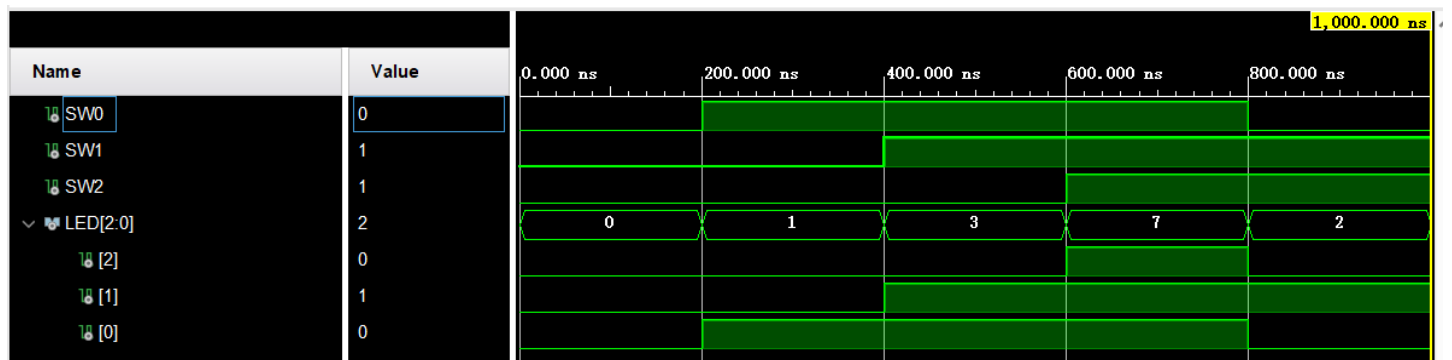


Rapport du TP FPGA

Weiyi GONG, Gr sesi
Zhuyu WEN, Gr syscom

- **Séance 1 : 1 ERE PARTIE – SYNTHÈSE VHDL SUR FPGA I) Prise en main de la carte et des outils 1) Présentation de la carte Digilent Nexys 2) Création d'un projet avec le logiciel Vivado** Pour les deux parties précédentes, nous avons suivi le sujet pas à pas.

4) Testbench et simulation



Nous pouvons voir que l'image résultante correspond au sujet demandé:

LED(0)=SW0, LED(1)=SW1, LED(2)=SW0SW1SW2

5) Implémentation sur la carte FPGA Et pour la partie 4) et 5), nous avons aussi suivi le sujet. Pour connecter la carte Basys 3, il faut changer le general-board-Basys3 dans le setting.

II) Cas d'études – Synthèse VHDL 1) Compteurs Imbriqués Observation:

Ajouter au projet les fichiers **Test_CPT.vhd** et **Test_CPT_Nexys4DDR.xdc**.

- Quelle est la fonctionnalité décrite dans le VHDL?
- A quel résultat peut-on s'attendre sur la carte ?

Il décrit que chaque fois que nous appuyons sur le bouton_left, le compteur de 4bits augmentera progressivement d'un. Nous pouvons voir sur la carte que les quatre voyants de droite s'allumeront sous la forme 0001, 0010, 0011, 0100... à chaque appui.

Implémenter le design.

- Astuce : vous pouvez cliquer directement sur **Generate Bitstream** pour faire d'un trait toutes les étapes de l'implémentation. (Le flot s'arrêtera en cas d'erreur)

Tester sur la carte. Que se passe-t-il ?

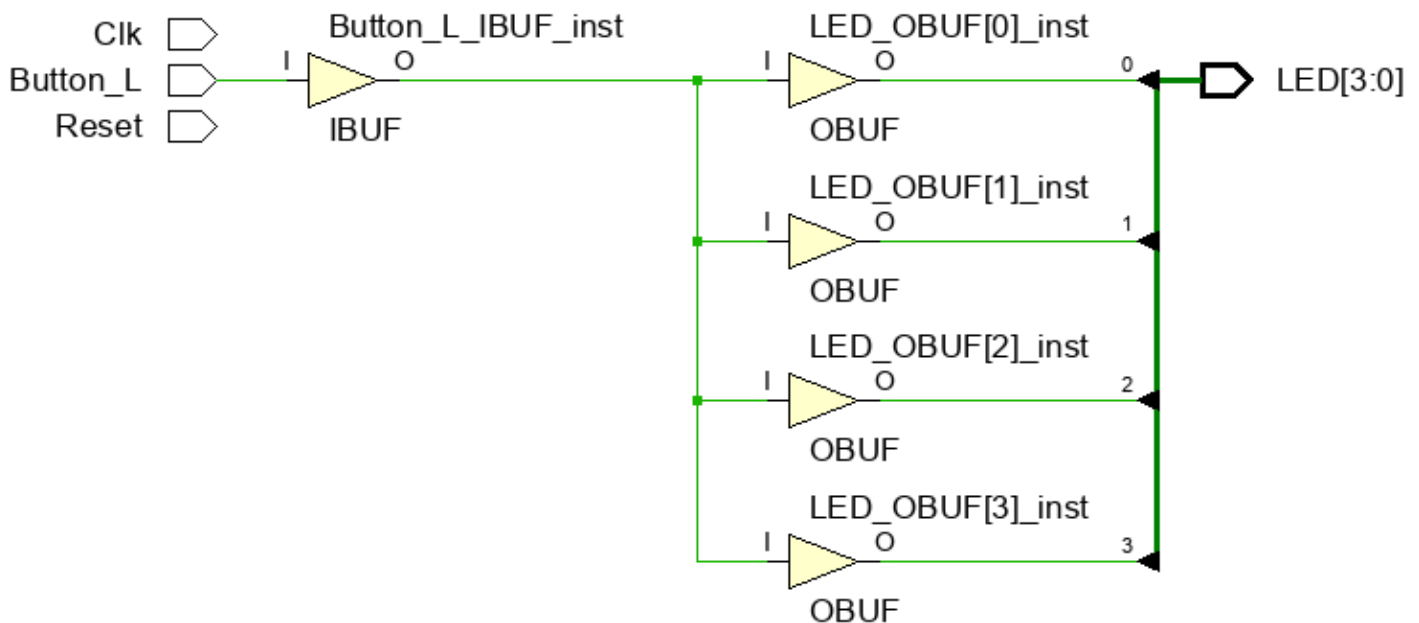
Après la connexion, rien ne se passe sur la carte, si nous appuyons sur le bouton gauche, les quatre voyants LED sont allumés

Analyse

Dans le **Flow Navigator**, dérouler la ligne **Open Synthesized Design** et cliquer sur **Schematic** pour générer un schéma RTL de l'architecture synthétisée.

- - Est-ce que cela correspond au code VHDL de départ ?

On obtient le Schematic comme au dessous:



Il ne correspond au code VHDL de départ car le Clk et le Reset ne sont pas connectés dans le circuit.

En revenant dans le **Project Manager** (fenêtre **Flow Navigator**), aller dans la fenêtre **Console** et cliquer sur l'onglet **Reports**. Afficher le **Synthesis Report**. faire une recherche sur l'expression « **[synth** » et analyser les messages **WARNING**.

- - Que comprenez-vous ? Y a-t-il quelque chose qui vous semble anormal ?
 - En faisant le lien entre les Warning et le Schematic vu précédemment, quel signal dans la description VHDL peut avoir posé problème lors de la synthèse ?

E:/M1 syscom semestre 2/FPGA/TP1/project_218TPchongxinzuo/project_218TPchongxinzuo.runs/synth_1/Test_CPT.vds

Q

📁

↶

↷

✂

📄

📁

✕

//

📊

💡

synth

Next

Previous

Highlight

☐ Match Case

☐ Whole Words

6 Match(es)

54

55 Finished Constraint Validation : Time (s): cpu = 00:00:05 ; elapsed = 00:00:09 . Memory (MB): peak = 1041.973 ; gain = 24.742

56 -----

57 -----

58 Start Loading Part and Timing Information

59 -----

60 Loading part: xc7a35tcbg236-1

61 -----

62 Finished Loading Part and Timing Information : Time (s): cpu = 00:00:05 ; elapsed = 00:00:09 . Memory (MB): peak = 1041.973 ; gain = 24.742

63 -----

64 -----

65 Start Applying 'set_property' XDC Constraints

66 -----

67 -----

68 Finished applying 'set_property' XDC Constraints : Time (s): cpu = 00:00:05 ; elapsed = 00:00:09 . Memory (MB): peak = 1041.973 ; gain = 24.742

69 -----

70 WARNING: [Synth 8-327] inferring latch for variable 'start_reg' [E:/M1 syscom semestre 2/FPGA/TP1/project_218TPchongxinzuo/project_218TPchongxin

71 -----

72 Finished RTL Optimization Phase 2 : Time (s): cpu = 00:00:05 ; elapsed = 00:00:09 . Memory (MB): peak = 1041.973 ; gain = 24.742

Comme le problème on trouve dans le schématic,c'est clk et reset qui pose problème.Donc on vérifie le process(Clk,Reset) et on trouve que la taille du range de cpt est incorrecte.Donc on a changé d'abord le range du signal Cpt de 20000000 à 200000000 pour bien fonctionner le code,car le Cpt est égal à 70000000 ce qui devrait être inférieur au range du signal Cpt dans le code suivant.

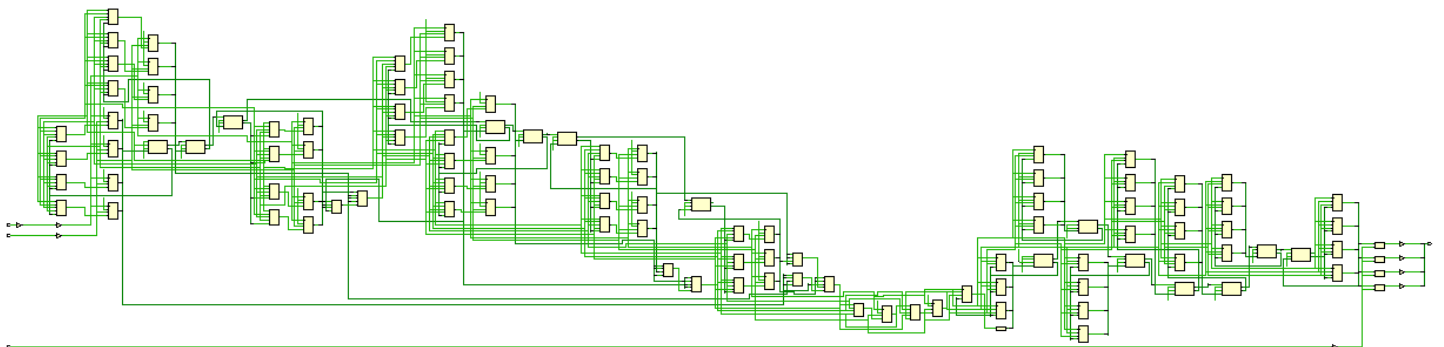
correction et vérification

Une fois l'erreur corrigée, refaire une synthèse et ouvrir le rapport.

- Vérifier en fin de rapport qu'il n'y a plus de warnings.
- Recharger le schéma RTL de l'architecture synthétisée pour constater la différence

- Poursuivre l'implémentation et porter le design sur la carte. Valider le bon fonctionnement de l'architecture.

Après la correction du code,nous pouvons voir que les 4 lumières LED à droite comptent automatiquement, en ajoutant une à chaque fois. Si nous appuyons et maintenons le bouton gauche enfoncé, les quatre voyants LED seront allumés.Ceci correspond au résultat qu'on attend précédent. On obtient un nouveau schematic:



2)Compteur d'Impulsions Observation

Retirer toutes les sources de votre projet et ajouter les fichiers ***Impulse_Count.vhd*** et ***Impulse_Count_Nexys4DDR.xdc***.

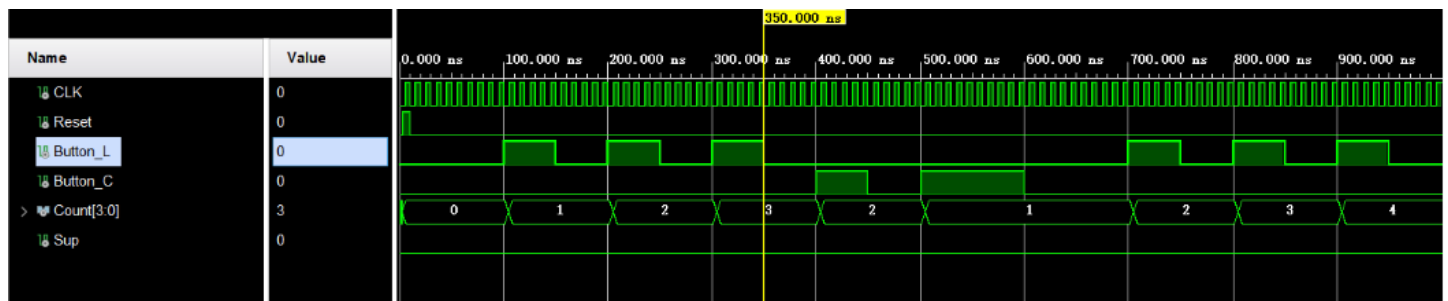
- ○ Quelle est la fonctionnalité décrite par le code VHDL ?

Le code VHDL décrit que si nous appuyons sur le bouton gauche, le compteur incrémente de un, et si nous appuyons sur le bouton du milieu, le compteur diminue de un.

Ajouter au projet le testbench ***TB_Impulse_Count.vhd***.

- ○ Analyser le Testbench pour savoir ce que l'on va effectuer dans la simulation.
- ○ Simuler et vérifier le bon comportement de l'architecture.

Ce testbench est pour tester le bouton_L toutes les 50ns entre 100ns et 350ns, et entre 700ns et 1550ns. Et il s'agit de tester le bouton_C toutes les 50ns entre 400ns et 500ns, et de tester le bouton_C une fois entre 500ns et 600ns. Avant faire la simulation, il faut ajouter un clk dans le testbench. Sinon quand nous faisons la simulation, il y a des erreurs.



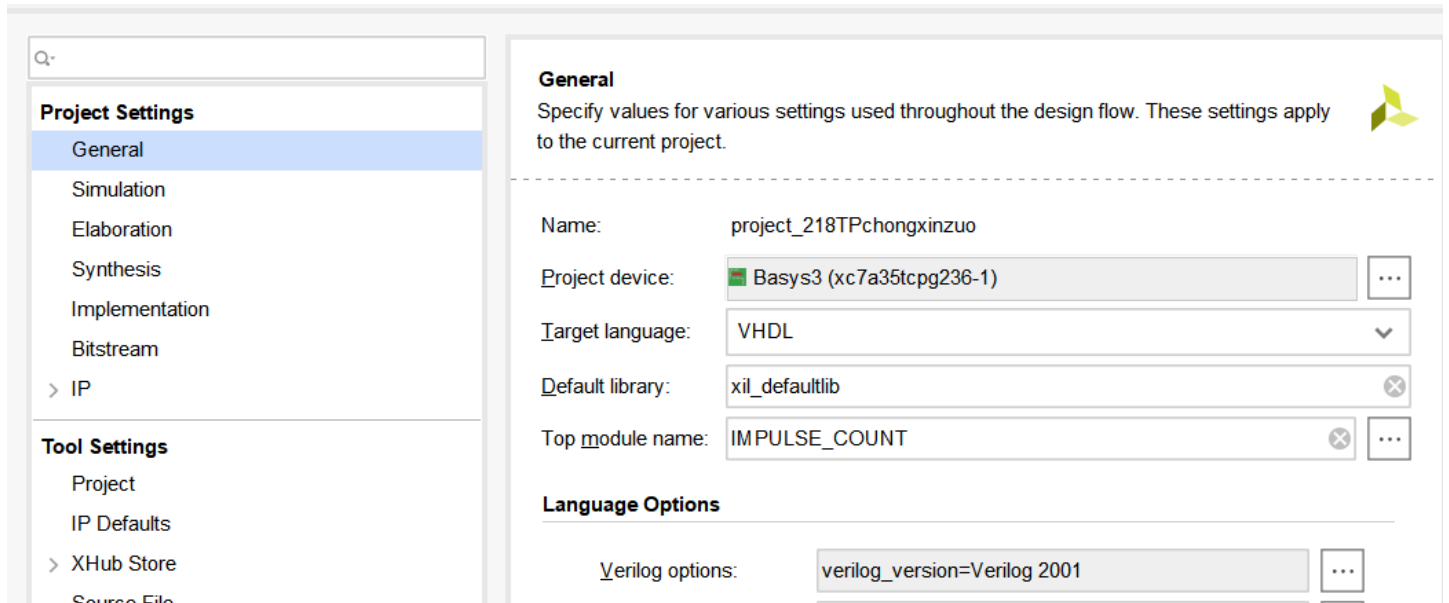
Nous voyons que le schéma de simulation obtenu est cohérent avec ce que nous avons dit précédemment.

Implémenter et vérifier le bon comportement sur la carte. Que constatez-vous ?

- ○ NB : il y a un phénomène de rebond sur les boutons de la carte qui peut entraîner plusieurs fronts du signal pour un seul et même appui sur le bouton.

Nous constatons que si nous appuyons sur le bouton gauche, le compteur incrémente de un, et si nous appuyons sur le bouton du milieu, le compteur diminue de un. Comme nous avons répondu pour la fonctionnalité décrite par le code *impulse_count.vhd*.

Analyse



Project Settings

- General
- Simulation
- Elaboration
- Synthesis
- Implementation
- Bitstream
- > IP

Tool Settings

- Project
- IP Defaults
- > XHub Store
- Source File

General

Specify values for various settings used throughout the design flow. These settings apply to the current project.

Name: project_218TPchongxinzuo

Project device: Basys3 (xc7a35tcbg236-1)

Target language: VHDL

Default library: xil_defaultlib

Top module name: IMPULSE_COUNT

Language Options

Verilog options: verilog_version=Verilog 2001

Nous vérifions bien le target language est de VHDL.

Relancer ensuite une simulation, mais en choisissant cette fois la **Post Synthesis Functional Simulation**.

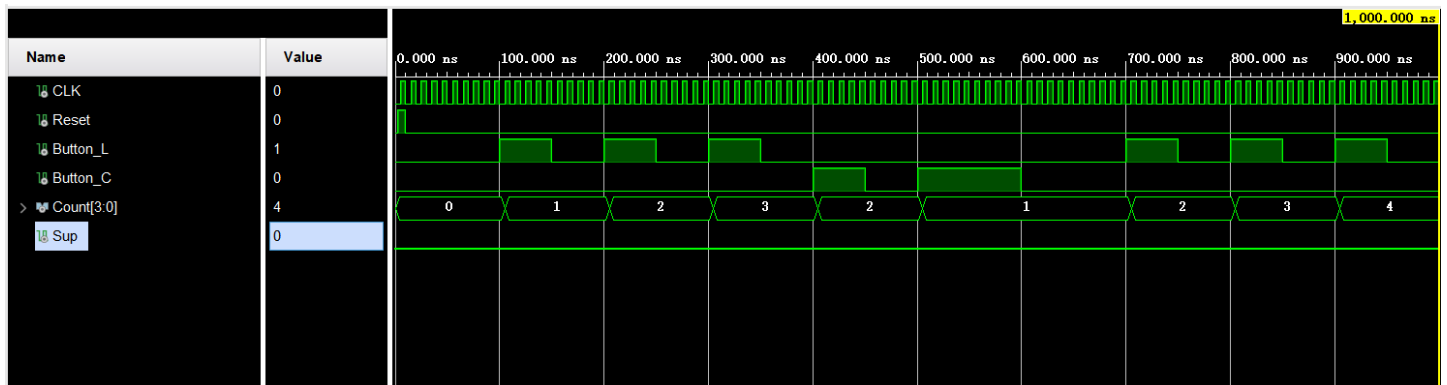
- L'outil va simuler un fichier VHDL correspondant à l'architecture synthétisée par **Vivado**. Ce fichier, généré lors de la synthèse est consultable dans le simulateur en cliquant sur le module **UUT** dans la fenêtre **Scopes**.
- Ouvrir et regarder la structure du code. Comment est-il composé ?
- Comparer le chronogramme de la simulation post-synthèse avec la simulation pré-synthèse vue précédemment. Le comportement est-il différent ?
- Que constatez-vous entre le comportement de la simulation post synthèse et celui du design testé sur la carte ?.

Le module UUT pré-synthèse:

Le module UUT post-synthèse:

Nous voyons que après le synthèse, il y a plusieurs autres éléments dans le module UUT, tels que : Reset_IBUF, Sup_OBUF.....

Mais pour le chronogramme, le comportement ne change pas, c'est la même que le chronogramme précédent:

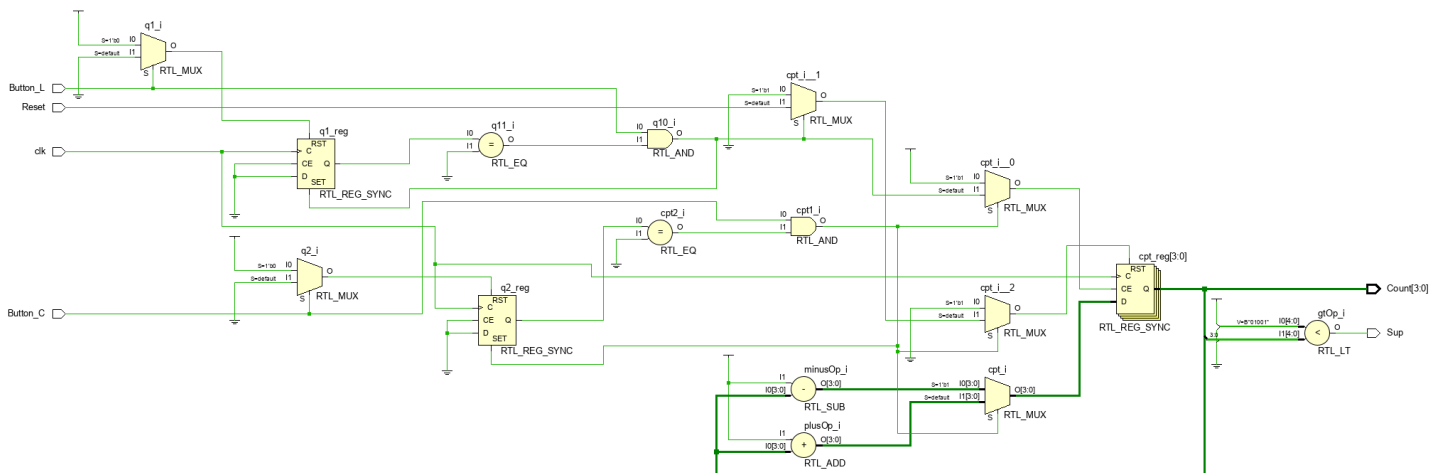


Correction et vérification

Comment modifier le code VHDL pour faire en sorte que le compteur d'impulsions fonctionne correctement ?

- Décrire cette nouvelle architecture en VHDL
- Modifier le testbench en conséquence puis lancer une simulation (**Behavioral Simulation**) pour valider fonctionnellement cette nouvelle description.

Pour fonctionner le compteur d'impulsion correctement, il faut ajouter deux variables q1, q2 pour enregistrer l'état précédent du bouton. Ainsi, nous pouvons juger que le compteur peut compter avec précision chaque fois que le bouton a été poussé. Après finir la modification du code, nous lançons la simulation schématique RTL et nous obtenons :



nous vérifions bien que le schéma correspond au code et il marche aussi sur la carte.

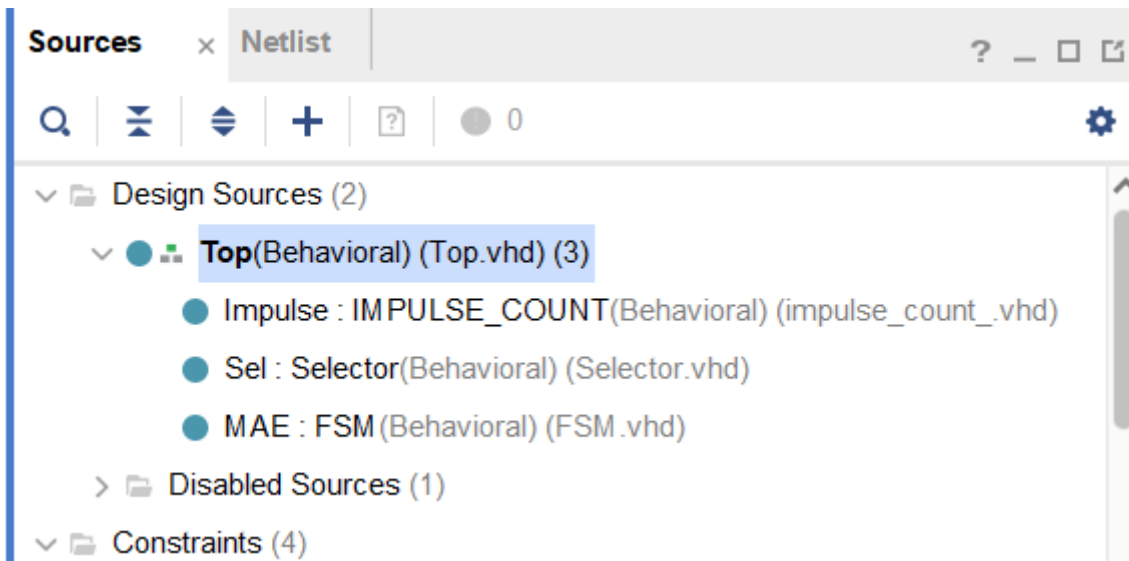
3) Décodeur & Machine à Etats Observation

Dans le même projet, ajouter de nouvelles sources (**Design Sources**) correspondant aux fichiers suivants :

- **Selector.vhd, FSM.vhd et Top.vhd**
- Analyser les codes pour comprendre quelle est la fonctionnalité et la hiérarchie du système.

Ces codes doivent contrôler les quatre lumières LED pour qu'elles clignent en même temps, et la taille du compteur peut contrôler la durée de l'intervalle de clignotement, plus le compteur est grand, plus l'intervalle de clignotement est long.

- Implémenter le design et tester sur la carte.
 - o Appuyer 3 fois sur le bouton Left (sélection du mode 1) puis 1 fois sur le bouton Right (validation), ce qui doit entraîner un clignotement des **LEDs**. Que constatez-vous ?



Nous mettons le fichier Top.vhd comme top level et nous connectons la carte. Nous appuyons 3 fois sur le bouton Left puis 1 fois sur le bouton Right et nous constatons que les 4 LEDs sont allumées mais ne clignotent pas.

- o Faites un bref reset du système (Tirer le Switch 0 vers le bas, puis le remettre vers le haut). Cette action devrait normalement éteindre les LED. Que se passe-t-il ?

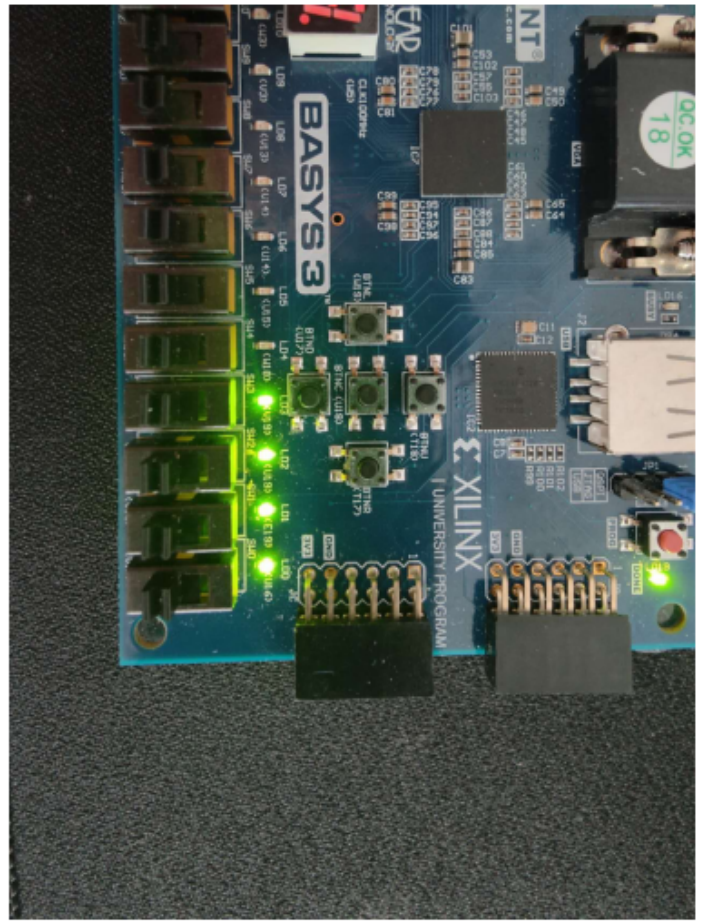
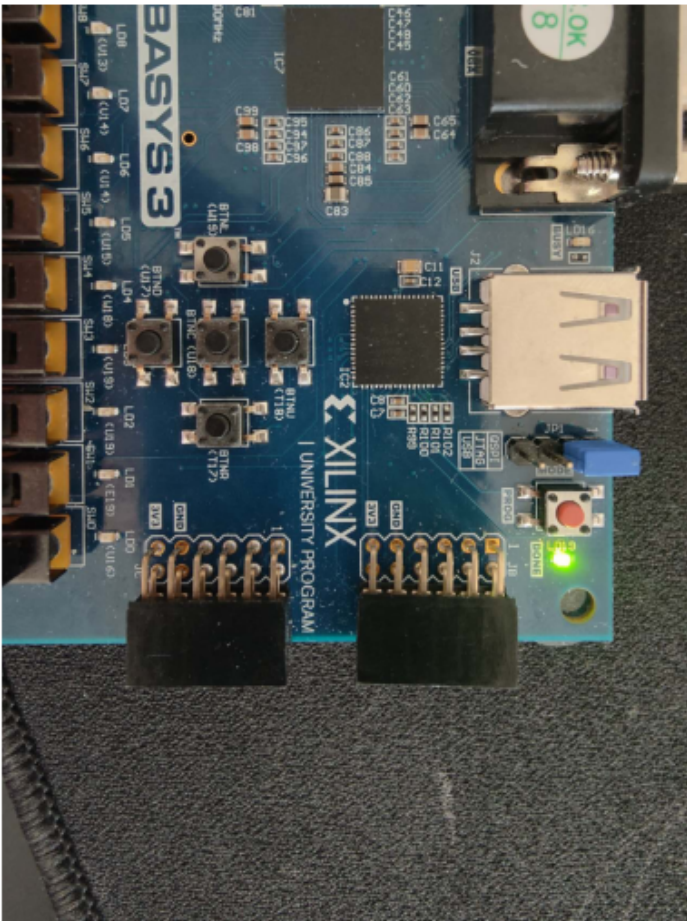
Nous voyons que après avoir tiré le Switch 0 vers le haut, les 4 LEDs sont éteintes. Après l'avoir remis vers le bas, les 4 LEDs sont allumées mais l'intensité lumineuse est plus faible qu'avant.

Analyse et Correction

- Ouvrir le rapport de synthèse et repérer les warnings..
 - o Quelle(s) erreur(s) dans le code a pu provoquer cela ?

Normalement, le problème est dans MAE. Lorsque nous avons vérifié le MAE, nous avons constaté qu'il manquait une étape dans chaque état. Cette étape est le cas où l'état future revient à l'état actuel dans le cas de else.

Après modification de MAE, il n'y a pas de warning en synthèse et les LEDs sur la carte peuvent clignoter normalement.



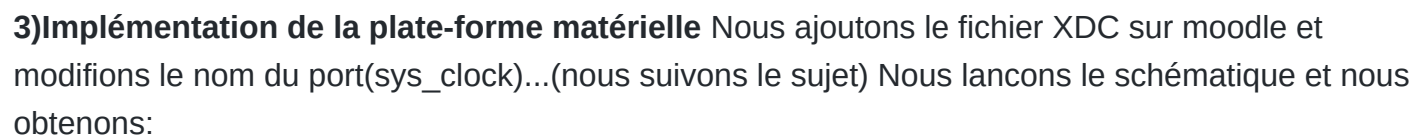
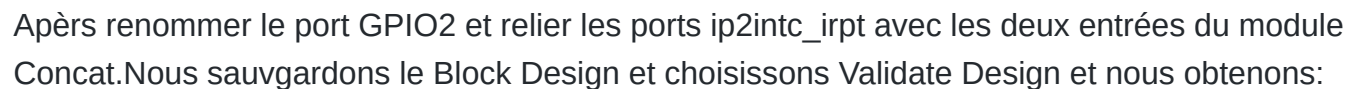
• Séance 2 : 2 EME PARTIE – CODESIGN MATERIEL/LOGICIEL

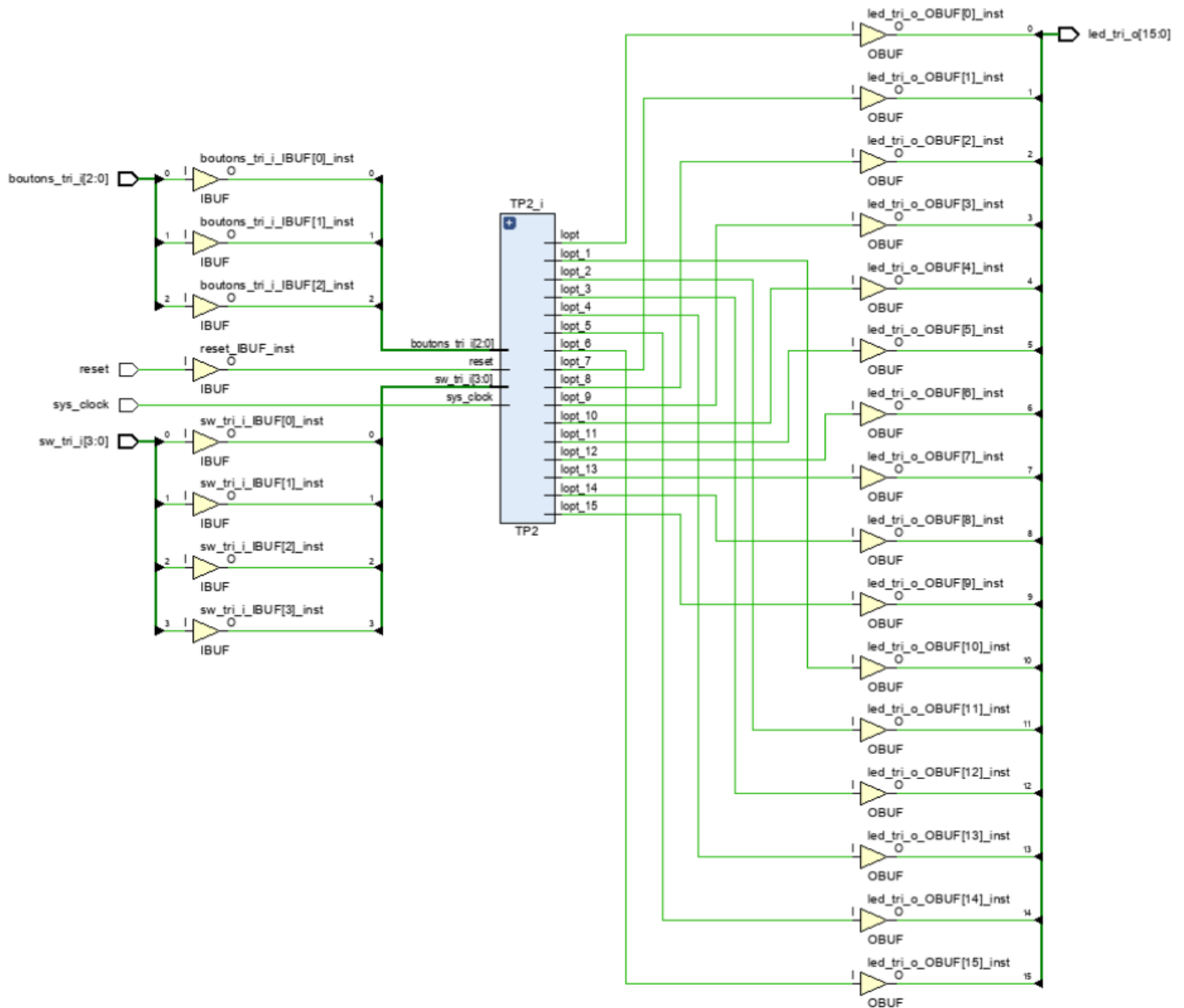
2)Spécification de la plate-forme matérielle Nous créerons un projet et ajoutons un IP (MicroBlaze) dans la fenêtre diagram. Nous cliquons sur Run Block Automation et il ajoute cinq modules distinctes automatiquement: Clocking Wizard et Processor System Reset, Microblaze Debug Module, Microblaze Local Memory, AXI Interconnect, AXI Interrupt Controller.

Après nous ajoutons un nouveau IP--"led-switch". Dans ce nouveau IP, le port 1 du bloc GPIO sera associé aux 4 interrupteurs à droite de la carte et le port 2 du bloc GPIO sera associé aux 16 LEDs.

Nous ajoutons une autre IP--"boutons", il contient 1 port d'entrée de 3 bits et pouvant également interrompre le processeur.

Après avoir cliqué sur Run Connection Automation, nous configurons les 5 périphériques: Périphérique boutons, Périphérique clk_wiz_1, Périphérique led_switch et Périphérique rst_clk_wiz_1_100M. Nous obtenons:

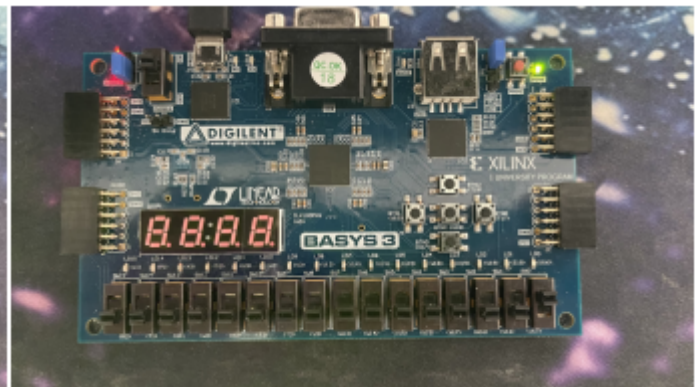
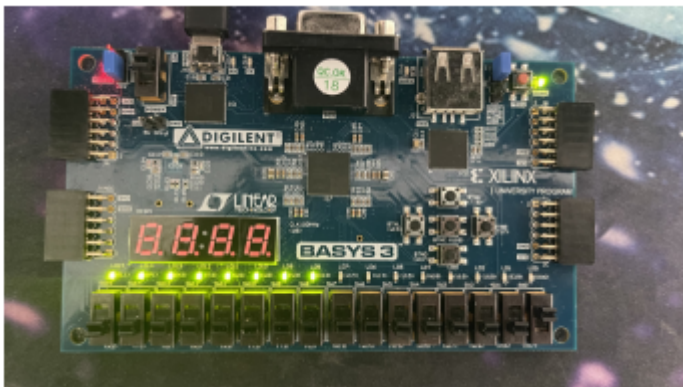




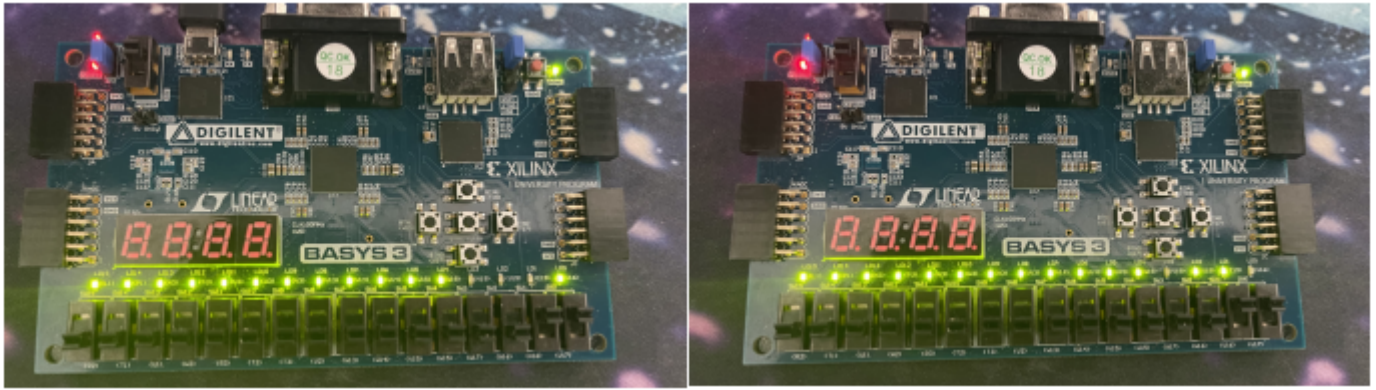
Nous pouvons voir que dans ce schéma tout s'est bien passé.

4)Création des projets pour le développement logiciel sous Vitis Maintenant nous passons à Vitis qui nous permet de coder en langage C pour réaliser la fonction sur la carte Basy3.

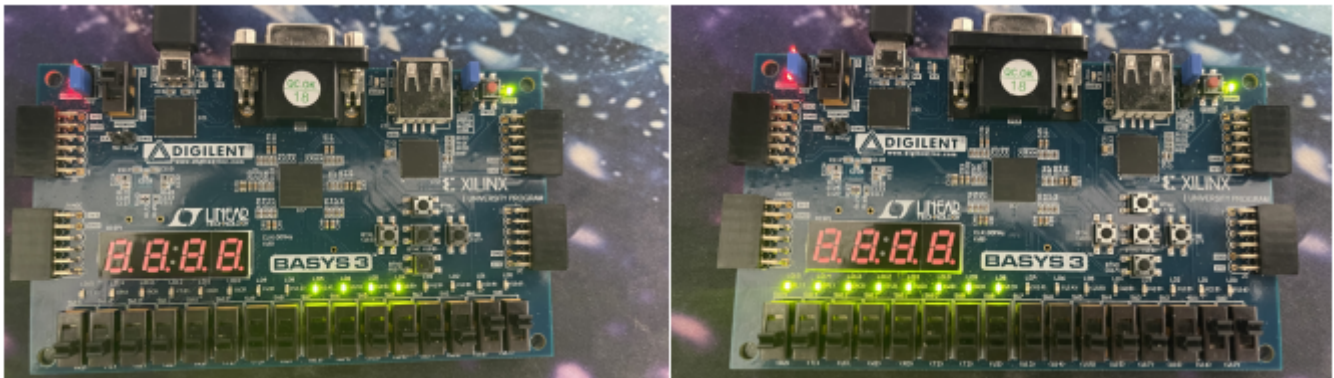
5)Développement de l'application logicielle Les 8 LED de gauche puissent clignoter si l'interrupteur 0 est relevé:



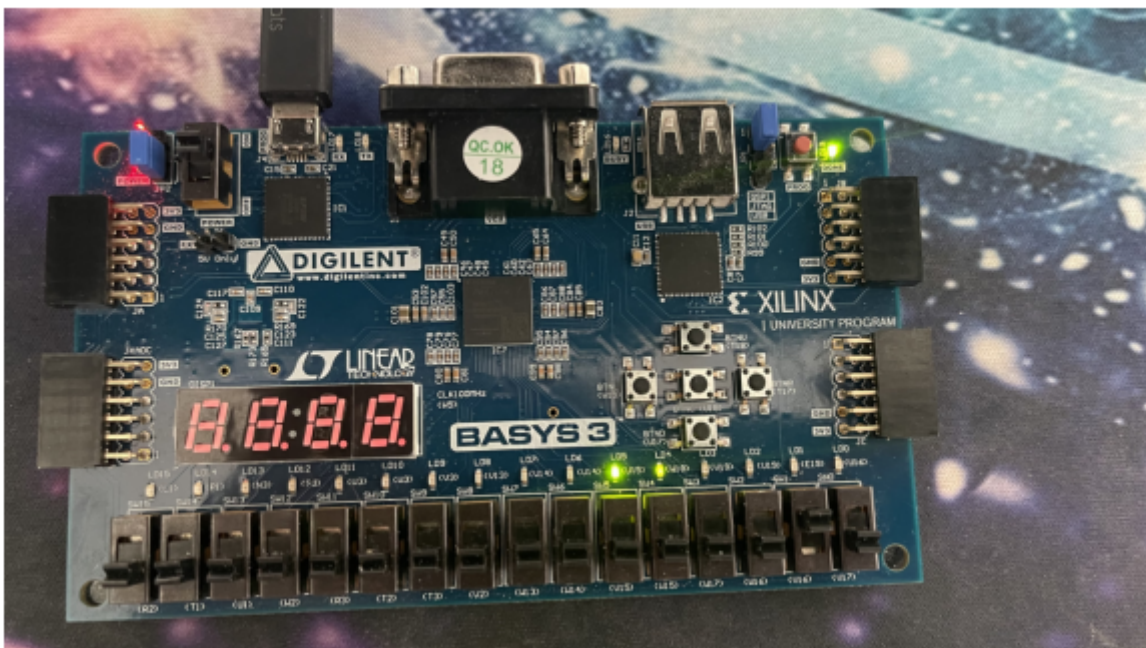
Si l'interrupteur 1 est relevé, les 4 LED de droite affichent la valeur d'un compteur modulo 16 incrémenté par le bouton Center:



Si on appuie sur le bouton Left, les LED 7 à 4 s'allument et si on appuie sur le bouton Right, les 4 LED 7 à 4 s'éteignent:

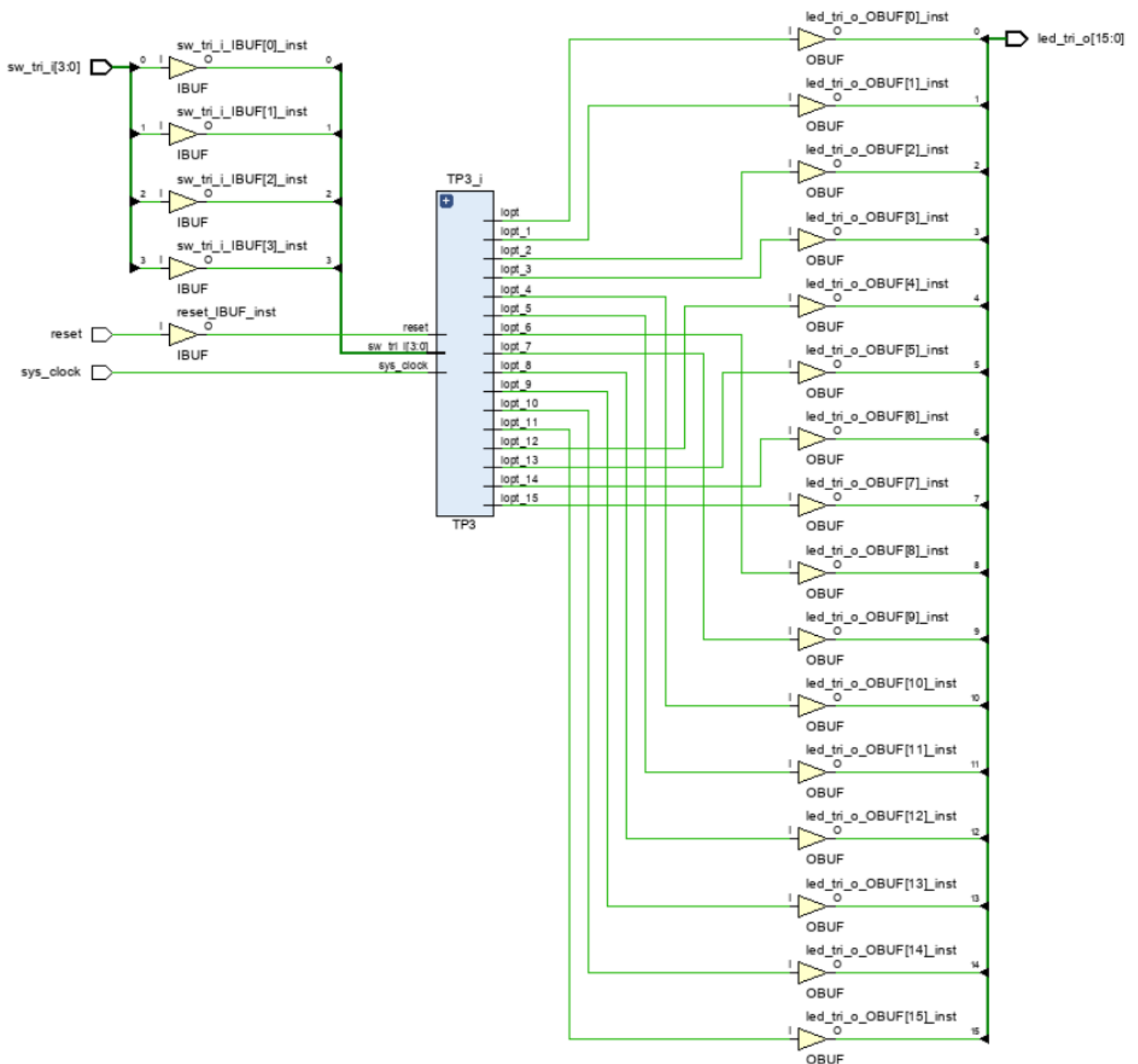
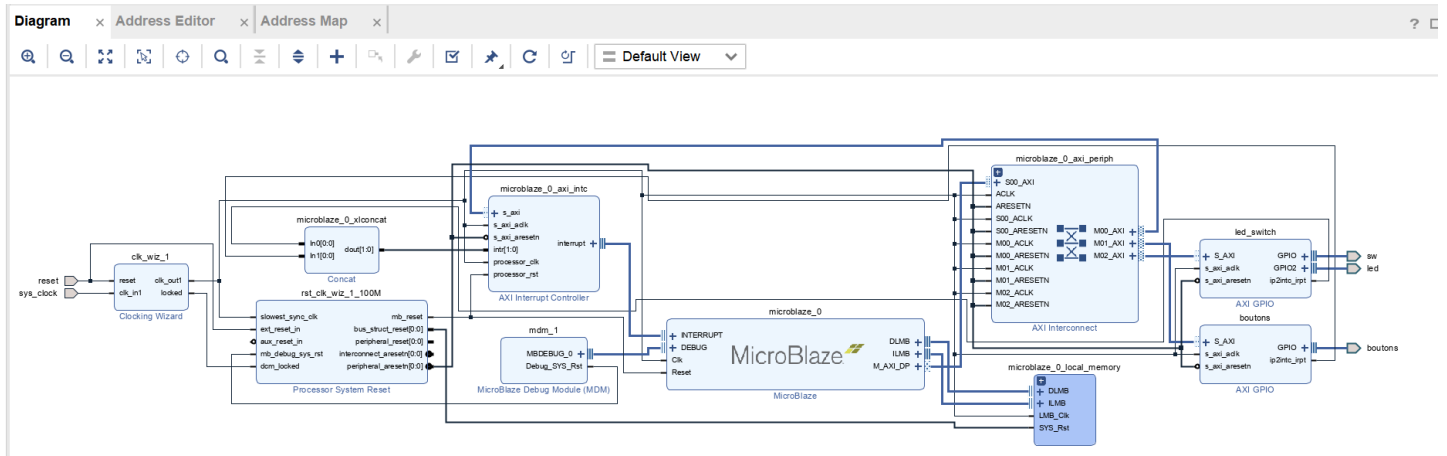


Faire un chenillard qui parcourt les 16 LEDs de la carte avec une vitesse de défilement paramétrable grâce aux 4 interrupteurs:



Il est difficile de capturer un chenillard devant la caméra, mais nous voyons quand même un peu que les LEDs clignotent de droite à gauche.

- **Séance 3 : 3 EME PARTIE – – CONCEPTION D'IP POUR LE MICROBLAZE** Nous continuons le TP3 avec ce que nous avons fait en TP2. Nous suivons les instructions dans le sujet et décrivons un module VHDL dans le fichier my_led.vhd qui possède 1 entrée sur 4 bits et 1 sortie sur 16 bits...etc Après nous ouvrons le Block design et nous obtenons les figures au dessous:



Nous voyons que il y a 4 entrées et 16 sorties dans ce schéma qui répondent aux exigences dans le sujet.

Pour la partie du code,nous voulons que les 4 interrupteurs de droite contrôlent respectivement les leds sur les bits 15-12, 11-8, 7-4 et 3-0,comme indiqué ci-dessous:

