Introducción a VHDL

Federico G. Zacchigna

FIUBA

13/01/2017

Outline

Lenguajes HDL

Es un lenguajes utilizaado para una descripción formal del hardware.

- Esquemático.
- Arquitectura.
- Micro-arquitectura.
- Comportamiento.

Ejemplos

VHDL, Verilog, SystemC, SystemVerilog, AHDL.

Bibliotecas y tipos de datos

Bibliotecas típicas

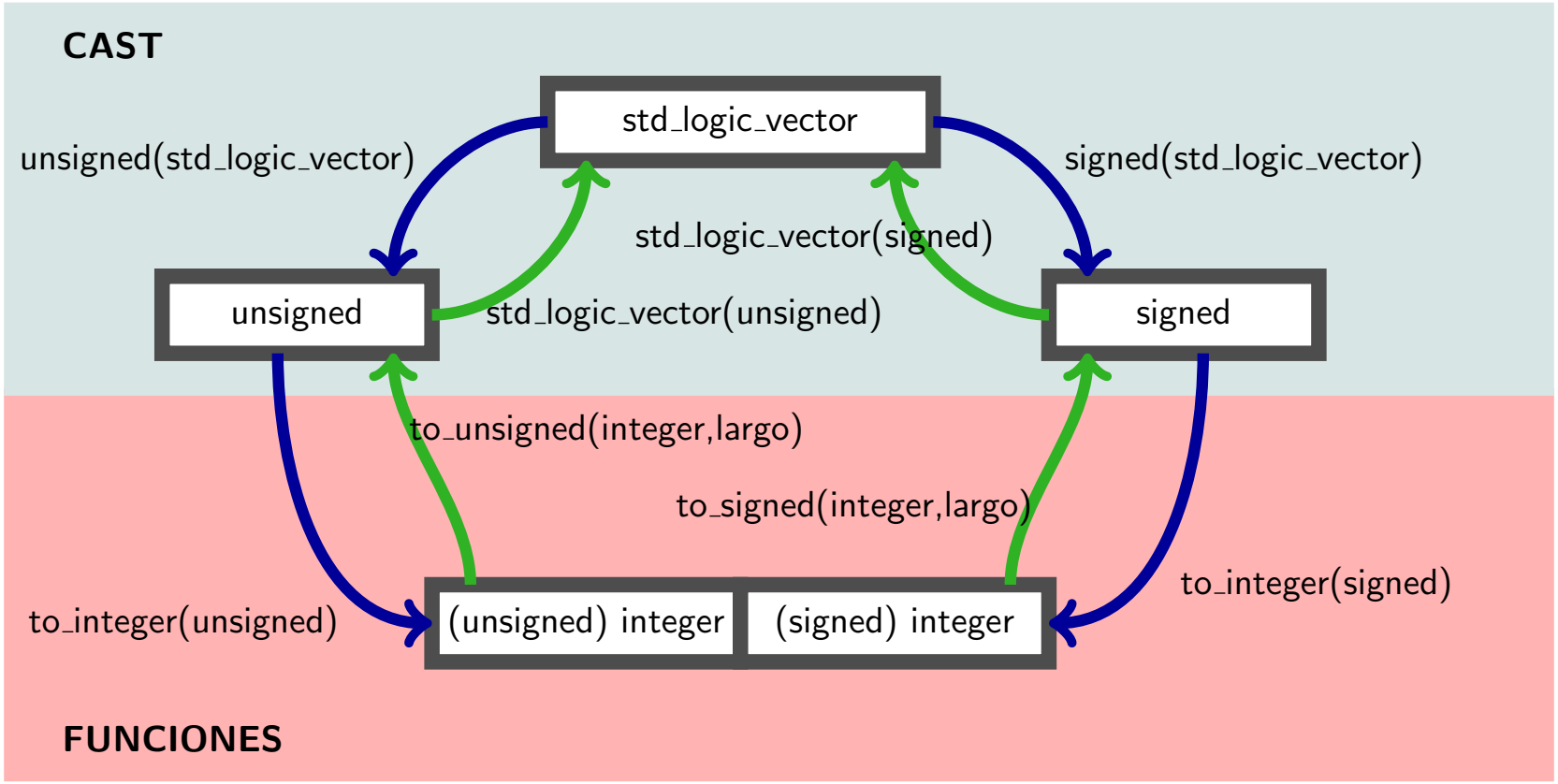
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric.all;
```

Tipos de datos

```
std_logic
std_logic_vector(1 downto 0)
singed
unsigned
integer
```

Conversiones entre datos

Para aquellas señales que tengan más de un bit:



Es necesario incluir las bibliotecas:

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

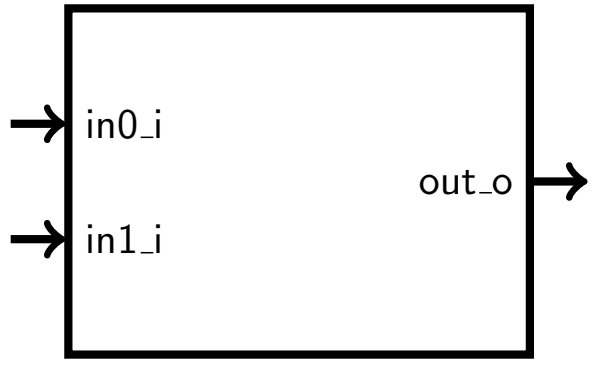
Estructura de un archivos

Se utilizará un archivo por cada bloque de diseño.

```
1  ----------------------------------------------------------
2  -- bibliotecas
3  library ieee;
4  use ieee.std_logic_1164.all;
5  use ieee.numeric_std.all;
6  ----------------------------------------------------------
7
8  ----------------------------------------------------------
9  -- se definen el nombre los puertos de la ENTIDAD
10 entity andgate is
11   -- puertos de la arquitectura
12
13 end entity andgate;
14 ----------------------------------------------------------
15
16 ----------------------------------------------------------
17 -- se define la ARQUITECTURA del bloque
18 architecture rtl of andgate is
19   -- declaracion de variables internas
20 begin
21   -- comportamiento/estructura del bloque
22
23 end architecture rtl;
24 ----------------------------------------------------------
```

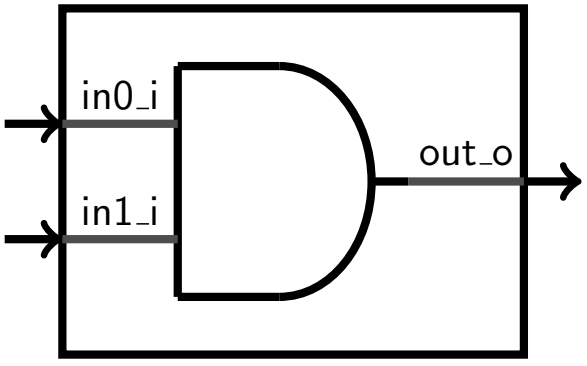
Puertos de entrada y salida

```
1  -- entidad
2  entity andgate is
3    port (
4      in0_i : in  std_logic;
5      in1_i : in  std_logic;
6      out_o : out std_logic
7    );
8  end entity andgate;
```



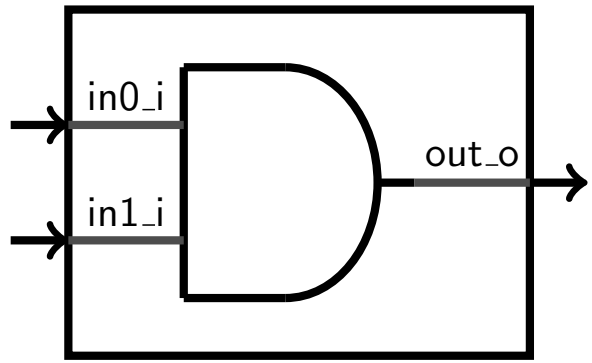
Descripción del comportamiento

```
1  -- arquitetura
2  architecture rtl of andgate is
3    -- declaracion de senales,
4    -- variables y constantes internas
5  begin
6    -- descripcion de la arquitectura
7    out_o <= in0_i and in1_i;
8  end architecture rtl;
```



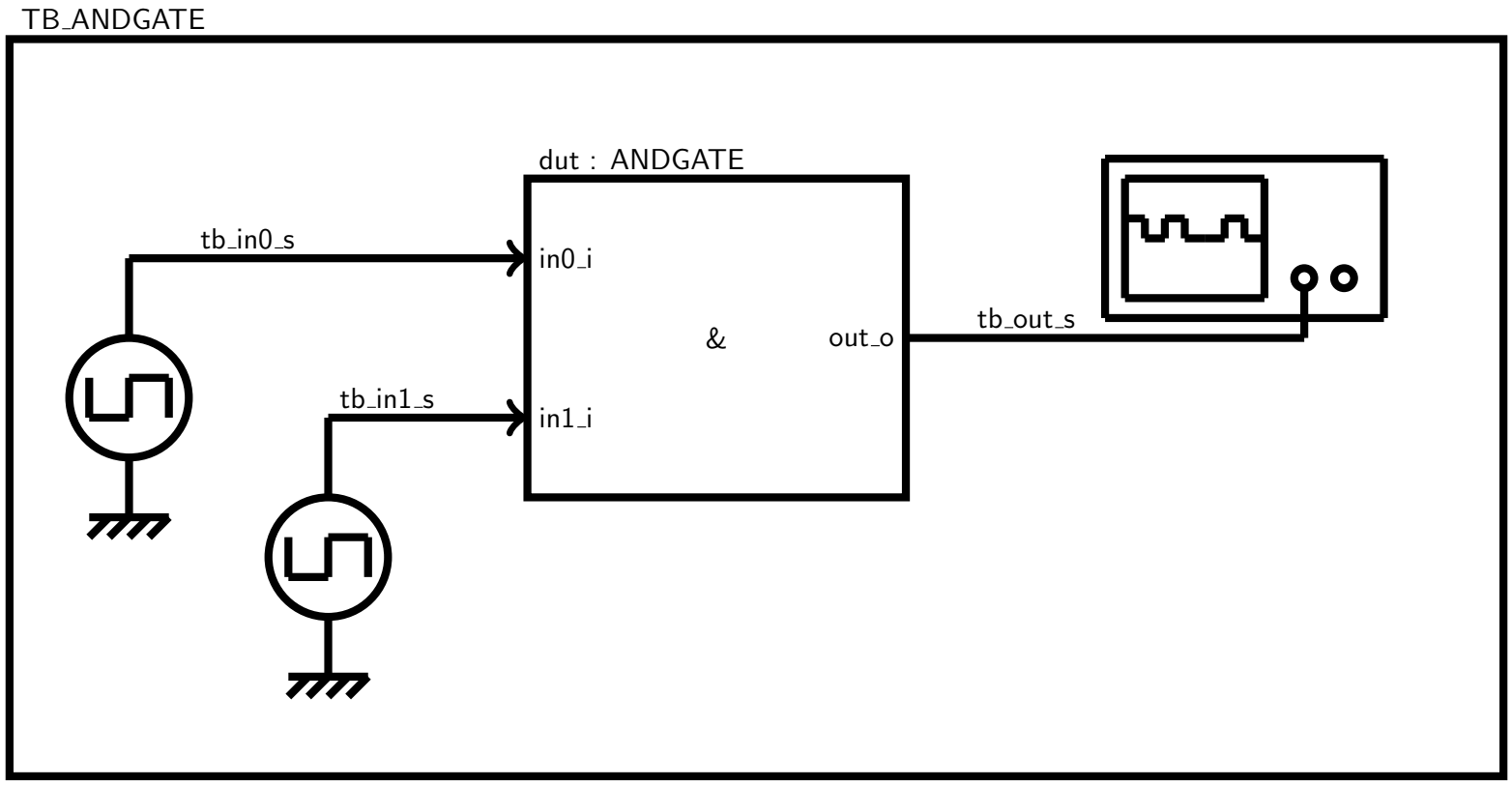
Bloque AND completo

```vhdl
-- bibliotecas
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

-- entidad
entity andgate is
  port (
    in0_i : in  std_logic;
    in1_i : in  std_logic;
    out_o : out std_logic
  );
end entity andgate;

-- arquitetura
architecture rtl of andgate is
  -- declaracion de senales,
  -- variables y constantes internas
begin
  -- descripcion de la arquitectura
  out_o <= in0_i and in1_i;
end architecture rtl;
```



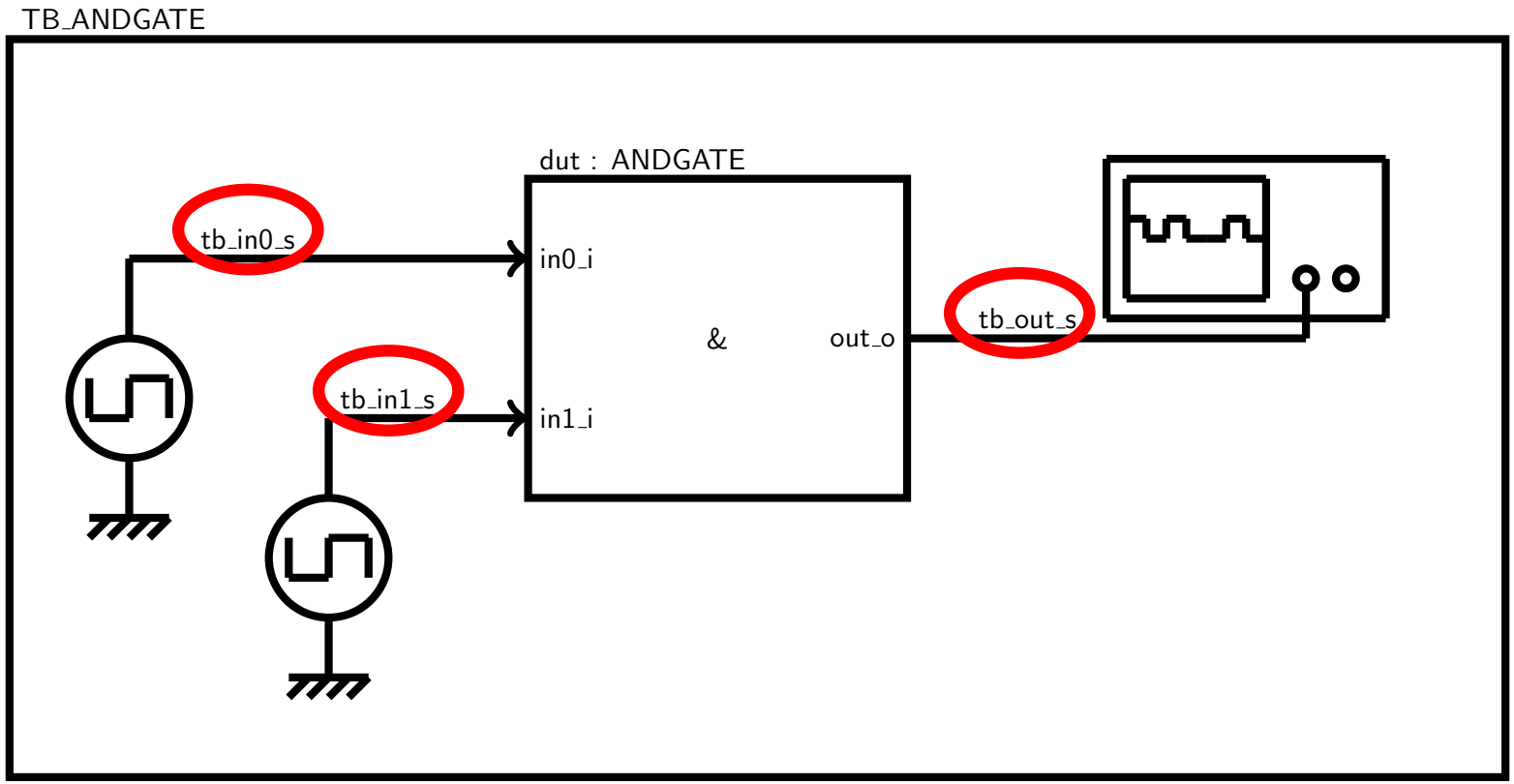
Testbench para compuerta AND

- Para testear otros bloques.
- Se generan señaes de pruebas que se conectan a las entradas del DUT.
- Se observan las salidas del DUT.



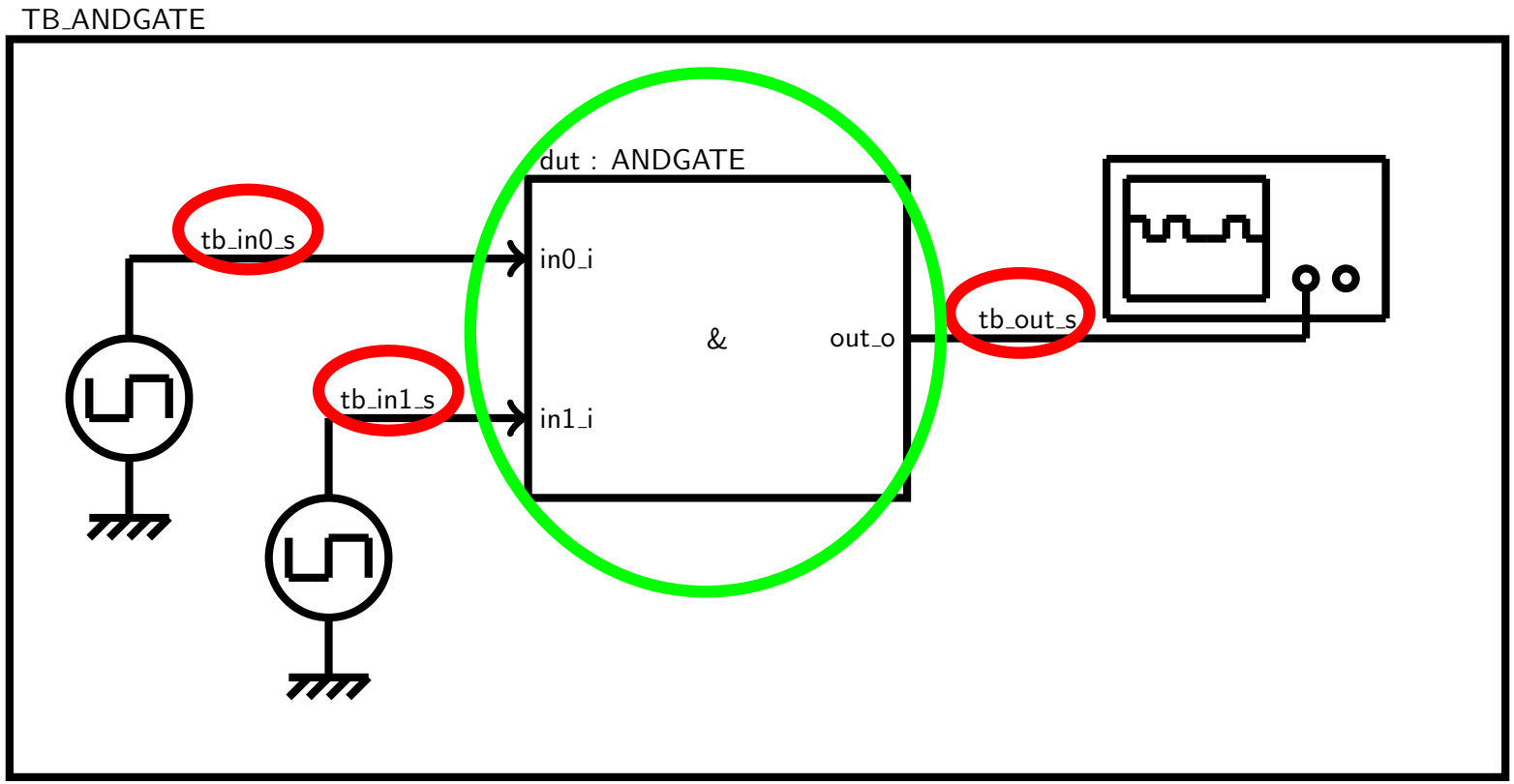
Testbench para compuerta AND

- Para testear otros bloques.
- Se generan señaes de pruebas que se conectan a las entradas del DUT.
- Se observan las salidas del DUT.



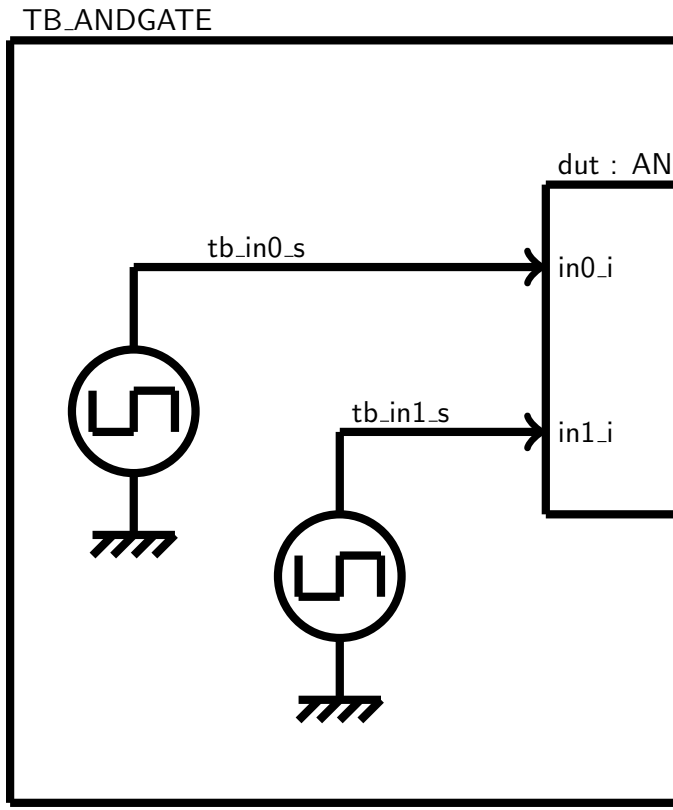
Testbench para compuerta AND

- Para testear otros bloques.
- Se generan señaes de pruebas que se conectan a las entradas del DUT.
- Se observan las salidas del DUT.



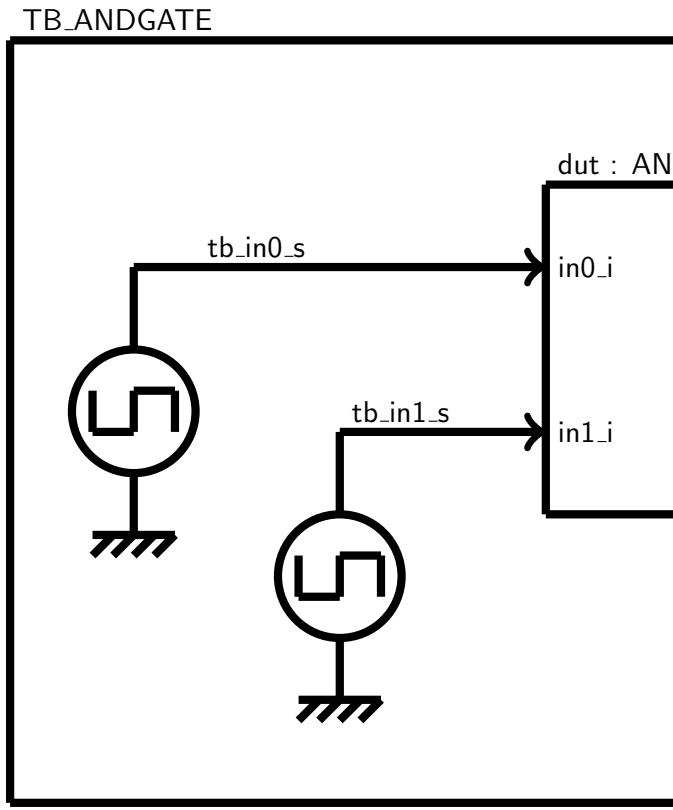
Testbench para compuerta AND

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

-- entidad
entity tb_andgate is
end entity tb_andgate;

-- arquitetura
architecture rtl of tb_andgate is
  component andgate is
    port (
      in0_i : in  std_logic;
      in1_i : in  std_logic;
      out_o : out std_logic
    );
  end component andgate;
  signal tb_in0_s : std_logic;
  signal tb_in1_s : std_logic;
  signal tb_out_s : std_logic;
begin
```



Testbench para compuerta AND

```
21  begin
22
23    dut : andgate
24    port map(
25      in0_i => tb_in0_s,
26      in1_i => tb_in1_s,
27      out_o => tb_out_s
28    );
29
30    -- asignacion de senales de entrada
31    -- senal 0
32    tb_in0_s <= '0',
33                '1' after 10 ns,
34                '0' after 23 ns;
35
36    -- senal 1
37    tb_in1_s <= '1',
38                '0' after 17 ns,
39                '1' after 27 ns;
40
41  end architecture rtl;
```

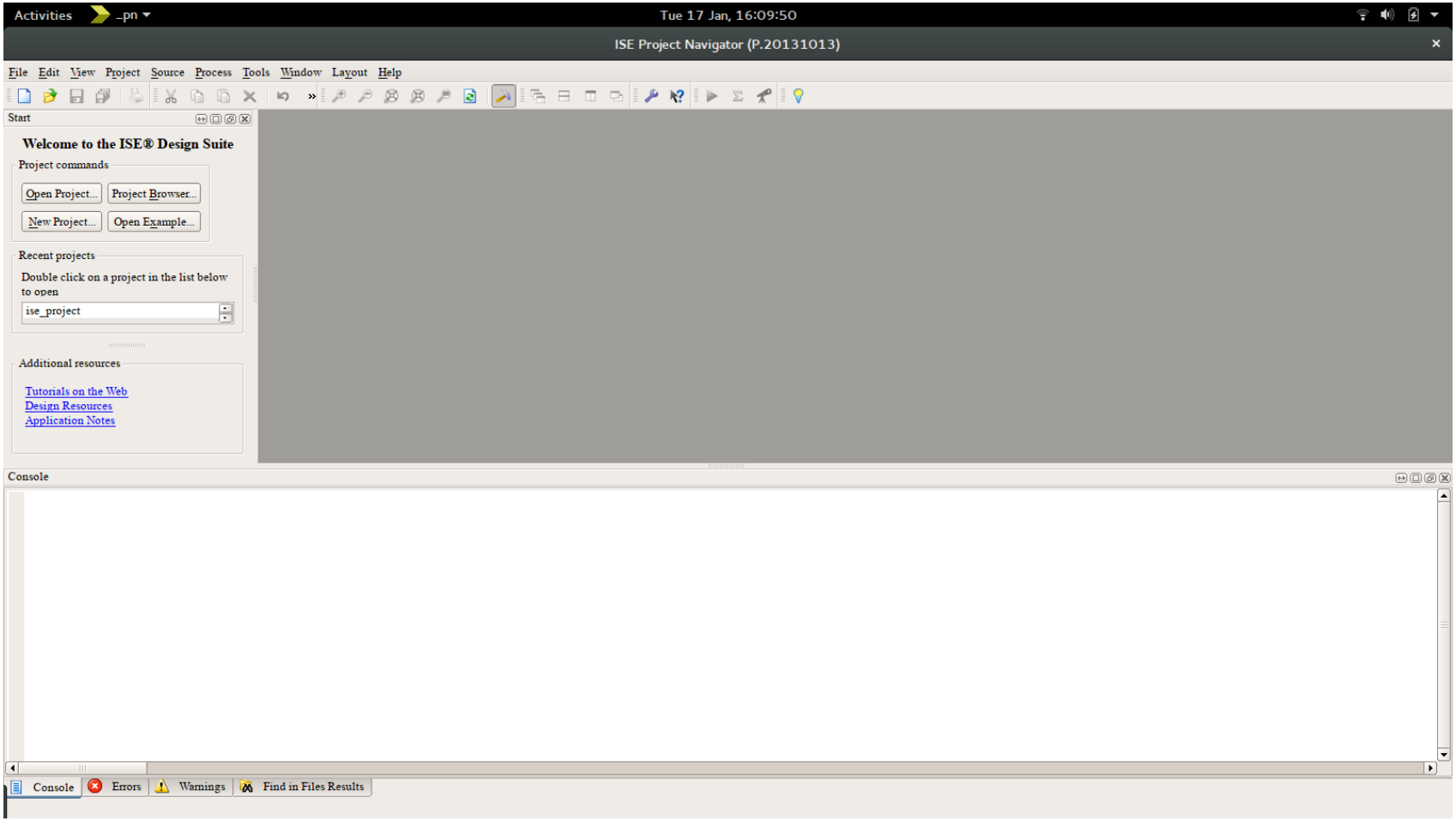


Testbench para compuerta AND

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

-- entidad
entity tb_andgate is
end entity tb_andgate;

-- arquitetura
architecture rtl of tb_andgate is
  component andgate is
    port (
      in0_i : in  std_logic;
      in1_i : in  std_logic;
      out_o : out std_logic
    );
  end component andgate;
  signal tb_in0_s : std_logic;
  signal tb_in1_s : std_logic;
  signal tb_out_s : std_logic;
begin
  -- device under test
  dut : andgate
  port map(
    in0_i => tb_in0_s,
    in1_i => tb_in1_s,
    out_o => tb_out_s
  );

  -- asignacion de senales de entrada
  -- senal 0
  tb_in0_s <= '0',
              '1' after 10 ns,
              '0' after 23 ns;

  -- senal 1
  tb_in1_s <= '1',
              '0' after 17 ns,
              '1' after 27 ns;

end architecture rtl;
```

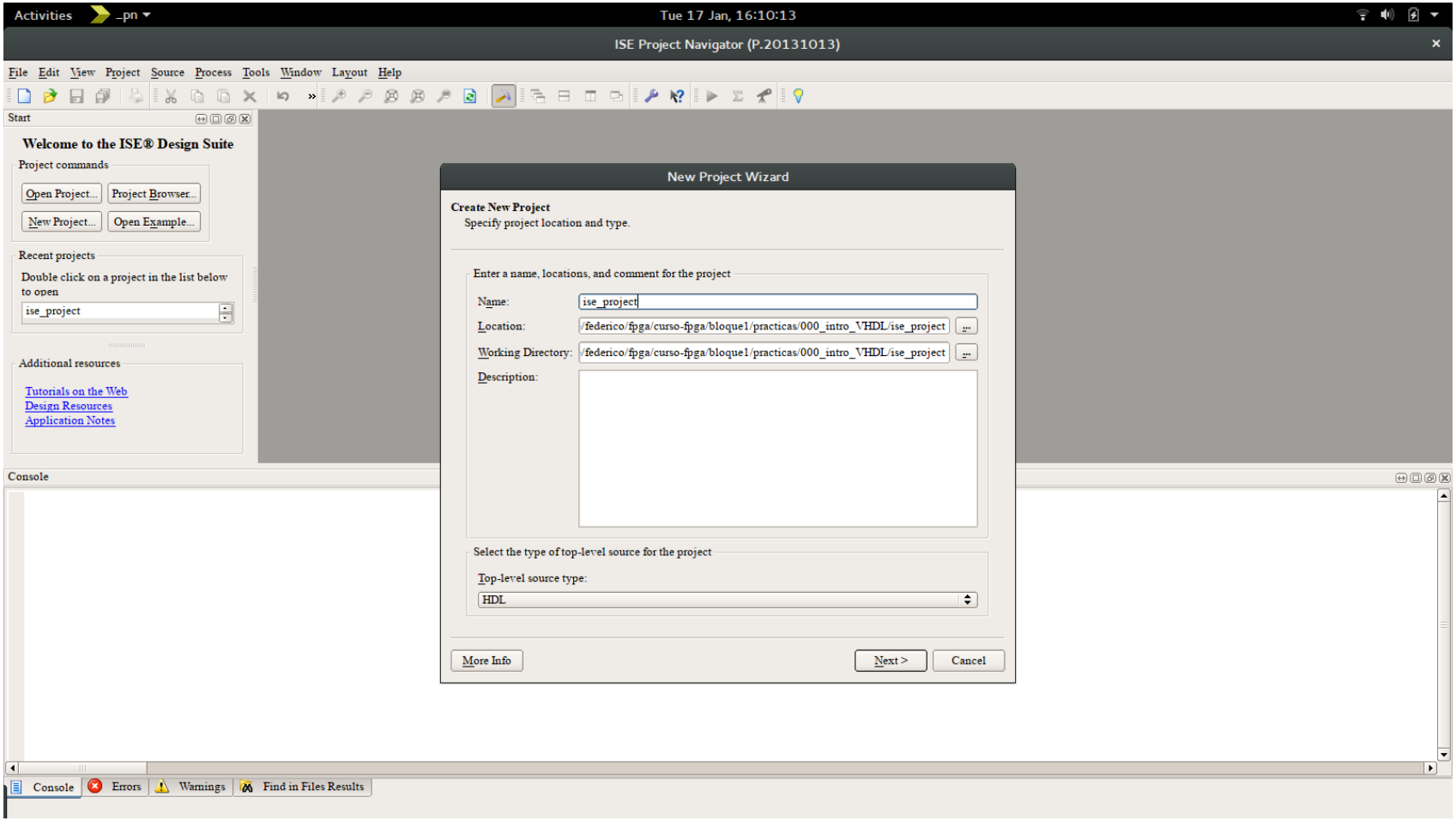
Simulación del testbench

Abrimos el ise.



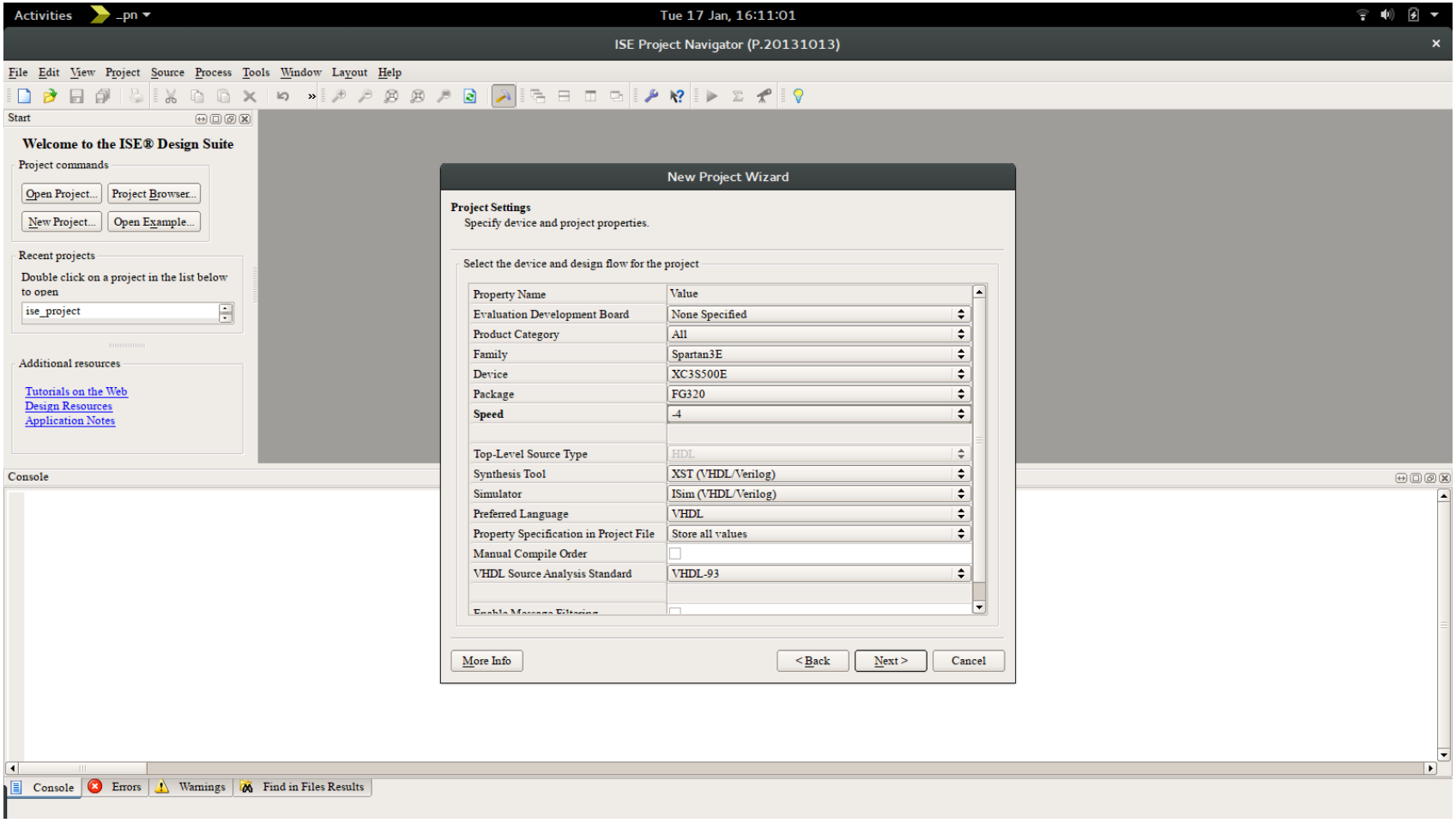
Simulación del testbench

Luego “File → New Project ...”
Elegimos directorio de trabajo
Elegimos nombre del proyecto



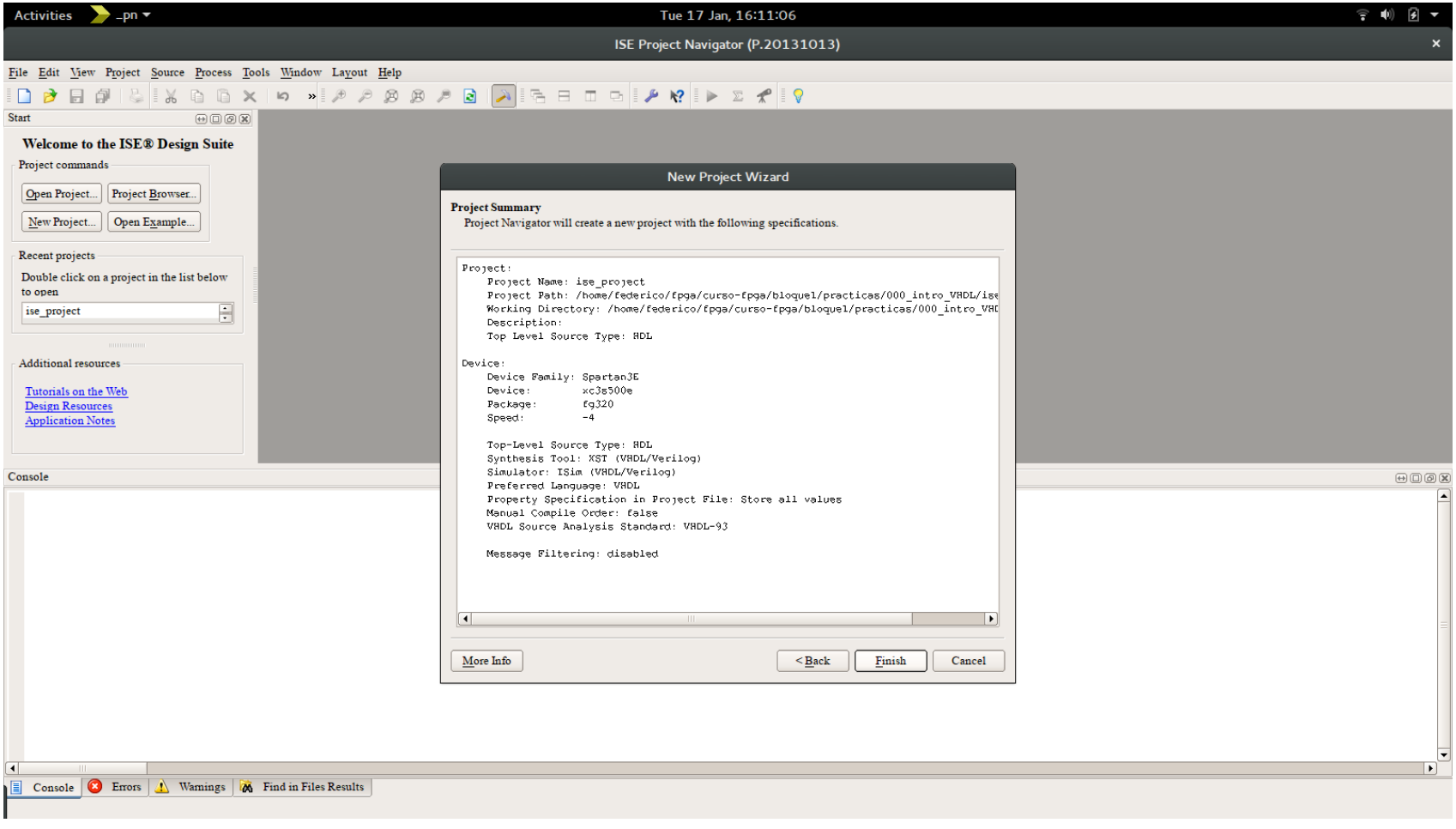
Simulación del testbench

Elegimos el kit (FPGA), lenguaje y herramientas



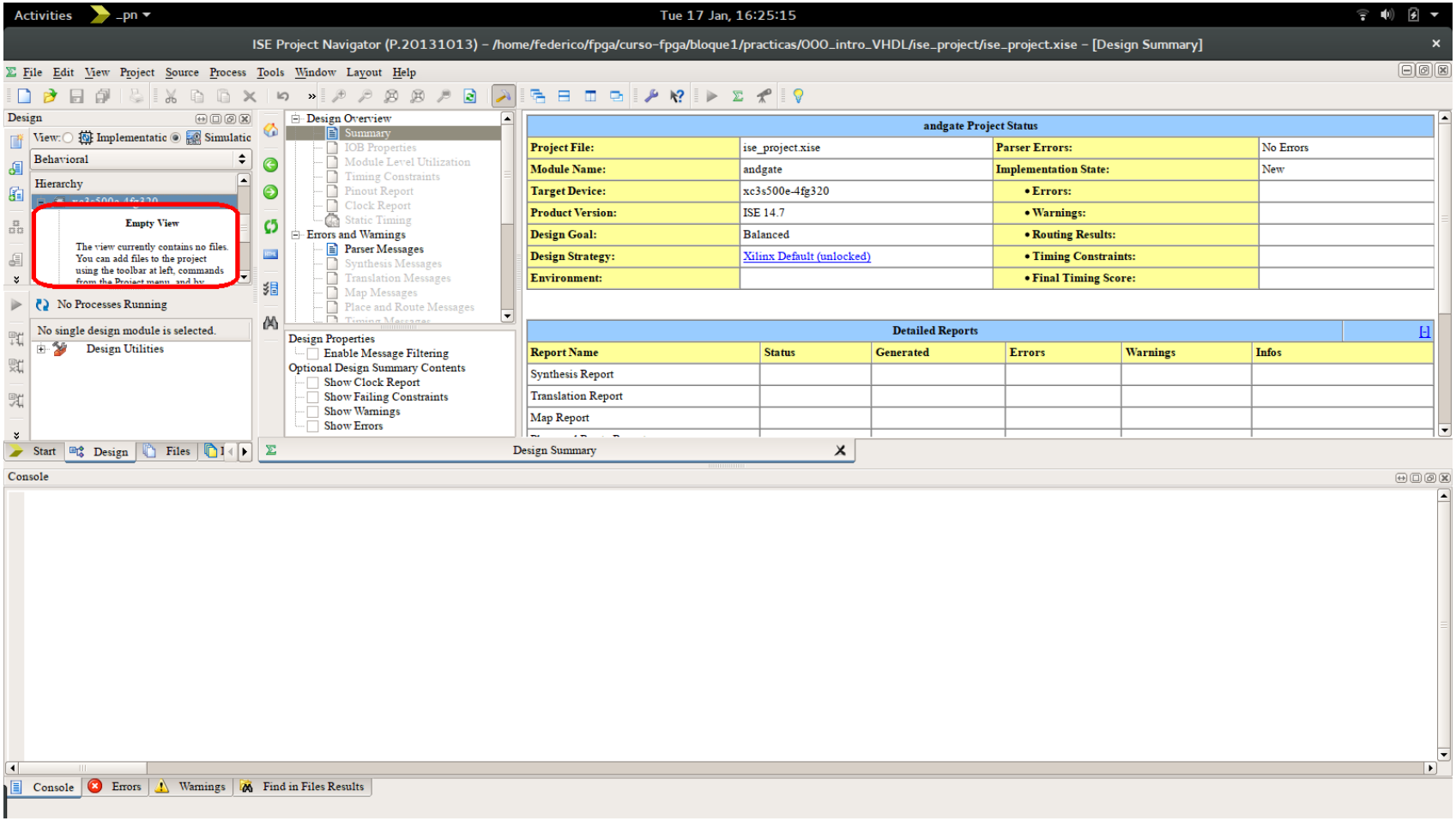
Simulación del testbench

"Finish"



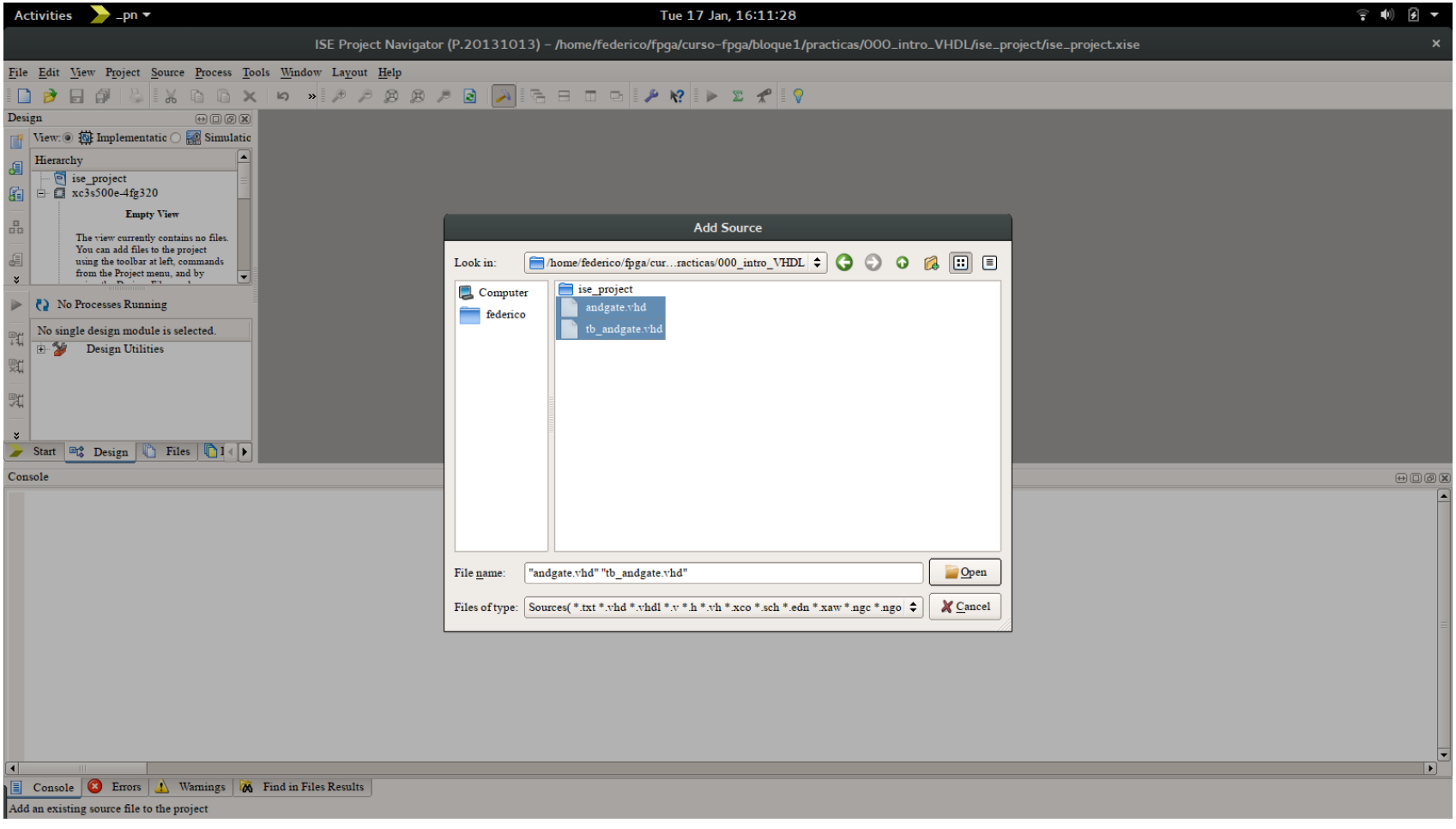
Simulación del testbench

Clic derecho sobre la zona marcada y “Add Source ...”.



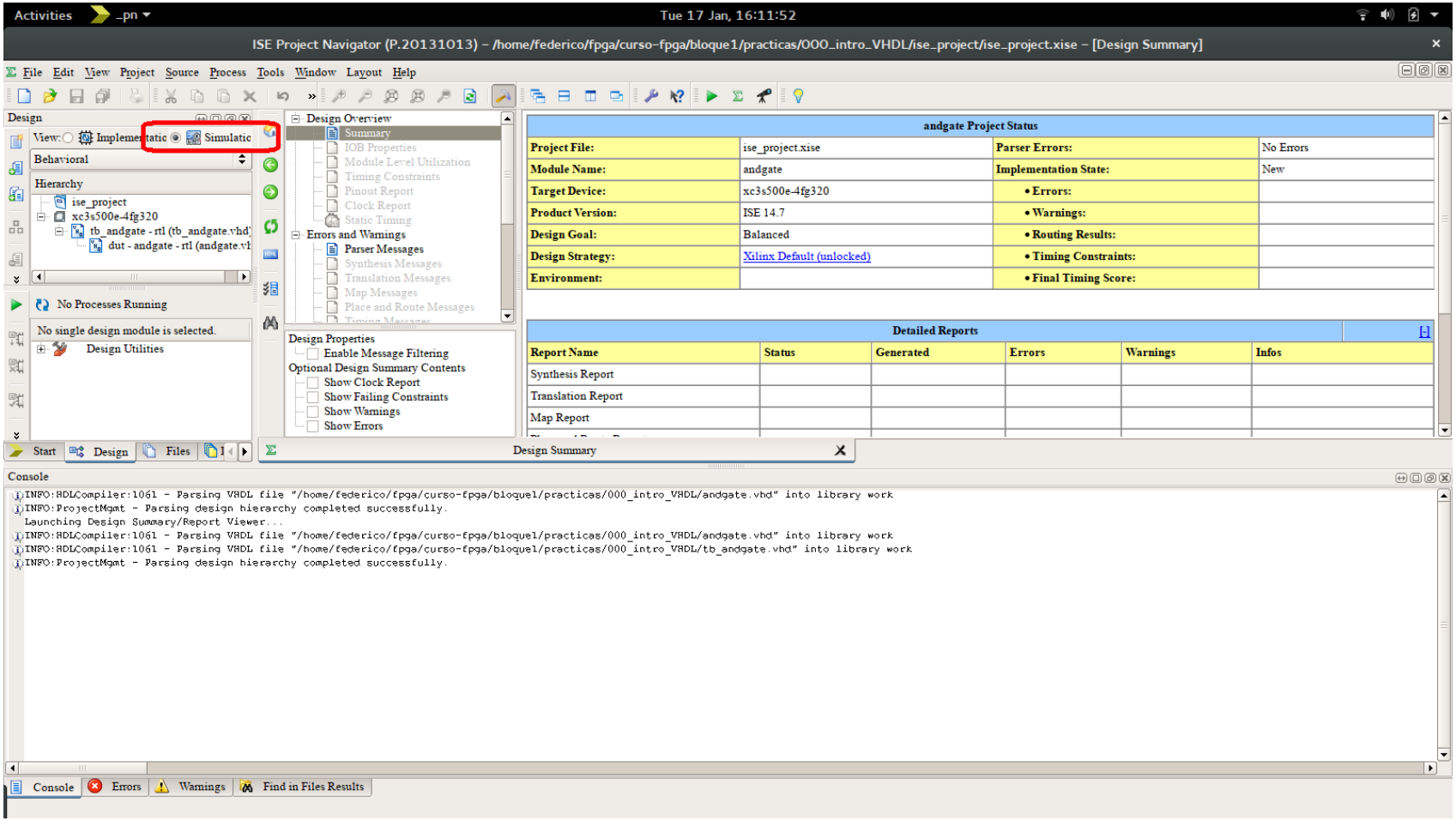
Simulación del testbench

Elegimos los archivos fuentes de nuestros archivos.
NOTA: El “tb_andgate.vhd”, agregarlo solo para la simulación



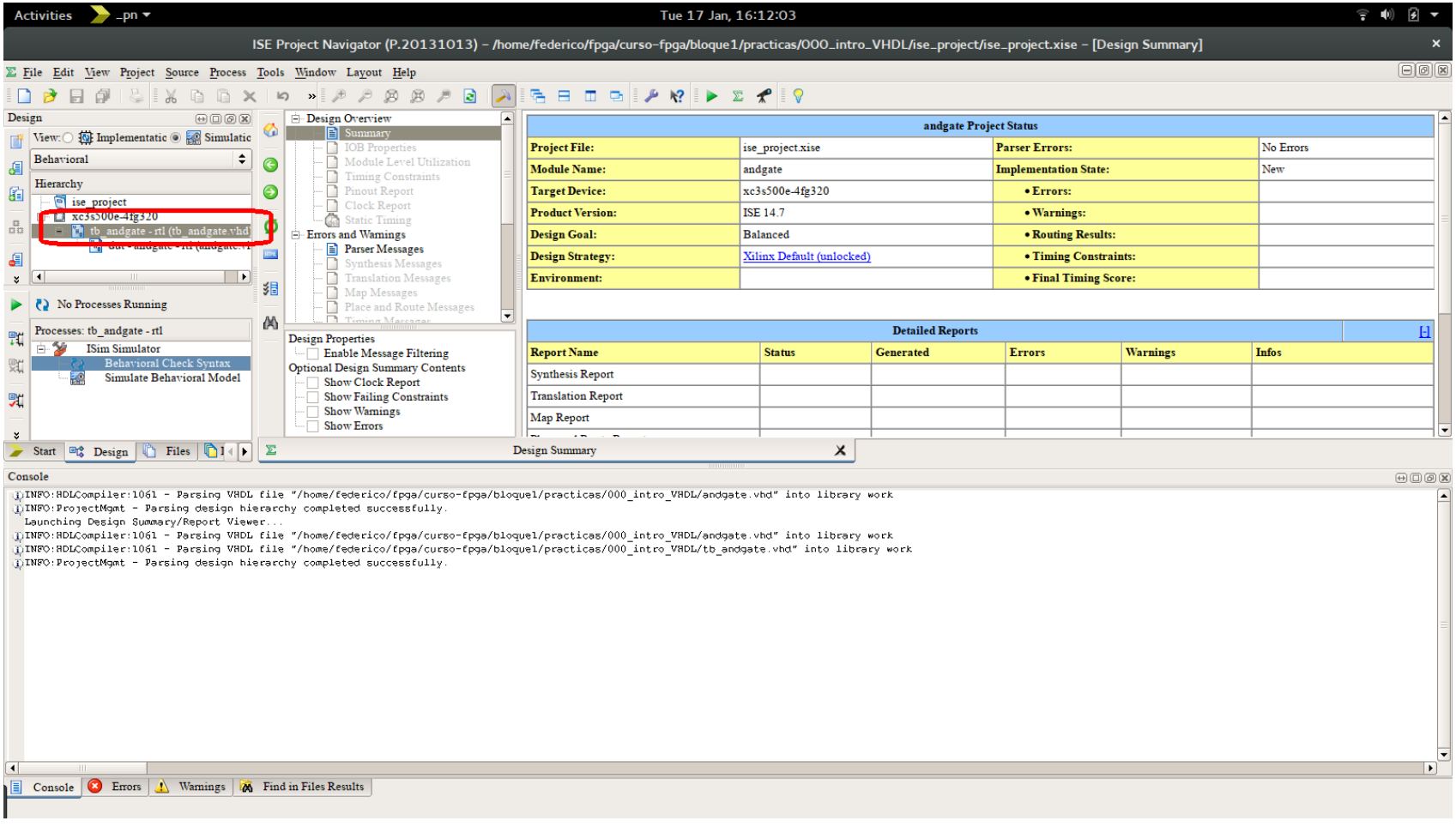
Simulación del testbench

Clic sobre “Simulación”



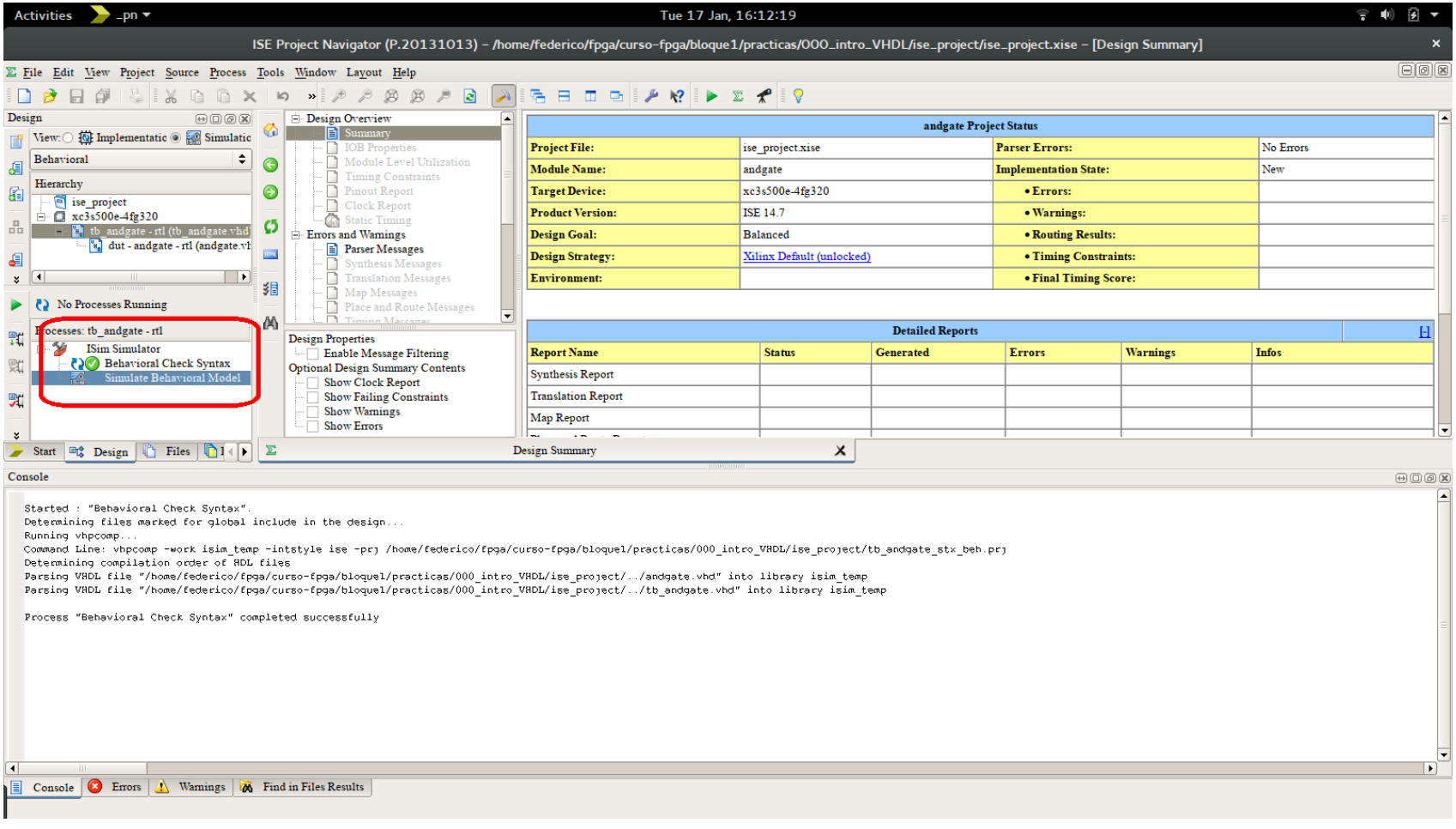
Simulación del testbench

Clic sobre el archivo a simular: “tb_andgate.vhd”



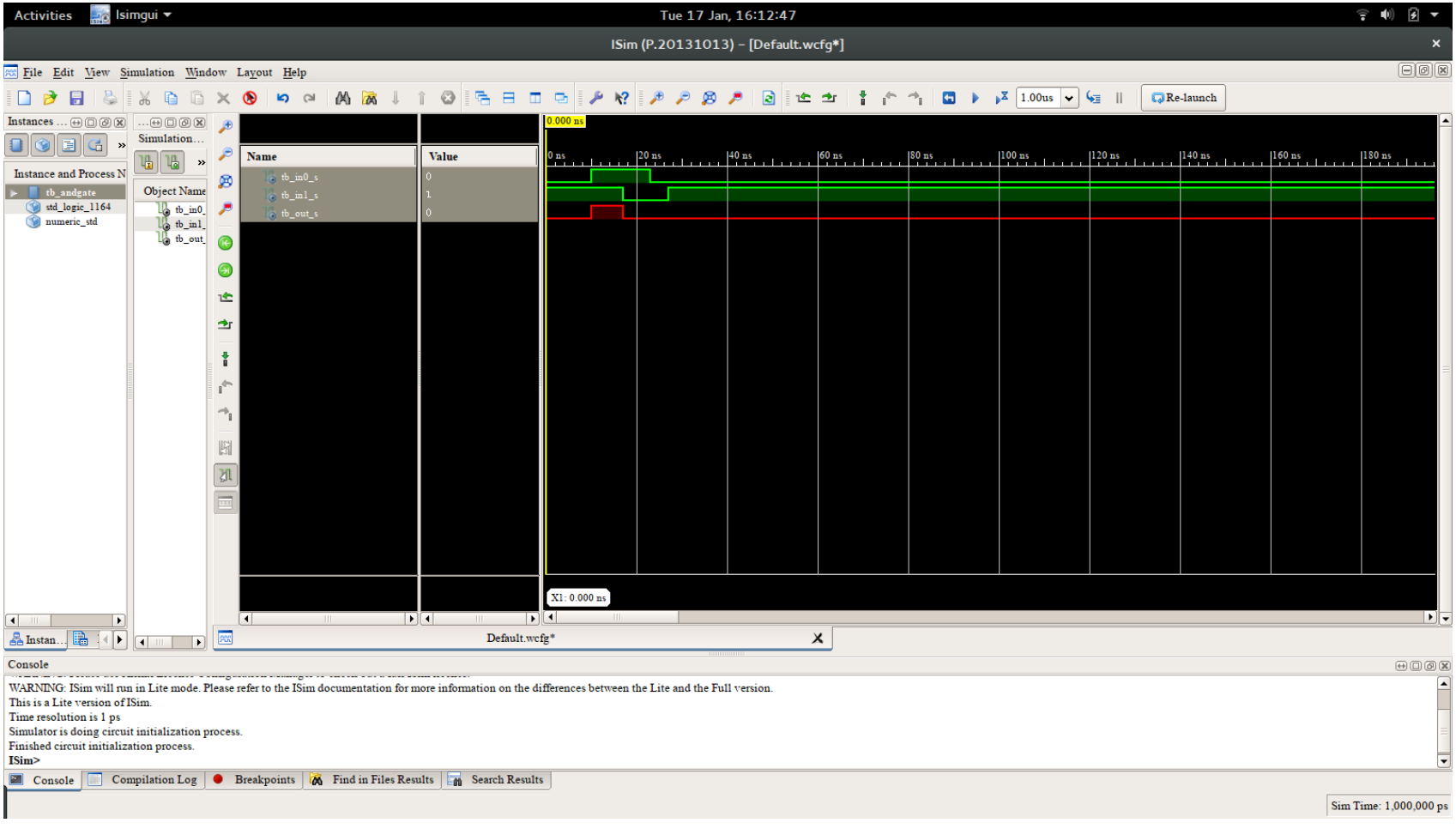
Simulación del testbench

Doble clic sobre “Beahvioral Check Syntax”
Doble clic sobre “Simulate Behavioral Model”



Simulación del testbench

Se abre el ISim.
Comprobamos que la salida de nuestra compuerta sea correcta
Solo está en ‘1’, cuando las dos entradas están en ‘1’.



Compuerta AND completa de 1 y 4 entradas

```vhdl
-- bibliotecas
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

-- entidad
entity andgate is

  port (
    in0_i : in  std_logic;

    in1_i : in  std_logic;

    out_o : out std_logic

  );
end entity andgate;

-- arquitetura
architecture rtl of andgate is
begin
  out_o <= in0_i and in1_i;
end architecture rtl;
```

```vhdl
-- bibliotecas
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

-- entidad
entity andgate4 is

  port (
    in0_i : in  std_logic_vector(
      3 downto 0);
    in1_i : in  std_logic_vector(
      3 downto 0);
    out_o : out std_logic_vector(
      3 downto 0)
  );
end entity andgate4;

-- arquitetura
architecture rtl of andgate4 is
begin
  out_o <= in0_i and in1_i;
end architecture rtl;
```

Compuerta AND completa de 4 y N entradas

```vhdl
-- bibliotecas
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

-- entidad
entity andgate4 is

  port (
    in0_i : in  std_logic_vector(
      3 downto 0);
    in1_i : in  std_logic_vector(
      3 downto 0);
    out_o : out std_logic_vector(
      3 downto 0)
  );
end entity andgate4;

-- arquitetura
architecture rtl of andgate4 is
begin
  out_o <= in0_i and in1_i;
end architecture rtl;
```

```vhdl
-- bibliotecas
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

-- entidad
entity andgateN is
  generic (N : integer);
  port (
    in0_i : in  std_logic_vector(
      N-1 downto 0);
    in1_i : in  std_logic_vector(
      N-1 downto 0);
    out_o : out std_logic_vector(
      N-1 downto 0)
  );
end entity andgateN;

-- arquitetura
architecture rtl of andgateN is
begin
  out_o <= in0_i and in1_i;
end architecture rtl;
```