

Lab 7: Stopwatch



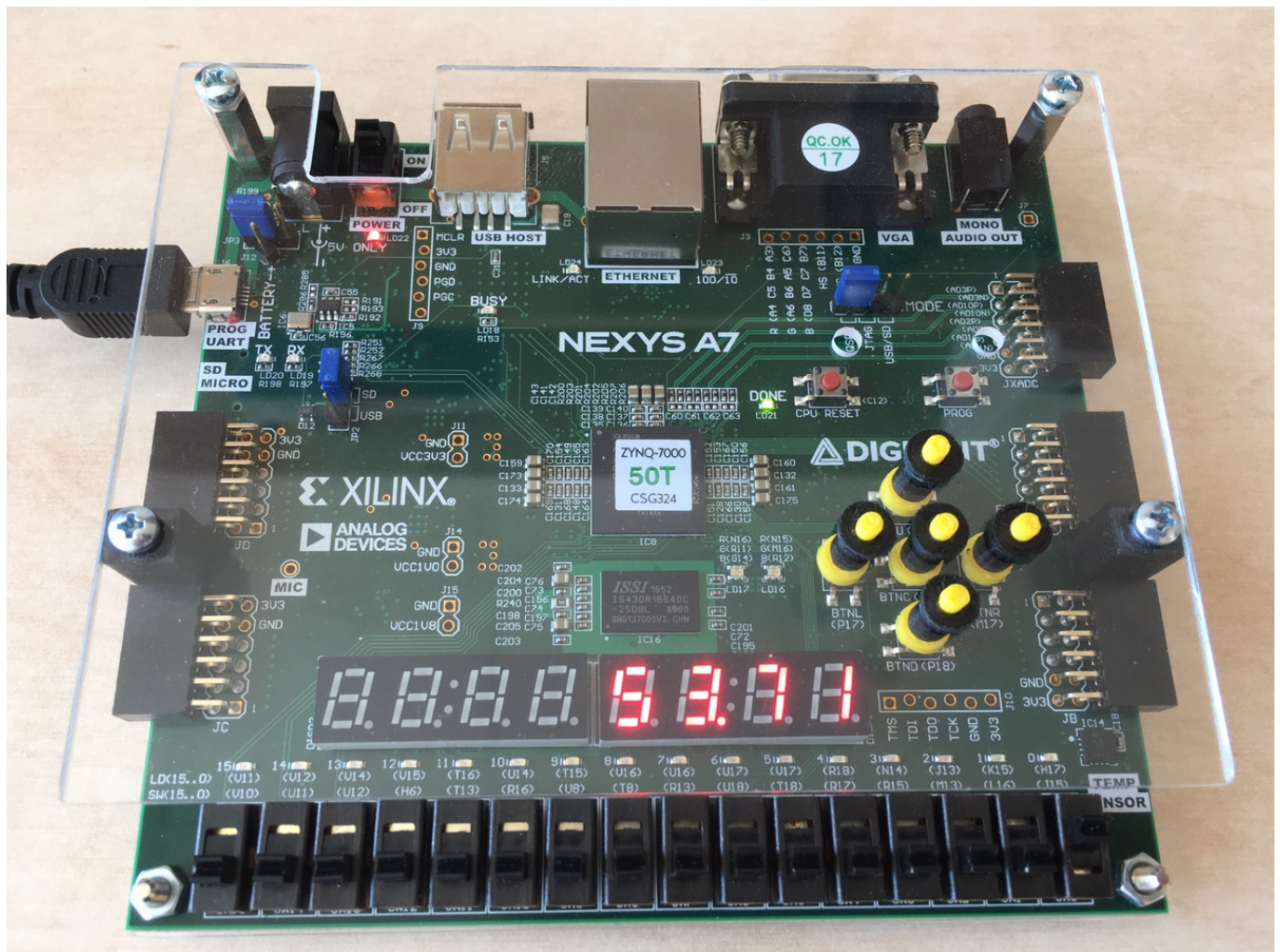
EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education



The Study of Modern and Developing Engineering BUT
CZ.02.2.69/0.0/0.0/18_056/0013325

Learning objectives

In this laboratory exercise, you will implement several decimal counters to create a stopwatch and display its value on 7-segment displays. Let the stopwatch has four places, from tens of seconds to hundreds of seconds.



Preparation tasks (done before the lab at home)

Consider the clock enable circuit from previous laboratories and find out the values of its parameter `g_MAX` so that the output signal `ce_o` has the frequency 100 Hz or 1 kHz. Let the frequency of the main clock signal be 100 MHz.

| Frequency | Number of clk periods (<code>g_MAX</code>) | Number of clk periods in hex | Number of clk periods in binary |
|-----------|---|---------------------------------|------------------------------------|
| 100 Hz | | | |
| 1000 Hz | | | |

Part 1: Synchronize repositories and create a new folder

Run Git Bash (Windows) or Terminal (Linux), navigate to your working directory, and update local repository. Create a new working folder `Labs/07-stopwatch` for this exercise.

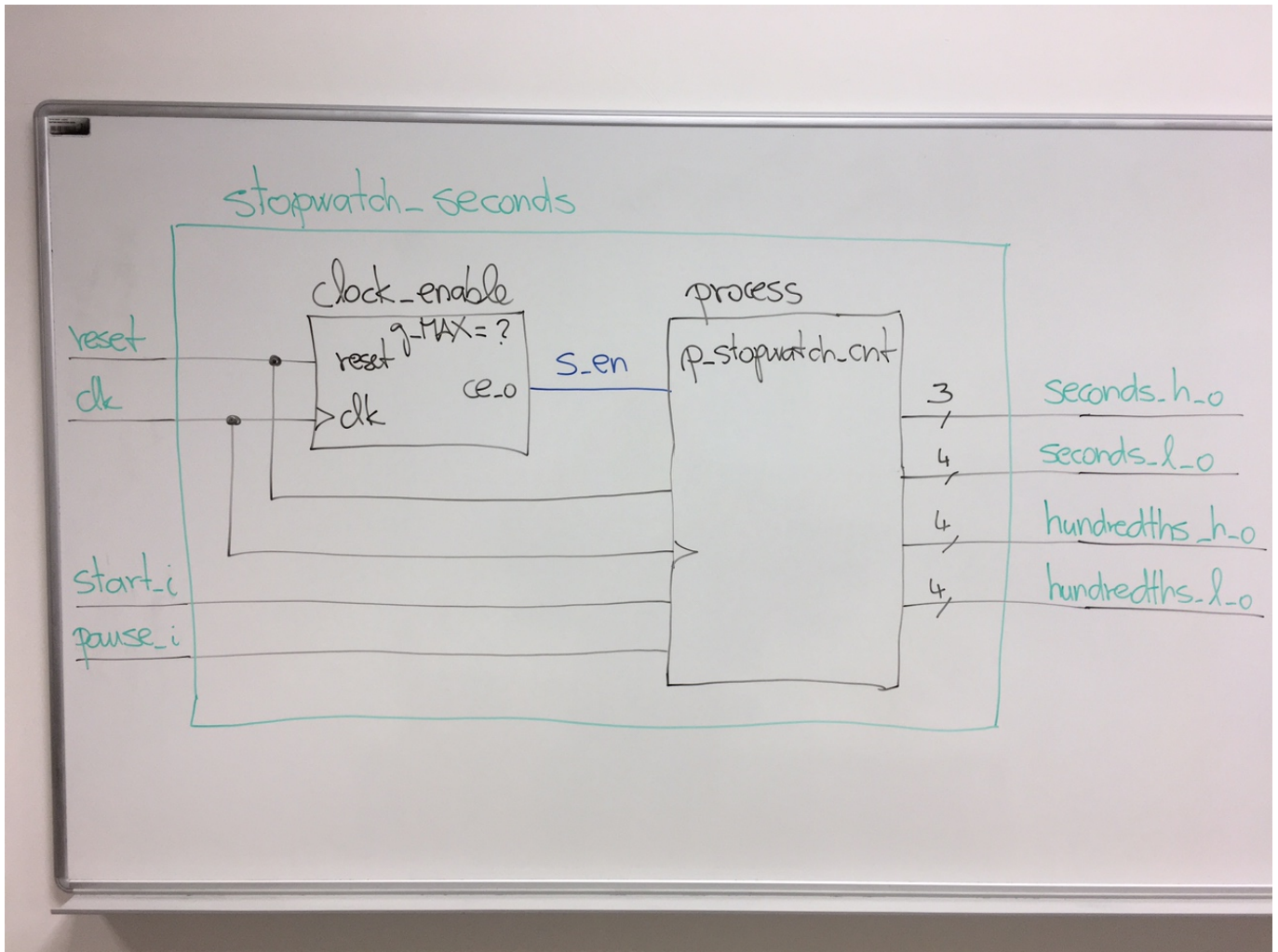
Part 2: VHDL code for stopwatch

Let the stopwatch counts the time in the form of a **seconds.hundredths** and the maximum value is `59.99` (1 minute), then the time is reset to `00.00` and the counting continues. The counter increment must be performed every 10 ms (one hundredth of a second) with the clock enable signal. In addition, the counting is started and paused by input signals. The counter value is reset by the synchronous reset input.

Use an approach that uses four different counters (one counter for each decade) and counts from 0 to 9 (or to 5). The lowest of the counters is incremented every 10 ms, and each higher-order counter is incremented if all lower-order counters are equal to the maximum value.

Perform the following steps to model the stopwatch counter.

1. Create a new Vivado RTL project `stopwatch` in your `Labs/07-stopwatch` working folder.
2. Create a VHDL source file `stopwatch_seconds` for the stopwatch circuit.
3. Choose default board: `Nexys A7-50T`.
4. Open the `Stopwatch` example and copy/paste the `design.vhd` code to your `stopwatch_seconds.vhd` file. Copy source file of clock enable circuit from previous labs to `stopwatch/stopwatch.srcs/sources_1/new/` folder and add it to the project.
5. Complete the stopwatch code according to the following block diagram.



Part 3: Stopwatch simulation

Perform the following steps to simulate stopwatch counter.

1. Create a VHDL simulation source `tb_stopwatch_seconds`, copy/paste the `testbench.vhd` code from EDA Playground example. Note that the maximum value of clock_enable circuit is set to 1 for the simulation, ie the stopwatch increments its value with a frequency of 100 MHz and not 100 Hz.
2. Change the duration of simulation to 100000ns in **Tools > Settings... > Simulation > Simulation**.
3. Complete the input data process `p_stimulus`, run the simulation, and verify that the stopwatch is started, paused, and reset correctly.

Part 4: Top level VHDL code

Perform the following steps to implement the stopwatch counter on the Nexys A7 board.

1. Create a new design source `top` in your project.
2. Define an entity `top` as follows.

| Port name | Direction | Type | Description |
|-----------|-----------|-----------|-------------------|
| CLK100MHZ | input | std_logic | Main clock |
| BTNC | input | std_logic | Synchronous reset |
| BTND | input | std_logic | Start button |

| Port name | Direction | Type | Description |
|-----------|-----------|-------------------------------------|---|
| SW | input | std_logic_vector(1 - 1 downto 0) | Pause |
| CA | output | std_logic | Cathod A |
| CB | output | std_logic | Cathod B |
| CC | output | std_logic | Cathod C |
| CD | output | std_logic | Cathod D |
| CE | output | std_logic | Cathod E |
| CF | output | std_logic | Cathod F |
| CG | output | std_logic | Cathod G |
| DP | output | std_logic | Decimal point |
| AN | output | std_logic_vector(8 - 1 downto 0) | Common anode signals to individual displays |

3. Create a new [constraints XDC](#) file: `nexys-a7-50t` and uncomment used pins according to the entity.
4. Copy source files of binary counter, 7-segment decoder, display driver modules from previous labs to `stopwatch/stopwatch.srcs/sources_1/new/` folder and add then to the project.
5. Use direct instantiation and define an architecture of the top level.

```

-----
-- Architecture body for top level
-----
architecture Behavioral of top is

    -- Local counters
    signal s_cnt3  : std_logic_vector(3 - 1 downto 0);
    signal s_cnt2  : std_logic_vector(4 - 1 downto 0);
    signal s_cnt1  : std_logic_vector(4 - 1 downto 0);
    signal s_cnt0  : std_logic_vector(4 - 1 downto 0);

begin
    -----
    -- Instance (copy) of stopwatch_seconds entity
    stopwatch_sec : entity work.stopwatch_seconds
        port map(
            clk     => CLK100MHZ,
            reset  => BTNC,
            --- WRITE YOUR CODE HERE
        );

    -----
    -- Instance (copy) of driver_7seg_4digits entity
    driver_seg_4 : entity work.driver_7seg_4digits
        port map(

```



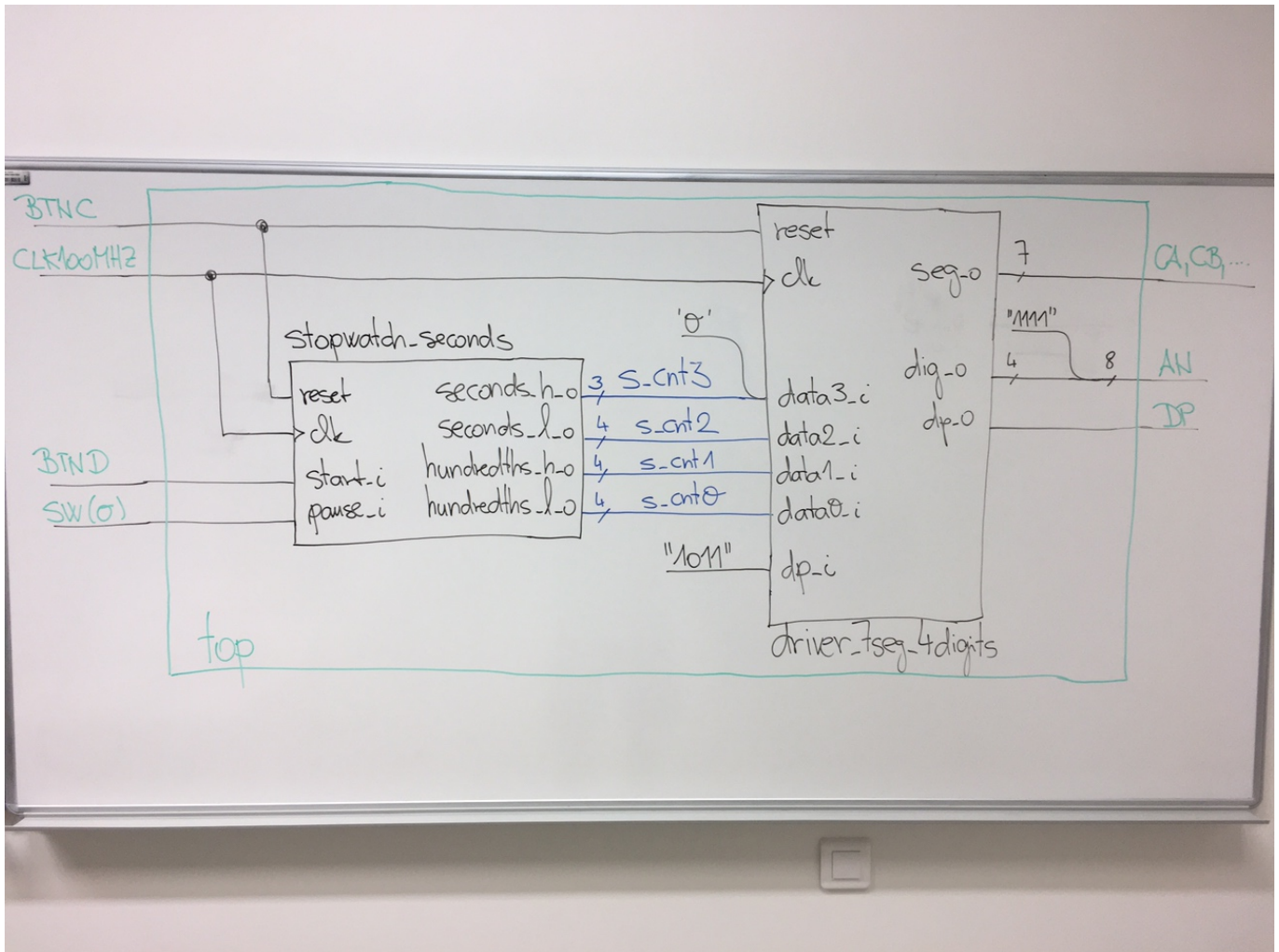
```

        clk                => CLK100MHZ,
        reset               => BTNC,
        data3_i(3)          => '0',
        data3_i(2 downto 0) => s_cnt3,
        --- WRITE YOUR CODE HERE
    );

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

end architecture Behavioral;

```



5. Compile the project and download the generated bitstream `stopwatch/stopwatch.runs/impl_1/top.bit` into the FPGA chip.
6. Test the functionality of the stopwatch by pressing the buttons, toggling the switch, and observing the display.
7. Use **IMPLEMENTATION > Open Implemented Design > Schematic** to see the generated structure.
8. Use digital oscilloscope or logic analyser and display stopwatch signals via Pmod ports. See schematic or reference manual of the Nexys A7 board and find out to which FPGA pins Pmod ports JA, JB, JC, and JD are connected.

Synchronize repositories

Use [git commands](#) to add, commit, and push all local changes to your remote repository. Check the repository at GitHub web page for changes.

Experiments on your own

1. Design the structure of `stopwatch_minutes` module, which also counts minutes to the maximum form `59.59.99` (1 hour).
2. Suggest a way to verify the accuracy of the stopwatch.

Lab assignment

1. Stopwatch. Submit:
 - VHDL code of the process `p_stopwatch_cnt`,
 - VHDL code of the simulation process `p_stimulus`,
 - Screenshot(s) of the simulation, from which it is clear that start, pause and reset work correctly,
 - VHDL code of the top layer architecture.
2. Stopwatch with minutes. Submit:
 - (Hand-drawn) sketch of the stopwatch schematic.

Prepare all parts of the assignment on a computer (not by hand), insert them in your README file [Digital-electronics-1/Labs/07-stopwatch/README.md](#), export the formatted output (not the listing in markdown language) from [HTML to PDF](#), use [BUT e-learning](#) web page and submit a single PDF file. The deadline for submitting the task is the day before the next laboratory exercise.