CAN2301T

Philippe TANGUY

Sep 20, 2022

CONTENTS:

CHAPTER

ONE

TUTORIELS

1.1 Vivado Zedboard PS-PL

L'objectif de ce guide est de vous présenter l'outil Vivado pour développer sur une carte Zedboard.

1.1.1 Exemple: PS et PL avec Ringcounter

Introduction

Les objectifs de cet exemple sont:

- Exemple guidé pour utiliser Vivado
- Exemple simple Zynq PS avec PL et AXI GPIO

Les pré-requis sont:

- Installation Vivado v2019.1
- Carte Zedboard

Cahier des charges

Voici les exigences:

- La partie PS pilote un ringcounter avec un signal start, stop et reset
- Le ringcounter est connecté à 8 LEDs sur la carte Zedboard
- La partie PS fournit l'horloge au composant ringcounter
- Les LEDs change toutes les secondes environ

Spécifications

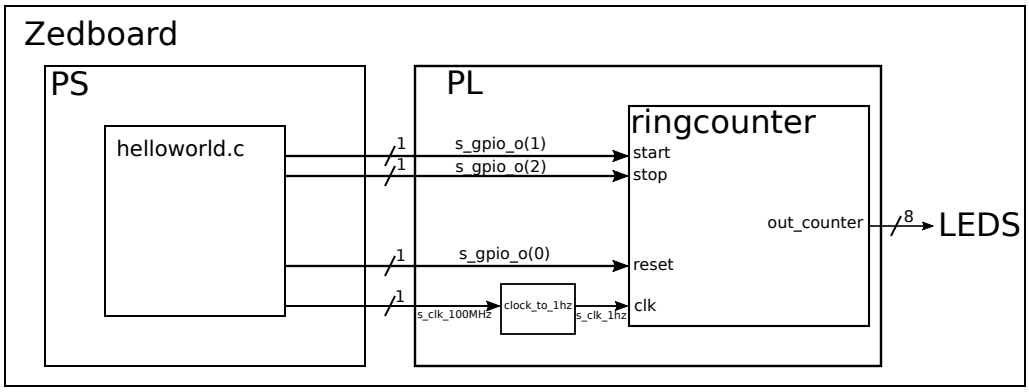


Fig. 1: Synoptique de la conception du tutoriel

Partie matérielle

Table 1: Ringcounter

Nom	Direction	Type	Description
Start	in	std_logic	signal permettant de démarrer le compteur, actif à '1'
stop	in	std_logic	signal permettant d'arrêter le compteur, actif à '1'
reset	in	std_logic	signal permettant de démarrer le compteur, actif à '1'
clk	in	std_logic	signal d'horloge
out_counter	out	std_logic_vector, 8 bits	signal de sortie

Table 2: Clock_to_1hz

Nom	Direction	Type	Description
clk_in	in	std_logic	signal à 100 Mhz
clk_1hz	out	std_logic	signal de sortie à 1hZ

Partie logicielle

Le logiciel est dit *baremetal* et suit exécute la séquence suivante:

1. reset
2. start
3. attente de 5s
4. stop
5. attente de 5s
6. start
7. boucle infinie

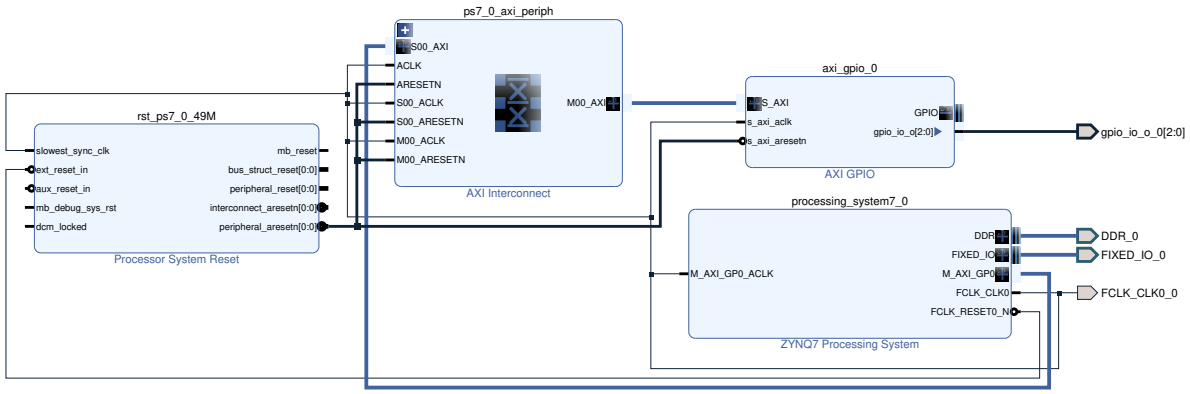
Conception matérielle

Créez un projet dans Vivado en configurant son projet par rapport à la cible (Zedboard) Une fois le projet créé nous allons créer un *Block Design* en suivant les étapes suivantes dans Vivado

1. Create block design
2. Add IP button puis ajoutez zynq
3. Run block automation
4. Clique droit sur FCLK_CLK0_0 et faire make external

Nous allons ensuite ajouter une IP appelé *AXI GPIO*. Pour cela suivez les étapes suivantes:

1. Add IP button puis ajoutez AXI GPIO
2. **Configurez le block GPIO pour avoir 3 bits qui correspondront aux** signaux start, stop et reset
3. run connection automation} et cochez uniquement AXI
4. Sur la sortie du bloc GPIO faire un clique droit sur gpio_io_o et faire *make external*
5. validate design



Nous allons à présent créer le *Wrapper* en suivant les étapes suivantes:

1. Clique droit sur le block design (dans la fenêtre *sources*) puis cliquez sur *Generate Output Products*
2. Clique droit sur le block design (dans la fenêtre *sources*) puis cliquez sur *Create HDL Wrapper*, attention à bien cochez l'option *Copy generated wrapper to allow user edits* dans la fenêtre qui s'affiche.

Dans notre conception le choix a été fait ici de modifier le wrapper. Nous pourrions faire autrement comme par exemple créer un périphérique afin d'ajouter notre code HDL. Pour la modification du wrapper suivez les étapes suivantes:

1. Modifiez le wrapper en ajoutant les signaux: s_gpio_o (vecteur 3bits), s_clk_100Mhz, s_clk_1hz et le composant ringcounter
2. Faire la connection des signaux avec les composants
3. Ajoutez une sortie LEDS sur 8 bits au wrapper

Enfin il nous faut gérer notre fichier de contrainte

1. Création fichier de contrainte
2. Ajout des LEDs

Une fois la conception terminé vous pouvez faire la synthèse, l'implantation et générer le bitstream.

Conception logicielle

Nous allons à présent préparer notre environnement pour travailler avec le SDK. Pour ce faire il va falloir exporter notre conception en faisant dans Vivado: *File -> export -> export hardware*

Ensuite nous pouvons lancer le SDK depuis Vivado en faisant *File -> launch SDK* Une fois dans le SDK nous allons créer un nouveau projet *File -> new -> Application project*

Enfin, nous allons utiliser le programme fournin *helloworld.c* en y copiant le code fourni sur la plateforme pédagogique.

Test sur carte

Pour valider le tutoriel il faudra charger le programme et l'exécuter:

1. Icône Program FPGA
2. Icône Run
3. Test visuel sur carte

Vous devez observer la séquence définie dans le cahier des charge

References

1. http://easytp.cnam.fr/alexandre/index_fichiers/support/zynq_cours_tp_vivado_zc702.pdf de C. Alexandre et du site fpgadevelopper https://www.fpgadeveloper.com/2014/07/creating-a-base-system-for-the-zynq-in-vivado.html
2. https://reference.digilentinc.com/vivado/getting_started_with_zynq/start

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search