

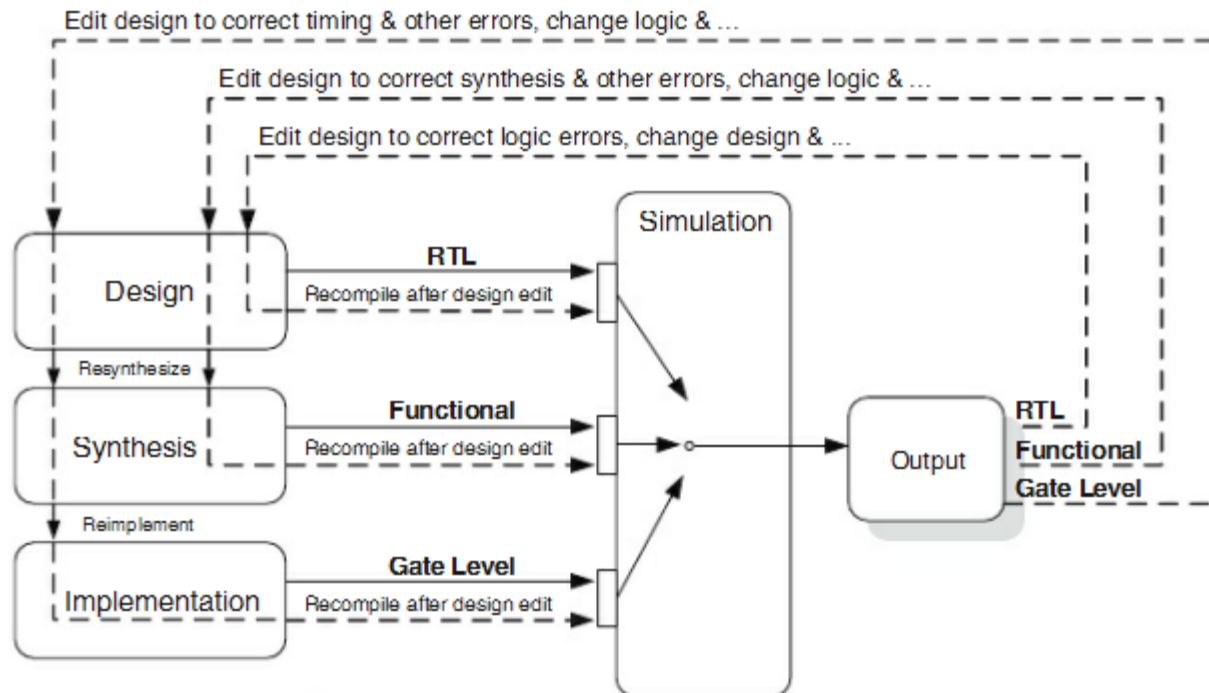
## **TDC\_Práctica2. Simulación**

## TDC\_Práctica2. Simulación

- **Simular** consiste en aplicar **estímulos** a las entradas del diseño y observar los valores de sus salidas, **respuestas**.
- **Herramientas de simulación**: herramienta que compila o conecta estímulos al diseño (**Unit Under Device**) y tras su ejecución muestra los valores de salida en tres formatos:
  - Lista de datos
  - Forma de onda
  - Fichero de texto
- **Simuladores**:
  - De terceras compañías: ModelSim de Mentor Graphics
  - Libres y gratuitos (GHDL)
  - De los fabricantes de FPGA: **Xilinx ISim**

## TDC\_Práctica2. Simulación

- Hay tres tipos de simulación
  - **RTL**: se realiza en la fase de codificación. No incluye información de tiempo.
  - **Funcional**: se realiza tras las síntesis a partir de la netlist generada por ésta. Incluye información de tiempo pero estimada.
  - **A nivel de puertas**: se realiza tras la implementación. Información de tiempo real. Simulación más realista

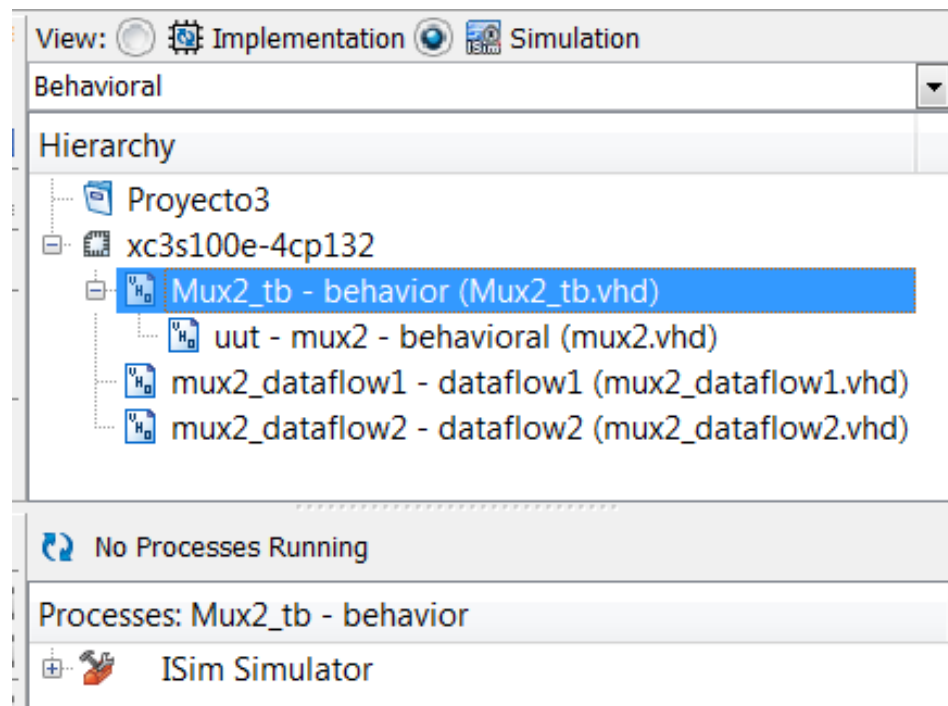


- Métodos para la creación de estímulos
  - Descripción del banco de pruebas (testbench) en HDL
    - Válido para cualquier herramienta de simulación.
    - Se puede usar un VHDL más amplio que el de Síntesis
  - Mediante interfaz gráfico (GUI)
    - Válido para un simulador concreto
    - Asistente
    - Línea de comandos

# TDC\_Práctica2. Simulación

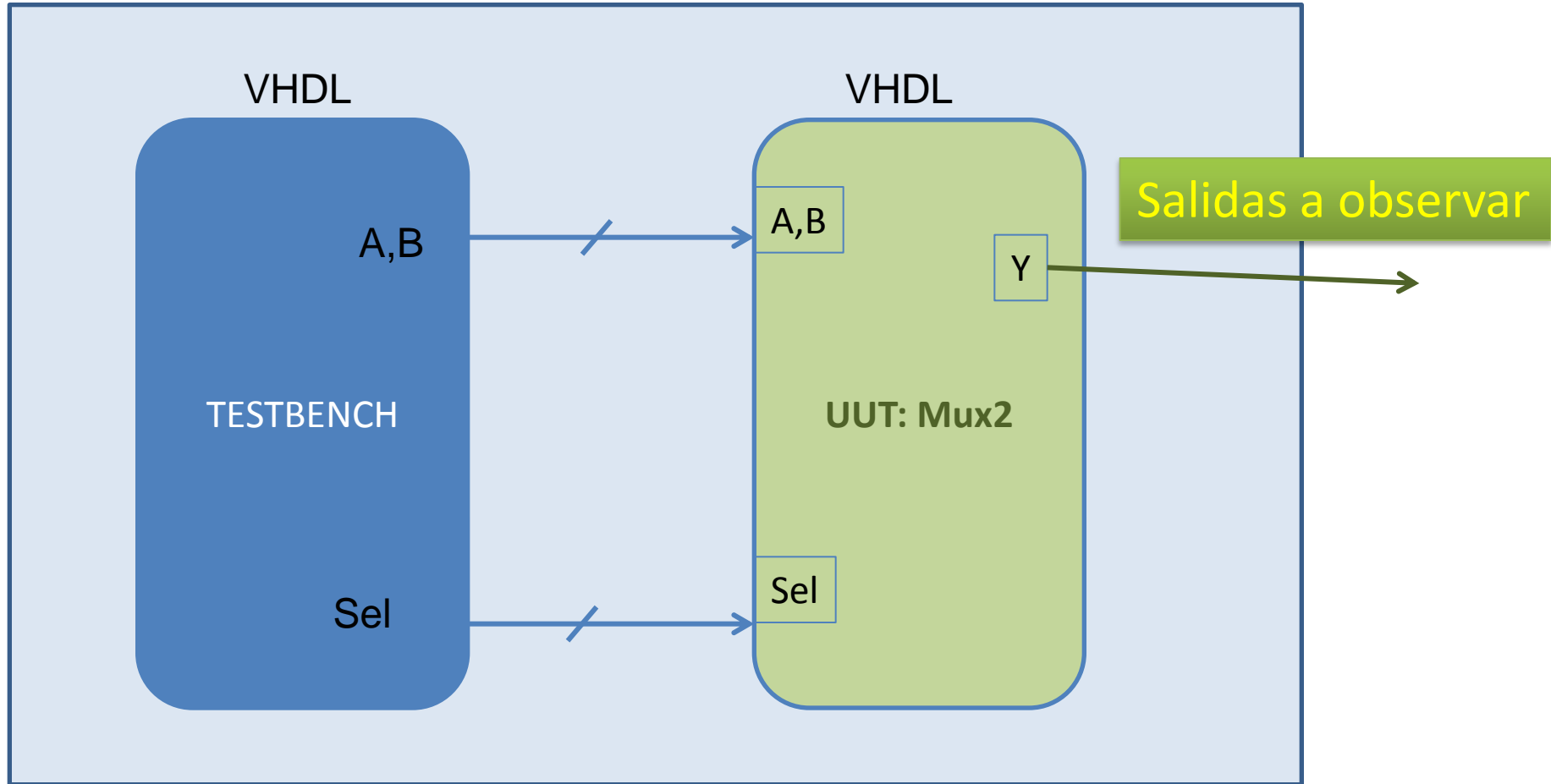
## EJEMPLO. Generar estímulos mediante VHDL

1. Abrir el **Proyecto03. Multiplexor de 2 entradas**.
2. Añadir un nuevo fichero del tipo **VHDL Test Bench**. Dar el mismo nombre que a la entidad añadiéndole el sufijo **\_tb**: “**Mux2\_tb**”.
3. De vuelta en la ventana principal de ISE, seleccionar el nuevo fichero y cambiar la vista de *Implementation* a *Simulation*



# TDC\_Práctica2. Simulación

TOP LEVEL



# TDC\_Práctica2. Simulación

## Diseño del Testbench en VHDL: librerías, entidad y declaración del componente UUT

```
LIBRARY ieee;  
USE ieee.std_logic_1164.ALL;  
-- Uncomment the following library declaration if using  
-- arithmetic functions with Signed or Unsigned values  
--USE ieee.numeric_std.ALL;  
ENTITY Mux2_tb IS  
END Mux2_tb ;
```

1

La entidad carece de  
entradas/salidas hacia el  
exterior

```
ARCHITECTURE behavior OF Mux2_tb IS  
-- Component Declaration for the Unit Under Test (UUT)  
COMPONENT Mux2  
    PORT ( A : IN std_logic;  
          B : IN std_logic;  
          Sel: IN std_logic;  
          Y  : OUT std_logic);  
END COMPONENT;
```

La UUT se instancia como un  
componente. (Estructural)

# TDC\_Práctica2. Simulación

## Diseño del Testbench en VHDL: Declaración de señales internas

--Inputs

```
signal A : std_logic := '0';  
signal B : std_logic := '0';  
signal sel : std_logic := '0';
```

--Outputs

```
signal Y : std_logic;
```

-- No clocks detected in port list. Replace <clock> below with  
-- appropriate port name

```
constant <clock>_period : time := 10 ns;
```

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

```
 uut: Mux2 PORT MAP (  
     A => A,  
     B => B,  
     Sel => Sel,  
     Y => Y );
```

Se declaran las señales que conectarán el TESTBENCH con la UUT. Se asignan valores por defecto para esas señales.

Eliminar esta línea para este ejemplo. No hay señal de reloj.



# TDC\_Práctica2. Simulación

## Diseño del Testbench en VHDL: Definición de procesos (estímulos)

```
-- Clock process definitions
```

```
<clock>_process :process
```

```
begin <clock> <= '0';
```

```
wait for <clock>_period/2;
```

```
<clock> <= '1';
```

```
wait for <clock>_period/2;
```

```
end process;
```

Eliminar estas líneas  
para este ejemplo: no  
hay reloj

```
-- Stimulus process
```

```
stim_proc: process
```

```
begin
```

```
-- hold reset state for 100 ns.
```

```
wait for 100 ns;
```

```
wait for <clock>_period*10;
```

```
-- insert stimulus here
```

```
wait;
```

```
end process;
```

```
END;
```

Eliminar estas líneas  
para este ejemplo: no  
hay reset

Añadir aquí los  
estímulos para A,B y  
Sel. (Ver siguiente)

## TDC\_Práctica2. Simulación

```
-- Stimulus process
stim_proc: process
begin
    -- insert stimulus here

    -----
    -- Combinación 1: 40ns de espera
    -- antes de dar valor a "sel".
    -----

    A <= '0' ;
    B <= '0' ;
    wait for 40 ns;
    sel <= '0' ;      --Selecciona la entrada A
    wait for 60 ns;  --Mantiene este valor 60ns

    -----
    -- Combinación 2: 40ns de espera
    -- antes de dar valor a "sel".
    -----

    A <= '0' ;
    B <= '1' ;
    wait for 40 ns;
    sel <= '0' ;      --Selecciona la entrada A
    wait for 60 ns;  --Mantiene este valor 60ns
```

T0 = 0 ns

Dura 100ns  
en total

T1 = 100 ns

T2 = 200 ns

# TDC\_Práctica2. Simulación

```
-- Combinación 7: 40ns de espera  
-- antes de dar valor a "sel".  
-----
```

T7 = 600 ns

```
A <='1';  
B <='0';  
wait for 40ns;  
sel<='1';      --Selecciona la entrada B  
wait for 60ns; --Mantiene este valor 60ns mas
```

Dura 100ns  
en total

```
-----  
-- Combinación 8: 40ns de espera  
-- antes de dar valor a "sel".  
-----
```

T8 = 700 ns

```
A <='1';  
B <='1';  
wait for 40ns;  
sel<='1';      --Selecciona la entrada B  
wait for 60ns; --Mantiene este valor 60ns mas
```

Dura 100ns  
en total

```
wait;--Espera, no se repite el proceso
```

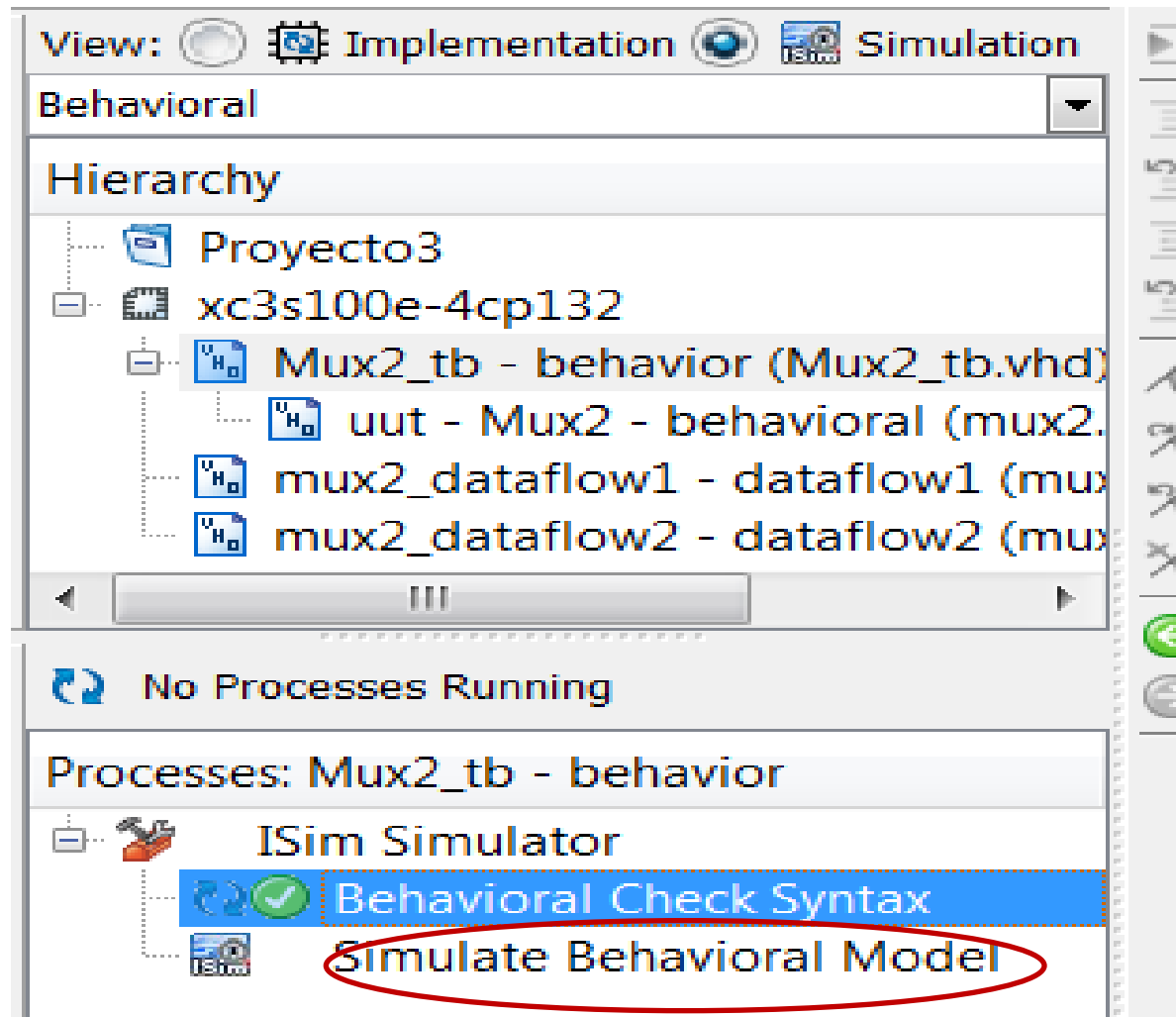
T7 = 800 ns

```
end process;
```

```
END;
```

## TDC\_Práctica2. Simulación

- Comprobar la sintaxis del módulo de testbench
- Simular



# TDC\_Práctica2. Simulación

**Componentes**

**Señales**

Name	Value
a	1
b	0
sel	0
y	1

**Señales observadas**

Console

ISim P.28xd (signature 0x1048c146)

WARNING:Security:42 - Your software subscription period has lapsed. Your current version of Xilinx tools will continue to function, but you no longer qualify for Xilinx software updates or new releases.

-----

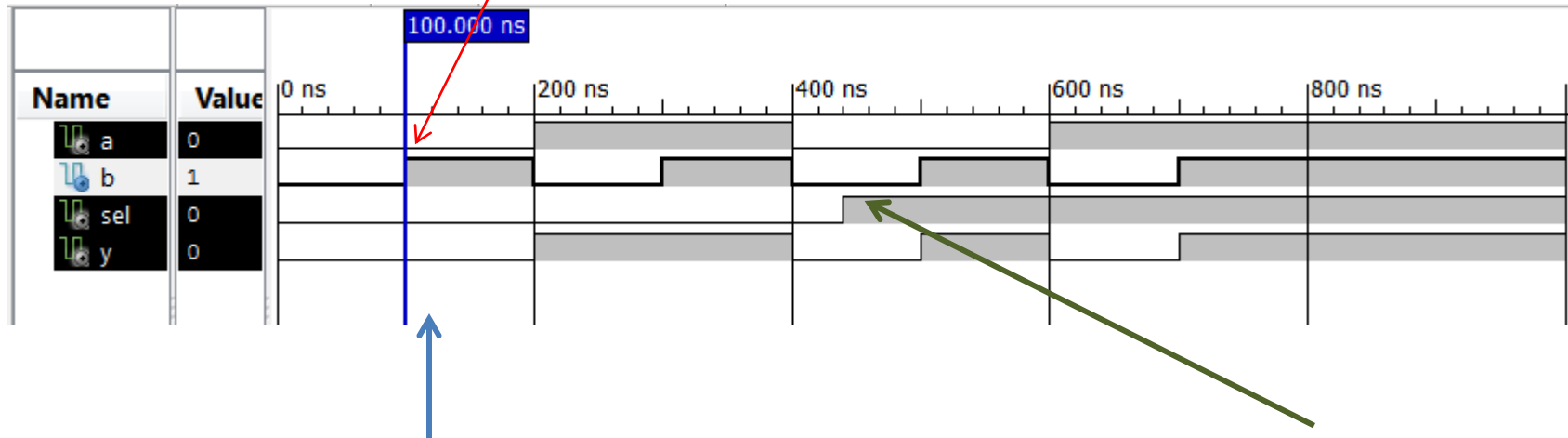
This is a Full version of ISim.  
Time resolution is 1 ps  
Simulator is doing circuit initialization process.  
Finished circuit initialization process.  
ISim> |

Sim Time: 1,000,000 ps

# TDC\_Práctica2. Simulación

## Ondas de salida (Waveform)

Cambio de la entrada B tras 100ns



Cursor, mover a la zona de la gráfica a leer

Cambio de Sel tras 440 ns

# TDC\_Práctica2. Simulación

**BEGIN**

```
-- Instantiate the Unit Under Test  
(UUT)
```

```
    uut: mux2_dataflow2 PORT MAP (  
        A => A,  
        B => B,  
        sel => sel,  
        Y => Y  
    );
```

```
-- Stimulus process
```

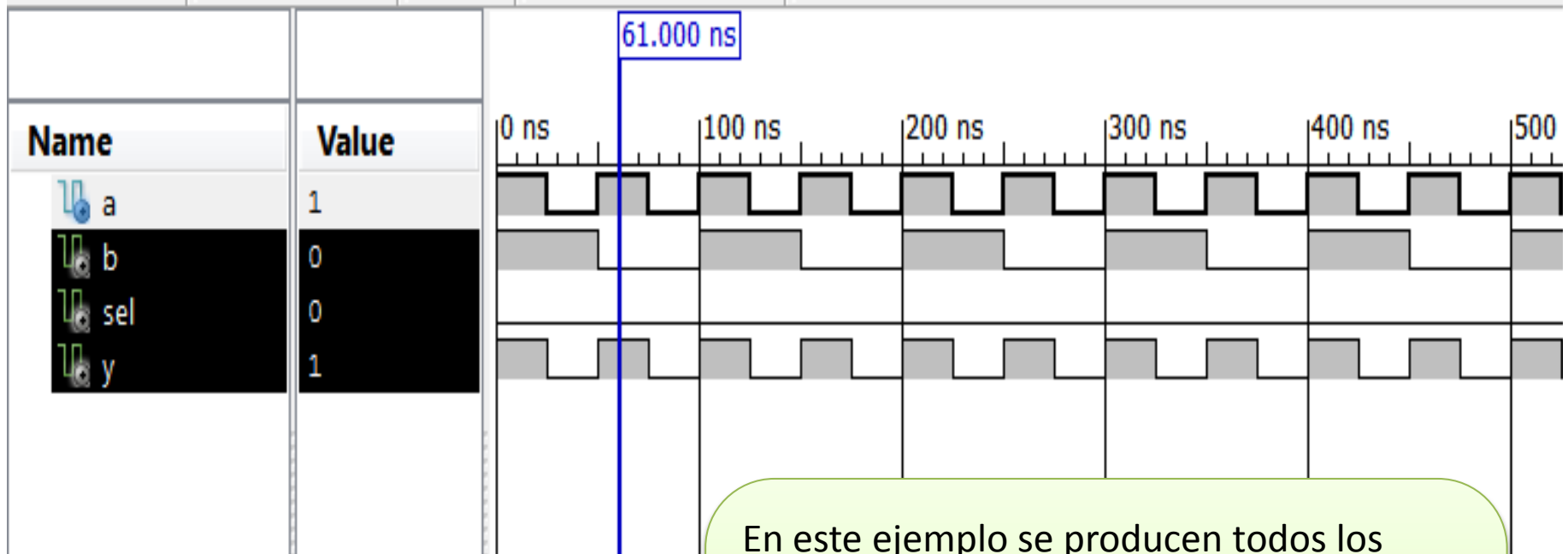
```
A_process : process  
begin  
    A <= not A;  
    wait for 25ns;  
end process;
```

```
B_process: process  
begin  
    B <= not B;  
    wait for 50ns;  
end process;
```

**END;**

Cada proceso se repite  
cada 25ns y cada 50ns

## TDC\_Práctica2. Simulación



Cada 25ns A se complementa.  
Cada 50ns B se complementa.

En este ejemplo se producen todos los vectores de test necesarios :

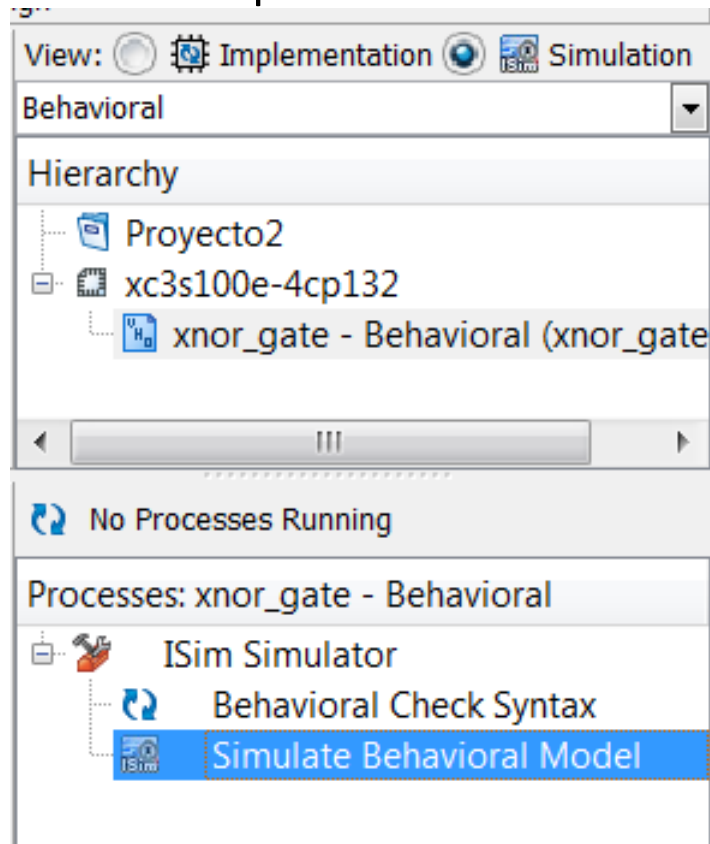
1.  $a=1, b=1 \rightarrow 25\text{ns}$
2.  $a=0, b=1 \rightarrow 50\text{ns}$
3.  $a=1, b=0 \rightarrow 75\text{ns}$
4.  $a=0, b=0 \rightarrow 100\text{ns}$



# TDC\_Práctica2. Simulación

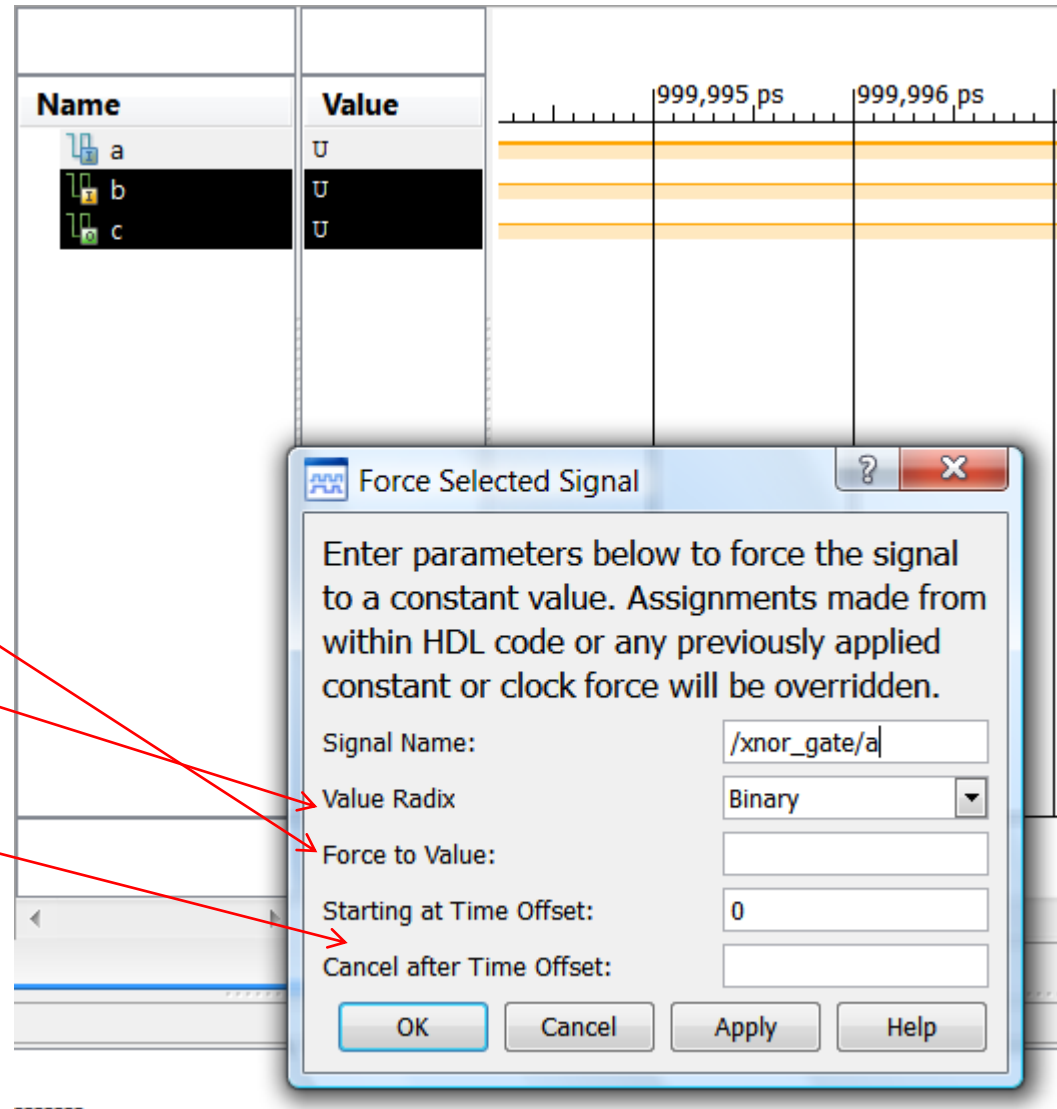
## EJEMPLO. Generar estímulos usando el GUI

1. Abrir el [Proyecto02](#), puerta XNOR.
2. Seleccionar la vista Simulation
3. Hacer doble click sobre el proceso *"Simulate Behavioral Model"*



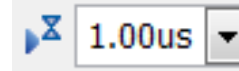
## TDC\_Práctica2. Simulación

- Seleccionar la entrada a la que se le quiere asignar valor, y mediante el menú contextual elegir la opción “Force Constant...”
- En la ventana que aparece se escribirá el **valor** en el **formato** elegido
- Si se desea es posible establecer **un tiempo de inicio y fin** para el valor asignado
- **Asignar un valor binario ‘1’ para ‘a’ y ‘0’ para ‘b’.**  
**No especificar tiempo.**

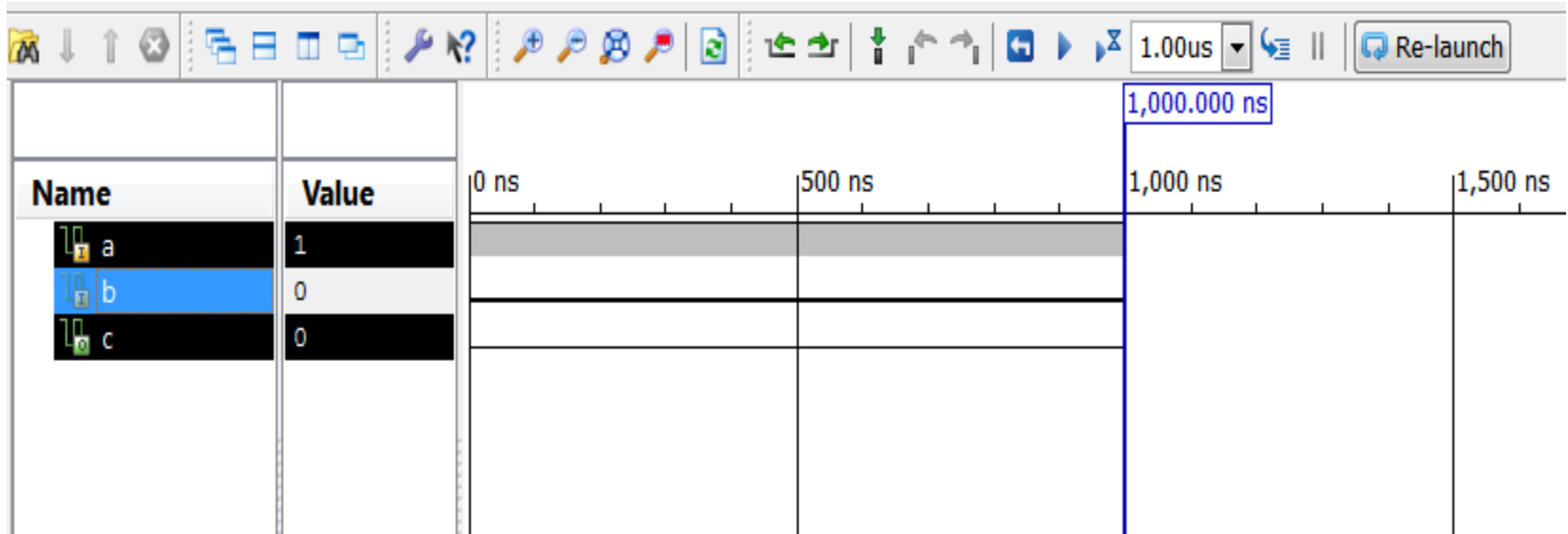


## TDC\_Práctica2. Simulación


Tras asignar valores a “a” y “b” pulsa sobre



Se ejecutará la simulación por el tiempo especificado en la lista desplegable.



Repite la operación cambiando los valores de las entradas y volviendo a simular. Se añadirá a la onda 1us más de simulación

Usa Restart  si en algún momento quieres borrar la forma de onda.

## TDC\_Práctica2. Simulación

- Es posible asignar valores a las señales usando la opción existente para definir señales de tipo reloj. Asigna una señal periódica.
- Tras seleccionar la señal escoger en el menú contextual la opción *Force Clock*.

Valor para “a”

La onda empezará con valor ‘1’,  
flanco 1º de caída

Período de la onda.  
La unidad por defecto es el picosegundo

Define Clock

Enter parameters below to force the signal to an alternating pattern (clock).  
Assignments made from within HDL code or any previously applied constant or clock force will be overridden

Signal Name: /xnor\_gate/a

Value Radix: Binary

Leading Edge Value: 1

Trailing Edge Value: 0

Starting at Time Offset: 0

Cancel after Time Offset:

Duty Cycle (%): 50

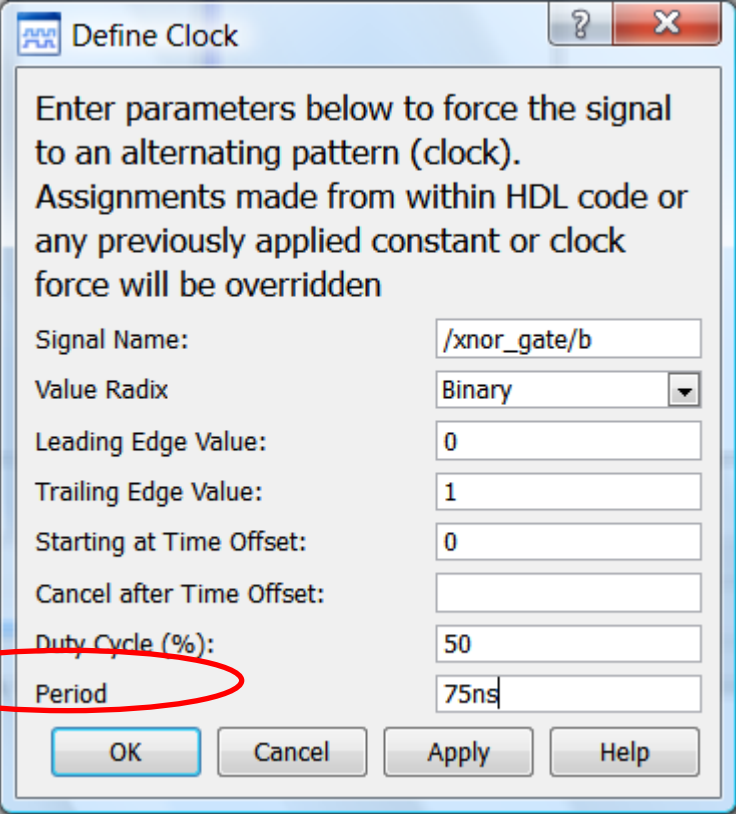
Period: 50ns

OK Cancel Apply Help

## TDC\_Práctica2. Simulación

- De este modo se puede definir una onda periódica para cada señal. Modificando sus períodos o incluyendo un tiempo de retraso se consiguen distintos patrones de entrada.

Valor para “b”



Define Clock

Enter parameters below to force the signal to an alternating pattern (clock).  
Assignments made from within HDL code or any previously applied constant or clock force will be overridden

Signal Name: /xnor\_gate/b

Value Radix: Binary

Leading Edge Value: 0

Trailing Edge Value: 1

Starting at Time Offset: 0

Cancel after Time Offset:

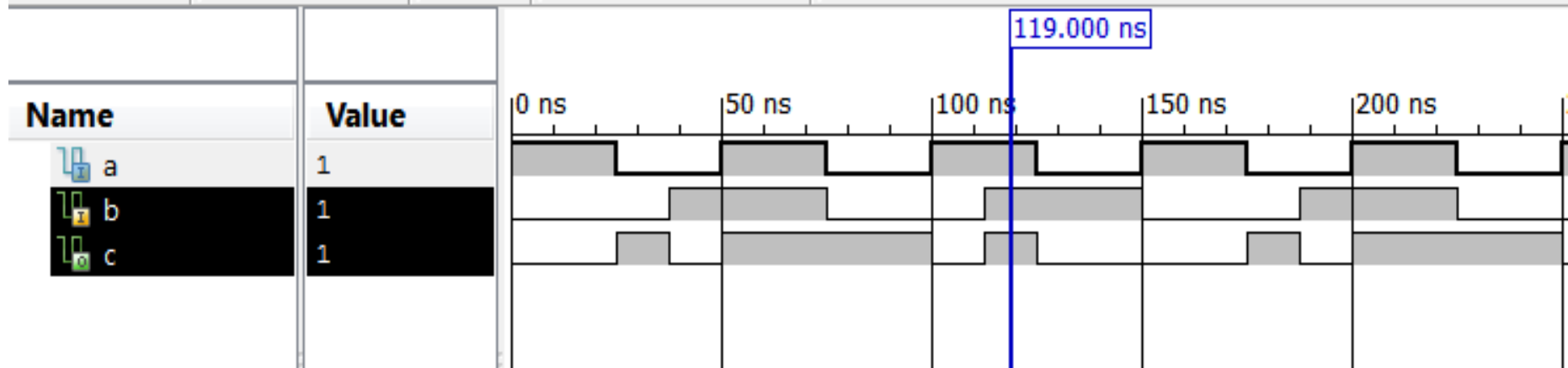
Duty Cycle (%): 50

Period: 75ns

OK Cancel Apply Help

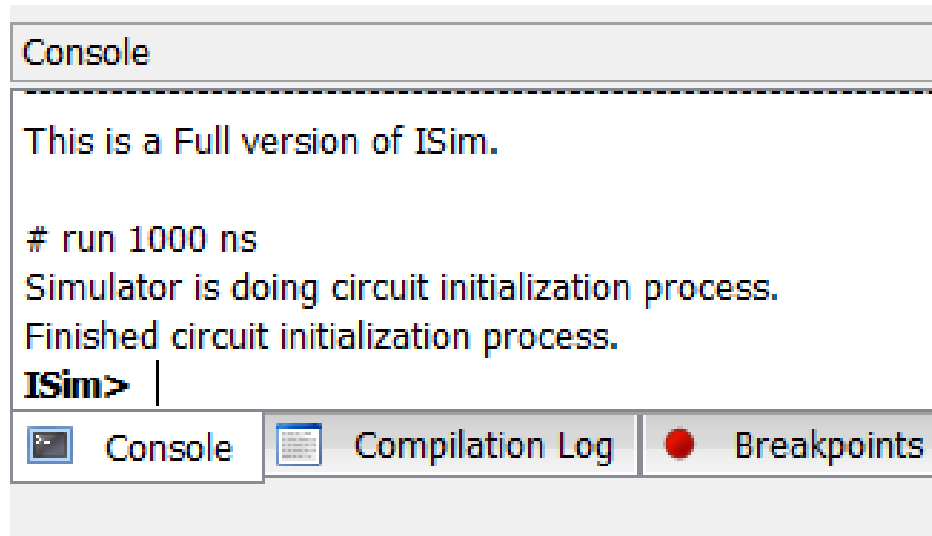
## TDC\_Práctica2. Simulación

- Posicionando el cursor en distintos puntos pueden recopilarse todas las combinaciones a testear



# TDC\_Práctica2. Simulación

EJEMPLO. Generar estímulos usando el GUI (Comandos Tcl en la consola)



- Asignar valores constantes:

ISim> `isim force add a 0`

ISim> `isim force add b 1`

- Crear ondas periódicas

`isim force add a 1 -time 0 ns -value 0 -time 25 ns -repeat 50 ns`

`isim force add b 0 -time 0 ns -value 1 -time 50 ns -repeat 100 ns`

Espacios

## TDC\_Práctica2. Simulación

1. Añade módulos de testbench en VHDL a los proyectos siguientes:

- Proyecto05 → Deco2to4
- Proyecto11 → Inc1\_3bits
- Proyecto12 → ALU\_4bits

El nombre para estos módulos se recomienda que sea el de la entidad más “\_tb”, por ejemplo [Add\\_1bit\\_tb](#).