Simulation in ModelSim (Libero)

Created by: David Rubio G.

Blog post: https://soceame.wordpress.com/2025/03/11/simulation-in-modelsim-libero/

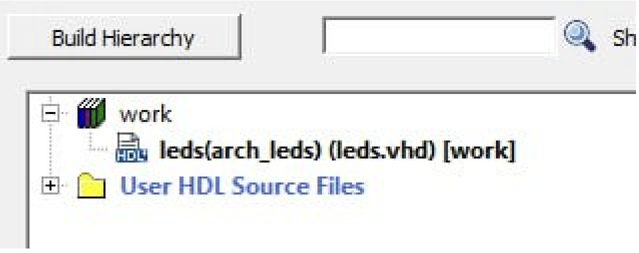
Blog: https://soceame.wordpress.com/

GitHub: https://github.com/DRubioG

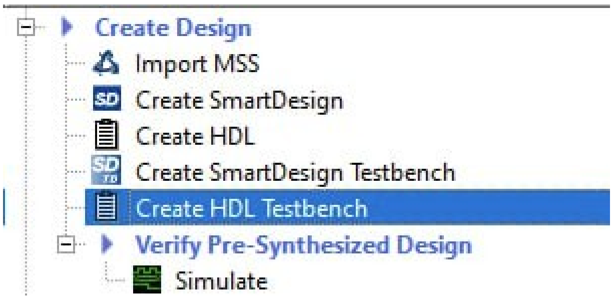
Last modification date: 11/03/25

In this post I will explain how to simulate using the ModelSim that comes included with Libero.

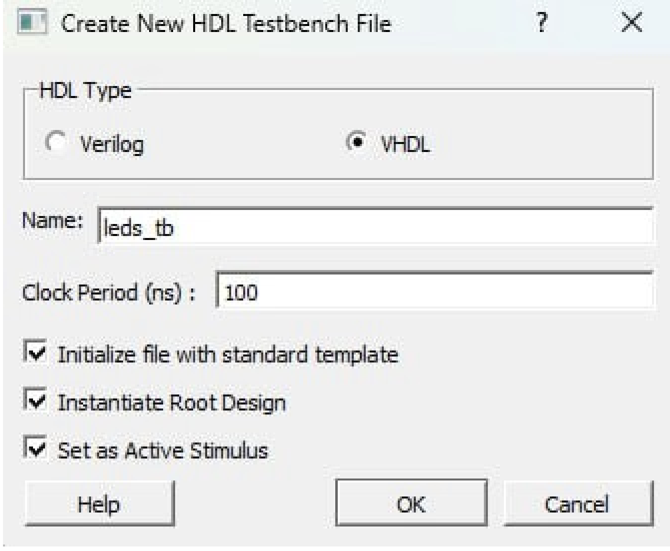
To do this, the first thing to do is to have a FW file in the Libero hierarchy.



The next thing we do is create a testbench by clicking on *Create HDL Testbench*.



It asks us for the name of the testbench.



HDL Type

○ Verilog ● VHDL

Name: leds_tb

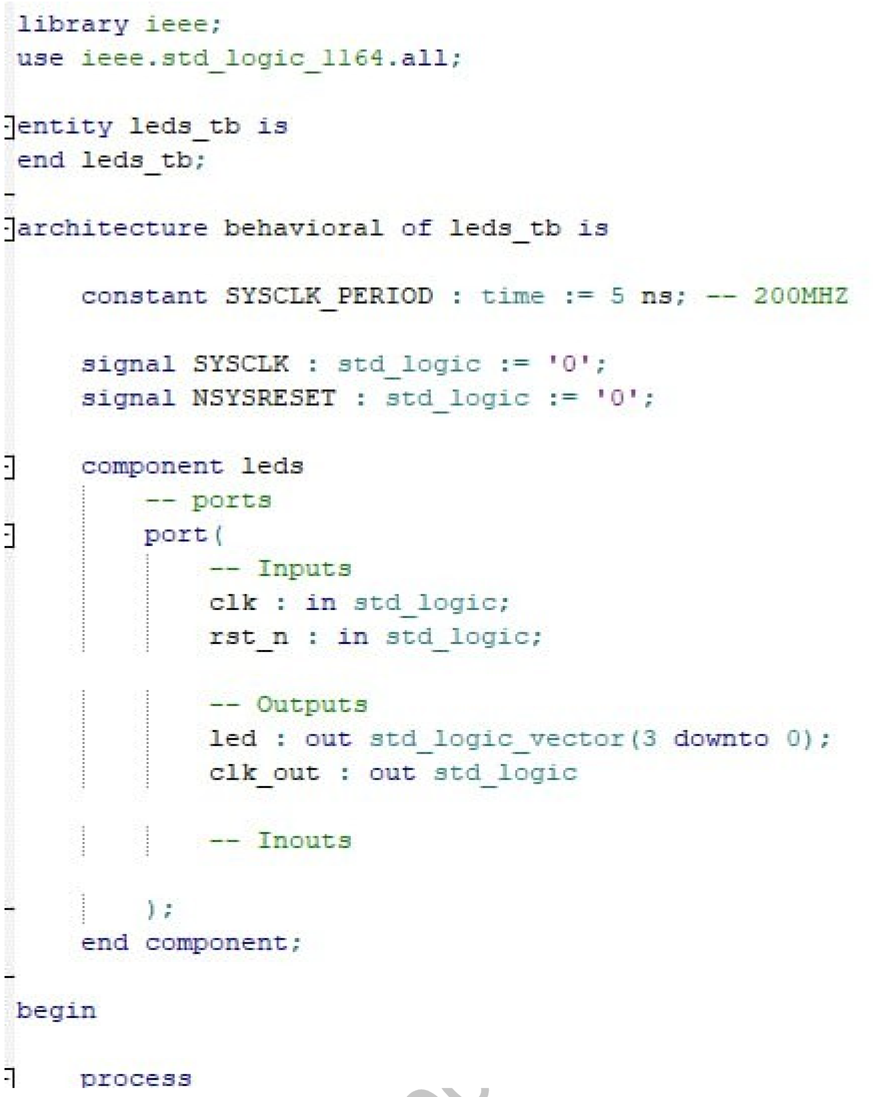
Clock Period (ns) : 100

☑ Initialize file with standard template

☑ Instantiate Root Design

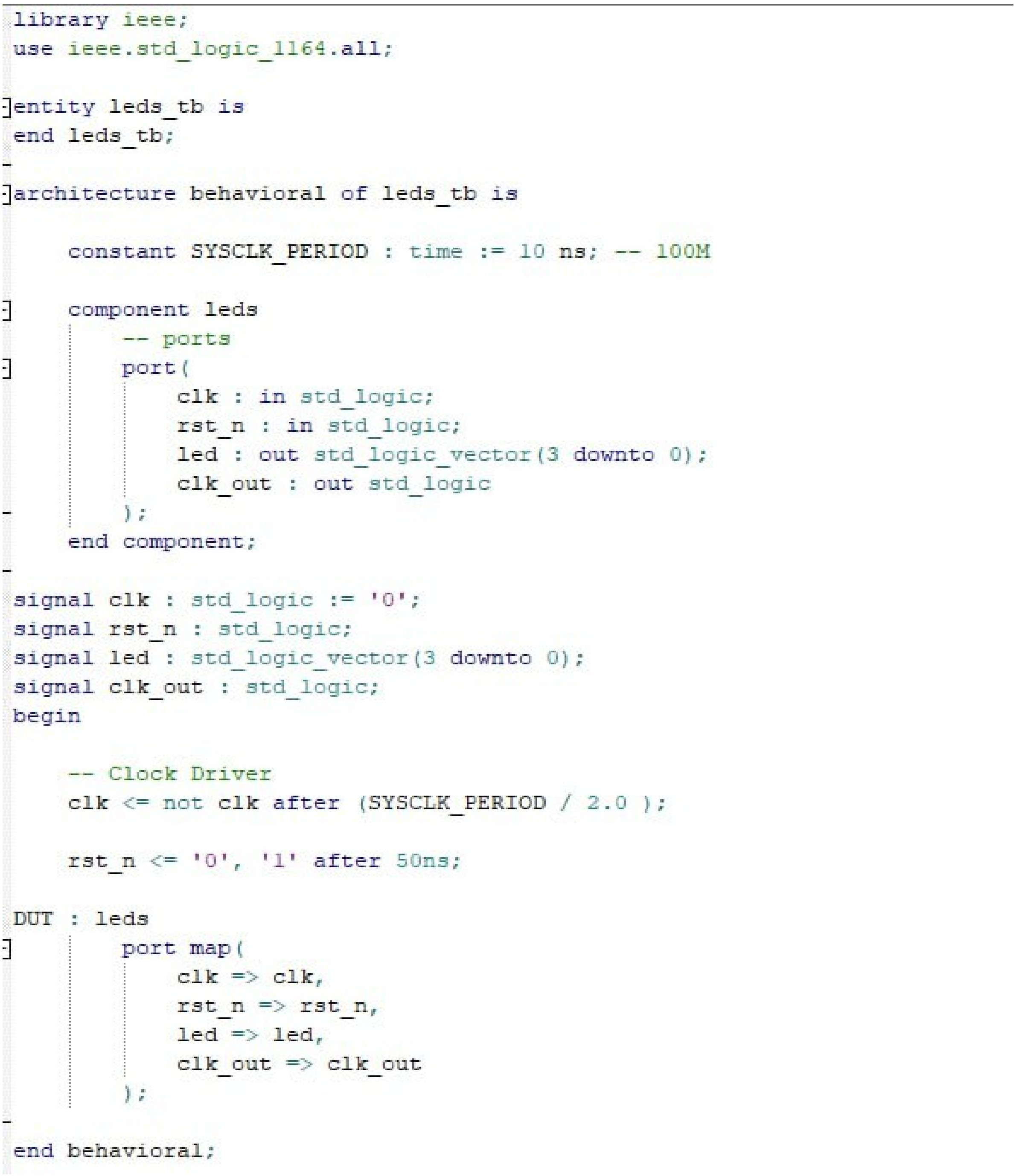
☑ Set as Active Stimulus

The first thing it does is create a testbench model.



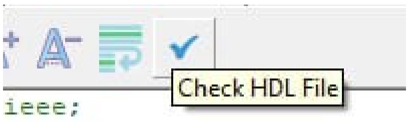
```vhdl
library ieee;
use ieee.std_logic_1164.all;

entity leds_tb is
end leds_tb;

architecture behavioral of leds_tb is

    constant SYSCLK_PERIOD : time := 5 ns; -- 200MHZ

    signal SYSCLK : std_logic := '0';
    signal NSYSRESET : std_logic := '0';

    component leds
        -- ports
        port(
            -- Inputs
            clk : in std_logic;
            rst_n : in std_logic;

            -- Outputs
            led : out std_logic_vector(3 downto 0);
            clk_out : out std_logic

            -- Inouts

        );
    end component;

begin

    process
```

We can change this model to the testbench model we want.

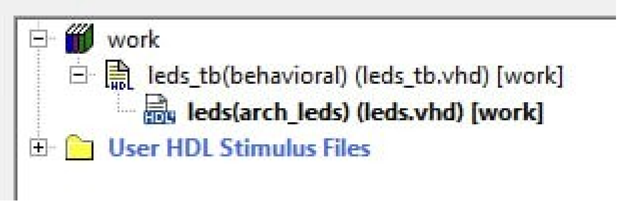


```
library ieee;
use ieee.std_logic_1164.all;

entity leds_tb is
end leds_tb;

architecture behavioral of leds_tb is

    constant SYSCLK_PERIOD : time := 10 ns; -- 100M

    component leds
        -- ports
        port(
            clk : in std_logic;
            rst_n : in std_logic;
            led : out std_logic_vector(3 downto 0);
            clk_out : out std_logic
        );
    end component;

signal clk : std_logic := '0';
signal rst_n : std_logic;
signal led : std_logic_vector(3 downto 0);
signal clk_out : std_logic;
begin

    -- Clock Driver
    clk <= not clk after (SYSCLK_PERIOD / 2.0 );

    rst_n <= '0', '1' after 50ns;

DUT : leds
        port map(
            clk => clk,
            rst_n => rst_n,
            led => led,
            clk_out => clk_out
        );

end behavioral;
```

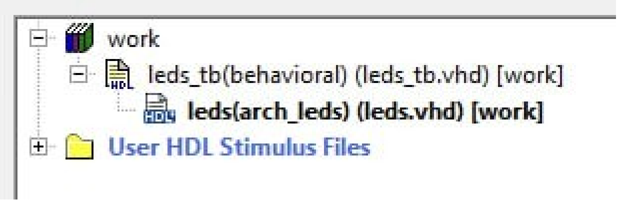
We can use the *Check HDL File* option to check if we have no errors in the code.



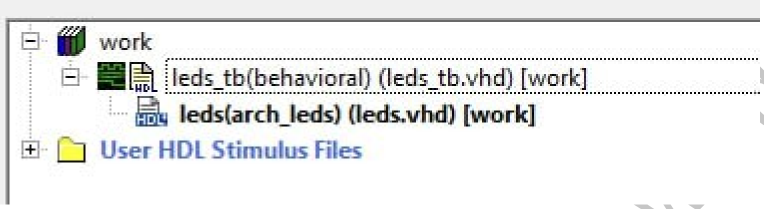
Then in the *Stimulus* tab, the structured testbench appears with the FW to simulate.


work
leds_tb(behavioral) (leds_tb.vhd) [work]
leds(arch_leds) (leds.vhd) [work]
User HDL Stimulus Files

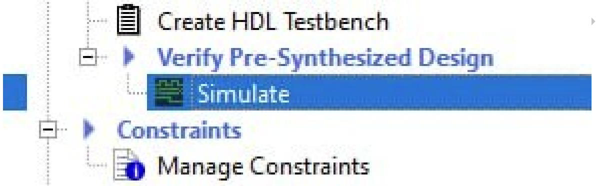
The next thing to do is to convert the testbench into the simulation file, to do this we click on *Set as active stimulus*.


work
leds_tb(behavioral) (leds_tb.vhd) [work]
leds(arch_leds) (leds.vhd) [work]
User HDL Stimulus Files

When you click on it, the simulation symbol appears next to it, this is good because we can have different simulations in a single project.


work
leds_tb(behavioral) (leds_tb.vhd) [work]
leds(arch_leds) (leds.vhd) [work]
User HDL Stimulus Files

The next step is to click *Simulate* to open ModelSim.



ModelSim will open automatically.

SIEMENS

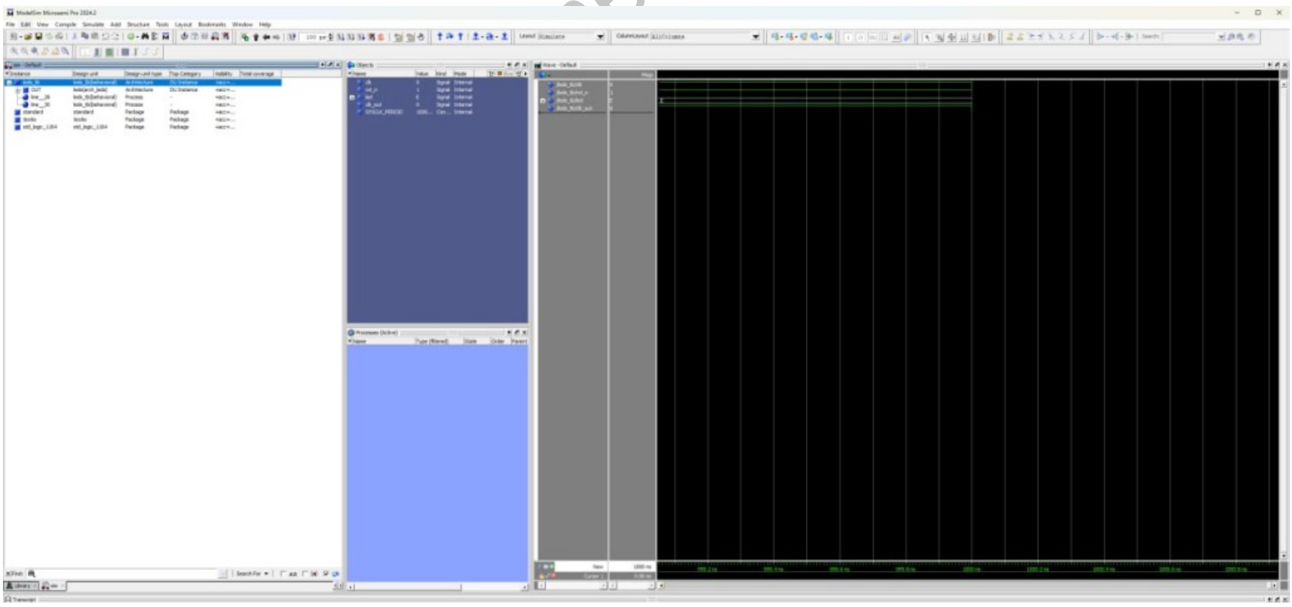
MODELSIM™

Simulation Environment

Unpublished work. ©2024 Siemens

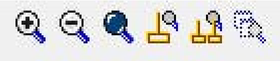
This material contains trade secrets or otherwise confidential information owned by Siemens Industry Software Inc. or its affiliates (collectively, "SISW"), or its licensors. Access to and use of this information is strictly limited as set forth in the Customer's applicable agreements with SISW.

When ModelSim opens, it appears that you have done a mini-simulation of the project.

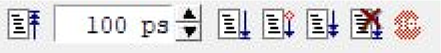

ModelSim Microsemi Pro 2024.2
File Edit View Compile Simulate Add Structure Tools Layout Bookmarks Window Help
Objects

The ModelSim controls are as follows.

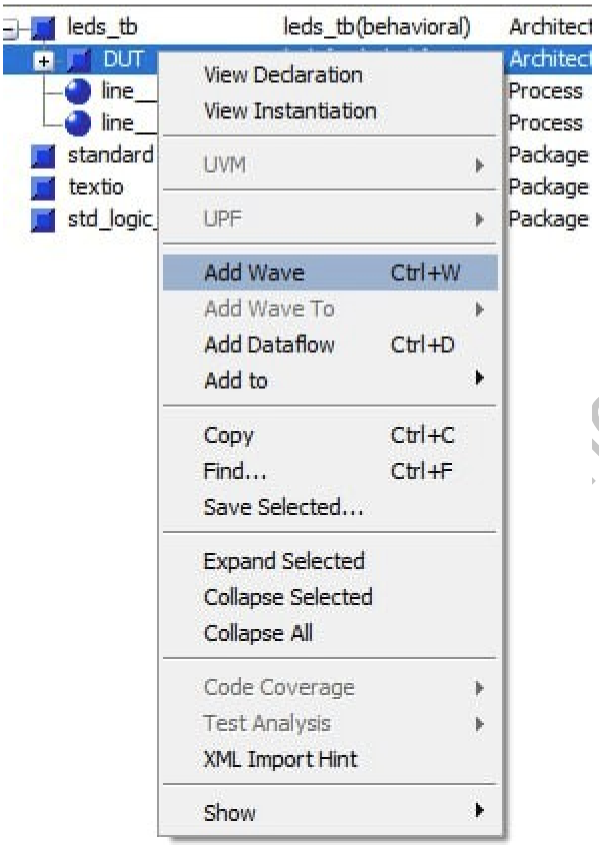
- **Visual control:** this option allows you to zoom in or put the entire simulation in a single tab (the blue magnifying glass).



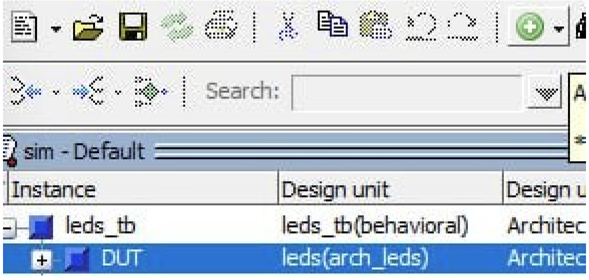
- **Simulation control:** from left to right, the first icon is the simulation reset; the next is the simulation step; the next is the simulation with the previous time; the next is the point-to-point simulation; the next is the simulation without time restriction (the simulation will not be seen until it is stopped); the next is the simulation cancellation; and the last is the simulation stop without time restriction.



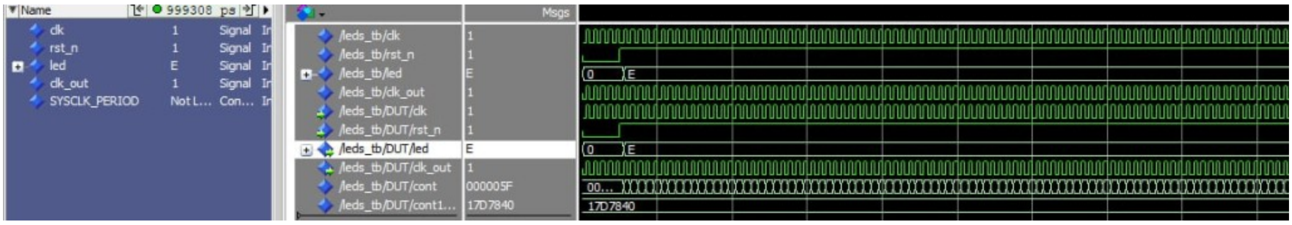
To add more ports and signals to the simulation you have two ways. Right-click the module you want to add the signals to and then *Add Wave*.



Or mark the module and click on the plus with the green symbol.



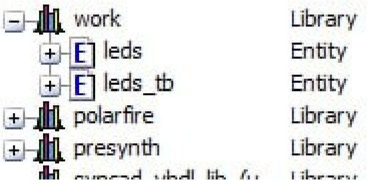
The simulation would look like this.



Name	999308 ps		
clk	1	Signal	In
rst_n	1	Signal	In
led	E	Signal	In
clk_out	1	Signal	In
SYSCLK_PERIOD	Not L...	Con...	In

	Msgs	
/leds_tb/clk	1	
/leds_tb/rst_n	1	
/leds_tb/led	E	0 E
/leds_tb/clk_out	1	
/leds_tb/DUT/clk	1	
/leds_tb/DUT/rst_n	1	
/leds_tb/DUT/led	E	0 E
/leds_tb/DUT/clk_out	1	
/leds_tb/DUT/cont	000005F	00...
/leds_tb/DUT/cont1...	17D7840	17D7840

NOTE: All files for both FW and testbench are in the work library.



work	Library
leds	Entity
leds_tb	Entity
polarfire	Library
presynth	Library

The rest of the simulations, such as post-synthesis and post-implementation, are done in the same way.