



UNIVERSITÉ DE
SHERBROOKE

Atelier « Introduction à Vivado »

Formation au développement d'un projet de
circuit programmable pour FPGA

avec

Xilinx® Vivado® design suite

©2018-2023 tous droits réservés

Département de génie électrique et de génie informatique

Préparé par Daniel Dalle, Réjean Fontaine, Sébastien Roy, Marc-André Tétrault

Aperçu général du travail proposé

Objectif de l'atelier:

- Créer et gérer un dossier de projet Vivado,
- Créer un nouveau circuit et modifier un circuit existant* par l'ajout d'une composante,
- Réaliser des tests par simulation,
- Implémenter des circuits sur une carte ZYBO,
- Réaliser des tests de validation sur la carte Zybo pour les deux circuits



Source : Zybo Z7 Board Reference Manual, rev B, Digilent

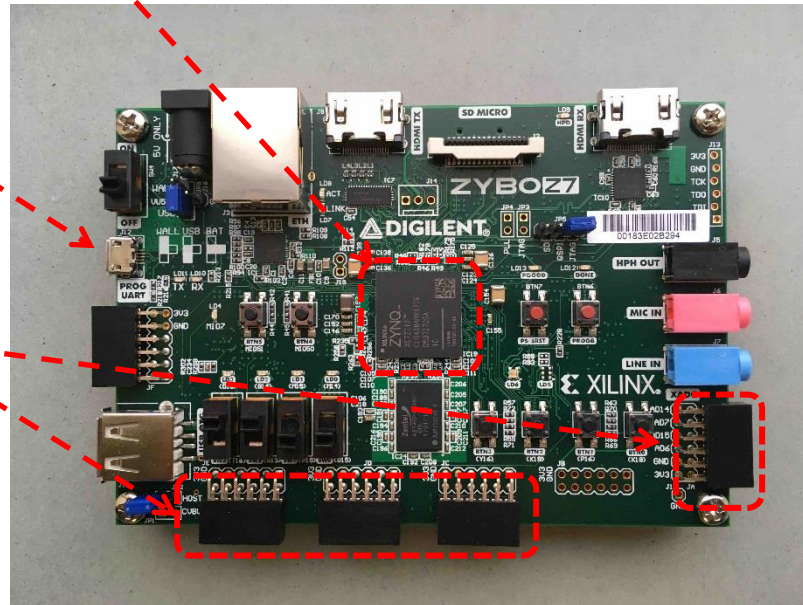
(*) L'atelier sera fait en 2 étapes: « circuit_1 » et « circuit_2 »

Voir les références à la fin de cette présentation.

Cet atelier requiert une connaissance préliminaire minimale du logiciel Vivado® et de la conception de circuits logiques élémentaires.

Ressources matérielles

- Circuit FPGA ZYNQ-7000 de [XILINX](#)[®], carte [Digilent](#)[®] [ZYBO Z7-10](#) avec circuit xc7z010clg400-1
- Communication et alimentation par un port USB
- Connecteurs Pmods



Note

- Les figures montrent Vivado 2018.2.2, mais l'interface de Vivado 2020.2 est identique.

Étapes de la partie 1 « circuit_1.vhd »

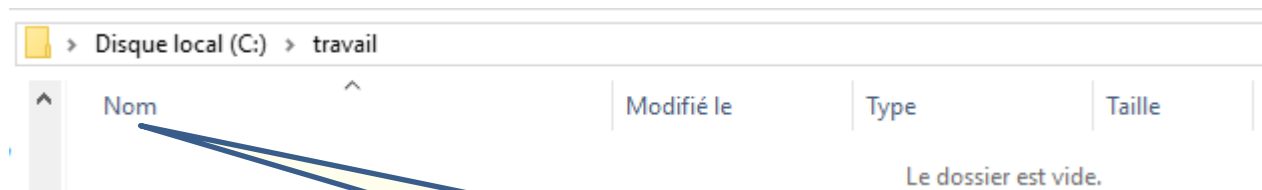
- Créer un nouveau dossier de projet Vivado
- Insérer des fichiers de contraintes
- Créer un fichier de conception (design source) : « circuit_1.vhd »
- Coder l'entité « half adder » en VHDL selon le modèle décrit
- Simuler une entité sans recours à un « test bench » en utilisant des signaux forcés sur les entrées
- Faire analyse RTL, générer un schéma
- Générer le fichier « bitstream » de configuration
- Utiliser le « hardware manager » avec la carte Zybo-Z7-10
- Programmer le circuit
- Faire un test physique sur la carte

Étapes de la partie 2 « circuit_2.vhd »

- Insérer des fichiers « circuit_2.vhd » dans la hiérarchie
- Définir l'association du fichier de contraintes requis
- Explorer et vérifier l'arborescence du circuit (« top-module »)
- Analyse RTL et schéma avant d'intégrer la composante *compteur_simple*
- Intégrer la composante *compteur_simple* dans le code VHDL
- Faire l'analyse RTL du circuit complet
- Insérer (ou créer) un code de « test bench »
- Simuler le circuit
- Générer le fichier de configuration « bitstream »
- Programmer et tester sur la carte
- Archiver le dossier de projet Vivado

Créer un nouveau dossier de projet Vivado

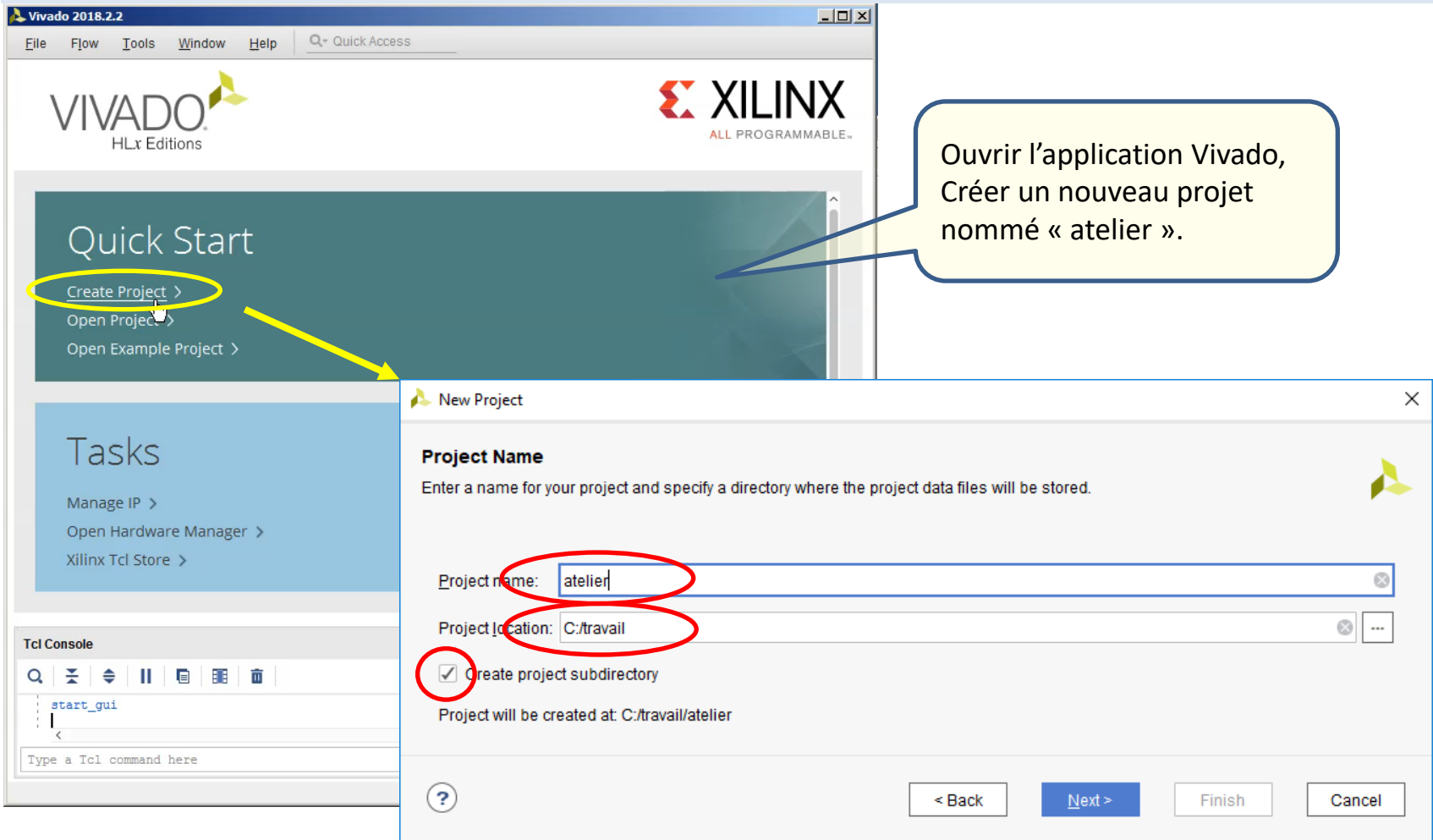
- Choisir un emplacement pour le dossier, par exemple : « **C:\travail** » choisi pour illustrer la démarche



Pour illustrer cet atelier, on a choisit une hiérarchie minimale. Choisir un emplacement quelconque dans votre espace de travail.

- Lancer l'application Vivado
- Créer un projet ...

Créer un nouveau dossier de projet Vivado



Ouvrir l'application Vivado, Créer un nouveau projet nommé « atelier ».

Quick Start

- Create Project >
- Open Project >
- Open Example Project >

Tasks

- Manage IP >
- Open Hardware Manager >
- Xilinx Tcl Store >

Tcl Console

```
start_gui
```

Type a Tcl command here

New Project

Project Name

Enter a name for your project and specify a directory where the project data files will be stored.

Project name: atelier

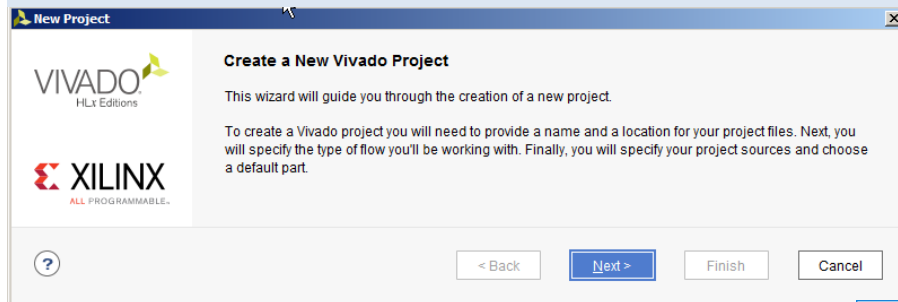
Project location: C:/travail

☒ Create project subdirectory

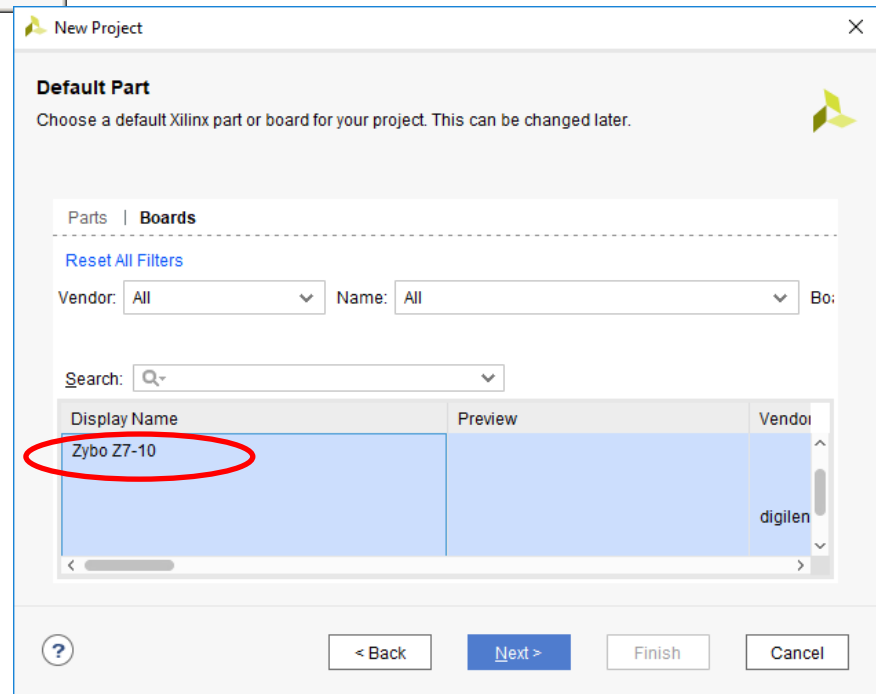
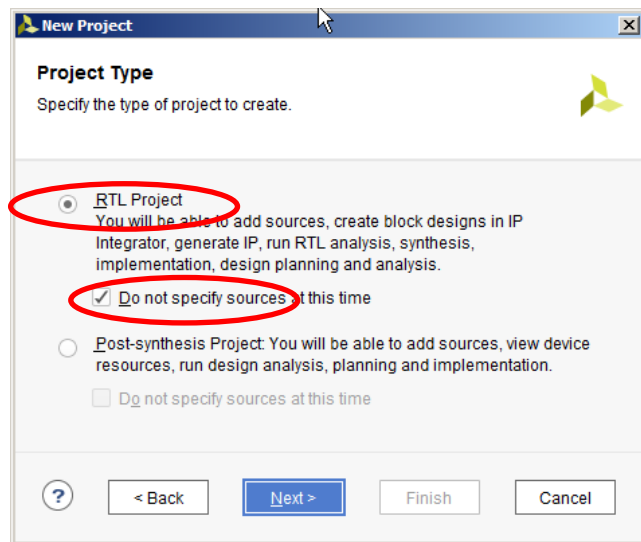
Project will be created at: C:/travail/atelier

< Back Next > Finish Cancel

Créer un nouveau dossier de projet Vivado

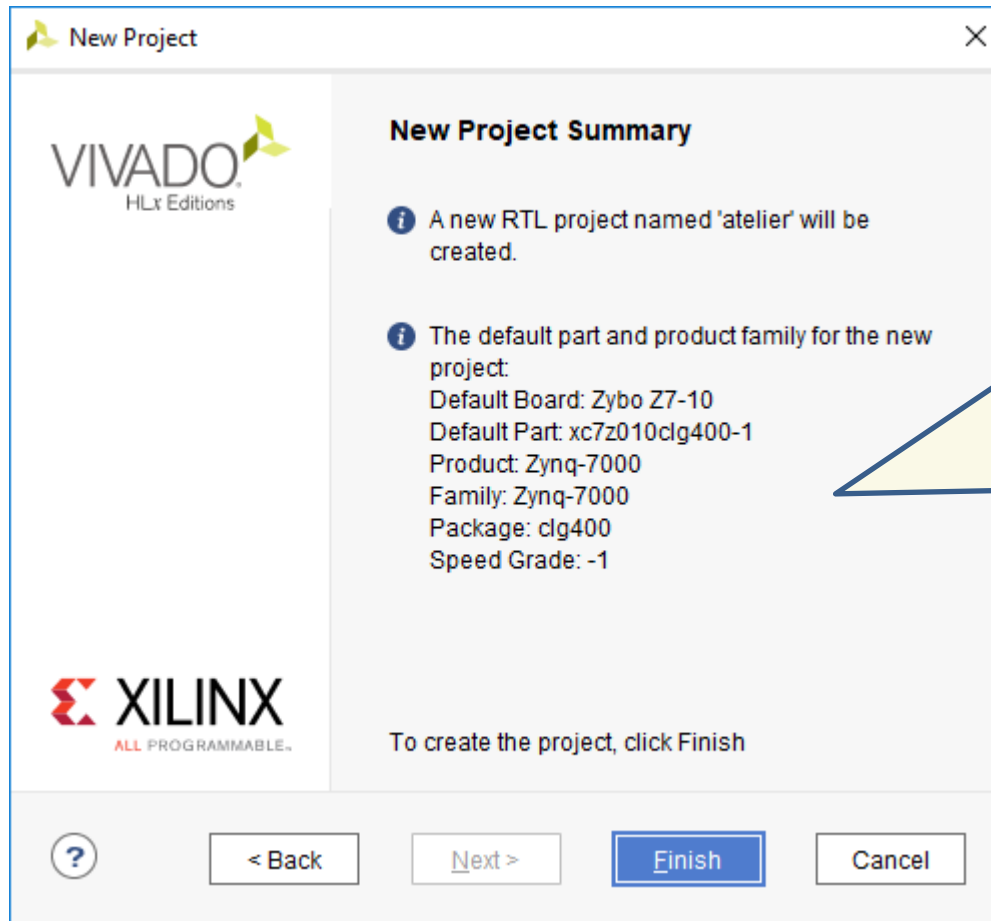


Création du dossier de projet :
spécifier le matériel, « board part »
ZYBO Z7-10.



Créer un nouveau dossier de projet Vivado

Vérification



Cette fenêtre apparaît dans la séquence d'opérations.

Vérifier la cohérence de ces informations avec la carte utilisée.

On peut toujours changer cette configuration par la suite.

Vérification du sommaire de projet en l'état actuel

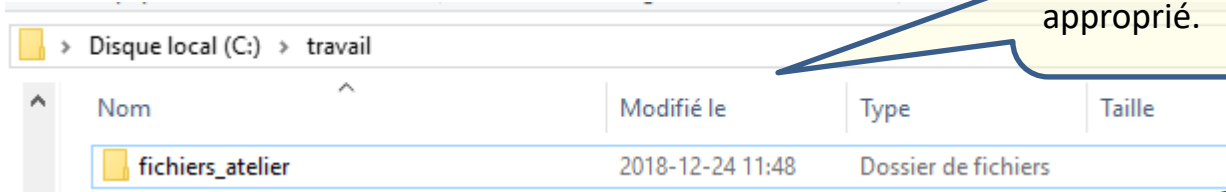
The screenshot shows the Vivado IDE interface. The top menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. The 'Layout' menu is highlighted with a red circle and an arrow pointing to it from a callout box. The callout box contains the text: "Le menu Help est très riche en informations sur l'utilisation de Vivado." The 'Sources' panel is also highlighted with a red circle and an arrow pointing to it from the same callout box. The 'Sources' panel shows a tree view of the project structure, including Design Sources, Constraints, and Simulation Sources. The 'Properties' panel is empty, showing a message: "Select an object to see properties". The 'Design Runs' panel shows a table of design runs, including a run named 'synth_1' with status 'Not started'.

Le menu Help est très riche en informations sur l'utilisation de Vivado.

Noter que dans le contexte de l'atelier on cible des opérations élémentaires de synthèse de circuits simple à partir de modules codés en VHDL, sans avoir recours aux capacités avancées de conception à haut niveau avec des modules de bibliothèques et des processeur intégrés en « bloc design ».

Insérer des fichiers de contraintes

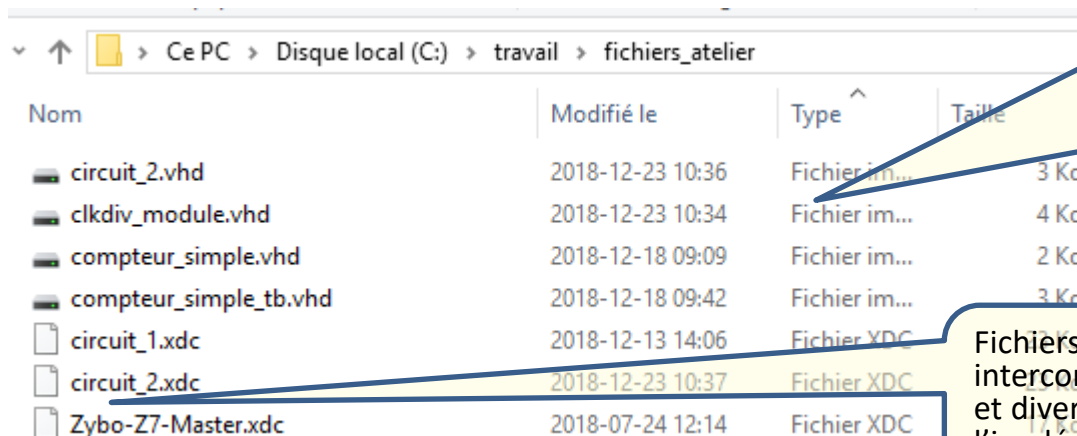
Dans le système de fichiers, choisir un emplacement approprié.



Nom	Modifié le	Type	Taille
fichiers_atelier	2018-12-24 11:48	Dossier de fichiers	

Les fichiers disponibles pour l'atelier sont dans un dossier distinct.

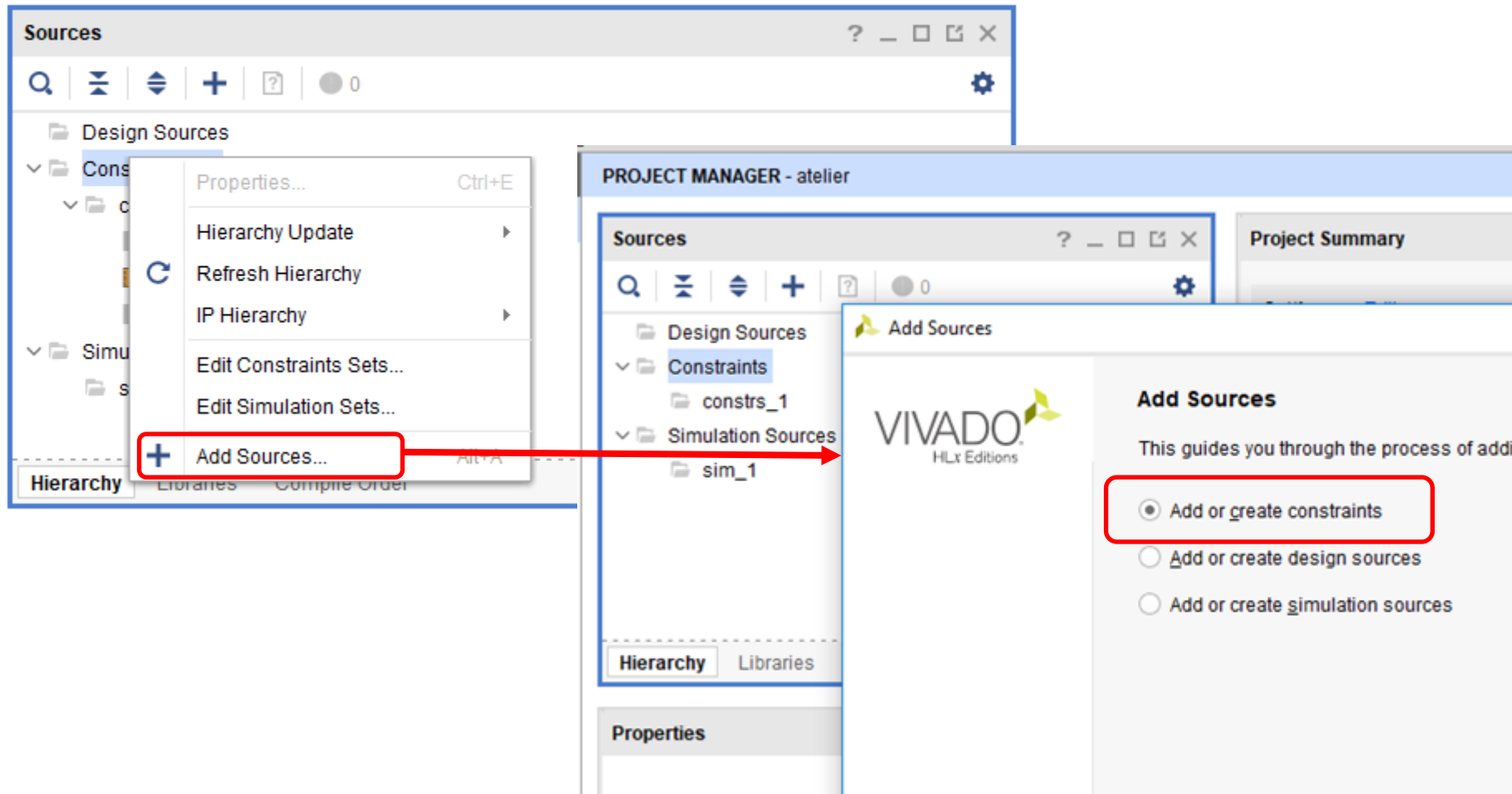
Il faudra insérer ceux qui sont nécessaires dans le dossier de projet (ce sera fait avec les commandes de l'interface Vivado).



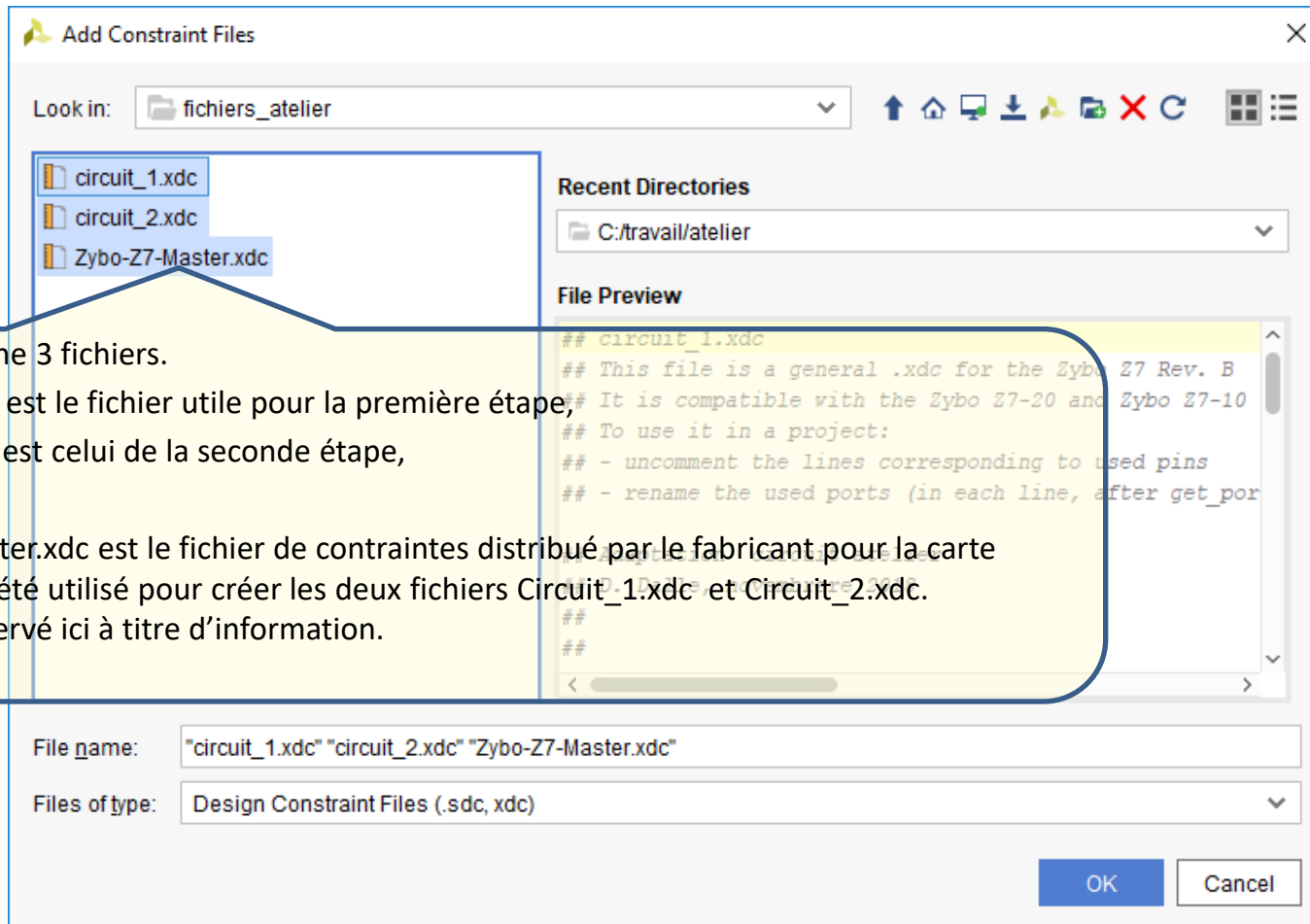
Nom	Modifié le	Type	Taille
circuit_2.vhd	2018-12-23 10:36	Fichier im...	3 Ko
clkdiv_module.vhd	2018-12-23 10:34	Fichier im...	4 Ko
compteur_simple.vhd	2018-12-18 09:09	Fichier im...	2 Ko
compteur_simple_tb.vhd	2018-12-18 09:42	Fichier im...	3 Ko
circuit_1.xdc	2018-12-13 14:06	Fichier XDC	
circuit_2.xdc	2018-12-23 10:37	Fichier XDC	
Zybo-Z7-Master.xdc	2018-07-24 12:14	Fichier XDC	

Fichiers de contraintes qui précisent les interconnexions avec l'extérieur du FPGA et diverses contraintes pour l'implémentation. Ils seront insérés aussi dans le projet Vivado.

Insérer des fichiers de contraintes



Insérer des fichiers de contraintes



Insérer des fichiers de contraintes

Add Sources

Add or Create Constraints

Specify or create constraint files for physical and timing constraint to add to your project.

Specify constraint set: **constrs_1 (active)**

Constraint File	Location
Zybo-Z7-Master.xdc	C:\travail\ fichiers_atelier
circuit_1.xdc	C:\travail\ fichiers_atelier
circuit_2.xdc	C:\travail\ fichiers_atelier

☒ **Copy constraints files into project**

Finish

Dans le contexte présent, on copie les fichiers dans le dossier Vivado. Ceci est important en regard de l'archivage ultérieur du projet pour s'assurer que les sources soient incluses. Ceci vaut aussi pour les sources en VHDL.

Insérer des fichiers de contraintes

Avec le menu contextuel, s'assurer d'obtenir la mention « enable » pour le seul fichier circuit_1.xdc.

Appliquer « Disable file » pour les deux autres fichiers circuit_2.xdc et Zybo-Z7-Master.xdc qui apparaîtront alors en grisé.

Pour circuit_1.xdc: appliquer cette commande est valide, mais non indispensable.

PROJECT MANAGER - atelier

Sources

Design Sources

Constraints (3)

constrs_1 (3)

Zybo-Z7-Master.xdc

circuit_1.xdc

Source File Properties...

Open File

Replace File...

Copy File Into Project

Copy All Files Into Project

Remove File from Project...

Enable File

Disable File

Hierarchy Update

Refresh Hierarchy

IP Hierarchy

Set as Top

Make Active

Set as Target Constraint File

Set File Type...

Set Used In...

Edit Constraints Sets...

Edit Simulation Sets...

Add Sources...

PROJECT MANAGER - atelier

Sources

Design Sources

Constraints (3)

constrs_1 (3)

Zybo-Z7-Master.xdc

Source File Properties...

Open File

Replace File...

Copy File Into Project

Copy All Files Into Project

Remove File from Project...

Enable File

Disable File

Hierarchy Update

Refresh Hierarchy

IP Hierarchy

Set as Top

Make Active

Set as Target Constraint File

Set File Type...

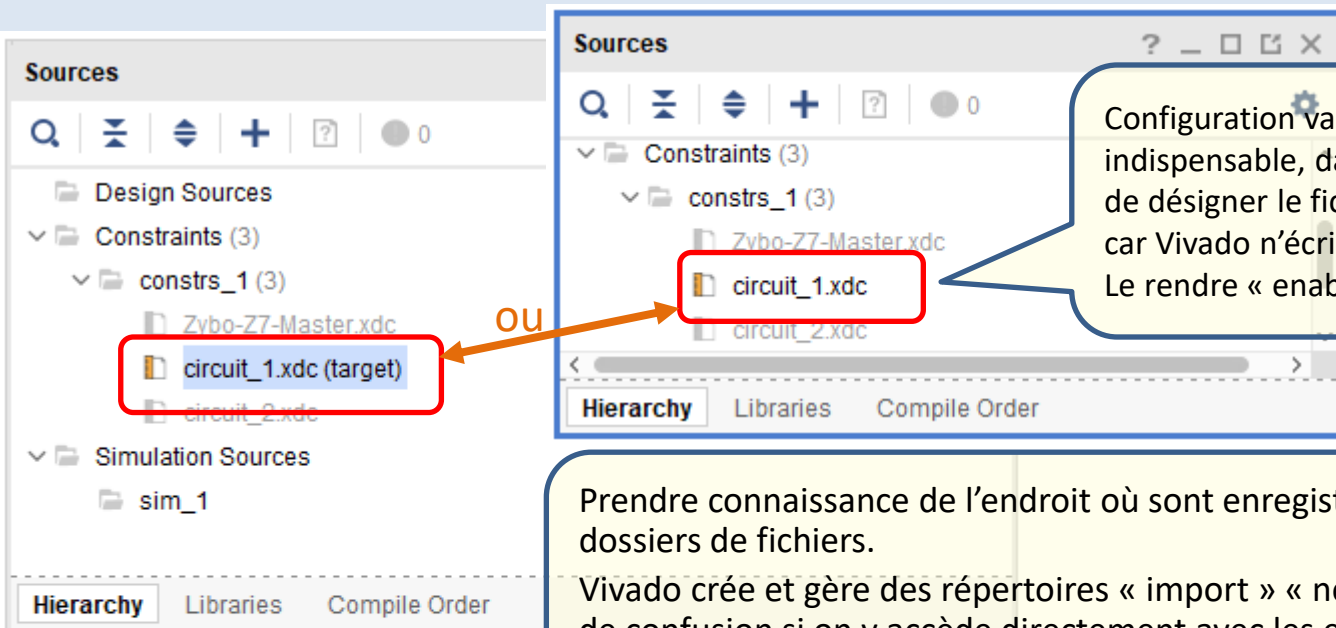
Set Used In...

Edit Constraints Sets...

Edit Simulation Sets...

Add Sources...

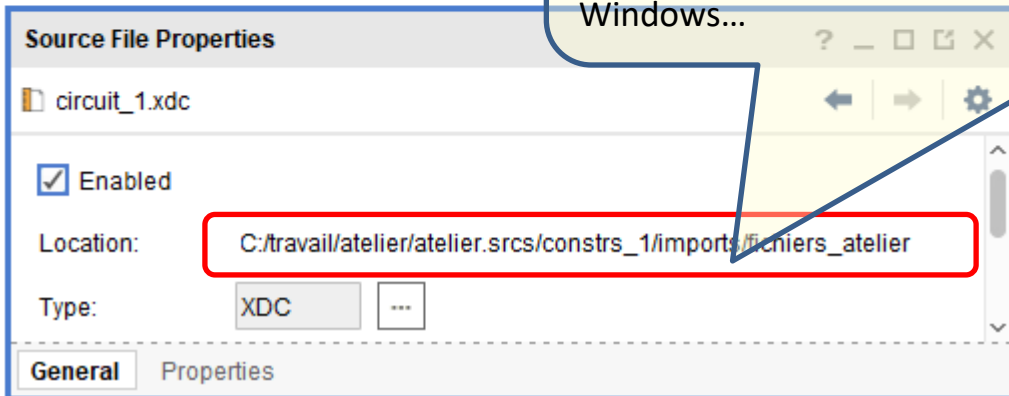
Insérer des fichiers de contraintes



The screenshot shows the 'Sources' window in Vivado. Under 'Design Sources' > 'Constraints (3)' > 'constrs_1 (3)', there are three files: 'Zybo-Z7-Master.xdc', 'circuit_1.xdc (target)', and 'circuit_2.xdc'. The 'circuit_1.xdc (target)' file is highlighted with a red box. An orange arrow labeled 'OU' points from this file to the 'circuit_1.xdc' file in the same directory structure shown in a separate window view.

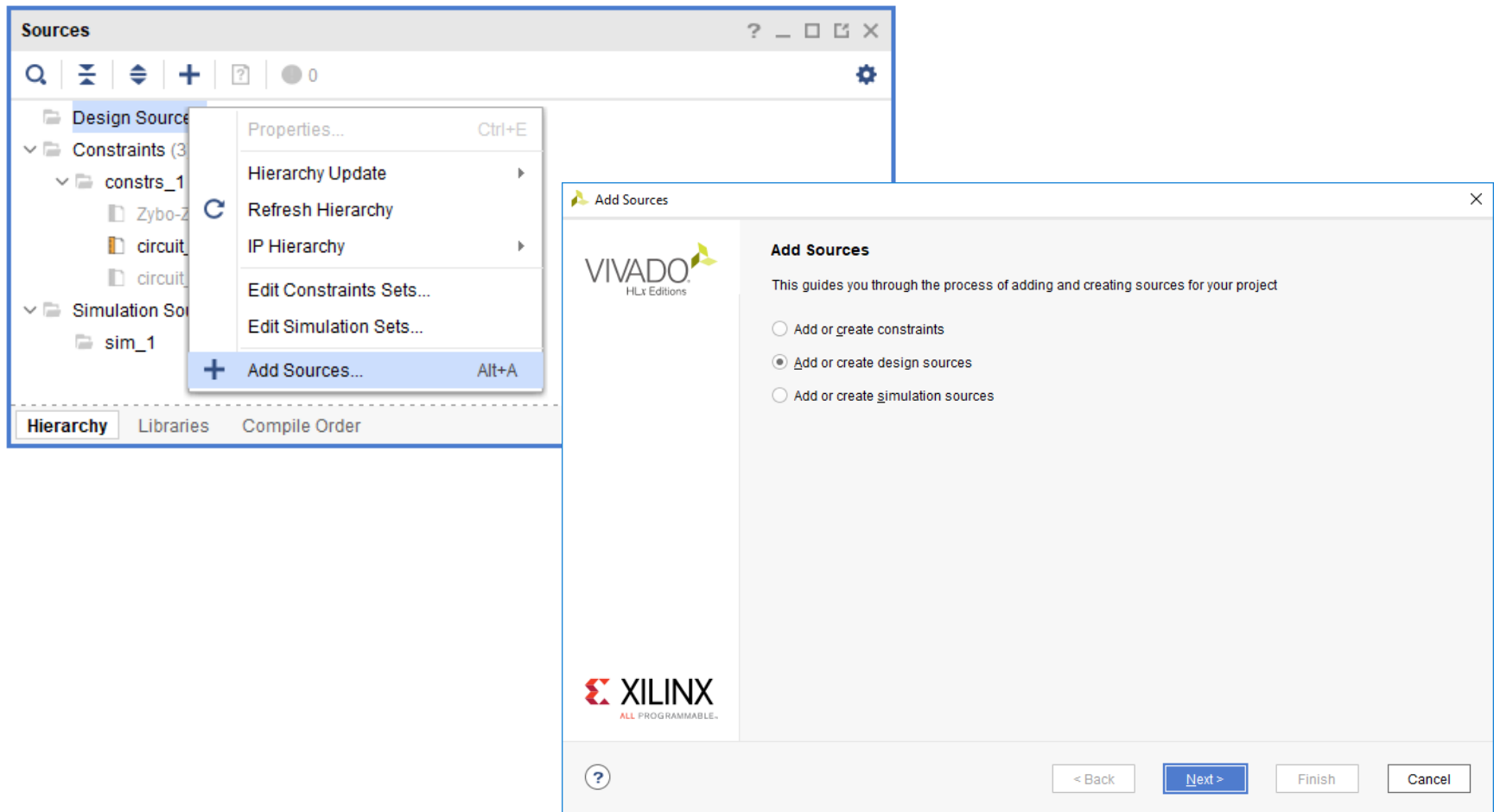
Configuration valide aussi, il n'est pas indispensable, dans le présent contexte, de désigner le fichier comme « target » car Vivado n'écrira pas dedans. Le rendre « enable » suffit.

Prendre connaissance de l'endroit où sont enregistrés ces fichiers dans les dossiers de fichiers. Vivado crée et gère des répertoires « import » « new », ce qui peut être source de confusion si on y accède directement avec les explorateurs de fichiers de Windows...

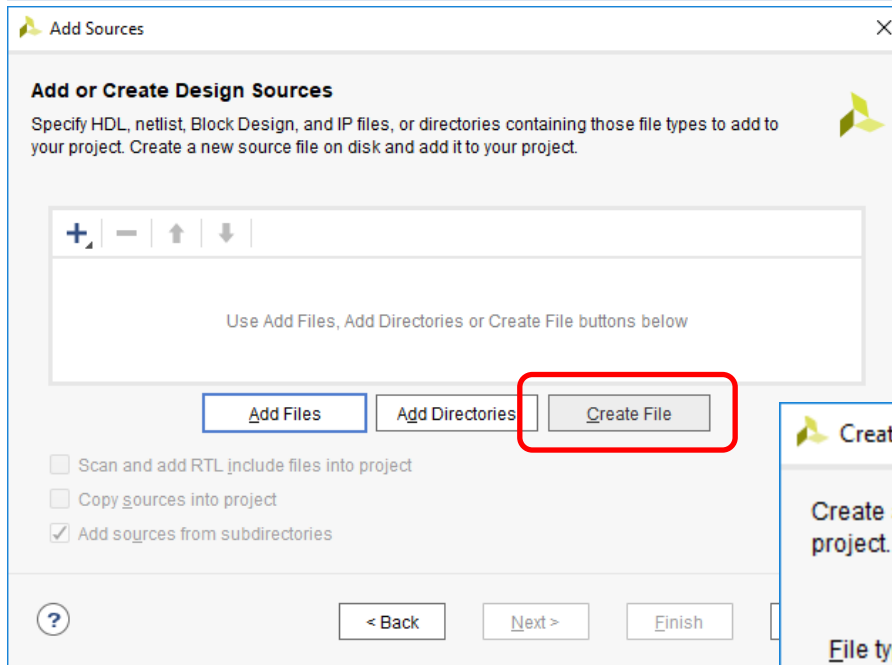


The 'Source File Properties' dialog for 'circuit_1.xdc' is shown. The 'Enabled' checkbox is checked. The 'Location' field is highlighted with a red box and contains the path: 'C:/travail/atelier/atelier.srscs/constrs_1/imports/fichiers_atelier'. The 'Type' is set to 'XDC'.

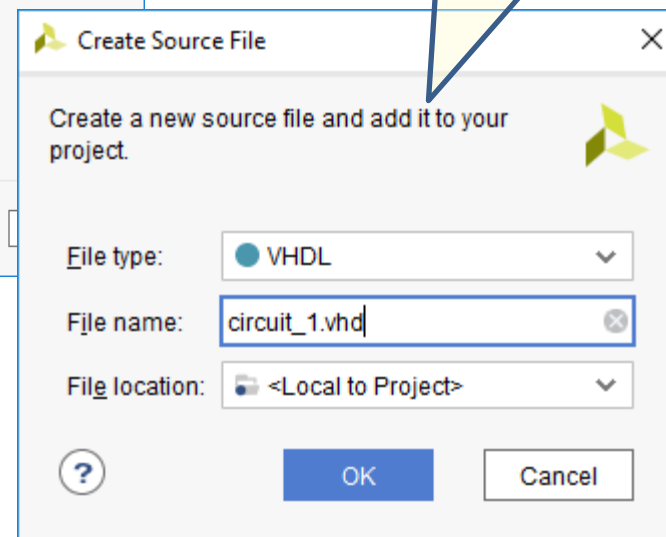
Créer un fichier de conception (design source) : *circuit_1.vhd*



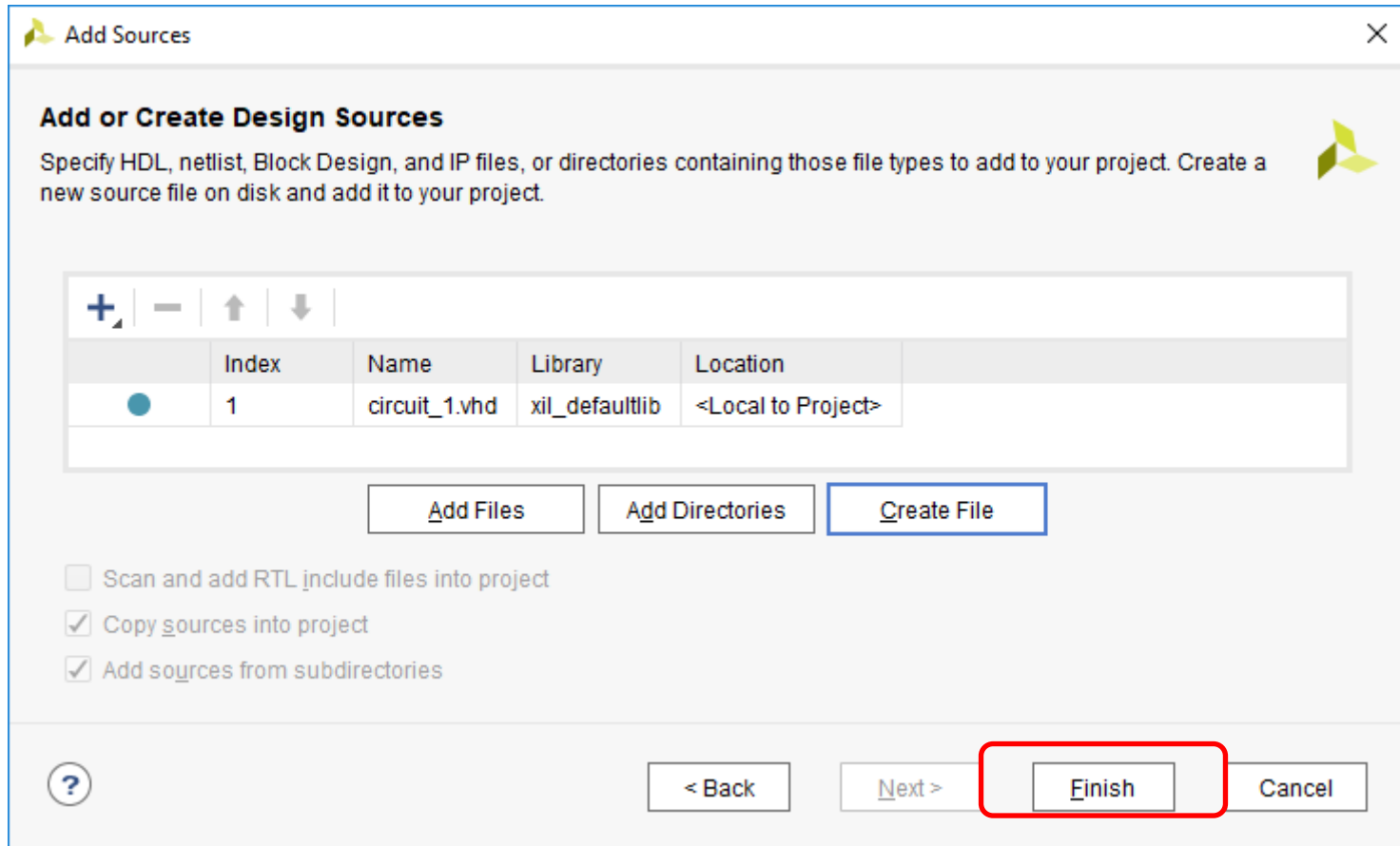
Créer un fichier de conception (design source) : *circuit_1.vhd*



On choisit un nom de fichier.
Le nom de l'entité interne peut
différer du nom de fichier.



Créer un fichier de conception (design source) : *circuit_1.vhd*



Add Sources

Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

	Index	Name	Library	Location
●	1	circuit_1.vhd	xil_defaultlib	<Local to Project>

☐ Scan and add RTL include files into project

☒ Copy sources into project

☒ Add sources from subdirectories

Créer un fichier de conception (design source)

Possibilité : on ne définit pas le nom de l'entité qui est par défaut le nom du fichier, ou ...

On peut préciser le nom de l'entité et déclarer les signaux d'interface.

Si on ne le fait pas à ce moment, on peut toujours le faire (ou modifier) en codant le VHDL du module.

Define Module

Define a module and specify I/O Ports to add to your source file.
For each port specified:
MSB and LSB values will be ignored unless its Bus column is checked.
Ports with blank names will not be written.

Module Definition

Entity name:

Architecture name:

I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
	in	<input type="checkbox"/>	0	0

?

OK

Define Module

Define a module and specify I/O Ports to add to your source file.
For each port specified:
MSB and LSB values will be ignored unless its Bus column is checked.
Ports with blank names will not be written.

Module Definition

Entity name:

Architecture name:

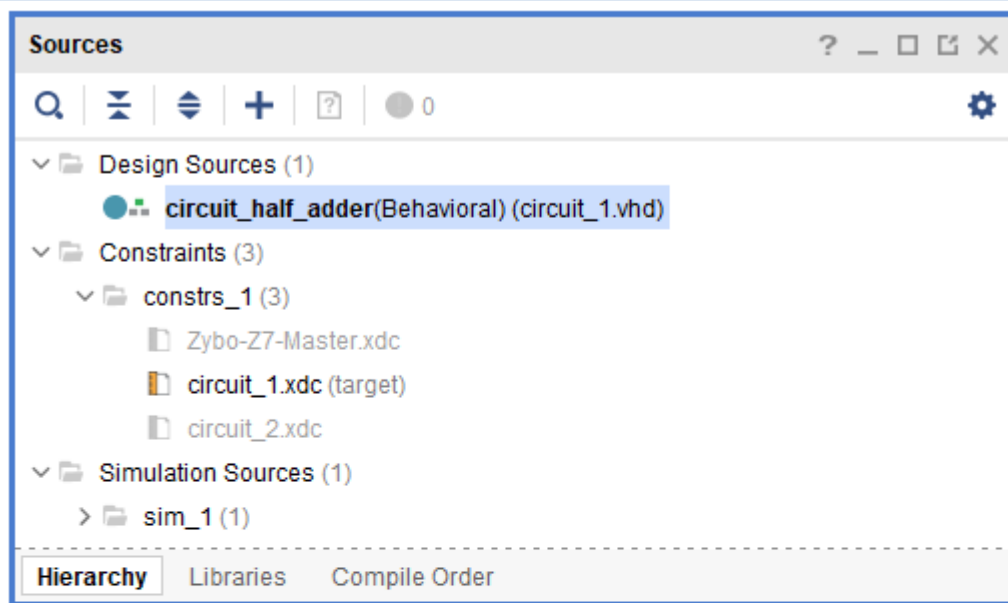
I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
a	in	<input type="checkbox"/>	0	0
b	in	<input type="checkbox"/>	0	0
s	out	<input type="checkbox"/>	0	0
c	out	<input type="checkbox"/>	0	0

?

OK Cancel

Créer un fichier de conception (design source)



Créer un fichier de conception (design source)

The screenshot displays the Xilinx ISE Project Manager interface. The 'Sources' window on the left shows a project named 'atelier'. Under 'Design Sources (1)', the file 'circuit_half_adder(Behavioral) (circuit_1.vhd)' is listed. Below it, 'Constraints (3)' includes 'constrs_1 (3)' with files 'Zybo-Z7-Master.xdc', 'circuit_1.xdc (target)', and 'circuit_2.xdc'. 'Simulation Sources (1)' includes 'sim_1 (1)'. The 'Source File Properties' window at the bottom left shows 'circuit_1.vhd' is 'Enabled', located at 'C:/travail/atelier/atelier.srsrcs/sources_1/new', and its type is 'VHDL'. The main editor window shows the 'circuit_1.vhd' file. The code includes comments about library declarations and the start of an entity definition. Two callouts provide context: one points to the entity declaration and the other points to the architecture line.

Sources

- Design Sources (1)
 - circuit_half_adder(Behavioral) (circuit_1.vhd)
- Constraints (3)
 - constrs_1 (3)
 - Zybo-Z7-Master.xdc
 - circuit_1.xdc (target)
 - circuit_2.xdc
- Simulation Sources (1)
 - sim_1 (1)

Source File Properties

circuit_1.vhd

☒ Enabled

Location: C:/travail/atelier/atelier.srsrcs/sources_1/new

Type: VHDL

Project Summary | **circuit_1.vhd**

C:/travail/atelier/atelier.srsrcs/sources_1/new/circuit_1.vhd

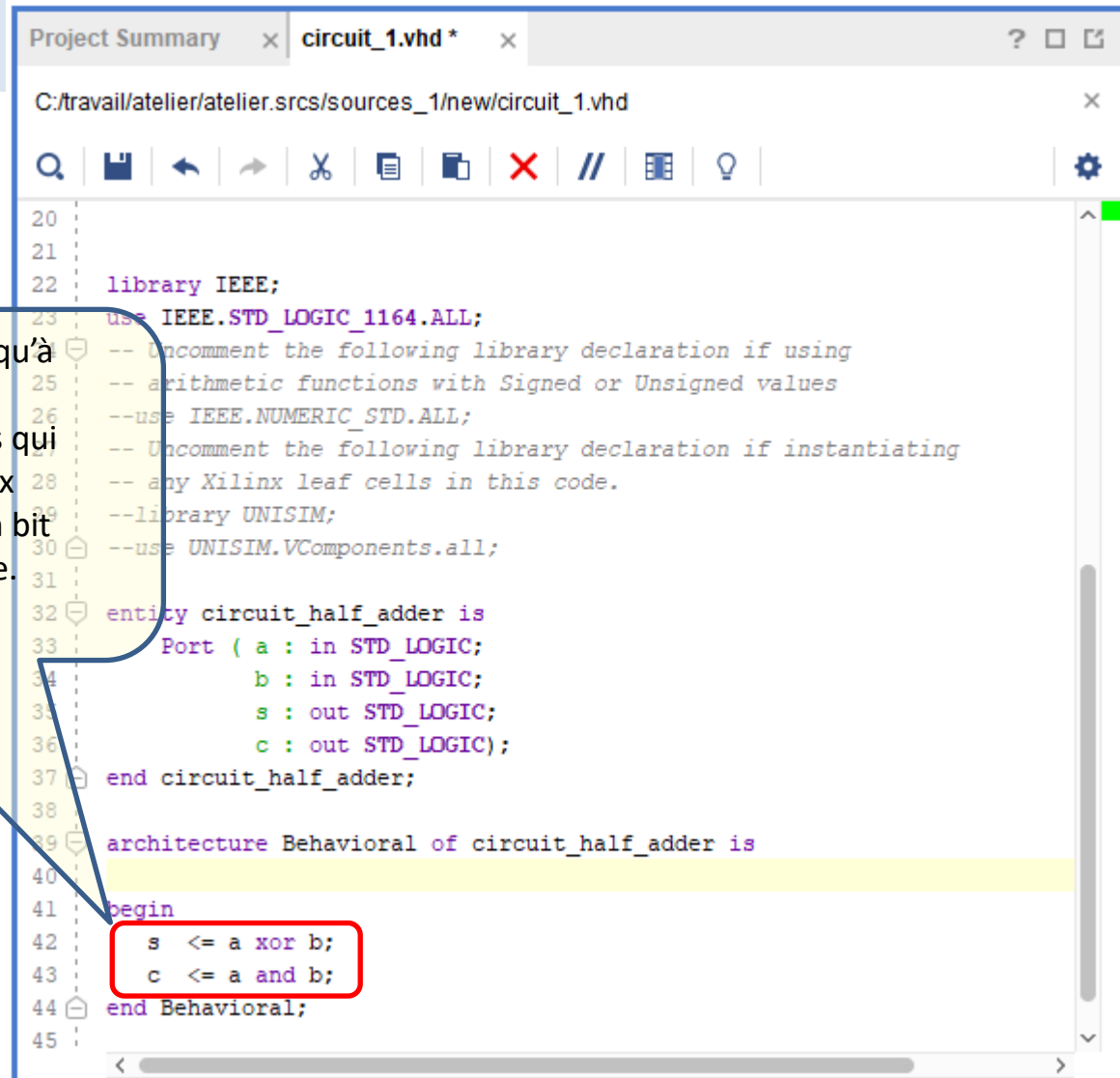
```
25 -- Uncomment the following library declaration if using
26 -- arithmetic functions with Signed or Unsigned values
27 --use IEEE.NUMERIC_STD.ALL;
28
29 -- Uncomment the following library declaration if instantiating
30 -- any Xilinx leaf cells in this code.
31 --library UNISIM;
32 --use UNISIM.VComponents.all;
33
34 entity circuit_half_adder is
35     Port ( a : in STD_LOGIC;
36           b : in STD_LOGIC;
37           s : out STD_LOGIC;
38           c : out STD_LOGIC);
39 end circuit_half_adder;
40
41 architecture Behavioral of circuit_half_adder is
42
43 begin
44
45
46 end Behavioral;
```

La définition de l'entité VHDL est alors déjà construite.

Il manque le code de l'architecture.

Coder l'entité « half adder » en VHDL selon le modèle décrit

Il ne reste, pour cet exemple simple, qu'à coder la définition de l'architecture comportementale par ces deux lignes qui affectent des fonctions booléennes aux signaux de sorties, soit l'addition à un bit simple avec production d'une retenue.



```
20
21
22 library IEEE;
23 use IEEE.STD_LOGIC_1164.ALL;
24 -- Uncomment the following library declaration if using
25 -- arithmetic functions with Signed or Unsigned values
26 --use IEEE.NUMERIC_STD.ALL;
27 -- Uncomment the following library declaration if instantiating
28 -- any Xilinx leaf cells in this code.
29 --library UNISIM;
30 --use UNISIM.VComponents.all;
31
32 entity circuit_half_adder is
33 Port ( a : in STD_LOGIC;
34       b : in STD_LOGIC;
35       s : out STD_LOGIC;
36       c : out STD_LOGIC);
37 end circuit_half_adder;
38
39 architecture Behavioral of circuit_half_adder is
40
41 begin
42     s <= a xor b;
43     c <= a and b;
44 end Behavioral;
45
```


Simuler une entité sans recours à un « test bench »

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
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design
- PROGRAM AND DEBUG
 - Generate Bitstream
 - Open Hardware Manager

Sources

- Design Sources (1)
 - circuit_half_adder(Behavioral) (circuit_1.vhd)
- Constraints (3)
 - constrs_1 (3)
 - Zybo-Z7-Master.xdc
 - circuit_1.xdc (target)
 - circuit_2.xdc
- Simulation Sources (1)
 - sim_1 (1)
 - circuit_half_adder(Behavioral) (circuit_1.vhd)

Hierarchy Libraries Compile Order

Run Simulation

Run Behavioral Simulation

Run Post-Synthesis Functional Simulation

Run Post-Synthesis Timing Simulation

Run Post-Implementation Functional Simulation

Run Post-Implementation Timing Simulation

NOTE: il est possible de faire des simulations avancées soit après la synthèse ou après l'implémentation. Ces simulations sont plus réaliste mais prennent aussi beaucoup plus de temps. Ces simulations pourront vous être utiles à l'APP de logique séquentielle

La hiérarchie est maintenant complète pour circuit_1.vhd.

On peut débiter la simulation globale de circuit_1. Pour ce circuit combinatoire simple, on peut le faire sans coder un « test bench ».
(ce sera différent pour circuit_2 de l'étape 2 de l'atelier.

Simuler une entité sans recours à un « test bench »

La simulation est lancée.

The screenshot displays the Vivado 2018.2.2 interface during a behavioral simulation. The main window shows the 'Scope' tab with a table of signals: 'circuit_half_adder' (Design U..., Block Type VHDL En...). The 'Objects' tab shows a table of signals: 'a', 'b', 's', 'c' (Value U, Data Type Logic). The 'Tcl Console' shows the simulation results, including a warning about no top-level signals found and a message indicating the simulation completed successfully.

Scope

Name	Design U...	Block Type
circuit_half_adder	circuit_h...	VHDL En...

Objects

Name	Value	Data Type
a	U	Logic
b	U	Logic
s	U	Logic
c	U	Logic

Tcl Console

```
# if { [length [get_objects]] > 0 } {  
#   add_wave /  
#   set_property needs_save false [current_wave_config]  
# } else {  
#   send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave window. If you want to open a wave window go to 'File' -> 'Add Waveform to Simulation' -> 'Add Signals to Waveform'."  
# }  
# }  
# run 1000ns  
INFO: [USF-XSim-96] XSim completed. Design snapshot 'circuit_half_adder_behav' loaded.  
INFO: [USF-XSim-97] XSim simulation ran for 1000ns  
launch_simulation: Time (s): cpu = 00:00:04 ; elapsed = 00:00:23 . Memory (MB): peak = 801.293 ; gain = 8.105
```

Simuler une entité sans recours à un « test bench »

En l'absence de « test bench », il faut trouver un moyen de créer des signaux de simulation aux entrées: deux signaux périodiques (de type horloge) avec des périodes distinctes permettent de générer les combinaisons binaires sur les entrées a et b.

Force Clock: /circuit_half_adder/a

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: /circuit_half_adder/a

Value radix: Hexadecimal

Leading edge value: 1

Trailing edge value: 0

Starting after time offset: 0ns

Cancel after time offset:

Duty cycle (%): 50

Period: 100ns

OK Cancel

Simuler une entité sans recours à un « test bench »

The screenshot shows a circuit simulation software interface. On the left, a table lists signals 'a', 'b', 's', and 'c'. Signal 'b' is selected, and a context menu is open with options like 'Go To Source Code', 'Show in Object Window', 'Report Drivers', 'Force Constant...', 'Force Clock...', and 'Remove Force'. The 'Force Clock...' option is highlighted. A yellow callout box points to this option with the text: "On spécifie l'entrée b avec une fréquence différente." To the right, the 'Force Clock: /circuit_half_adder/b' dialog box is open. It contains fields for 'Signal name', 'Value radix', 'Leading edge value', 'Trailing edge value', 'Starting after time offset', 'Cancel after time offset', 'Duty cycle (%)', and 'Period'. The 'Period' field is set to '200ns' and is highlighted with a red box. The 'Leading edge value' is set to '1' and the 'Trailing edge value' is set to '0', both also highlighted with red boxes. The dialog box has 'OK' and 'Cancel' buttons at the bottom.

On spécifie l'entrée b avec une fréquence différente.

Force Clock: /circuit_half_adder/b

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: /circuit_half_adder/b

Value radix: Hexadecimal

Leading edge value: 1

Trailing edge value: 0

Starting after time offset: 0ns

Cancel after time offset:

Duty cycle (%): 50

Period: 200ns

OK Cancel

Simuler une entité sans recours à un « test bench »

The screenshot shows the simulation interface with the following components:

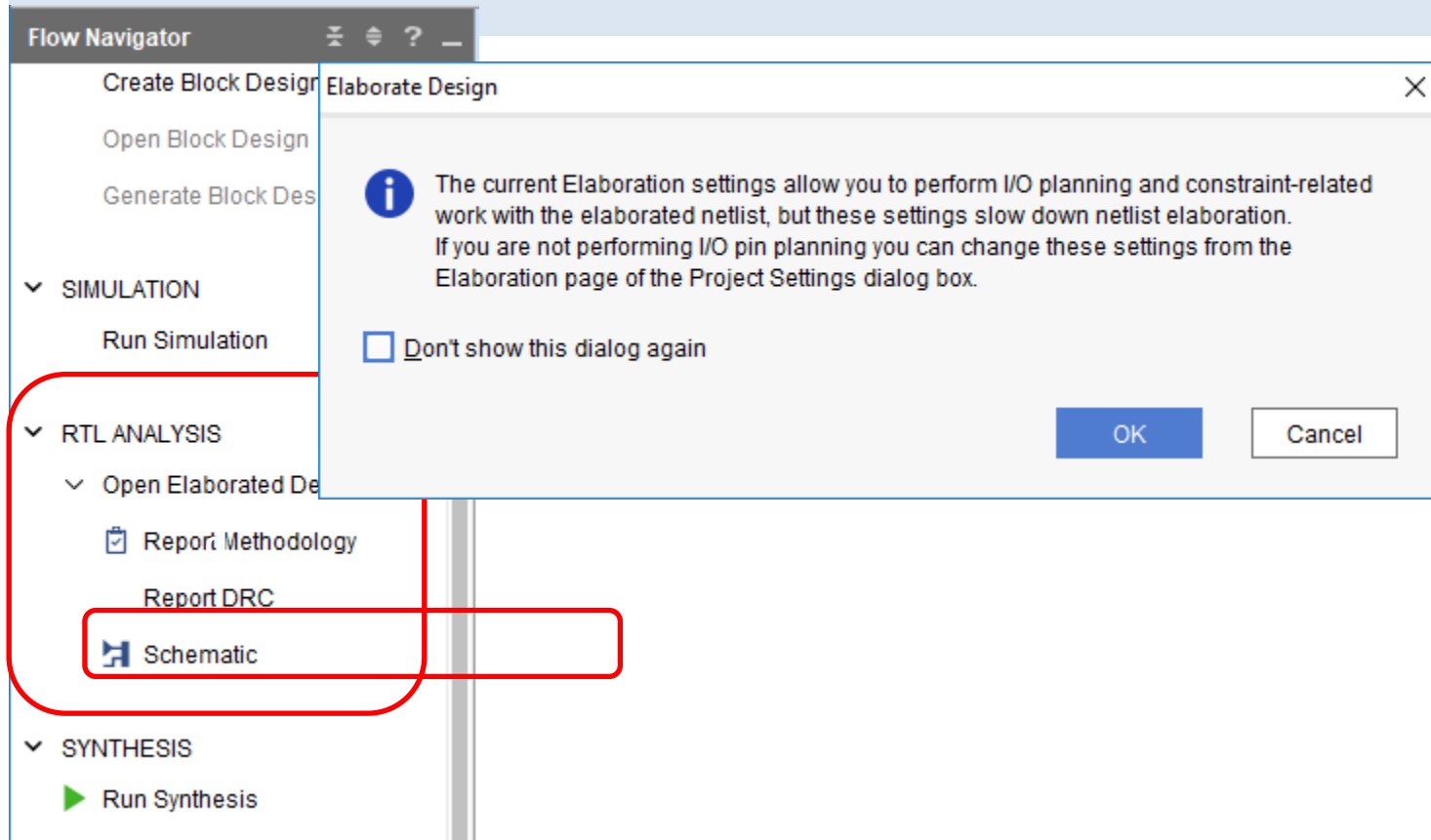
- Run menu:** Located at the top left, with a yellow arrow pointing to it.
- Simulation controls:** A toolbar with buttons for simulation (play, pause, stop, etc.) and a text box showing "2000 ns". A yellow arrow points to the simulation controls.
- Signal list:** A table on the left showing the current values of signals a, b, s, and c.
- Waveform:** A plot showing the digital signals over time.

Callouts provide additional information:

- Le menu de simulation permet de définir la durée de simulation et de redémarrer ou arrêter une simulation.** (Points to the Run menu)
- Si on modifie un code VHDL, ce bouton exécute une nouvelle compilation pour refaire la simulation sans quitter cet interface.** (Points to the simulation controls)
- Contrôle de la mise en page du chronogramme.** (Points to the waveform)

En sélectionnant un signal, par exemple a, ces boutons permettent de glisser aux transitions du signal (dans les deux directions). C'est une commande très utile.

Faire analyse RTL, générer un schéma



Faire analyse RTL, générer un schéma

The screenshot displays the Vivado 2018.2.2 IDE interface. The top menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. The main workspace is divided into several panes:

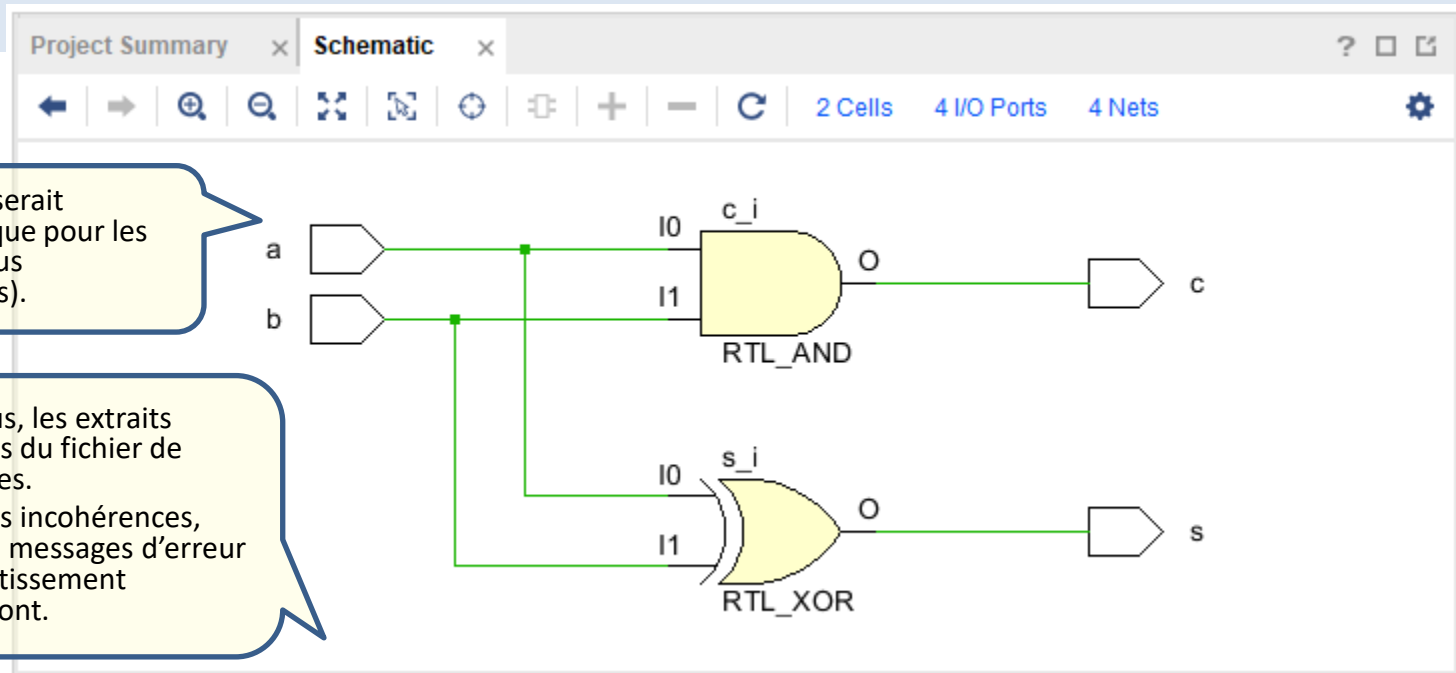
- Flow Navigator (Left):** Contains project management and analysis options. The **RTL ANALYSIS** section is highlighted with a red box, showing options like **Open Elaborated Design**, **Report Methodology**, **Report DRC**, **Report Noise**, and **Schematic** (which is selected).
- ELABORATED DESIGN - xc7z010clg400-1 (active):** This pane shows the **Netlist** view of the circuit. It lists sources like **circuit_half_adder**, nets (**a**, **b**, **c**, **s**), and leaf cells (**c_i (RTL_AND)** and **s_i (RTL_XOR)**). A callout box points to this netlist, stating: "Information utile pour compléter l'analyse et en cas de processus de déverminage quand cela est nécessaire." Below the netlist is a **Properties** pane.
- Schematic (Right):** Displays the visual representation of the circuit. It shows two logic blocks: **RTL_AND** (labeled **c_i**) and **RTL_XOR** (labeled **s_i**). The **RTL_AND** block has inputs **a** and **b** and output **c**. The **RTL_XOR** block has inputs **a** and **b** and output **s**.
- Design Runs (Bottom):** A table showing the status of various design runs. The table has columns: Name, Constraints, Status, WNS, TNS, WHS, THS, TPWS, Total Power, Failed Routes, LUT, FF, BRAMs, and URA. The first two rows show runs for **synth_1** and **impl_1**, both with a status of **Not started**.

A callout box at the bottom right explains the purpose of the RTL analysis: "Cette analyse n'est pas requise pour programmer le FPGA, mais elle est très utile pour analyser une conception et éventuellement mettre en évidence des caractéristiques et des problèmes."

Faire analyse RTL, générer un schéma

Schéma (serait hiérarchique pour les entités plus complexes).

Ci-dessous, les extraits pertinents du fichier de contraintes.
S'il y a des incohérences, alors des messages d'erreur ou d'avertissement apparaîtront.



```
25 ##Switches (circuit_atelier circuit_1)
26 set_property -dict { PACKAGE_PIN G15 IOSTANDARD LVCMOS33 } [get_ports { a }]; #IO_L19N_T3_VREF_35 Sch=sw[0]
27 set_property -dict { PACKAGE_PIN P15 IOSTANDARD LVCMOS33 } [get_ports { b }]; #IO_L24P_T3_34 Sch=sw[1]
```

```
56 ##LEDs (circuit_atelier circuit_1)
57 set_property -dict { PACKAGE_PIN M14 IOSTANDARD LVCMOS33 } [get_ports { s }]; #IO_L23P_T3_35 Sch=led[0]
58 set_property -dict { PACKAGE_PIN M15 IOSTANDARD LVCMOS33 } [get_ports { c }]; #IO_L23N_T3_35 Sch=led[1]
```


Générer le fichier « bitstream » de configuration

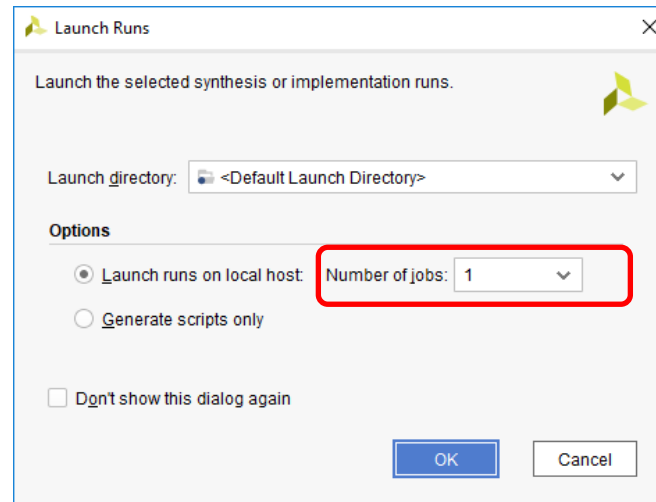
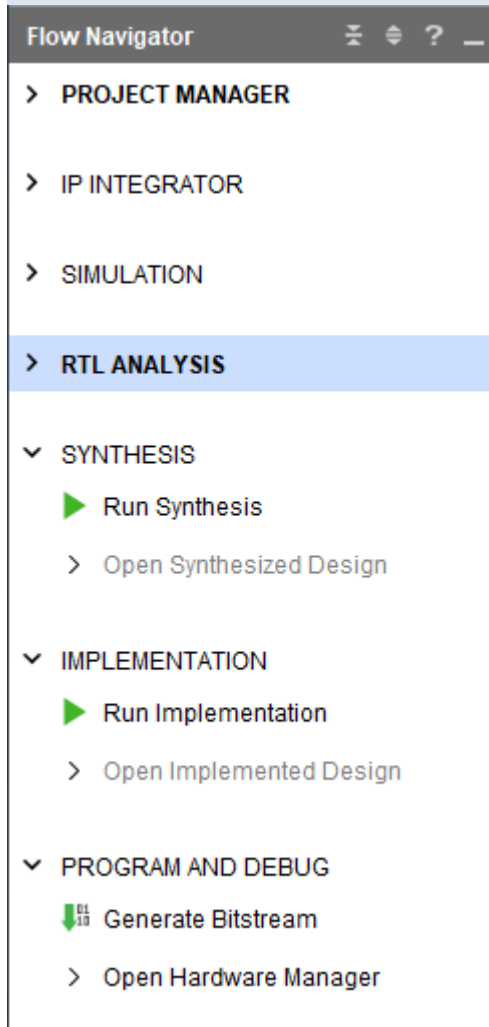
The screenshot shows the 'Flow Navigator' on the left with the following structure:

- PROJECT MANAGER
- IP INTEGRATOR
- SIMULATION
- RTL ANALYSIS
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design
- PROGRAM AND DEBUG
 - Generate Bitstream** (highlighted with a red box)
 - Open Hardware Manager

Three dialog boxes are displayed:

- No Implementation Results Available**: A message box asking if the user wants to launch synthesis and implementation. It includes a 'Yes' button and a 'No' button. A red arrow points from the 'Yes' button to the 'Launch Runs' dialog.
- Launch Runs**: A dialog box for launching synthesis or implementation runs. It includes a 'Launch directory' dropdown, 'Options' (radio buttons for 'Launch runs on local host' and 'Generate scripts only'), and a 'Don't show this dialog again' checkbox. A red arrow points from the 'Launch Runs' dialog to the 'Bitstream Generation Completed' dialog.
- Bitstream Generation Completed**: A message box indicating that bitstream generation is complete. It includes a 'Next' section with radio buttons for 'Open Implemented Design', 'View Reports', 'Open Hardware Manager' (selected), and 'Generate Memory Configuration File'. It also has a 'Don't show this dialog again' checkbox and 'OK' and 'Cancel' buttons.

Accélérer la synthèse (utile au projet S4-GE)



Sur des ordinateurs puissants, il est possible de partiellement paralléliser la synthèse. À l'APP1, le projet est très petit et cette configuration n'accélère rien.

Pour les plus gros projets, avec les 8Go de RAM des laboratoires, ne pas mettre plus de 2 « jobs ». Au pire, l'interface graphique de Vivado se ferme sans avertissement, au mieux la synthèse prends plus de temps.

Générer le fichier « bitstream » de configuration

The screenshot shows the Messages window with the following structure:

- Warnings (9):
 - [Constraints 18-23] No constraint will be written out.
 - Implementation (4 warnings):
 - Place Design (1 warning)
 - Route Design (2 warnings):
 - [Power 33-232] No user defined clocks were found in the design! Resolution: Please specify clocks using create_clock/create_generated_clock for sequential elements. For pure combinatorial circuits, please specify a virtual clock, otherwise the vectorless estimation might be inaccurate.
 - [Timing 38-313] There are no user specified timing constraints. Timing constraints are needed for proper timing analysis.
 - Write Bitstream (1 warning):
 - DRC (1 warning):
 - PS7 (1 warning):
 - Zynq requires PS7 block (1 warning):
 - PS7 (1 warning):
 - [DRC ZPS7-1] PS7 block required: The PS7 cell must be used in this Zynq design in order to enable correct default configuration.

Three callout boxes provide context for the warnings:

- Top callout:** Il faut s'attendre à des avertissements. Certains sont acceptables pour la poursuite du processus. C'est le cas dans ce contexte:
- Middle callout:** Il y a absence d'horloge pour l'implémentation de ce circuit purement combinatoire.
- Bottom callout:** Seule la partie PL (« programmable Logic ») est configurée ici alors que la partie PS (« processing system ») est absente de cette configuration. Ref. Zynq reference manual.

Utiliser le « hardware manager » avec la carte Zybo-Z7-10

The screenshot shows the Vivado 2018.2.2 interface. The top menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. The top right shows a status bar with 'write_bitstream Complete' and a green checkmark. The left sidebar contains the Flow Navigator with the following sections: PROJECT MANAGER, IP INTEGRATOR, SIMULATION, RTL ANALYSIS, SYNTHESIS (with Run Synthesis and Open Synthesized Design), IMPLEMENTATION (with Run Implementation and Open Implemented Design), and PROGRAM AND DEBUG (highlighted in blue). Under PROGRAM AND DEBUG, the options are Generate Bitstream, Open Hardware Manager, Open Target, and Program Device. The 'Open Target' option is highlighted with a red rectangle. The main workspace is divided into three panes: Hardware Manager - unconnected (top), Hardware (middle), and Properties (bottom). The Hardware Manager pane shows a message 'No hardware target is open' with an 'Open target' button highlighted by a red rectangle. The Hardware pane shows 'No content'. The Properties pane shows 'Select an object to see properties'. A yellow callout bubble points to the Hardware pane with the text 'La carte Zybo doit être branchée.' The bottom pane is the Tcl Console, showing the following output: 'Run output will be captured here: C:/travail/atelier/atelier.runs/synth_1/runme.log', '[Tue Dec 25 16:12:46 2018] Launched impl_1...', 'Run output will be captured here: C:/travail/atelier/atelier.runs/impl_1/runme.log', 'launch_runs: Time (s): cpu = 00:00:03 ; elapsed = 00:00:07 . Memory (MB): peak = 1259.910 ; gain = 0.000', and 'open_hw'. The bottom of the Tcl Console has a text input field labeled 'Type a Tcl command here'.

Flow Navigator

- PROJECT MANAGER
- IP INTEGRATOR
- SIMULATION
- RTL ANALYSIS
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design
- PROGRAM AND DEBUG
 - Generate Bitstream
 - Open Hardware Manager
 - Open Target
 - Program Device
 - Add Configuration Memory Device

HARDWARE MANAGER - unconnected

No hardware target is open. Open target

Hardware

No content

Properties

Select an object to see properties

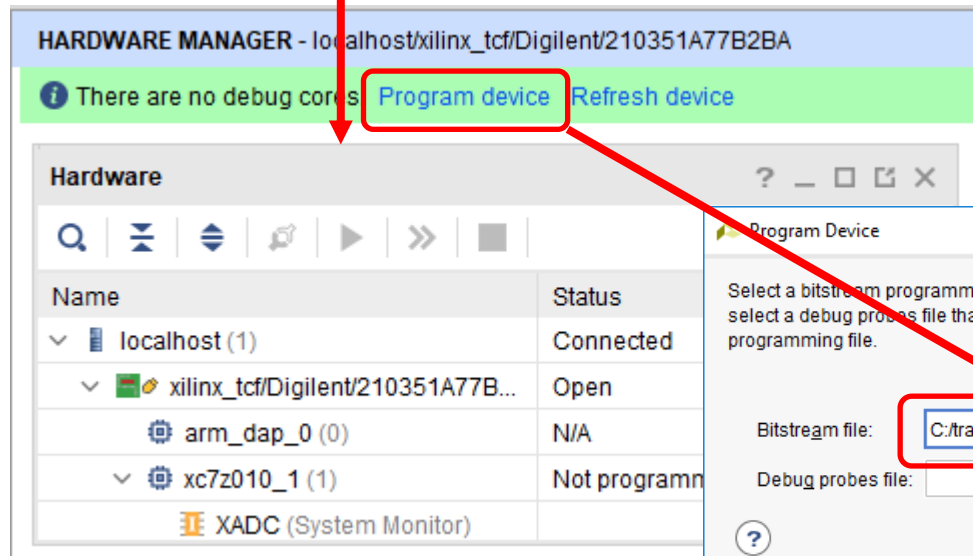
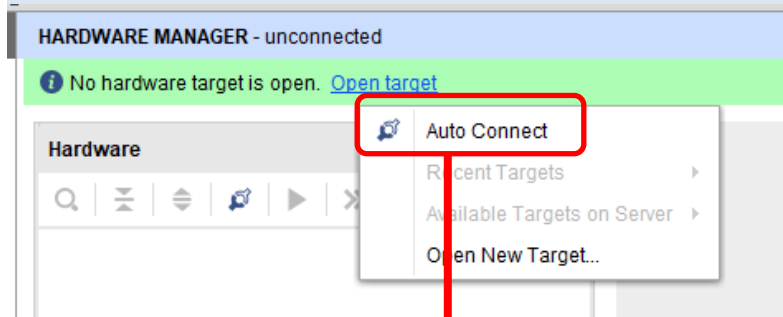
Tcl Console

```
Run output will be captured here: C:/travail/atelier/atelier.runs/synth_1/runme.log
[Tue Dec 25 16:12:46 2018] Launched impl_1...
Run output will be captured here: C:/travail/atelier/atelier.runs/impl_1/runme.log
launch_runs: Time (s): cpu = 00:00:03 ; elapsed = 00:00:07 . Memory (MB): peak = 1259.910 ; gain = 0.000
open_hw
```

Type a Tcl command here

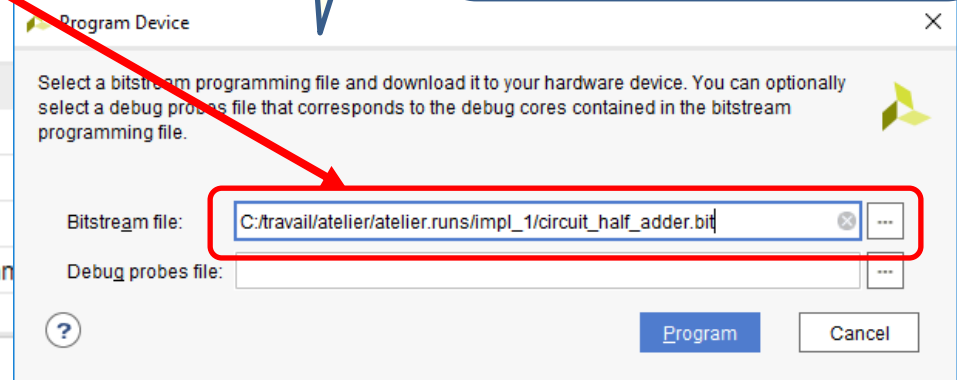
La carte Zybo doit être branchée.

Programmer le circuit



Le nom du fichier bitstream est associé à l'entité du « top-module ».

Le circuit est programmé. Vérifier maintenant le fonctionnement sur la carte.
Manœuvrer les interrupteurs sw0 et sw1 pour faire un test complet.



Test de validation sur la carte Zybo

HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210351A77B2BA

There are no debug cores. [Program device](#) [Refresh device](#)

Hardware

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210351A77B...	Open
arm_dap_0 (0)	N/A
xc7z010_1 (1)	Programmed

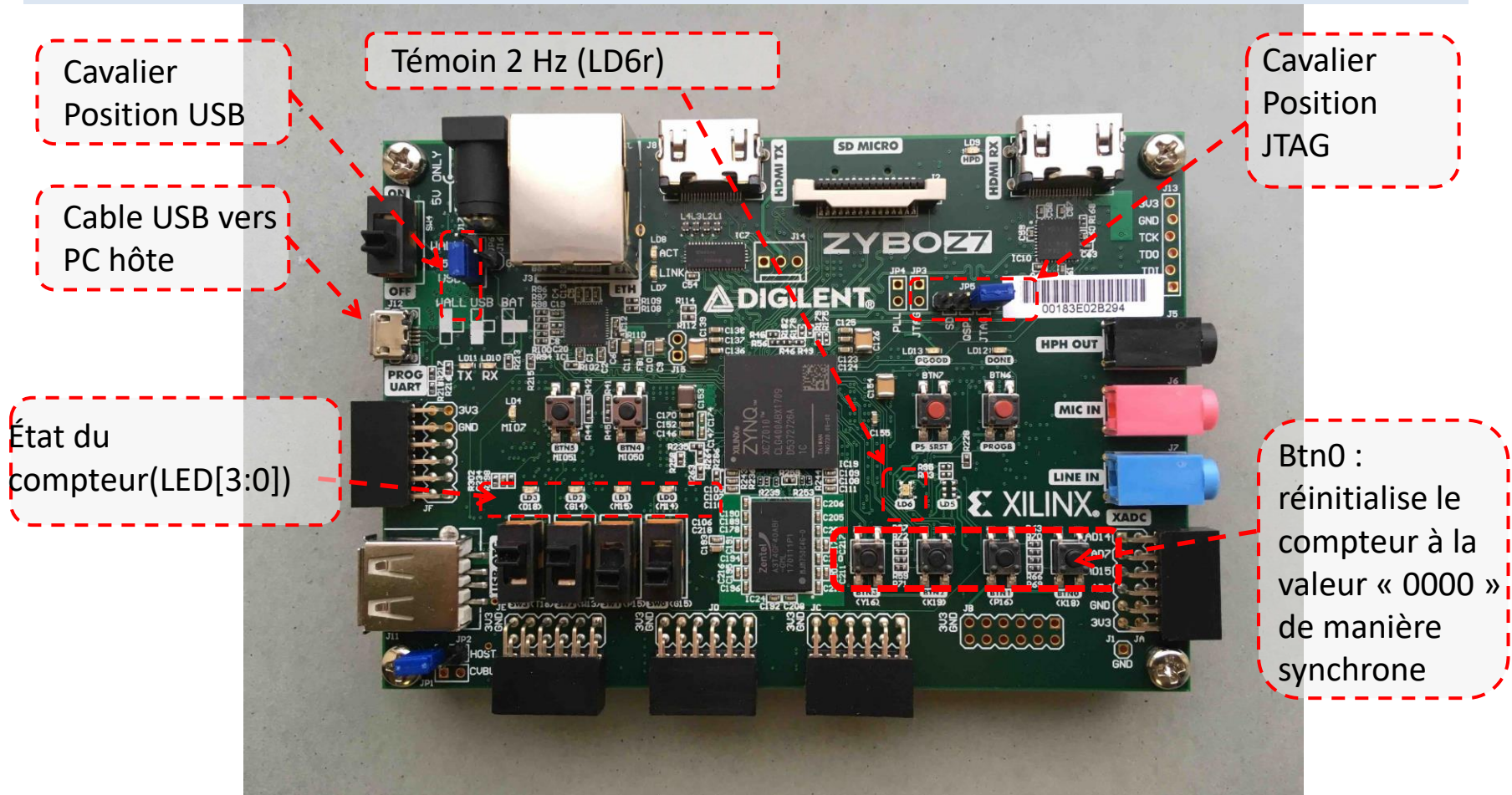
Le circuit est programmé. Vérifier maintenant le fonctionnement sur la carte.
Manœuvrer les interrupteurs sw0 et sw1 pour faire un test complet.

Étapes de la partie 2 « circuit_2.vhd » RAPPEL

- Insérer des fichiers « circuit_2.vhd » dans la hiérarchie
- Définir l'association du fichier de contraintes requis
- Explorer et vérifier l'arborescence du circuit (« top-module »)
- Analyse RTL et schéma avant d'intégrer la composante *compteur_simple*
- Intégrer la composante *compteur_simple* dans le code VHDL
- Faire l'analyse RTL du circuit complet
- Insérer (ou créer) un code de « test bench »
- Simuler le circuit
- Générer le fichier de configuration « bitstream »
- Programmer et tester sur la carte
- Archiver le dossier de projet Vivado

Fonctions du circuit « *circuit_compteur* »

Le circuit compteur implémente un compteur binaire à 4 bits qui s'incrémente à une cadence de 2 Hz et un bouton fait une remise à zéro.



Insérer des fichiers « circuit_2.vhd » dans la hiérarchie

The image shows a sequence of three windows from the Vivado IDE:

- PROJECT MANAGER - atelier**: The left sidebar shows the 'Sources' pane. A right-click context menu is open over the 'Sources' list. The 'Add Sources...' option at the bottom is highlighted with a red box. A red arrow points from this option to the 'Add Sources' dialog.
- Add Sources**: This dialog box guides the user through adding sources. The 'Add or create design sources' radio button is selected. The 'Next >' button is highlighted with a red arrow pointing to the 'Add Source Files' dialog.
- Add Source Files**: This dialog shows the 'Look in' directory as 'fichiers_atelier'. A list of files is shown: 'circuit_2.vhd', 'clkdiv_module.vhd', 'compteur_simple.vhd', and 'compteur_simple_tb.vhd'. The 'File Preview' section shows the content of 'circuit_2.vhd'. The 'File name' field contains the names of the three selected files. The 'Files of type' dropdown is set to 'Design Source Files (.vhd, .vhdl, .vhf, .vhdp, .vho, .v, .vf, .verilog, .vr, .vg, .vb, .tf, .vlog, .vp, .vm, .veo, .svo, .vh, .h, .svh, .vhp, .sv)'. The 'OK' button is highlighted with a red arrow pointing from the 'Add Sources' dialog.

A yellow callout box with a blue border contains the text: "On ajoute les 3 fichiers requis pour la conception du circuit."

Insérer des fichiers « circuit_2.vhd » dans la hiérarchie

Add Sources

Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

	Index	Name	Library	Location
●	1	circuit_2.vhd	xil_defaultlib	C:/travail/fichiers_atelier
●	2	clkdiv_module.vhd	xil_defaultlib	C:/travail/fichiers_atelier
●	3	compteur_simple.vhd	xil_defaultlib	C:/travail/fichiers_atelier

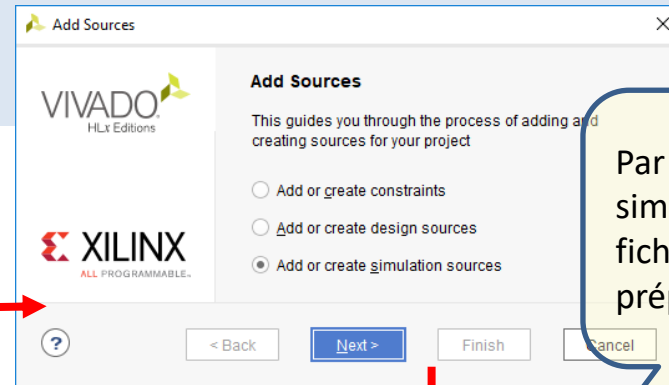
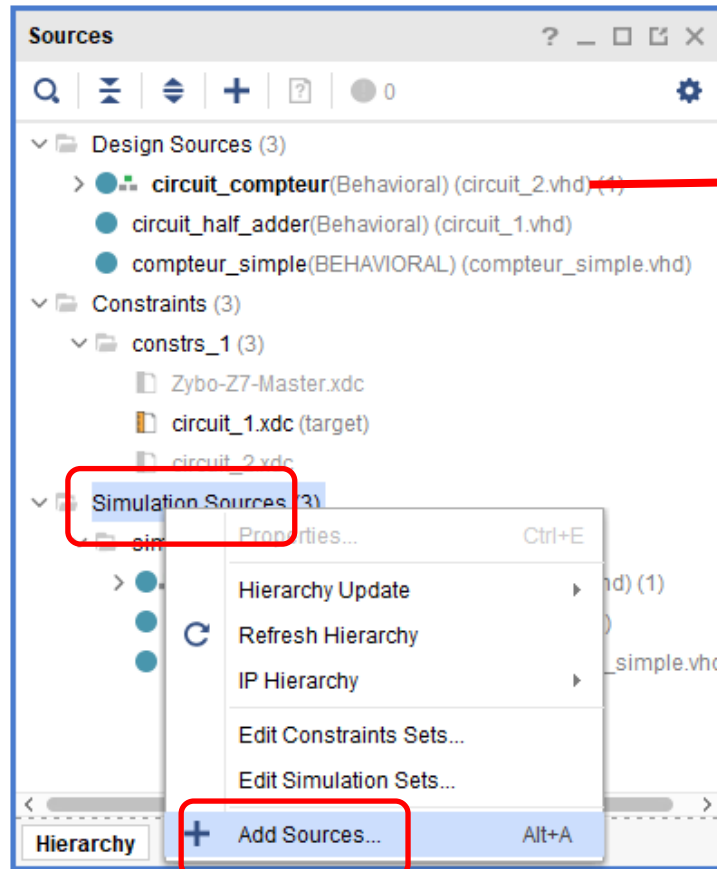
☐ Scan and add RTL include files into project

☒ **Copy sources into project**

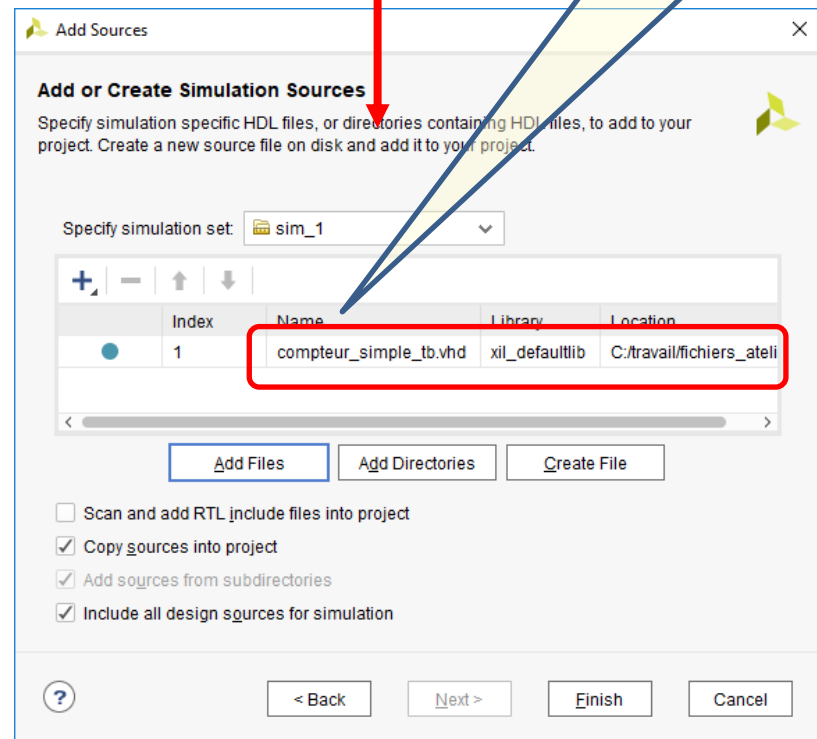
☒ Add sources from subdirectories

Si on ne coche pas cette case, les fichiers seront seulement référés, mais non copiés.

Insérer des fichiers « circuit_2.vhd» dans la hiérarchie

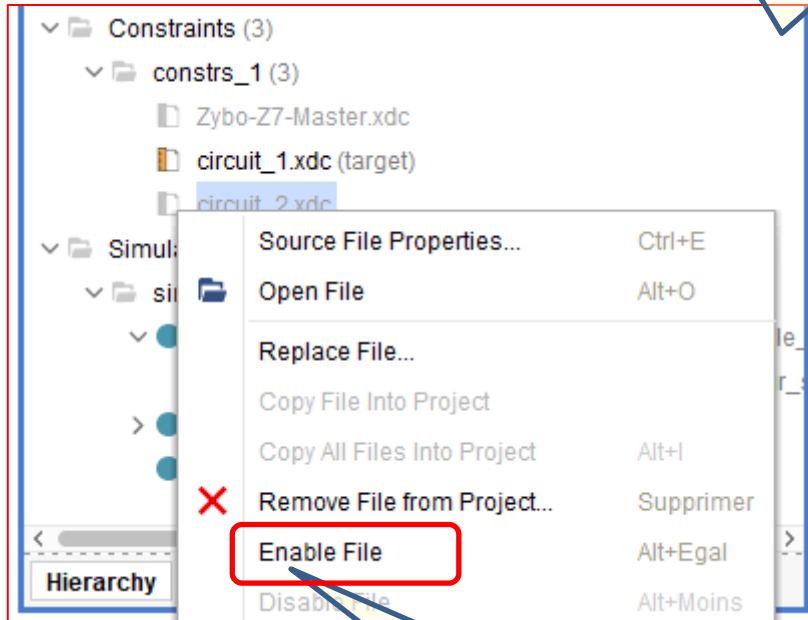


Par une procédure similaire, on insère le fichier « test bench » préparé d'avance.

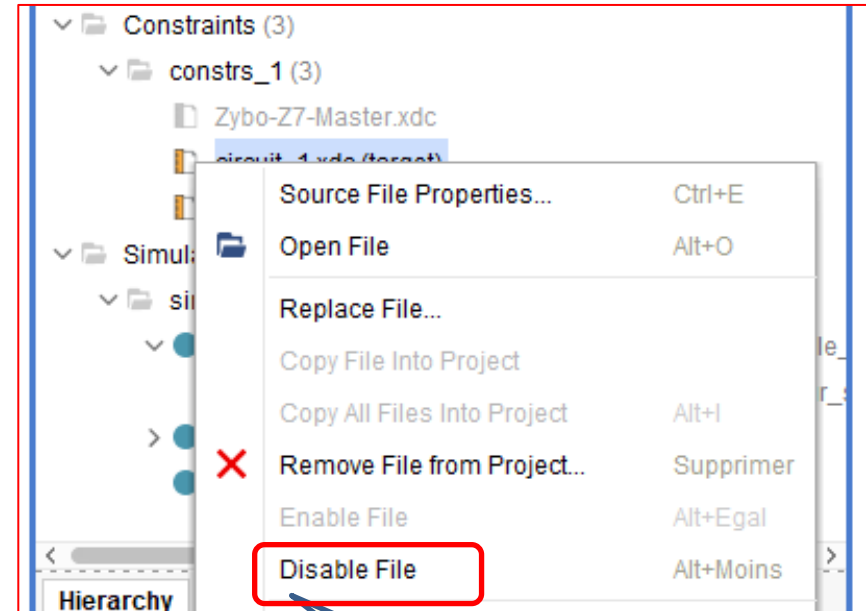


Définir l'association du fichier de contraintes requis

Il faut associer (exclusivement) le fichier de contraintes adapté au circuit circuit_2.



Pour circuit_2.xdc



Pour circuit_1.xdc

Définir l'association du fichier de contraintes requis

Pour information: lignes déterminantes dans circuit_2.xdc

```
13 | ##Clock signal
14 | set_property -dict { PACKAGE_PIN K17    IOSTANDARD LVCMOS33 } [get_ports { sysclk }];
15 | create_clock -add -name sys_clk_pin -period 8.00 -waveform {0 4} [get_ports { sysclk }];

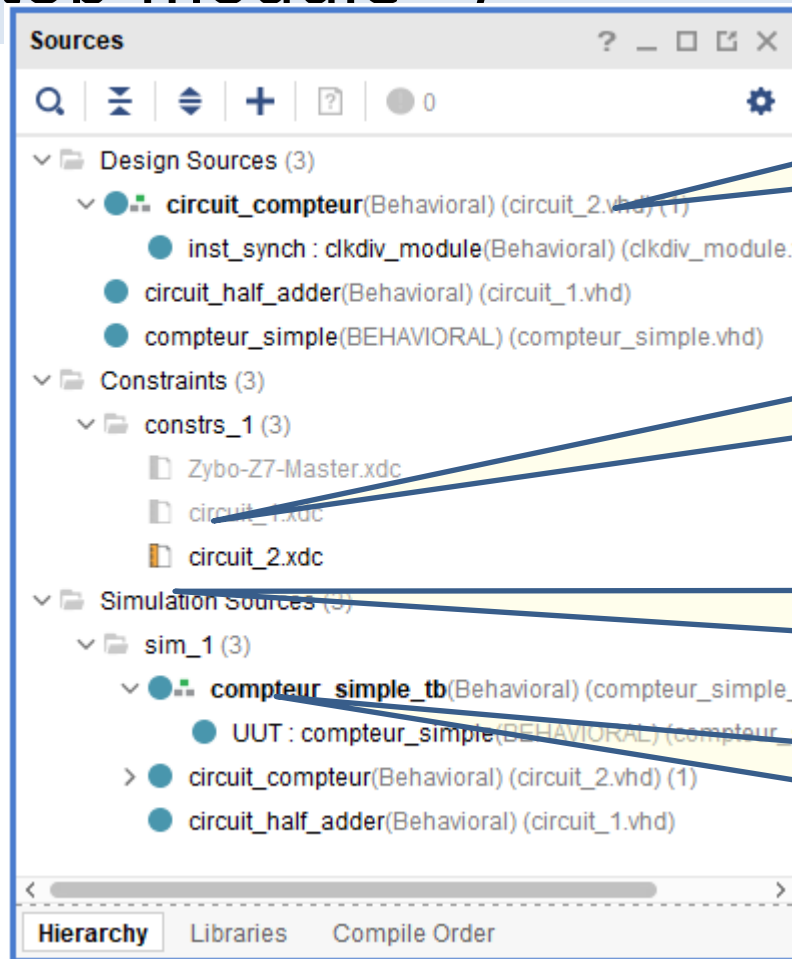
30 | ##Switches (circuit_atelier circuit_2)
31 | set_property -dict { PACKAGE_PIN G15    IOSTANDARD LVCMOS33 } [get_ports { i_sw[0] }];
32 | set_property -dict { PACKAGE_PIN P15    IOSTANDARD LVCMOS33 } [get_ports { i_sw[1] }];
33 | set_property -dict { PACKAGE_PIN W13    IOSTANDARD LVCMOS33 } [get_ports { i_sw[2] }];
34 | set_property -dict { PACKAGE_PIN T16    IOSTANDARD LVCMOS33 } [get_ports { i_sw[3] }];

42 | ##Buttons (circuit_atelier)
43 | set_property -dict { PACKAGE_PIN K18    IOSTANDARD LVCMOS33 } [get_ports { i_btn[0] }];
44 | set_property -dict { PACKAGE_PIN P16    IOSTANDARD LVCMOS33 } [get_ports { i_btn[1] }];
45 | set_property -dict { PACKAGE_PIN K19    IOSTANDARD LVCMOS33 } [get_ports { i_btn[2] }];
46 | set_property -dict { PACKAGE_PIN Y16    IOSTANDARD LVCMOS33 } [get_ports { i_btn[3] }];

55 | ##LEDs (circuit_atelier)
56 | set_property -dict { PACKAGE_PIN M14    IOSTANDARD LVCMOS33 } [get_ports { o_led[0] }];
57 | set_property -dict { PACKAGE_PIN M15    IOSTANDARD LVCMOS33 } [get_ports { o_led[1] }];
58 | set_property -dict { PACKAGE_PIN G14    IOSTANDARD LVCMOS33 } [get_ports { o_led[2] }];
59 | set_property -dict { PACKAGE_PIN D18    IOSTANDARD LVCMOS33 } [get_ports { o_led[3] }];
60 |

71 | ##RGB LED 6 (circuit_atelier)
72 | set_property -dict { PACKAGE_PIN V16    IOSTANDARD LVCMOS33 } [get_ports { o_led6_r }];
```

Explorer et vérifier l'arborescence du circuit (« top-module »)



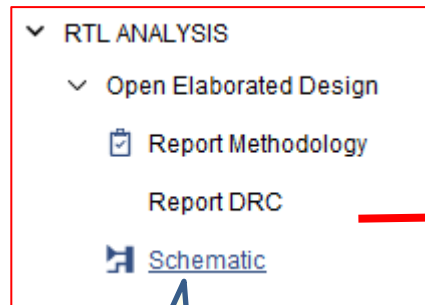
Circuit_2.vhd contenant l'entité circuit_compteur a été promue implicitement « top-module ». Vérifier cet état dans votre hiérarchie et l'imposer si ce n'est pas le cas.

On a spécifié « disable » pour le fichier de contraintes circuit_1.xdc et « enable » pour circuit_2.xdc.

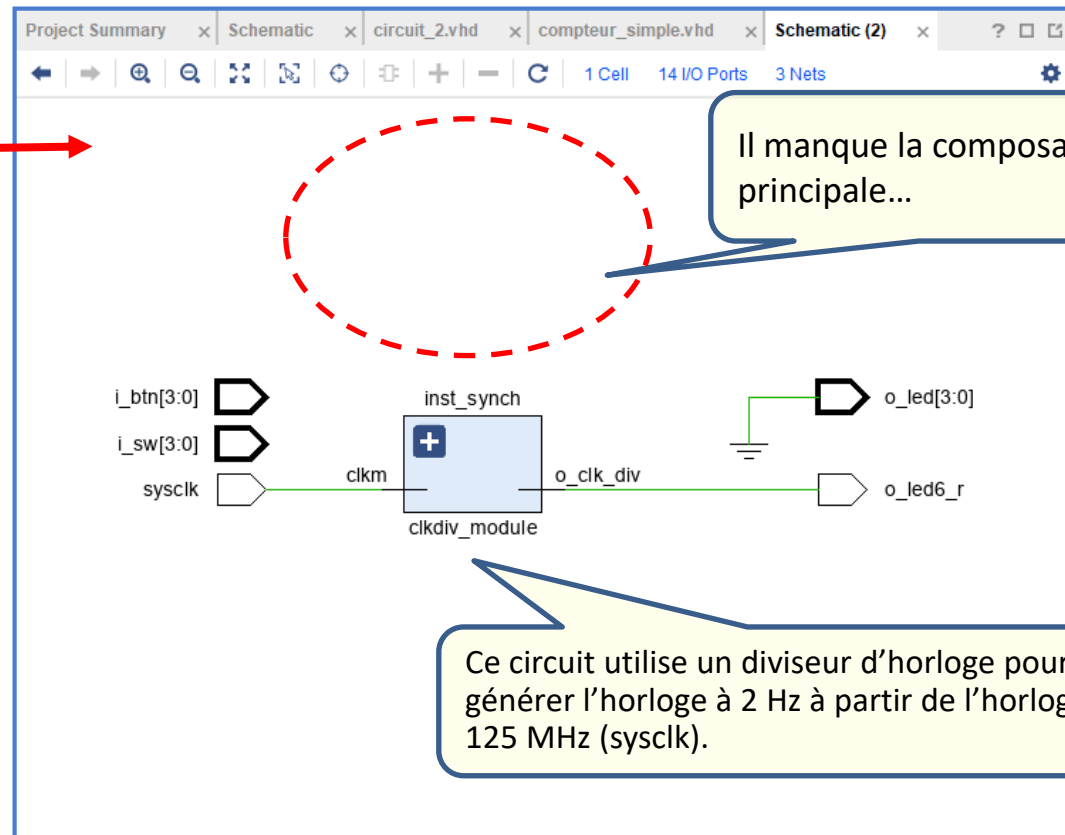
Noter que l'on a pas spécifié circuit_2.xdc come « target file ». Ce n'est pas indispensable ici car l'IDE Vivado n'écrit pas de nouvelle contraintes.

Ne pas oublier de spécifier le fichier du « test bench » ciblé comme « top-module » aussi.

Analyse RTL et schéma avant d'intégrer la composante *compteur_simple*



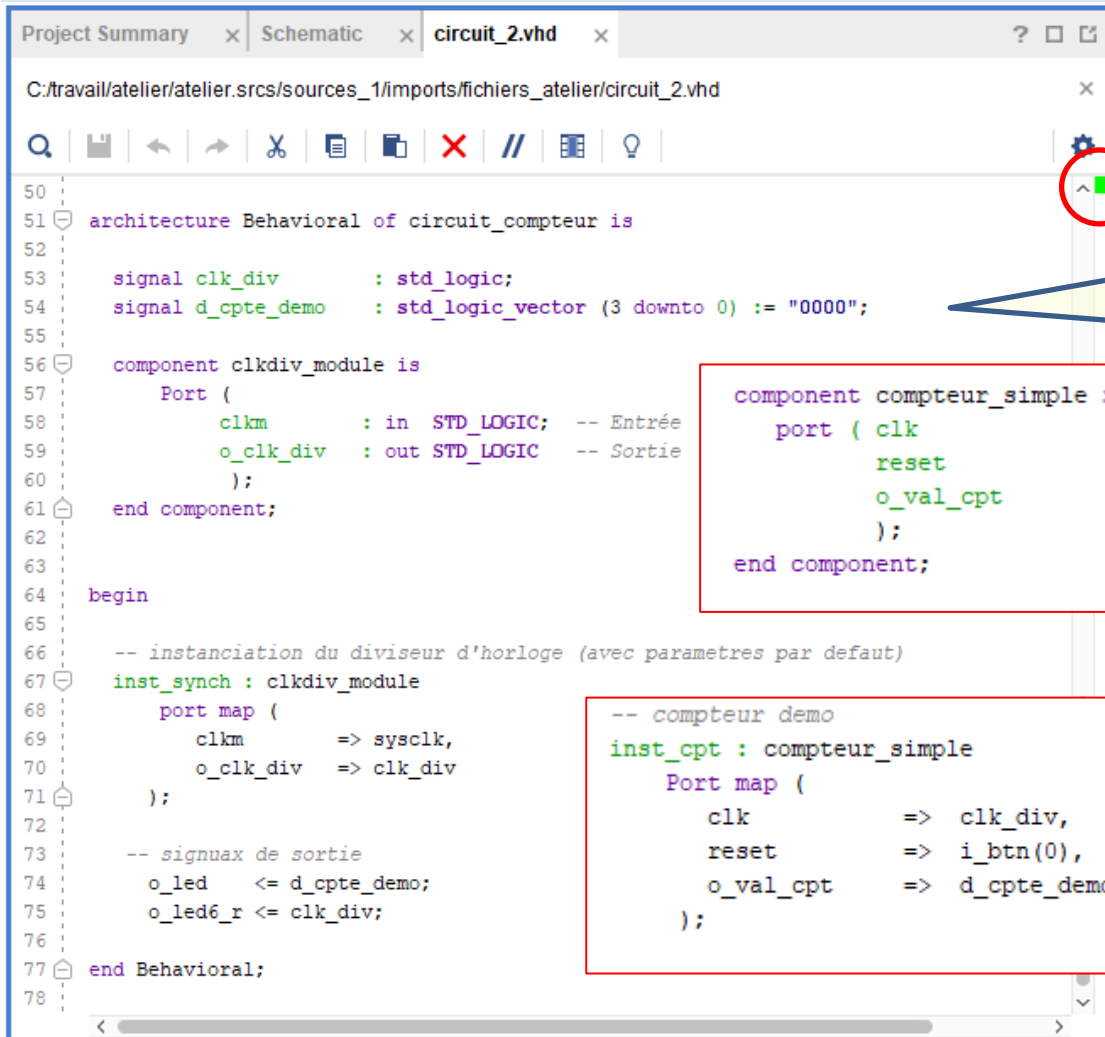
Étape utile pour vérifier et documenter un circuit en conception.



Il manque la composante principale...

Ce circuit utilise un diviseur d'horloge pour générer l'horloge à 2 Hz à partir de l'horloge à 125 MHz (*sysclk*).

Intégrer la composante *compteur_simple* dans le code VHDL – ajouter le code encadré dans le .vhd



```
50
51 architecture Behavioral of circuit_compteur is
52
53     signal clk_div       : std_logic;
54     signal d_cpte_demo   : std_logic_vector (3 downto 0) := "0000";
55
56     component clkdiv_module is
57     Port (
58         clkkm       : in  STD_LOGIC; -- Entrée
59         o_clk_div    : out STD_LOGIC  -- Sortie
60     );
61 end component;
62
63 begin
64
65     -- instanciación du diviseur d'horloge (avec parametres par default)
66     inst_synch : clkdiv_module
67     port map (
68         clkkm      => sysclk,
69         o_clk_div  => clk_div
70     );
71
72     -- signaux de sortie
73     o_led    <= d_cpte_demo;
74     o_led6_r <= clk_div;
75
76 end Behavioral;
77
78
```

Signal de validation de la syntaxe.

L'éditeur (pour les fichiers de la hiérarchie seulement) réagit aux erreurs de syntaxe et donne des messages à l'écran.

```
component compteur_simple is
    port ( clk       : in  std_logic;
          reset      : in  std_logic;
          o_val_cpt   : out std_logic_vector (3 downto 0)
        );
end component;
```

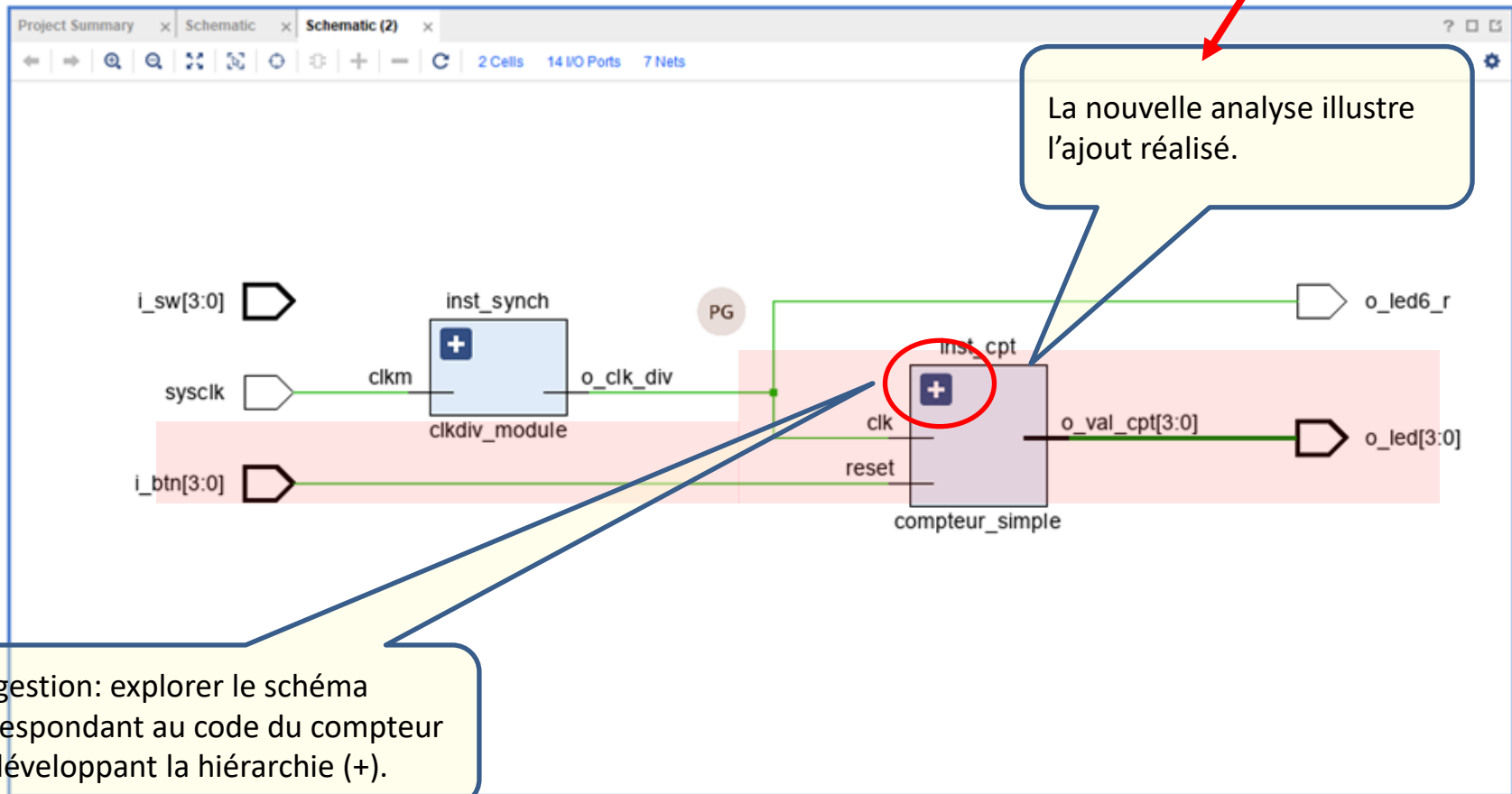
```
-- compteur demo
inst_cpt : compteur_simple
Port map (
    clk       => clk_div,
    reset     => i_btn(0),
    o_val_cpt => d_cpte_demo
);
```

Intégrer le code de ces deux composantes dans le code VHDL.

Faire l'analyse RTL du circuit complet

ELABORATED DESIGN - xc7z010clg400-1 (active)

! Elaborated Design is out-of-date. Design sources were modified. [details](#) [Reload](#)

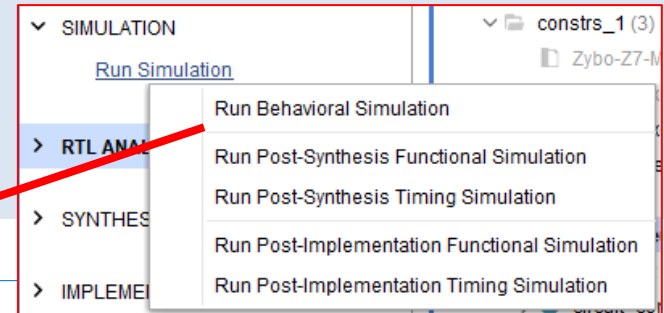


Insérer (ou créer) un code de « test bench »

```
28 library IEEE;
29 use IEEE.STD_LOGIC_1164.ALL;
30
31 entity compteur_simple_tb is
32 -- Port ( );
33 end compteur_simple_tb;
34
35 architecture Behavioral of compteur_simple_tb is
36
37 COMPONENT compteur_simple
38     port (
39         clk          : in    std_logic;
40         reset        : in    std_logic;
41         o_val_cpt    : out   std_logic_vector (3 downto 0)
42     );
43 end COMPONENT;
44
45 signal clk_sim      : STD_LOGIC := '0';
46 signal reset        : STD_LOGIC := '0';
47 signal val_cpt      : std_logic_vector
48 constant sysclk_Period : time := 10 ns;
49
52 begin
53
54 -- UUT : Unit Under Test
55 UUT: compteur_simple
56     PORT MAP(
57         clk      => clk_sim,
58         reset    => reset,
59         o_val_cpt => val_cpt
60     );
61
62 -- generation horloge simulee
63 process
64 begin
65     clk_sim <= '1';
66     loop
67         wait for sysclk_Period/2;
68         clk_sim <= not clk_sim;
69     end loop;
70 end process;
71
72 -- test bench
73 tb : PROCESS
74
75 BEGIN
76     -- simuler une sequence de valeurs a l'entree
77     reset <= '0';
78     wait for 100 ns;
79     reset <= '1';
80     wait for sysclk_Period;
81     reset <= '0';
82     wait for 10*sysclk_Period;
83     reset <= '1';
84     wait for sysclk_Period;
85     reset <= '0';
86     WAIT; -- will wait forever
87 END PROCESS;
88
89 END Behavioral;
```

Ce fichier de simulation
existe d'avance pour l'atelier.
Il a déjà été inséré.
Voici les parties essentielles.

Simuler le circuit



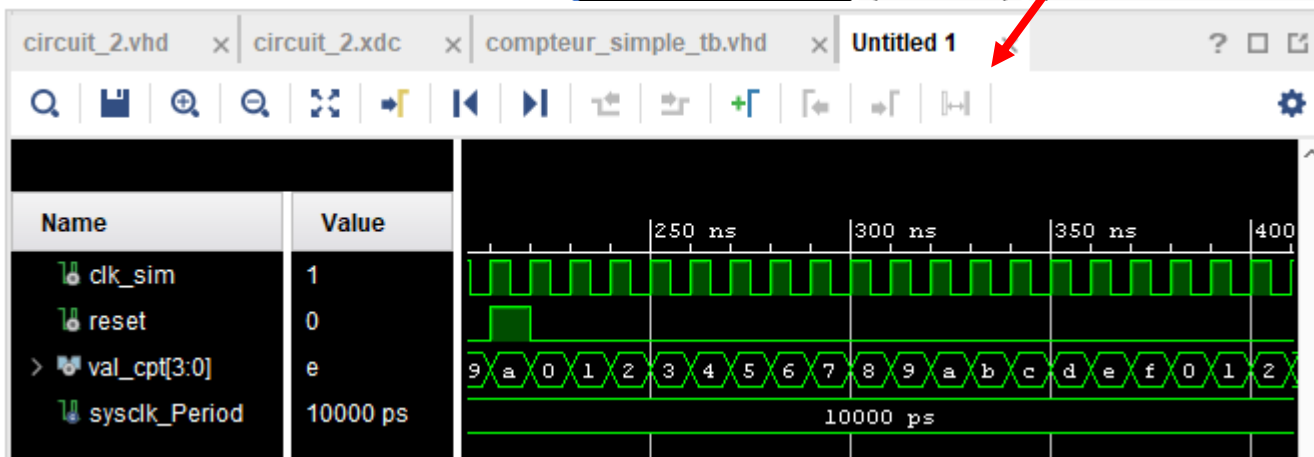
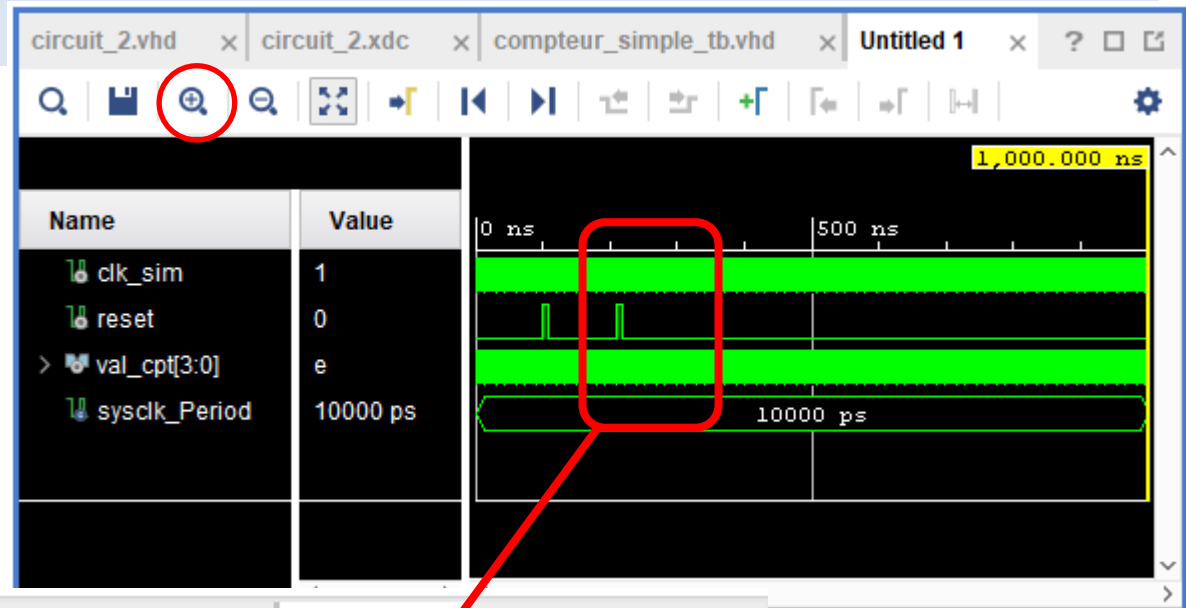
A screenshot of the Vivado IDE interface. The top toolbar shows the 'Run' button (a green play icon) with a dropdown menu. A red arrow points from the 'Run' button to the 'Run Simulation' menu shown in the previous image. The main workspace is divided into several panes:

- Flow Navigator:** Shows the project hierarchy with 'SIMULATION' selected.
- Scope:** Lists the design units and block types for the simulation.
- Objects:** Lists the objects in the simulation, including 'clk_sim', 'reset', 'val_cpt...', and 'sysclk...'. It shows their values and data types.
- Waveform:** Displays a timing diagram for the selected objects, showing signals like 'clk_sim', 'reset', 'val_cpt[3:0]', and 'sysclk_Period' over time.
- Tcl Console:** Shows the command history and output of the simulation, including the command 'with args "compteur_simple_tb_behav -key {Behavioral:sim_1:Functional:compteur_simple_tb} -tclbatch {compteur_simple_tb.tcl} -log {simulate.log}"' and the output 'INFO: [USF-XSim-96] XSim completed. Design snapshot 'compteur_simple_tb_behav' loaded.'

The bottom status bar indicates 'Sim Time: 1 us'.

Simuler le circuit

Explorer le synchrogramme résultant.



Simuler le circuit

SIMULATION - Behavioral Simulation - Functional - sim_1 - compteur_simple_tb

Scope x Sources

Name	Design U...	Block Type
compteur_si...	compteu...	VHDL En...
UUT	compteu...	VHDL En...

Objects ? _ □ □ X

Name	Value
clk	1
reset	0
> o_val_cpt[3:0]	6
> NQ[3:0]	7

Explorer les signaux de l'unité testée.

Ajouter ceux que l'on désire observer...

Objects ? _ □ □ X

Name	Value
clk	1
reset	0
> o_val_c...	e
> NQ[3:0]	f
> Q[3:0]	

Add to Wave Window
Log to Wave Database
Show in Wave Window
Go to Source Code
Radix
Show as Enumeration

circuit_2.vhd x circuit_2.xdc x compteur_simple_tb.vhd x Untitled 1 x

Name Value

clk_sim	1
reset	0
> o_val_cpt[3:0]	e
sysclk_Period	10000 ps

250 ns 300 ns 350 ns 400 ns

9 a 0 1 2 3 4 5 6 7 8 9 a b c d e f 0 1 2

10000 ps

Simuler le circuit

The screenshot shows the top of the simulation tool. The 'Run' menu is highlighted with a red box, and a yellow arrow points to it. Below the menu, the 'Objects' table is visible:

Name	Value
clk	1
reset	0
> o_val_c...	e
> NQ[3:0]	f
> Q[3:0]	e

The 'circuit_2.vhd' and 'circuit_2.xdc' files are open, and the 'compteur_simple_tb.vhd' file is selected. The 'Untitled 1*' window shows a waveform with a clock signal and a reset signal. The clock signal is a square wave with a period of 10000 ps. The reset signal is a single pulse. The waveform also shows the output of the counter, which is a sequence of hexadecimal digits: 9, a, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, 0, 1, 2.

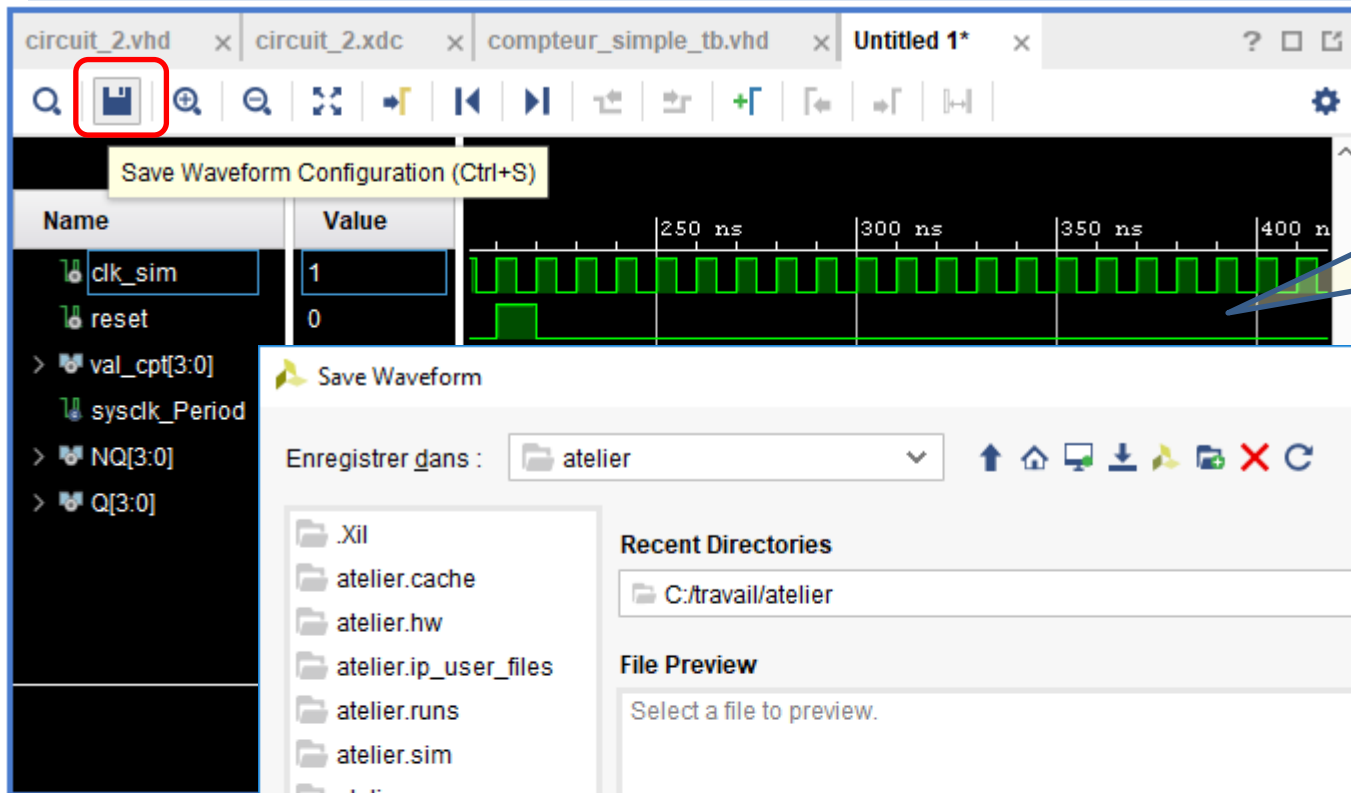
Les signaux sont présents, il faut rejouer la simulation pour afficher leurs tracés.

The screenshot shows the simulation tool after the simulation has been restarted. The 'Run' menu is highlighted with a red box, and a red arrow points to it. Below the menu, the 'Objects' table is visible:

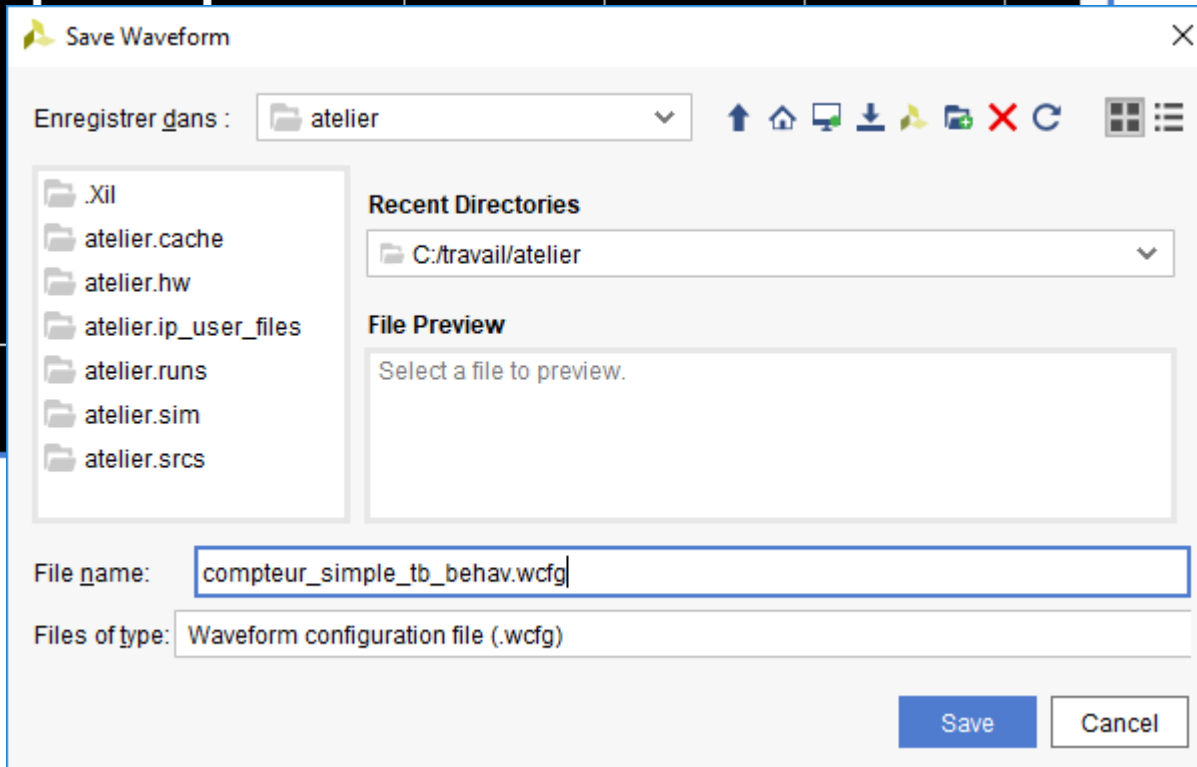
Name	Value
clk_sim	1
reset	0
> valcpt[3:0]	6
sysclk_Period	10000 ps
> NQ[3:0]	7
> Q[3:0]	6

The 'circuit_2.vhd' and 'circuit_2.xdc' files are open, and the 'compteur_simple_tb.vhd' file is selected. The 'Untitled 1*' window shows a waveform with a clock signal and a reset signal. The clock signal is a square wave with a period of 10000 ps. The reset signal is a single pulse. The waveform also shows the output of the counter, which is a sequence of hexadecimal digits: a, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, 0, 1, 2.

Simuler le circuit



On peut sauvegarder les fichiers de configuration d'une simulation pour usage ultérieur.



Simuler le circuit

Waveform Configuration File



Would you like to add the waveform configuration file: 'compteur_simple_tb_behav.wcfg' to your project?

☐ Don't show this dialog again

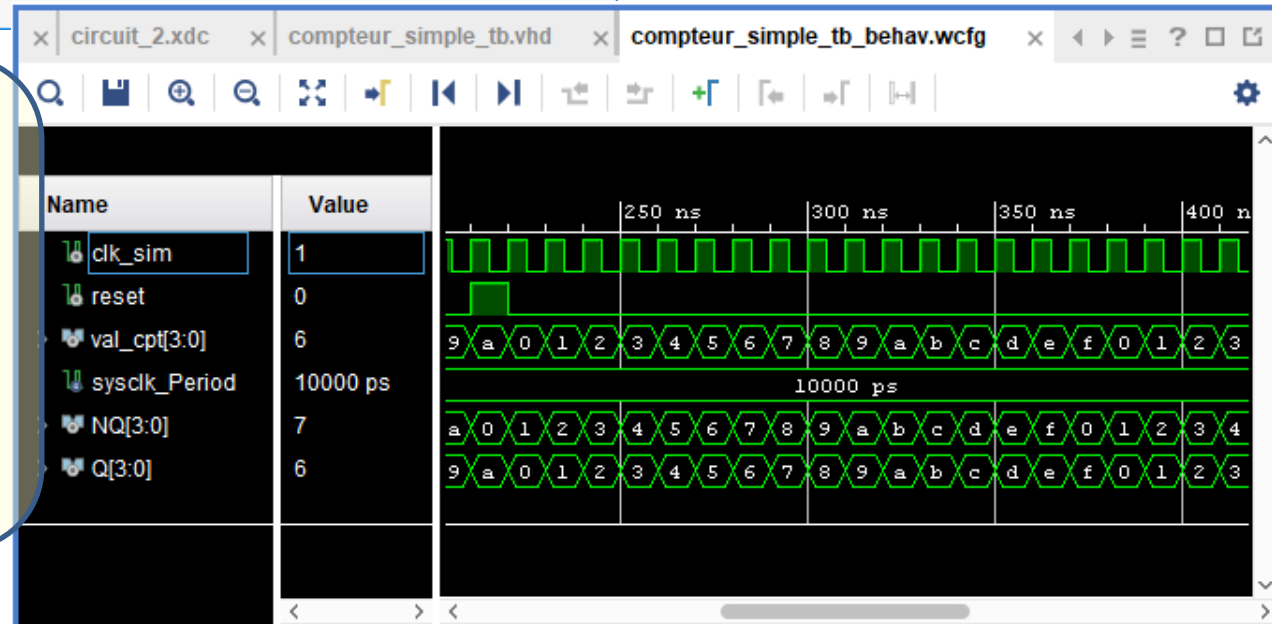
Yes

No

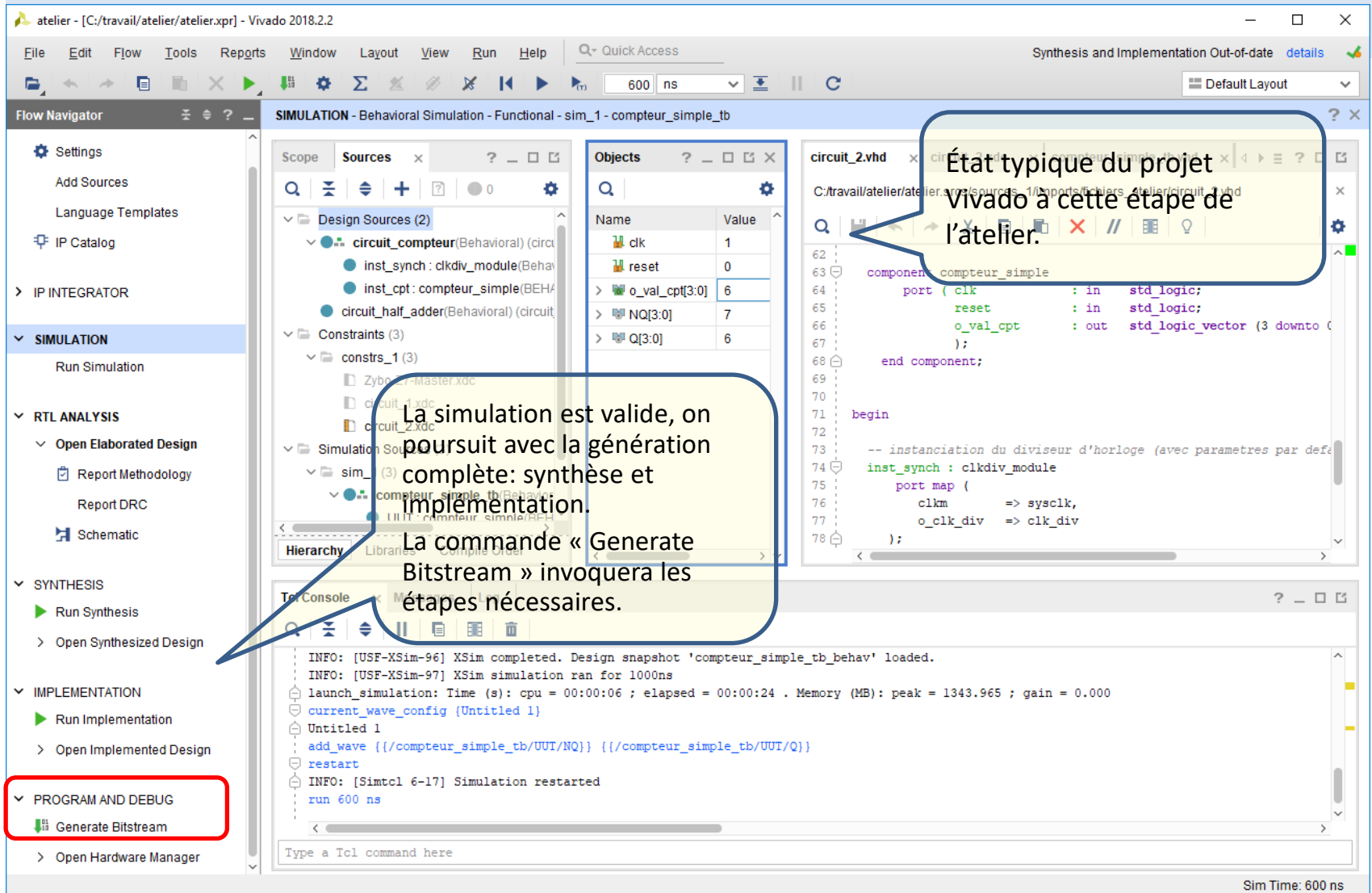
Option qui permet de faire apparaître ces fichiers dans l'interface de la hiérarchie.

Une remarque: comme les fichiers de contraintes, l'interface de la hiérarchie permet d'activer (« enable ») ou de désactiver (« disable ») un fichier.

C'est important à considérer quand plusieurs tests coexistent dans un projet.



Générer le fichier de configuration « bitstream »



atelier - [C:/travail/atelier/atelier.xpr] - Vivado 2018.2.2

File Edit Flow Tools Reports Window Layout View Run Help

Quick Access

Synthesis and Implementation Out-of-date details

Default Layout

Flow Navigator

- Settings
- Add Sources
- Language Templates
- IP Catalog
- IP INTEGRATOR
- SIMULATION**
 - Run Simulation
- RTL ANALYSIS
 - Open Elaborated Design
 - Report Methodology
 - Report DRC
 - Schematic
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION**
 - Run Implementation
 - Open Implemented Design
- PROGRAM AND DEBUG**
 - Generate Bitstream**
 - Open Hardware Manager

SIMULATION - Behavioral Simulation - Functional - sim_1 - compteur_simple_tb

Scope Sources

Design Sources (2)

- circuit_compteur(Behavioral) (circuit)
- inst_synth: clkdiv_module(Behavioral) (circuit)
- inst_cpt: compteur_simple(Behavioral) (circuit)
- circuit_half_adder(Behavioral) (circuit)

Constraints (3)

- constrs_1 (3)
- Zybo-Z7010-master.xdc
- circuit_1.xdc
- circuit_2.xdc

Simulation Sources (3)

- sim_1 (3)
- compteur_simple_tb(Behavioral) (circuit)
- compteur_simple_tb(Behavioral) (circuit)
- compteur_simple_tb(Behavioral) (circuit)

Hierarchy

Library

Objects

Name	Value
clk	1
reset	0
o_val_cpt[3:0]	6
NQ[3:0]	7
Q[3:0]	6

circuit_2.vhd

C:/travail/atelier/atelier.sources/1/sources/fichiers/atelier/circuit_2.vhd

```
62
63
64 component compteur_simple
65 port ( clk           : in  std_logic;
66       reset          : in  std_logic;
67       o_val_cpt       : out  std_logic_vector (3 downto 0) );
68
69
70
71
72
73
74 -- instantiation du diviseur d'horloge (avec parametres par defaut)
75 inst_synth : clkdiv_module
76 port map (
77     clk     => sysclk,
78     o_clk_div => clk_div
79 );
```

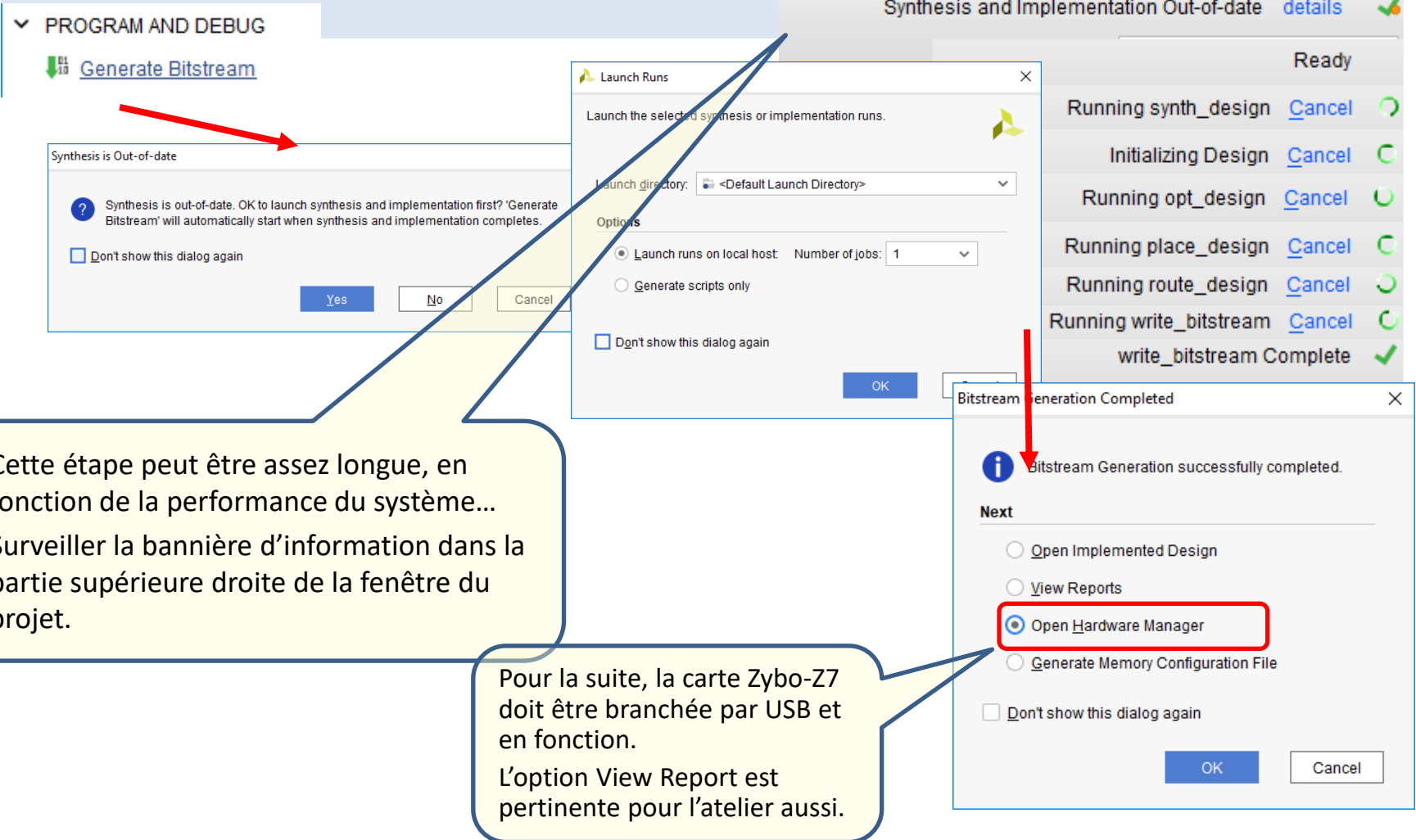
Console

```
INFO: [USF-XSim-96] XSim completed. Design snapshot 'compteur_simple_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:06 ; elapsed = 00:00:24 . Memory (MB): peak = 1343.965 ; gain = 0.000
current_wave_config {Untitled 1}
Untitled 1
add_wave {[compteur_simple_tb/UUT/NQ]} {[compteur_simple_tb/UUT/Q]}
restart
INFO: [Simtcl 6-17] Simulation restarted
run 600 ns
```

Type a Tcl command here

Sim Time: 600 ns

Programmer et tester sur la carte



PROGRAM AND DEBUG

[Generate Bitstream](#)

Synthesis is Out-of-date

? Synthesis is out-of-date. OK to launch synthesis and implementation first? 'Generate Bitstream' will automatically start when synthesis and implementation completes.

☐ Don't show this dialog again

Yes No Cancel

Launch Runs

Launch the selected synthesis or implementation runs.

Launch directory: <Default Launch Directory>

Options

☒ Launch runs on local host: Number of jobs: 1

☐ Generate scripts only

☐ Don't show this dialog again

OK

Ready

Running synth_design Cancel

Initializing Design Cancel

Running opt_design Cancel

Running place_design Cancel

Running route_design Cancel

Running write_bitstream Cancel

write_bitstream Complete

Bitstream Generation Completed

i Bitstream Generation successfully completed.

Next

☐ Open Implemented Design

☐ View Reports

☒ Open Hardware Manager

☐ Generate Memory Configuration File

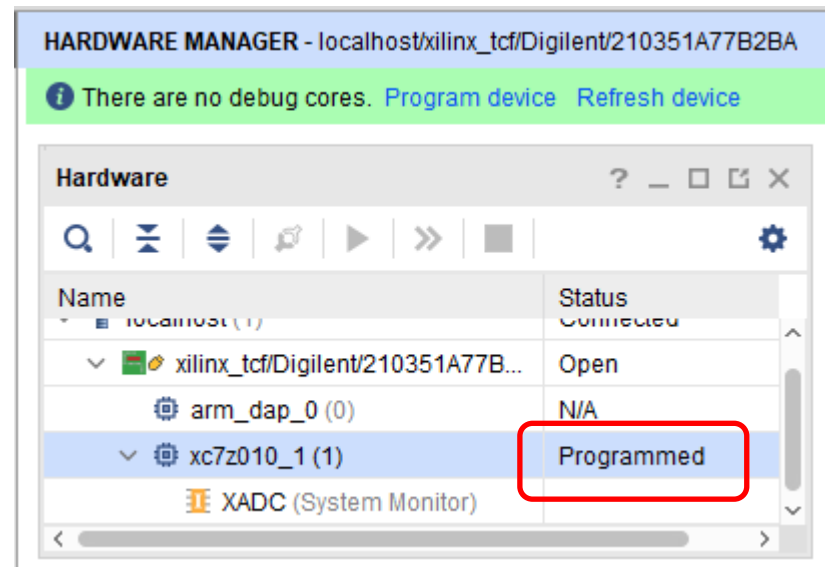
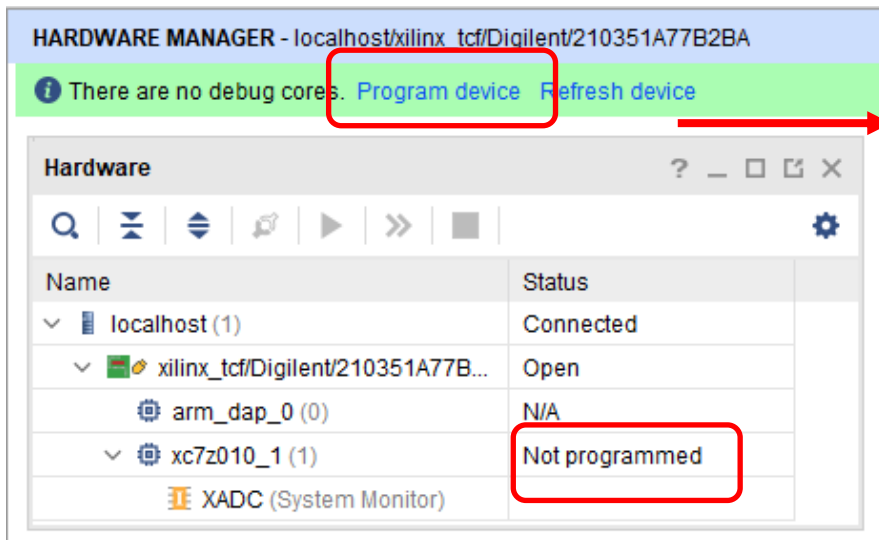
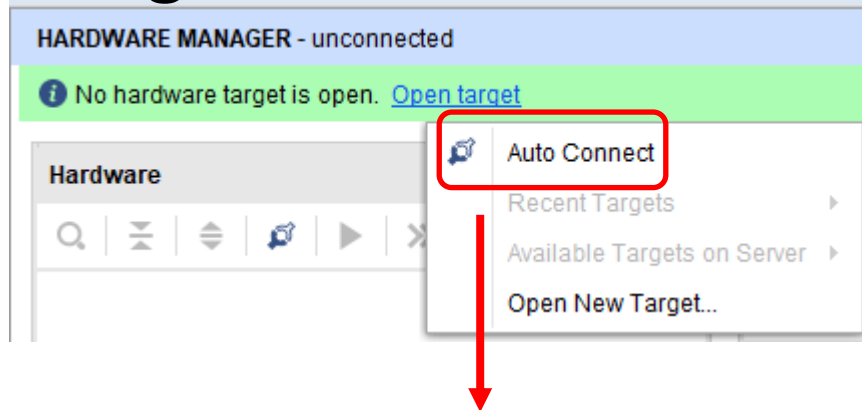
☐ Don't show this dialog again

OK Cancel

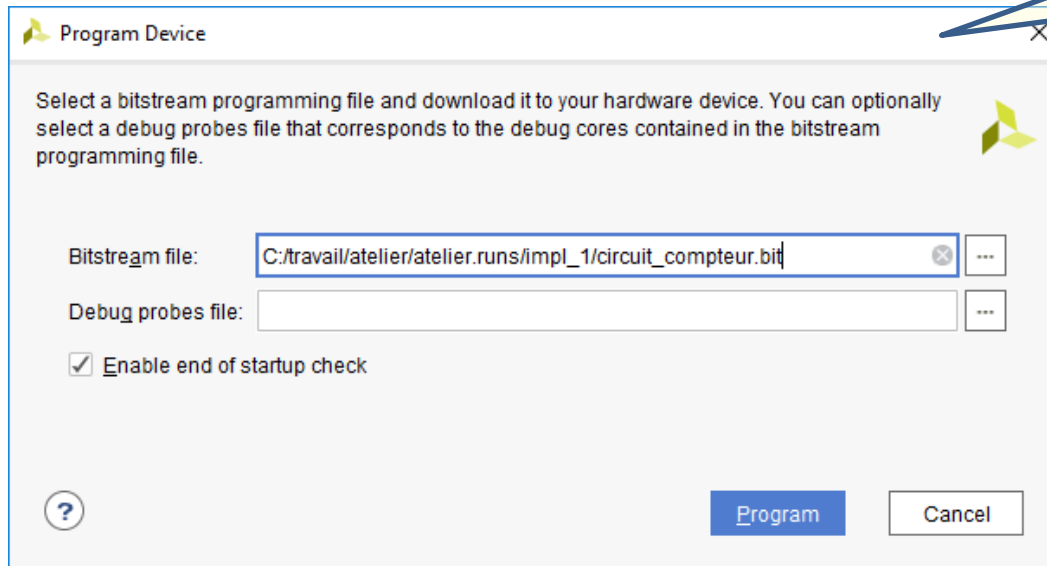
Cette étape peut être assez longue, en fonction de la performance du système... Surveiller la bannière d'information dans la partie supérieure droite de la fenêtre du projet.

Pour la suite, la carte Zybo-Z7 doit être branchée par USB et en fonction. L'option View Report est pertinente pour l'atelier aussi.

Programmer et tester sur la carte



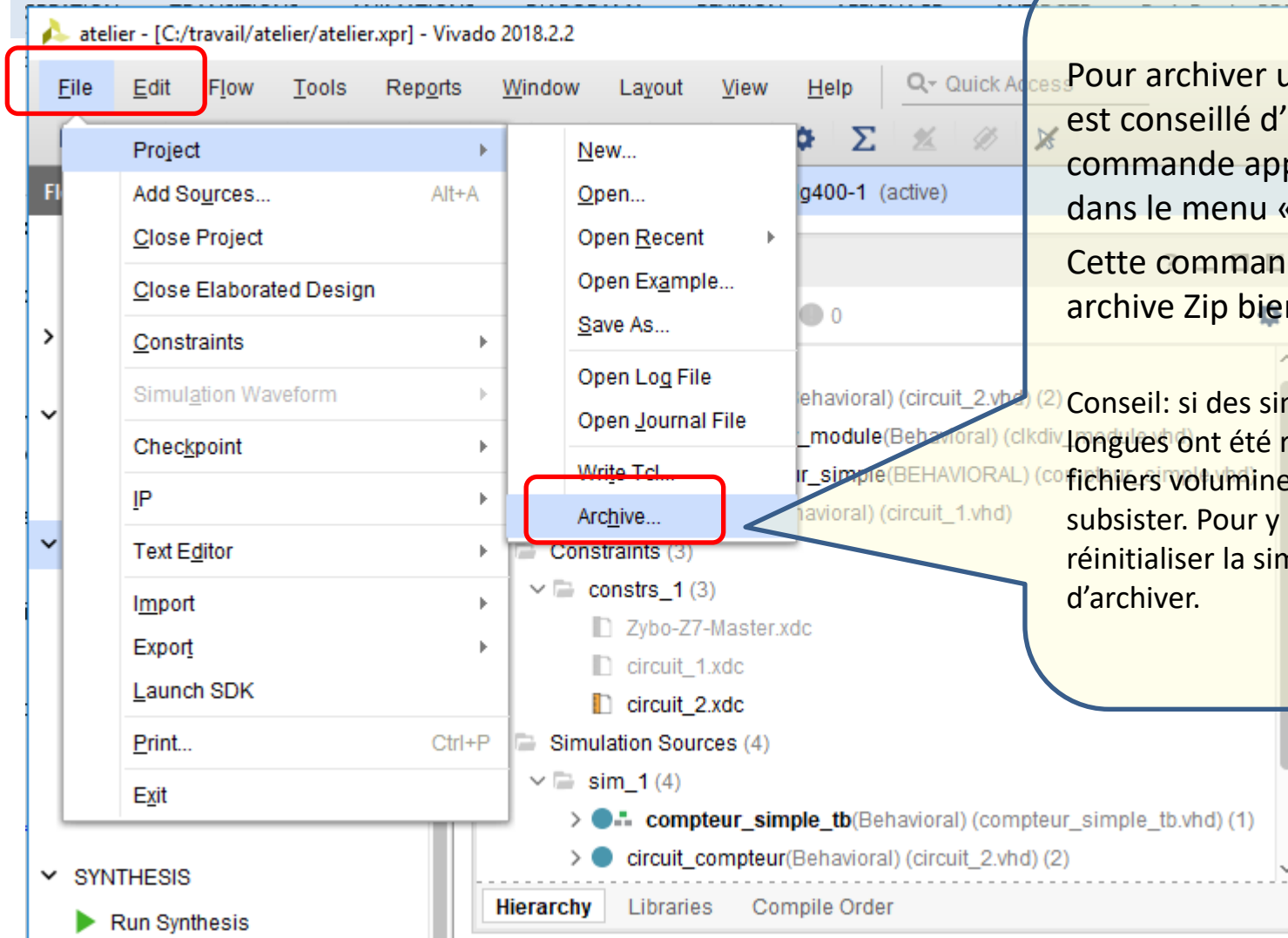
Programmer et tester sur la carte



Il reste à lancer le transfert du fichier « Bitstream » vers le FPGA pour sa configuration.

Vérifier alors le fonctionnement de la carte.

Archiver le dossier de projet Vivado



Pour archiver un projet, il est conseillé d'utiliser la commande appropriée dans le menu « File ».

Cette commande crée une archive Zip bien organisée.

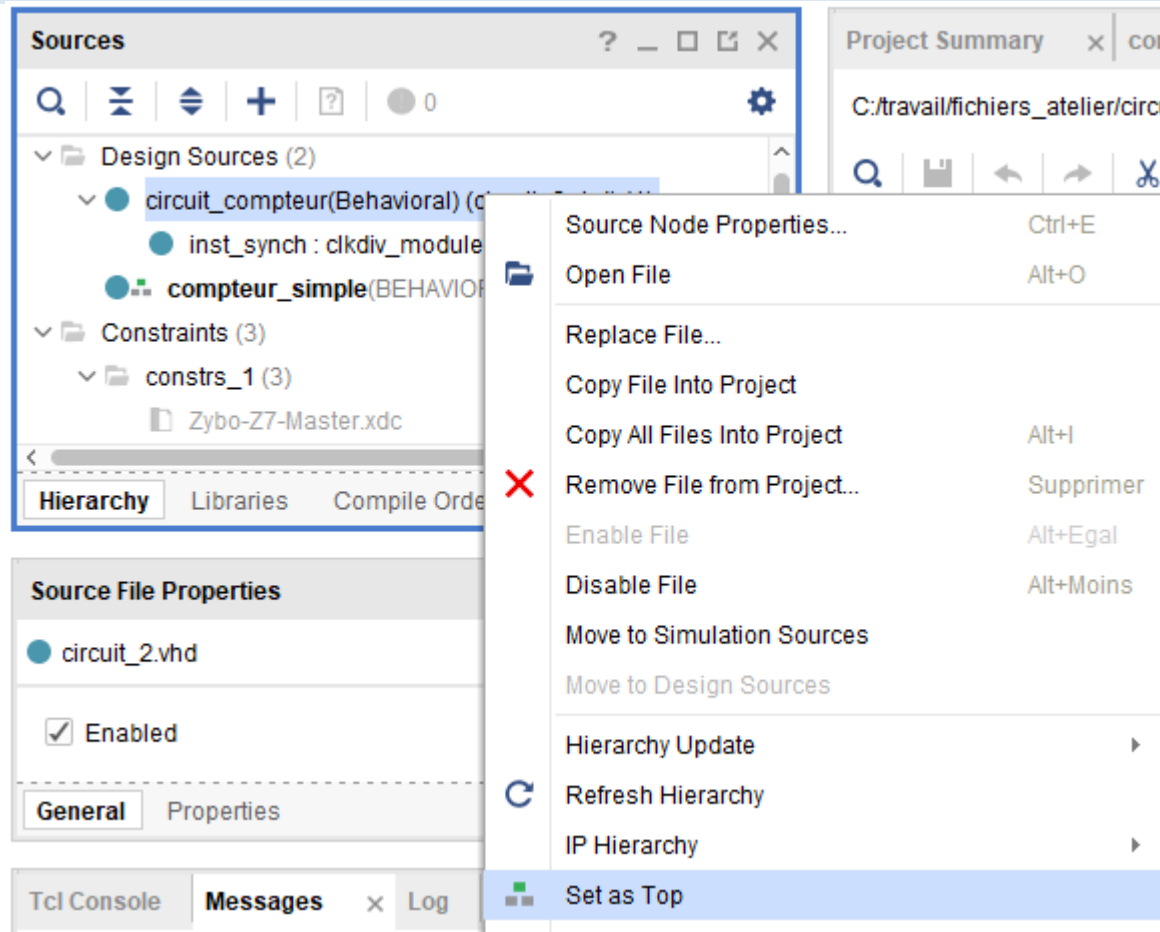
Conseil: si des simulation très longues ont été réalisées. Des fichiers volumineux peuvent subsister. Pour y remédier, réinitialiser la simulation avant d'archiver.

Annexes

Compléments sur l'utilisation de Vivado

- Changer le « top module »
- Configurer la présentation de la fenêtre des synchrogrammes (waveform)
- Nouvelle présentation du chronogramme
- Le bouton « Next Transition » pour parcourir une simulation réalisée est très utile.
- Que faire quand il y des erreurs...

Changer le « top module »



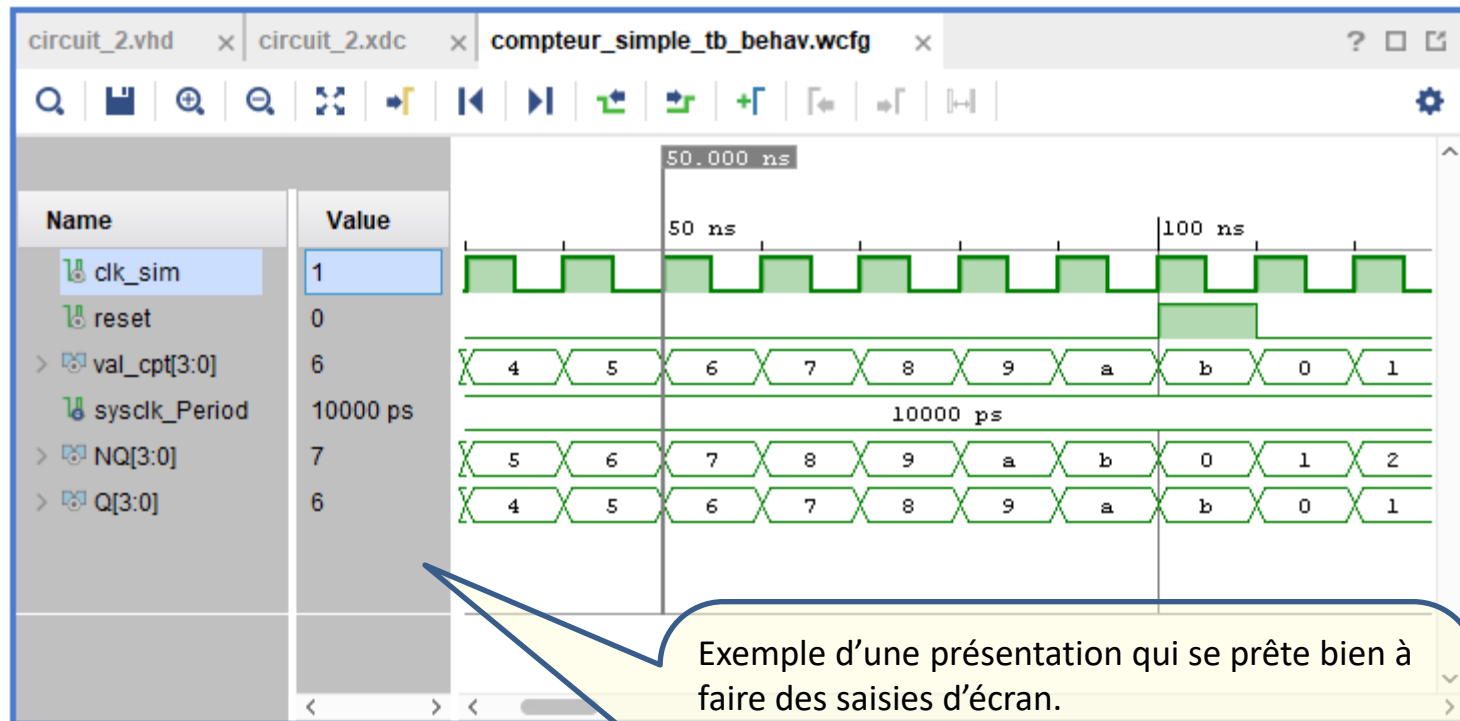
Configurer la présentation de la fenêtre des synchrogrammes (waveform)

Il est possible de modifier la présentation des chronogrammes, ce qui peut être utile pour réaliser des saisies d'écran plus lisibles à insérer dans un document.

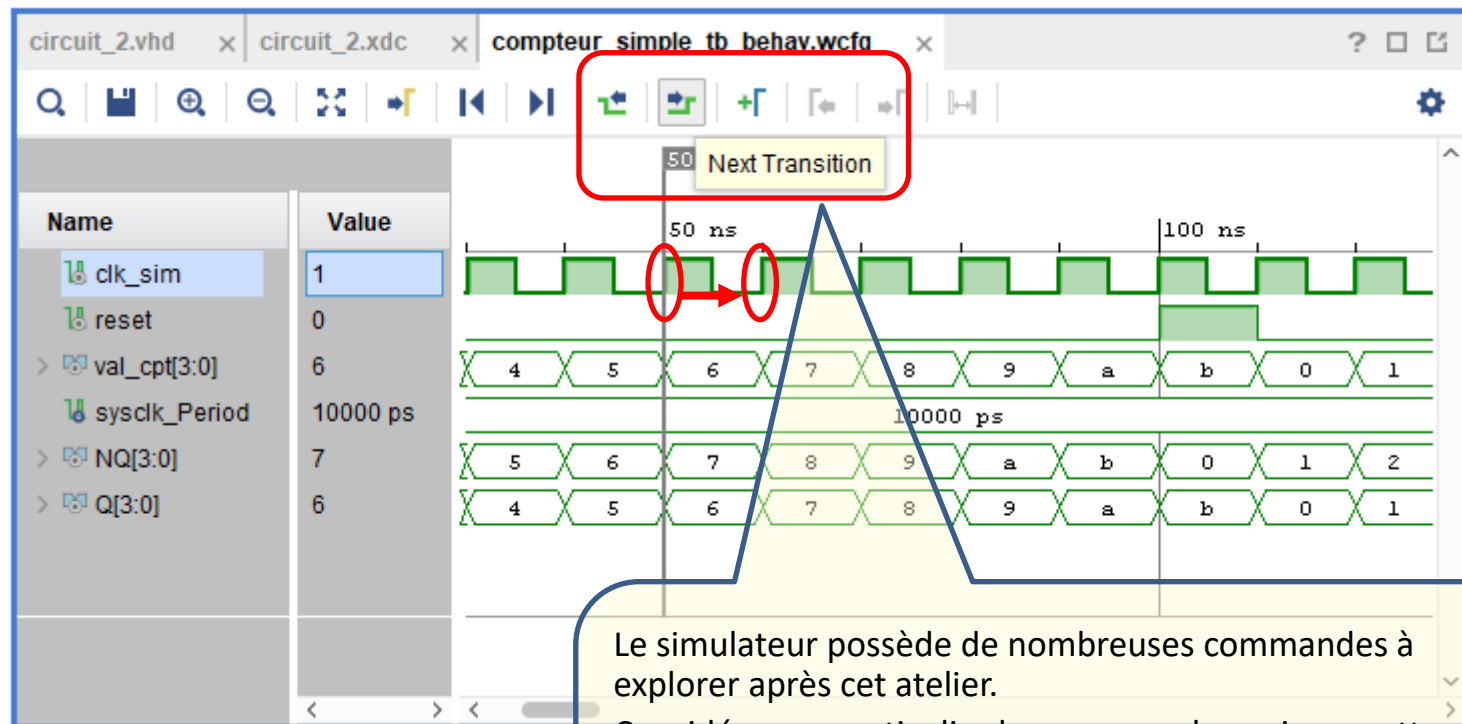
The screenshot shows the Vivado IDE interface. The top tabs include 'circuit_2.vhd', 'circuit_2.xdc', 'compteur_simple_tb.vhd', and 'compteur_simple_tb_behav.wcfg'. The main window displays a waveform for 'clk_sim' (a green square wave) and 'reset' (a single green pulse). Below these, there are two rows of data labeled 'val_cpt[3:0]' and 'sysclk_Period'. The 'val_cpt[3:0]' row shows a sequence of values: 9, a, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b. The 'sysclk_Period' row shows a sequence of values: a, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c. The waveform is displayed on a black background with green text and green outlines for the data values. The 'Colors' panel on the right shows the configuration for the waveform area. The 'Waveform Area Background' is set to black (0, 0, 0). The 'Waveform Area Foreground' is set to white (255, 255, 255). The 'Name/Value Area Background' is set to black (0, 0, 0). The 'Waveform Area Text' is set to white (255, 255, 255). The 'Name/Value Area Text' is set to white (255, 255, 255). The 'Selected Waveform Area Item' is set to white (255, 255, 255). The 'Cursor' is set to yellow (255, 255, 0). The 'Marker' is set to blue (0, 128, 255). The 'Trigger Point' is set to red (255, 0, 0). The 'Default Waveform' is set to green (0, 255, 0). The 'Waveform Text' is set to white (255, 255, 255). The 'Waveform Value Uninitialized' is set to orange (255, 165, 0).

Description	Color
Waveform Area Background	0, 0, 0
Waveform Area Foreground	255, 255, 255
Name/Value Area Background	0, 0, 0
Waveform Area Text	255, 255, 255
Name/Value Area Text	255, 255, 255
Selected Waveform Area Item	255, 255, 255
Cursor	255, 255, 0
Marker	0, 128, 255
Trigger Point	255, 0, 0
Default Waveform	0, 255, 0
Waveform Text	255, 255, 255
Waveform Value Uninitialized	255, 165, 0

Présentation modifiée du chronogramme



Le bouton « Next Transition » pour parcourir une simulation réalisée est très utile.



Le simulateur possède de nombreuses commandes à explorer après cet atelier.

Considérer en particulier les commandes qui permettent de suivre les transitions d'un signal. C'est très utile, par exemple pour suivre des événements marqués par des impulsions isolées. Il faut que le signal soit sélectionné comme l'horloge « clk_sim » dans cette illustration.

Que faire quand il y a des erreurs...

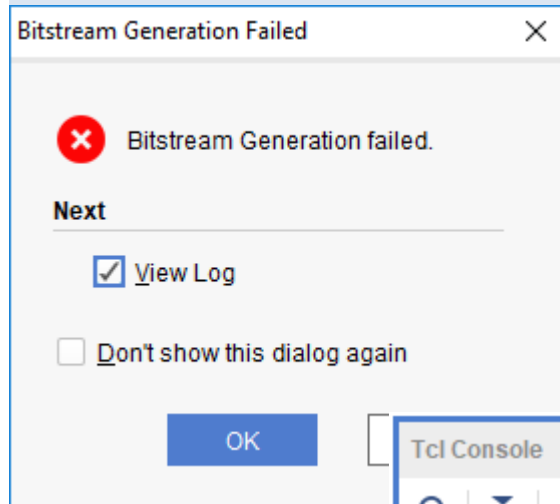
Utiliser les boutons de filtrage des messages pour faire apparaître et sélectionner les messages voulus.

The screenshot shows the Vivado Messages window with the following structure:

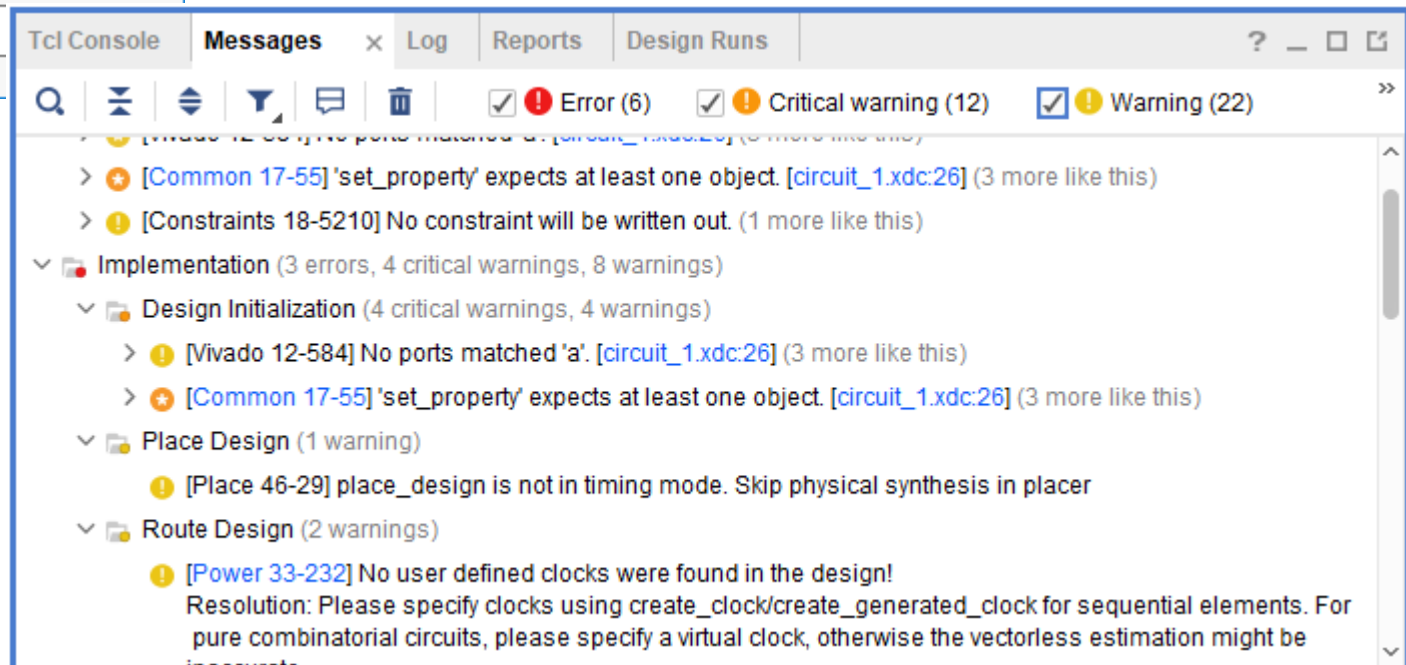
- Messages** (selected tab)
- Filter buttons:** ☒ Error (6), ☒ Critical warning (57), ☐ Warning (67), ☐ Info (183), ☐ Status (401)
- Message List:**
 - ▼ Synthesis (19 critical warnings)
 - > [Common 17-55] 'set_property' expects at least one object. [circuit_1.xdc:26] (17 more like this)
 - [Vivado 12-4739] create_clock:No valid object(s) found for '-objects [get_ports sysclk]'. [circuit_2.xdc:15]
 - ▼ Implementation (3 errors, 19 critical warnings)
 - ▼ Design Initialization (19 critical warnings)
 - > [Common 17-55] 'set_property' expects at least one object. [circuit_1.xdc:26] (17 more like this)
 - [Vivado 12-4739] create_clock:No valid object(s) found for '-objects [get_ports sysclk]'. [circuit_2.xdc:15]
 - ▼ Write Bitstream (3 errors)
 - ▼ DRC (2 errors)
 - ▼ Pin Planning (2 errors)
 - [DRC NSTD-1] Unspecified I/O Standard: 6 out of 6 logical ports use I/O standard (IOSTANDARD) value 'DEFAULT', instead of a user assigned specific value. This may cause I/O contention or incompatibility with the board power or connectivity affecting performance, signal integrity or in extreme cases cause damage to the device or the components to which it is connected. To correct this violation, specify all I/O standards. This

Dans la situation présente, les fichiers de contraintes (étape 1) ne sont pas désignés « enable » correctement dans la hiérarchie.

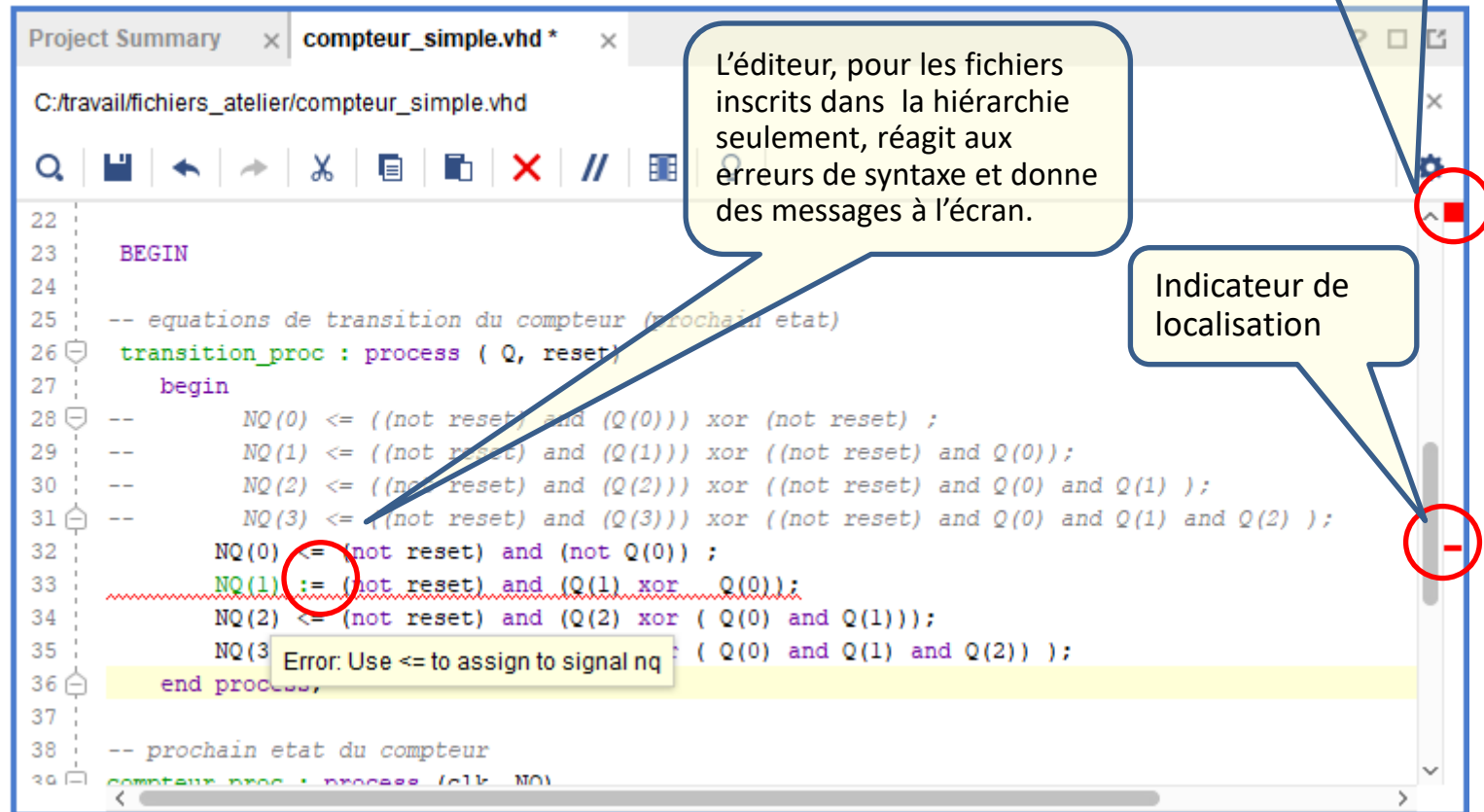
Que faire quand il y des erreurs...



Le fichier de contraintes est inadéquat.



Que faire quand il y a des erreurs...



FIN DE L'ATELIER



Références

Carte Zybo-Z7

- <https://reference.digilentinc.com/reference/programmable-logic/zybo-z7/reference-manual>
- <https://reference.digilentinc.com/reference/programmable-logic/zybo-z7/start>

Vivado[®]

- Xilinx[®]: Vivado Design Suite **User Guide**: Using the Vivado IDE UG893 (v2018.2) June 6, 2018
- Vidéo illustrant comment réaliser un circuit additionneur identique à celui de la partie 1 <https://www.youtube.com/watch?v=aPDT0sPr4jE>

Circuit Zynq[®]

- Xilinx[®]: Zynq-7000 SoC, Technical Reference Manual, UG585 (v1.12.2) July 1, 2018