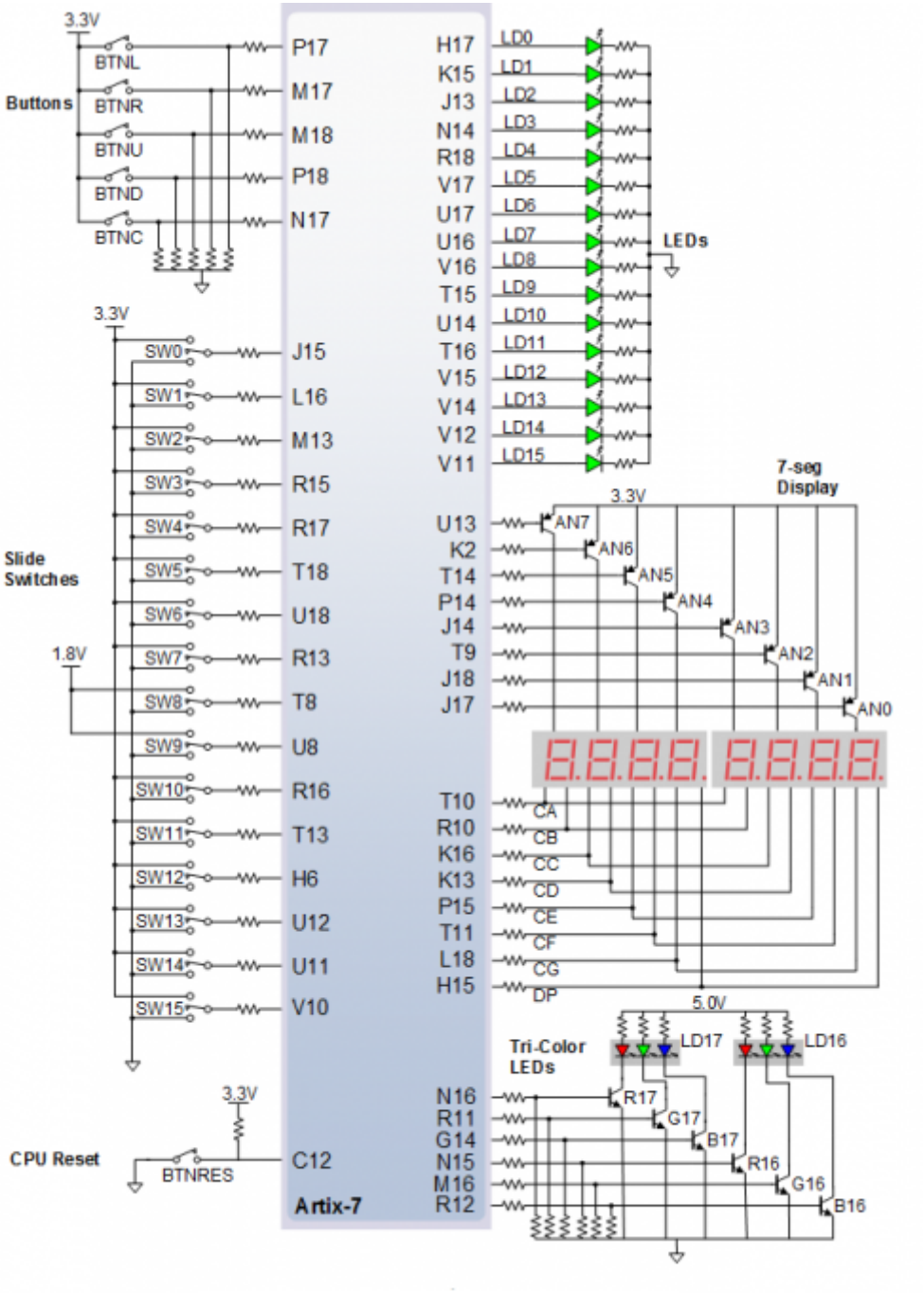
Lab 5

1. Preparation tasks

- Figure or table with connection of push buttons on Nexys A7 board, pins: P17, M17, M18, P18, N17; voltage value: 3.3



- Table with calculated values

Time interval	Number of clk periods	Number of clk periods in hex	Number of clk periods in binary
2 ms	200 000	`x"3_0d40"`	`b"0011_0000_1101_0100_0000"`
4 ms	400 000	`x"6_1A80"`	`b"0110_0001_1010_1000_0000"`
10 ms	1 000 000	`x"F4240"`	`b"01111_0100_0010_0100_0000"`

Time interval	Number of clk periods	Number of clk periods in hex	Number of clk periods in binary
250 ms	25 000 000	x"17D7840"	b"10111_1101_0111_1000_0100_0000"
500 ms	50 000 000	x"2FAF080"	b"0010_1111_1010_1111_0000_1000_0000"
1 sec	100 000 000	x"5F5_E100"	b"0101_1111_0101_1110_0001_0000_0000"

2. Bidirectional counter

- process cnt_up_down

```
p_cnt_up_down : process(clk)
begin
    if rising_edge(clk) then

        if (reset = '1') then               -- Synchronous reset
            s_cnt_local <= (others => '0'); -- Clear all bits

        elsif (en_i = '1') then       -- Test if counter is enabled
            s_cnt_local <= s_cnt_local + 1;

            -- TEST COUNTER DIRECTION HERE
        elsif (cnt_up_i = '1') then

            s_cnt_local <= s_cnt_local + 1;


        end if;
    end if;
end process p_cnt_up_down;
```

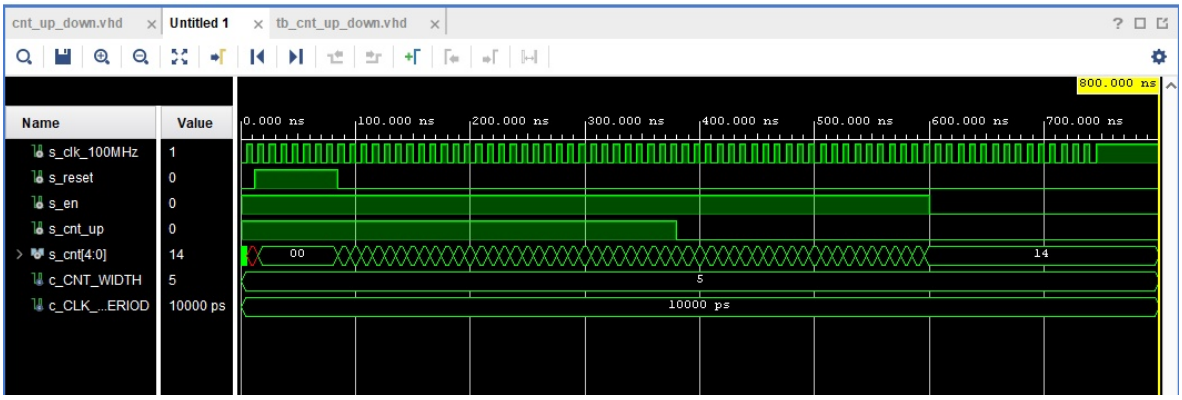
*testbench

```
architecture testbench of tb_cnt_up_down is

    -- Number of bits for testbench counter
    constant c_CNT_WIDTH         : natural := 5;
    constant c_CLK_100MHZ_PERIOD : time    := 10 ns;

    --Local signals
    signal s_clk_100MHz : std_logic;
```

```
    signal s_reset      : std_logic;
    signal s_en         : std_logic;
    signal s_cnt_up     : std_logic;
    signal s_cnt        : std_logic_vector(c_CNT_WIDTH - 1 downto 0);

begin
    -- Connecting testbench signals with cnt_up_down entity
    -- (Unit Under Test)
    uut_cnt : entity work.cnt_up_down
        generic map(
            g_CNT_WIDTH  => c_CNT_WIDTH
        )
        port map(
            clk      => s_clk_100MHz,
            reset    => s_reset,
            en_i     => s_en,
            cnt_up_i => s_cnt_up,
            cnt_o    => s_cnt
        );

    ----------------------------------------------------------------------
    -- 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;
        s_reset <= '1';                 -- Reset activated
        wait for 73 ns;
        s_reset <= '0';
        wait;
    end process p_reset_gen;

    ----------------------------------------------------------------------
    -- Data generation process
    ----------------------------------------------------------------------
    p_stimulus : process
    begin
        report "Stimulus process started" severity note;

        s_en     <= '1';                -- Enable counting
```

```
        s_cnt_up <= '1';
        wait for 380 ns;                -- Change counter direction
        s_cnt_up <= '0';
        wait for 220 ns;
        s_en     <= '0';                -- Disable counting

        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;

end architecture testbench;
```

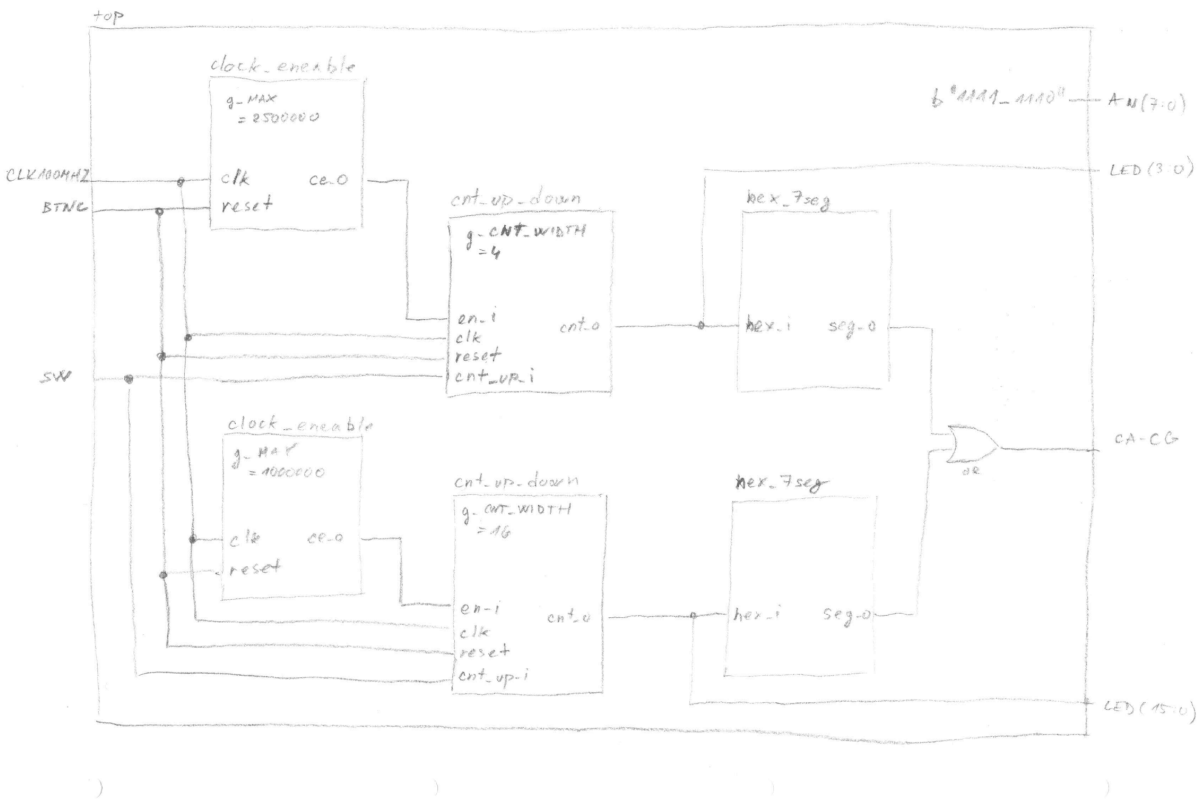
- testbench



3. Top level

```
architecture Behavioral of top is

    -- Internal clock enable
    signal s_en  : std_logic;
    -- Internal counter
    signal s_cnt : std_logic_vector(4 - 1 downto 0);

begin

    ----------------------------------------------------------------------
    -- Instance (copy) of clock_enable entity
    clk_en0 : entity work.clock_enable
        generic map(
            g_MAX   =>  25000000
            --- WRITE YOUR CODE HERE
        )
        port map(
            clk     =>  CLK100MHZ,
            reset   =>  BTNC,
            ce_o    =>  s_en
            --- WRITE YOUR CODE HERE
        );

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

```
    -- Instance (copy) of cnt_up_down entity
    bin_cnt0 : entity work.cnt_up_down
        generic map(
            g_CNT_WIDTH => 4
            --- WRITE YOUR CODE HERE
        )
        port map(
            clk     => CLK100MHZ,
            reset   => BTNC,
            en_i    => s_en,
            cnt_up_i => SW(0),
            cnt_o   => s_cnt
            --- WRITE YOUR CODE HERE
        );

    -- Display input value on LEDs
    LED(3 downto 0) <= s_cnt;

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

    -- Connect one common anode to 3.3V
    AN <= b"1111_1110";

end architecture Behavioral;
```


top
clock_enable
g_MAX
= 2500000
CLK100MHZ
BTNC
clk
reset
ce_o
cnt_up_down
g_CNT_WIDTH
= 4
en_i
clk
reset
cnt_up_i
cnt_o
hex_7seg
hex_i
seg_o
SW
clock_enable
g_MAX
= 1000000
cnt_up_down
g_CNT_WIDTH
= 16
hex_7seg
b"1111_1110"
AN(7:0)
LED(3:0)
CA-CG
OR
LED(15:0)