



Práctica 1

- Utilización de la herramienta Xilinx ISE con VHDL
- Diseño y simulación de circuitos sencillos



Pantalla de bienvenida

The screenshot shows the ISE Project Navigator (P.15xf) interface. The main window displays the "Welcome to the ISE® Design Suite" screen. On the left, there's a "Project commands" panel with buttons for "Open Project...", "Project Browser...", "New Project...", and "Open Example...". Below it is a "Recent projects" section with a placeholder message: "Double click on a project in the list below to open it". The right side of the main window is a large gray area. A blue callout bubble points from the text "Aquí aparecerán los proyectos ya existentes" to the "Recent projects" list.

Aquí aparecerán los proyectos ya existentes

Did you know...

Project Management:

You can search for missing modules in the design hierarchy by opening the **Find** toolbar in the **Design Hierarchy view** (*right-click > Find... in the Hierarchy window*), then selecting **Missing Module** as the Type, and entering * in the Find field.

OK
Next Tip
Previous Tip
Help

Show Tips at Startup

Console

Console Errors Warnings Find in Files Results



Crear un proyecto (I)

■ Desde la ventana de bienvenida

The screenshot shows the ISE Project Navigator interface. On the left, there's a sidebar with 'Project commands' containing 'Open Project...', 'New Project...', and 'Open Example...'. A blue callout bubble points to 'New Project...' with the text 'Siempre en hlocal.
NO USAR ESPACIOS
EN LA RUTA!!!'. The main area displays the 'Create New Project' dialog from the 'New Project Wizard'. The dialog has fields for 'Name', 'Location' (set to 'C:\Users'), 'Working Directory' (also set to 'C:\Users'), and 'Description'. Below this, it says 'Select the type of top-level source for the project' and 'Top-level source type' with 'HDL' selected. At the bottom are 'More Info', 'Next', and 'Cancel' buttons.

Seleccionar dónde se va a implementar



New Project Wizard

Project Settings

Specify device and project properties.
Select the device and design flow for the project

Property Name	Value
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan3
Device	XC3S1000
Package	FT256
Speed	-5
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog) Default: HDL
Preferred Language	VHDL
Property Specification in Project	None
Manual Compile Order	None
VHDL Source Analysis Standard	None
Enable Message Filtering	None

Seleccionar el entorno de simulación ISim

More Info Next Cancel

Spartan 3
XC3S1000
FT256
-5



Crear un archivo VHDL (I)

- Siempre se inicia creando un fichero tipo VHDL Module

The screenshot shows the Xilinx ISE software interface. On the left, the 'Design' view displays a project structure with a 'practica1' folder containing an 'xc3s1000-5ft256' file. A blue callout bubble labeled 'New source' points to the 'File' icon in the toolbar. On the right, the 'New Source Wizard' dialog is open, titled 'Select Source Type'. It lists various source types: IP (CORE Generator & Architecture Wizard), Schematic, User Document, Verilog Module, Verilog Test Fixture, VHDL Module (which is selected and highlighted in blue), VHDL Library, VHDL Package, VHDL Test Bench, and Embedded Processor. To the right of the list are fields for 'File name:' (empty) and 'Location:' (set to 'C:\Xilinx\practica1'). A blue callout bubble labeled 'Nombre del fichero VHDL' points to the 'File name:' field. At the bottom of the dialog is a 'More Info' button.

New source

Design

Hierarchy

- practica1
- xc3s1000-5ft256

Empty View

The view currently contains
You can add files to the project
using the toolbar at left, com
from the Project menu, and
using the Design, Files, and
Libraries panels.

No Processes Running

No single design module is selected

Design Utilities

New Source Wizard

Select Source Type

Select source type, file name and its location.

- IP (CORE Generator & Architecture Wizard)
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module
- VHDL Library
- VHDL Package
- VHDL Test Bench
- Embedded Processor

File name:

Location:

C:\Xilinx\practica1

More Info

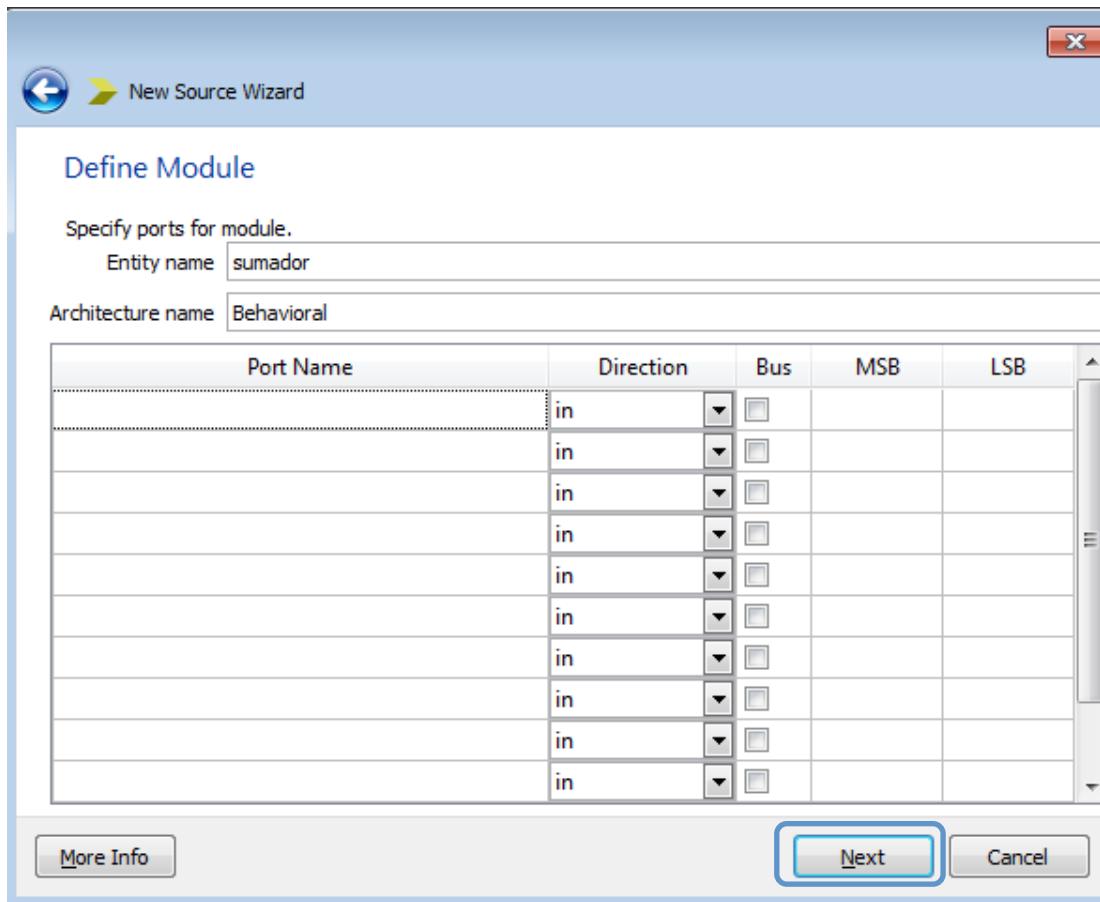
Nombre del fichero VHDL

Si traéis el fichero VHDL desde casa hay que pulsar en Add Source (pasar a transparencia 8)



Crear un archivo VHDL (II)

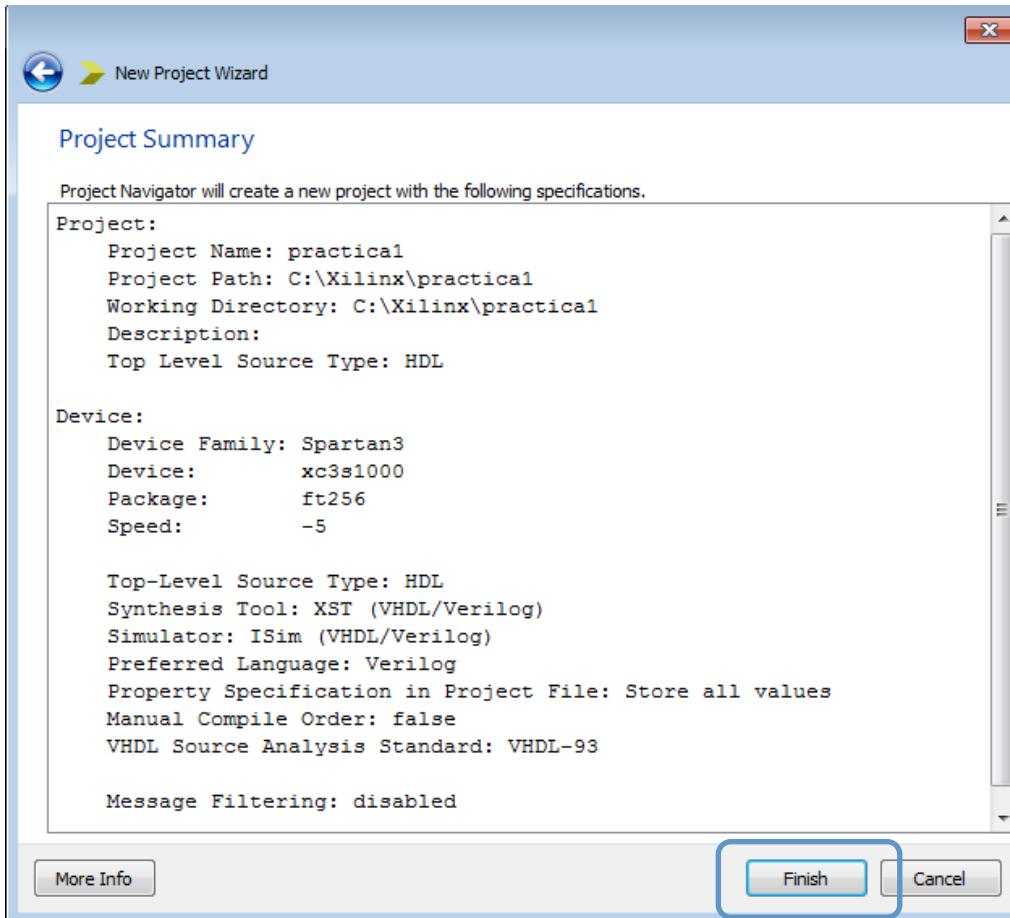
- Wizard para definir las señales de entrada y salida
(NO UTILIZAR. Hacer click en “Next”)





Crear un archivo VHDL (III)

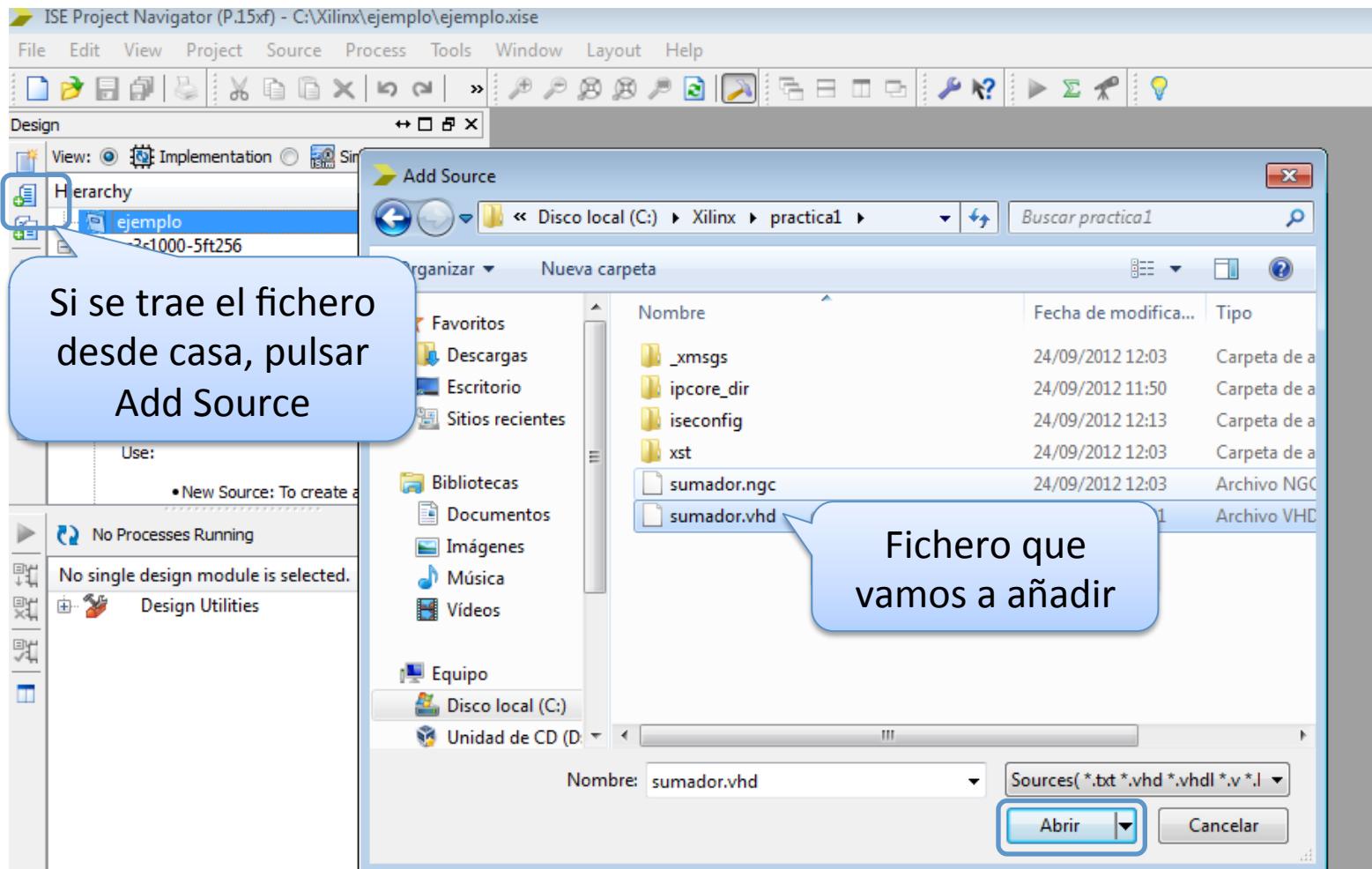
- Todo ha salido correcto





Añadir fuentes a nuestro proyecto

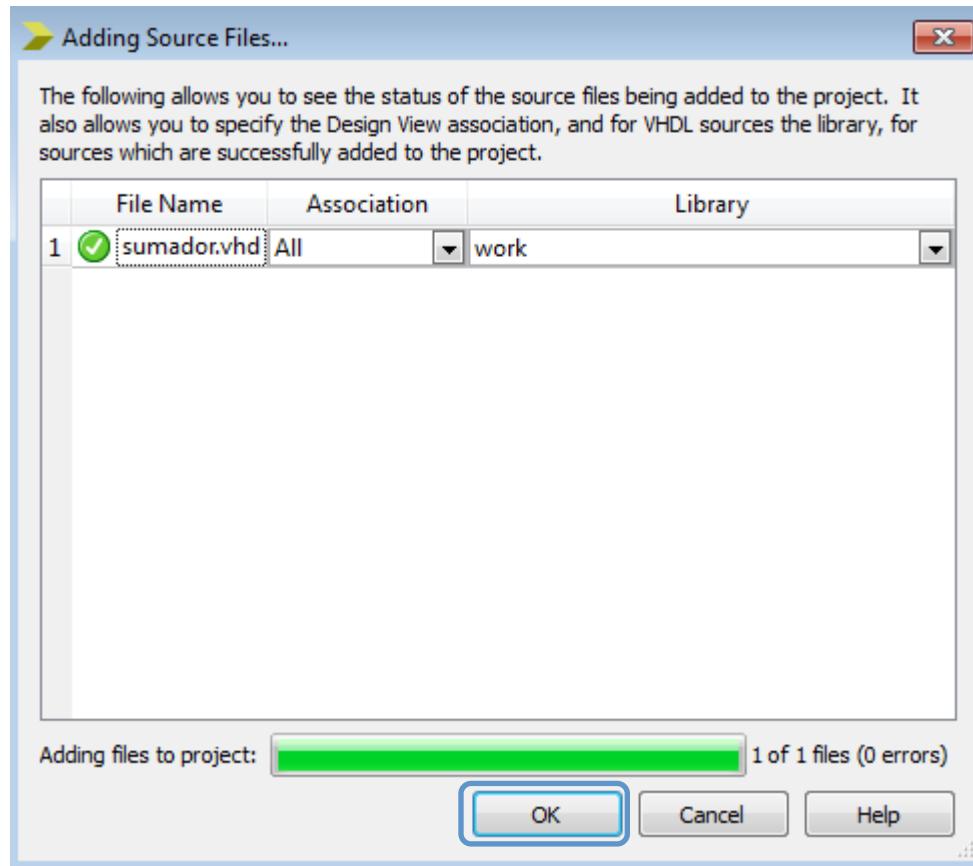
- Se pueden añadir otros ficheros fuente a nuestro proyecto
- Los ficheros a añadir deben ser previamente grabados en el directorio hlocal





Añadir fuentes a nuestro proyecto

- Aparece la siguiente ventana que indica que el fichero se está añadiendo al proyecto





Añadir fuentes a nuestro proyecto

- El fichero se ha añadido correctamente

ISE Project Navigator (P.15xf) - C:\Xilinx\ejemplo\ejemplo.xise - [sumador.vhd]

File Edit View Project Source Process Tools Window Layout Help

Design View: Implementation Simulation

Hierarchy

- ejemplo
 - xc3s1000-5ft256
 - sumador - Behavioral (sumador.vhd)

El fichero se ha incorporado al proyecto

```
11 -- Description:  
12 --  
13 -- Dependencies:  
14 --  
15 -- Revision:  
16 -- Revision 0.01 - File Created  
17 -- Additional Comments:  
18 --  
19  
20 library IEEE;  
21 use IEEE.STD_LOGIC_1164.ALL;  
22 use IEEE.STD_LOGIC_ARITH.ALL;  
23 use IEEE.STD_LOGIC_UNSIGNED.ALL;  
24  
25 -- Uncomment the following library declaration if using  
26 -- arithmetic functions with Signed or Unsigned values  
27 --use IEEE.NUMERIC_STD.ALL;  
28  
29 -- Uncomment the following library declaration if instantiating  
30 -- any Xilinx primitives in this code.  
31 --library UNISIM;  
32 --use UNISIM.VComponents.all;  
33  
34 entity sumador is  
35     Port ( A : in STD_LOGIC_VECTOR (3 downto 0);  
36             B : in STD_LOGIC_VECTOR (3 downto 0);  
37             C : out STD_LOGIC_VECTOR (3 downto 0));  
38 end sumador;  
39  
40 architecture Behavioral of sumador is
```

No Processes Running

Processes: sumador - Behavioral

- Design Summary/Reports
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File
- Configure Target Device
- Analyze Design Using ChipScope



Crear un proyecto (otra vez)

- Una vez usado el entorno, el ISE Project Navigator se suele abrir con el último proyecto con el que hemos trabajado

Para
crear un
nuevo
proyecto

The screenshot shows the ISE Project Navigator interface. On the left, the 'File' menu is open, with 'New Project...' highlighted. An arrow points from this menu to the 'New Project Wizard' dialog box on the right. The 'New Project Wizard' dialog has a title bar 'New Project Wizard' and a main section titled 'Create New Project'. It asks 'Specify project location and type.' and provides fields for 'Name:', 'Location:', 'Working Directory:', and 'Description:'. Below this, it says 'Select the type of top-level source for the project' with a dropdown set to 'HDL'. At the bottom are 'More Info', 'Next', and 'Cancel' buttons. A blue callout bubble points to the 'Name:' field with the text 'Siempre en hlocal'.



Anotaciones



Pantalla de desarrollo

ISE Project Navigator (P.15xf) - C:\Xilinx\practical\practical1.xise - [sumador.vhd*]

File Edit View Project Source Process Tools Window Layout Help

Design View: Implementation Simulation

Hierarchy

- practical
- xc3s1000-5ft256
- sumador - Behavioral (sumador)

No Processes Running

Processes: sumador - Behavioral

- Design Summary/Reports
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File
- Configure Target Device
- Analyze Design Using ChipScope

sumador.vhd*

```
12 --  
13 -- Dependencies:  
14 --  
15 -- Revision:  
16 -- Revision 0.01 - File Created  
17 -- Additional Comments:  
18 --  
19 --  
20 library IEEE;  
21 use IEEE.STD_LOGIC_1164.ALL;  
22  
23 -- Uncomment the following library declaration if using  
24 -- arithmetic functions with Signed or Unsigned values  
25 --use IEEE.NUMERIC_STD.ALL;  
26  
27 -- Uncomment the following library declaration if instantiating  
28 -- any Xilinx primitives in this code.  
29 --library UNISIM;  
30 --use UNISIM.VComponents.all;  
31  
32 entity sumador is  
33 end sumador;  
34  
35 architecture Behavioral of sumador is  
36 begin  
37  
38 end Behavioral;
```

Aquí se escribe el código VHDL que nos interese

Started : "Launching ISE Text Editor to edit sumador.vhd".
Launching Design Summary/Report Viewer...



Práctica 1.a

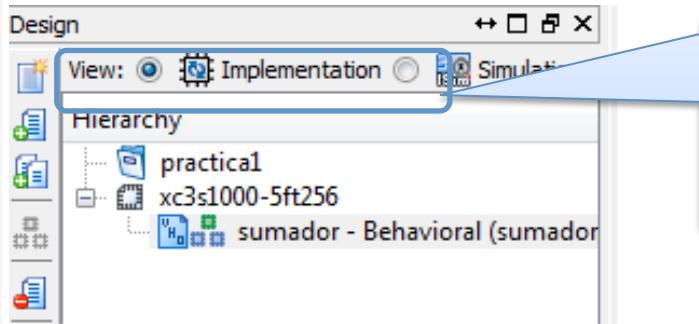
```
library IEEE;  
use IEEE.STD_LOGIC_1164.ALL;  
use IEEE.STD_LOGIC_ARITH.ALL;  
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

Diseñar en VHDL un circuito sumador de números de 4 bits

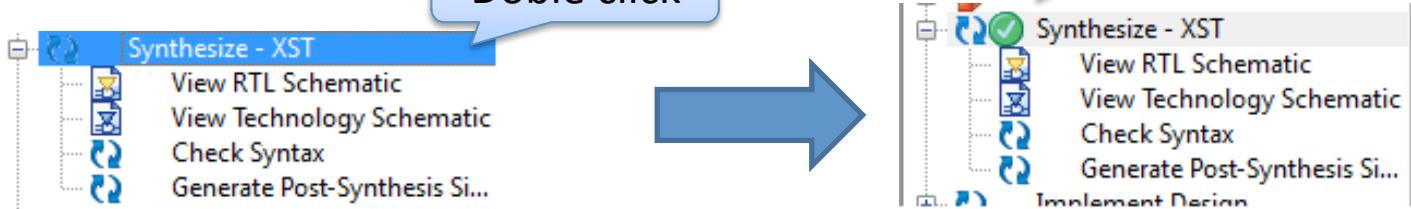
```
entity sumador is  
port (A, B: in std_logic_vector(3 downto 0);  
      C: out std_logic_vector(3 downto 0));  
end sumador;  
  
architecture beh of sumador is  
-- No hace falta definir señales intermedias  
begin  
  
C <= A + B;  
  
end beh;
```



Sintetizar el diseño



Para poder implementar el circuito tenemos que estar en la opción Implementation



Si sale este símbolo verde es que todo ha salido perfecto



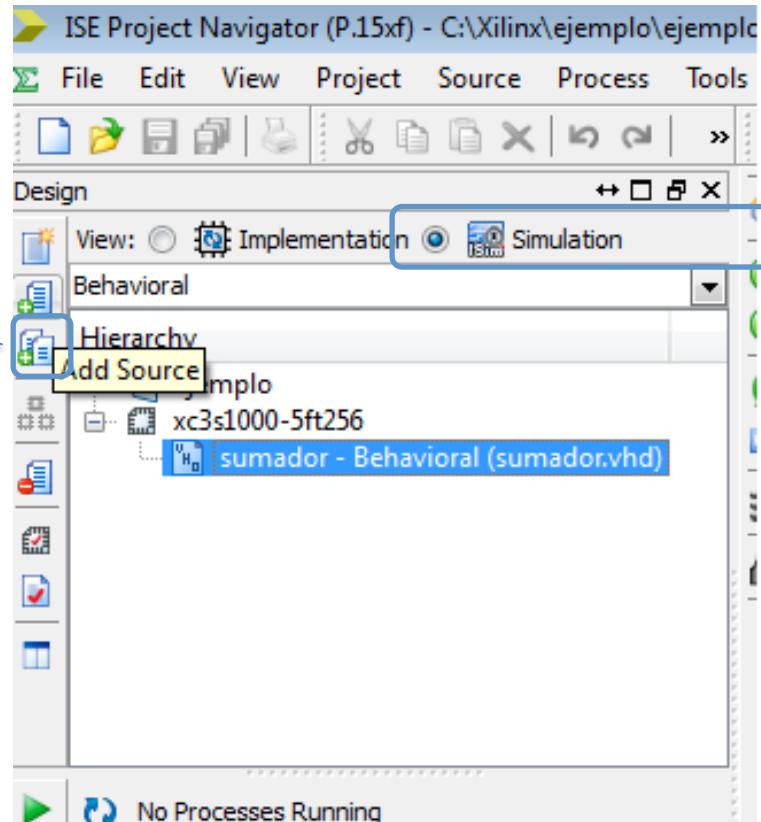
Comprobar que el Diseño es Correcto

- Para realizar esta comprobación hace falta simular su comportamiento:
 - En **casa y en el laboratorio** podéis crear automáticamente un test de simulación para comprobar que todo funciona correctamente. En esta primera práctica se os dará el fichero de simulación ya creado (ver transparencias Test de simulación)
 - Para **calificar la práctica** tendréis que pasar un archivo de simulación que se os dará en el laboratorio



Test de simulación (I)

- Añadir el fichero de simulación simsum.vhd



Pulsamos sobre
Add Source

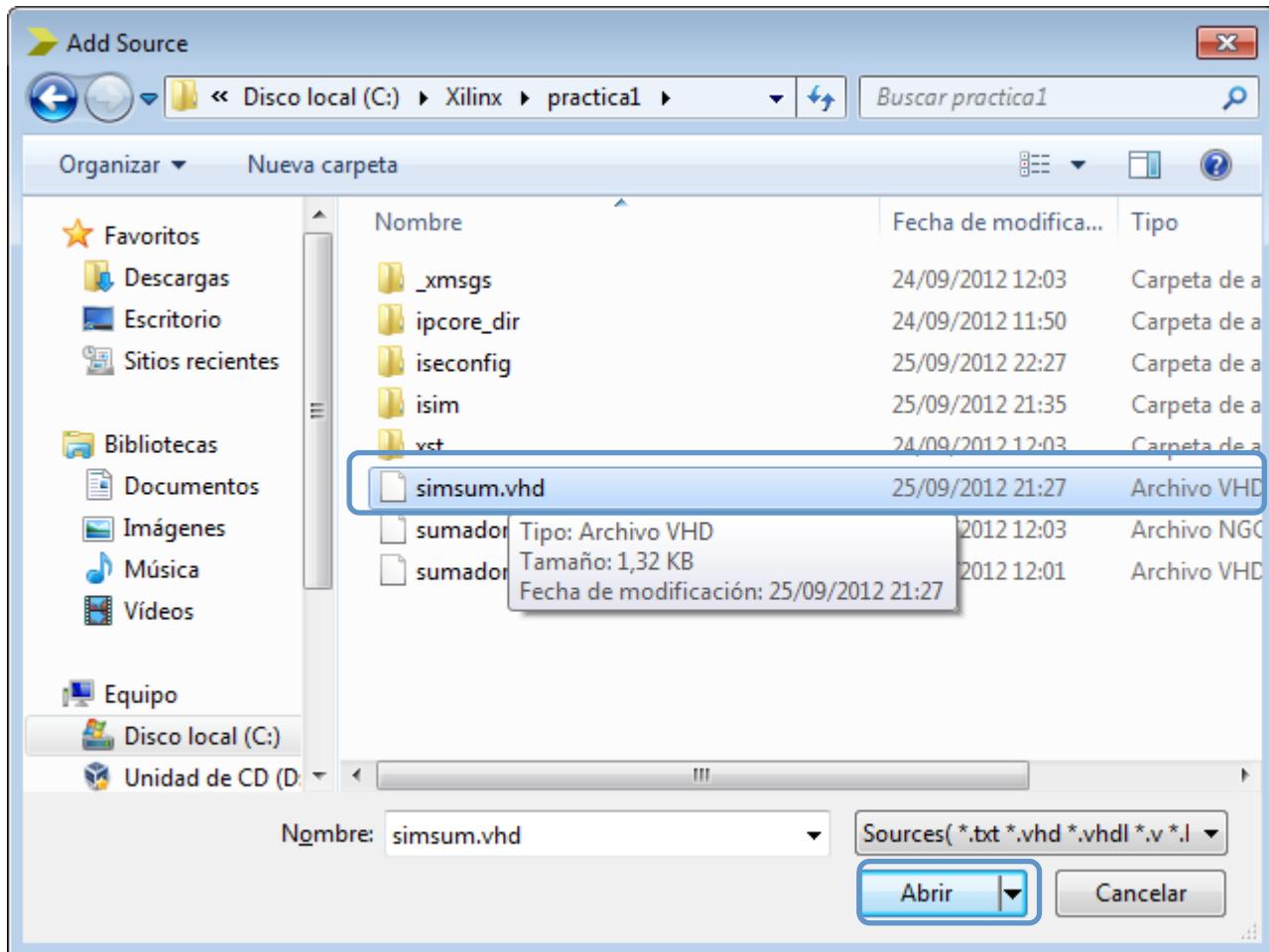
Marcamos la
opción Simulation

Importante: El fichero simsum.vhd se debe grabar en hlocal antes de añadirlo al proyecto



Test de simulación (II)

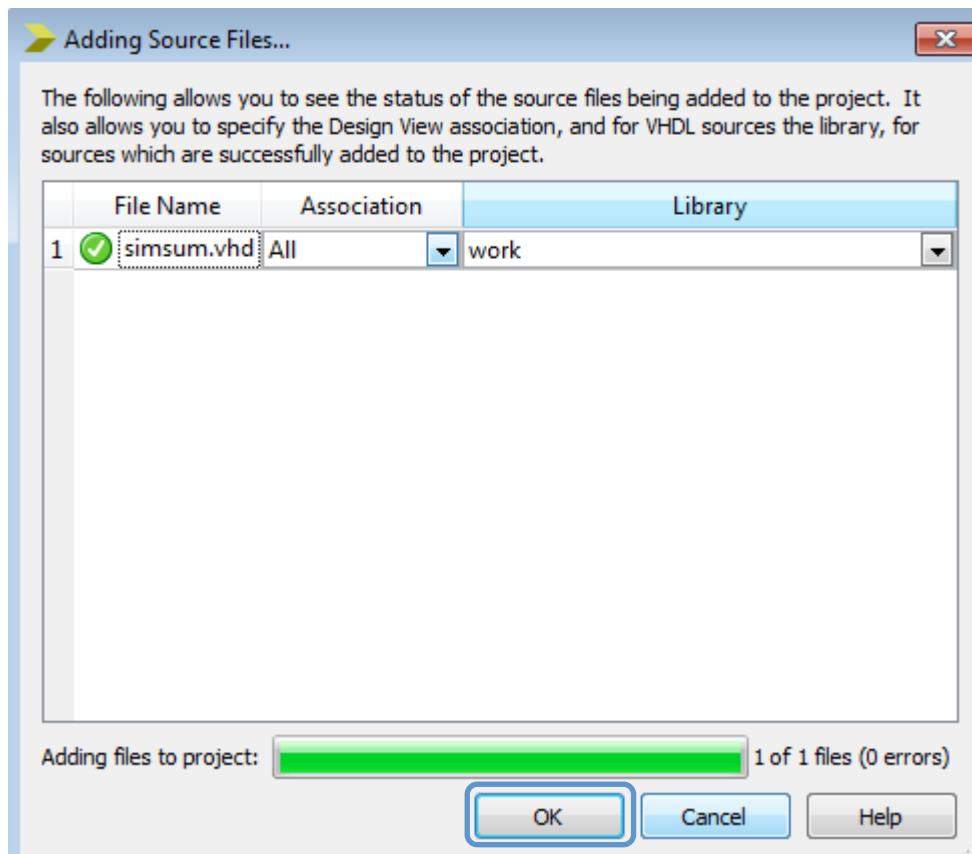
- Seleccionamos el fichero simsum.vhd





Test de simulación (III)

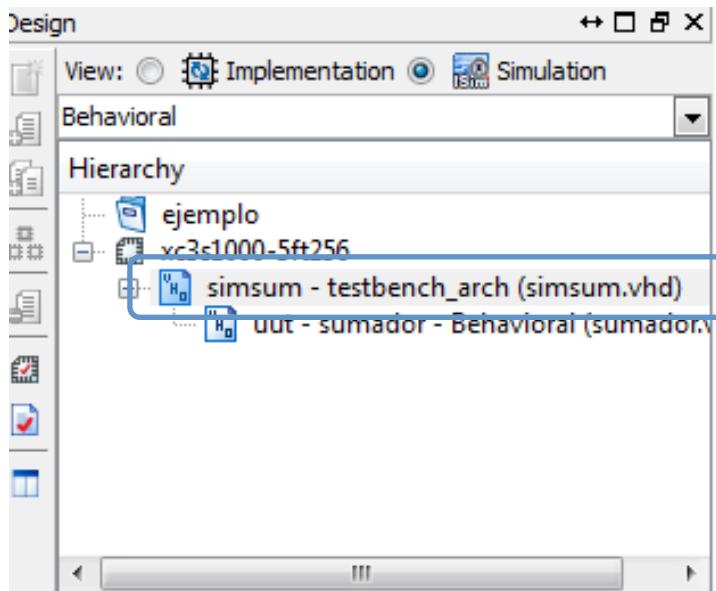
- El fichero se está añadiendo al proyecto





Test de simulación (IV)

- El fichero se ha añadido correctamente



Aparece en el árbol de ficheros



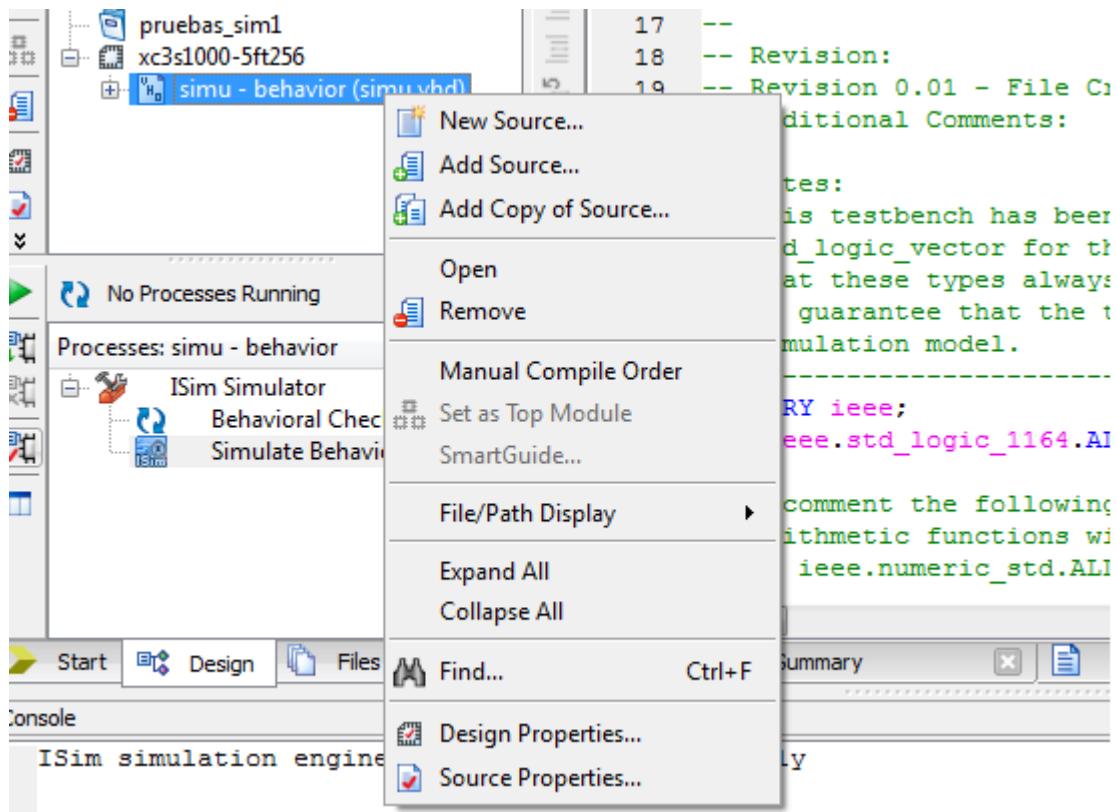
Fichero de simulación (I)

- Si el fichero de simulación se añade al proyecto no lo reconoce como fichero de simulación
- Asegurarse antes de hacer la implementación que se trata de un fichero de simulación



Fichero de simulación (II)

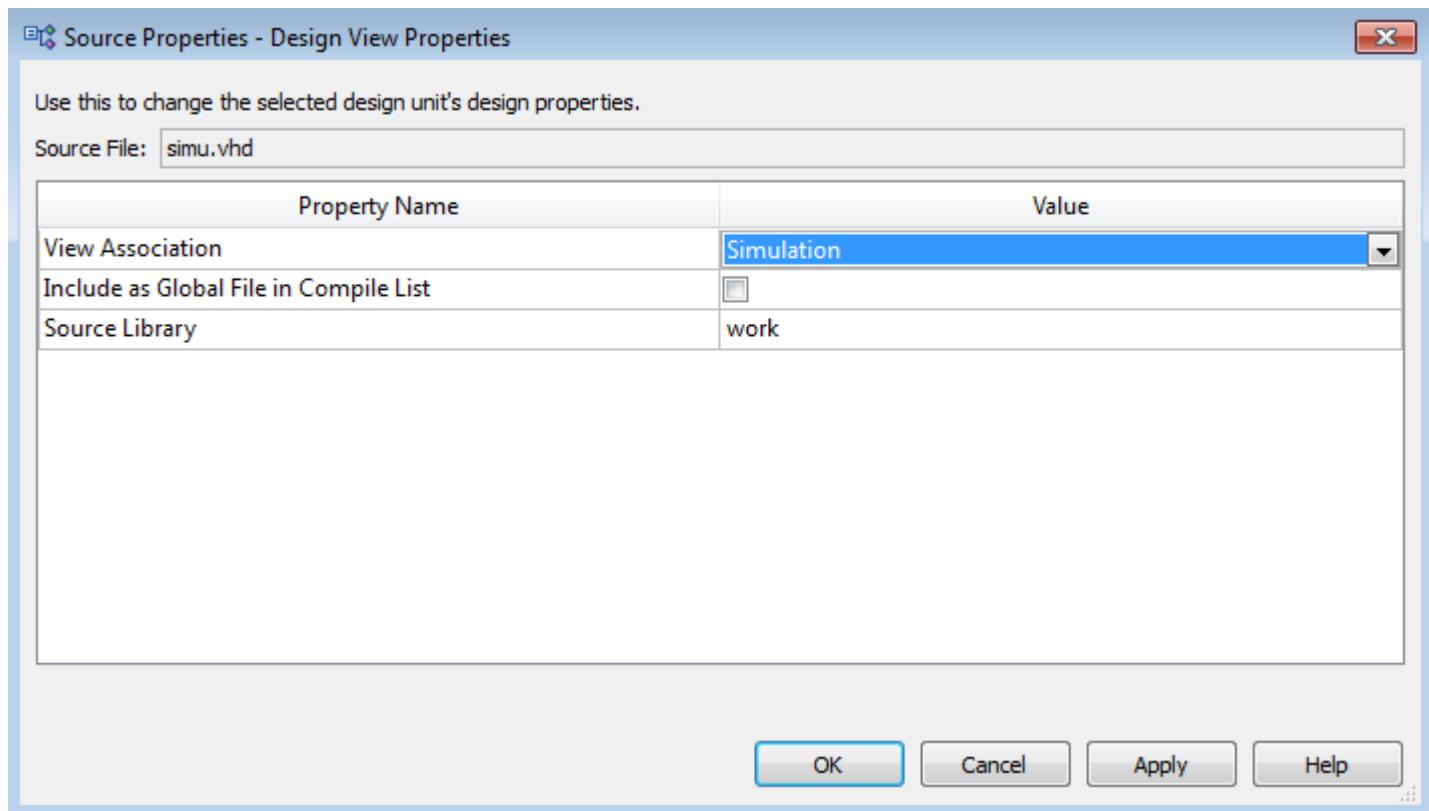
- Pulsar con el botón derecho sobre dicho fichero y elegir la opción *Source properties*





Fichero de simulación (III)

- En *View Association* aparecerá de tipo *All*, elegir tipo *Simulation*





Test de simulación (lo tenéis en el CV)

```
-- Añadimos las librerías necesarias
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.STD_LOGIC_TEXTIO.ALL;
use STD.TEXTIO.ALL;

--entidad
entity simsum is
end simsum;

--arquitectura
architecture testbench_arch of simsum is
-- Declaración del componente que vamos a simular
component sumador
    port( A : IN  std_logic_vector(3 downto 0);
          B : IN  std_logic_vector(3 downto 0);
          C : OUT  std_logic_vector(3 downto 0)
        );
end component;
--Entradas
signal op1      : std_logic_vector(3 downto 0) := (others => '0');
signal op2      : std_logic_vector(3 downto 0) := (others => '0');
--Salidas
signal result  : std_logic_vector(3 downto 0);
```



Test de simulación (lo tenéis en el CV)

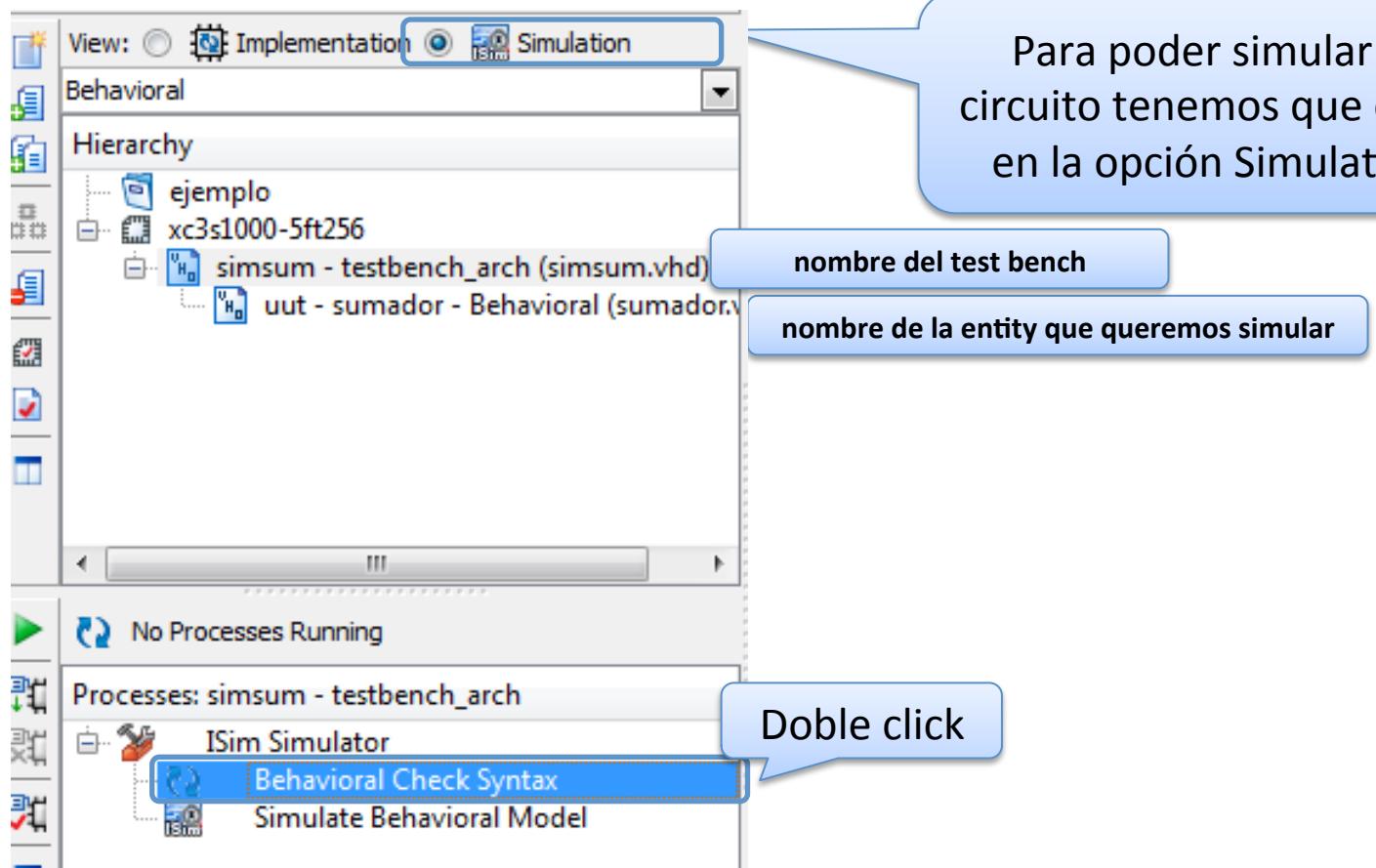
```
Begin
-- Instanciacion de la unidad a simular
uut: sumador port map (
    A => op1    ,
    B => op2    ,
    C => result
);
-- Proceso de estímulos
stim_proc: process
begin
    op1<="0000";
    op2<="0000";
    wait for 100 ns;
    op1<="0101";
    op2<="0100";
    wait for 100 ns;
    A<="0000";
    B<="0111";
    wait for 100 ns;
    A<="0011";
    B<="1000";
    wait for 100 ns;
    A<="1011";
    B<="1111";
    wait for 100 ns;
    A<="1001";
    B<="0110";
    wait;
end process;

end testbench_arch;
```



Simulación

■ Desplegar el menú de simulación



Para poder simular el circuito tenemos que estar en la opción **Simulation**

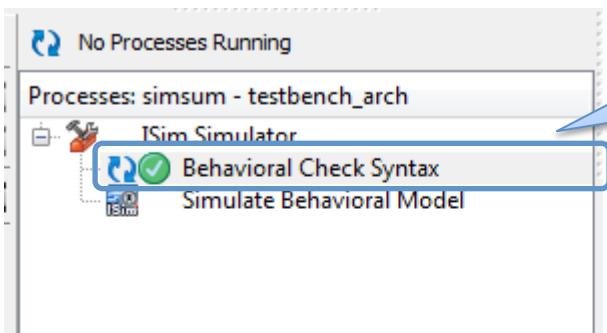
nombre del test bench

nombre de la entity que queremos simular

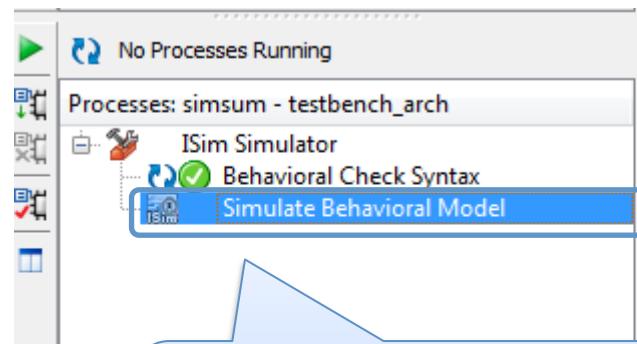
Doble click



Simulación



La simulación se ha ejecutado correctamente

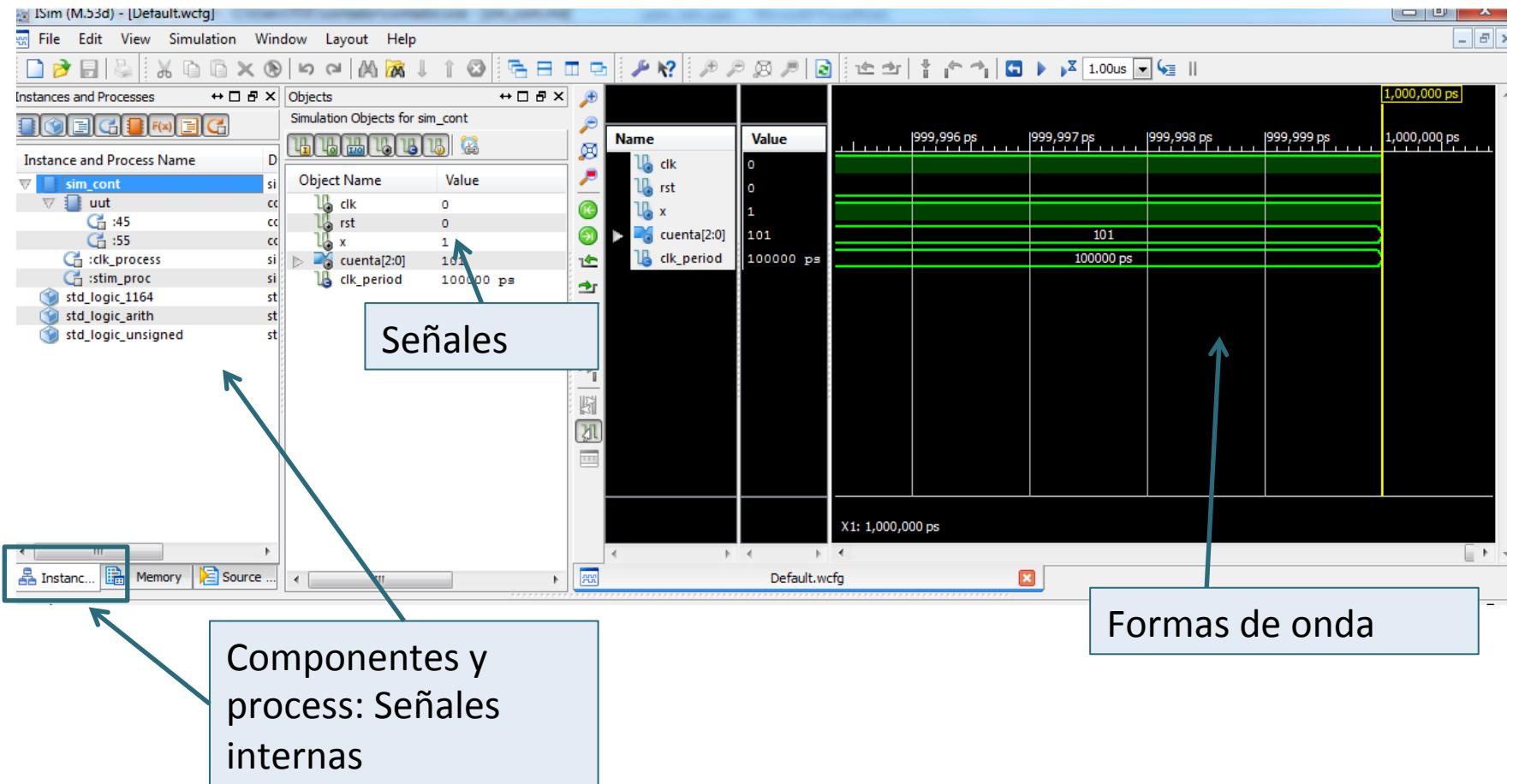


Necesitamos arrancar el ISIM para ver las formas de onda y comprobar que el circuito está bien diseñado



Simulación

■ Ventana simulador ISim



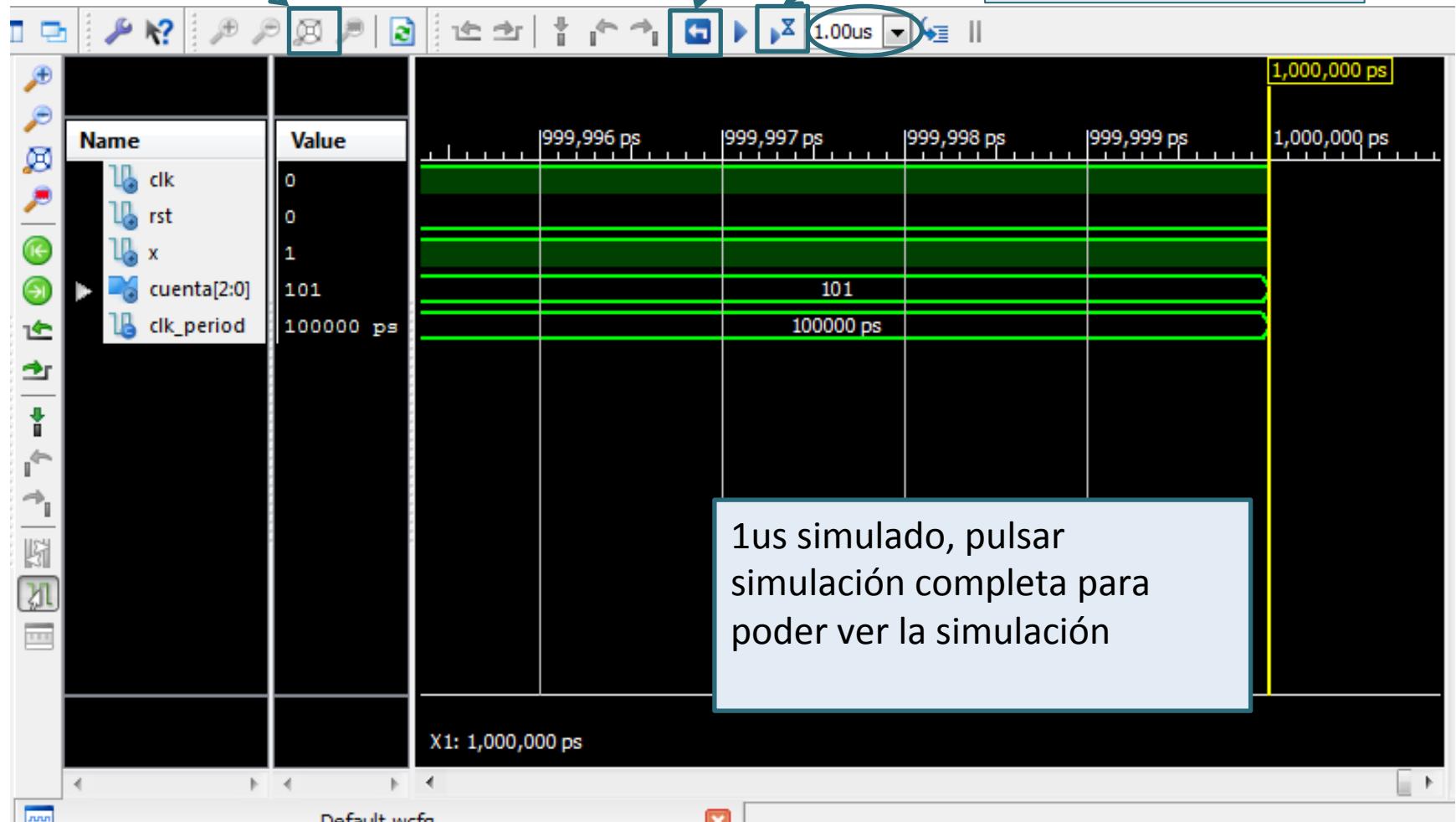


Simulación

Ver simulación completa

Reiniciar simulación

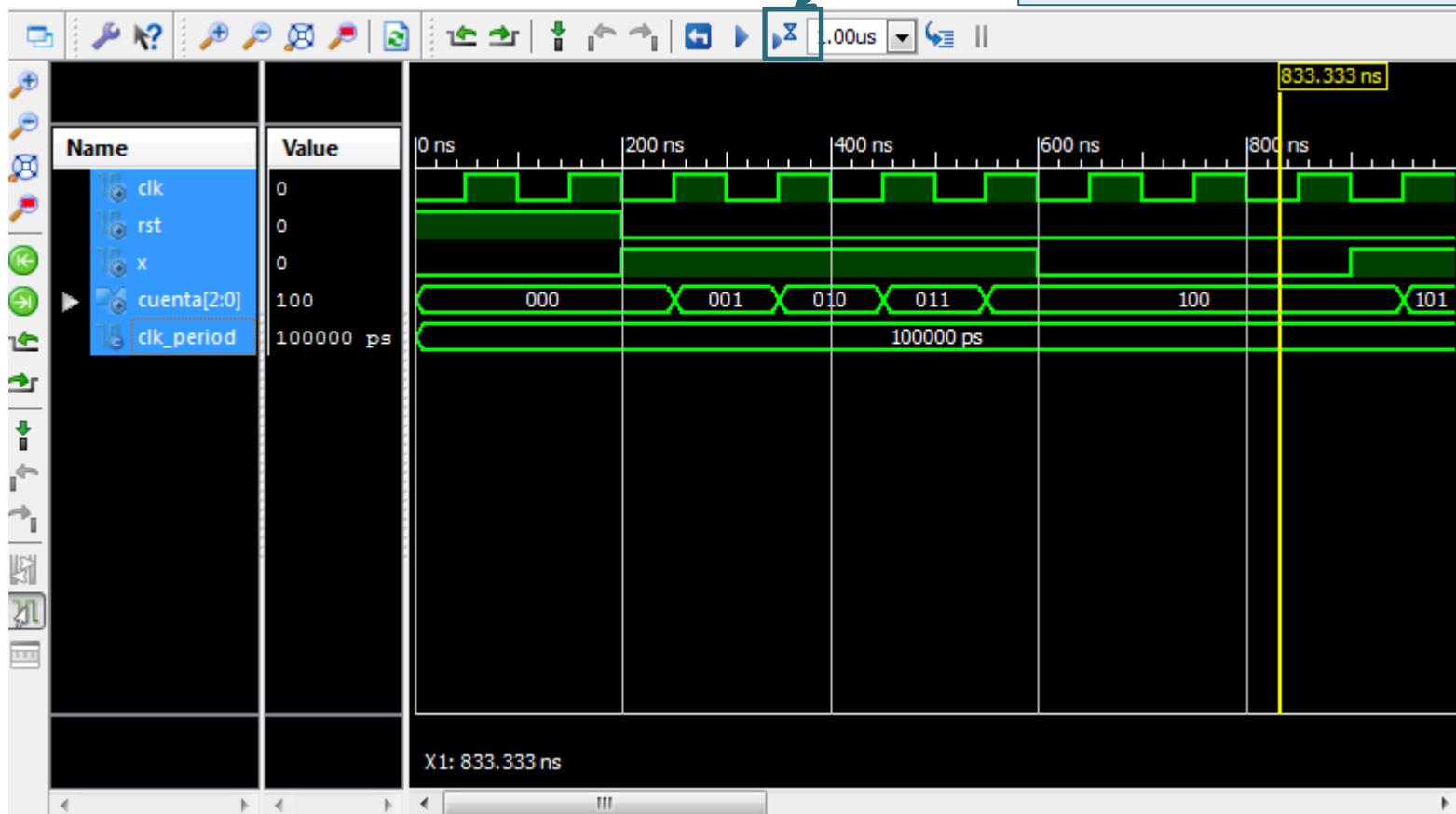
Cada vez que se pulsa este icono se simula 1us





Simulación

Si queremos simular más tiempo pulsar en este icono





Simulación

- Si la simulación no es correcta hay que depurar
 - Al abrir el simulador solo se muestra el valor de los puertos de entrada/salida
 - Es necesario para simular añadir las señales internas y los components

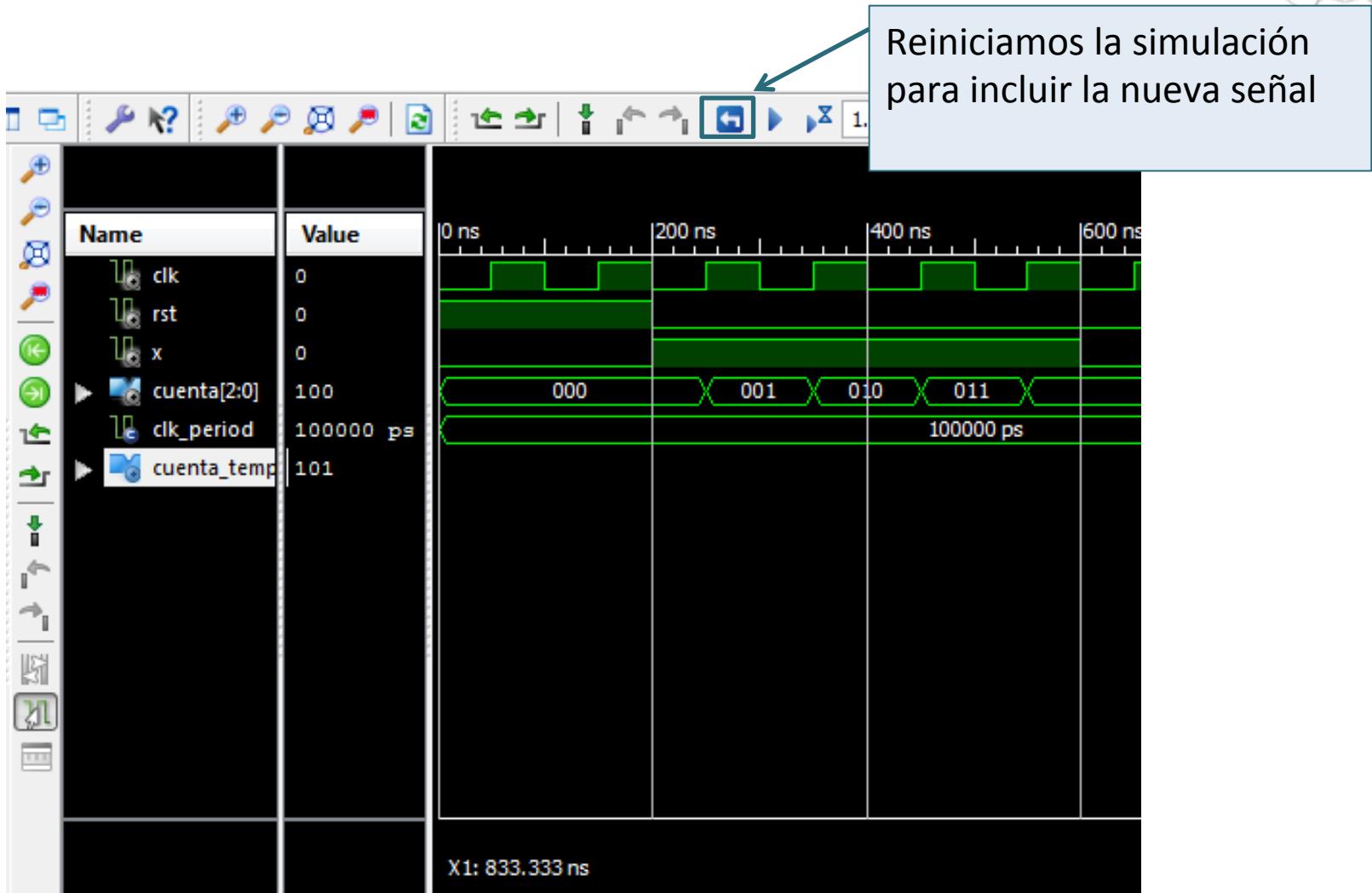
The screenshot shows the ModelSim simulation environment. On the left, the 'Instances and Processes' window lists simulation instances like 'sim_cont', 'uut', and various processes and std_logic components. The 'Objects' window shows simulation objects with their names and values, such as 'clk' (0), 'rst' (0), 'x' (1), and 'cuenta[2:0]' (101). A context menu is open over the 'Objects' window, with the 'Add To Wave Window' option highlighted. The 'Wave' window on the right displays waveforms for the signals, showing 'clk' at 0 ns, 'rst' at 0 ns, 'x' at 1, and 'cuenta[2:0]' at 101. The time scale is 100000 ps, and the current time is X1: 833.333 ns.

Buscar las señales internas, para añadirlas pulsar el botón derecho y elegir la opción marcada

This is a Full version of ISim.
Time resolution is 1 ns.

Simulación

- La señal se añadirá a la ventana de simulación





Anotaciones



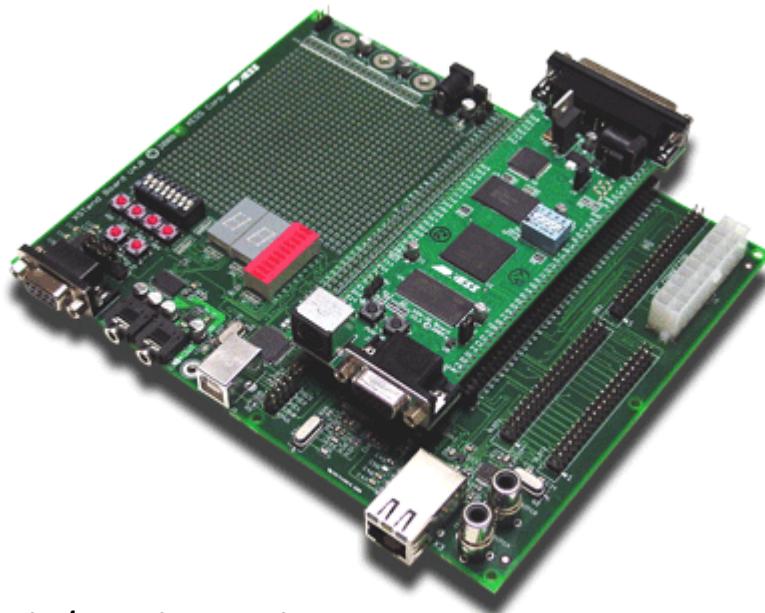
Implementación

- Podemos comprobar que el circuito funciona perfectamente en el *mundo real*
 - El circuito se va a implementar sobre una FPGA
 - Hay que indicar de dónde se leen las entradas (switches) y en dónde se escriben las salidas (LEDs o display de 7 segmentos)
 - Hay que añadir un fichero UCF donde se le indica a la herramienta lo anterior



Implementación

- Esta es la plataforma sobre la que se van a implementar todos los diseños de este curso:



<http://www.xess.com/prods/prod039.php>



Implementación

■ Fichero UCF

- En el Campus Virtual encontraréis el fichero UCF completo
- Aquí aparecen los pines particulares que se necesitan para la práctica 1.a

```
#switches placa extendida
NET A<0> LOC=P12;
NET A<1> LOC=J1;
NET A<2> LOC=H1;
NET A<3> LOC=H3;
NET B<0> LOC=G2;
NET B<1> LOC=K15;
NET B<2> LOC=K16;
NET B<3> LOC=F15;
```

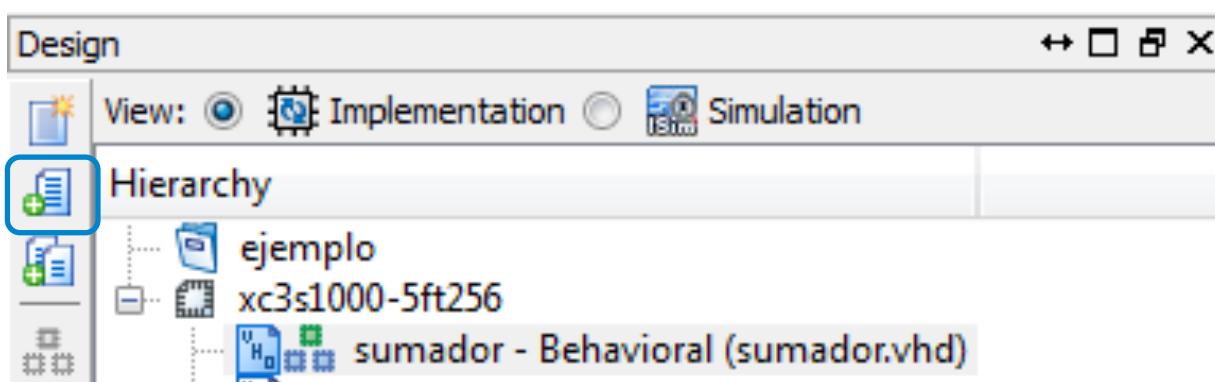
```
#barra de leds placa extendida
NET C<0> LOC=L5;
NET C<1> LOC=N2;
NET C<2> LOC=M3;
NET C<3> LOC=N1;
```

- En el fichero UCF deben aparecer todos los puertos de entrada/salida de la entity
- El puerto A<0> significa “puerto A, bit 0”
- A cada puerto de la entity se le asigna un pin físico de la FPGA
- Para que lo anterior sea efectivo hay que descomentar la línea correspondiente (quitar #)
- Los pines de la FPGA se instancian LOC = letra+número



Implementación

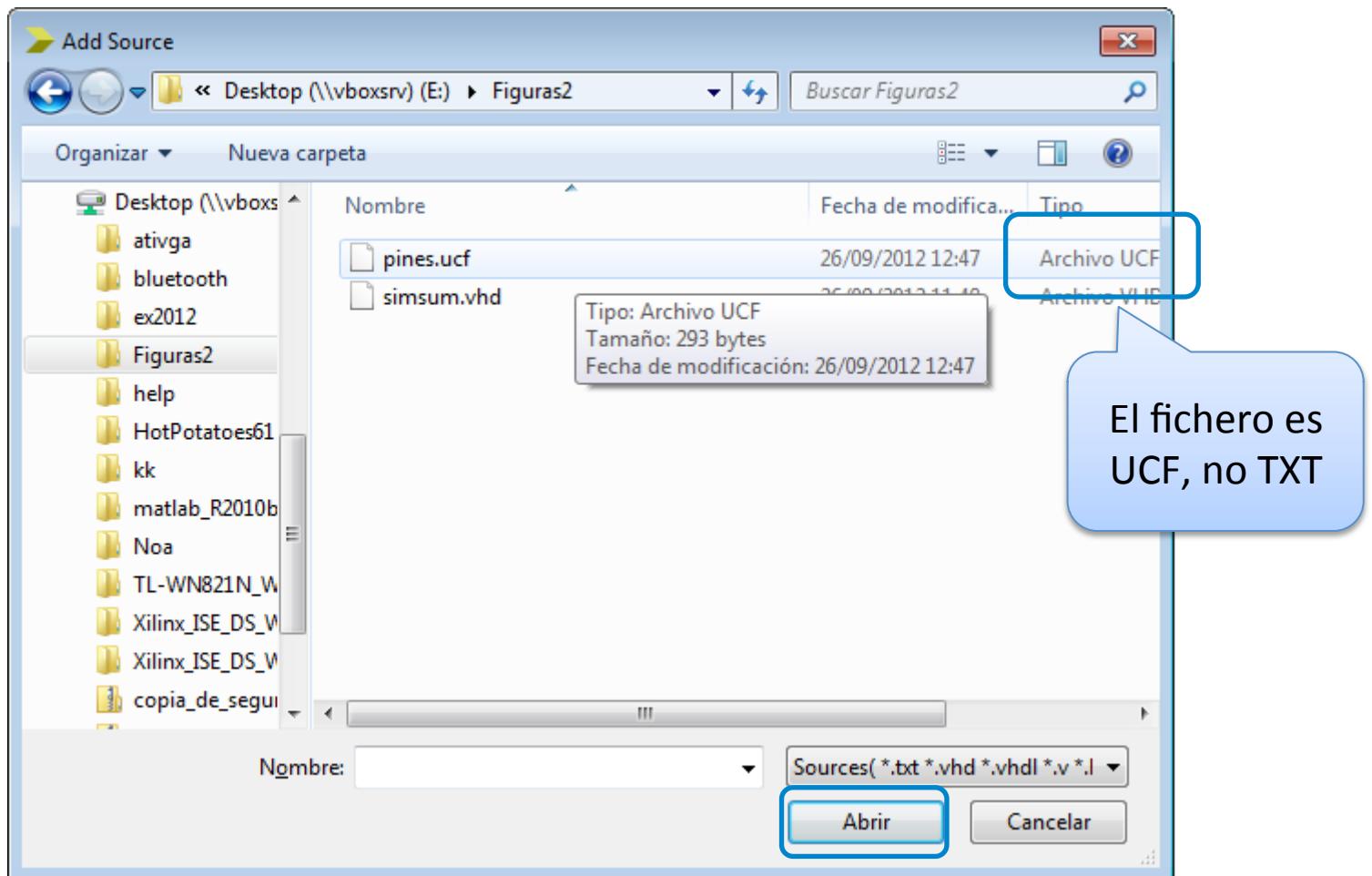
- El fichero pines.ucf se añade al proyecto mediante *Add Source*





Implementación

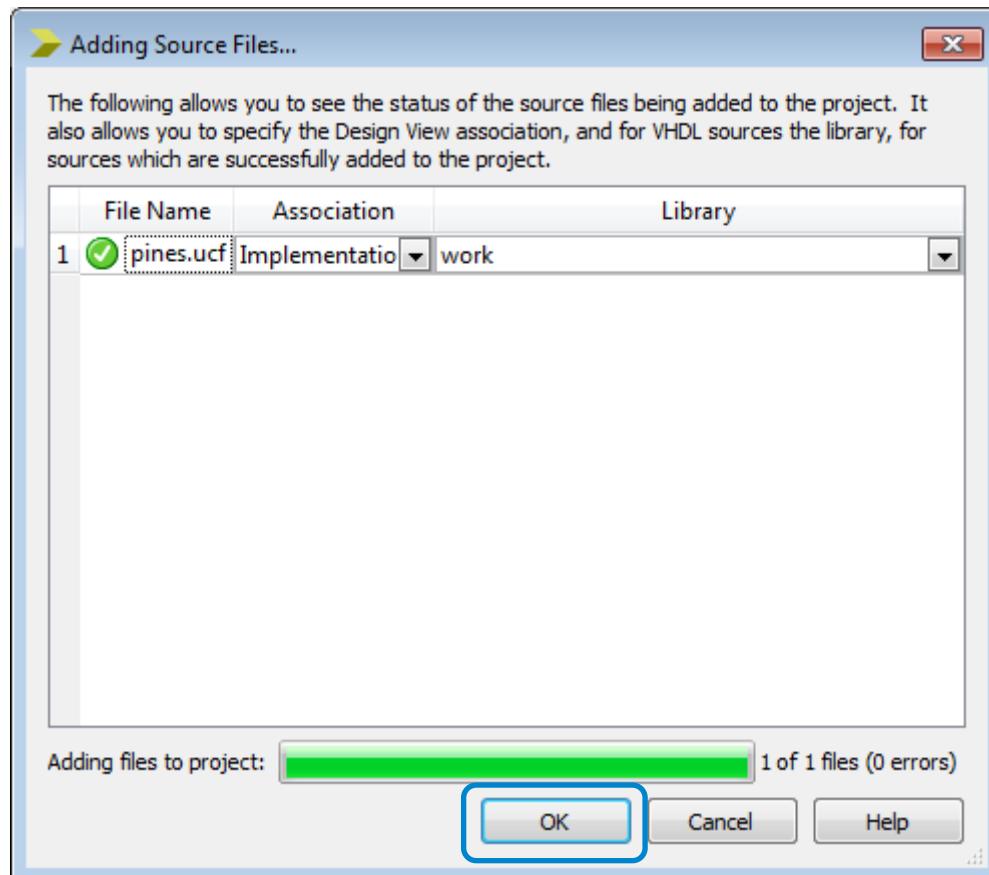
- Seleccionamos el fichero pines.ucf





Implementación

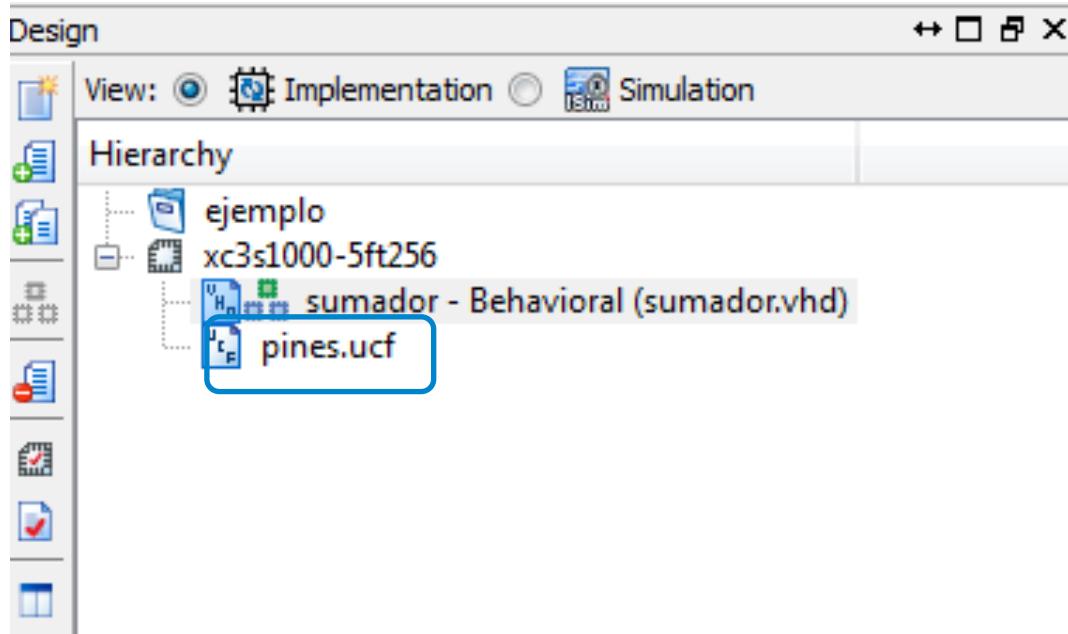
- El fichero se añade al proyecto





Implementación

- El fichero aparece en el árbol de jerarquía





Implementación

The screenshot shows the Xilinx Vivado software interface. A blue callout bubble points to the "Implement Design" option in the top-level menu bar. The "Implement Design" menu is open, displaying the following sub-options:

- Translate
 - Generate Post-Translate Simulation Model
- Map
 - Generate Post-Map Static Timing
 - Manually Place & Route (FPGA Editor)
 - Generate Post-Map Simulation Model
- Place & Route
 - Generate Post-Place & Route Static Timing
 - Analyze Timing / Floorplan Design (PlanAhead)
 - View/Edit Routed Design (FPGA Editor)
 - Analyze Power Distribution (XPower Analyzer)
 - Generate Text Power Report
 - Generate Post-Place & Route Simulation ...
- Generate IBIS Model
- Back-annotate Pin Locations

Doble click

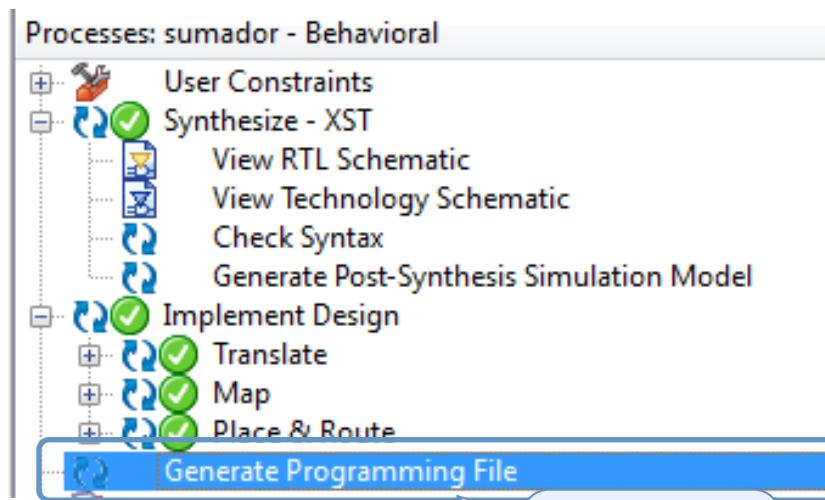
Si todo está correcto
aparecerá en:

- Implementation Design
- Map
- Place & Route

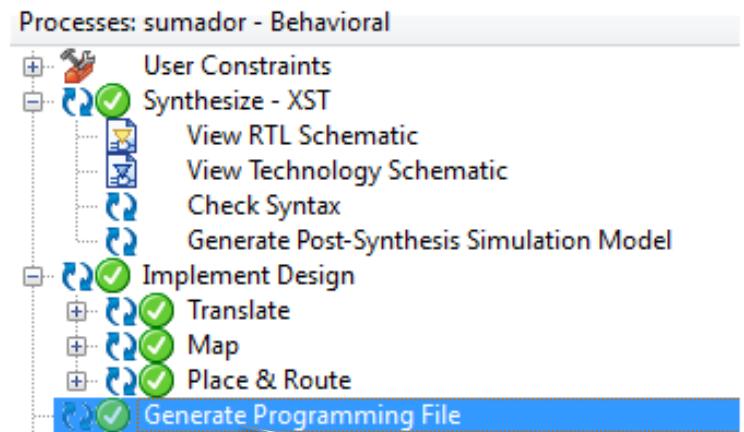


Implementación

- Se genera un fichero de 0s y 1s que describe el circuito, se podrá encontrar en la carpeta del proyecto (nombre_proyecto.bit)



Doble click



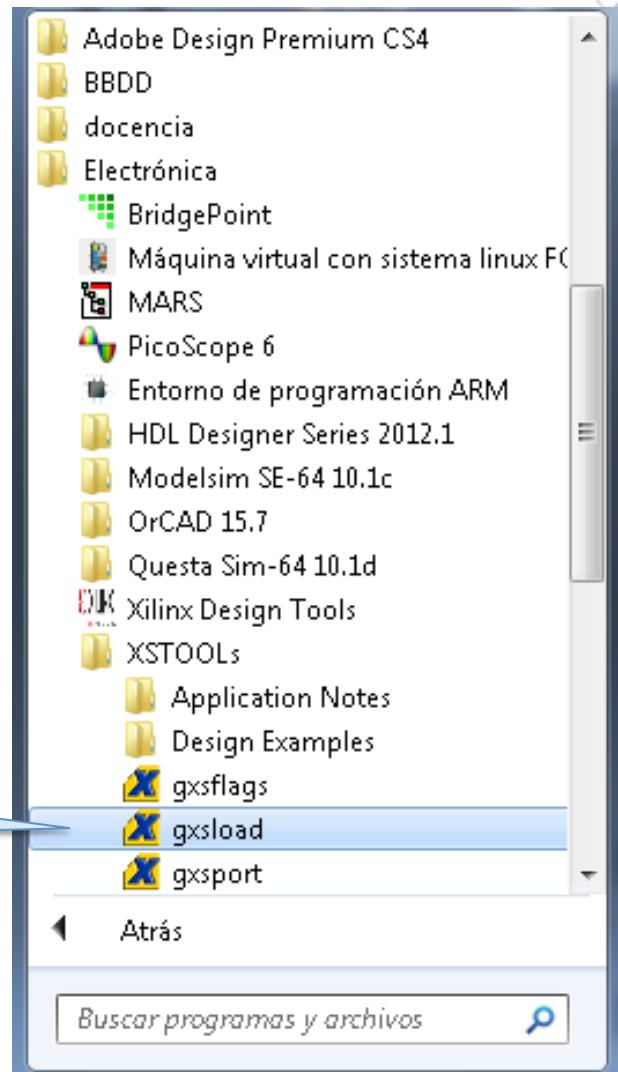
Si todo ha salido
correctamente



Descarga .bit sobre FPGA

- Comprobar jumper J9 de la FPGA en la posición XS, de no ser así solicitar otra FPGA
- Para poder volcar el fichero.bit a la FPGA, abrir el programa gxsload, que se encuentra en “Inicio->Electrónica->XSTools”

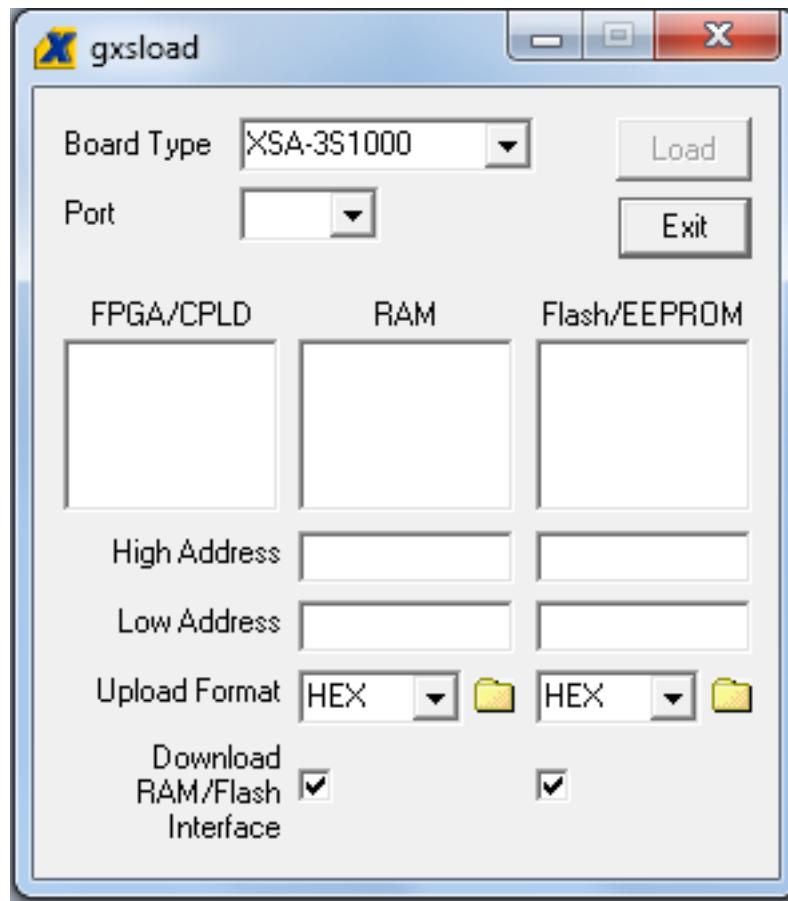
El programa es
accesible a través
de esta ruta





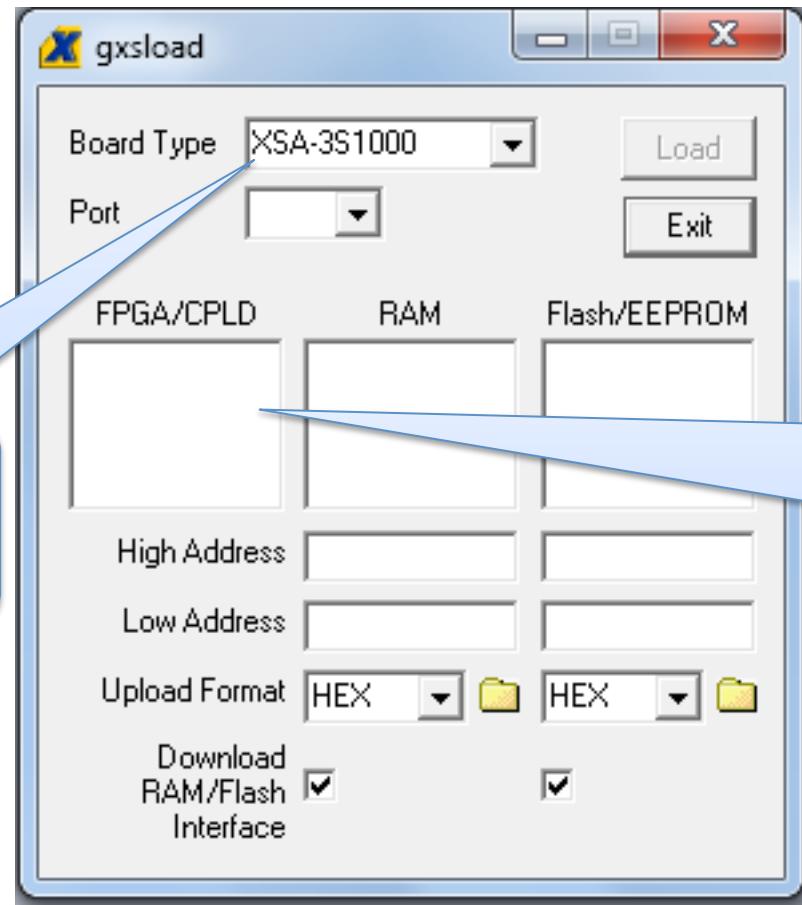
Descarga .bit sobre FPGA

- Se abrirá la ventana de gxsload





Descarga .bit sobre FPGA

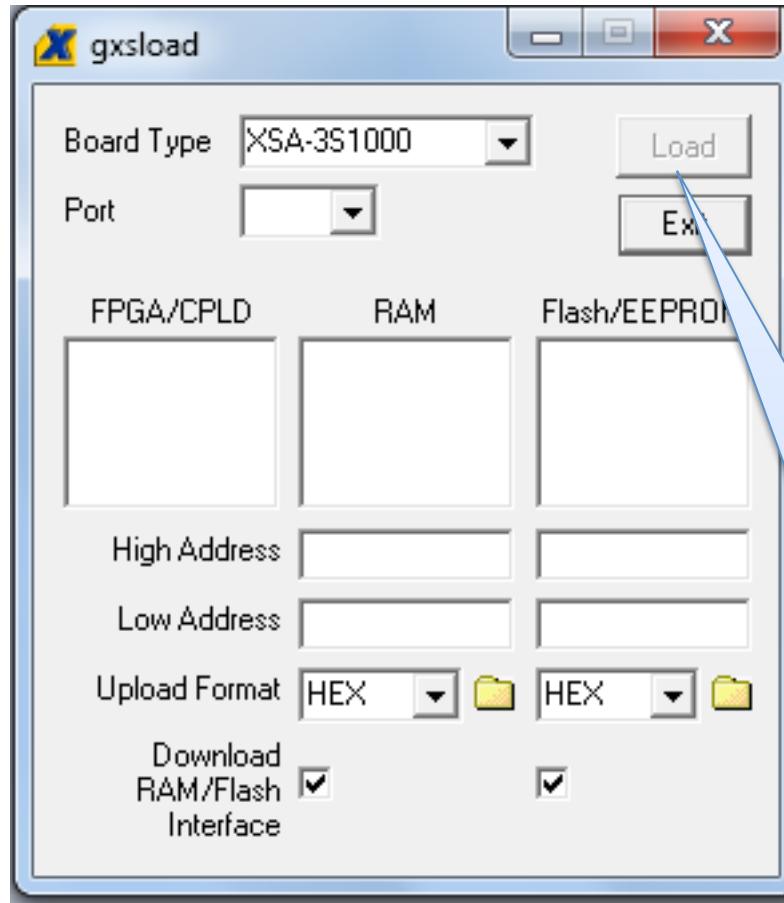


Aquí seleccionamos nuestro modelo de FPGA

En esta ventana arrastramos y soltamos el fichero .bit que acabamos de crear



Descarga .bit sobre FPGA



A continuación,
pulsamos el botón
“Load” y la carga
comienza



Descarga .bit sobre FPGA

- Mientras se hace la carga se verá una barra de progreso en la parte inferior de la ventana de gxsload
- Después de esto, ya se puede comprobar sobre la FPGA que nuestro diseño funciona
 - Cambiad las posiciones de los switches de la placa extendida y veréis que el resultado de la suma aparecerá en la barra de LEDs



Anotaciones



Práctica 1.b

- Simular e implementar un registro de entrada/salida paralela



Práctica 1.b

```
-- registro de desplazamiento entrada-salida paralelo
```

```
library IEEE;
use IEEE.std_logic_1164.all;

entity reg_paralelo is
    port (rst, clk, load: in std_logic;
          E: in std_logic_vector(3 downto 0);
          S: out std_logic_vector(3 downto 0));
end reg_paralelo;
```

```
architecture circuito of reg_paralelo is
```

Fichero p1b.vhd



Práctica 1.b

Begin

```
--Añadimos el resto del codigo del registro paralelo
process(rst, clk)
begin
    if rst='1' then
        S<="0000";
    elsif clk'event and clk='1' then
        if load='1' then
            S<=E;
        end if;
    end if;
end process;
end circuito;
```



Implementación Práctica 1.b

- Pin del reloj:

NET clk LOC=T9;

- Hasta que no se resetee el circuito no funcionará correctamente



Anotaciones



Calificación

- Seguiréis las instrucciones de las transparencias para simular e implementar los diseños de la práctica 1.a y 1.b.
- Debéis enseñar al profesor del laboratorio cada una de las partes funcionando sobre la FPGA (+0.2).
- Habrá un apartado OPCIONAL, que os permitirá sumar otro +0.2.



Calificación

- Simulación para calificación:
 - En la sesión de laboratorio el alumno dispondrá de dos ficheros VHDL
 - simu_1a.vhd
 - simu_1b.vhd
 - Añadir los ficheros de simulación
 - Add Source