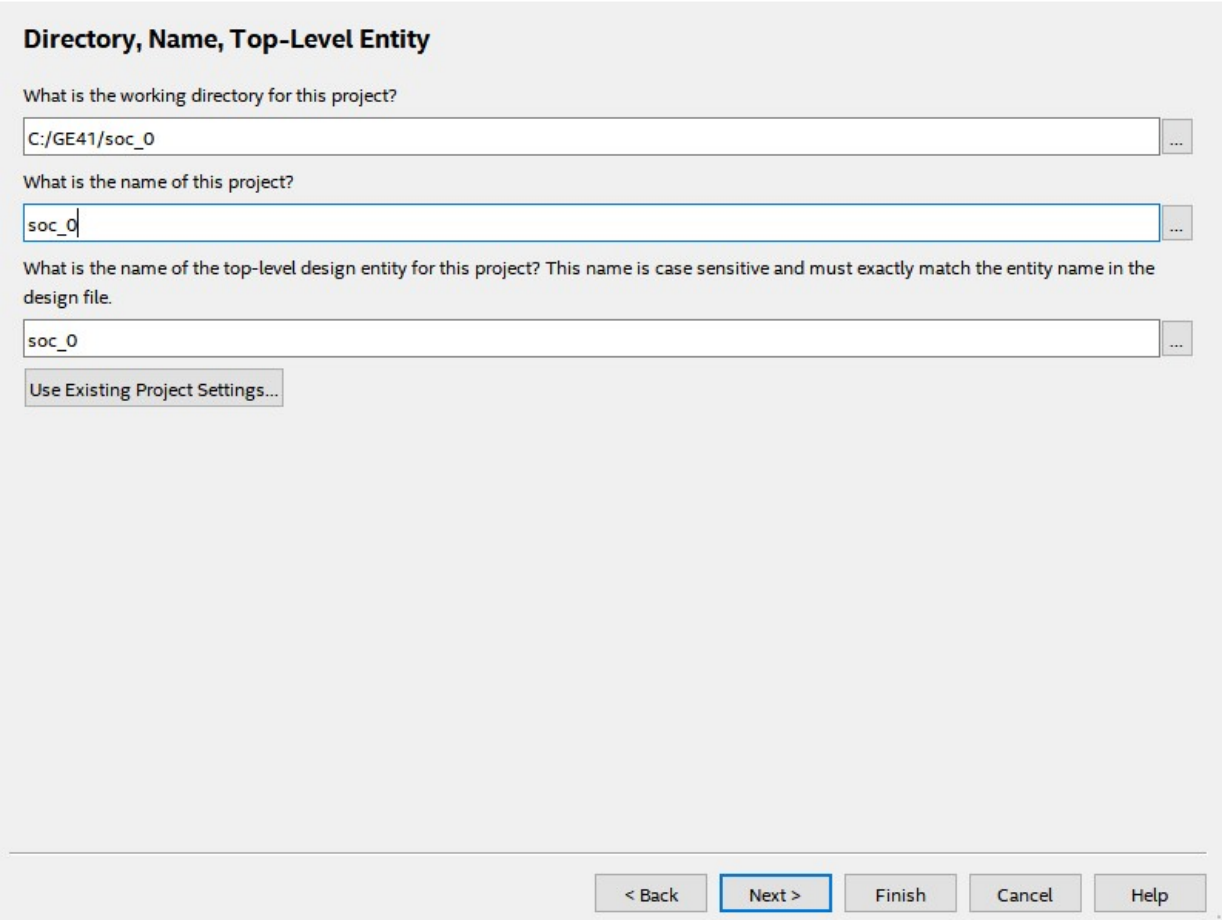
ENSA-Marrakech **Année Universitaire: 2024-2025**

M41
TP3

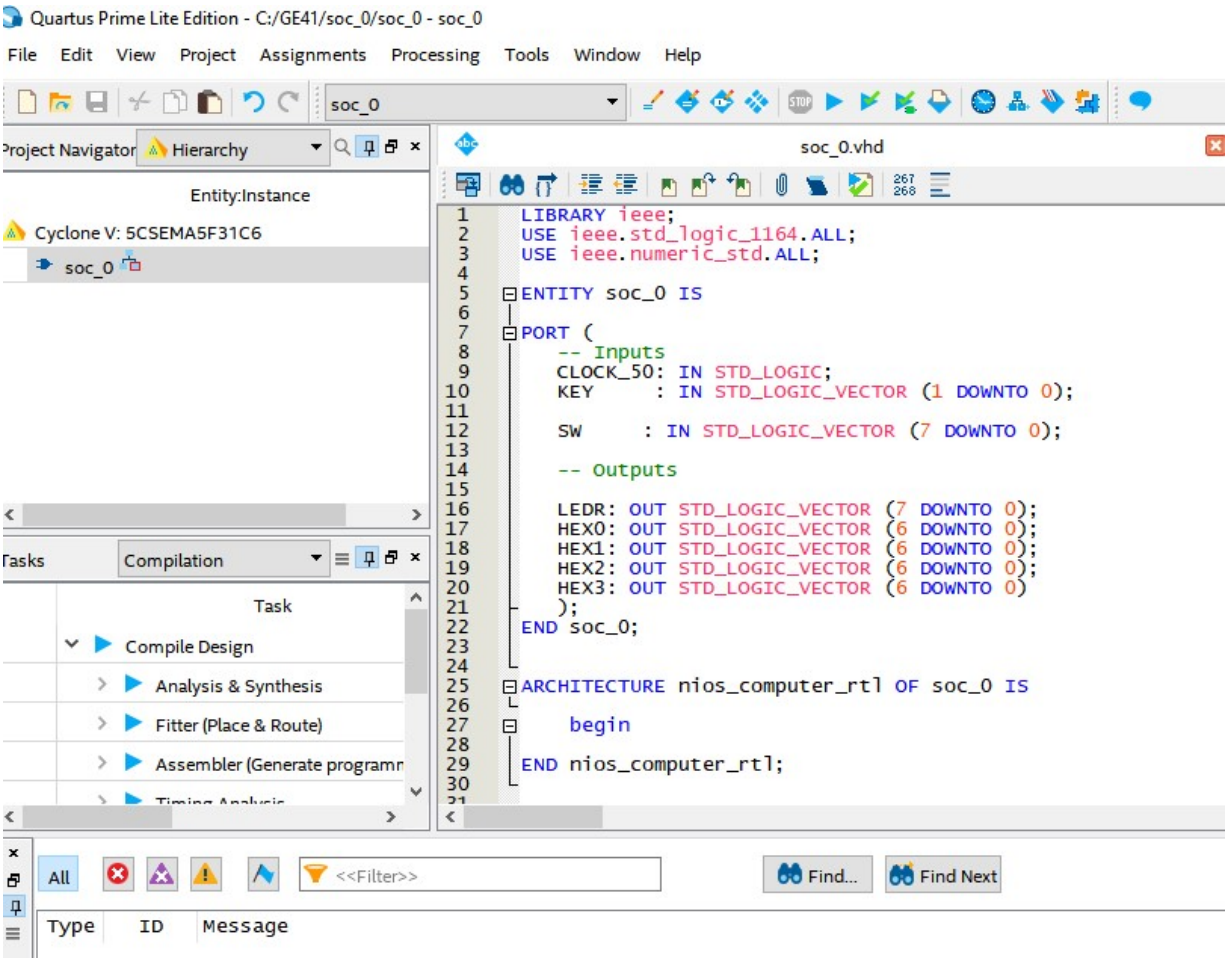
Introduction à l'outil Platform Designer

Ce TP présente une introduction à l'outil Intel® Platform Designer, qui permet de concevoir des systèmes matériels numériques contenant des composants tels que des processeurs, des mémoires, des interfaces d'entrée/sortie, des timers, etc. L'outil Platform Designer permet à un concepteur de choisir les composants souhaités dans le système en sélectionnant ces composants dans une interface utilisateur graphique. Il génère ensuite automatiquement le système matériel qui relie tous les composants entre eux.
Le flux de développement du système matériel est illustré en donnant des instructions étape par étape pour utiliser l'outil Platform Designer avec le logiciel Quartus pour implémenter un exemple de système simple. La dernière étape du processus de développement consiste à configurer le système matériel conçu dans un périphérique FPGA réel et à exécuter un programme d'application

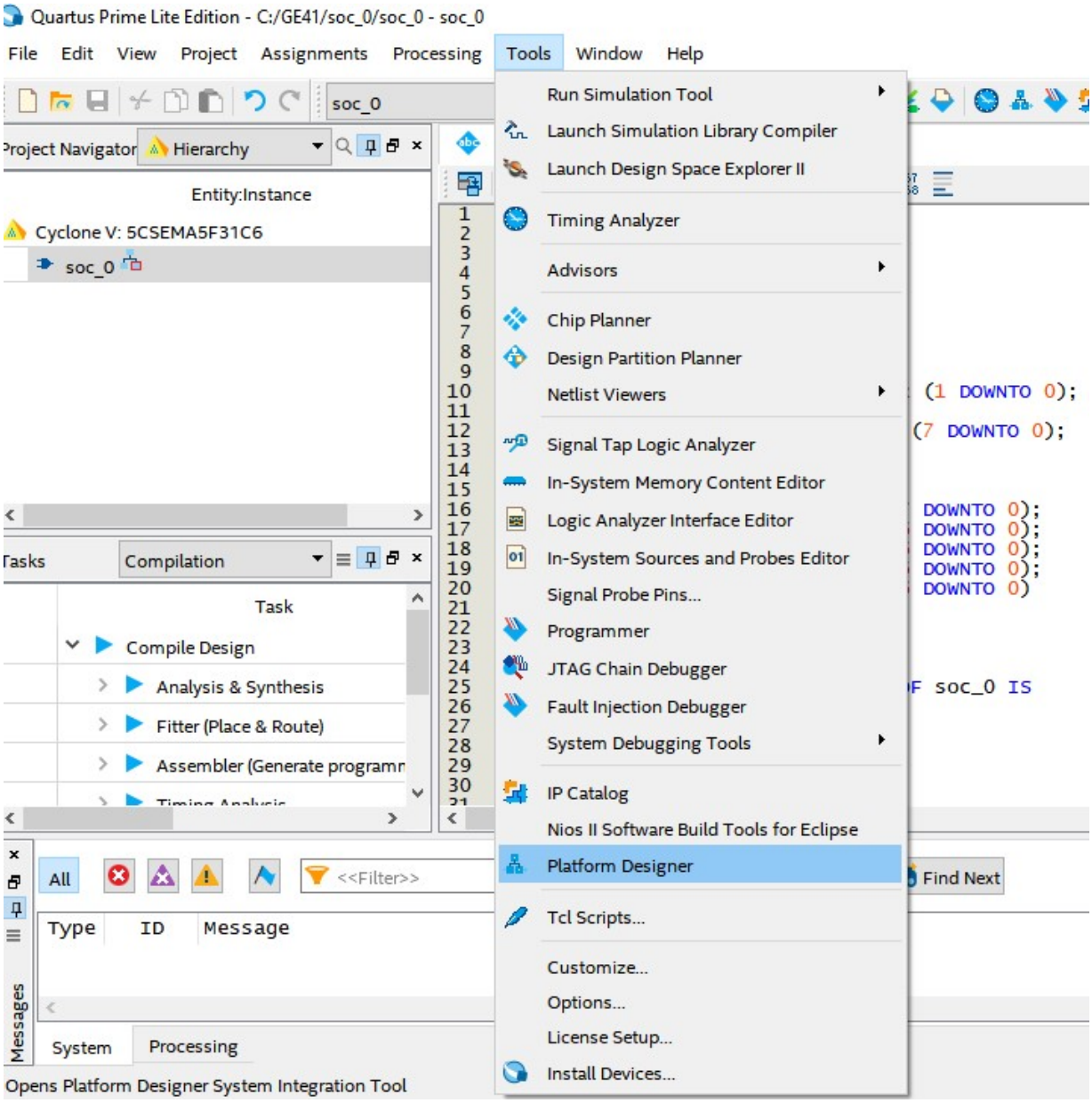
1) Créer un nouveau projet Quartus comme indiqué dans la figure 1.



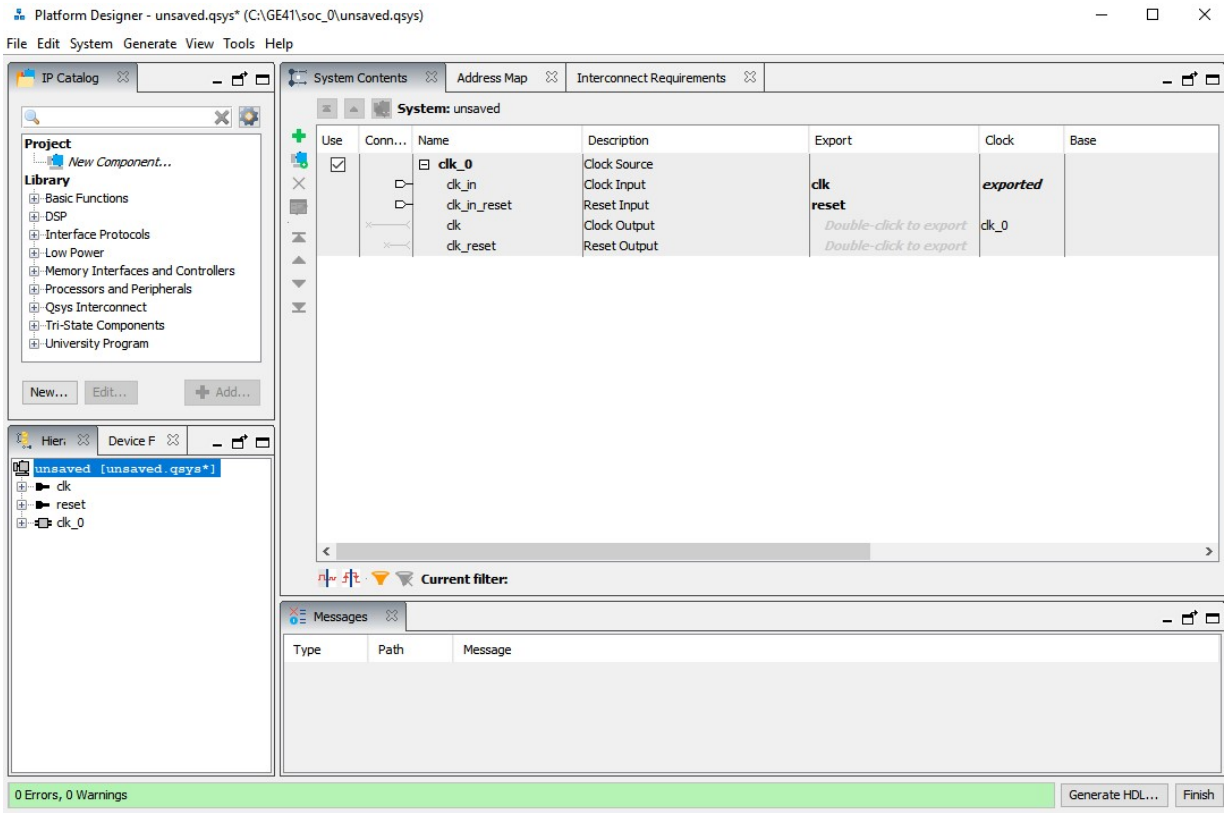
2) Ouvrir le fichier soc_0.vhd. Le contenu du fichier est indiqué dans la figure 2



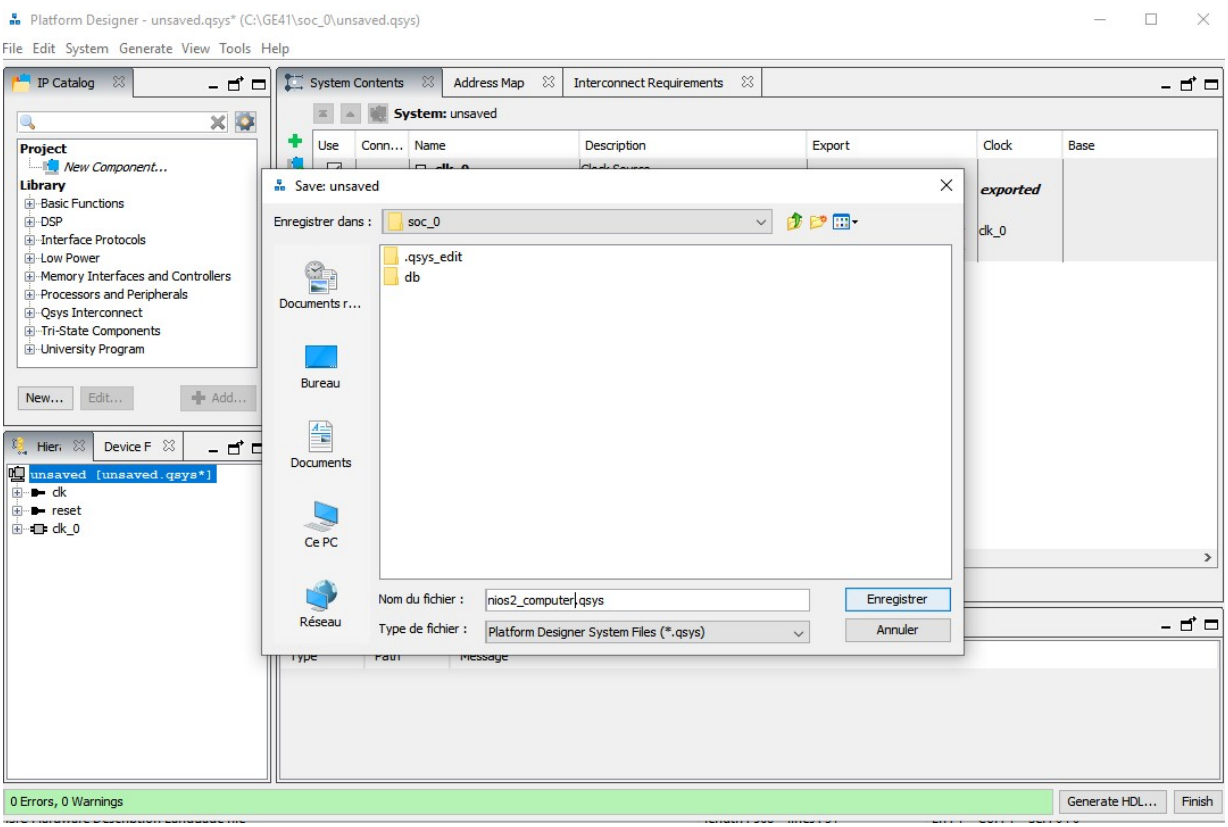
3) Dans la fenêtre Tools, choisir Platform Designer



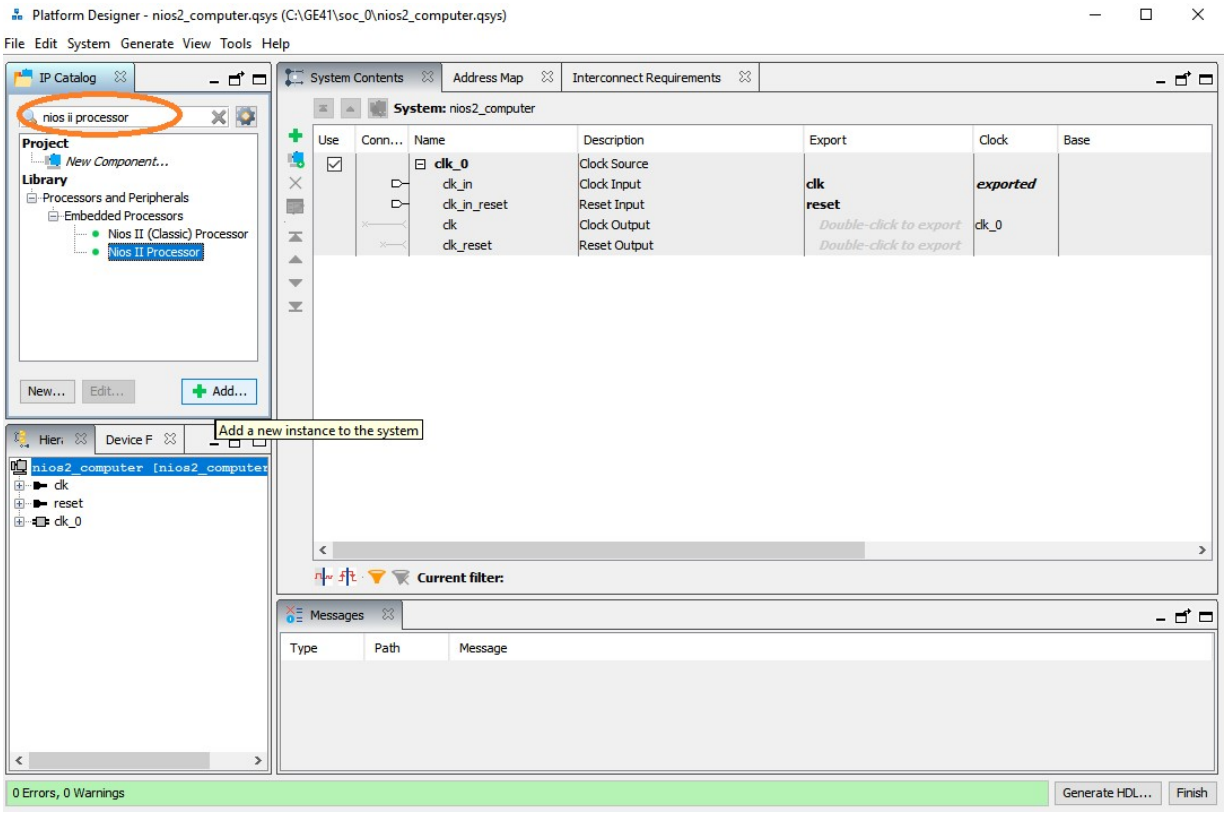
4) L'outil Platform Designer s'ouvre. Enregistrer le système comme



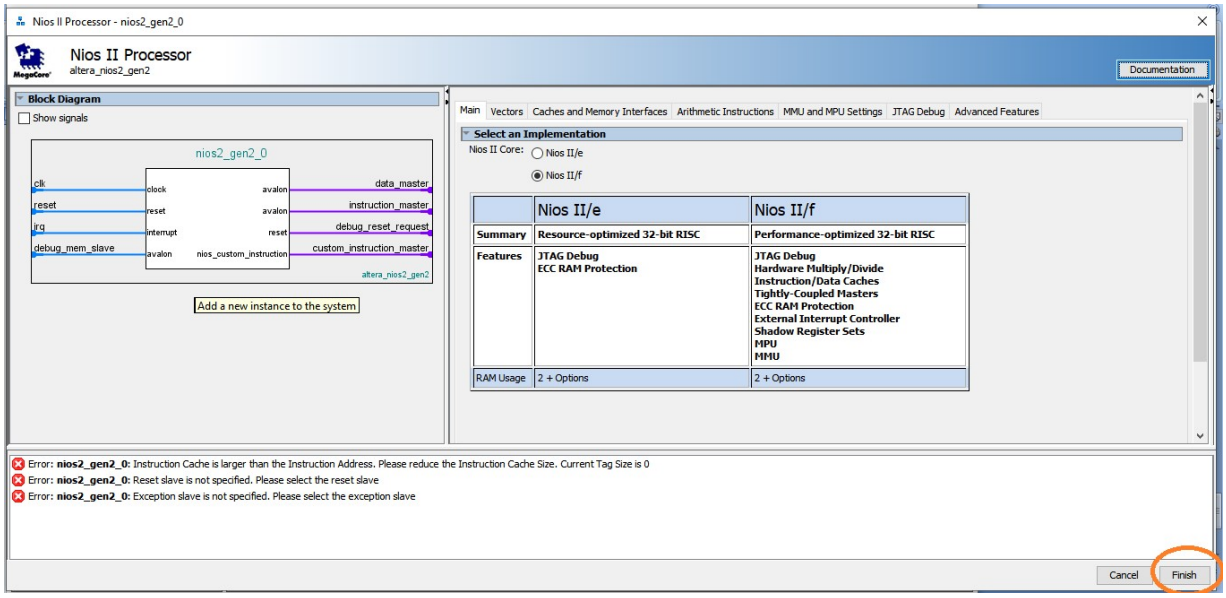
5) Enregistrer le système comme **nios2_computer.qsys**



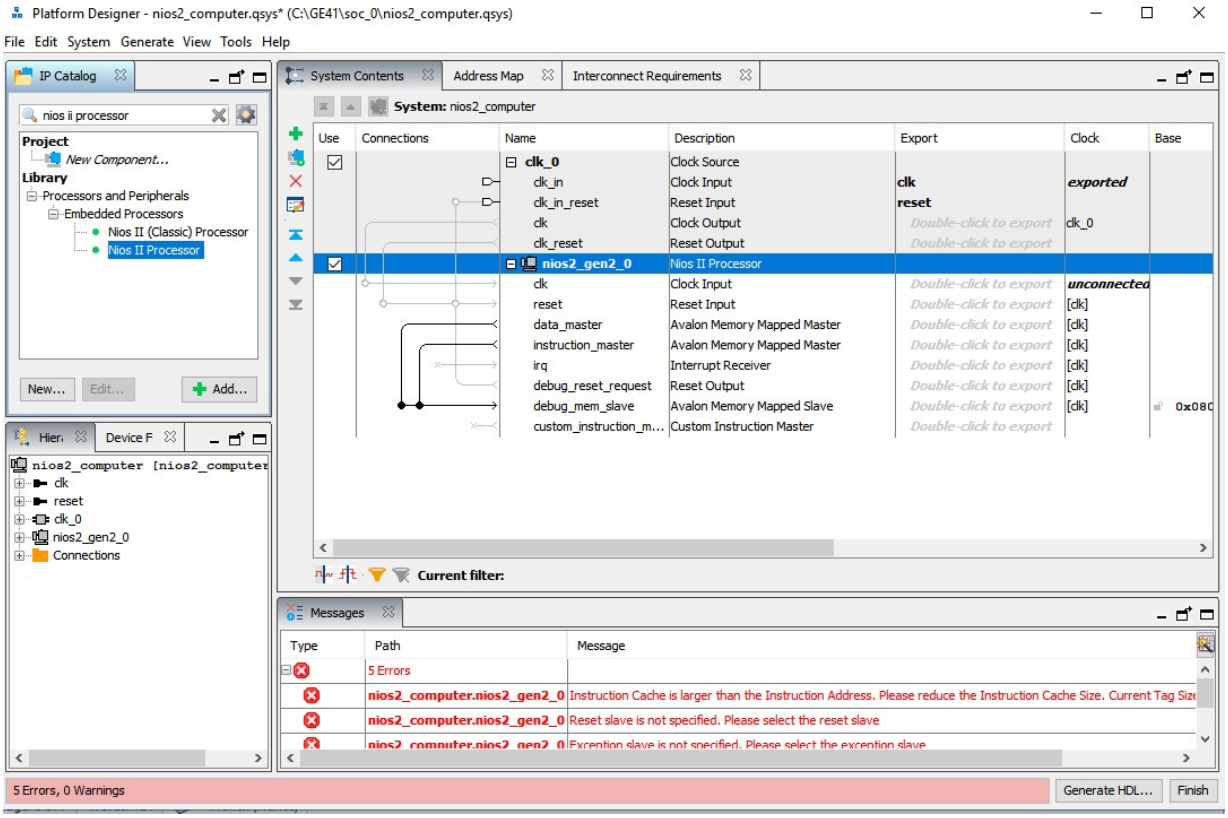
6) Chercher dans IP catalog le processeur « nios ii processor ». Le sélectionner comme indiqué en bleu dans la figure suivante et cliquer sur « Add »



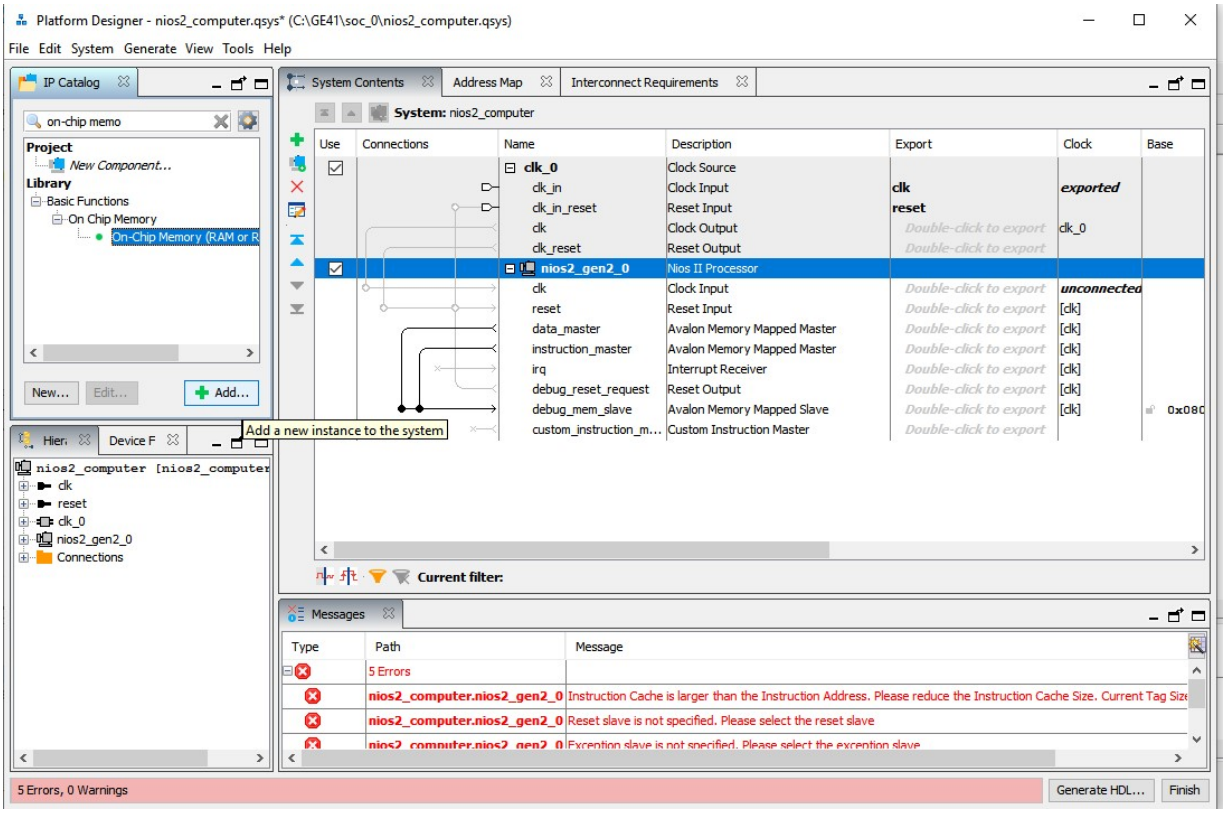
7) La fenêtre de nios ii processor s'ouvre. Cliquer « finish » dans la fenêtre « Nios II Processor »



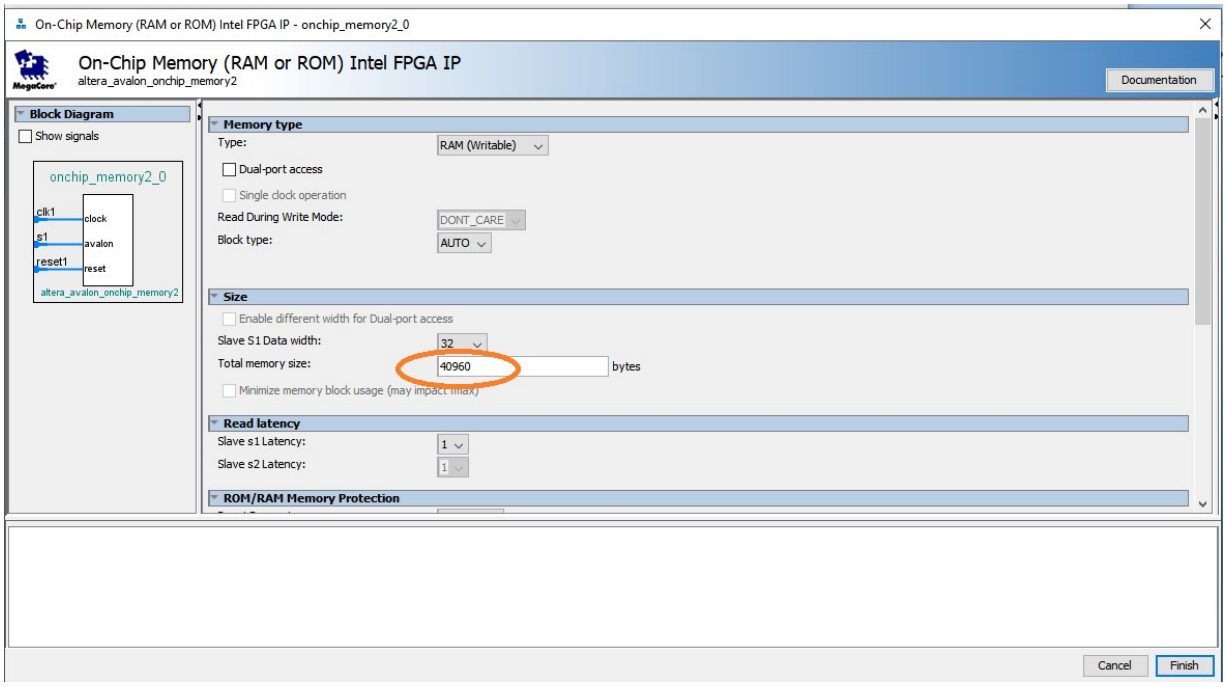
4) Le nios ii s'ajoute au soc comme indiqué sur la figure suivante



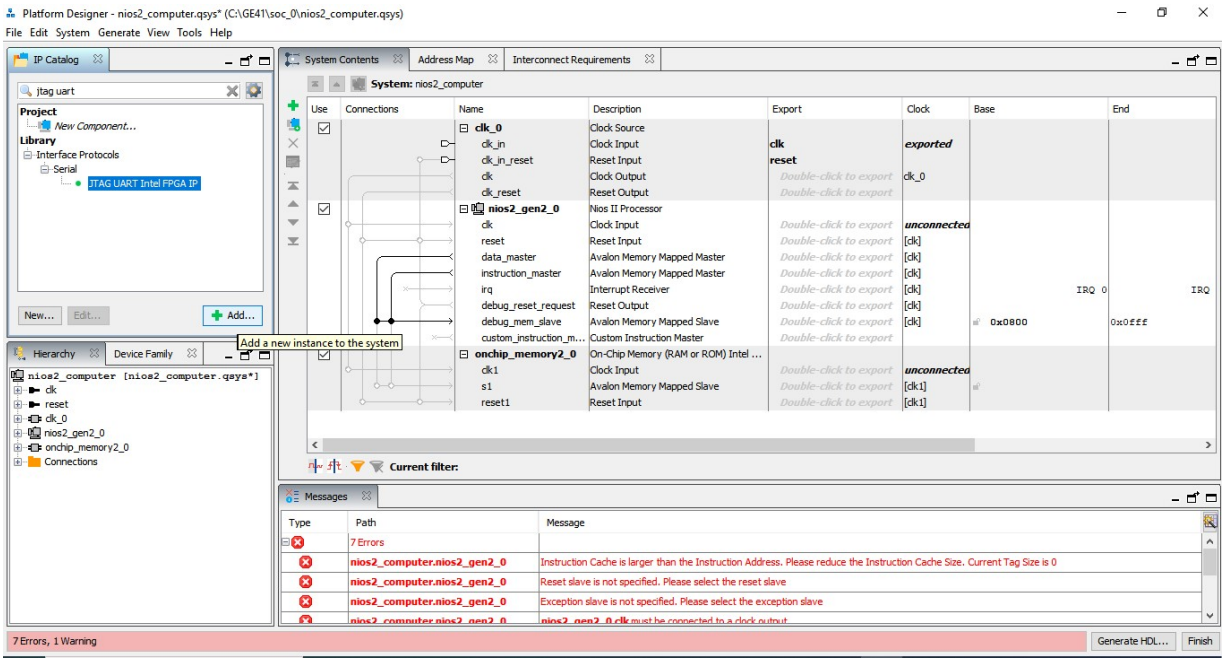
6) Chercher « on_chip memory » dans IP catalog et faire « Add »



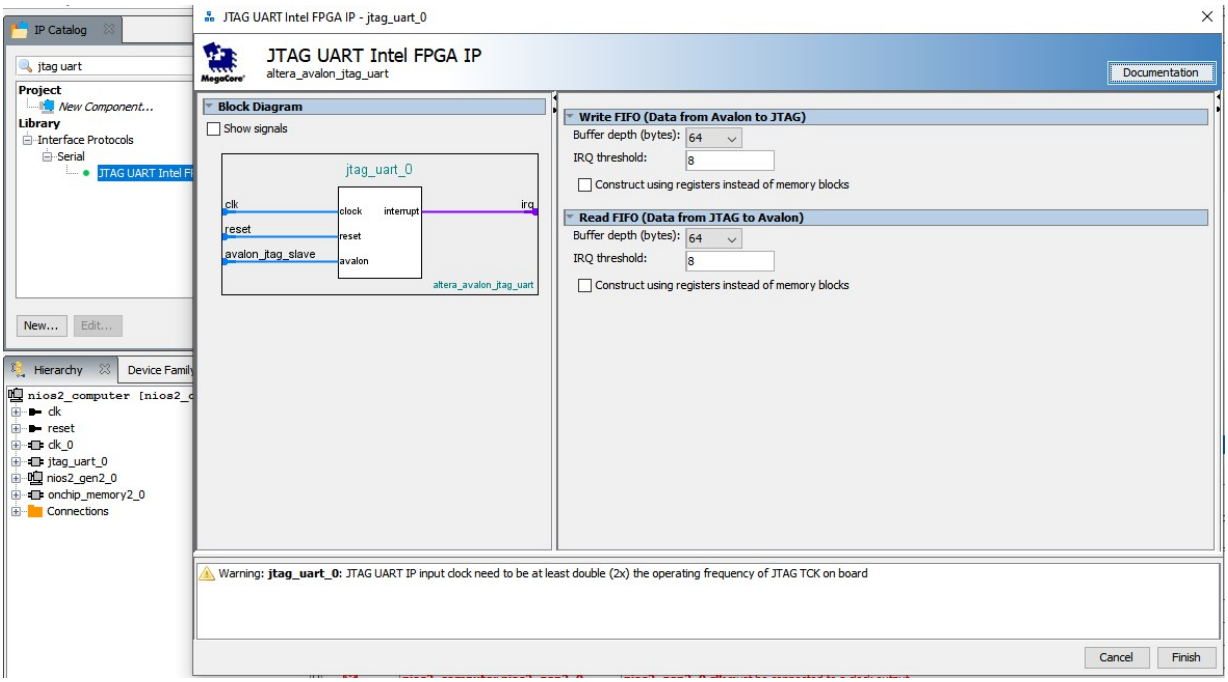
8) Changer « Total memory size » de 4096 byte à 40960 bytes dans la fenêtre « On-chip memory ». Cliquer « Finish » dans la fenêtre « On-chip memory ».



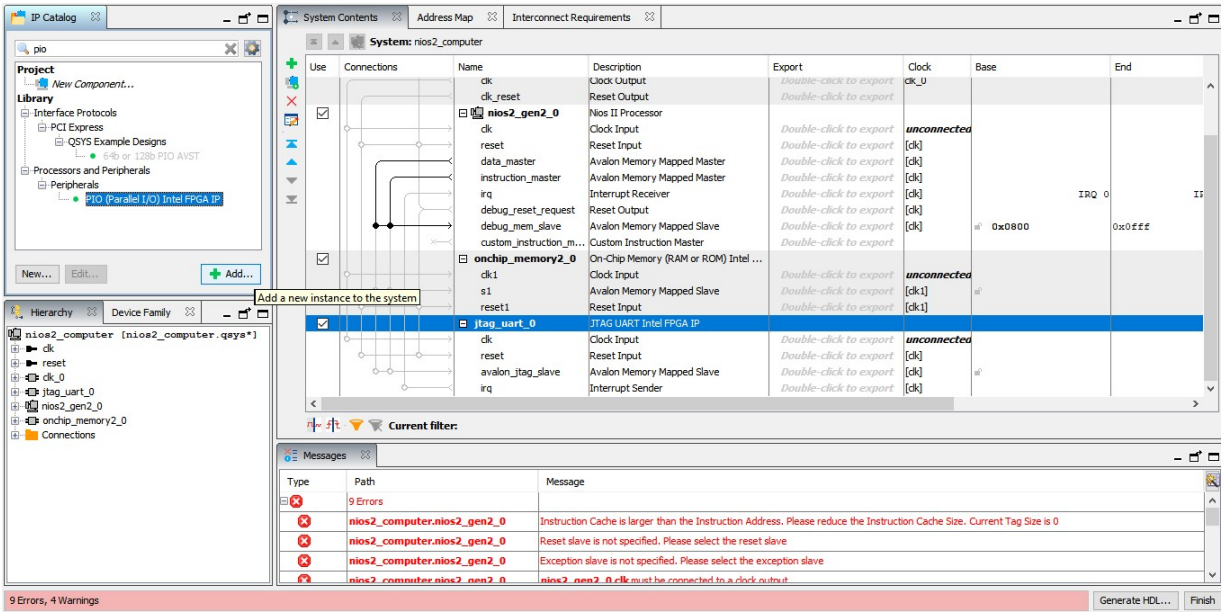
9) Chercher JTAG UART dans « IP Catalog » et faire « Add »



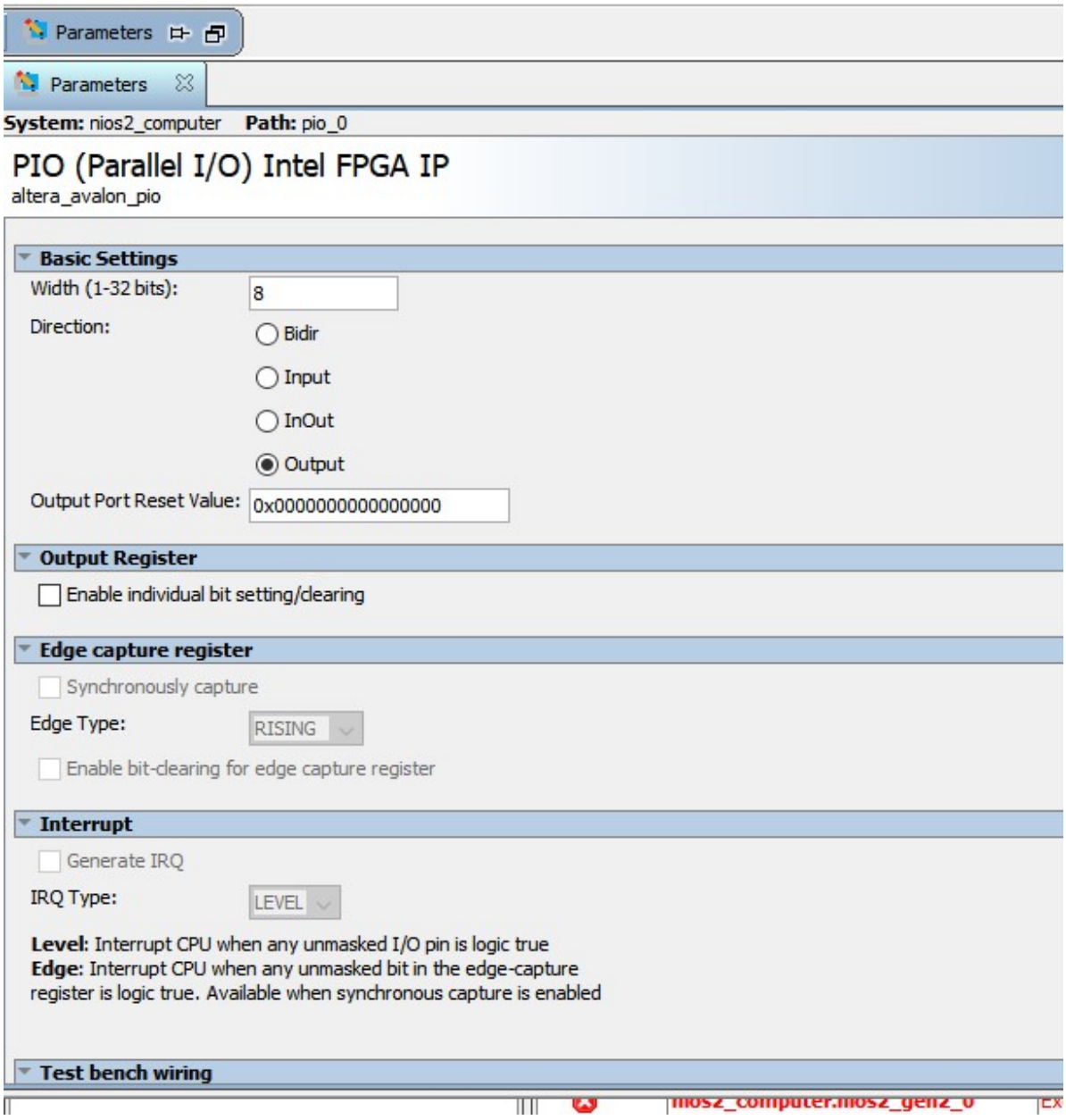
8) Ne rien changer. Cliquer « finish » dans la fenêtre JTAG UART



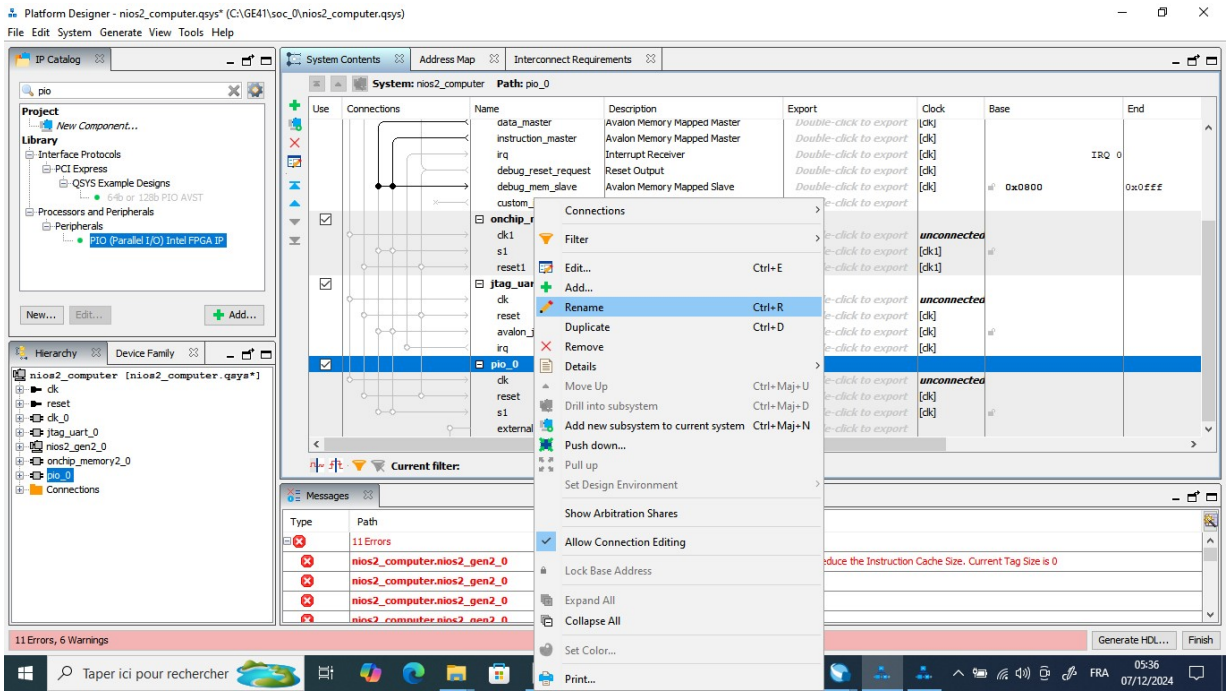
10) Chercher pio dans IP catalog



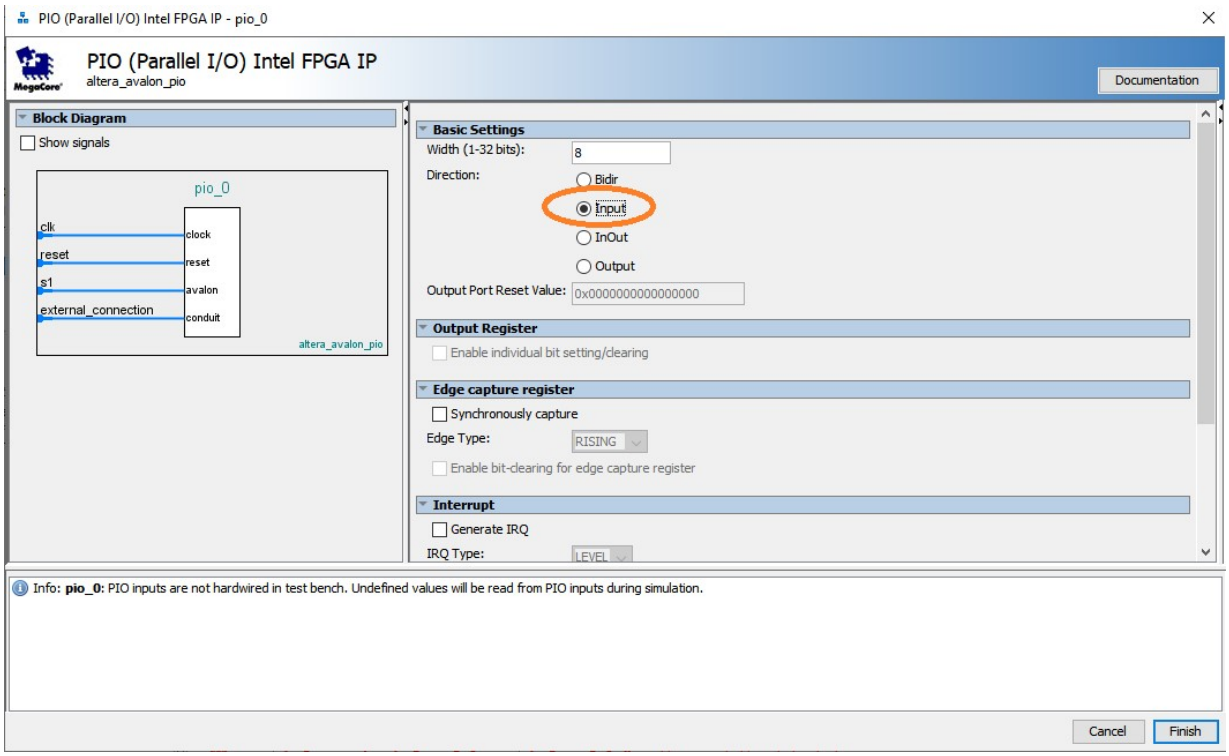
11) Ne rien changer et cliquer « Finish » sur la fenêtre PIO



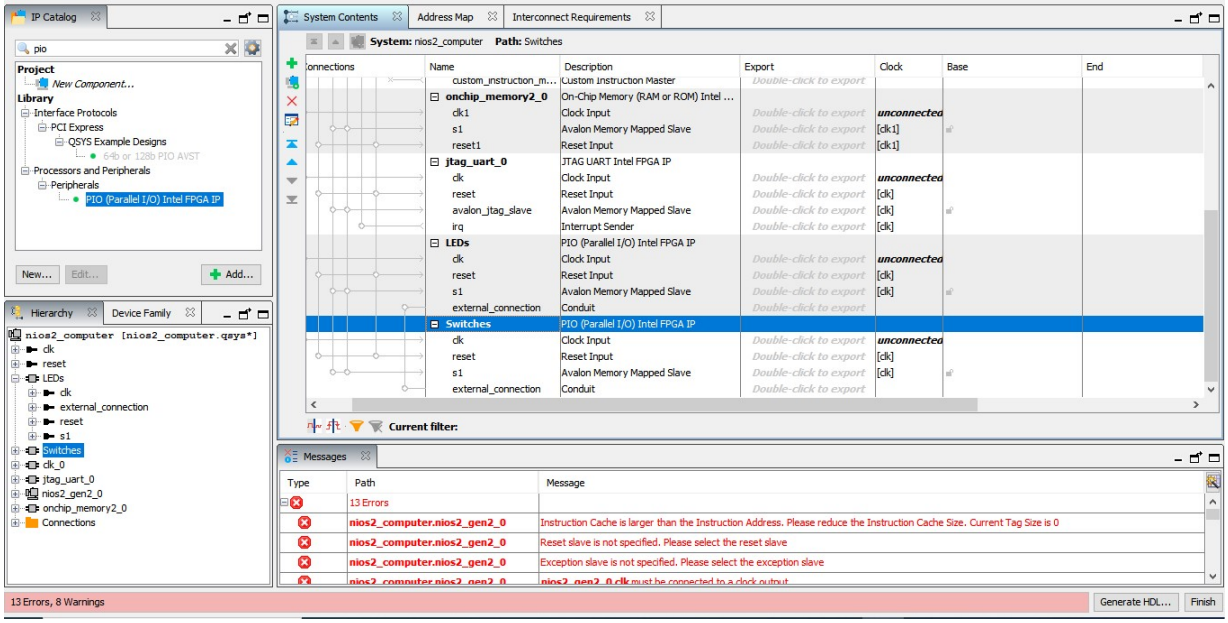
12) Changer le nom de pio_0 à LEDs



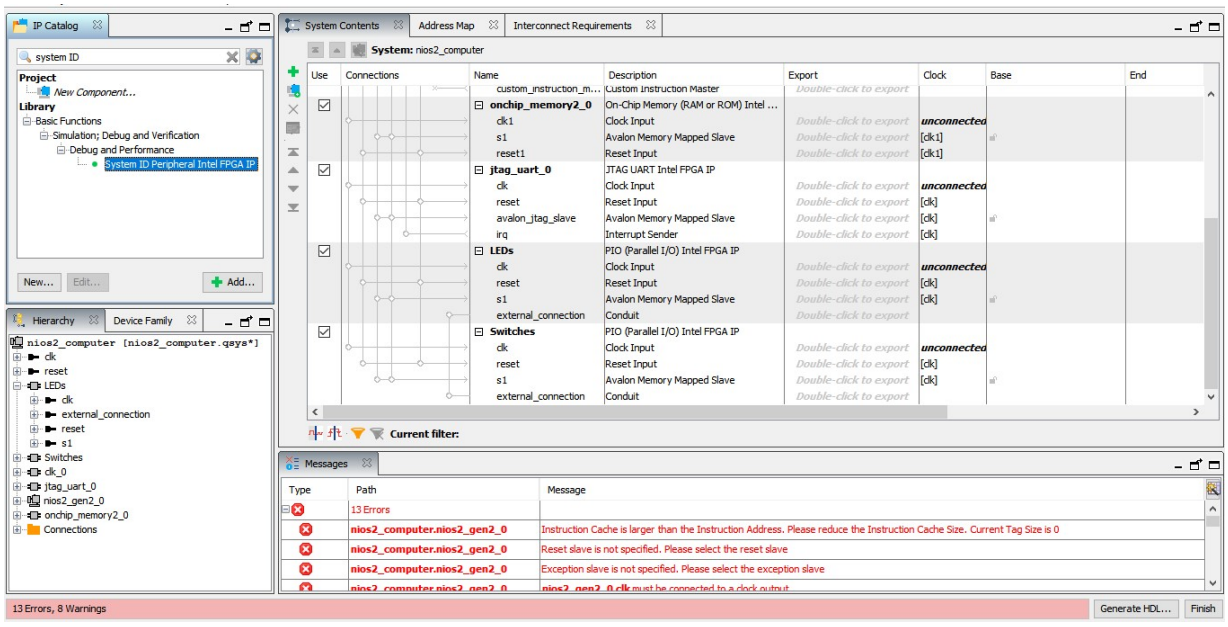
13) Cliquer « Add ». Choisir ce pio comme Input. Cliquer sur « Finish » dans la fenêtre pio



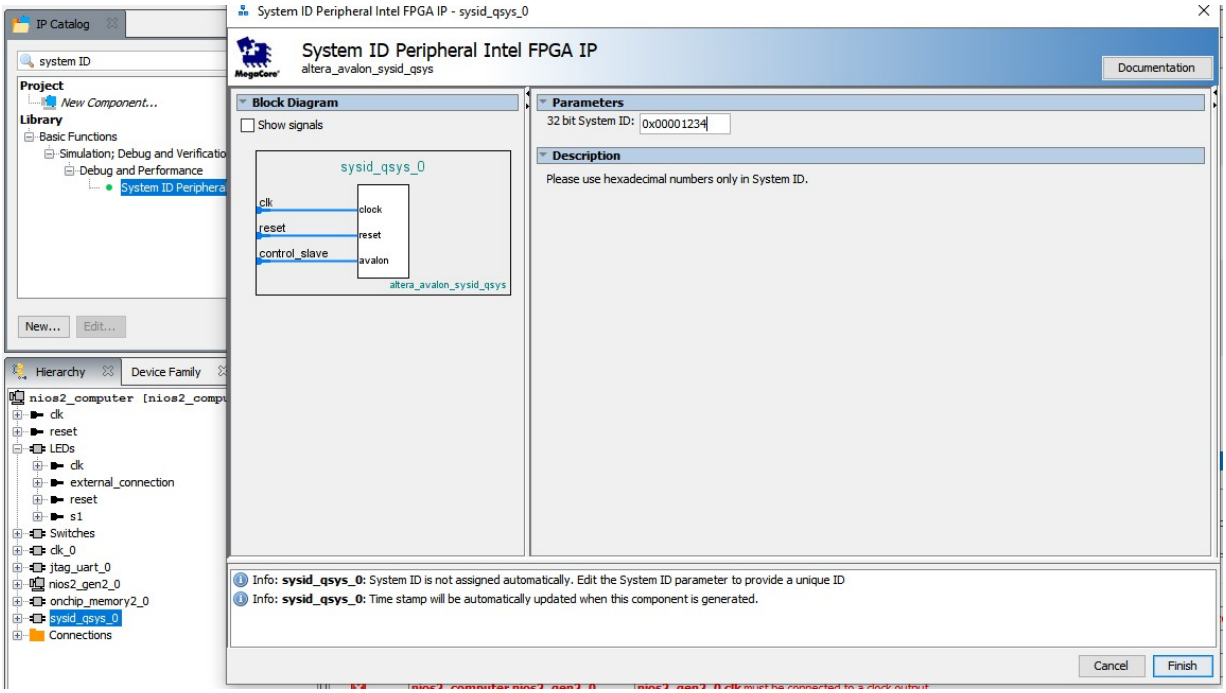
14) Changer le nom de pio_0 à Switches



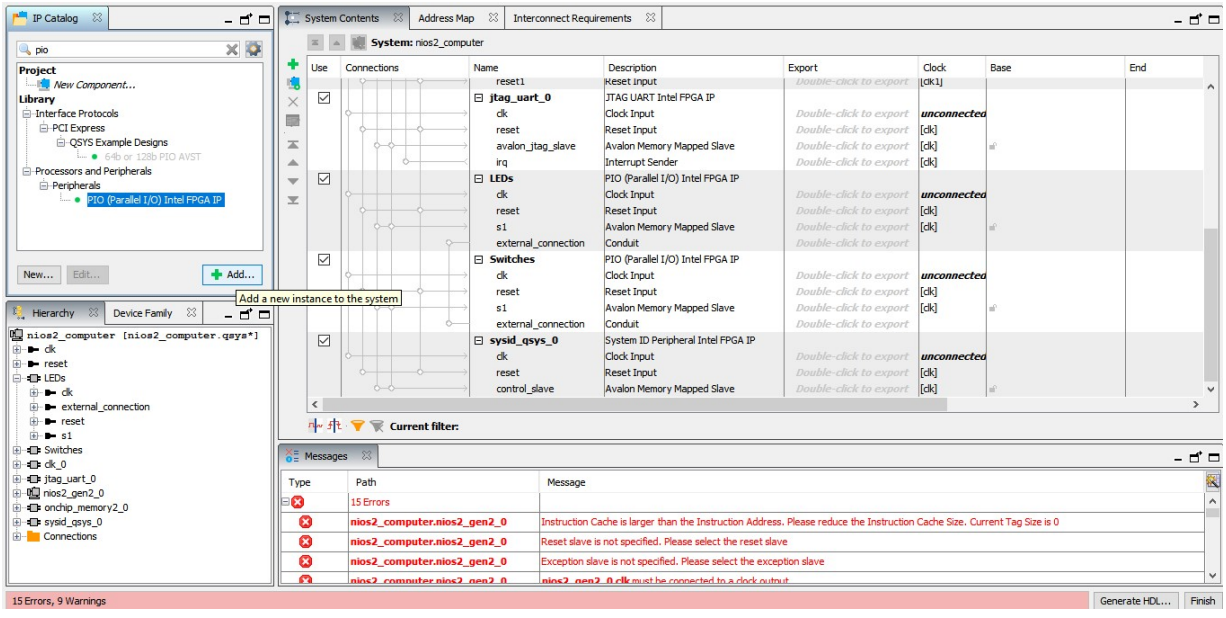
15) Chercher System ID dans IP catalog et cliquer « Add »



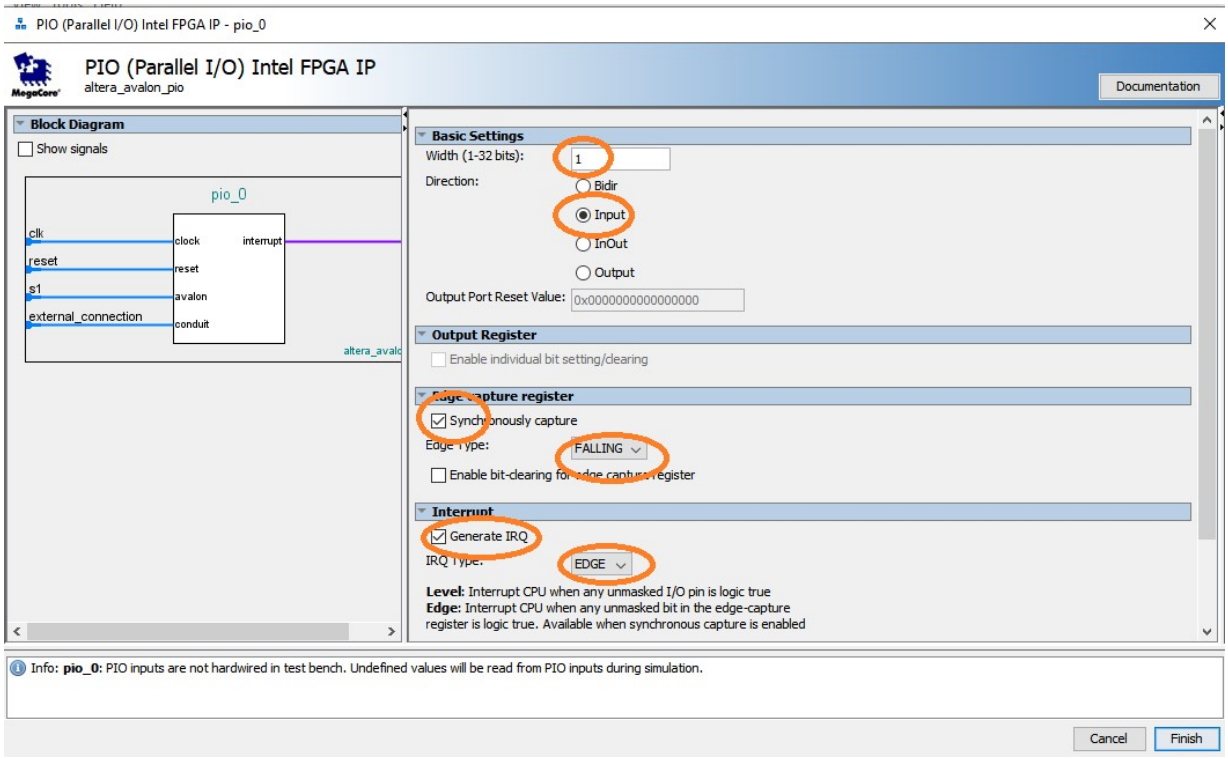
16) Choisir System ID=0x1234. Cliquer « finish » dans la fenêtre System ID



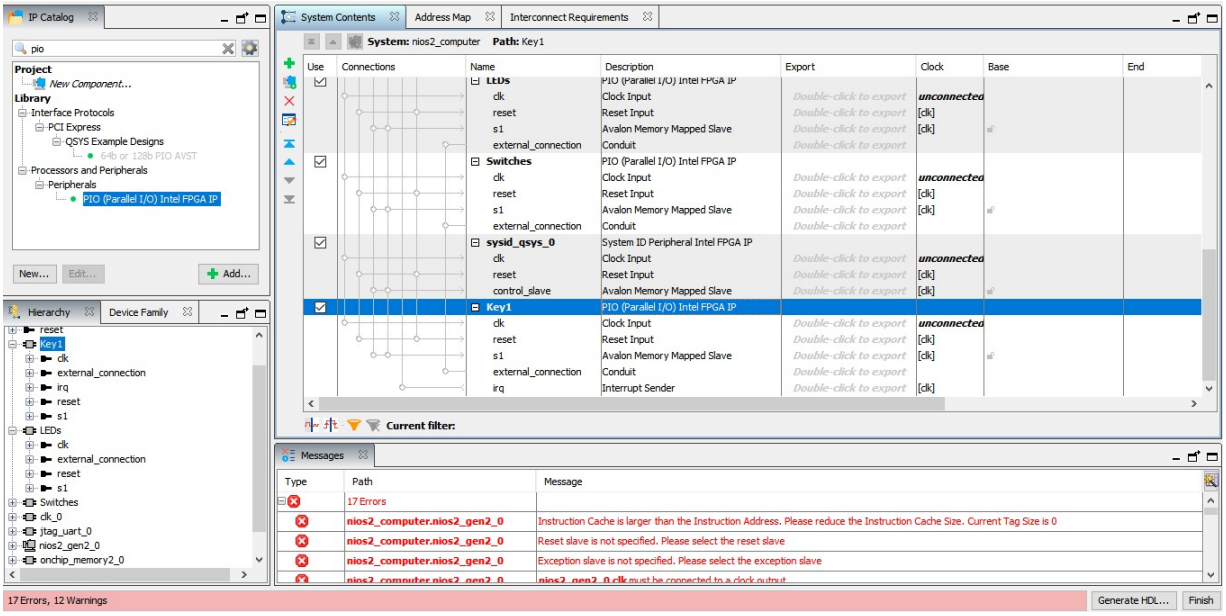
17) Chercher pio dans IP catalog



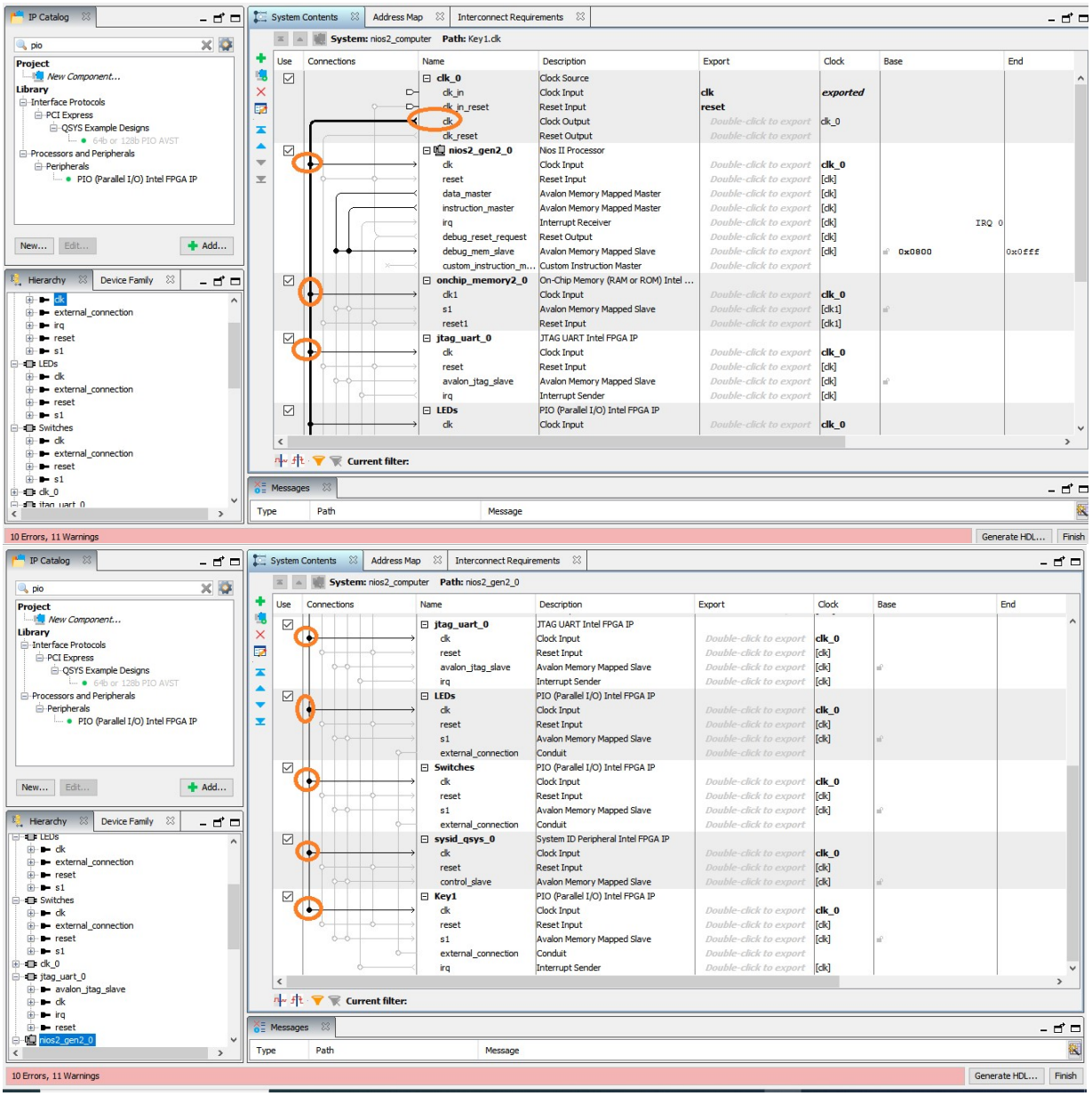
18) Configuer la fenêtre suivante comme indiqué dans la figure suivante



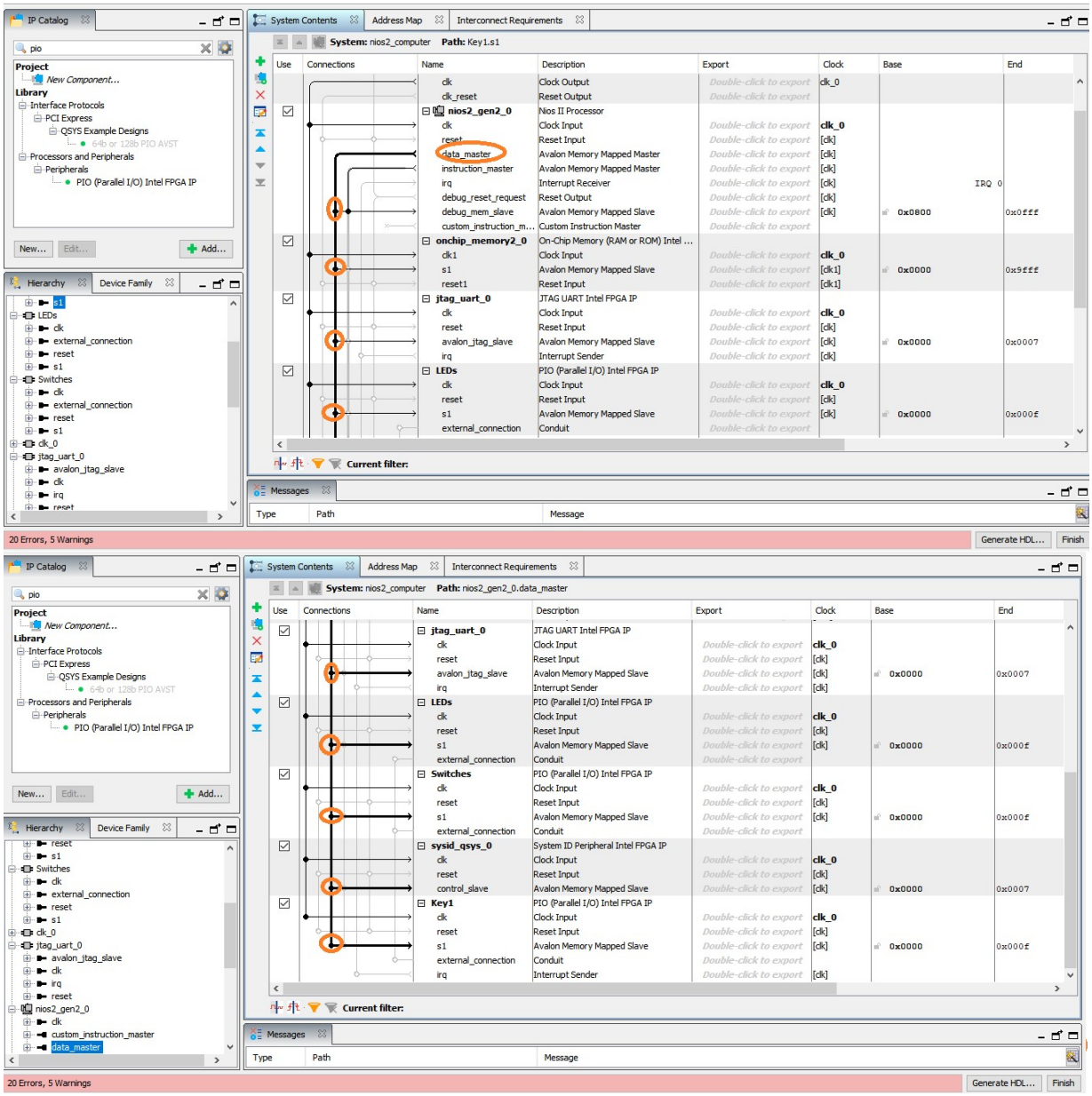
19) Changer le nom de pio_0 à Key1



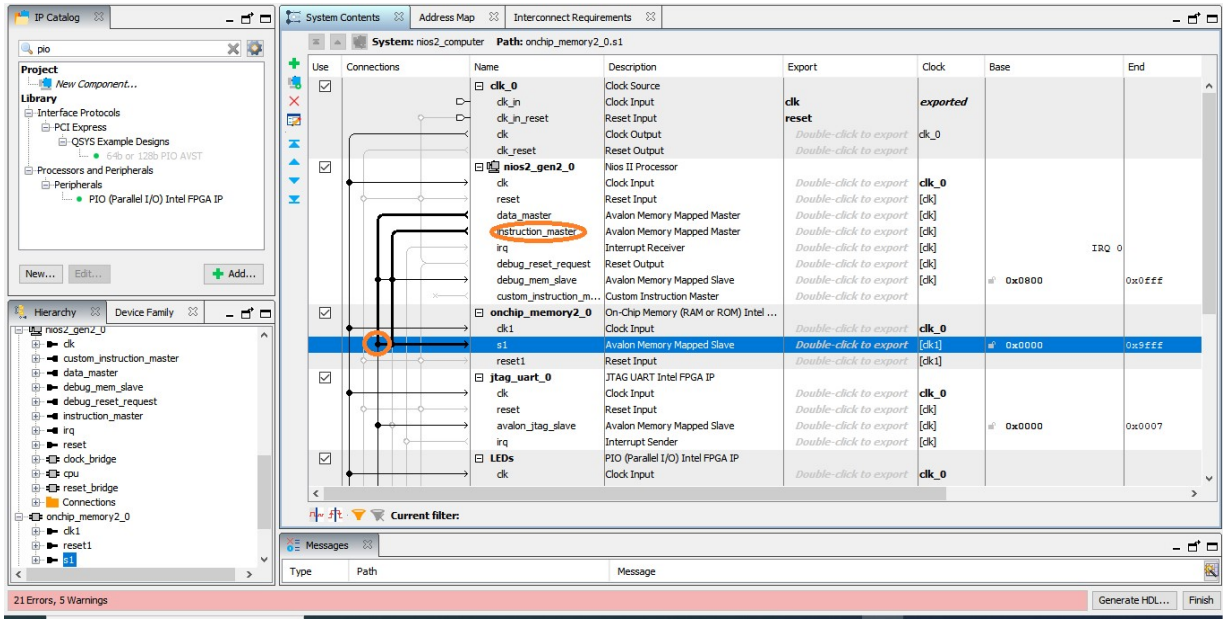
20) Relier le « clk » au nios et à tous les périphériques comme indiqué dans les deux figures suivantes



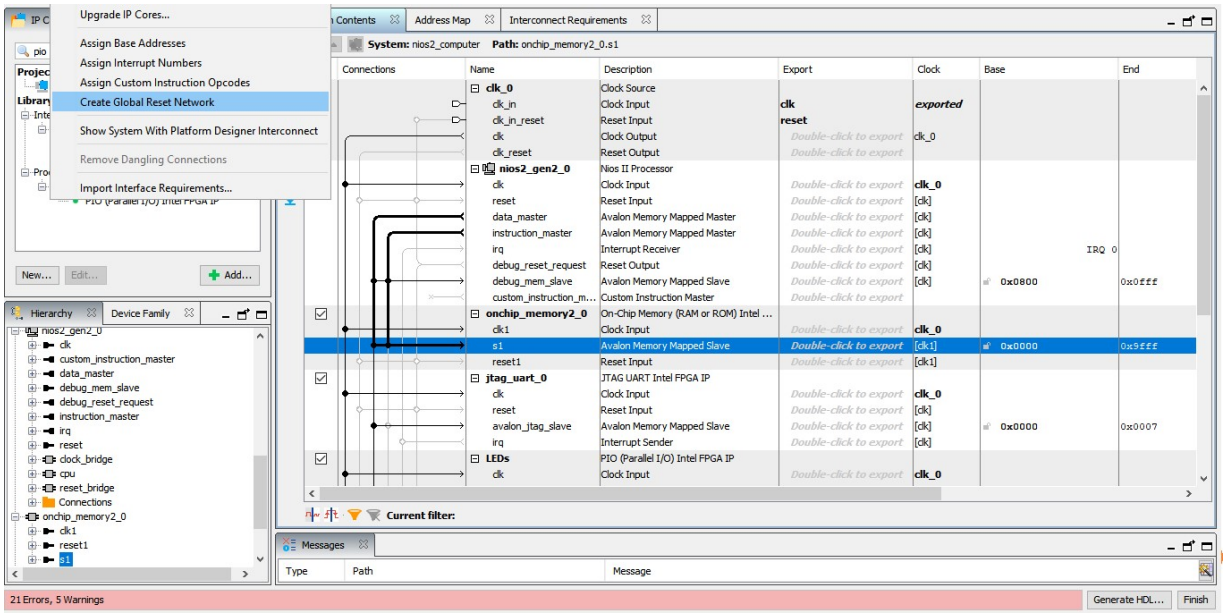
21) Relier le Data master à tous les périphériques



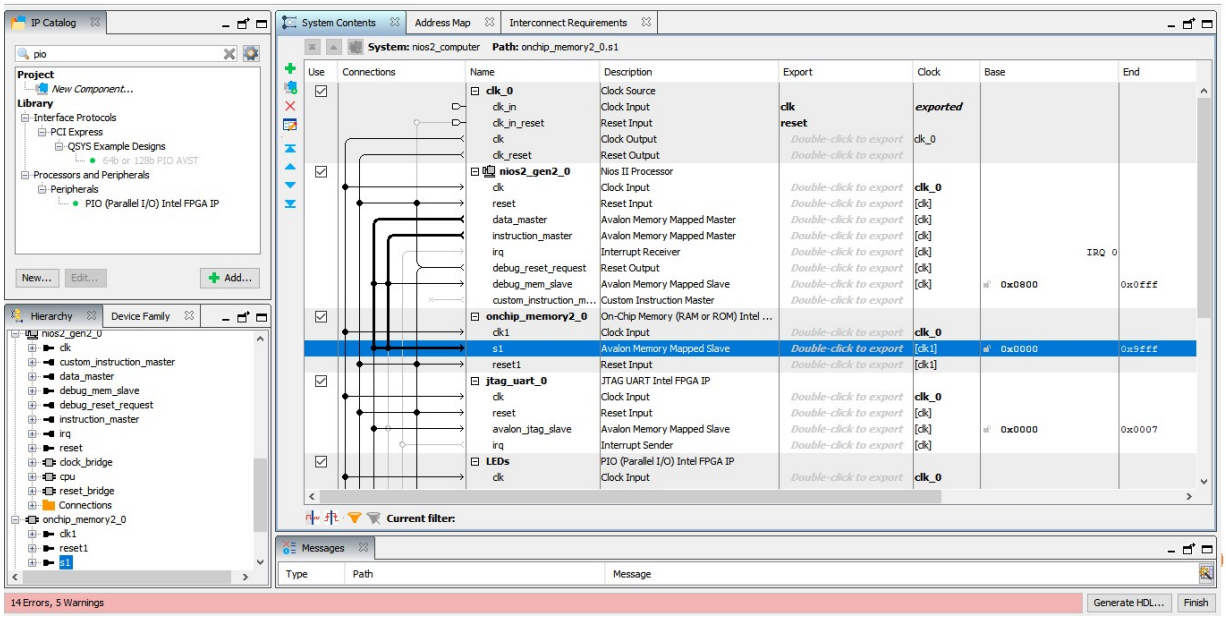
22) Relier Instruction master avec la mémoire



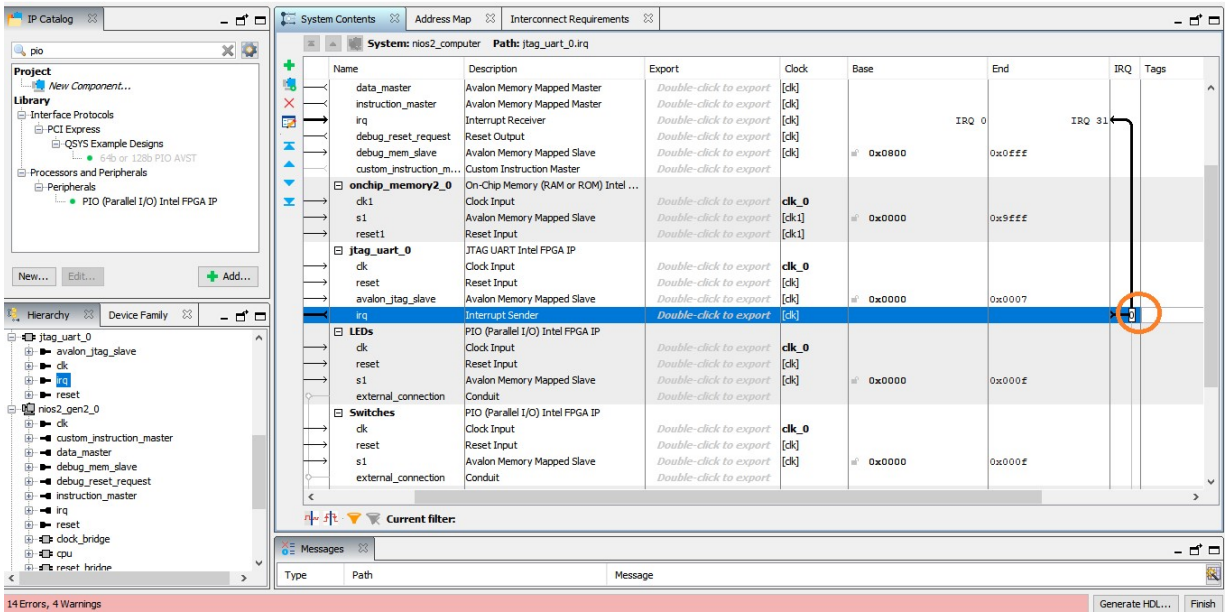
23) Dans la fenêtre System, cliquer « Create Global Reset Network »



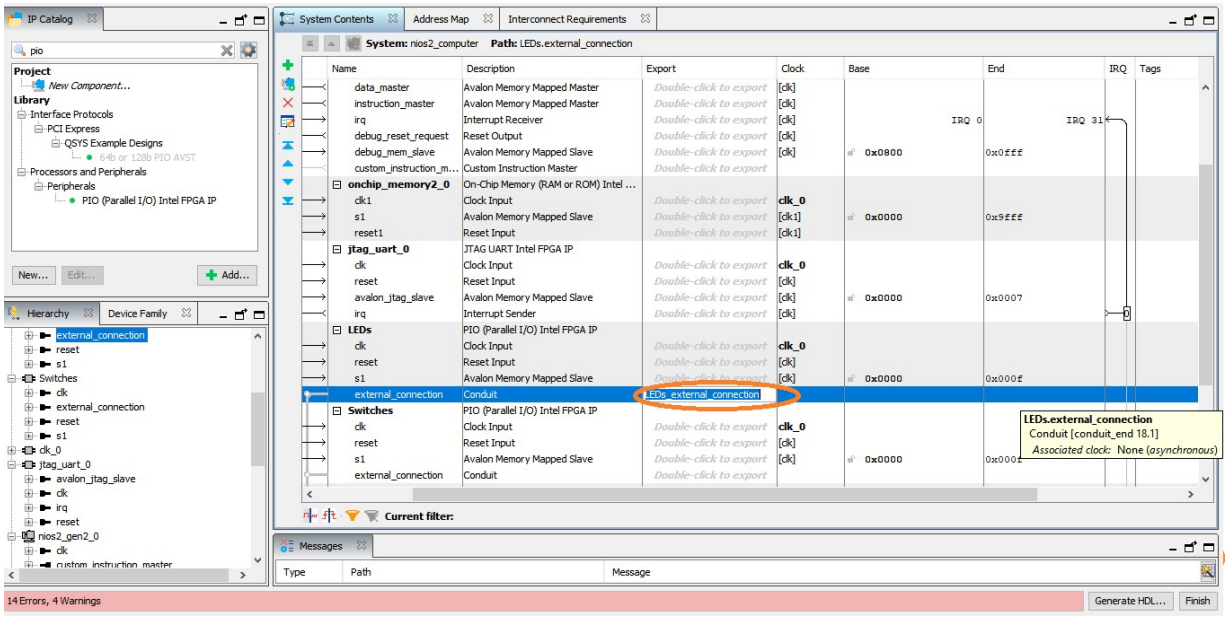
On obtient ceci



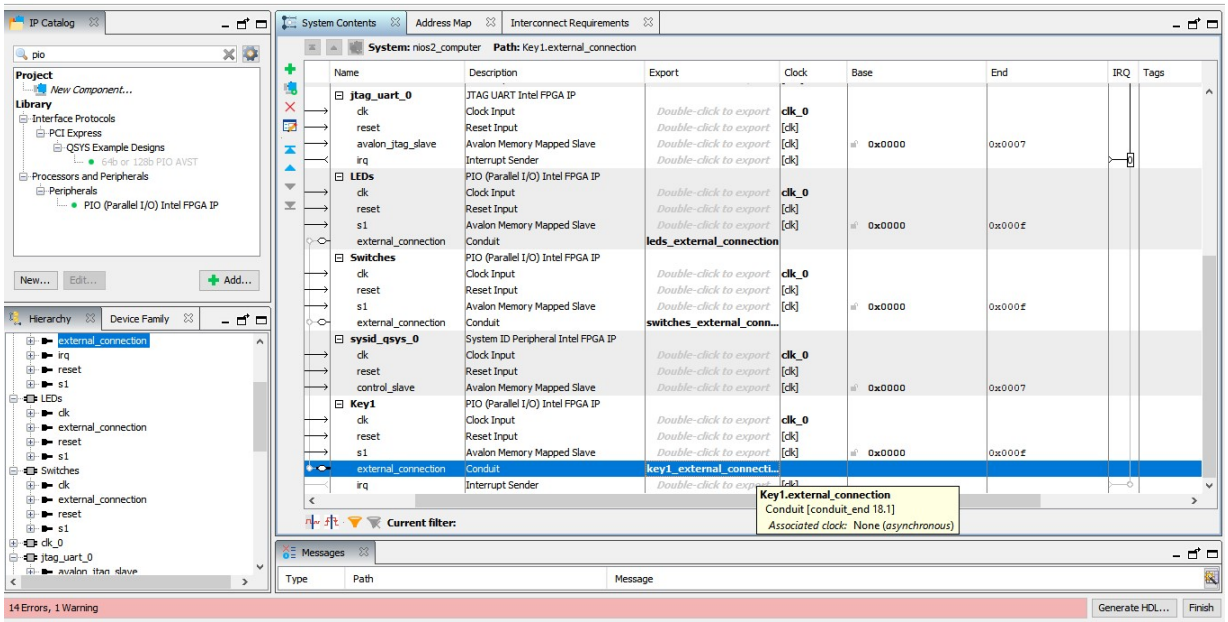
24) Faire la liaison JTAG UART au IRQ en cliquant le petit rond comme indiqué. Il va y avoir la valeur 0



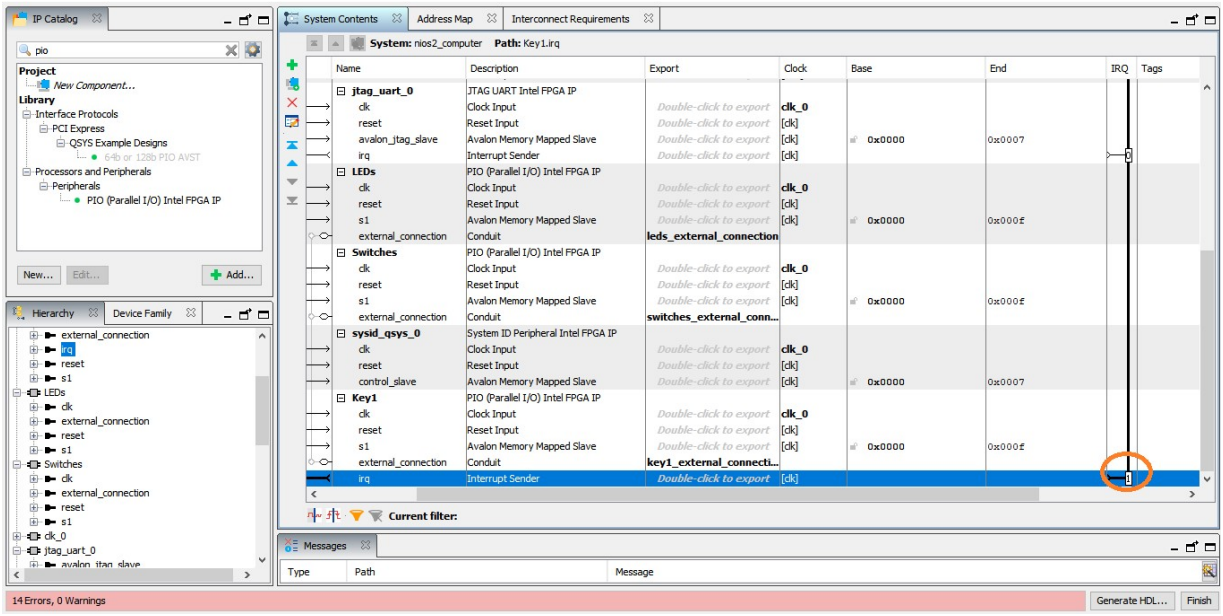
25) Pour le périphérique LEDs, double cliquer à l'intersection « external connexion » et Export comme indiqué dans la figure suivante



26) Faire la même chose pour Switches et Key1

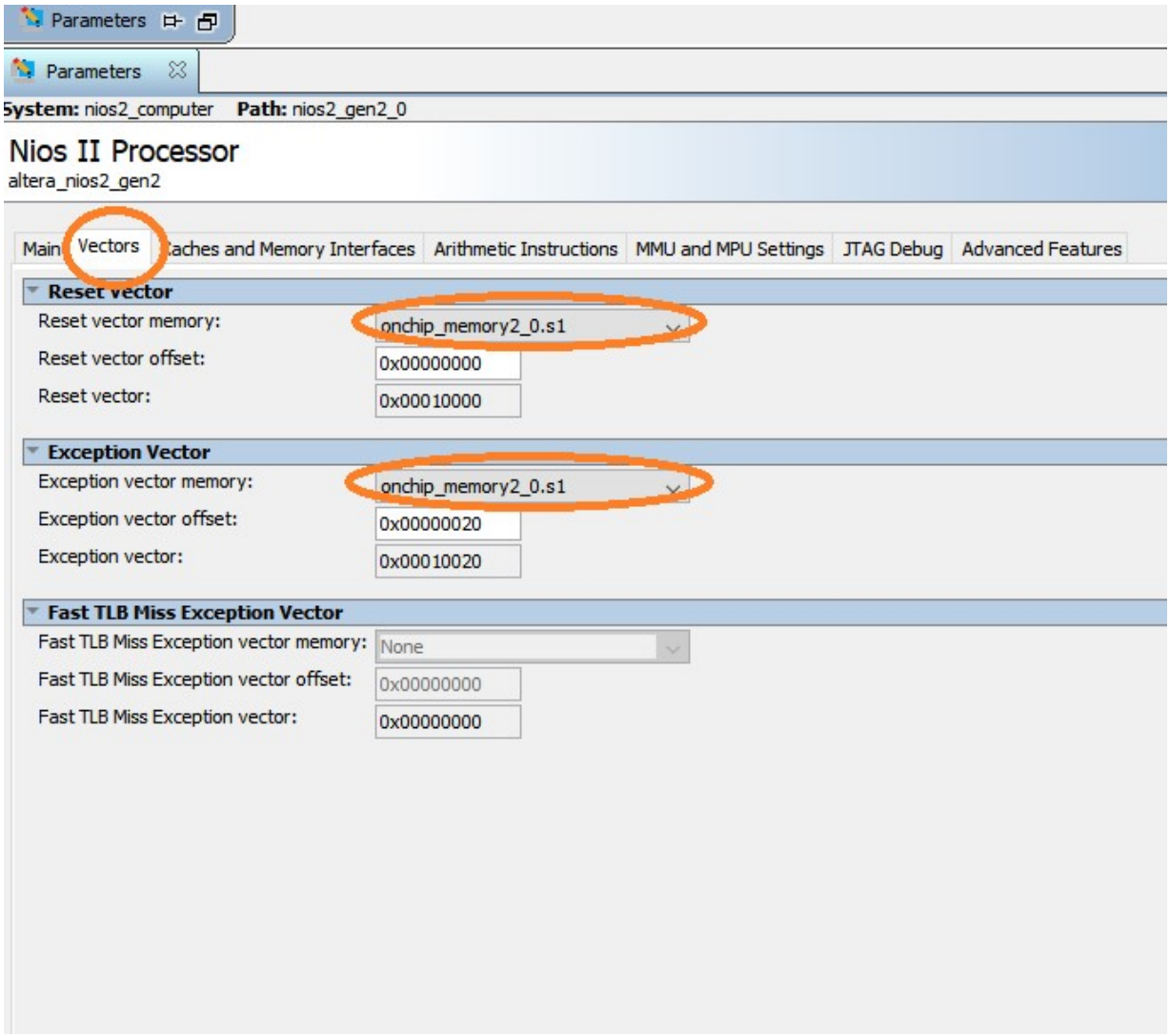


27) Faire la liaison Key1 au IRQ en cliquant le petit rond comme indiqué

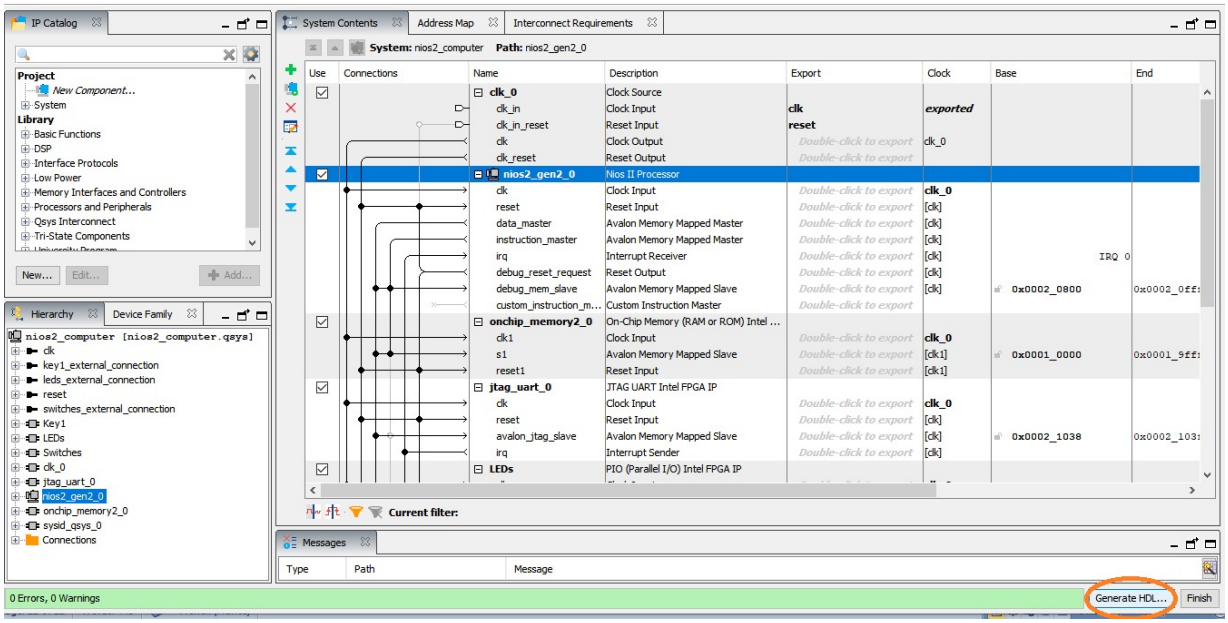


28) Dans la fenêtre System, cliquer « Assign Base Address »

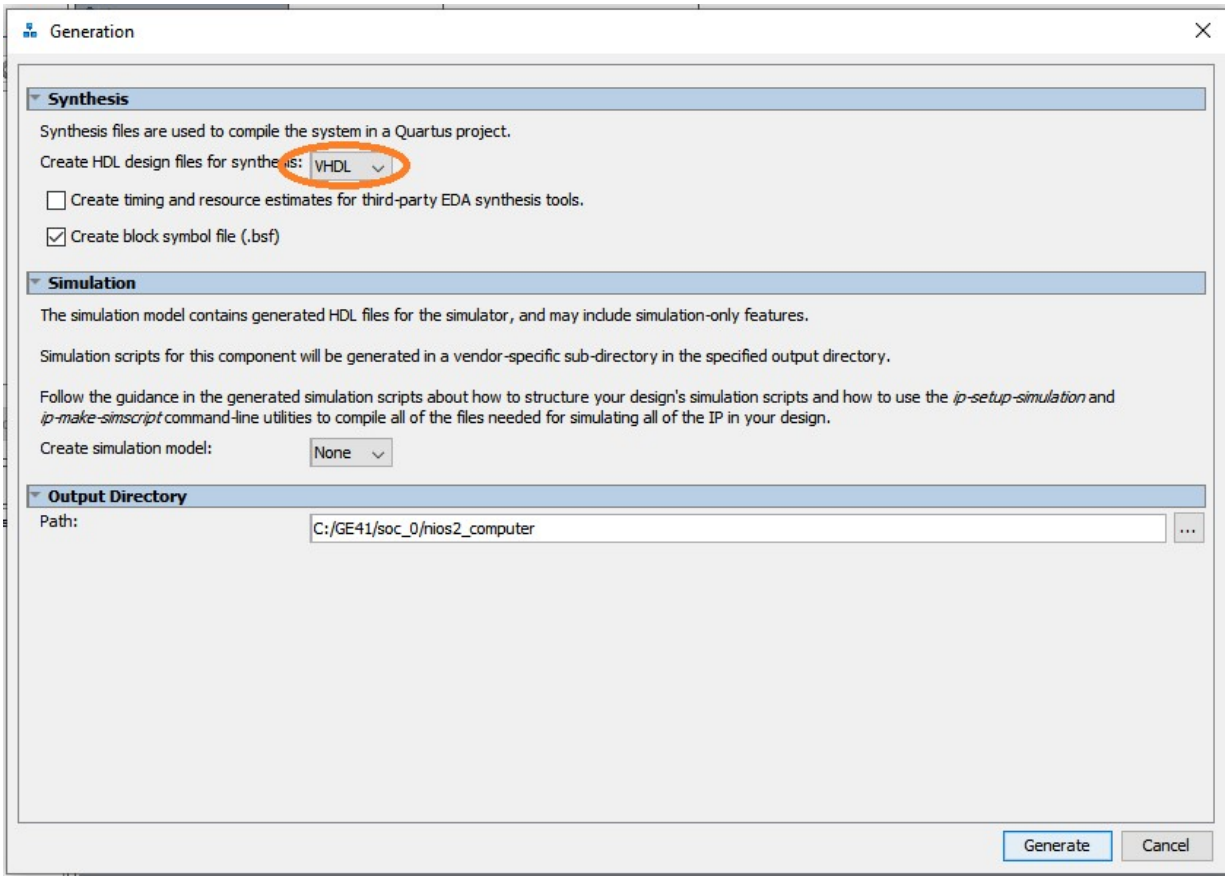
29) Double cliquer sur le « nios2_gen2_0 » et le configuer comme indiqué dans la figure suivante. Fermer la fenêtre après configuration



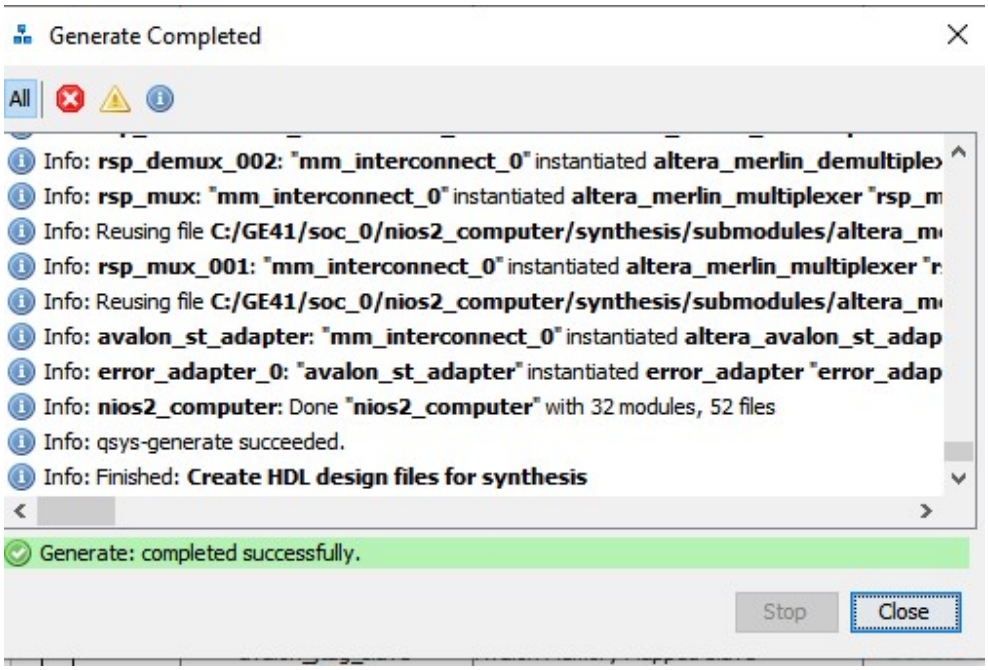
30) Cliquer « Generate HDL »



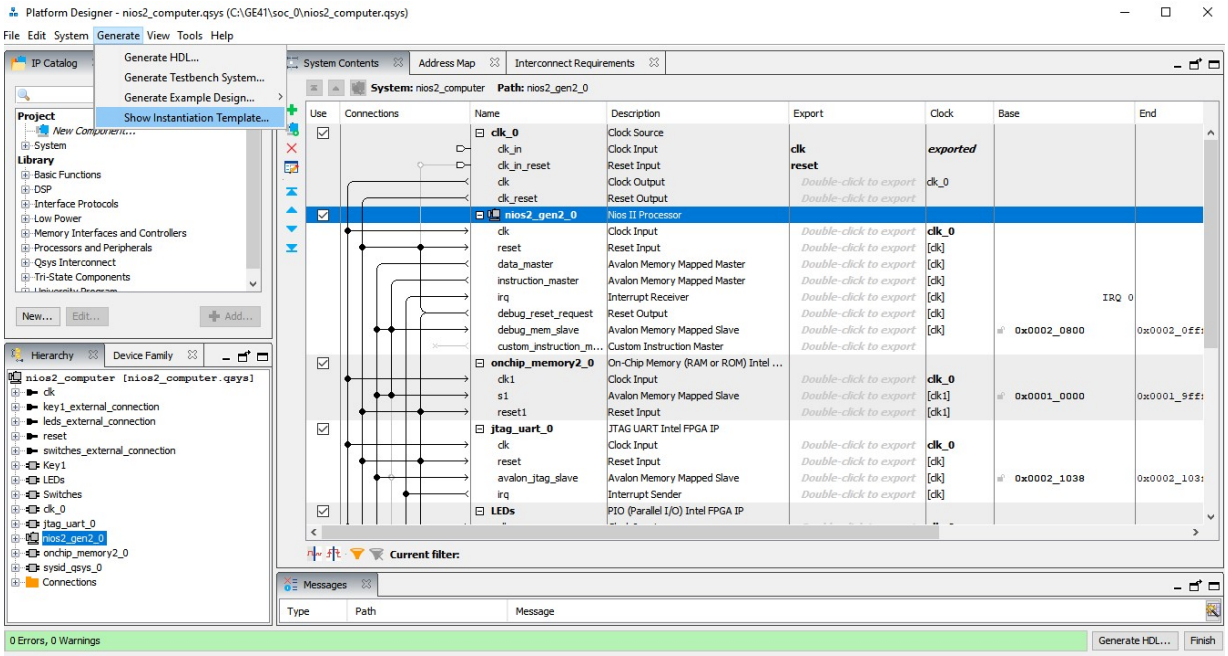
31) Choisir VHDL dans la figure suivante et cliquer « Generate »



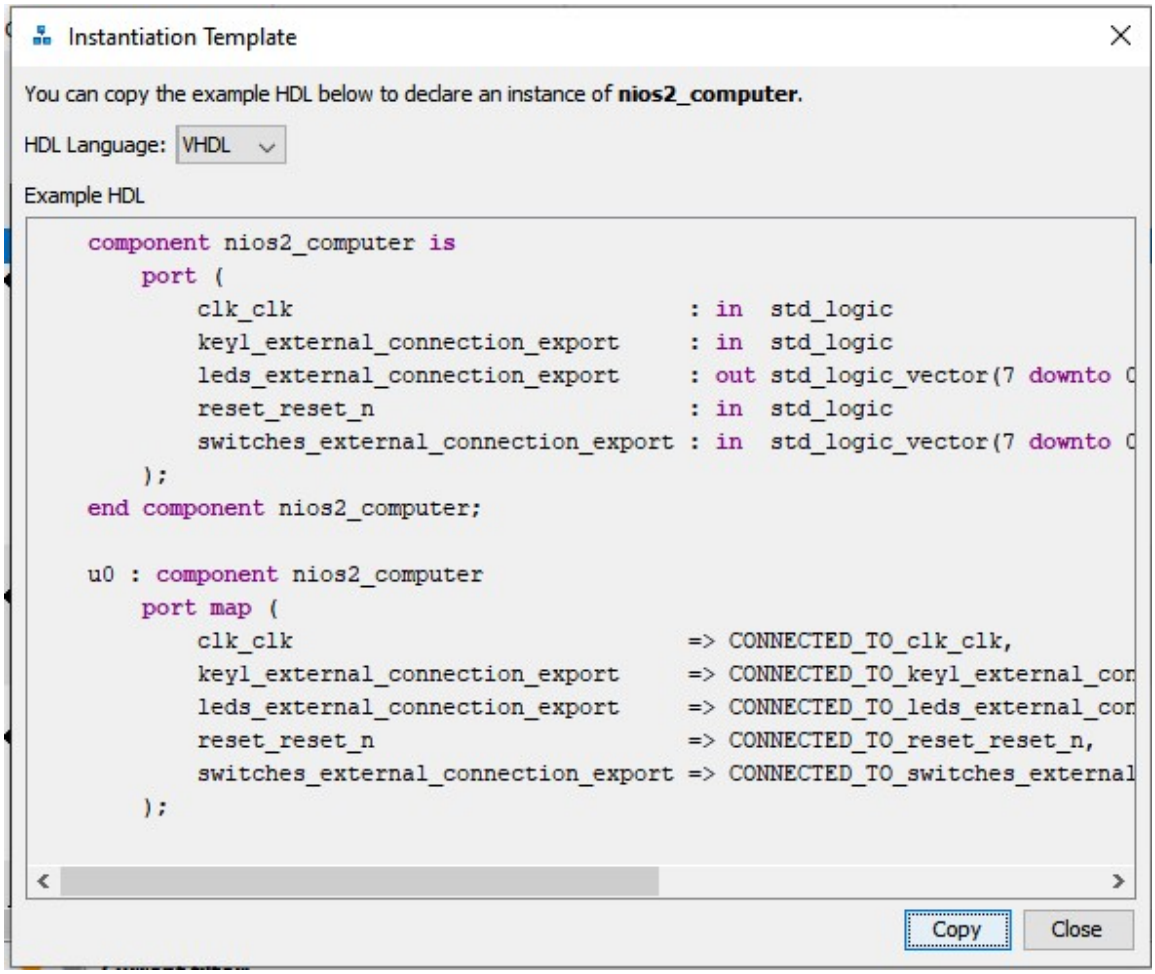
32) Après la génération du VHDL, la fenêtre suivante s'affiche. Cliquer « Close »



33) Dans la fenêtre « Generate », cliquer « Show Instantiation Template »



34) Cliquer « Copy »

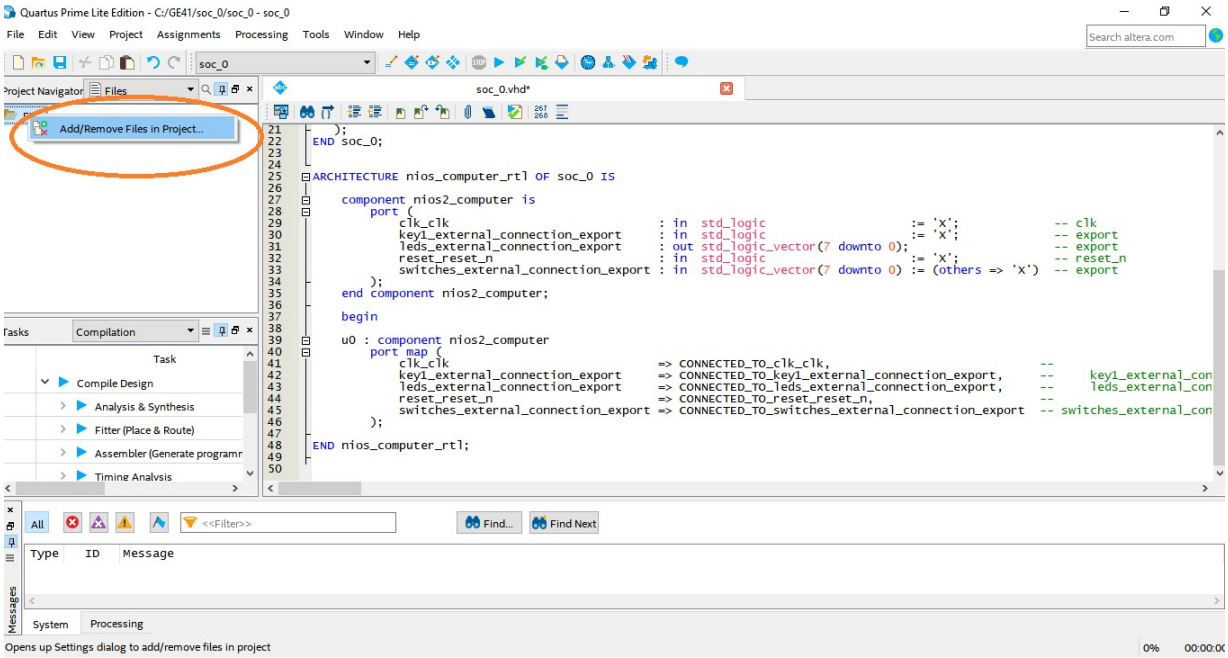


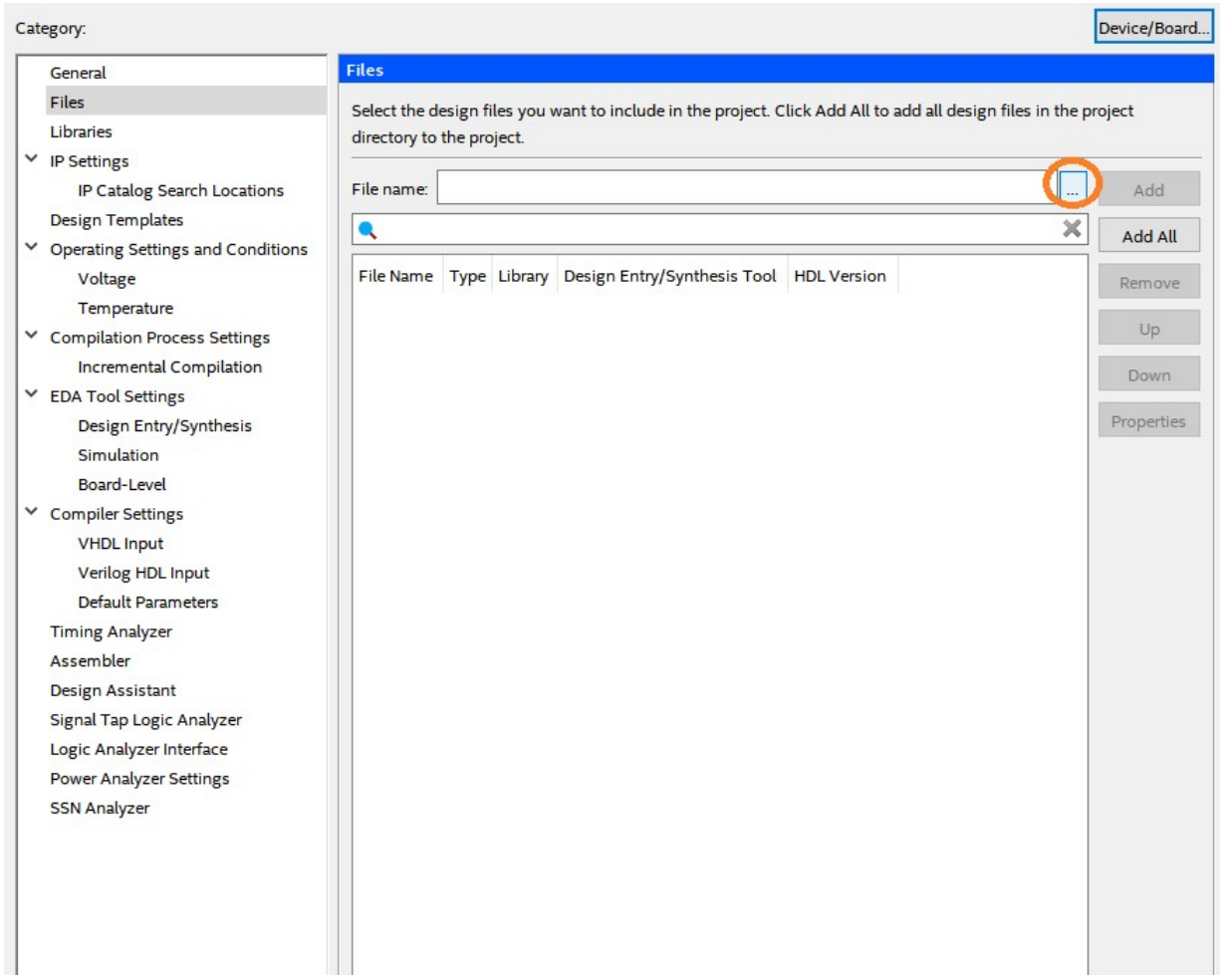
35) Ouvrir le fichier soc_0.vhd. Copier le code VHDL dans le fichier.

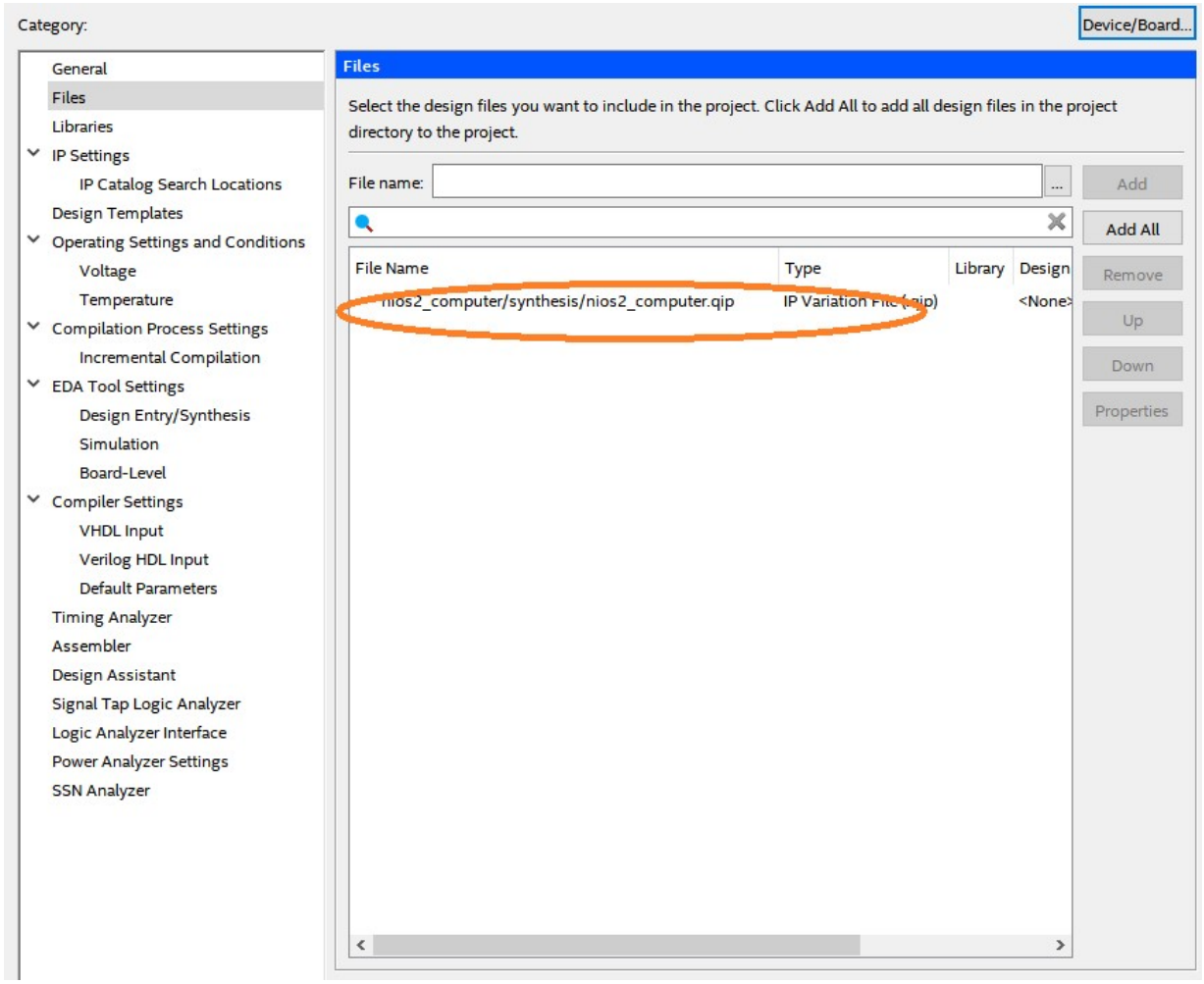
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

ENTITY soc_0 IS

PORT (
        -- Inputs
        CLOCK_50: IN STD_LOGIC;
        KEY     : IN STD_LOGIC_VECTOR (1 DOWNTO 0);

        SW          : IN STD_LOGIC_VECTOR (7 DOWNTO 0);

        -- Outputs

        LEDR: OUT STD_LOGIC_VECTOR (7 DOWNTO 0);
```

```
            HEX0: OUT STD_LOGIC_VECTOR (6 DOWNTO 0);
            HEX1: OUT STD_LOGIC_VECTOR (6 DOWNTO 0);
            HEX2: OUT STD_LOGIC_VECTOR (6 DOWNTO 0);
            HEX3: OUT STD_LOGIC_VECTOR (6 DOWNTO 0)
            );
    END soc_0;


ARCHITECTURE nios_computer_rtl OF soc_0 IS

  component nios2_computer is
    port (
      clk_clk       : in  std_logic                     := 'X';           -- clk
      key1_external_connection_export    : in  std_logic                     := 'X';           -- export
      leds_external_connection_export    : out std_logic_vector(7 downto 0);                   -- export
      reset_reset_n                 : in  std_logic                     := 'X';           -- reset_n
      switches_external_connection_export : in   std_logic_vector(7 downto 0) := (others => 'X')   --
export
    );
  end component nios2_computer;

        begin

  u0 : component nios2_computer
    port map (
 clk_clk                 => CONNECTED_TO_clk_clk,                    --                 clk.clk
key1_external_connection_export               =>   CONNECTED_TO_key1_external_connection_export,
leds_external_connection_export               =>   CONNECTED_TO_leds_external_connection_export,
reset_reset_n                                 => CONNECTED_TO_reset_reset_n,
switches_external_connection_export => CONNECTED_TO_switches_external_connection_export       );

END nios_computer_rtl;
```

36) Ajouter le fichier nios2_computer.qip dans le projet Quartus (C:\GE41\soc_0\nios2_computer\synthesis)


Quartus Prime Lite Edition - C:/GE41/soc_0/soc_0 - soc_0
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
soc_0
Project Navigator
Files
Add/Remove Files in Project...
soc_0.vhd*
Tasks
Compilation
Task
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate programr
Timing Analysis
);
END soc_0;
ARCHITECTURE nios_computer_rtl OF soc_0 IS
component nios2_computer is
port (
clk_clk : in std_logic := 'X'; -- clk
key1_external_connection_export : in std_logic := 'X'; -- export
leds_external_connection_export : out std_logic_vector(7 downto 0); -- export
reset_reset_n : in std_logic := 'X'; -- reset_n
switches_external_connection_export : in std_logic_vector(7 downto 0) := (others => 'X') -- export
);
end component nios2_computer;
begin
u0 : component nios2_computer
port map (
clk_clk => CONNECTED_TO_clk_clk,
key1_external_connection_export => CONNECTED_TO_key1_external_connection_export,
leds_external_connection_export => CONNECTED_TO_leds_external_connection_export,
reset_reset_n => CONNECTED_TO_reset_reset_n,
switches_external_connection_export => CONNECTED_TO_switches_external_connection_export
);
END nios_computer_rtl;
All
<<Filter>>
Find...
Find Next
Type ID Message
Messages
System
Processing
Opens up Settings dialog to add/remove files in project
0%


Category:
Device/Board...
General
Files
Libraries
IP Settings
IP Catalog Search Locations
Design Templates
Operating Settings and Conditions
Voltage
Temperature
Compilation Process Settings
Incremental Compilation
EDA Tool Settings
Design Entry/Synthesis
Simulation
Board-Level
Compiler Settings
VHDL Input
Verilog HDL Input
Default Parameters
Timing Analyzer
Assembler
Design Assistant
Signal Tap Logic Analyzer
Logic Analyzer Interface
Power Analyzer Settings
SSN Analyzer
Files
Select the design files you want to include in the project. Click Add All to add all design files in the project directory to the project.
File name:
...
Add
Add All
Remove
Up
Down
Properties
File Name
Type
Library
Design Entry/Synthesis Tool
HDL Version

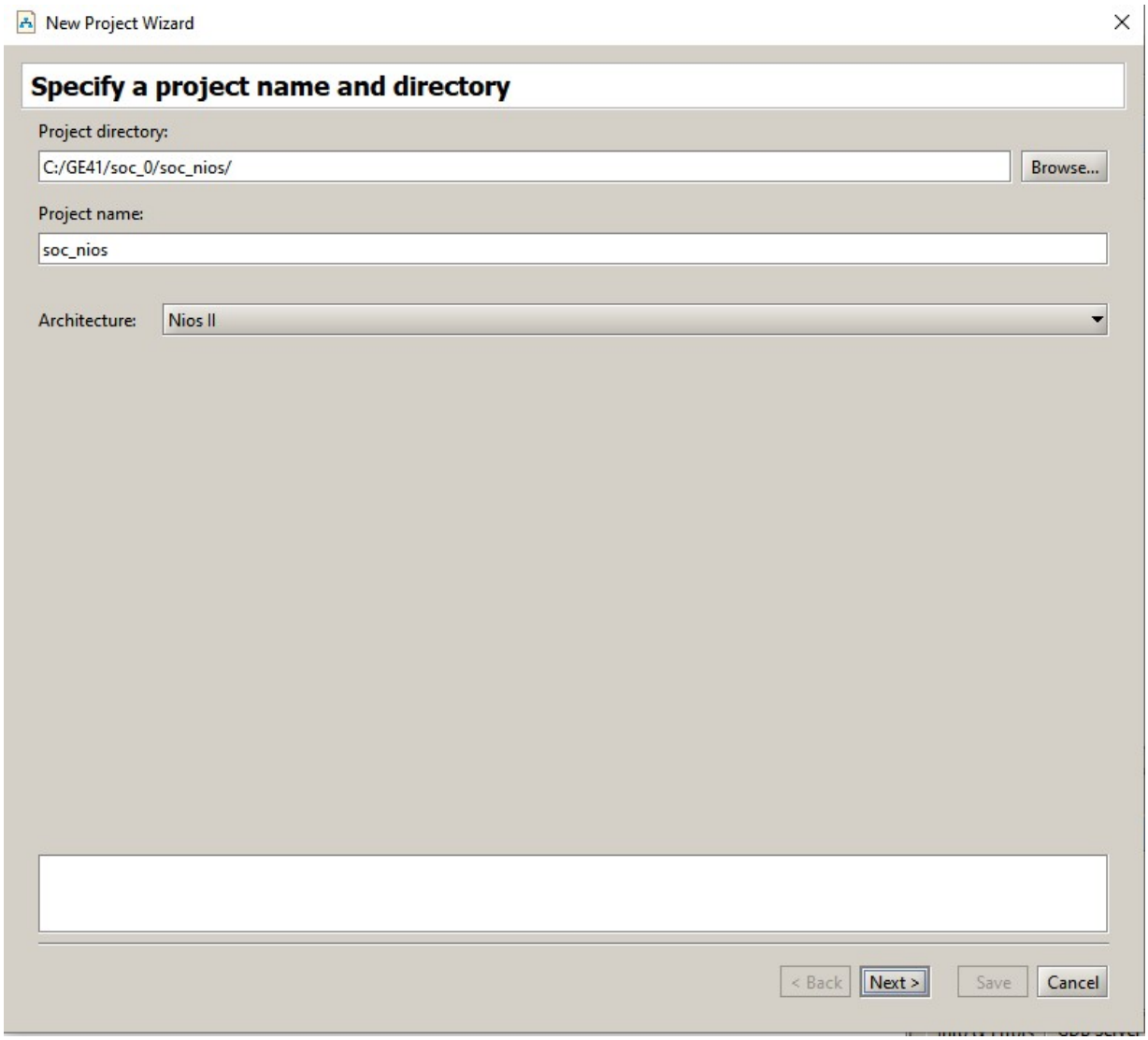


37) Dans le fichier soc_0.vhd, faire les liaisons avec les entrées et les sorties de la carte comme ceci

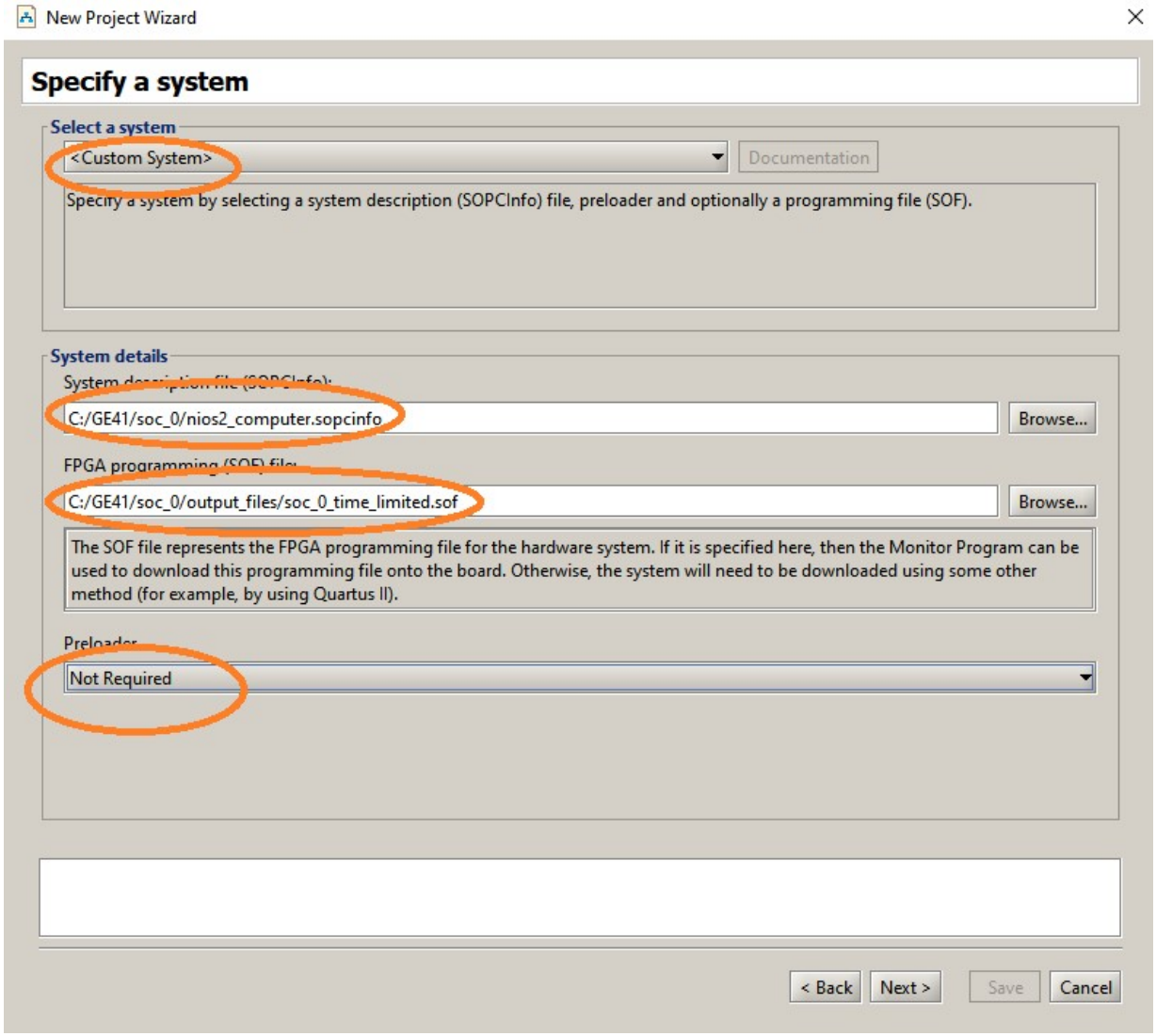
```
u0 : component nios2_computer
    port map (
        clk_clk                           => clock_50,
        key1_external_connection_export   => key(1),     --     key1_external_
        leds_external_connection_export   => ledr(7 downto 0) ,     --     le
        reset_reset_n                     => key(0),                --
        switches_external_connection_export => SW(7 downto 0)  -- switches_exte
    );
```

38) Compiler le projet Quartus

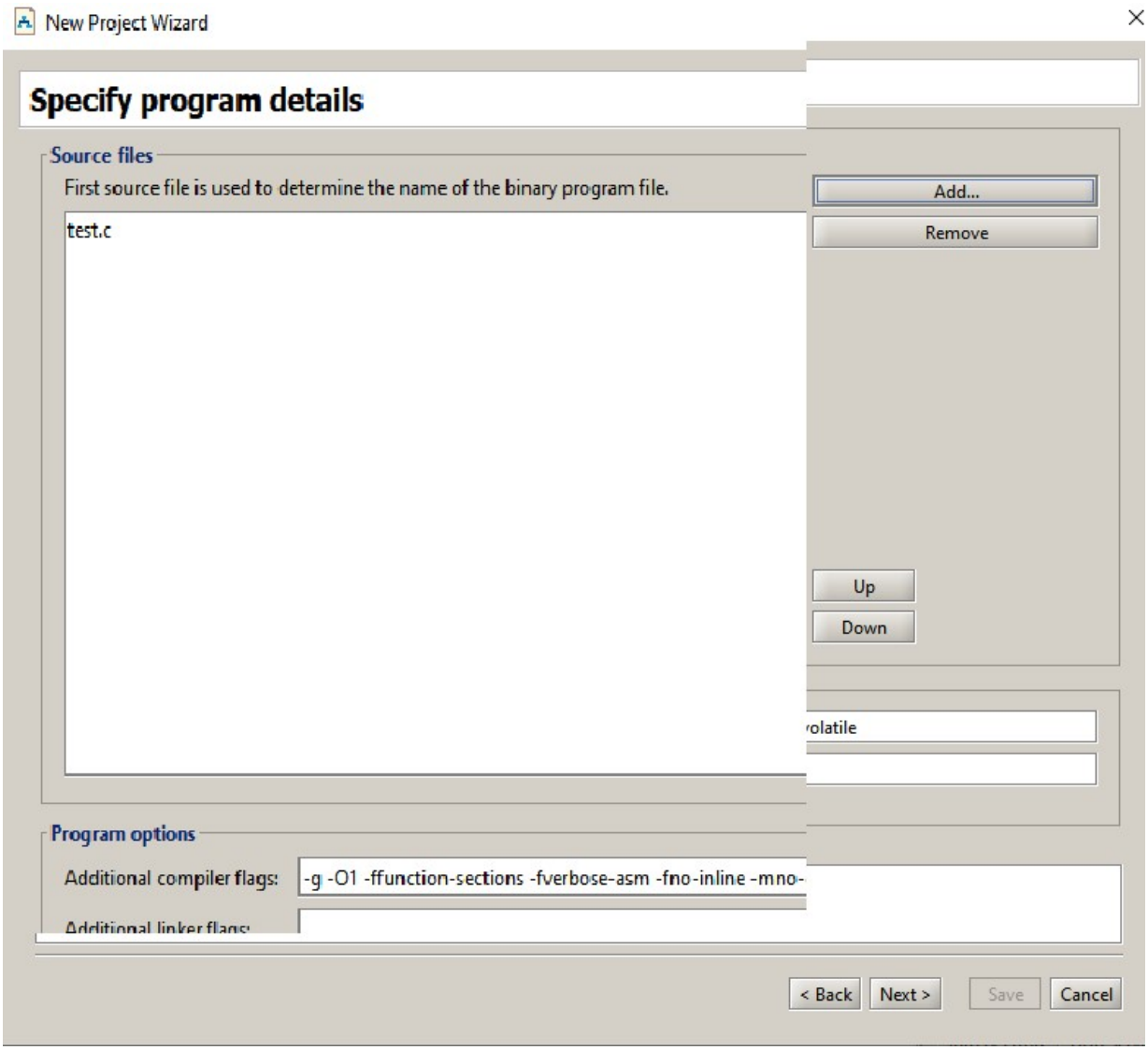
39) Ouvrir Intel FPGA Monitor Program. Créer un nouveau projet comme indiqué sur la figure suivante



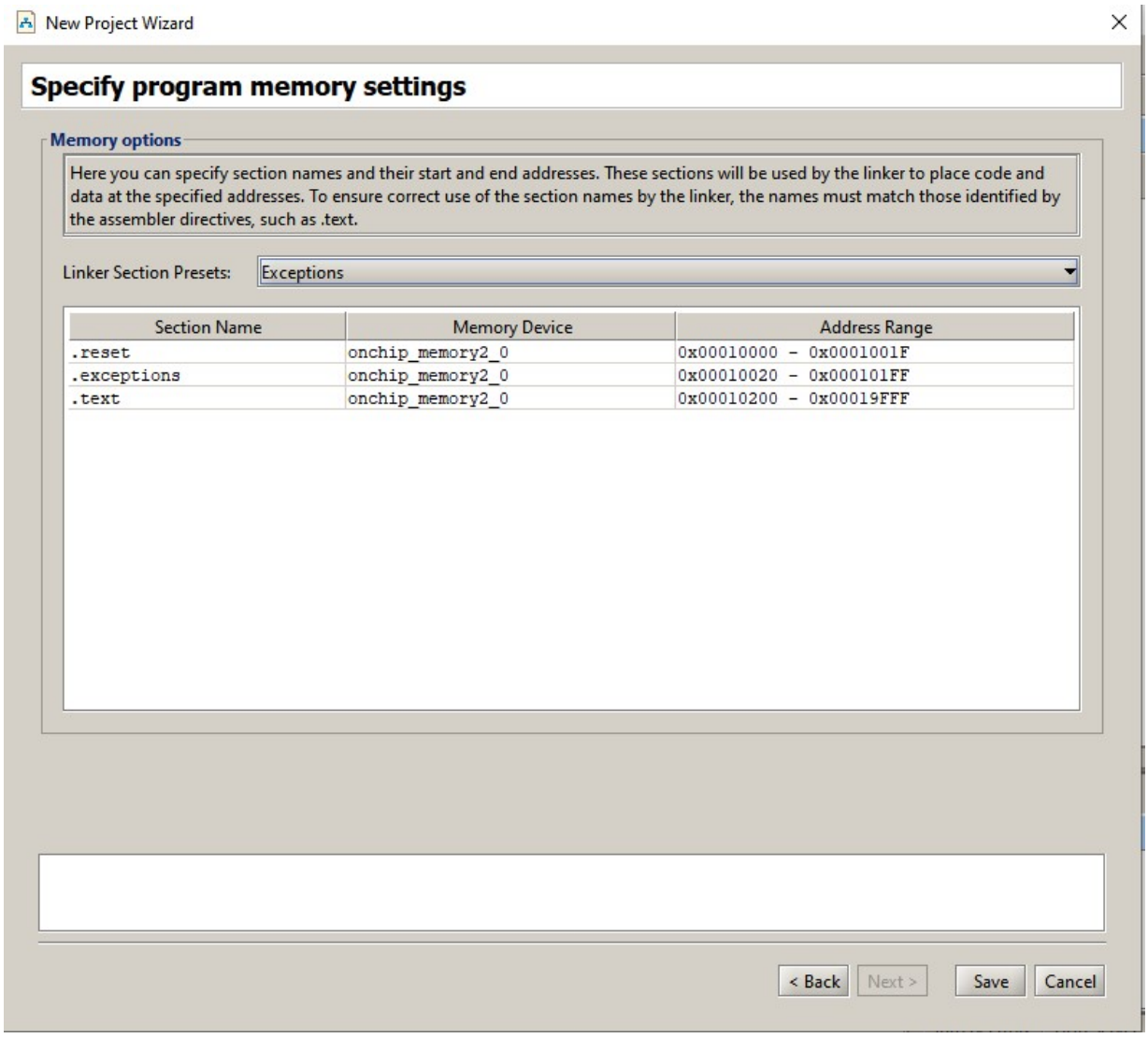
40) Choisir les options comme indiqué sur la figure suivante



41) Sélectionner le programme en C
42) Ajouter le fichier test.c



43) Les spécifications mémoires sont comme ceci



44) Cliquer sur Save
45) Continuer dans TP2. Exécuter le programme test.c sur la carte.
Remarque : le message suivant va appraître sur l'écran. Le laisser et ne pas cliquer.