INF1500 – logique des systèmes numériques

PARTIE IV

VHDL avec Vivado

Introduction à l'utilisation de Vivado et à la technologie FPGA

Révisions :
Alexy Torres Aurora Dugo - V1.0

Septembre 2018

Département de génie informatique et de génie logiciel
École Polytechnique de Montréal

1 Création d'un projet VHDL

Créer un projet VHDL avec Vivado se fait de la même manière que pour les projets avec des diagrammes.

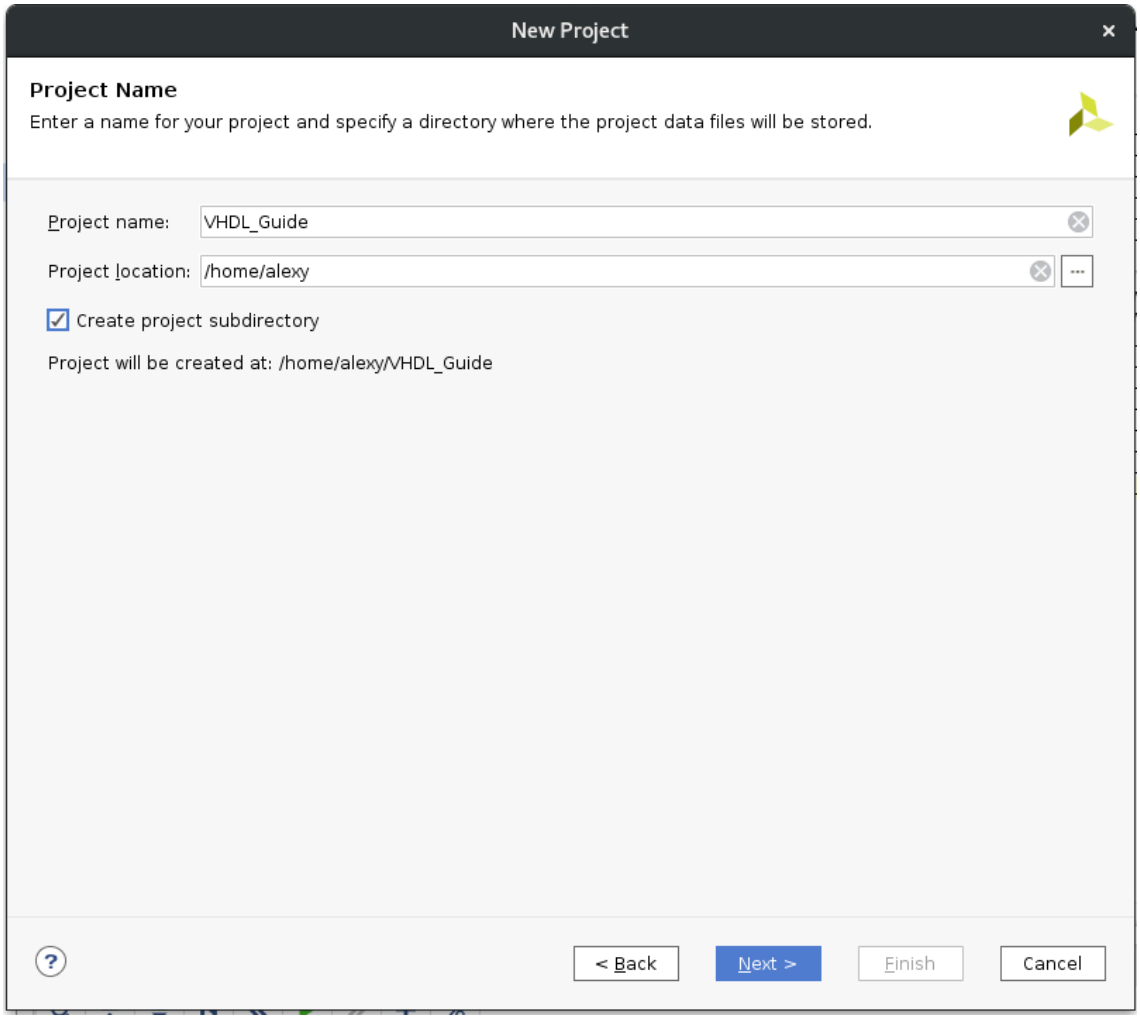

FIGURE 1 – Création du projet Vivado

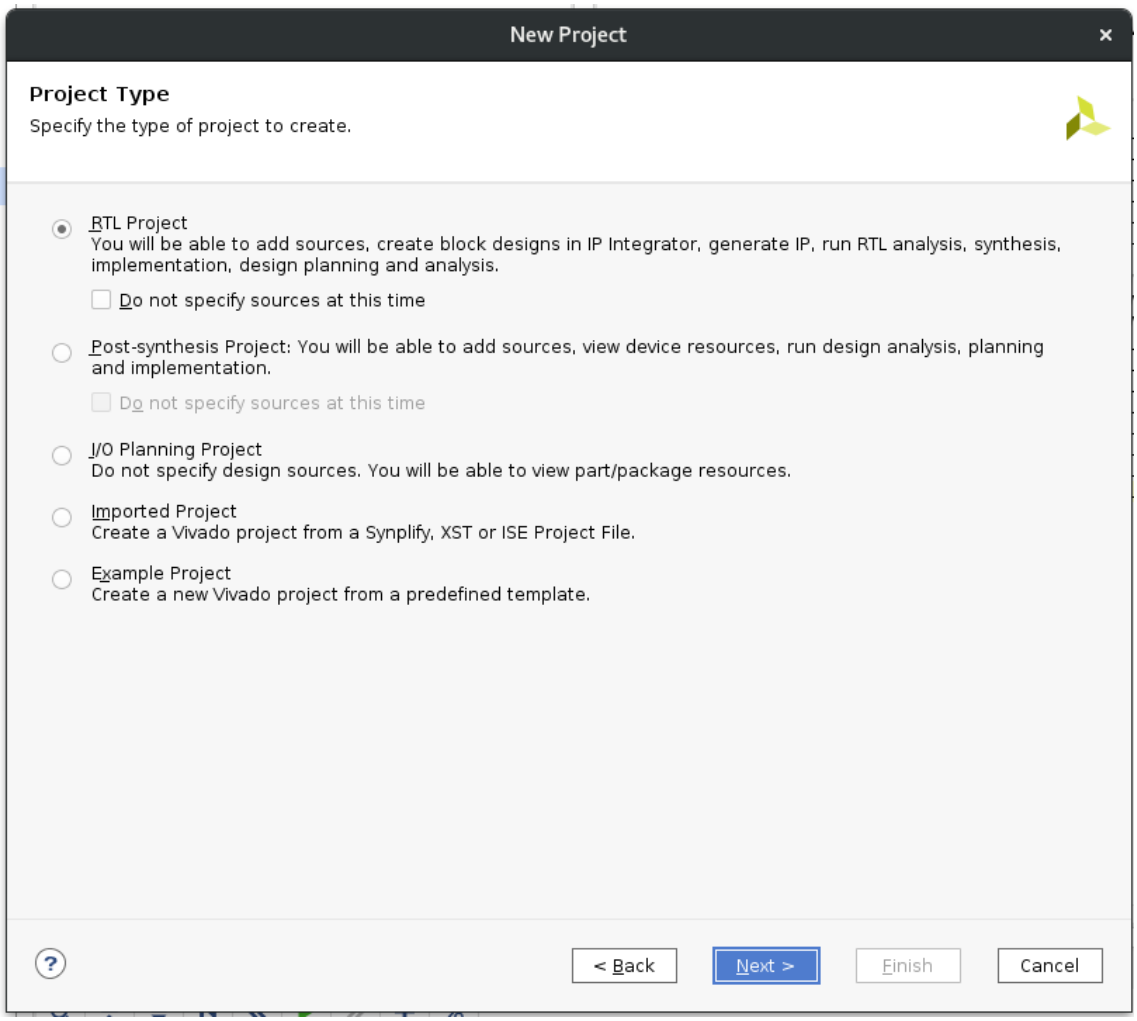

FIGURE 2 – Création du projet Vivado

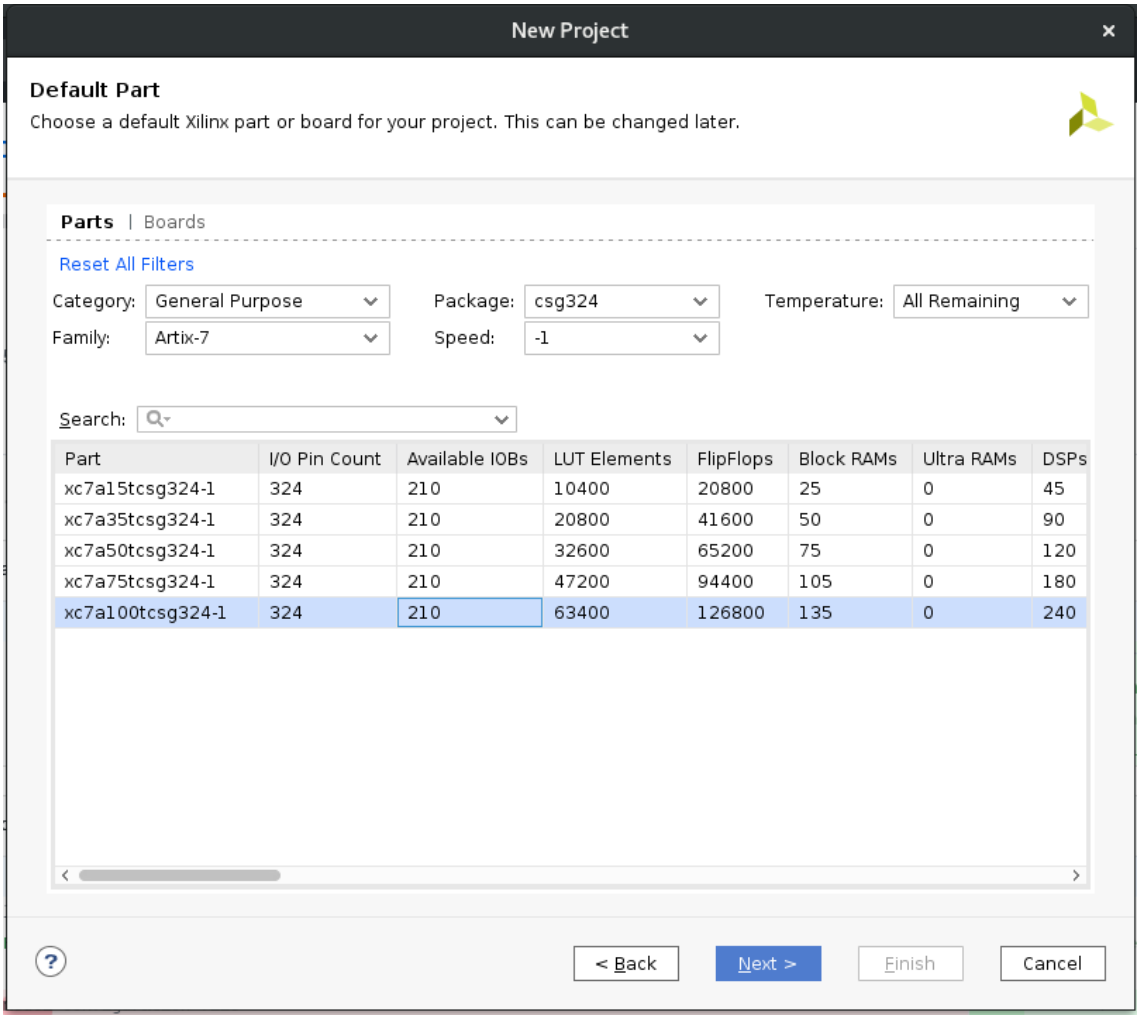

FIGURE 3 – Création du projet Vivado

Afin d'ajouter un nouveau circuit VHDL, allez dans **File** en haut à gauche de la fenêtre Vivado.

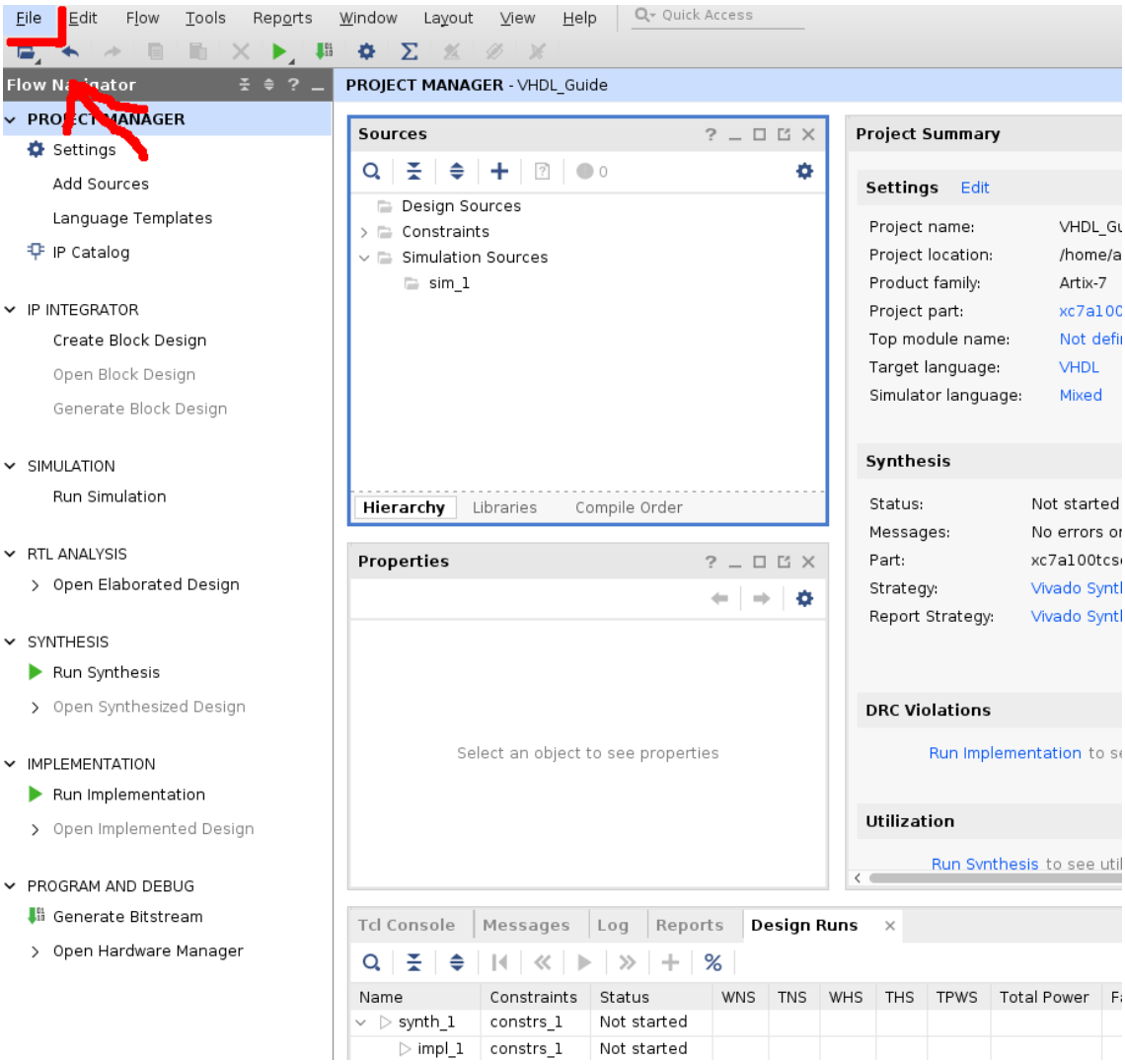

FIGURE 4 – Création du projet Vivado

Cliquez sur **Add Sources...**. Une fenêtre s'ouvre, sélectionnez **Add or create design sources**.

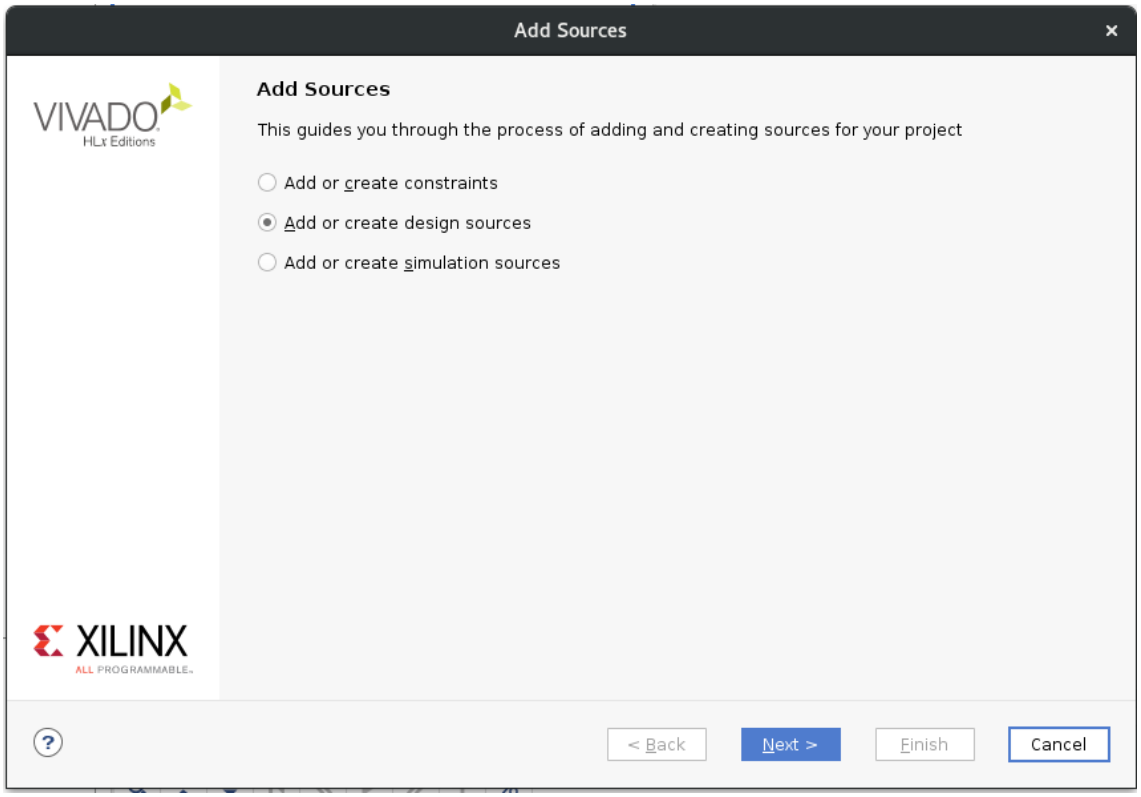

FIGURE 5 – Création du projet Vivado

Dans la fenêtre qui suit, cliquez sur **Create File**. Dans **File type**, vérifiez que **VHDL** soit bien sélectionné. Si ce n'est pas le cas, changez la sélection pour **VHDL**. Une fois fait, rentrez le nom de votre fichier VHDL dans **File name** puis cliquez sur **OK**. Enfin cliquez sur **Finish** dans la fenêtre d'addition de sources.

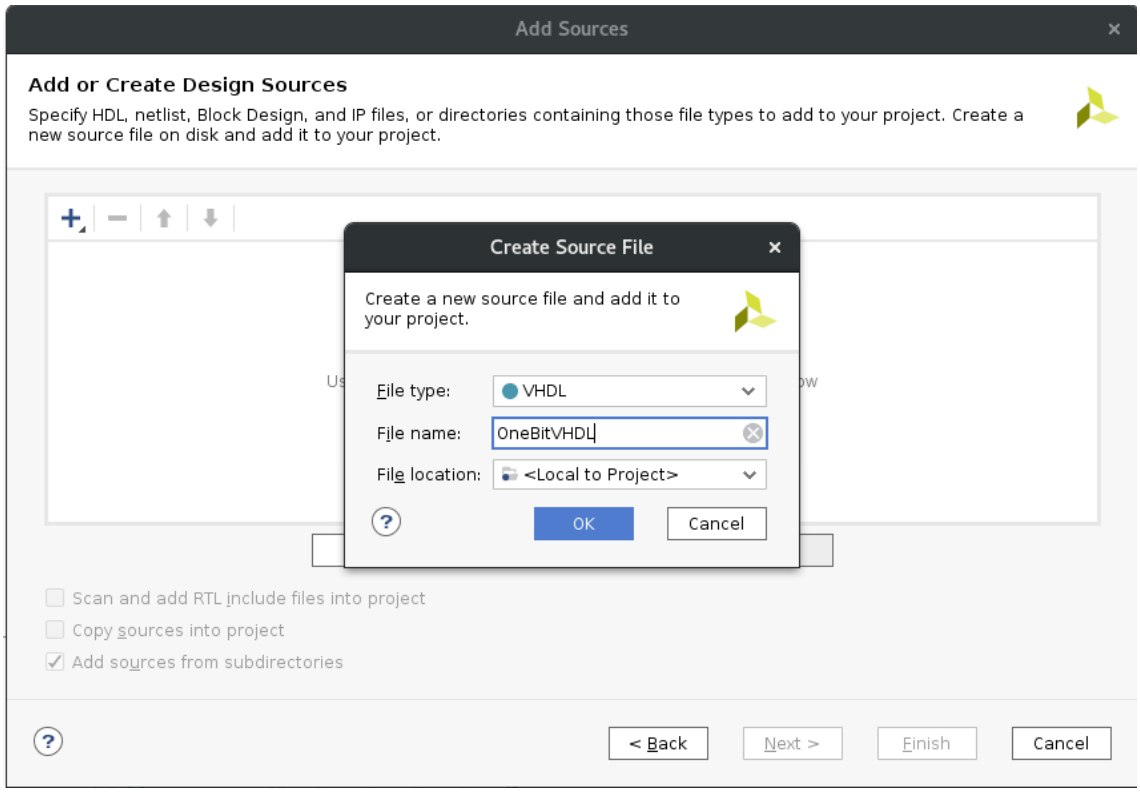

FIGURE 6 – Création du projet Vivado

La fenêtre suivante apparaît :

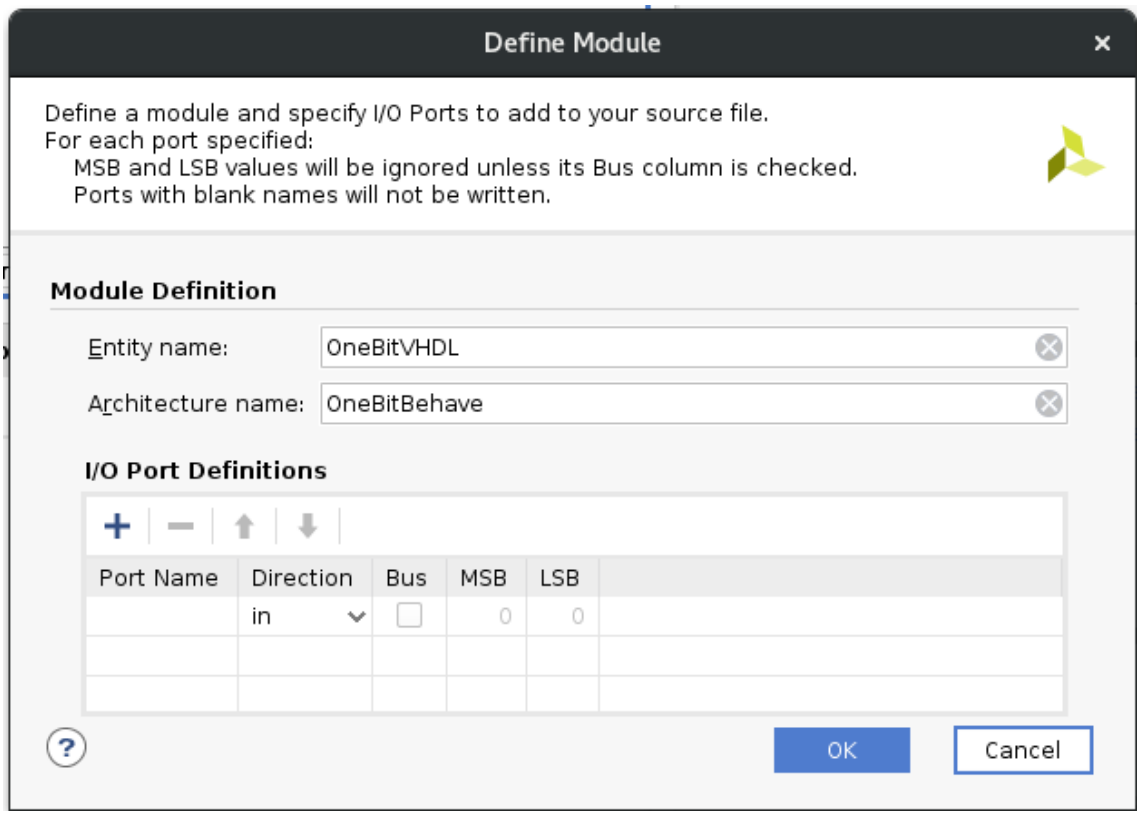

FIGURE 7 – Création du projet Vivado

Rentrez le nom de l'entité dans **Entity name** (vous avez vu ce qu'était une entité en cours). Rentrez aussi le nom de l'architecture correspondant à l'entité dans **Architecture name** (vous avez aussi vu en cours ce qu'était une architecture). Cliquez sur enfin **OK**. Si une fenêtre de confirmation apparaît, cliquez simplement sur **Yes**.

Votre fichier apparaît dans le **Sources** explorer. Double cliquez dessus. Voici votre fichier source VHDL. Écrivez le code VHDL correspondant a votre circuit dans ce fichier. Ici, j'ai choisi d'exprimer le One Bit Full Adder en description de type flot de données, mais c'est a vous de choisir quelle description utiliser.

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

entity OneBitVHDL is
    port(
        A, B, Cin: in STD_LOGIC;
        S, Cout:   out STD_LOGIC
    );
end OneBitVHDL;

architecture OneBitFlow of OneBitVHDL is
begin
```

```
    S <= A XOR B XOR Cin;
    Cout <= (A AND B) OR (A AND Cin) or (B AND Cin);
end OneBitFlow;
```

Pour utiliser notre one bit full adder dans une autre circuit, nous créons un nouveau fichier VHDL (il faut bien créer une ficheir VHDL pour chaque circuit que vous faites). Une fois ce nouveau circuit créé, il suffit d'utiliser le nom d'une l'entité VHDL créée précédemment comme un component pour l'utiliser.

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

entity FourBitsVHDL is
    port(
        A, B: in  STD_LOGIC_VECTOR(3 downto 0);
        Cin:  in  STD_LOGIC;
        S:    out STD_LOGIC_VECTOR(3 downto 0);
        Cout: out STD_LOGIC
    );
end FourBitsVHDL;

architecture FourBitsStrucural of FourBitsVHDL is

    component OneBitVHDL
        port(
            A, B, Cin: in STD_LOGIC;
            S, Cout:   out STD_LOGIC
        );
    end component;

    signal TMP_COUT0: STD_LOGIC;
    signal TMP_COUT1: STD_LOGIC;
    signal TMP_COUT2: STD_LOGIC;

begin
    U0 :
        OneBitVHDL port map(
            A => A(0), B => B(0), Cin => Cin, S => S(0), Cout => TMP_COUT0
        );
    U1:
        OneBitVHDL port map(
            A => A(1), B => B(1), Cin => TMP_COUT0, S => S(1), Cout => TMP_COUT1
        );
    U2:
        OneBitVHDL port map(
            A => A(2), B => B(2), Cin => TMP_COUT1, S => S(2), Cout => TMP_COUT2
        );
    U3:
        OneBitVHDL port map(
            A => A(3), B => B(3), Cin => TMP_COUT2, S => S(3), Cout => Cout
        );
end FourBitsStrucural;
```

Le circuit décrit de manière structurelle est une Four Bit Adder, il ne fait cependant pas

la soustraction comme celui que vous avez fait en TP.

Tous vos fichiers VHDL doivent être dans le même projet Vivado !

2 Simulation, synthèse, implémentation et programmation de la carte

Ici, pas besoin de créer un Wrapper HDL pour simuler votre circuit, hormis cela, toutes les étapes de simulation, synthèse, implémentation, création du fichier de contraintes et programmation de la carte sont les mêmes que d'habitude !