SimonCieslar Update Readme.md History

1 contributor

Raw Blame

185 lines (140 sloc) 6.24 KB

Labs - 05 - Counter

Experiments on your own

1. Add a second instantiation (copy) of the counter and clock enable entities and make a 16-bit counter with a 10 ms time base. Display its value on LED(15:0).

Lab assignment

1. Preparation tasks (done before the lab at home). Submit:
    - Figure or table with connection of push buttons on Nexys A7 board,
    - Table with calculated values.
2. Bidirectional counter. Submit:
    - Listing of VHDL code of the process `p_cnt_up_down` with syntax highlighting.
    - Listing of VHDL reset and stimulus processes from testbench file `tb_cnt_up_down.vhd` with syntax highlighting and asserts,
    - Screenshot with simulated time waveforms; always display all inputs and outputs,
3. Top level. Submit:
    - Listing of VHDL code from source file `top.vhd` with all instantiations for the 4-bit bidirectional counter.

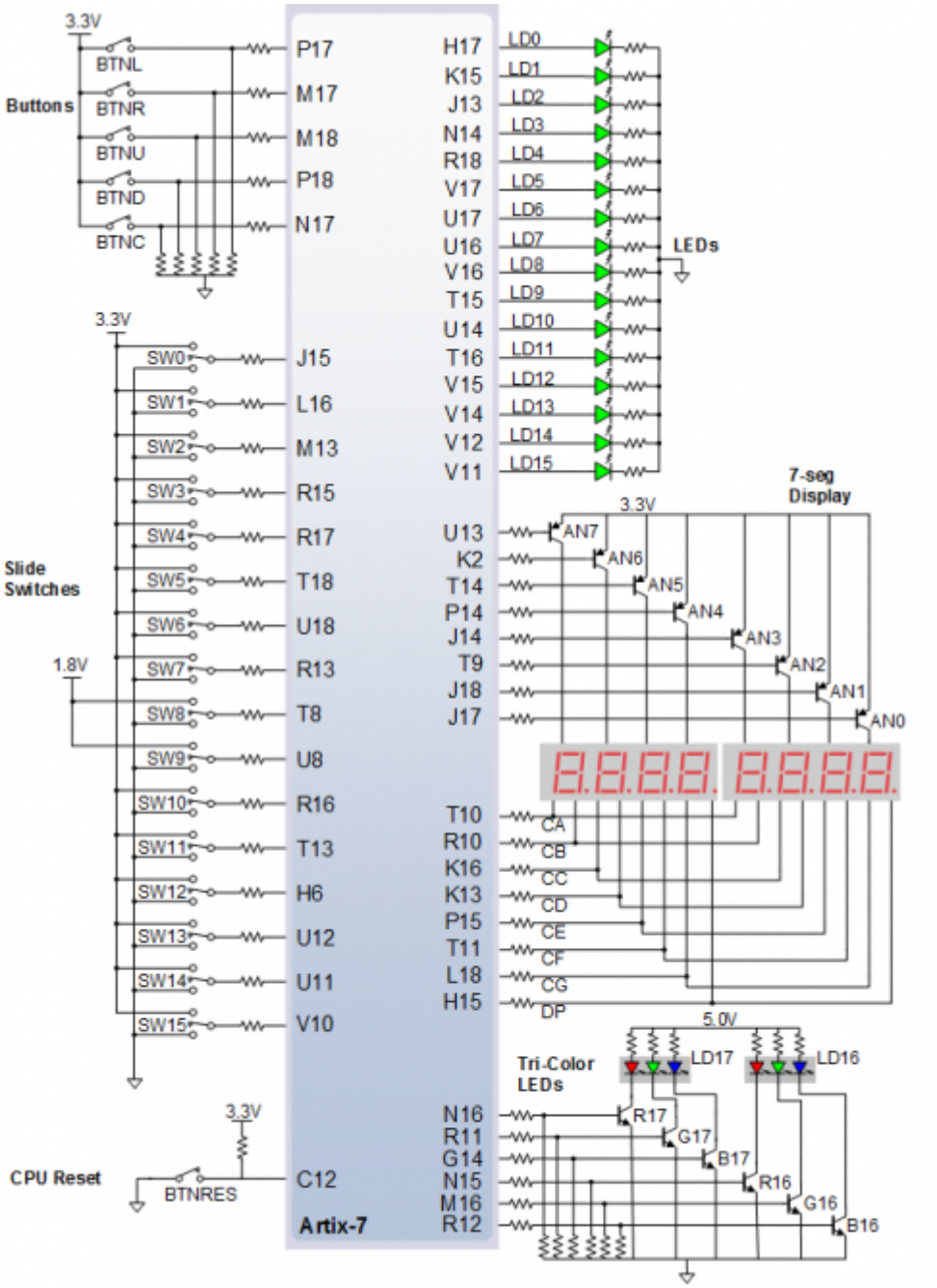
- Image of the top layer including both counters, ie a 4-bit bidirectional counter from Part 4 and a 16-bit counter with a 10 ms time base from Part Experiments on your own. The image can be drawn on a computer or by hand.

1. Preparation tasks (done before the lab at home)

1.1. Figure or table with connection of push buttons on Nexys A7 board

Button name	Voltage when on	Voltage when off	Pin
BTNL	3,3 V	0V	P17
BTNR	3,3 V	0V	M17
BTNU	3,3 V	0V	M18
BTND	3,3 V	0V	P18
BTNC	3,3 V	0V	N17
BTNRES	3,3 V	0V	C12

Circuit diagram



1.2. Table with calculated values

Calculate how many periods of clock signal with frequency of 100 MHz contain time intervals 2 ms, 4 ms, 10 ms, 250 ms, 500 ms, and 1 s. Write values in decimal, binary, and hexadecimal forms.

$$T_{clk} = \frac{1}{f_{clk}} =$$

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"`

Time interval	Number of clk periods	Number of clk periods in hex	Number of clk periods in binary
10 ms	1 000 000	x"F_4240"	b"1111_0100_0010_0100_0000"
250 ms	25 000 000	x"17D_7840"	b"0001_0111_1101_0111_1000_0100_0000"
500 ms	50 000 000	x"2FA_F080"	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

2.1. Listing of VHDL code of the process `p_cnt_up_down` with syntax highlighting

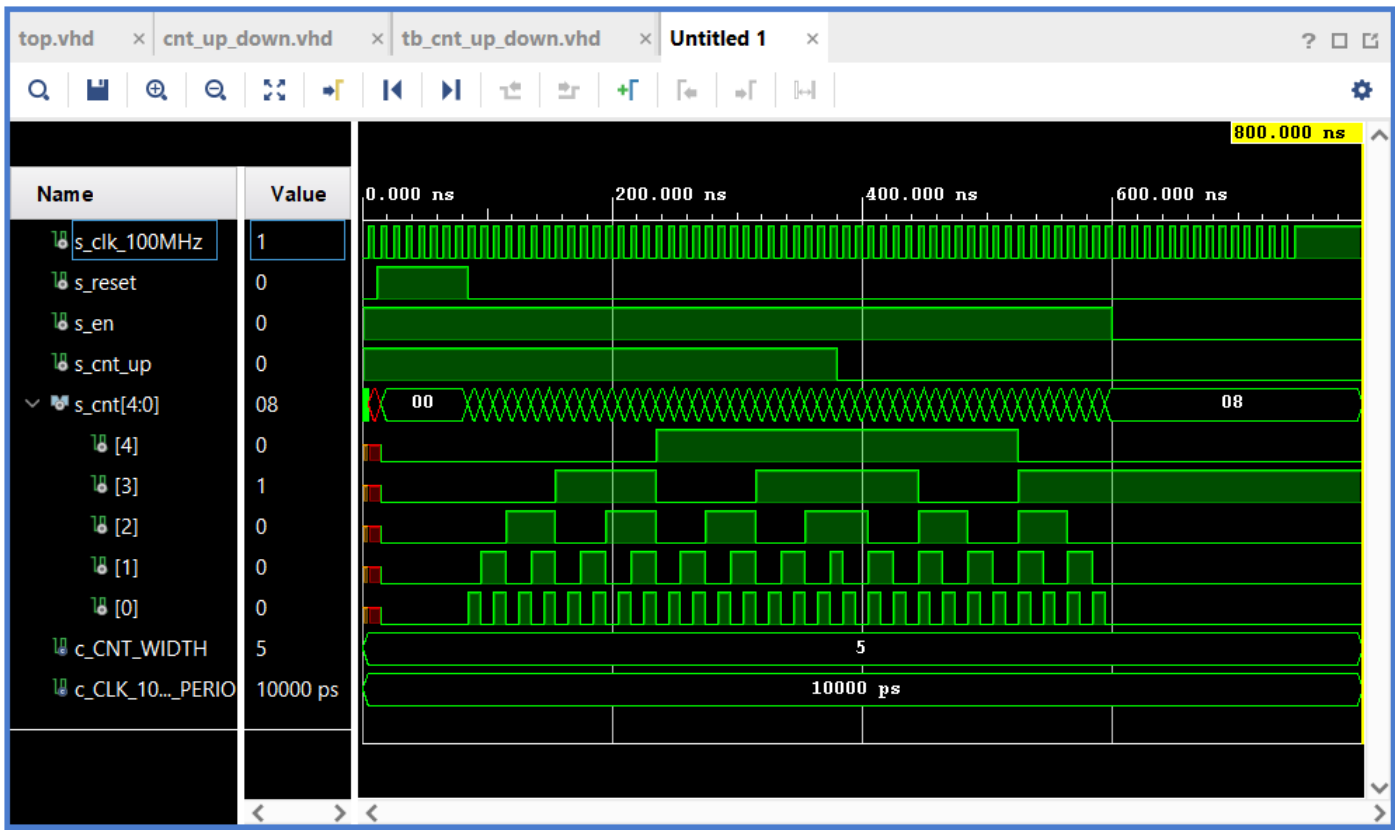
```
p_cnt_up_down : process(clk)
begin
    if rising_edge(clk) then

        if (reset = '1') then
            s_cnt_local <= (others => '0');

        elsif (en_i = '1') then

            if (cnt_up_i = '1') then

            s_cnt_local <= s_cnt_local + 1;
            else

            s_cnt_local <= s_cnt_local - 1;
            end if;

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

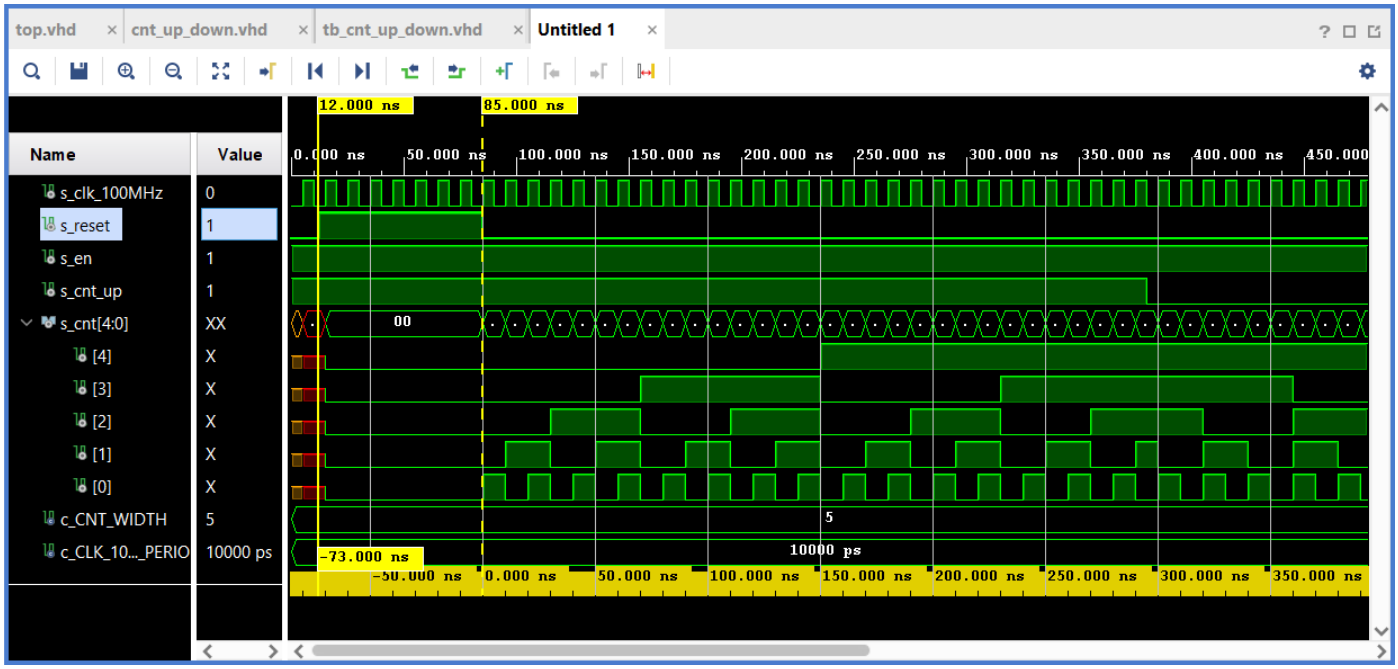
2.2. Listing of VHDL reset and stimulus processes from testbench file `tb_cnt_up_down.vhd` with syntax highlighting and asserts

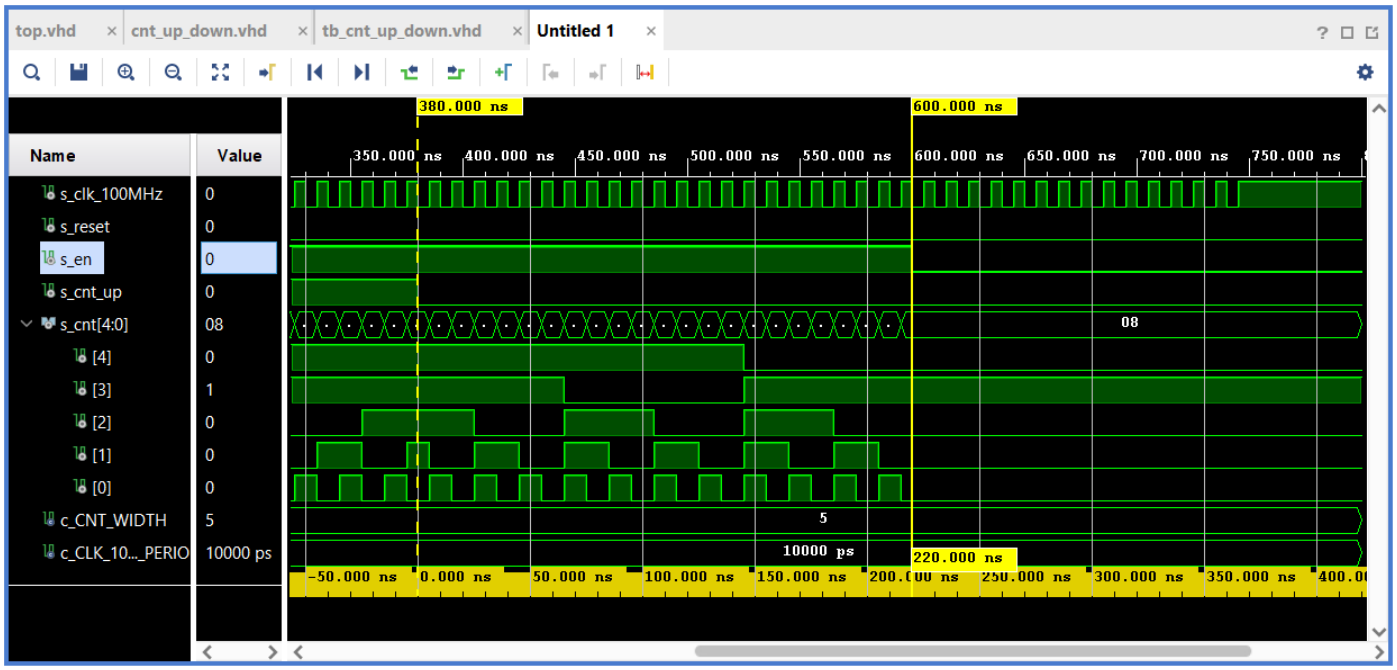
```
------------------------------------------------------------------
-- Reset generation process
------------------------------------------------------------------
p_reset_gen : process
begin
    s_reset <= '0';
    wait for 12 ns;

    -- Reset activated
    s_reset <= '1';
```

```
        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;

        -- Enable counting
        s_en     <= '1';

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

        -- Disable counting
        s_en     <= '0';

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

2.3. Screenshot with simulated time waveforms; always display all inputs and outputs







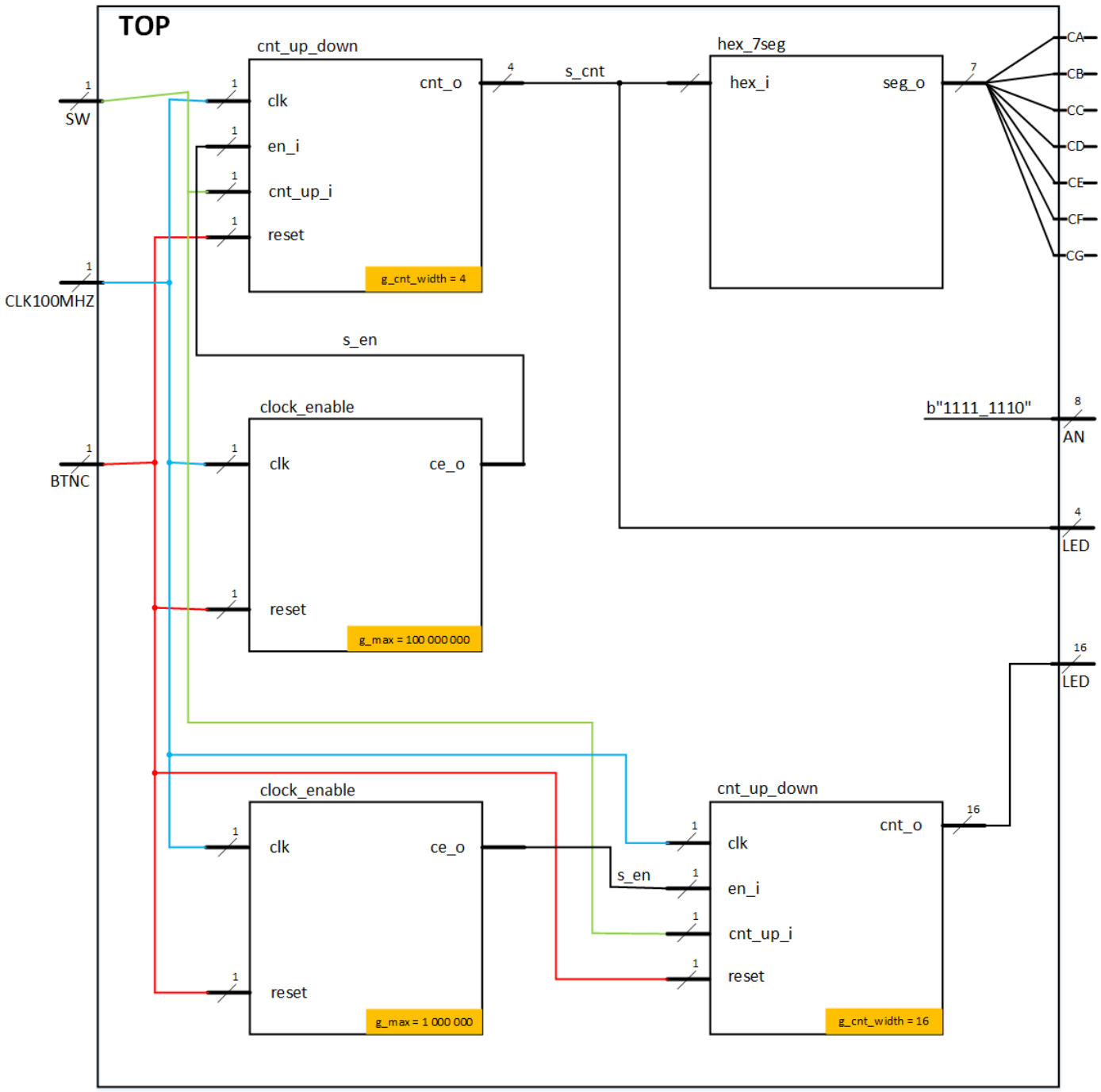
3. Top level

3.1. Listing of VHDL code from source file `top.vhd` with all instantiations for the 4-bit bidirectional counter

```
------------------------------------------------------------------------
-- Instance (copy) of clock_enable entity
clk_en0 : entity work.clock_enable
    generic map(
        g_MAX => 100000000
    )
    port map(
        clk   => CLK100MHZ,
        reset => BTNC,
        ce_o  => s_en
    );
```

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

-- 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
    );
```

3.2. Image of the top layer including both counters, ie a 4-bit bidirectional counter from Part 4 and a 16-bit counter with a 10 ms time base from Part Experiments on your own.


TOP
cnt_up_down
clk
en_i
cnt_up_i
reset
cnt_o
g_cnt_width = 4
s_cnt
hex_7seg
hex_i
seg_o
CA
CB
CC
CD
CE
CF
CG
SW
CLK100MHZ
BTNC
s_en
clock_enable
clk
reset
ce_o
g_max = 100 000 000
b"1111_1110"
AN
LED
clock_enable
clk
reset
ce_o
g_max = 1 000 000
cnt_up_down
clk
en_i
cnt_up_i
reset
cnt_o
g_cnt_width = 16
s_en
LED