

Questions sur le compteur 16 bits synchrone :

- Que cherchez-vous à implémenter ?
- Que veut dire synchrone ? Front montant ? Front descendant ? Niveau ?
- Combien a-t-il d'entrées ? De sorties ? Citez-les, indiquer leur rôle, leur type et leur fonctionnement.
- Classez les signaux d'entrée selon leur priorité.

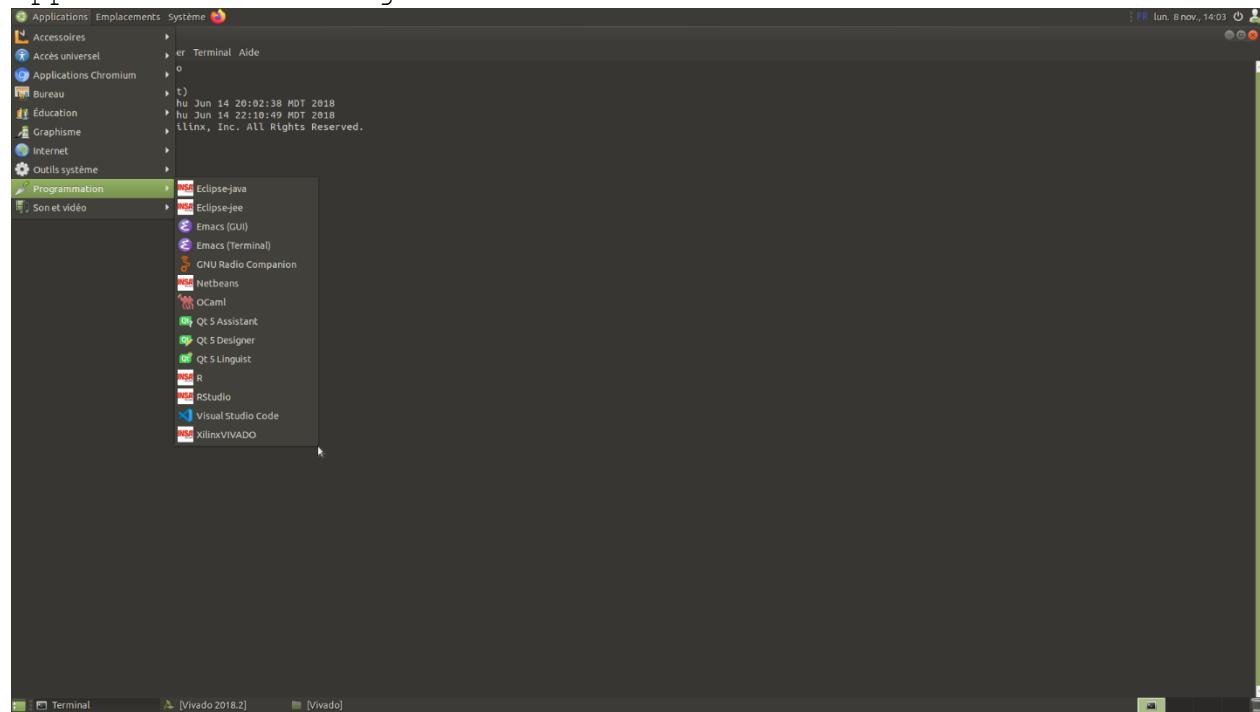
Sommaire

Démarrer VIVADO.
Créer un projet.
Créer une source d'implémentation.
Organisation de VIVADO.
Vérifier la syntaxe.
Synthétiser.
Afficher le rapport de synthèse.
Estimer la fréquence maximale de l'implémentation.
Afficher les circuits générés.
Ajouter des librairies.
Placer et router.
Observer le placement et routage sur le FPGA.
Créer une source de test.
Simuler après synthèse ou après placement et routage.

Démarrer VIVADO

- *Environnement Linux INSA*

Applications -> Programmation -> XilinxVIVADO



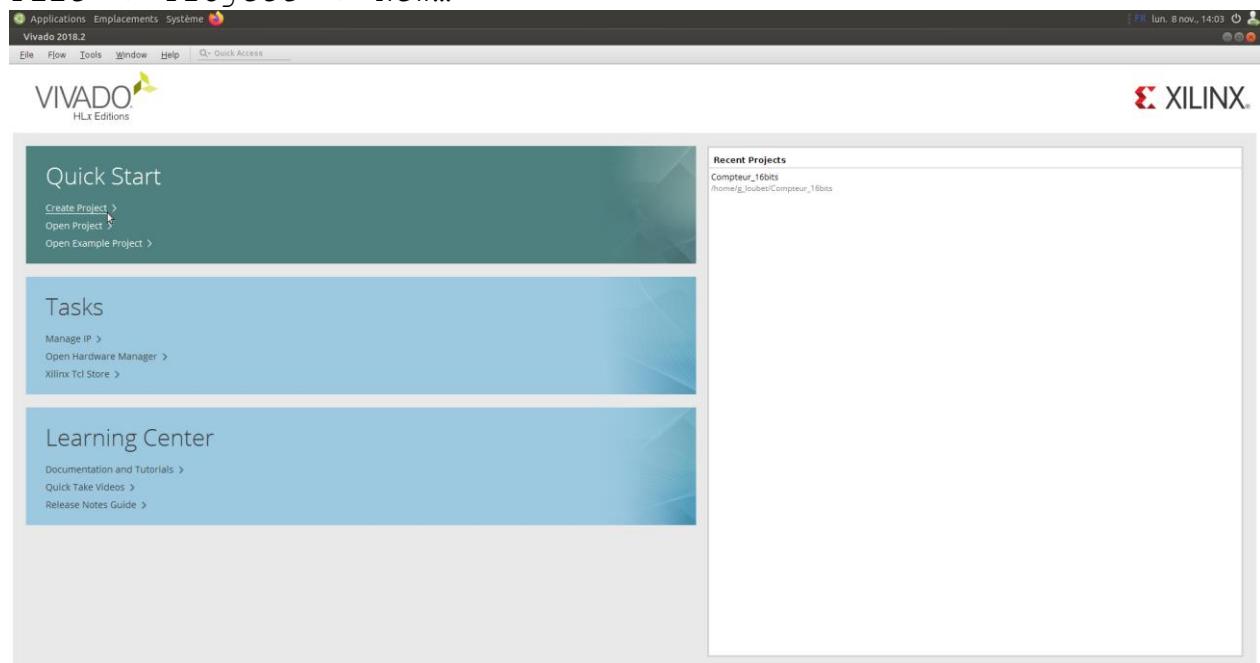
ou

Depuis un terminal (Ctrl+Alt+t) -> "vivado"



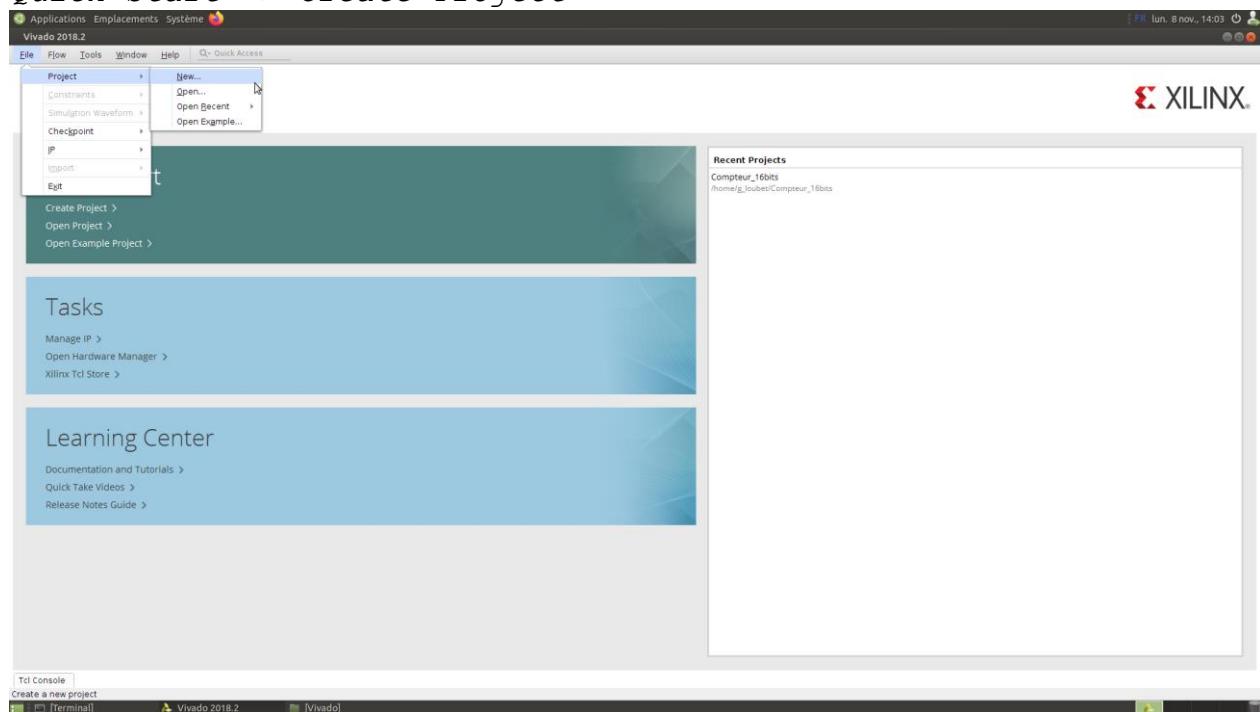
Créer un projet

File -> Project -> New...

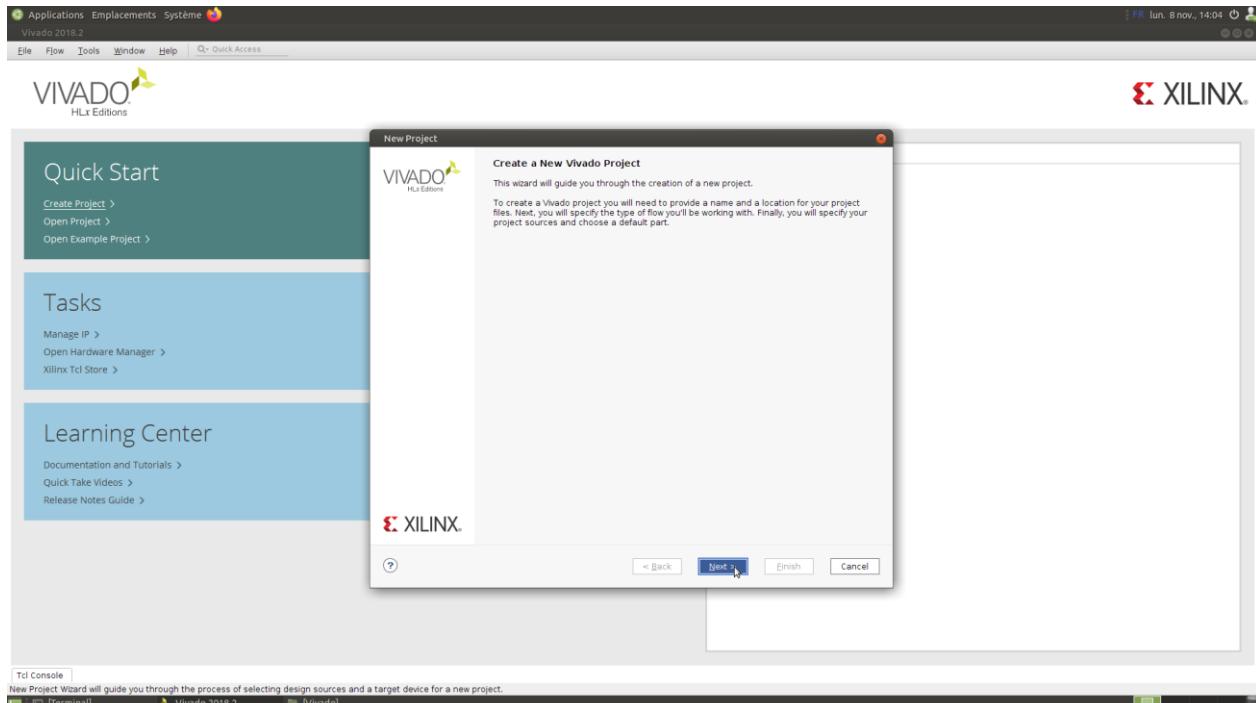


ou

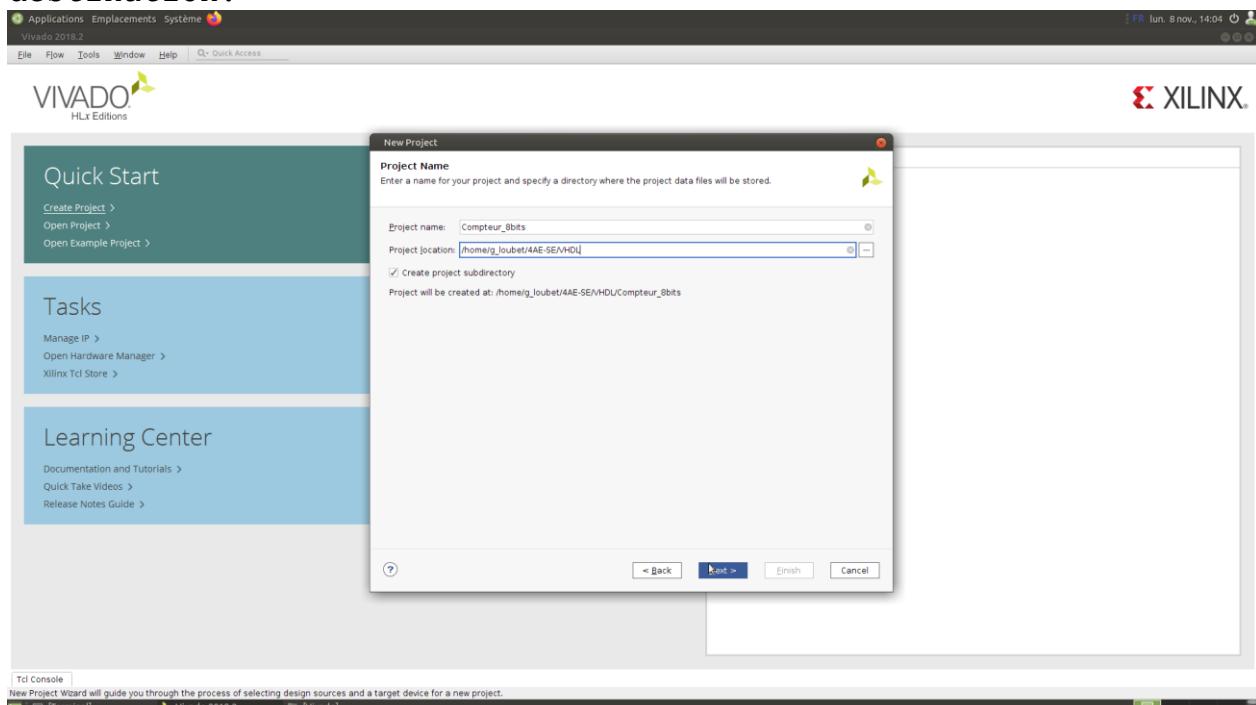
Quick Start -> Create Project



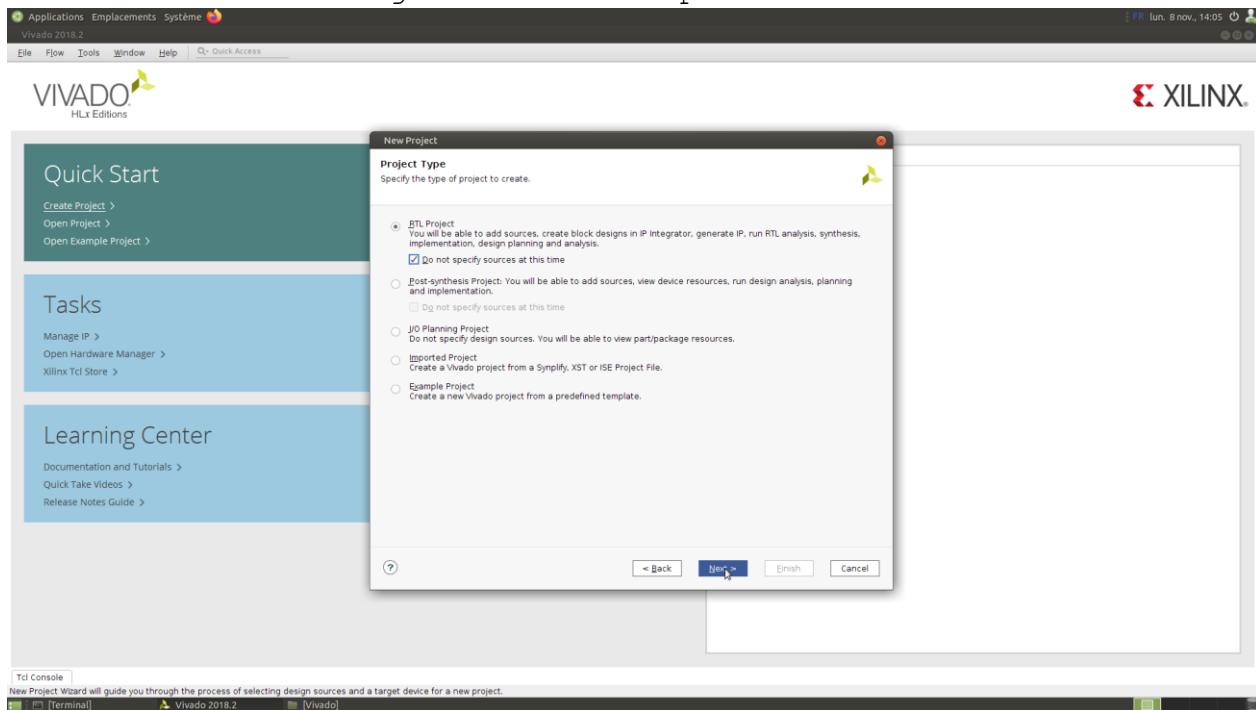
Puis Next



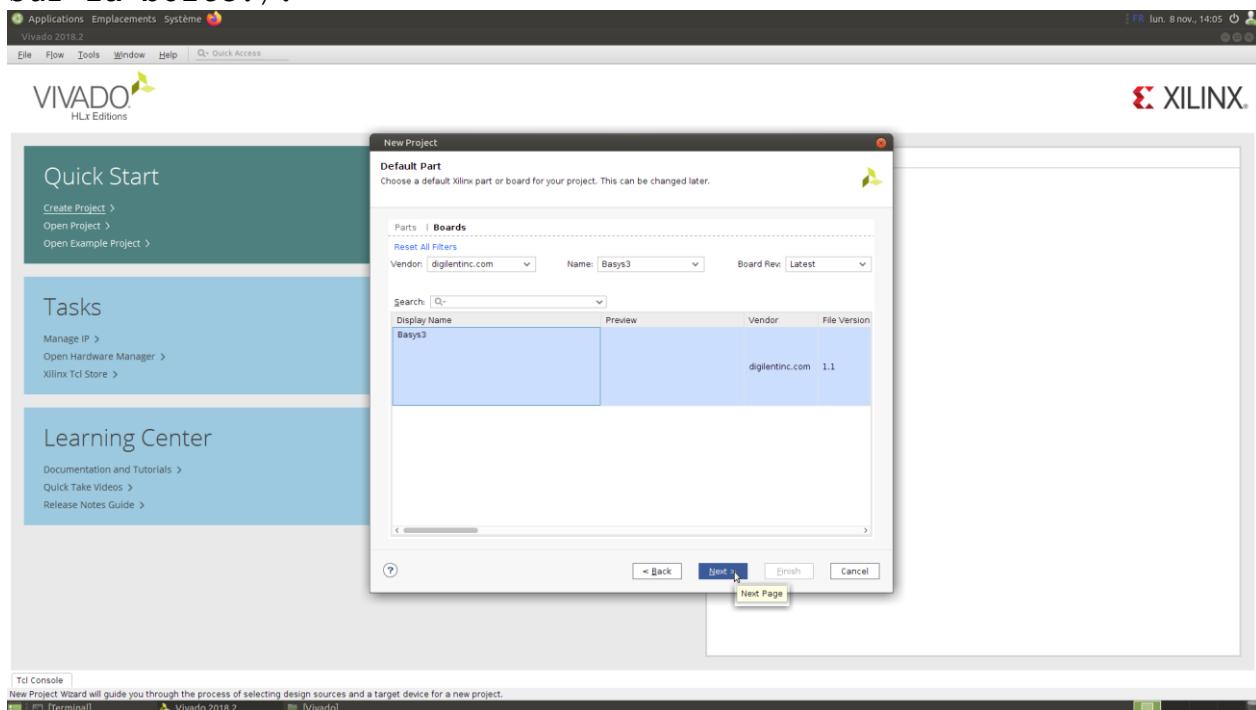
Nommez-le. (e.g. Compteur_16bits) et vérifiez le dossier de destination.



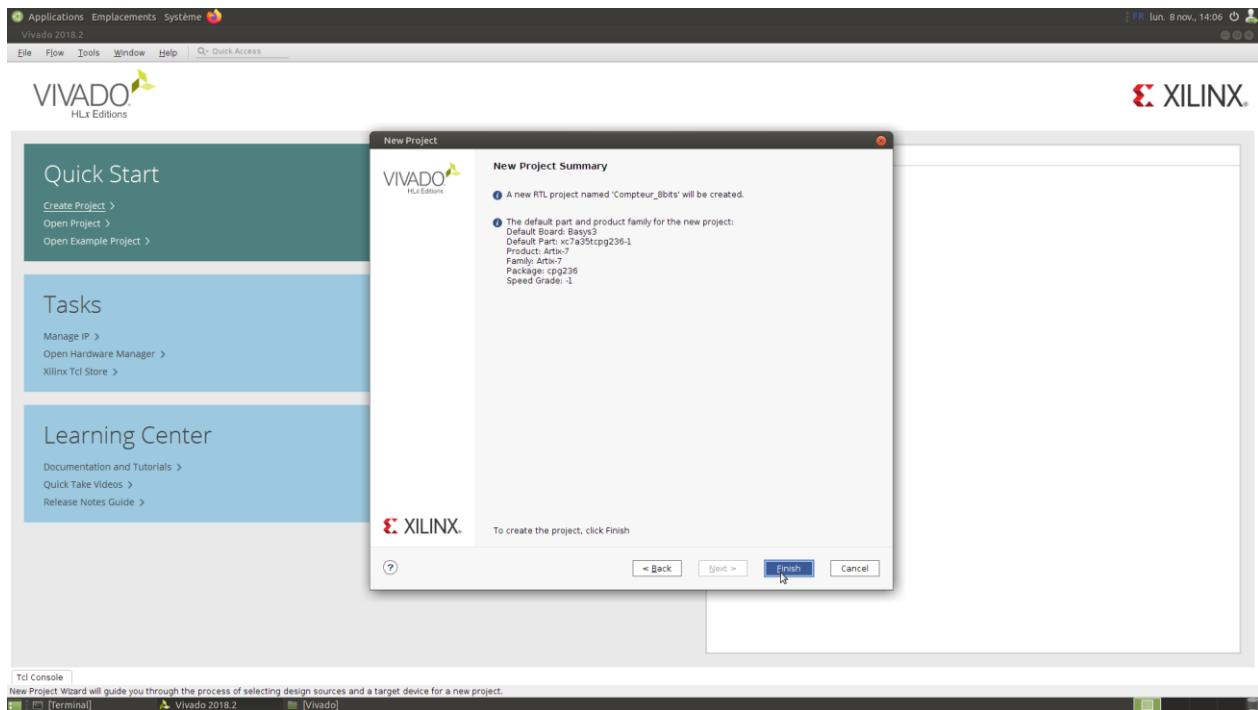
Sélectionnez *RTL Project*. Ne créez pas encore de sources.



Sélectionnez la carte ciblée (*Boards*, *Vendor: digilentinc.com*, *Name: Basys3*, *Product: Artix-7*. N.B. : Informations disponibles sur la boîte.).

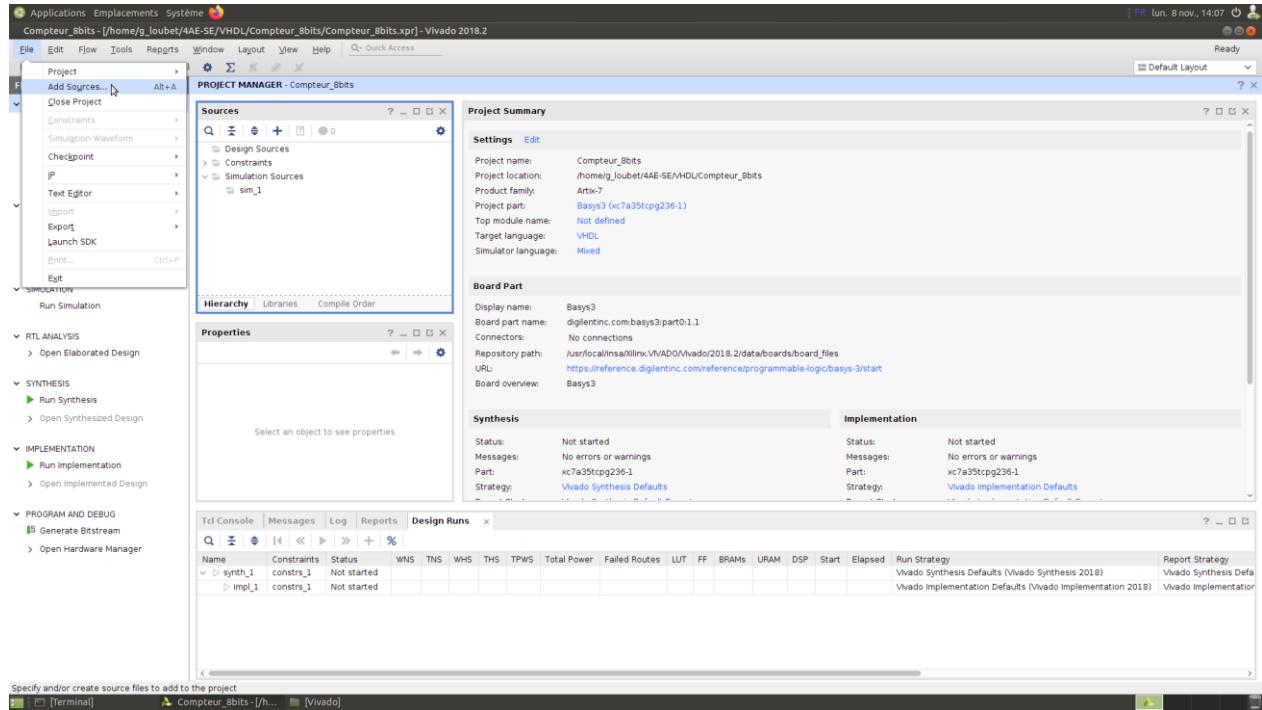


Vérifiez et validez.



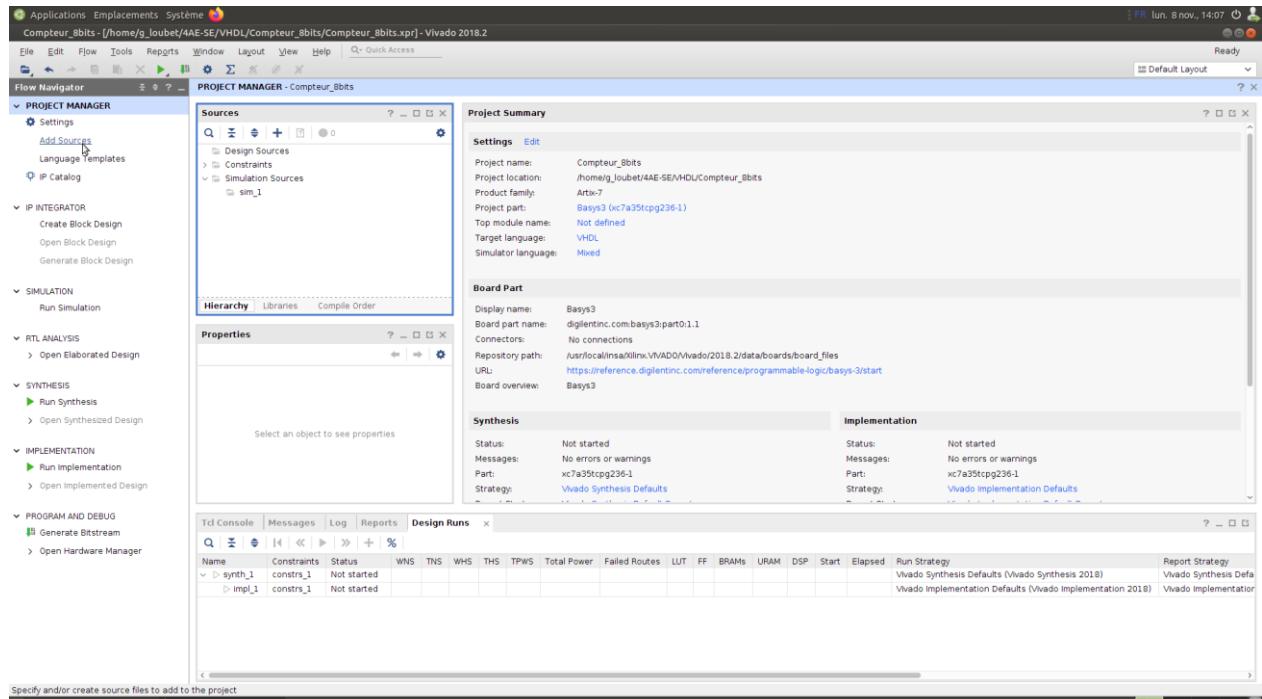
Créer une source d'implémentation

File -> Add Sources...



OU

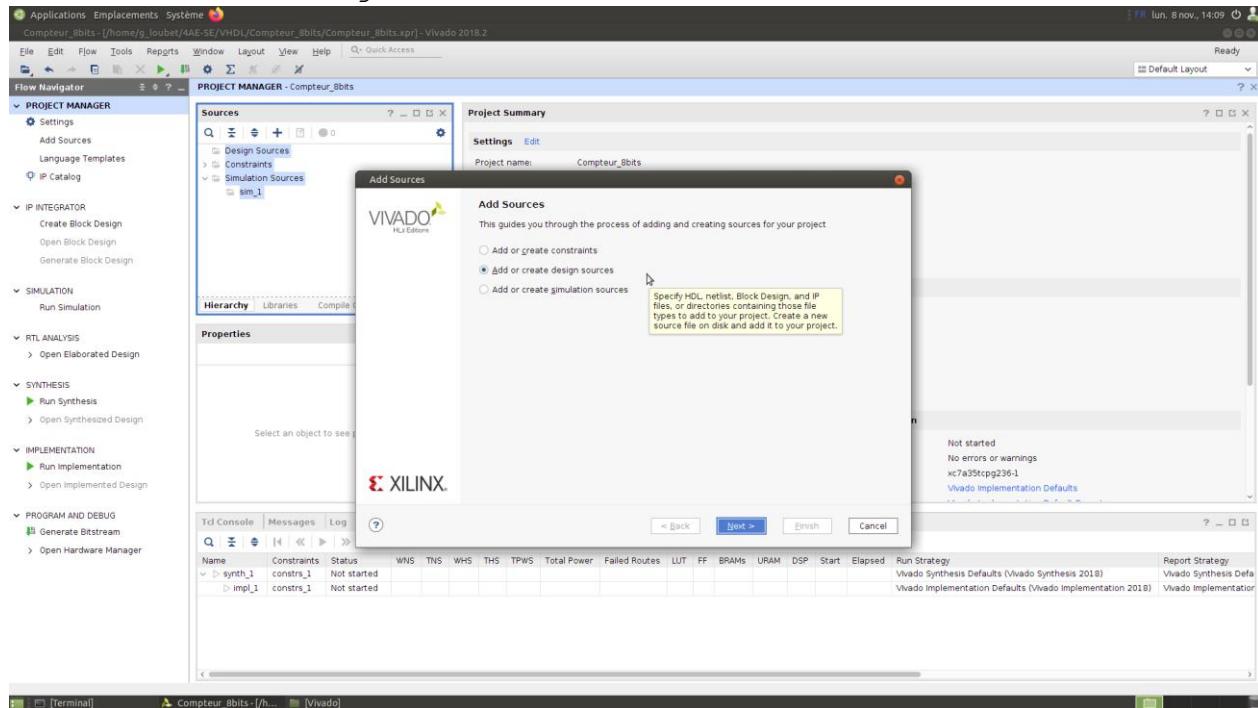
PROJECT MANAGER -> Add Sources



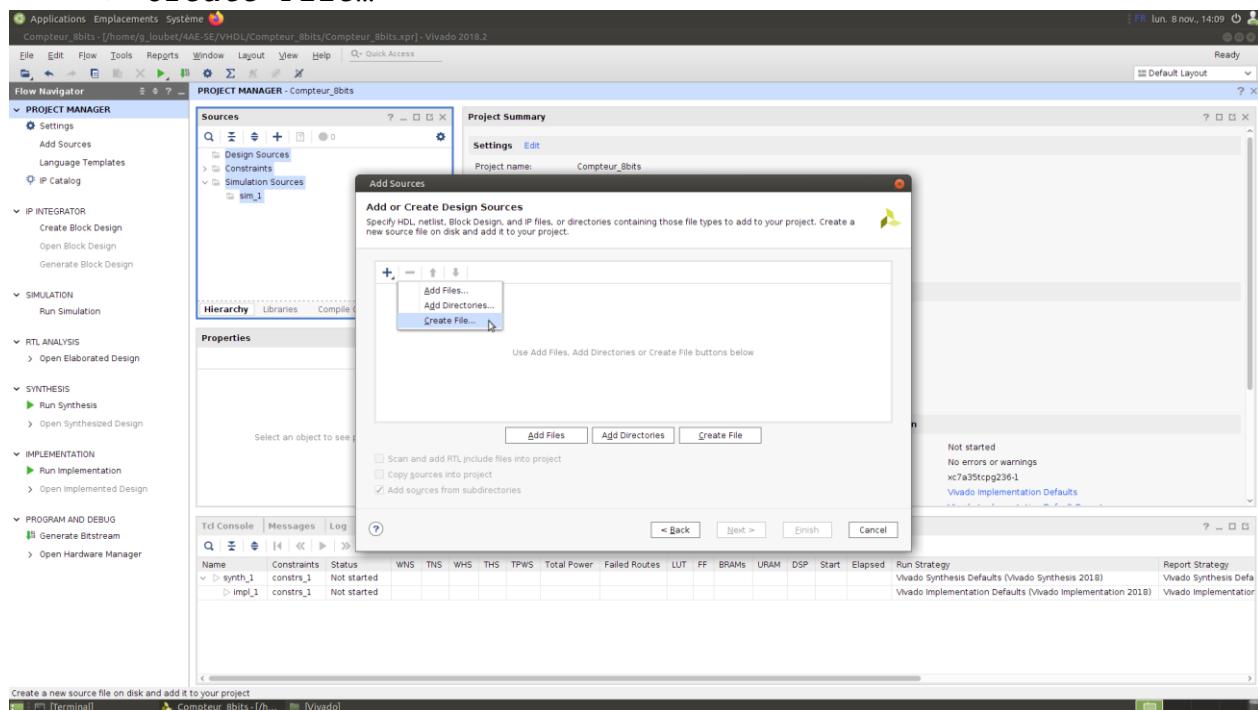
OU

Alt + A

Add or create design sources

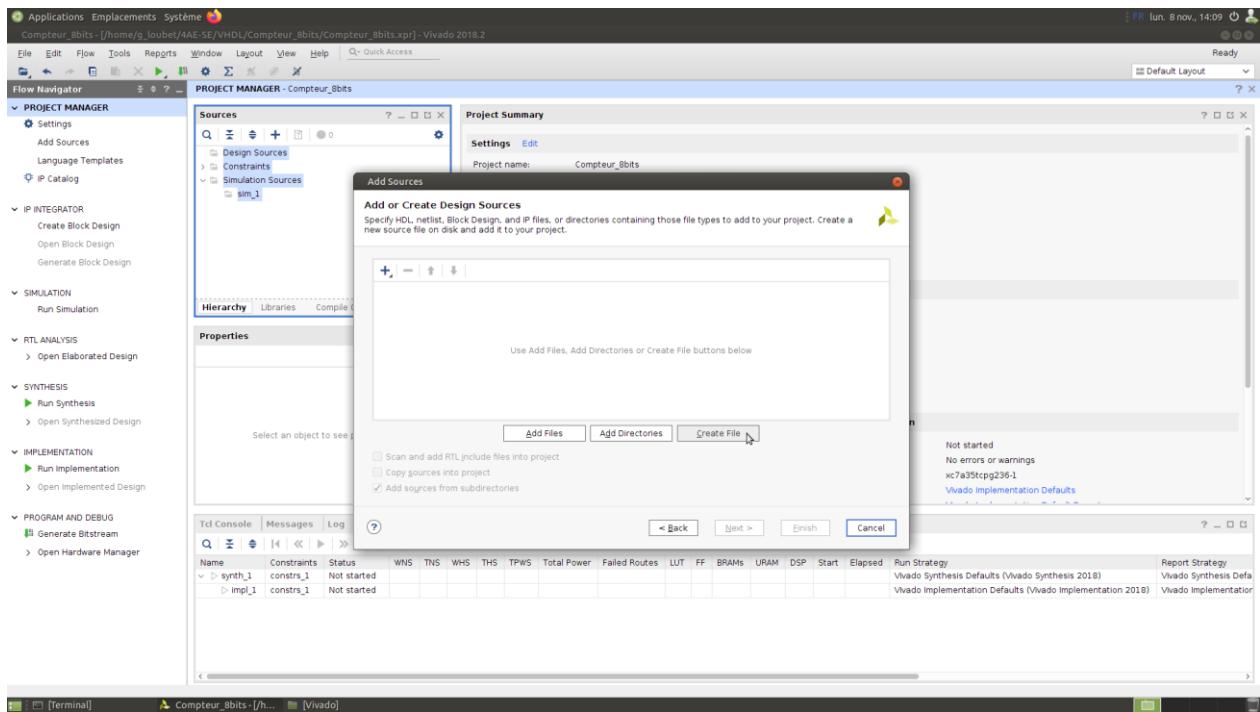


'+' -> Create File...



ou

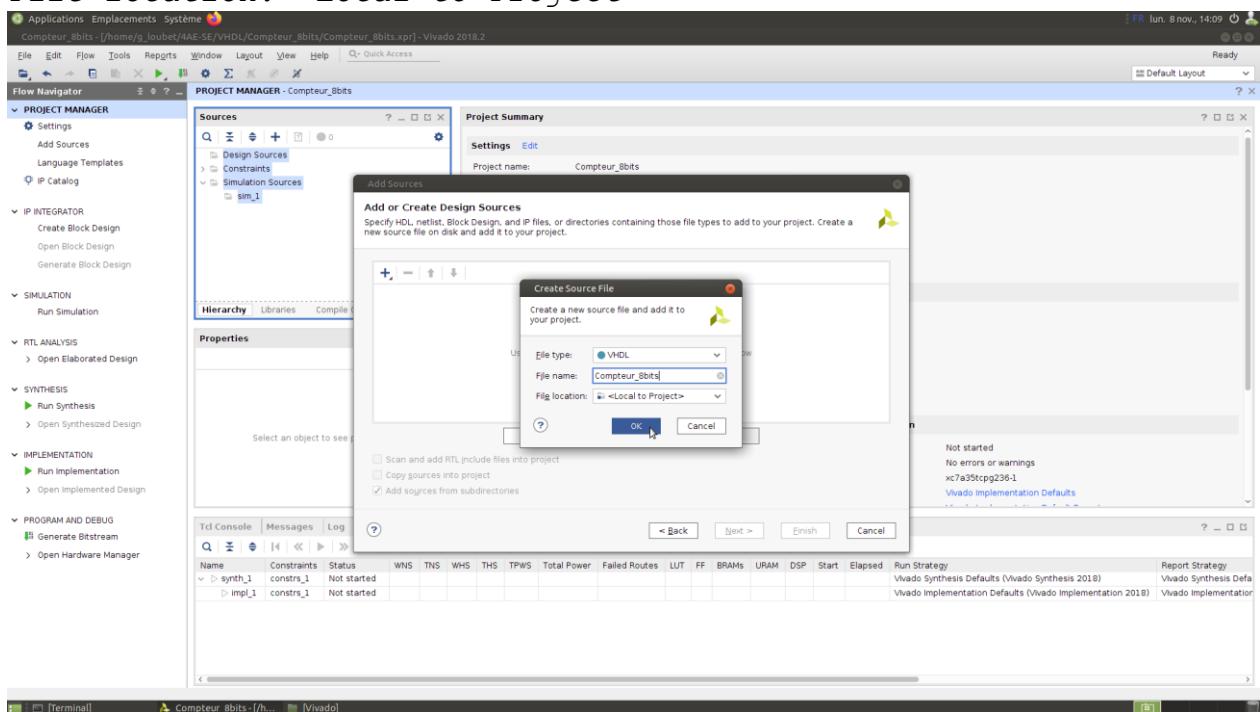
Create File



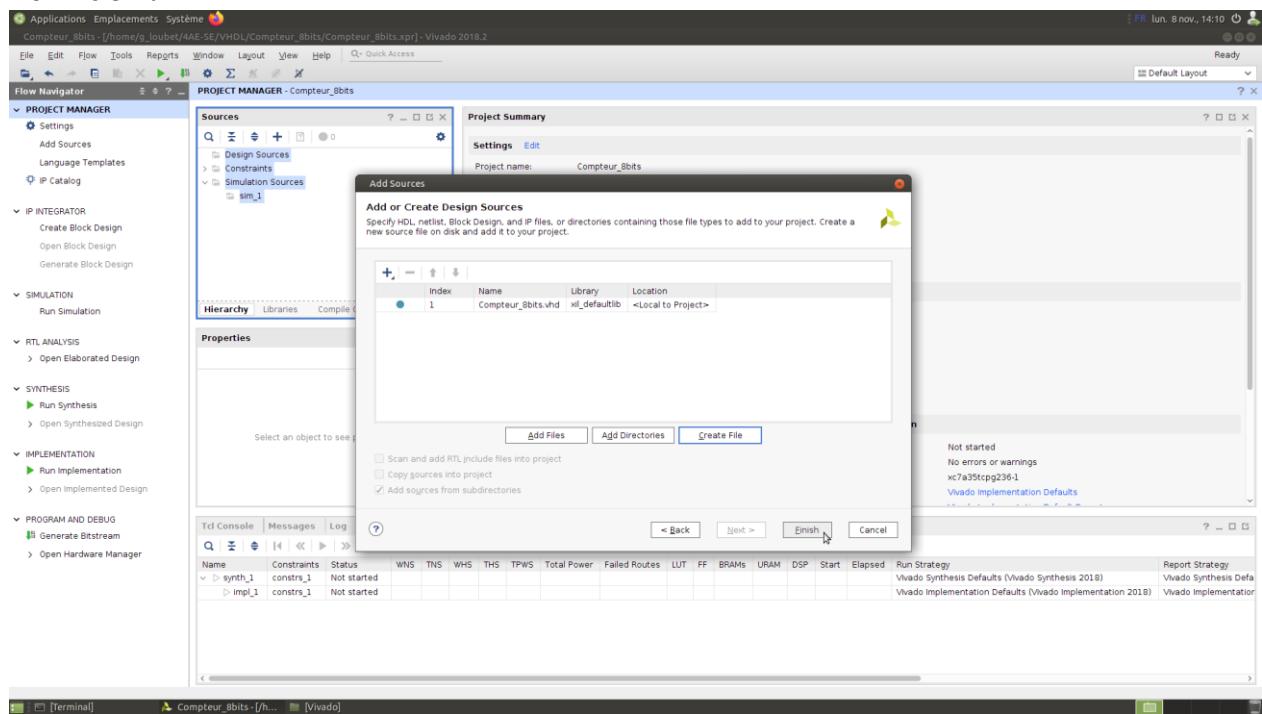
File type: VHDL

File name: Nommez votre fichier (e.g. Compteur)

File location: <Local to Project>



Validez.



Définissez l'entité, les entrées et les sorties.

Entity name: Nommez votre entité (e.g. le même que le fichier)

Architecture name: Behavioral (on va définir une architecture comportementale...).

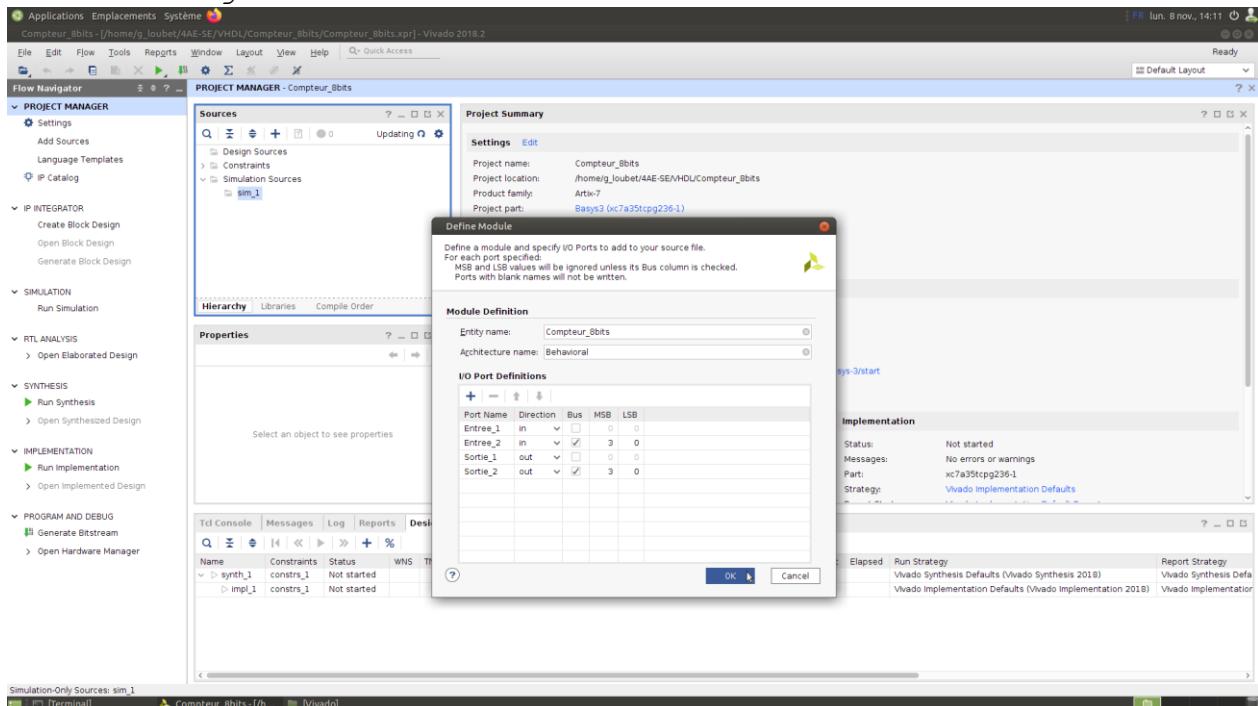
Port Name: Nom de l'entrée ou de la sortie

Direction: Entrée ou sortie (jamais de inout. Pourquoi ?)

Bus: Si ce sont des vecteurs binaires.

MSB: Most Significant Bit

LSB: Low Significant Bit

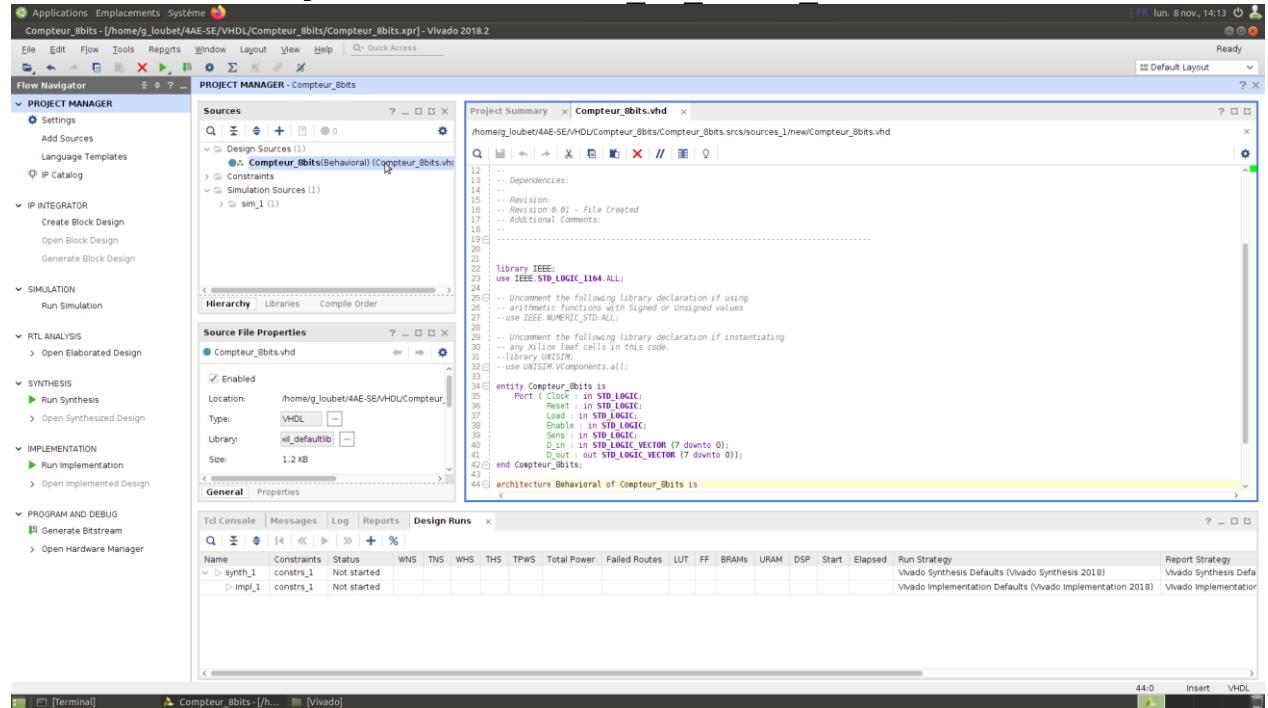


Vérifiez et validez.

! Ne vous trompez pas. En cas d'oubli et de modification, il vous faudra peut-être recréer un fichier et non pas seulement modifier le fichier généré (VIVADO crée des dépendances qu'il ne met pas forcément à jour quand on modifie les entrées et sorties (ajout ou même modification du nom...)).

Vous pouvez ouvrir votre fichier généré et commencez votre implémentation.

Sources -> Design Sources -> Nom de votre fichier.vhd



Organisation de VIVADO

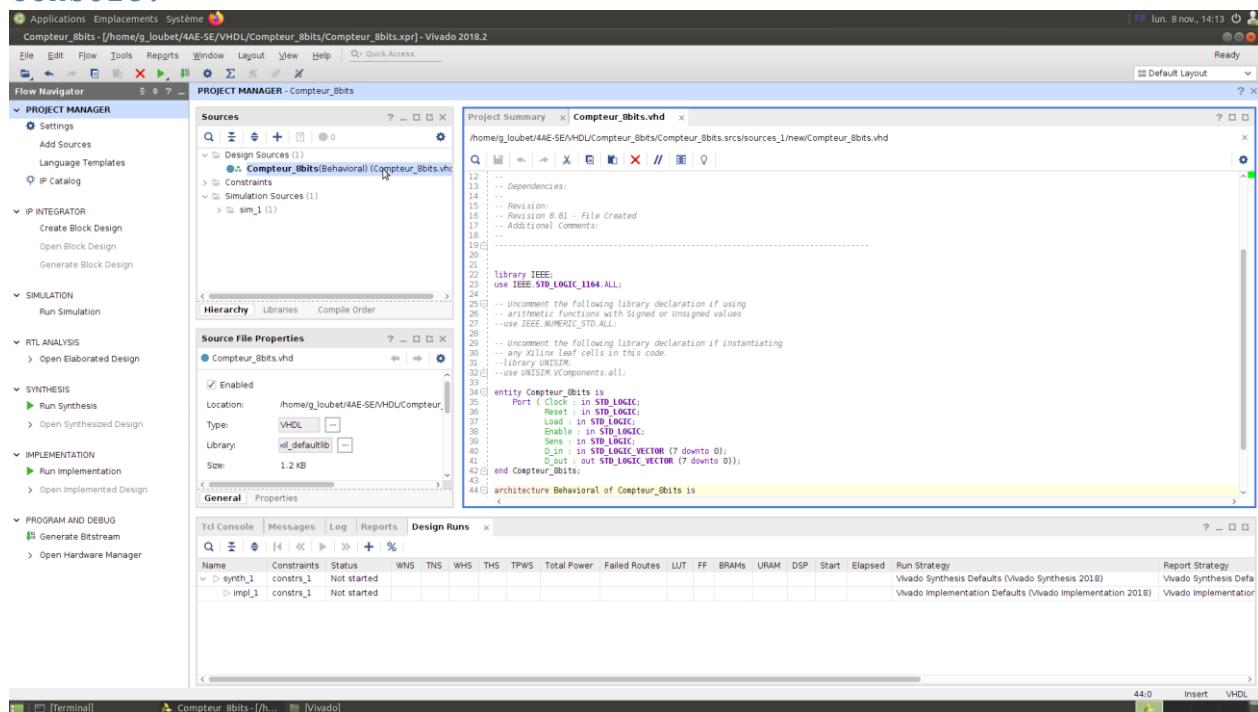
Editeur de texte

Arborescence du projet (*Sources*)

Source sélectionnée (*Source File Properties*)

Flux de conception (*PROJECT MANAGER*)

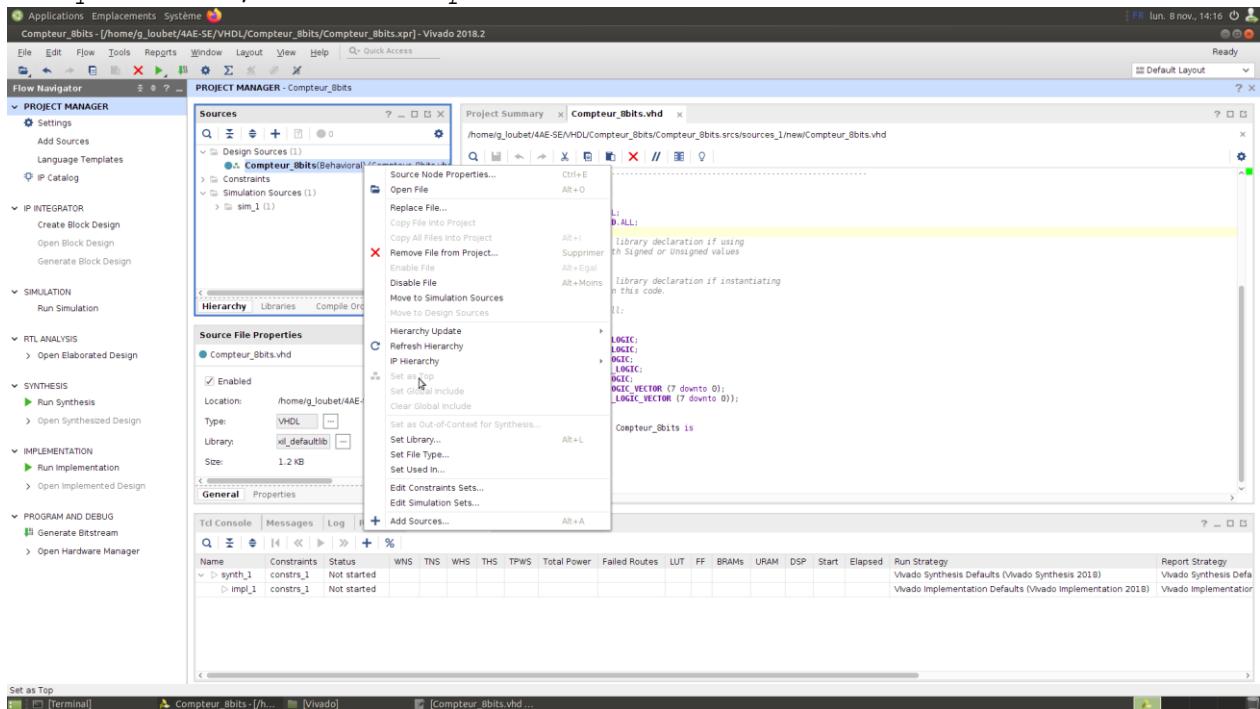
Console.



Vérifier la syntaxe

- Vérifiez que la source sur laquelle vous travaillez est celle sélectionnée comme *Top Module* :

Dans l'arborescence du projet sélectionner la source en question, cliquez droit, *Set as Top*.



- La syntaxe est automatiquement vérifier et les erreurs sont soulignées en rouge.

PROJECT MANAGER - Compteur_8bits

Sources

Design Sources (1)

Compteur_8bits(Behavioral) (Compteur_8bits.vhd)

Constraints

Simulation Sources (1)

sim_1 (1)

Hierarchy Libraries Compile Order

Source File Properties

Compteur_8bits.vhd

Enabled

Location: /home/g_loubet/AAE-SE/VHDL/Compteur_8bits

Type: VHDL

Library: xil_defaultlib

Size: 1.2 kB

General Properties

Design Runs

Name	Constraints	Status	WNS	TNS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy	Report Strategy
synth_1	constraints_1	Not started														Vivado Synthesis Defaults (Vivado Synthesis 2018)	Vivado Synthesis Defaults (Vivado Implementation 2018)
impl_1	constraints_1	Not started														Vivado Implementation Defaults (Vivado Implementation 2018)	Vivado Implementation

Terminal [Compteur_8bits - [h...]] [Vivado] [Compteur_8bits.vhd ...]

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.STD.TEXT.TEXT_UNSIGNED.ALL;
-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code
--use UNISIM.VComponents.all;
use UNISIM.VComponents.all;

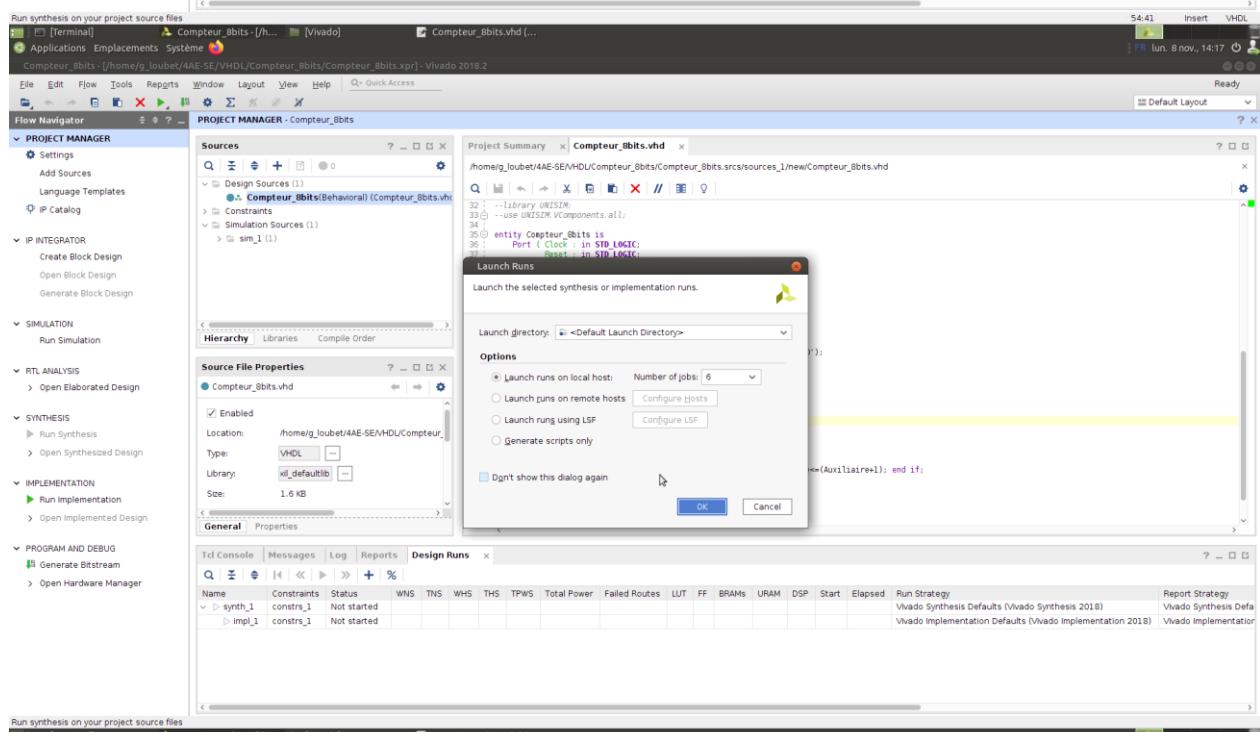
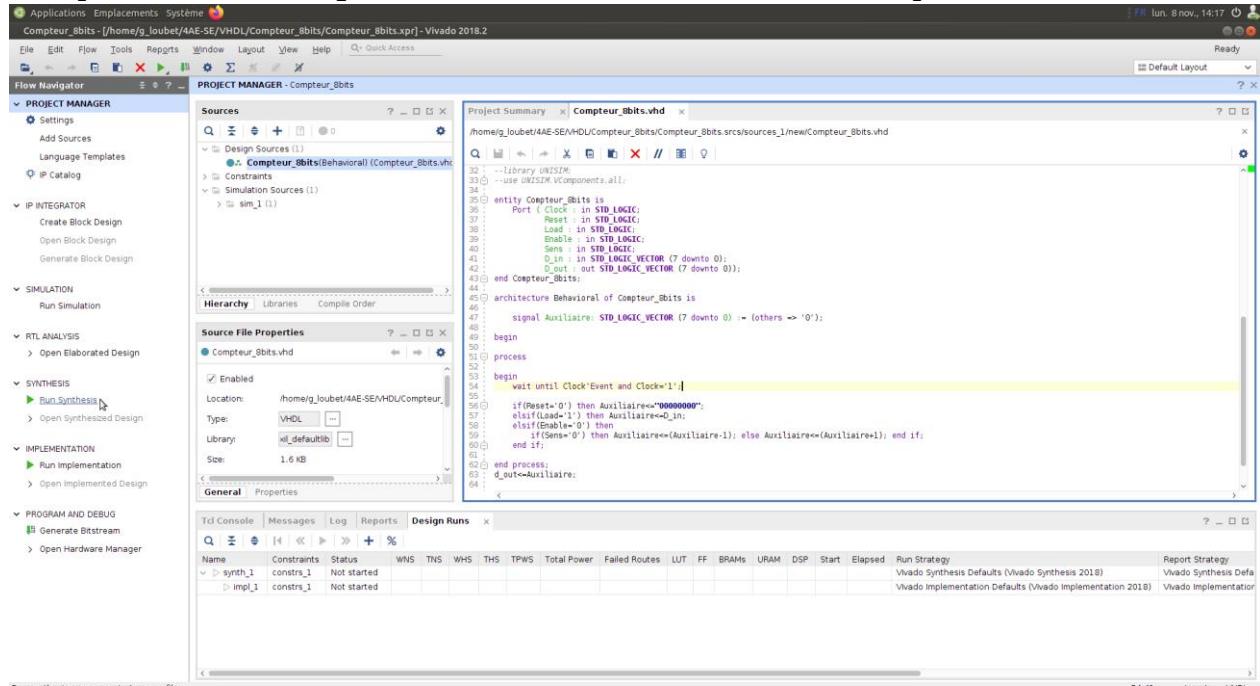
entity Compteur_8bits is
    Port ( Clock : in STD_LOGIC;
           Reset : in STD_LOGIC;
           Load : in STD_LOGIC;
           Enable : in STD_LOGIC;
           Sens : in STD_LOGIC;
           D_in : in STD_LOGIC_VECTOR (7 downto 0);
           Q_out : out STD_LOGIC_VECTOR (7 downto 0));
end Compteur_8bits;

architecture Behavioral of Compteur_8bits is
begin
    begin
        if Comp then
            Error: Syntax error near "if".
        end if;
    end begin;
end Behavioral;

```

Synthétiser

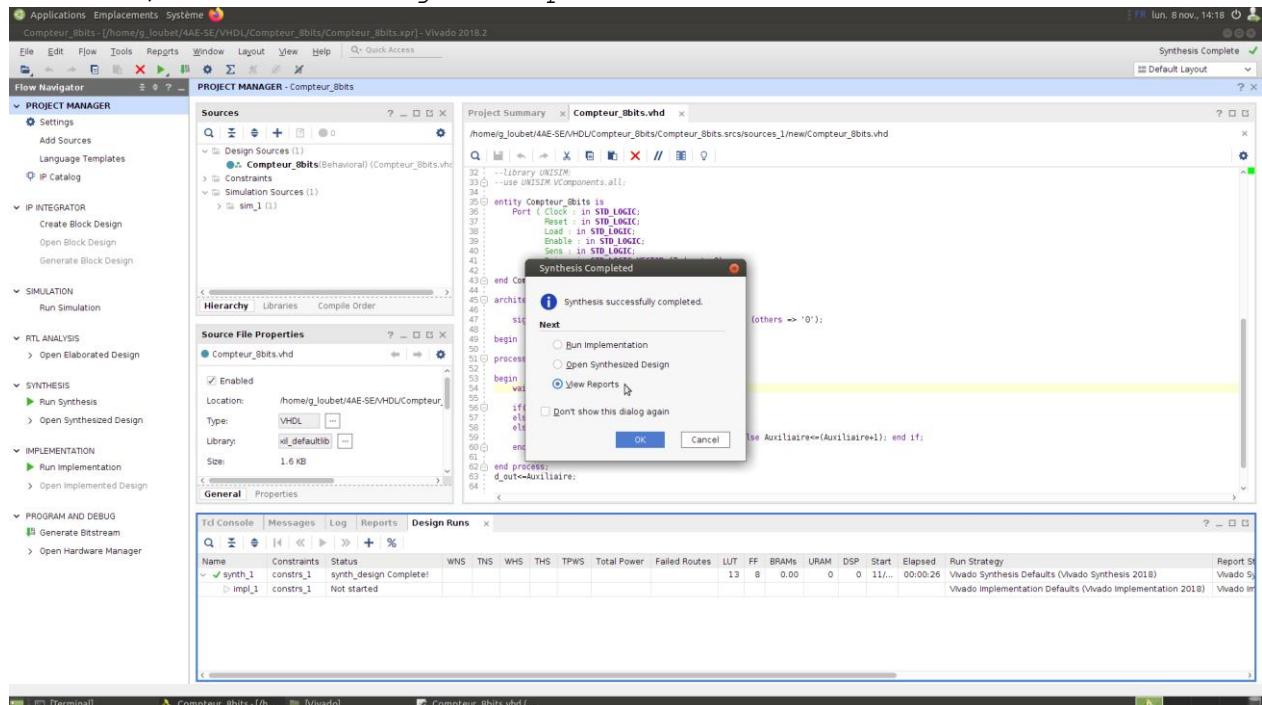
Cliquez sur *Run Synthesis* dans le flux de conception.



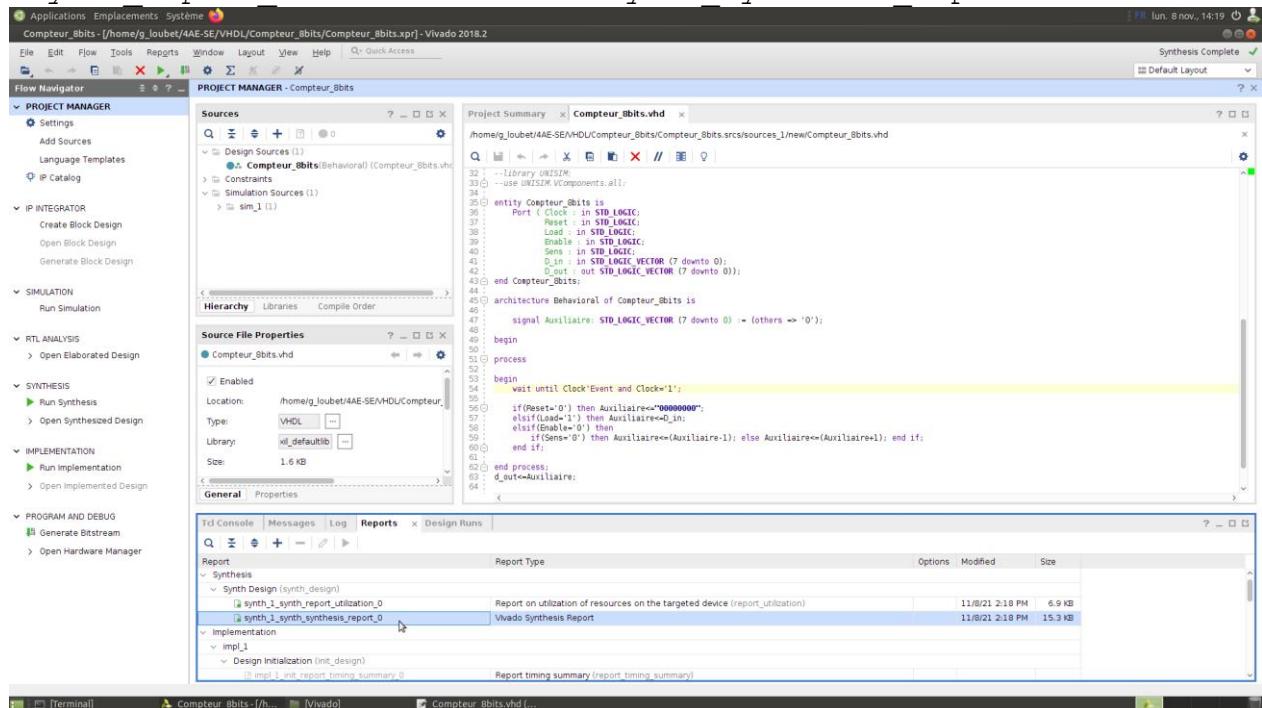
NB : Le check vert indique que c'est bon, la croix rouge qu'il y a des erreurs et le point d'interrogation orange qu'il y a des warnings. Penser à regarder la console, partie *Messages* !

Afficher le rapport de synthèse

Après la synthèse, sélectionnez *View Reports*, puis dans la partie console, allez à l'onglet *Reports*.



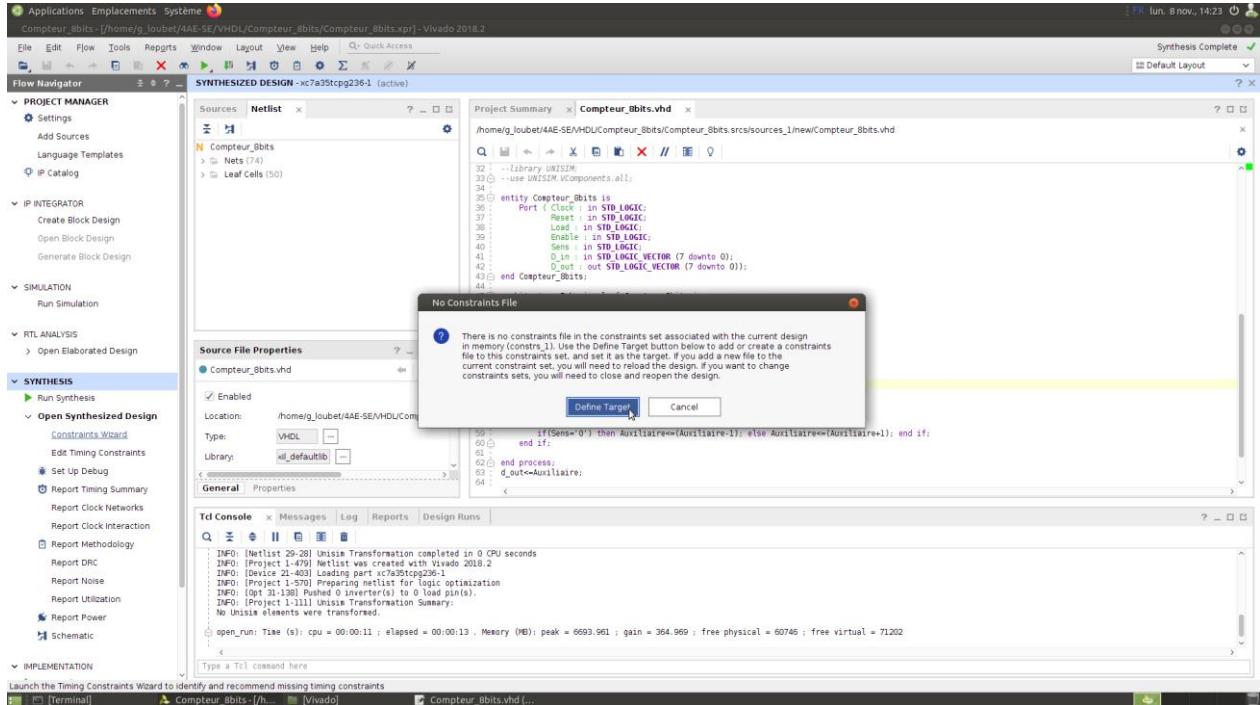
Double-cliquez sur le rapport d'intérêt :
...synth_report_utilization... ou ...synth_synthesis_report....



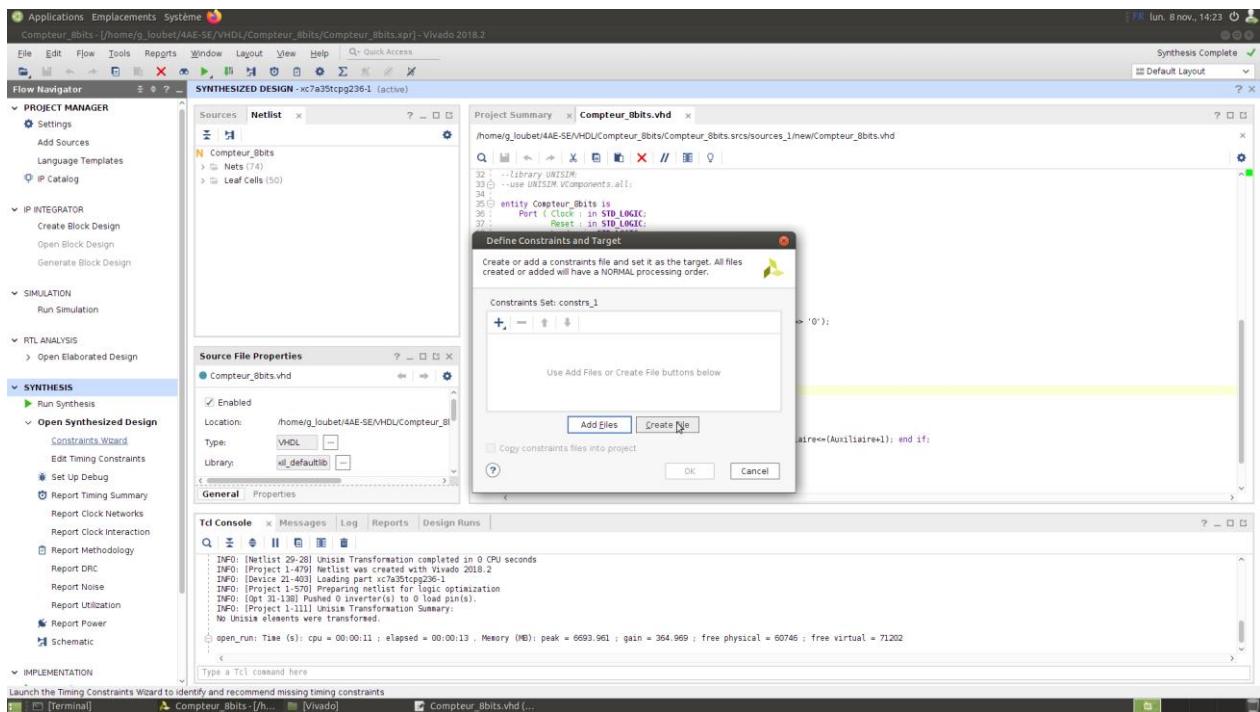
Dans le rapport de synthèse, vous trouverez certaines données pertinentes (e.g. nombre de bascules flip-flop, etc.)

Estimer la fréquence maximale de l'implémentation

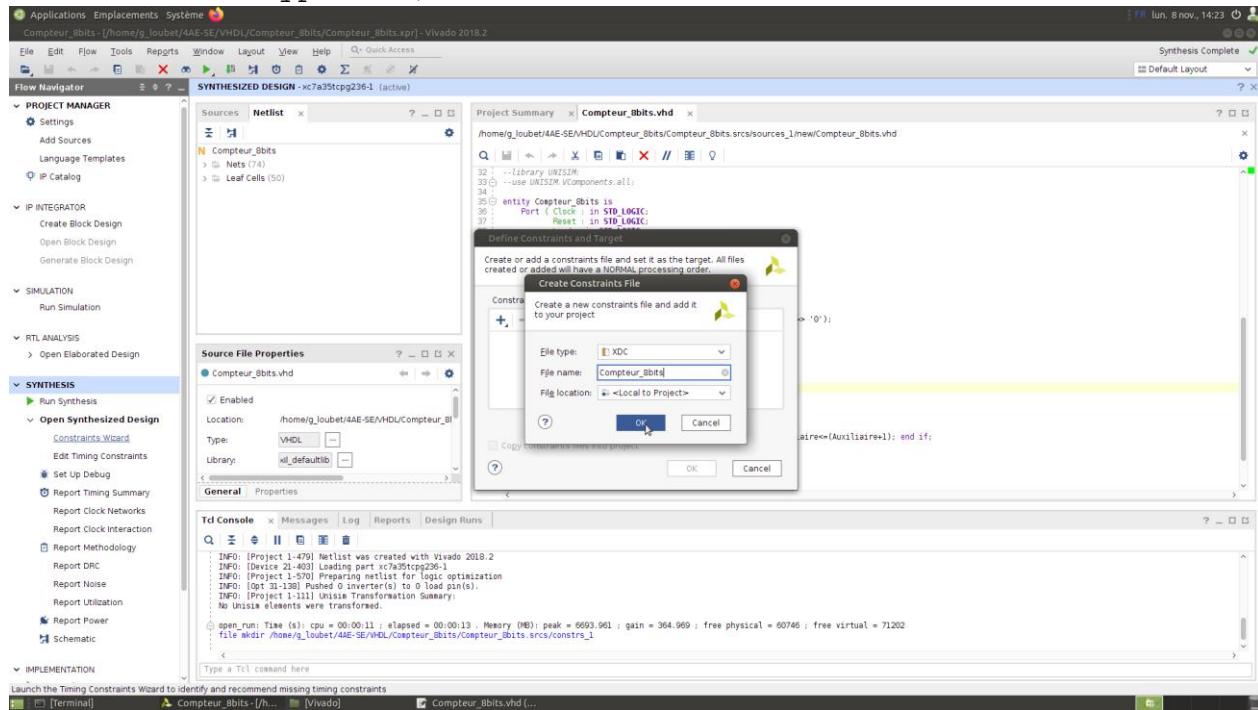
*SYNTHESIS -> Open Synthesized Design -> Constraints Wizard
Puis Define Target*



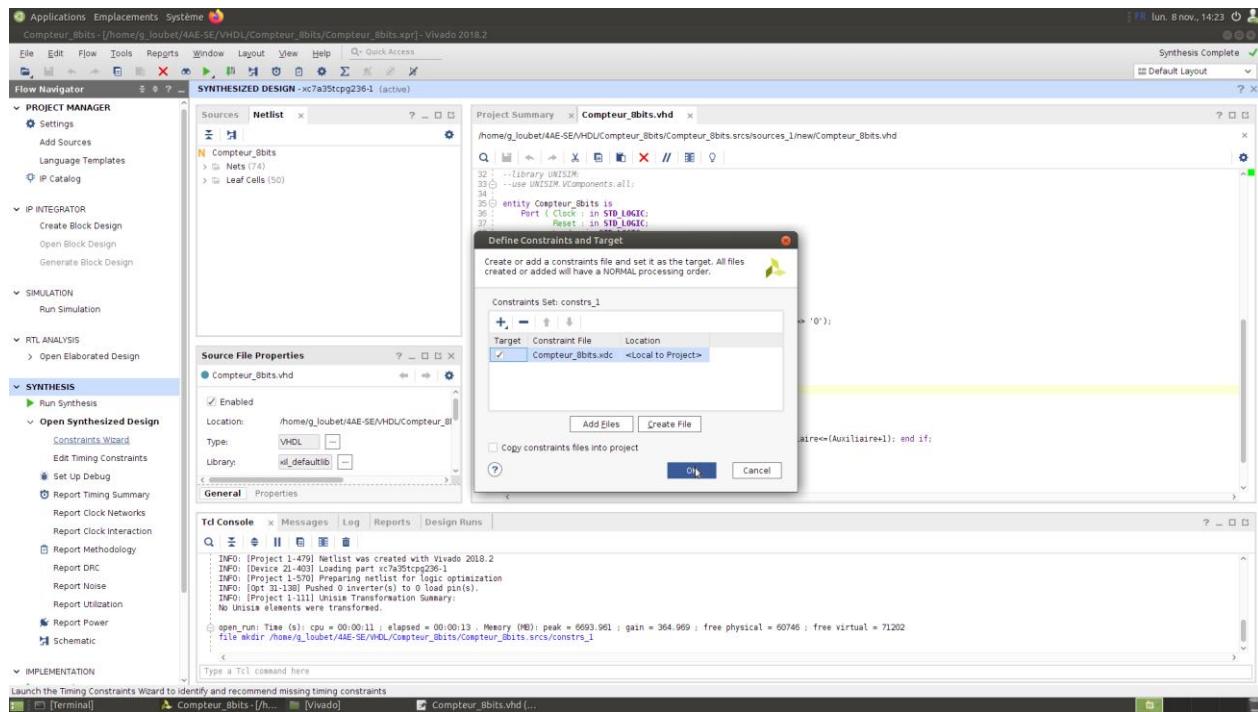
Créez un fichier.



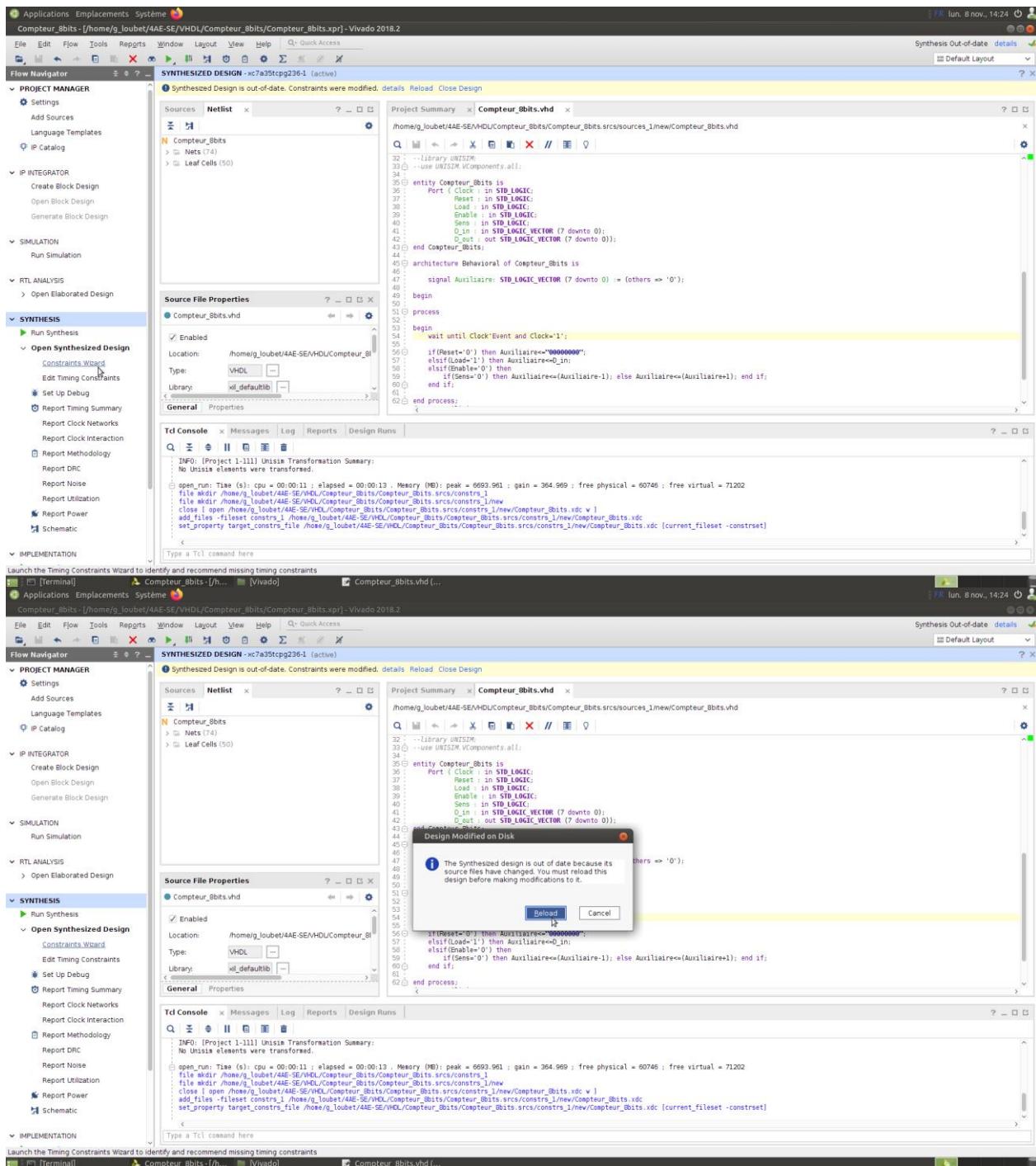
Conservez le type XDC, nommez-le et validez.



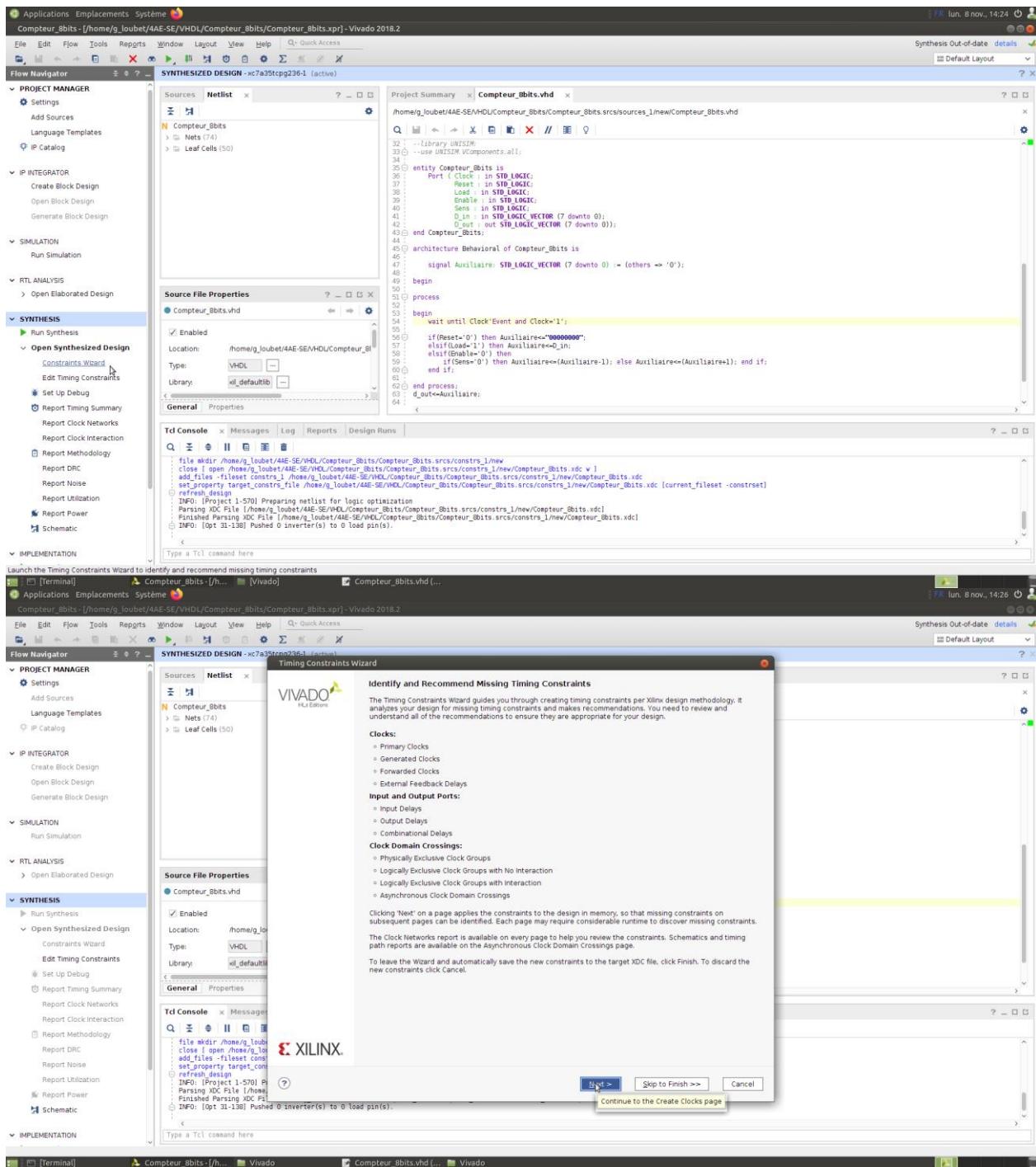
Selectionnez-le et validez.



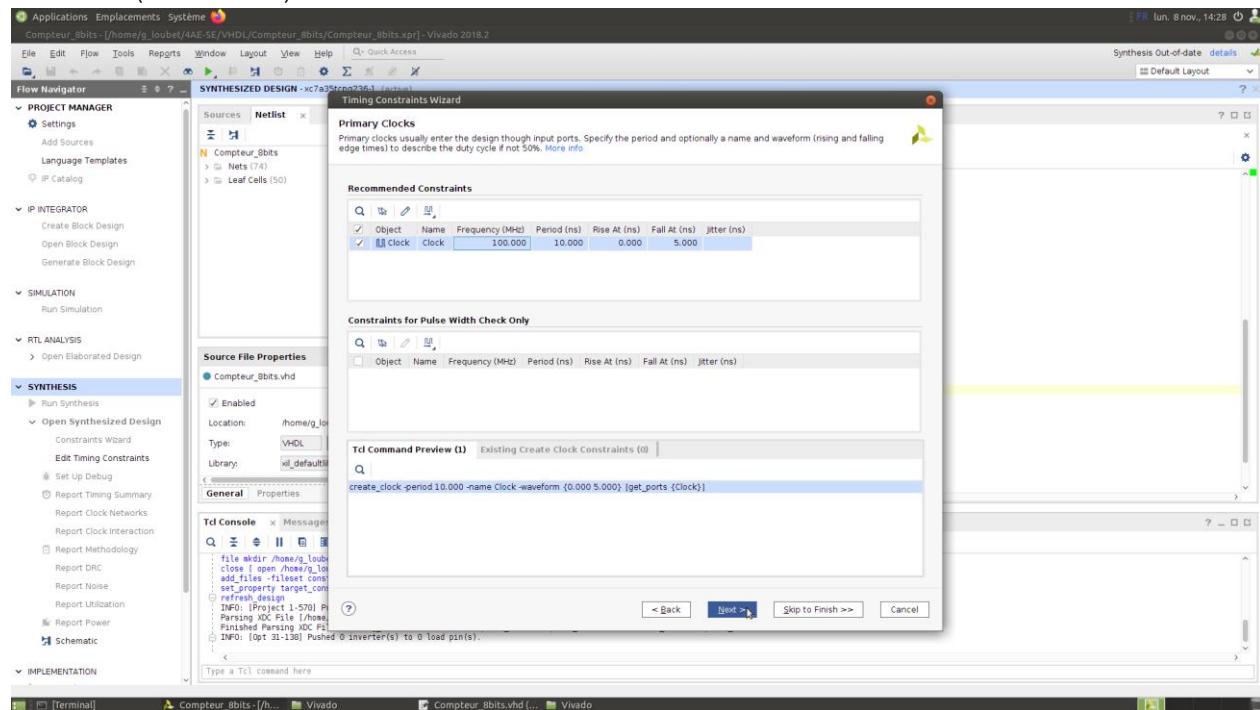
*SYNTHESIS -> Open Synthesized Design -> Constraints Wizard
Puis Reload*



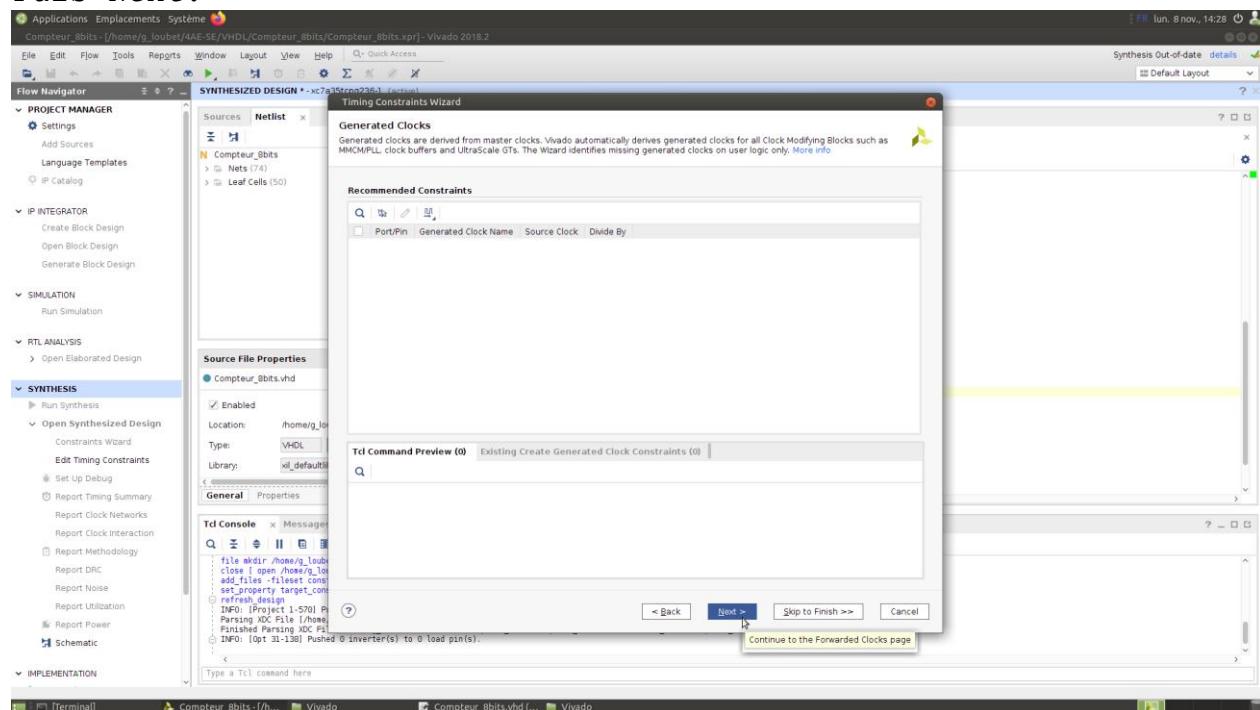
*SYNTHESIS -> Open Synthesized Design -> Constraints Wizard
Puis Next.*



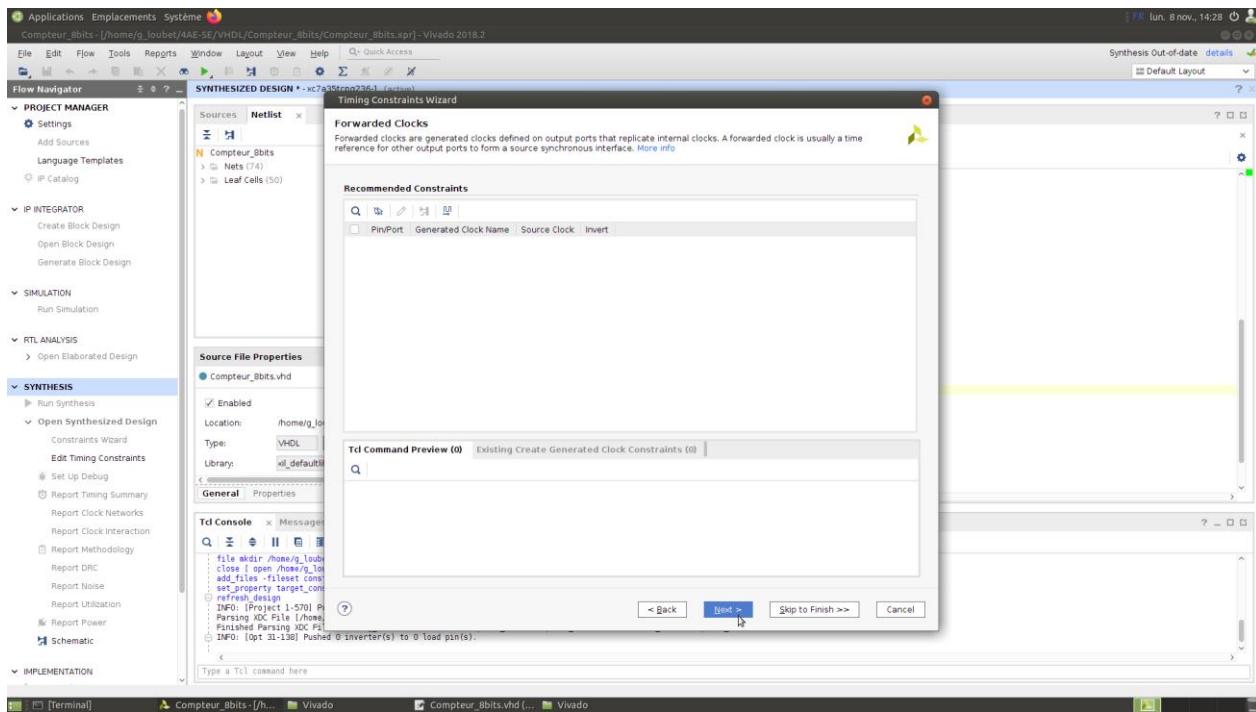
Vérifiez le nom et imposez la fréquence de votre oscillateur local (100 MHz).



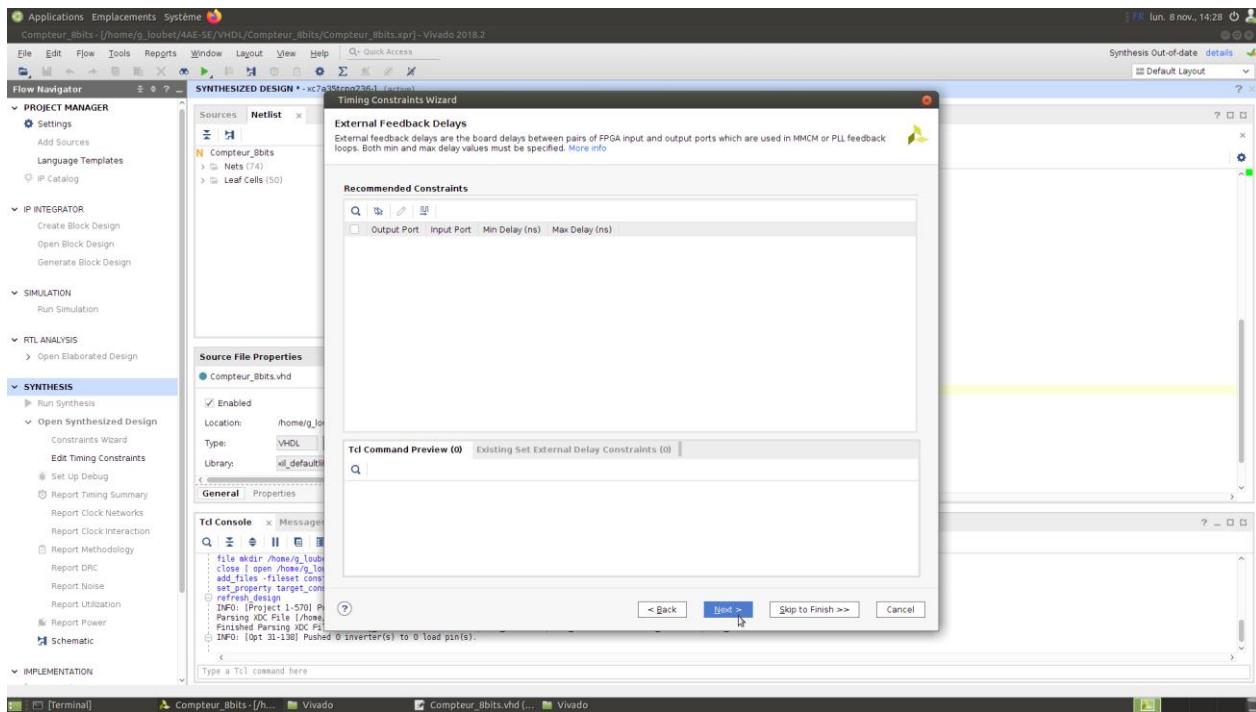
Puis Next.



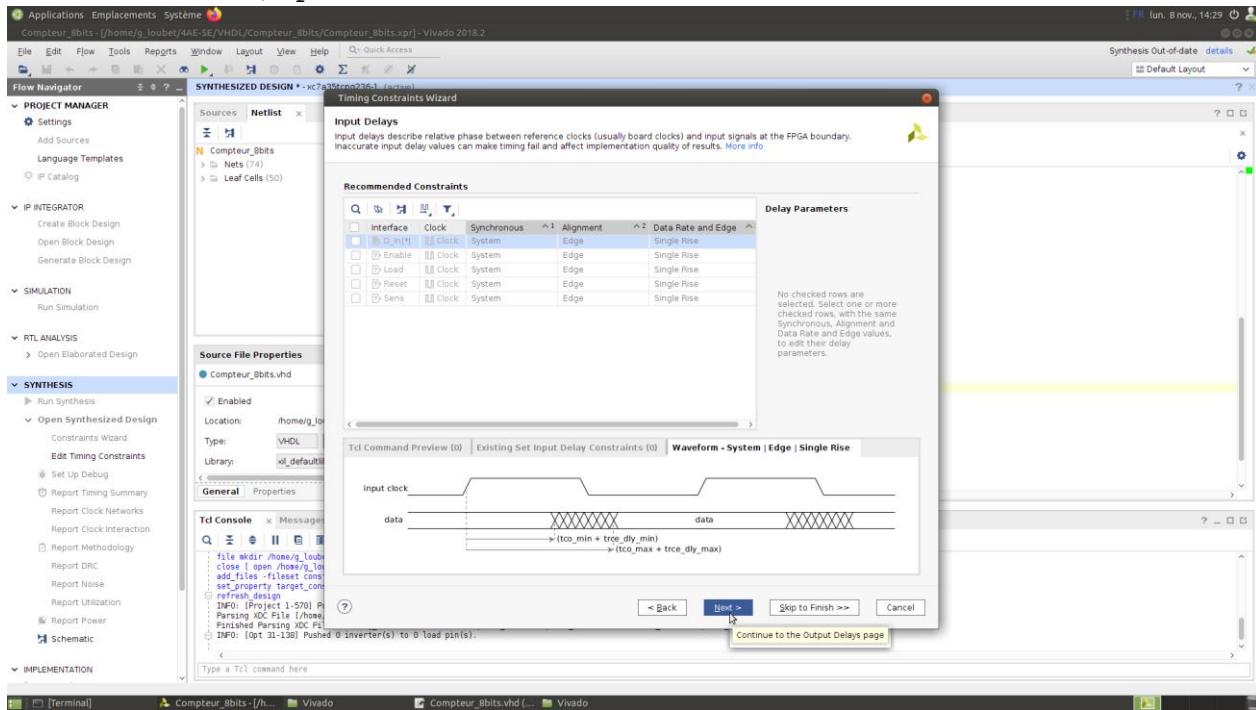
Puis Next.



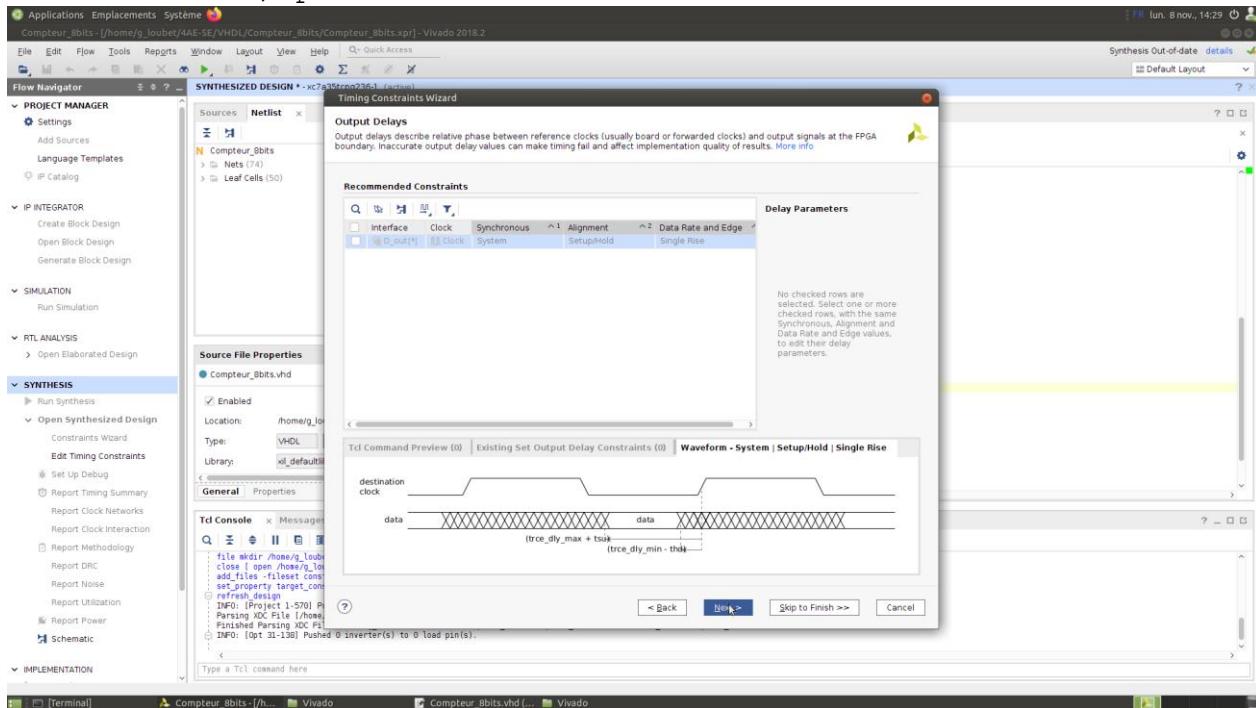
Puis Next.



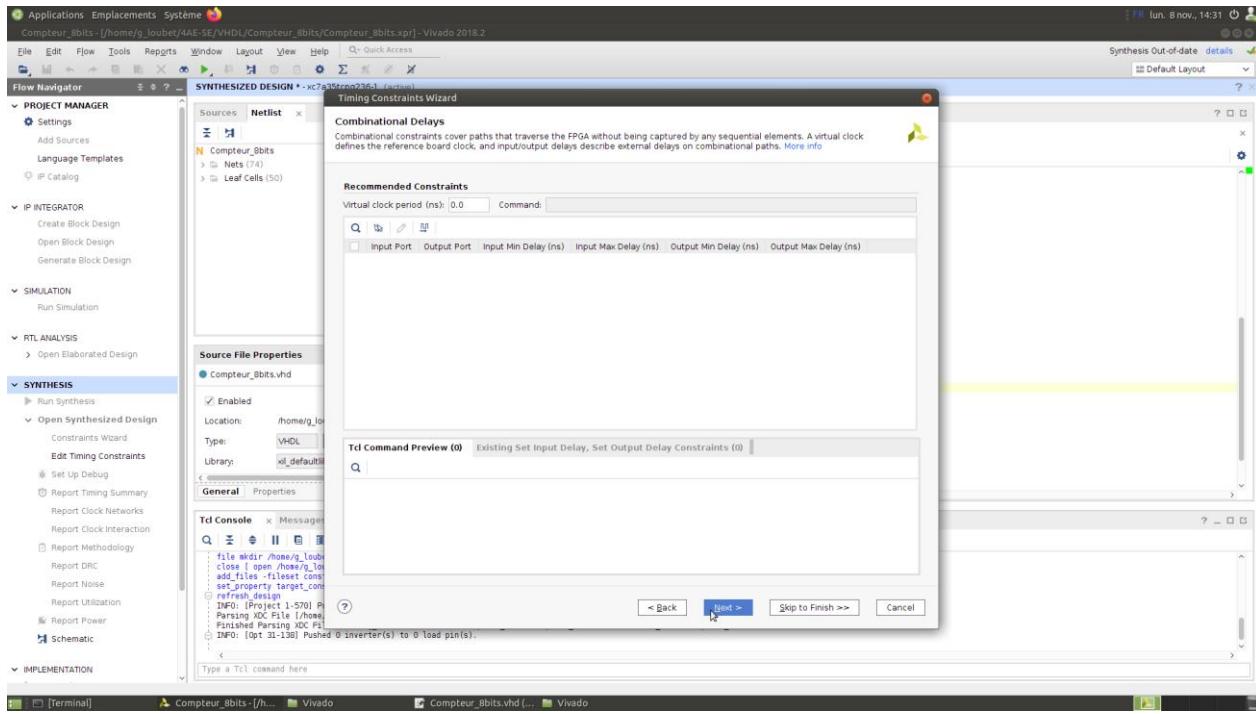
Décochez tous, puis Next.



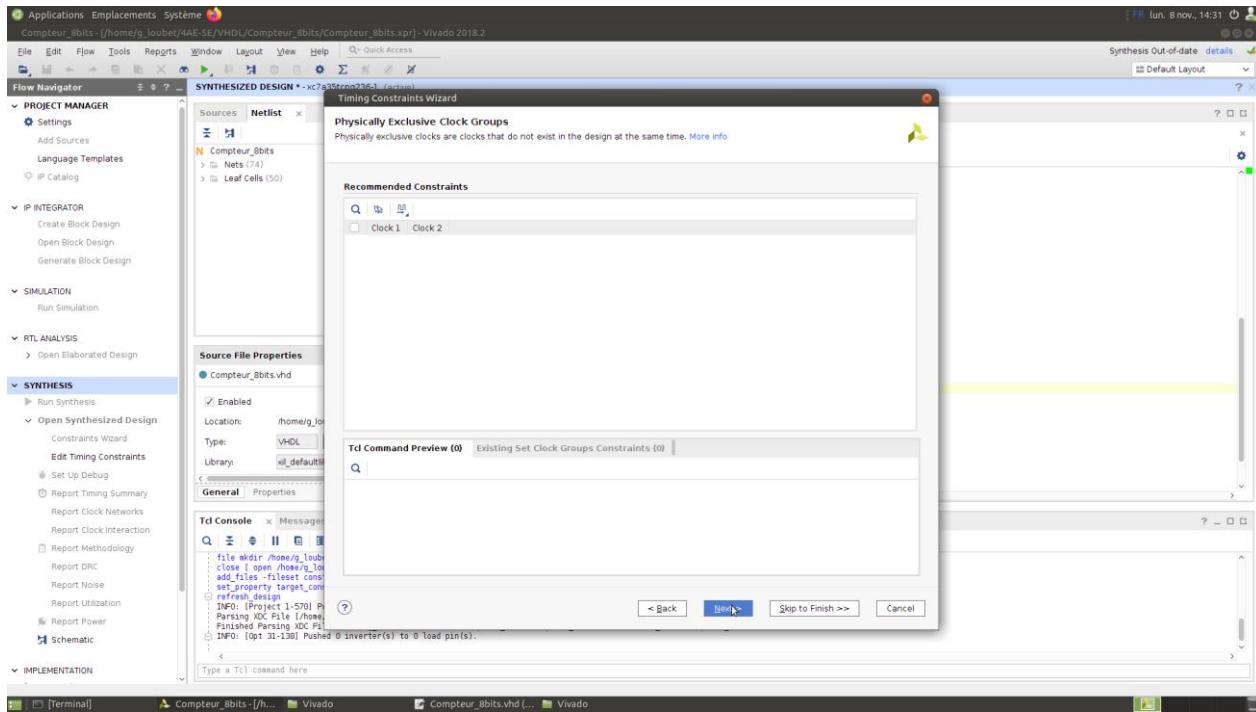
Décochez tous, puis Next.



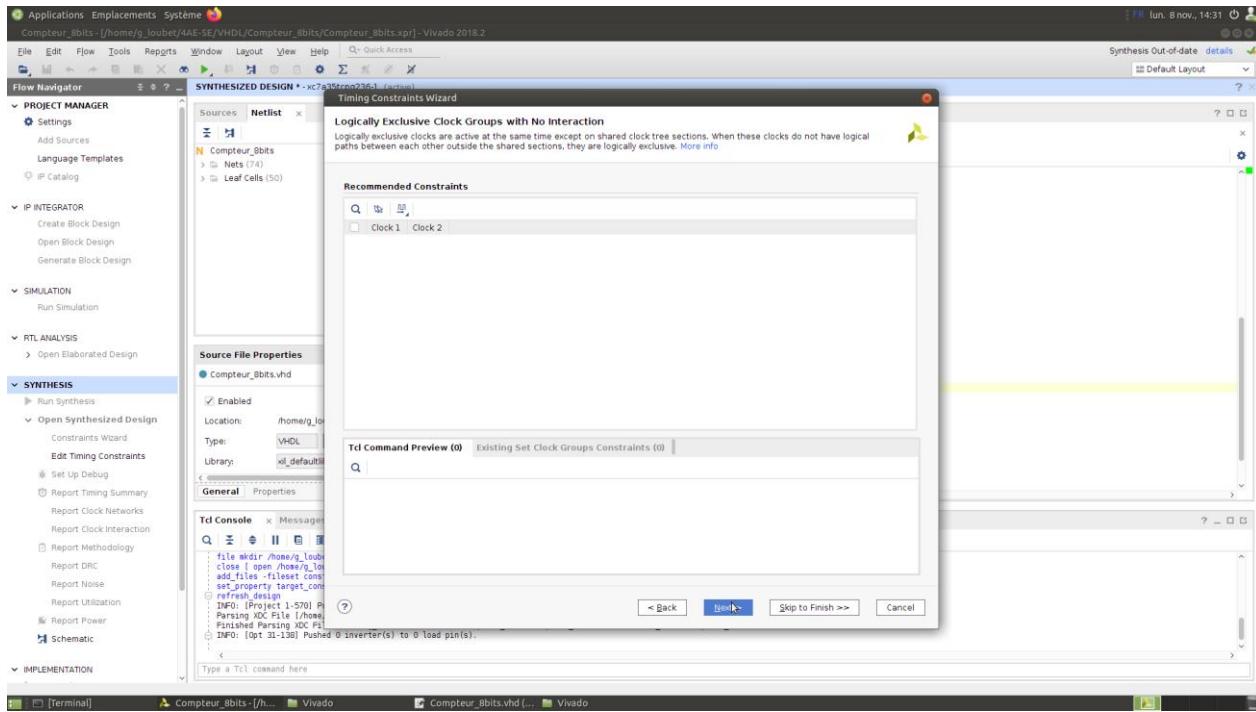
Puis Next.



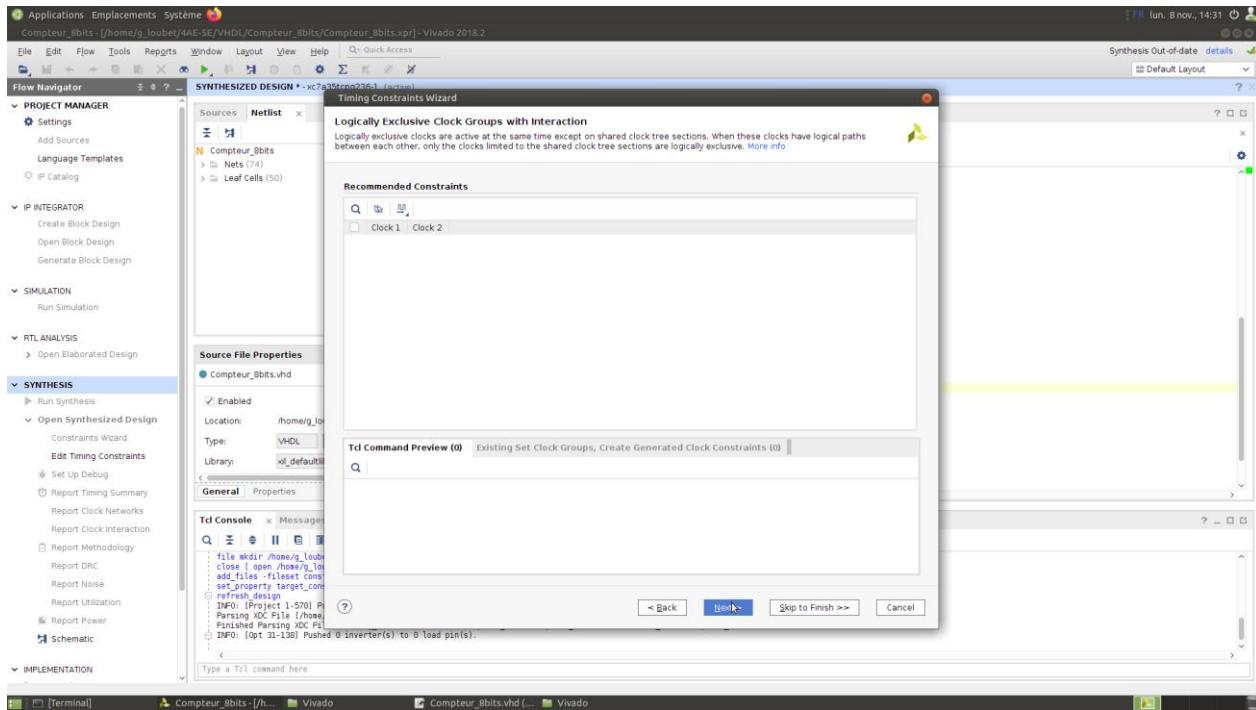
Puis Next.



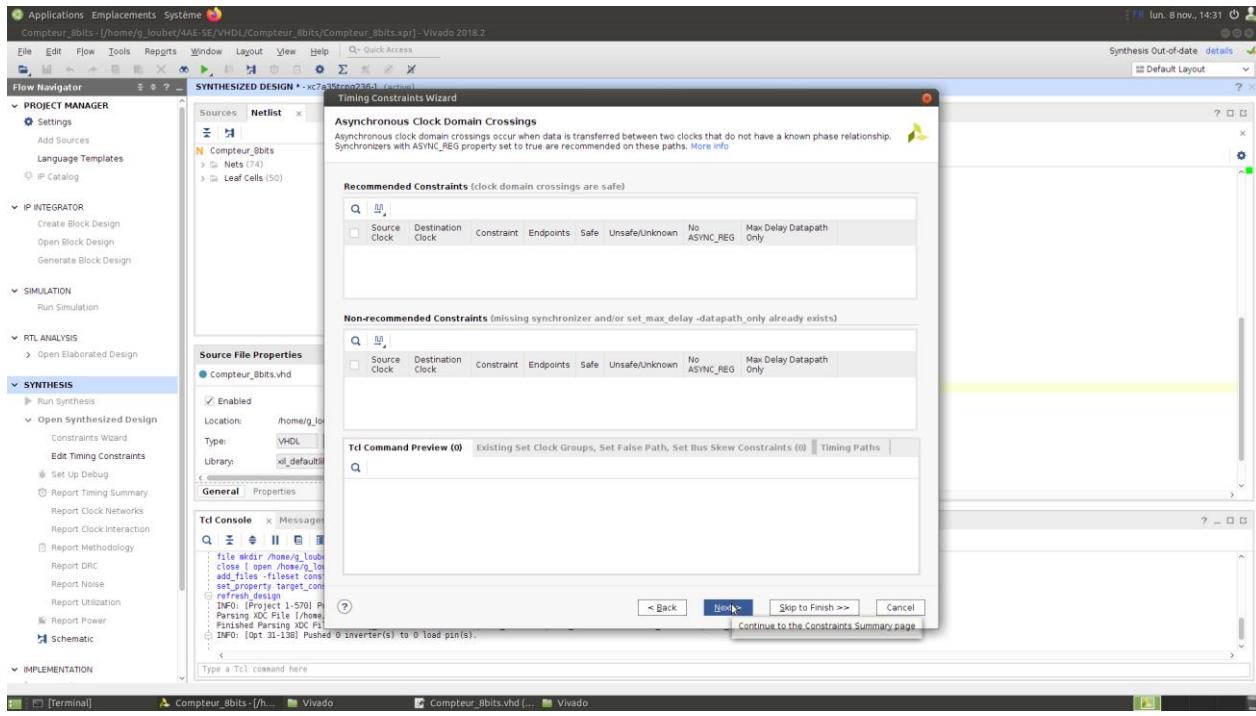
Puis Next.



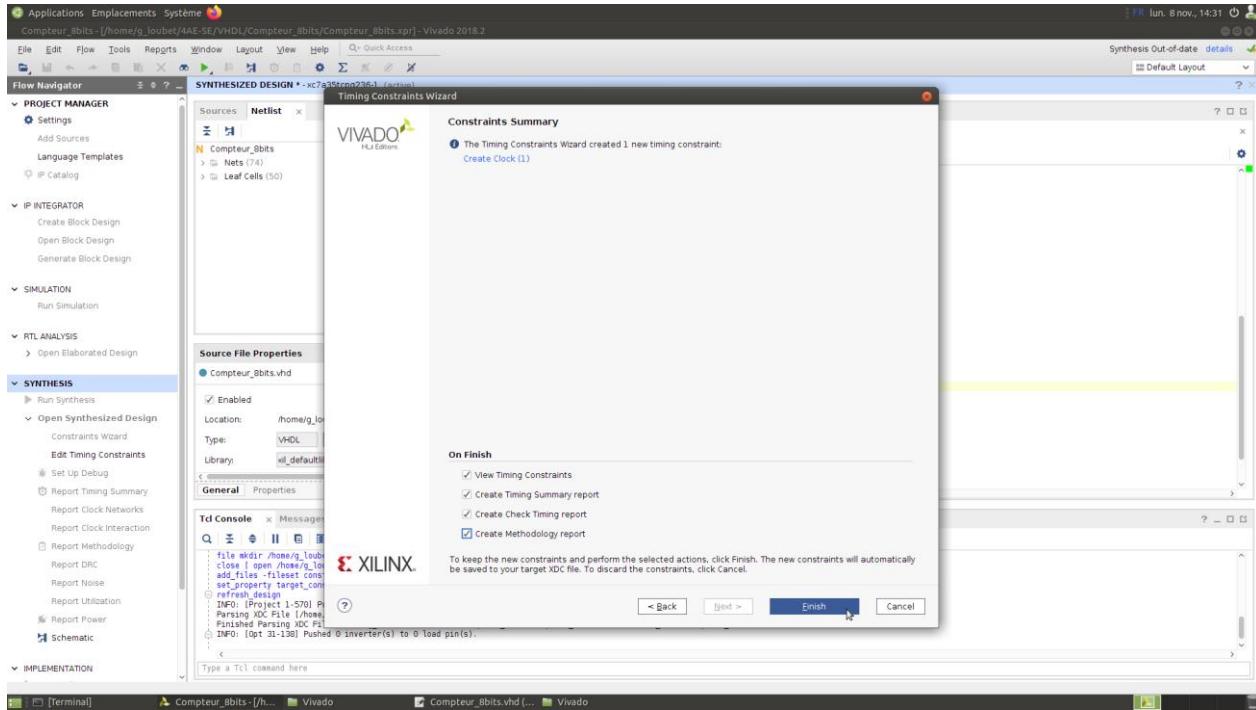
Puis Next.



Puis Next.



Cochez toutes les cases et *Finish*.



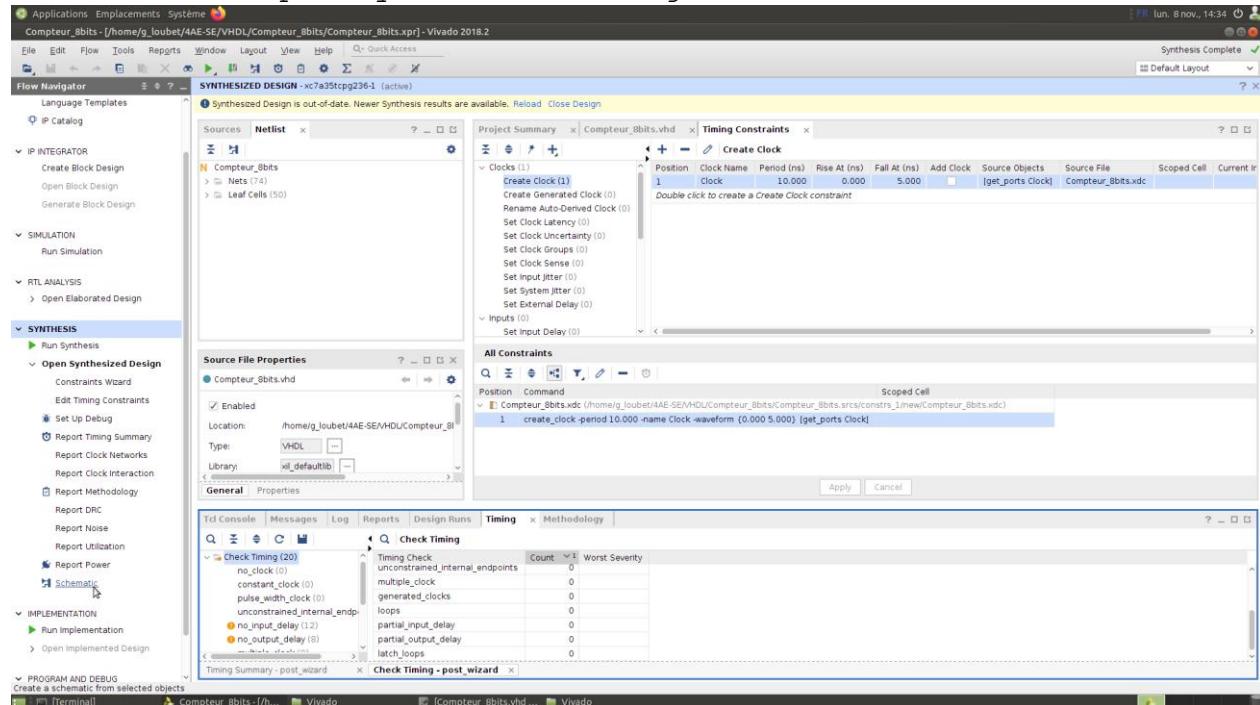
Relancez la synthèse ou l'implémentation.

Dans la console TCL, tapez *report_timing_summary*.

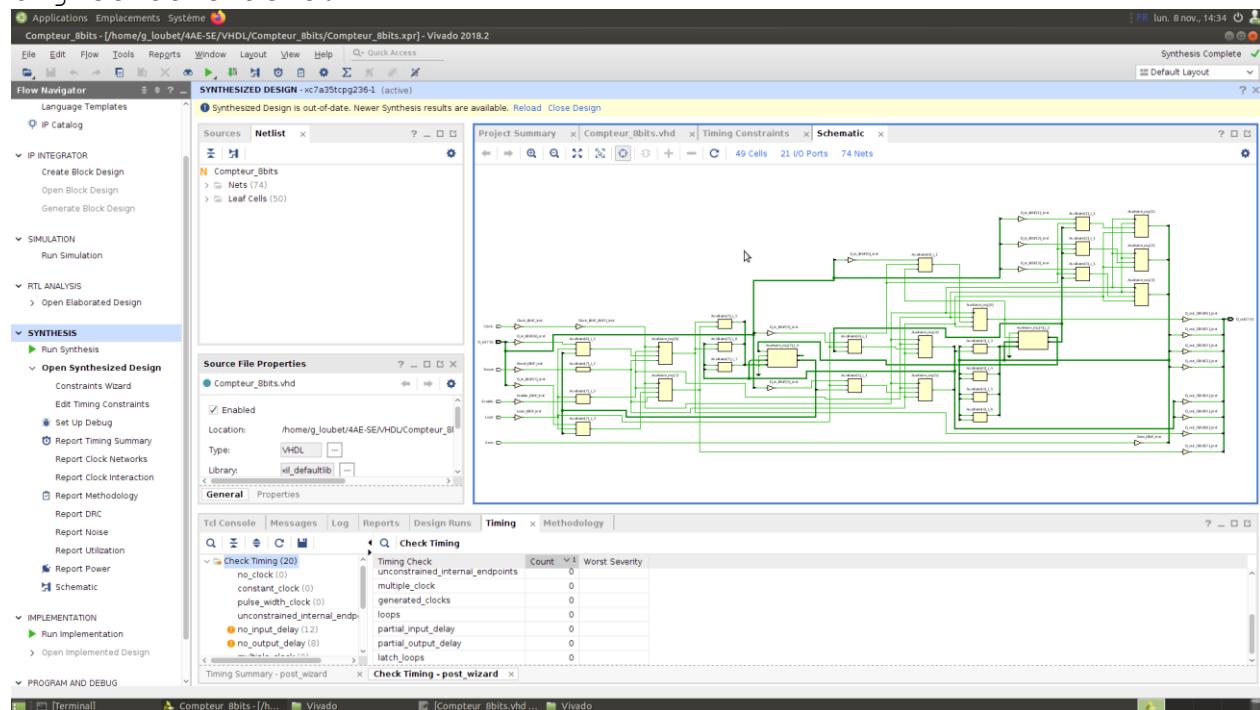
Afficher les circuits générés

Après la synthèse, sélectionnez *Open Synthesized Design*.
ou

SYNTHESIS -> Open Synthesized Design -> Schematic



Onglet Schematic.



Ajouter des librairies

Pour ajouter des librairies, ajoutez en début de fichier la ligne :

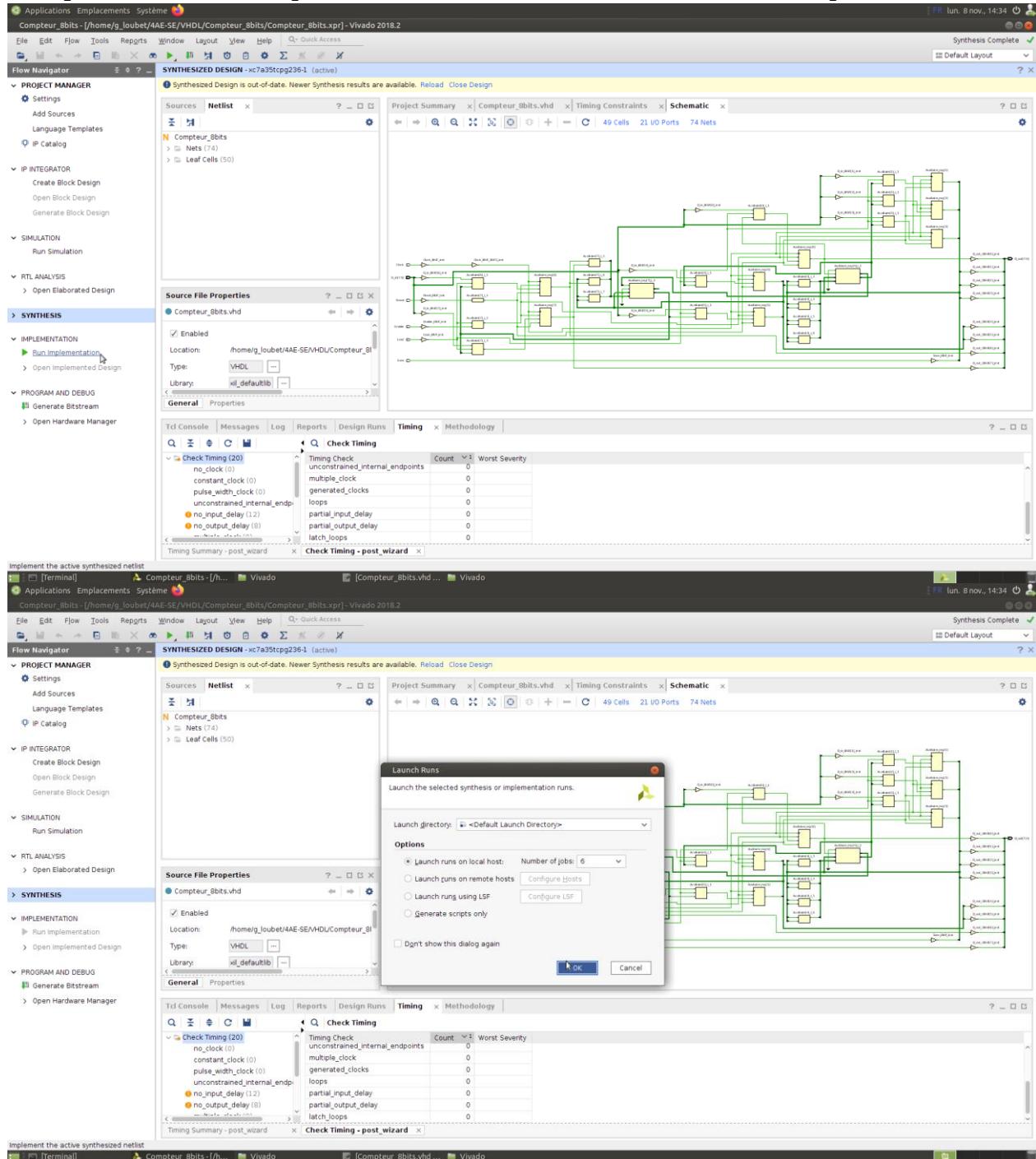
```
use librairie.name;
```

Vous aurez besoin des librairies *IEEE.STD_LOGIC_UNSIGNED.ALL* (notamment pour pouvoir utiliser les fonctions d'incrémentation et de décrémentation sur les vecteurs binaires non-signés) et *IEEE.NUMERIC_STD.ALL* (pour les opérations de base).

NB : Sachez expliquer pourquoi vous utilisez une librairie et quelles sont ses limitations et contraintes.

Placer et router

Cliquez sur *Run Implementation* dans le flux de conception.

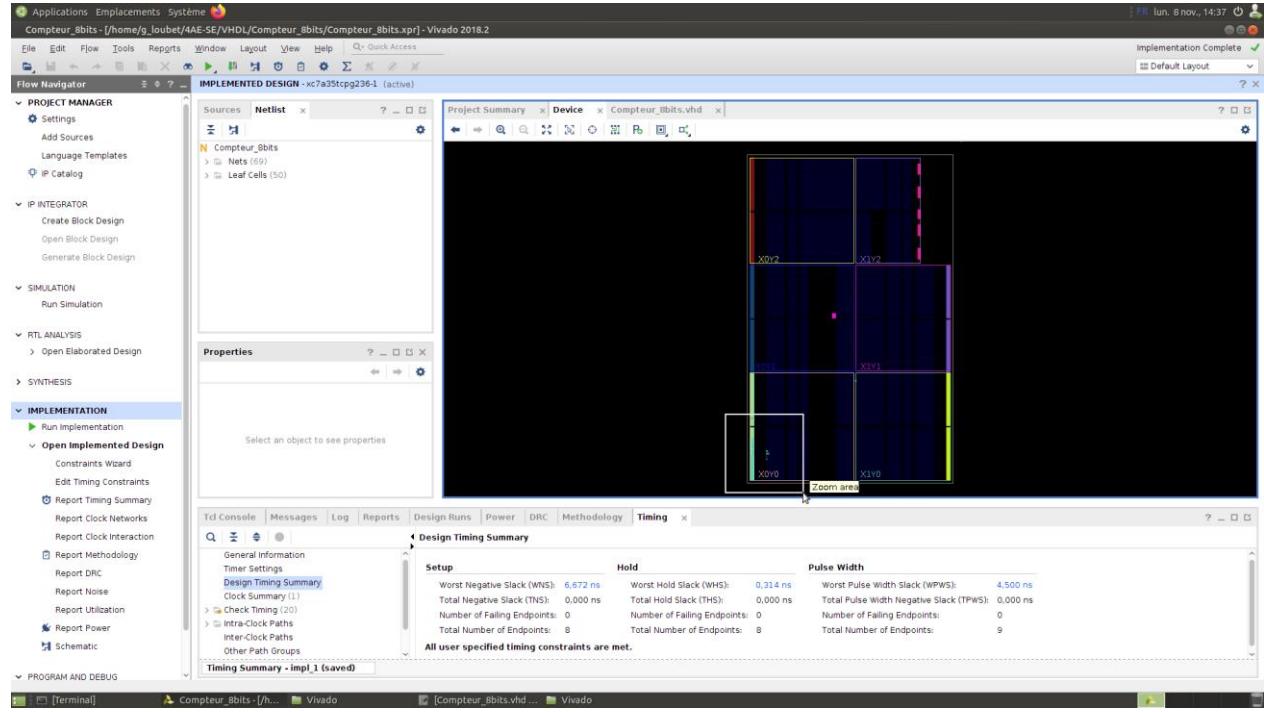


NB : Toutes les étapes nécessaires au flux de conception (Synthèse, Traduction, Mappage) seront automatiquement réalisées.

NB : Le *check* vert indique que c'est bon, la croix rouge qu'il y a des erreurs et le point d'interrogation orange qu'il y a des *warnings*. Pensez à passer la console, partie *Messages* !

Observer le placement et le routage sur le FPGA

Après l'implémentation, sélectionnez *Open Implemented Design*.
ou
Onglet *Devices*.



En cyan apparaissent les portes qui seront activées sur le FPGA pour répondre à votre implémentation. Il est évidemment possible de zoomer.

Créer une source de test

File -> Add Sources...

ou

PROJECT MANAGER -> Add Sources

Add or create simulation sources

'+' -> Create File...

ou

Create File

File type: VHDL

File name: Nommez votre fichier (e.g. Test_Compteur)

File location: <Local to Project>

Validez.

Définissez l'entité.

Entity name: Nommez votre entité (e.g. le même que le fichier)

Architecture name: Behavioral (on va définir une architecture comportementale...).

Vérifiez et validez.

Vous pouvez ouvrir votre fichier généré et commencez votre implémentation.

Sources -> Simulation Sources -> Nom_de_votre_fichier.vhd

NB : Dans votre fichier de test, vous devez définir un process d'horloge, puis votre process de test en fin de fichier.

Bien qu'il soit possible de faire des tests conditionnels, je vous conseille dans un premier temps le format suivant :

Entree_1 <= '0' after n ns, '1' after n+m ns, '0' after n+m+p ns;

Entree_2_Vecteur_3bits <= "101" after m ns, "010" after m+p ns;

Format du fichier de test :

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

entity TestCompteur is
end TestCompteur;

architecture Behavioral of TestCompteur is
    -- Component Declaration for the Unit Under Test (UUT)

COMPONENT Compteur
PORT(
    Clock : IN std_logic;
    Reset : IN std_logic;
    Load : IN std_logic;
    Sens : IN std_logic;
    Enable : IN std_logic;
    D_in : IN std_logic_vector(15 downto 0);
    D_out : OUT std_logic_vector(15 downto 0)
);
END COMPONENT;

--Inputs
signal Clock : std_logic := '0';
signal Reset : std_logic := '0';
signal Load : std_logic := '0';
signal Sens : std_logic := '0';
signal Enable : std_logic := '0';
signal D_in : std_logic_vector(15 downto 0) := (others => '0');

--Outputs
signal D_out : std_logic_vector(15 downto 0);

-- Clock period definitions
constant Clock_period : time := 10 ns;

begin
```

```

-- Instantiate the Unit Under Test (UUT)
uut: Compteur PORT MAP (
    Clock => Clock,
    Reset => Reset,
    Load => Load,
    Sens => Sens,
    Enable => Enable,
    D_in => D_in,
    D_out => D_out
);

-- Clock process definitions
Clock_process :process
begin
    Clock <= not(Clock);
    wait for Clock_period/2;
end process;

-- Stimulus process
stim_proc: process
begin

    -- insert stimulus here
    Reset<= '1', '0' after 250 ns, '1' after 280 ns;
    D_in<= "1111111000000001";

    wait;
end process;

end;

```

Simuler après synthèse ou après placement et routage

Après avoir réalisé la synthèse ou l'implémentation (qui ne doivent pas présenter ni d'erreur, ni de warning).

Vérifiez que le fichier de test est défini comme *Top Module*.

SIMULATION -> Run Simulation -> Run xxx Simulation

Utilisez les différents zooms.

Pour passer l'affichage en hexadécimal ou décimal, sélectionnez le nom du ou des signaux, puis clique-droit -> *Radix -> Hexadecimal*

Pour ajouter du temps à la simulation, mettez votre curseur dans la zone de temps de simulation, entrez une durée et à chaque appui sur la touche entrée, cette durée est ajoutée en fin de simulation.

/!\ Pensez à fermer la fenêtre de simulation à chaque fois pour éviter la saturation de l'espace de stockage dédié à VIVADO. Vous n'êtes pas obligé non plus d'enregistrer à chaque fois la configuration de test.

Créer une source de contraintes

File -> Add Sources...

ou

PROJECT MANAGER -> Add Sources

Add or create constraints

'+' -> Create File...

ou

Create File

File type: XDC

File name: Nommez votre fichier (e.g. Test_Compteur)

File location: <Local to Project>

Validez.

```
# Cheat sheet :  
# Switches:  
# R2, T1, U1, W2, R3, T2, T3, V2, W13, W14, V15, W15, W17, W16, V16, V17  
# Buttons:  
# T18, W19, U18, T17, U17  
# LEDs:  
# L1, P1, N3, P3, U3, W3, V3, V13, V14, U14, U15, W18, V19, U19, E19, U16
```

Format du fichier de contraintes :

```
#Horloge sur un bouton
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets Clock]
set_property -dict {PACKAGE_PIN XN IOSTANDARD LVCMOS33}
[get_ports Clock]

#Horloge avec un oscillateur
#set_property -dist { PACKAGE_PIN W5 IOSTANDARD LVCMOS33 }
[get_ports { Clock } ]
#create_clock -add -name sysclk_pin -period 10.00 -waveform { 0 5
} [get_ports { Clock } ]

set_property -dict {PACKAGE_PIN NX IOSTANDARD LVCMOS33}
[get_ports Reset]
set_property -dict {PACKAGE_PIN NX IOSTANDARD LVCMOS33}
[get_ports Sens]
set_property -dict {PACKAGE_PIN NX IOSTANDARD LVCMOS33}
[get_ports Load]
set_property -dict {PACKAGE_PIN NX IOSTANDARD LVCMOS33}
[get_ports Enable]

set_property -dict {PACKAGE_PIN NX IOSTANDARD LVCMOS33}
[get_ports {D_in[N]}]
...
set_property -dict {PACKAGE_PIN NX IOSTANDARD LVCMOS33}
[get_ports {D_out[N]}]
ports Clock
...
...
```