

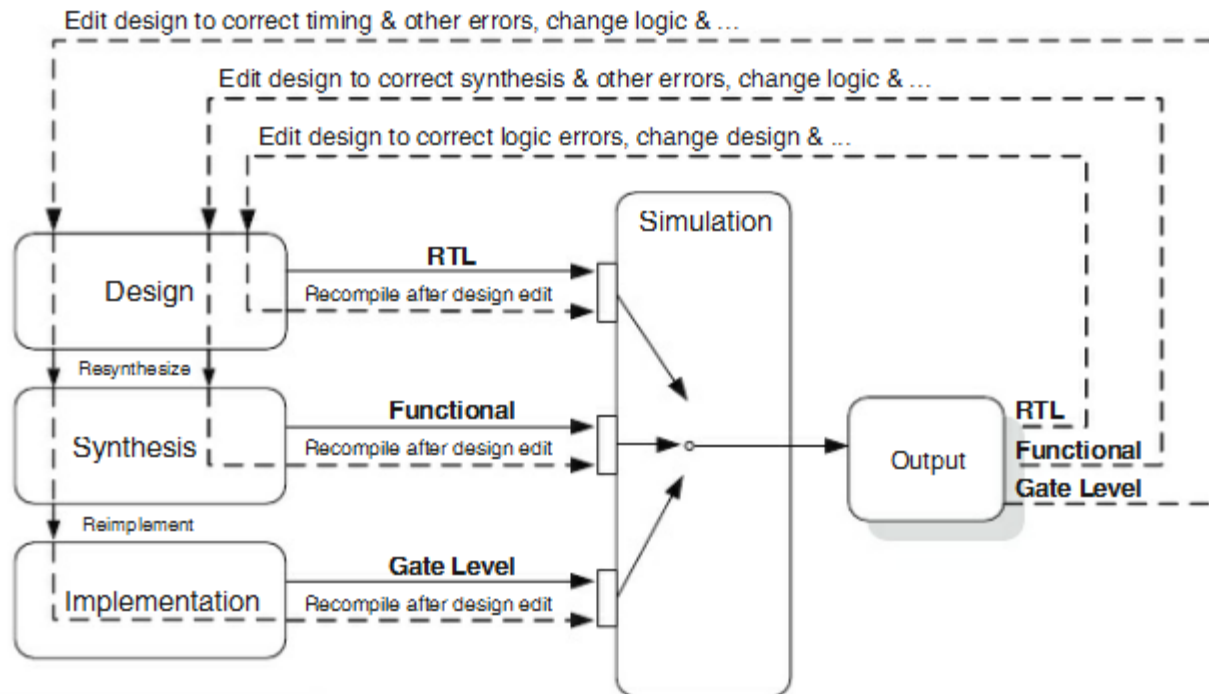
## TDC\_LAB\_2. Simulation

## TDC\_Lab\_2. Simulation

- **Simulation** is to provide **stimulus** to the input ports to verify whether the output values are as expected.
- **Simulator**: tool to test if the HDL code meets the functional requirements. Results can be shown as:
  - A data list
  - A waveform
  - A text file
- **Simulator tool in the market**:
  - Third-parties: ModelSim de Mentor Graphics
  - Open source and free (GHDL)
  - From Xilinx: **Xilinx Isim (ISE) and Vivado Simulator**

## TDC\_Lab\_2. Simulation

- Simulations can be:
  - **RTL**: a behavioral simulation
  - **Functional**: after the synthesis, the netlist is simulated. The timing information is only a estimation.
  - **Post-implementation**: The schematic targeted and the timing information are definitive



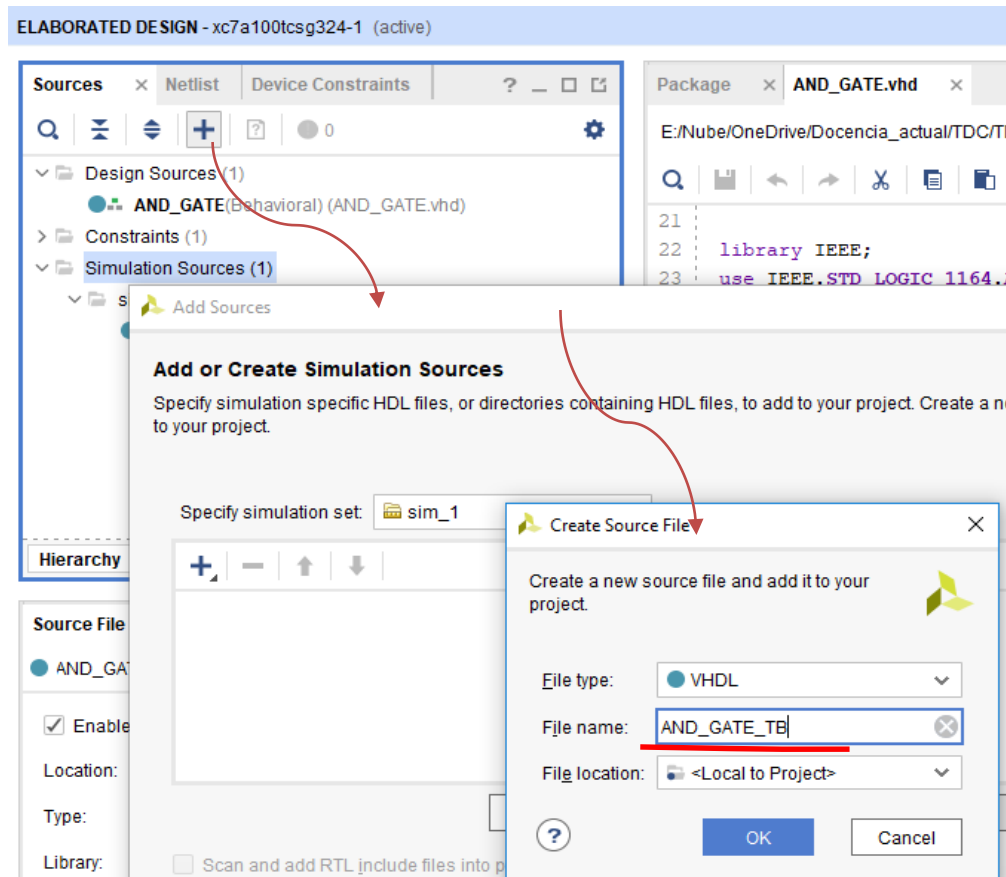
## TDC\_Lab\_2. Simulation

- Testbenches → to test the correctness of the VHDL implementations. It is a non-synthesizable VHDL file which applies a sequence of inputs (stimuli) to a circuit. Its outputs will be compared against the expected ones.
  - By means of VHDL
    - Compatible with other simulator.
    - A wider VHDL can be used
  - By means a graphical user interface (GUI)
    - Only for the simulator in use
    - A Wizard or Console commands are available

# TDC\_Lab\_2. Simulation

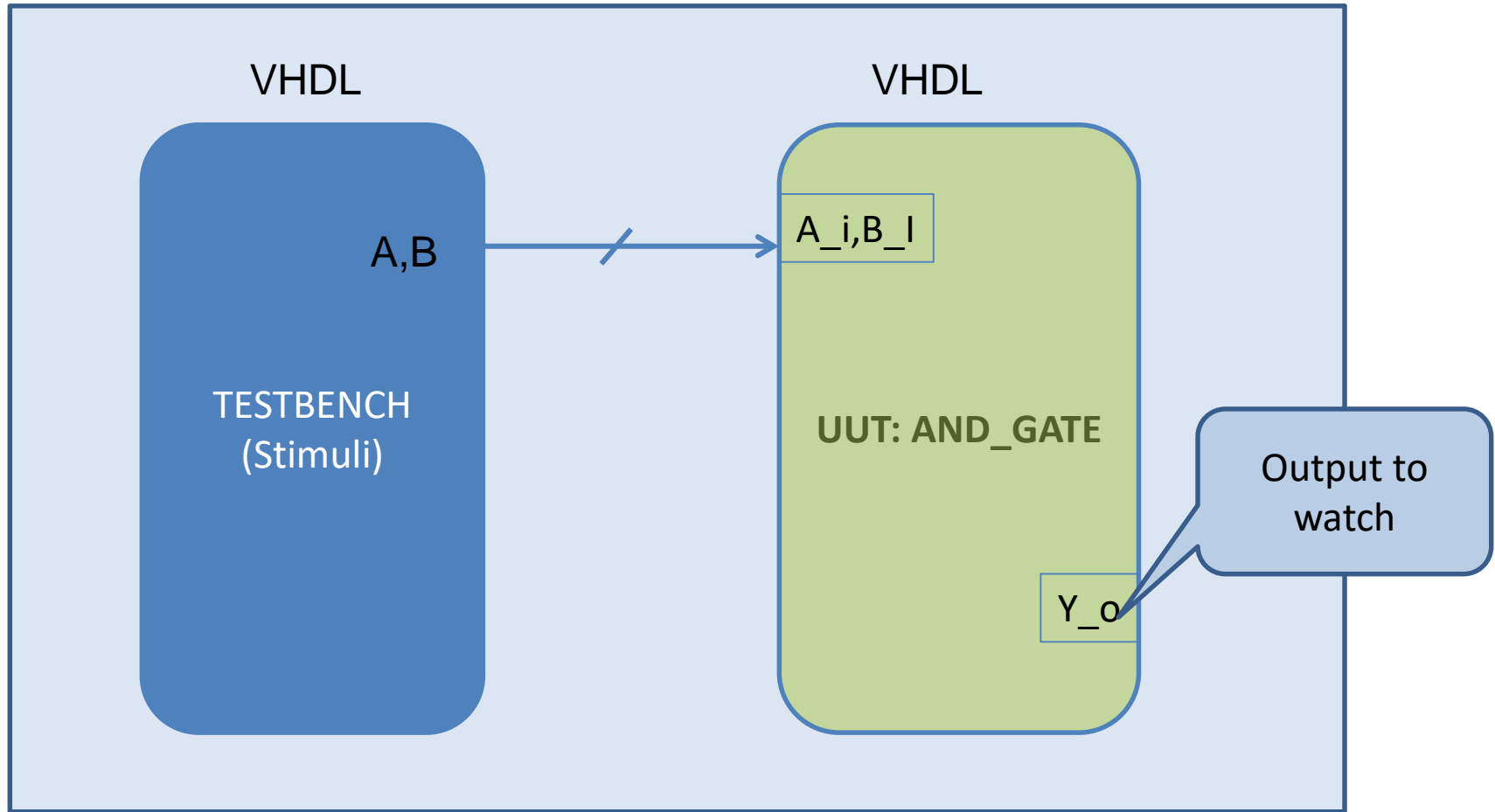
Example. How to create a testbench as a VHDL file

1. Open **PROJECT\_01. AND\_GATE**.
2. Add a *New source* → *Simulation Source*. It is recommended to name the file as the entity with the suffix “\_TB”: “**AND\_GATE\_TB**”.



# TDC\_Lab\_2. Simulation

TOP LEVEL



UUT → Unit under test

DUT: Device under test

## TDC\_Lab\_2. Simulation

Example. How to create a testbench with VHDL

3. Select and copy the VHDL code from “AND\_GATE.vhd” file.
4. Go to whatever of the below url →

[https://www.doulos.com/knowhow/perl/testbench\\_creation/](https://www.doulos.com/knowhow/perl/testbench_creation/)  
<http://vhdl.lapinoo.net/testbench/tb.php>

5. Paste the VHDL code of the circuit and click the button (Next slide)

# TDC\_Lab\_2. Simulation



**DOULOS**

Home | Company | Training | News | KnowHow | Products | myDoulos

## KnowHow

Contact Us

For more information about any of the Doulos training services

[get in touch »](#)

- VHDL
- FPGA
- Verilog
- SystemC
- TLM-2.0
- SystemVerilog
- OVM
- UVM
- VMM
- PSL
- Perl
- Tcl/Tk
- ARM / Embedded
- Video Gallery

Home > Knowhow > Perl > VHDL Testbench Creation Using Perl

### VHDL Testbench Creation Using Perl

Hardware engineers using VHDL often need to test RTL code using a testbench. Given an entity declaration writing a testbench skeleton is a standard text manipulation procedure. Each one may take five to ten minutes.

Every design unit in a project needs a testbench. Generating testbench skeletons automatically can save hours per project. However, a little Perl programming can reduce that time to seconds in future.

#### Demonstration

Our web server is set up to run Perl scripts. We've written a Perl script to generate a skeleton testbench. If an entity declaration is supplied then the Perl script will add in a Reset and Clock generator process and the testbench declaration(s) are generated too.

Copy and paste your own declarations or use our sample code below. Then click the [Generate VHDL Testbench](#) button.

```
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity AND_GATE is
  Port ( A_i : in STD_LOGIC;
        B_i : in STD_LOGIC;
        Z_o : out STD_LOGIC);
end AND_GATE;

architecture Behavioral of AND_GATE is
begin
  Z_o <= A_i and B_i; -- Logic function included in the UNISIM library
end Behavioral;
```

[Generate VHDL Testbench](#)

Paste the HDL code here

Click this button



# TDC Lab 2. Simulation

## VHDL Testbench

No reset code was detected. This may be because no architecture was supplied or because a supplied architecture

```
if Reset = '1' then
    . . .
elseif rising_edge( Clock ) then
    . . .
```

No clock code was detected. This may be because no architecture was supplied or because a supplied architecture

```
rising_edge( clock ) then

-- Testbench created online at:
-- www.doulos.com/knowhow/perl/testbench_creation/
-- Copyright Doulos Ltd
-- SD, 03 November 2002
```

```
library IEEE;
use IEEE.Std_logic_1164.all;
use IEEE.Numeric_Std.all;

entity AND_GATE_tb is
end;

architecture bench of AND_GATE_tb is

    component AND_GATE
        Port ( A_i : in STD_LOGIC;
              B_i : in STD_LOGIC;
              Z_o : out STD_LOGIC);
    end component;

    signal A_i: STD_LOGIC;
    signal B_i: STD_LOGIC;
    signal Z_o: STD_LOGIC;

begin

    uut: AND_GATE port map ( A_i => A_i,
                           B_i => B_i,
                           Z_o => Z_o );

    stimulus: process
    begin

        -- Put initialisation code here

        -- Put test bench stimulus code here

        wait;
    end process;

end;
```

A new window is open with the testbench template

Select and copy only  
the text into the  
rectangle

# TDC Lab 2. Simulation

Package x AND\_GATE.vhd x AND\_GATE\_TB.vhd\*

E:/Nube/OneDrive/Docencia\_actual/TDC/TDC\_PROJECTS\_VIVADO/Proje... Project

Q [Icons]

```
3 use IEEE.Numeric_Std.all;
4
5 entity AND_GATE_tb is
6 end;
7
8 architecture bench of AND_GATE_tb is
9
10 component AND_GATE
11     Port ( A_i : in STD_LOGIC;
12           B_i : in STD_LOGIC;
13           Z_o : out STD_LOGIC);
14 end component;
15
16 signal A_i: STD_LOGIC;
17 signal B_i: STD_LOGIC;
18 signal Z_o: STD_LOGIC;
19
20 begin
21
22 uut: AND_GATE port map ( A_i => A_i,
23                           B_i => B_i,
24                           Z_o => Z_o );
25
26 stimulus: process
27 begin
```

Open “AND\_GATE\_TB.vhd”  
in Vivado and paste the TB  
template generated.

## TDC\_Lab\_2. Simulation

Analyzing a testbench...

```
library IEEE;  
use IEEE.Std_logic_1164.all;  
use IEEE.Numeric_Std.all;
```

1

```
entity AND_GATE_tb is  
end;
```

There are no ports in the TB entity

```
architecture bench of AND_GATE_tb is  
  component AND_GATE  
    Port ( A_i : in STD_LOGIC;  
          B_i : in STD_LOGIC;  
          Z_o : out STD_LOGIC);  
  end component;
```

The UUT is instantiated as a component

## TDC\_Lab\_2. Simulation

```
signal A_i: STD_LOGIC;  
signal B_i: STD_LOGIC;  
signal Z_o: STD_LOGIC;  
  
begin uut: AND_GATE  
    port map (  
        A_i => A_i,  
        B_i => B_i,  
        Z_o => Z_o );
```

Signals to connect the TB to UUT are declared.  
(They can be initialized)

The instance of the component, “uut”, is mapped

## TDC\_Lab\_2. Simulation

```
stimulus: process
```

```
begin
```

```
-- Put initialisation code here
```

Write the initial values  
to apply to the inputs

```
-- Put test bench stimulus code here
```

```
wait;
```

```
end process;
```

Write the necessary  
stimuli to test the  
functional requirements

```
end;
```

## TDC\_Lab\_2. Simulation

```
-- Stimulus process  
stim_proc: process  
begin
```

Write the following stimuli

```
-----  
-- Combination 1: It lasts 40ns  
-----
```

```
A_i <='0';  
B_i <='0';  
wait for 40 ns;
```

T0 = 0 ns

```
-----  
-- Combination 2: It lasts 40ns  
-----
```

```
A_i <='0';  
B_i <='1';  
wait for 40 ns;
```

T1 = 40 ns

```
-----  
-- Combination 3: It lasts 40ns  
-----
```

```
A_i <='1';  
B_i <='0';  
wait for 40 ns;
```

T2 = 80 ns

## TDC\_Lab\_2. Simulation

```
-----  
-- Combination 4: It lasts 40ns  
-----
```

```
A_i <= '1';  
B_i <= '1';  
wait for 40 ns;
```

T0 = 120 ns

```
wait;  
end process;
```

```
end;
```

The process halts, never repeat. To achieve a loop, erase that line.

# TDC\_Lab\_2. Simulation

Notice the TB added to “Simulation sources

The screenshot shows the Xilinx Project Manager interface for 'Project\_01'. The left sidebar contains the 'Flow Navigator' with sections for 'PROJECT MANAGER', 'IP INTEGRATOR', 'SIMULATION', 'RTL ANALYSIS', and 'SYNTHESIS'. The 'SIMULATION' section is expanded, showing 'Run Simulation'. The main 'Sources' window on the right lists the project's files, including 'Design Sources', 'Constraints', and 'Simulation Sources'. Under 'Simulation Sources', the file 'AND\_GATE\_TB.vhd' is circled in red. The 'Source File Properties' window at the bottom shows the selected file's details, including its location and type (VHDL).

Flow Navigator

- PROJECT MANAGER
  - Settings
  - Add Sources
  - Language Templates
  - IP Catalog
- IP INTEGRATOR
  - Create Block Design
  - Open Block Design
  - Generate Block Design
- SIMULATION
  - Run Simulation
- RTL ANALYSIS
  - Open Elaborated Design
- SYNTHESIS
  - Run Synthesis
  - Open Synthesized Design

PROJECT MANAGER - Project\_01

Sources

- Design Sources (1)
  - AND\_GATE(Behavioral) (AND\_GATE.vhd)
- Constraints (1)
  - constrs\_1 (1)
    - AND\_GATE.xdc (target)
- Simulation Sources (1)
  - sim\_1 (1)
    - AND\_GATE\_TB.vhd (1)
    - uut: AND\_GATE(Behavioral) (AND\_GATE.vhd)

Hierarchy Libraries Compile Order

Source File Properties

AND\_GATE\_TB.vhd

Enabled

Location: E:/Nube/OneDrive/Docencia\_actual/TDC/TDC\_F

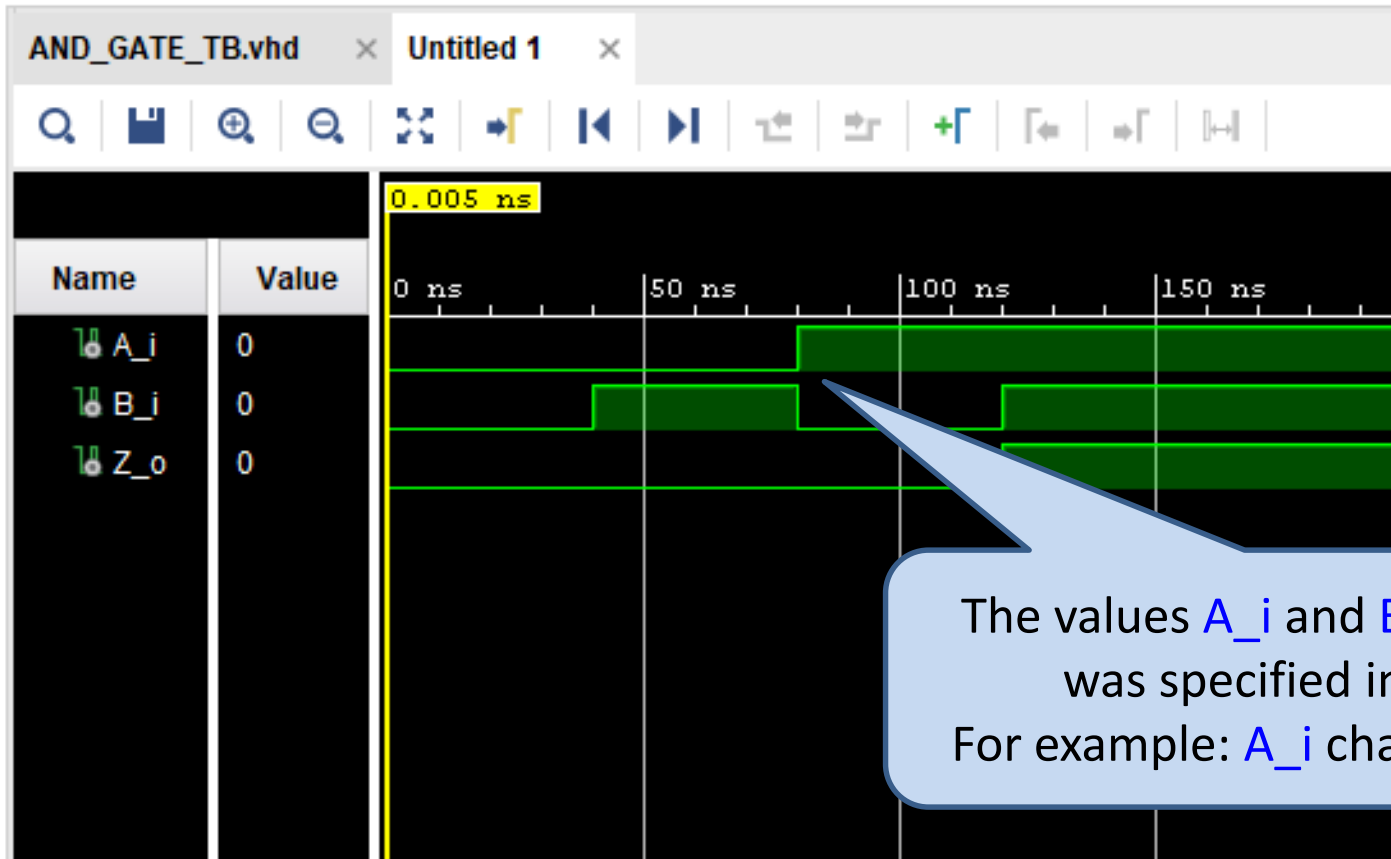
Type: VHDL

Click to  
run the  
behavioral  
simulation



## TDC\_Lab\_2. Simulation

The simulator opens in a new window and the stimuli applied and the outputs resulting are shown in a waveform.



Look at the waveform and check if "Z\_o" takes the expected values

# TDC\_Lab\_2. Simulation

Create and add a TB to test the XOR gate from PROJECT\_03 (With Generics)

1. Follow the previously explained steps and modify the testbench as below

```
entity TB_XOR_GATE_tb is
end;
architecture bench of TB_XOR_GATE_tb is
    component XOR_GATE
generic (WIDTH: integer:= 2);
    Port ( A_i : in STD_LOGIC_VECTOR (WIDTH-1 downto 0);
          B_i : in STD_LOGIC_VECTOR (WIDTH-1 downto 0);
          Z_o : out STD_LOGIC_VECTOR (WIDTH-1 downto 0));
end component;

constant WIDTH: integer:=2;
signal A_i: STD_LOGIC_VECTOR (WIDTH-1 downto 0);
signal B_i: STD_LOGIC_VECTOR (WIDTH-1 downto 0);
signal Z_o: STD_LOGIC_VECTOR (WIDTH-1 downto 0);

begin -- Insert values for generic parameters !!
    uut: XOR_GATE
        generic map ( WIDTH => WIDTH )
        port map ( A_i => A_i, B_i => B_i, Z_o => Z_o );
    stimulus: process
    Begin
        -- Put initialisation code here -- Put test bench stimulus code here
    wait;
end process;
```

A constant for every generic

Assign the “constant” to the “generic map”.

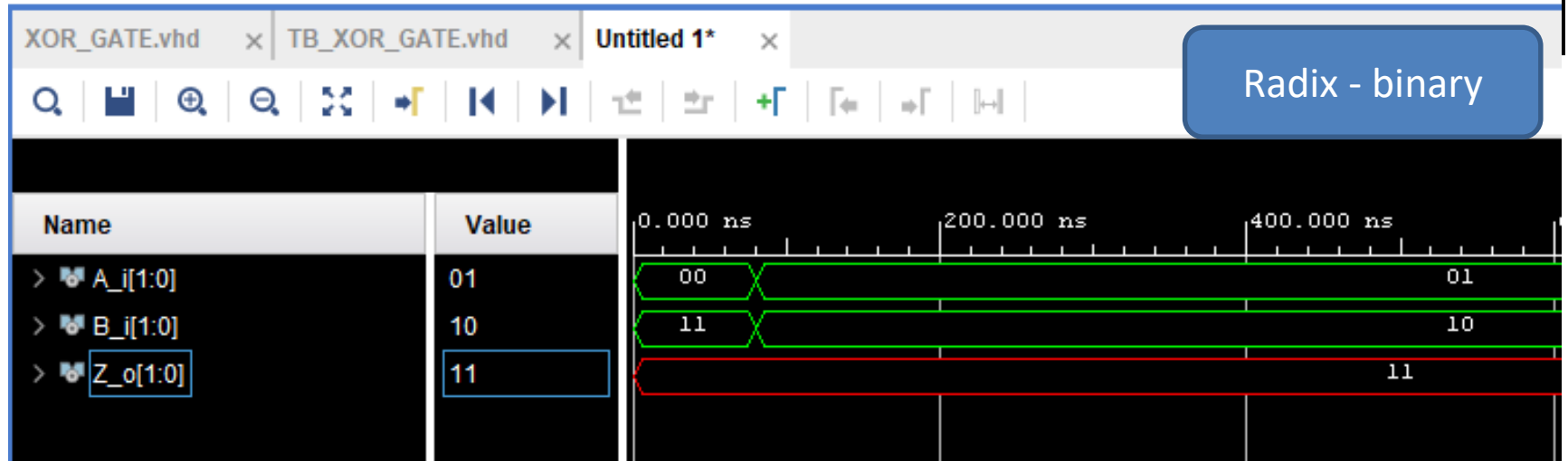
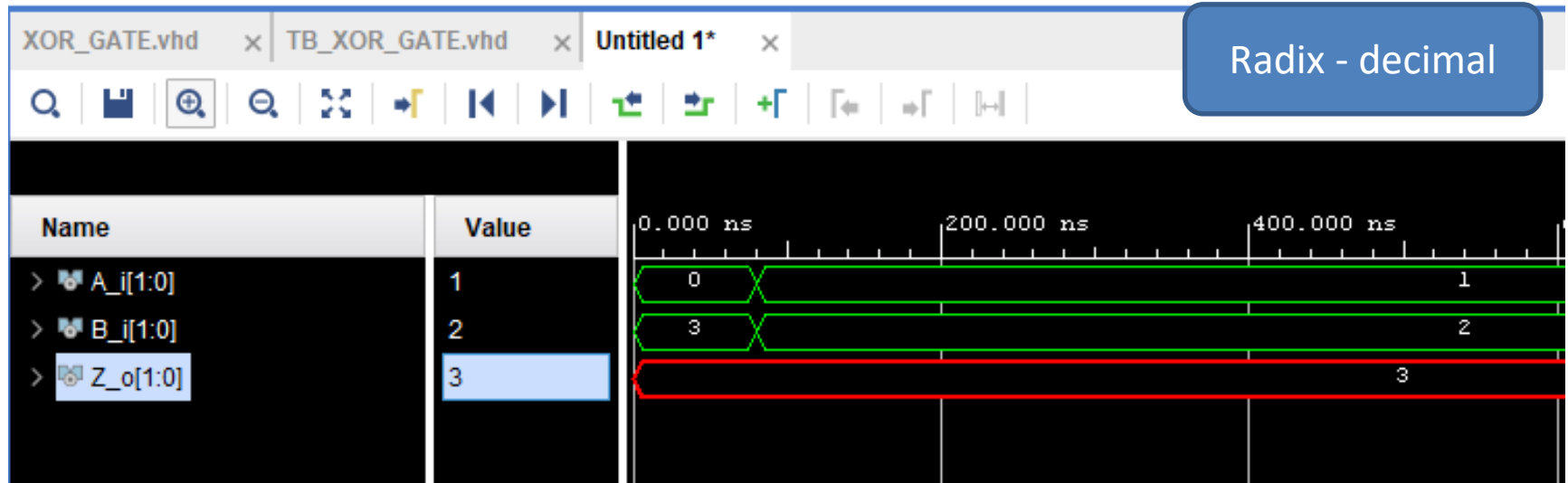
## TDC\_Lab\_2. Simulation

```
A_B: process
begin
  A_i<="00";
  B_i<="11";
  wait for 80 ns;

  A_i<="01";
  B_i<="10";
  wait for 80 ns;
end process;

end architecture bench;
```

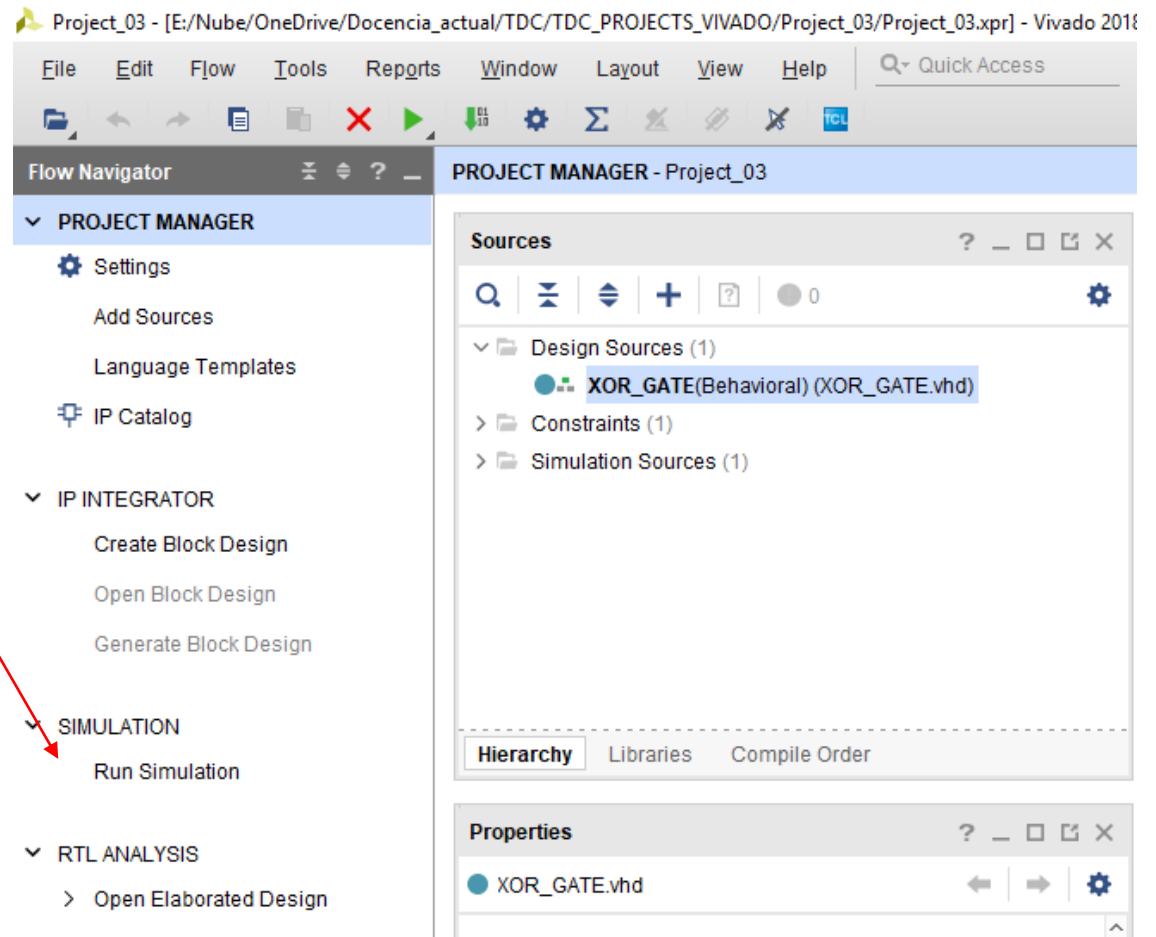
# TDC\_Lab\_2. Simulation



# TDC\_Lab\_2. Simulation

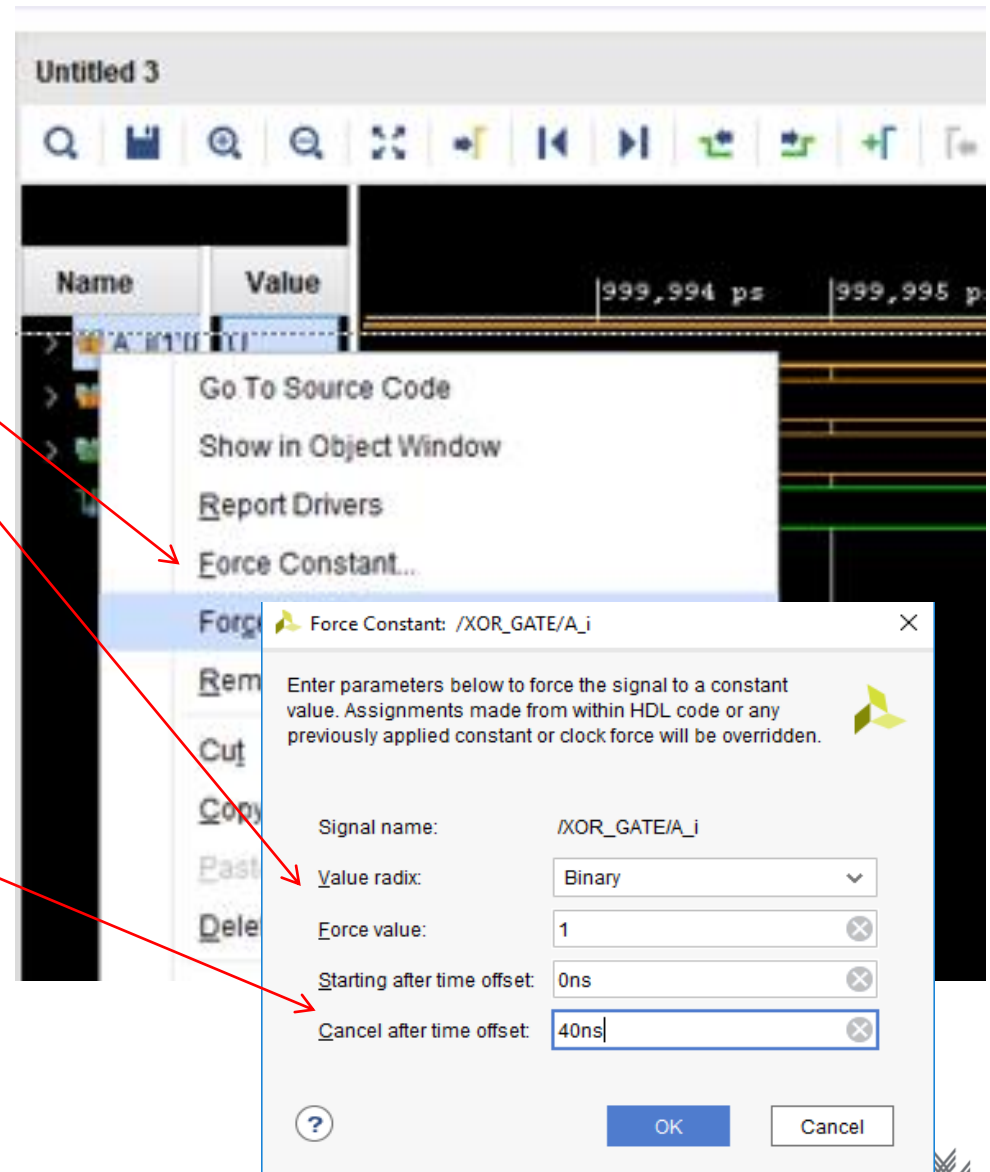
Example: Generating stimuli by means a GUI

1. Open the [Project\\_03](#), “XOR\_GATE”.
2. Run Simulation



## TDC\_Lab\_2. Simulation

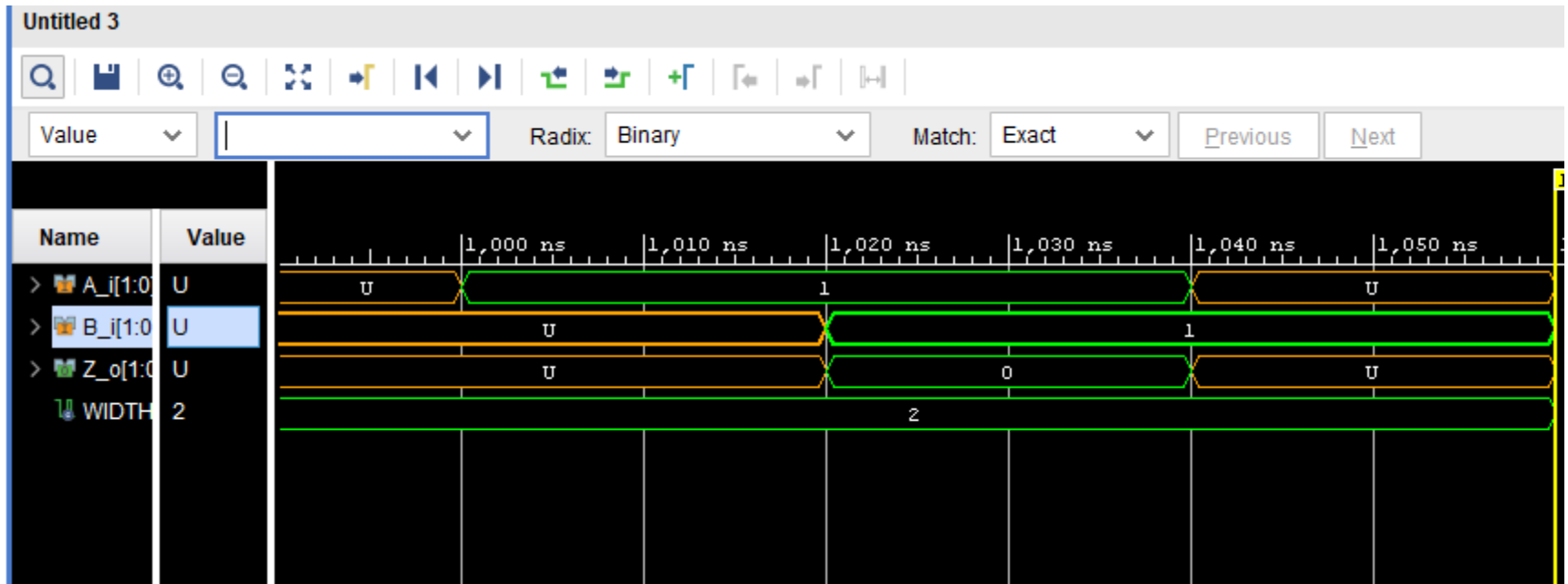
- In the waveform area select one input and choose “Force constant...” by means the right-click
- A new window opens, set “Binary radix” as *binay* and write 0 or 1
- The lasting of the value can be written in “Starting...” and “Cancel...” fields, write the initial and final time in *ns*.



## TDC\_Lab\_2. Simulation

After **A\_i** and **B\_i** input are assigned, run the simulation. 

Notice that the new values are applied after the previous simulation, so after 1000ns.



Use *Relaunch* to clear the previous waveform.

## TDC\_Lab\_2. Simulation

A periodical value can be assigned to the inputs selecting “*Force clock...*” rather than “*Force constant...*”

Initial value will be '0' rising to '1'

Waveform period

Force Clock: /XOR\_GATE/A\_i

Enter parameters below to force the signal to a constant value. Assignments made from within HDL code or any previously applied constant or clock force will be overridden.

Signal name: /XOR\_GATE/A\_i

Value radix: Binary

Leading edge value: 0

Trailing edge value: 1

Starting after time offset:

Cancel after time offset:

Duty cycle (%): 50

Period: 40ns

? OK Cancel



## TDC\_Lab\_2. Simulation

Repeat for the “B\_i” input

Initial value will be '0' rising to '1'

Waveform period

Force Clock: /XOR\_GATE/B\_i

Enter parameters below to force the signal to a constant value. Assignments made from within HDL code or any previously applied constant or clock force will be overridden.

Signal name: /XOR\_GATE/B\_i

Value radix: Binary

Leading edge value: 1

Trailing edge value: 0

Starting after time offset: 0ns

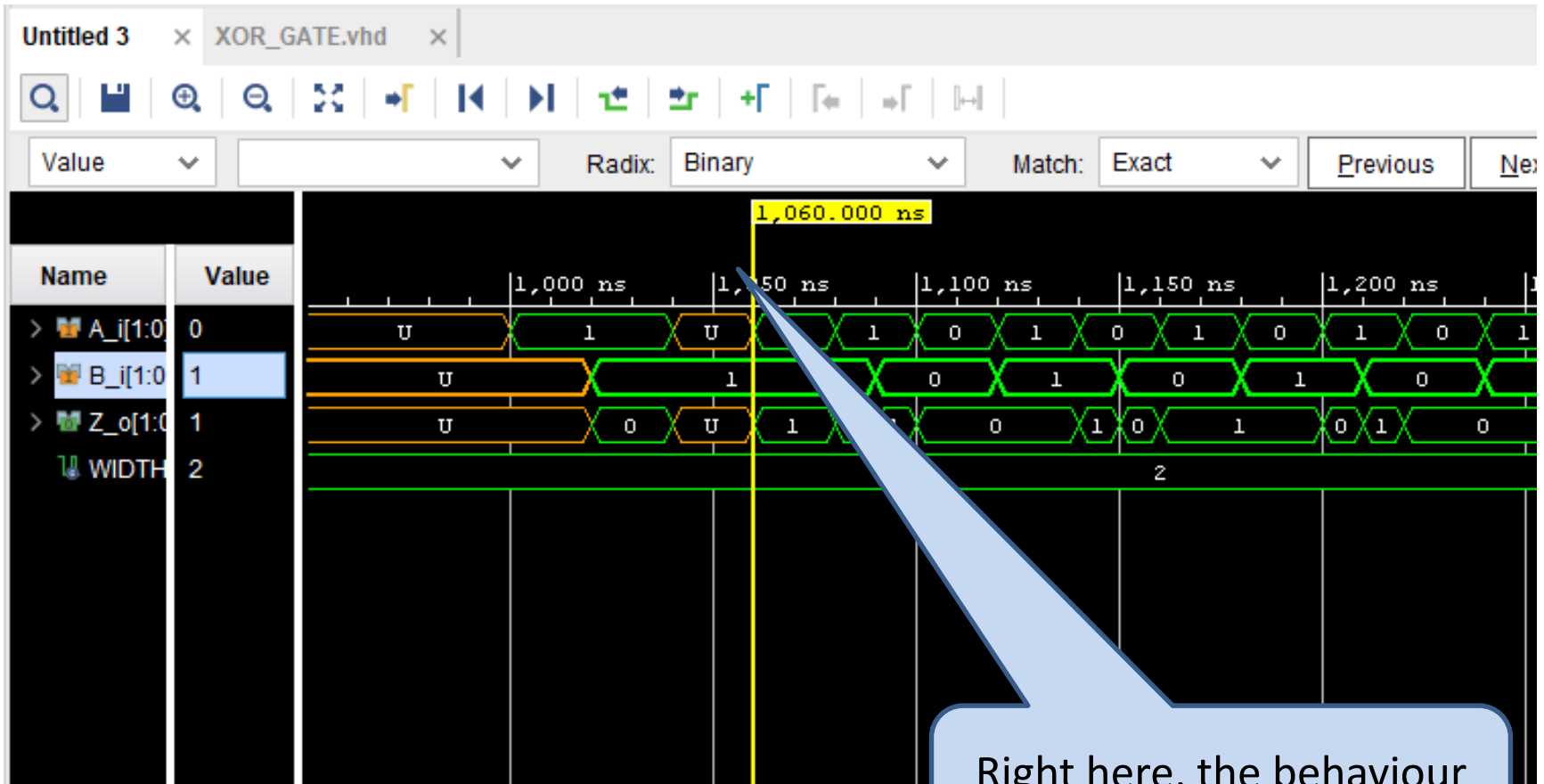
Cancel after time offset:

Duty cycle (%): 50

Period: 60ns

OK Cancel

# TDC\_Lab\_2. Simulation



Right here, the behaviour of the inputs is periodic

## TDC\_Lab\_2. Simulation

1. Add testbenches to all the projects created. Check the right behavior of the circuits designed.

# TDC\_Lab\_2. Simulation

## Bibliography

- 1.- VHDL Testbench Tutorial. École Polytechnique Fédérale de Lausanne  
[https://moodle.epfl.ch/pluginfile.php/1833321/mod\\_resource/content/1/vhdl\\_testbench\\_tutorial.pdf](https://moodle.epfl.ch/pluginfile.php/1833321/mod_resource/content/1/vhdl_testbench_tutorial.pdf)
- 2.- Writing Efficient Testbenches Author: Mujtaba Hamid. (XAPP199 (v1.1) May 17, 2010).  
[https://www.xilinx.com/support/documentation/application\\_notes/xapp199.pdf](https://www.xilinx.com/support/documentation/application_notes/xapp199.pdf)
- 3.- Xilinx® ISE Simulator (ISim) VHDL Test Bench Tutorial. Digilent.  
[https://learn.digilentinc.com/Classroom/Tutorials/Xilinx%20ISE%20Simulator%20\(ISim\)%20VHDL%20Test%20Bench%20Tutorial.pdf](https://learn.digilentinc.com/Classroom/Tutorials/Xilinx%20ISE%20Simulator%20(ISim)%20VHDL%20Test%20Bench%20Tutorial.pdf)
- 4.- Vivado Design Suite Tutorial Logic Simulation UG937 (v2018.1) April 4, 2018.  
[https://www.xilinx.com/support/documentation/sw\\_manuals/xilinx2018\\_1/ug937-vivado-design-suite-simulation-tutorial.pdf](https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_1/ug937-vivado-design-suite-simulation-tutorial.pdf)