HAL
open science

Générateur d'Architecture de FPGA

Adrien Blanchardon, Roselyne Chotin-Avot, Habib Mehrez

▶ **To cite this version:**

Adrien Blanchardon, Roselyne Chotin-Avot, Habib Mehrez. Générateur d'Architecture de FPGA. Colloque GDR SOC-SIP, Jun 2012, Paris, France. pp.1-3. hal-00987369

HAL Id: hal-00987369

https://hal.science/hal-00987369v1

Submitted on 6 May 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

GENERATEUR D'ARCHITECTURE DE FPGA

Adrien BLANCHARDON, Roselyne CHOTIN-AVOT, Habib MEHREZ

Lip6

1. INTRODUCTION

Dans le cadre du projet Robust FPGA, qui traite de la tolérance aux défauts, nous sommes amenés à créer notre propre architecture FPGA. Il existe plusieurs types d'architecture (mesh, Tree et mesh of tree FPGA) qui présentent chacune leurs avantages. Nous allons donc étudier l'impact de l'architecture sur la robustesse selon divers paramètres (nb I/O, nb CLB etc…), il faut donc pouvoir facilement jouer sur les paramètres de l'architecture d'où la nécessité du générateur.

2. ARCHITECTURE FPGA

2.1. Mesh FPGA

Cette structure classique utilisée dans le commerce possède une topologie matricielle ou « mesh ». Une tuile composée de blocs logiques et de blocs d'interconnexion est dupliqué pour créer une matrice. Le principal avantage de cette structure est donc sa généricité. En effet, dans le cadre de la création d'un générateur de FPGA, l'utilisation d'éléments générique est essentielle.

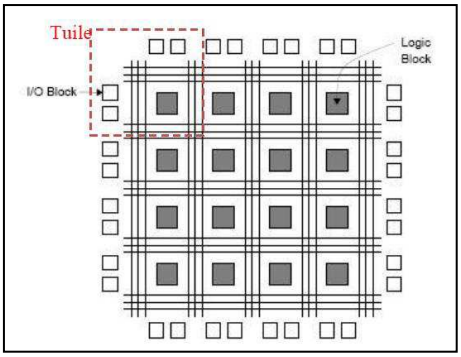

Figure 1 – Mesh FPGA [1]

2.2. Tree FPGA

Dans un FPGA classique (mesh) 90% de la surface est utilisé par les ressources d'interconnexion et seulement 10% par les éléments logiques, ce qui est préjudiciable en terme de délai et de surfaces. Une architecture en arbre appelé Tree FPGA a donc été créée et permet de gagner 40% en surface au niveau des réseaux d'interconnexion. De plus, les nombres d'éléments logiques ainsi que la structure des différents blocs d'interconnexion sont ajustable grâce à plusieurs paramètres architecturaux ce qui est un avantage pour la création de notre générateur.

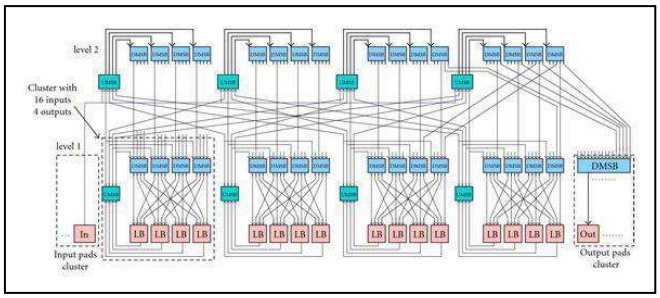

Figure 2 – Tree FPGA [2]

2.3. Mesh of Tree FPGA

Nous avons décidé de créer une architecture de type mesh of tree, car elle regroupe les avantages des deux structures décrites précédemment. Notre FPGA est ainsi composé d'une matrice de tuiles, chaque tuiles étant composé d'un cluster et de Switch boxes (blocs d'interconnections). Cette matrice permet d'avoir une généricité dans notre structure essentielle pour la création d'un générateur. De plus, les clusters et les SBs respectent quant à eux une architecture en arbre comme celle décrite ci-dessus.

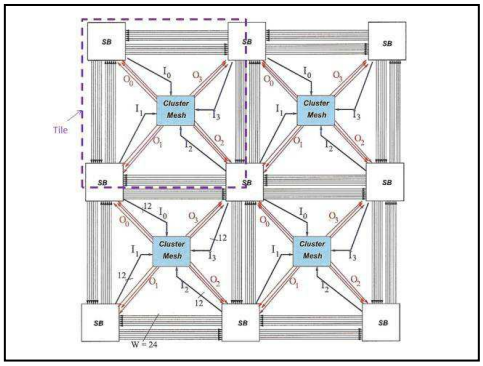

Figure 3 – Mesh of Tree FPGA [3]

Nous allons donc gagner en termes de surface et surtout disposer de plusieurs paramètres architecturaux qui sont là aussi indispensable pour la création d'un générateur. Pour finir, le générateur prendra comme paramètres le nombre d'I/O de notre FPGA, la taille des canaux de routage et le nombre de blocs logiques (CLBs).

3. INTERCONNEXION ENTRE LES SBS ET LES CLUSTERS

La spécificité de ce type de FPGA réside dans son réseau d'interconnexion. Chaque Switch boxes est reliées à ses 4 clusters voisins ainsi qu'à ses 4 switches boxes voisines. Avec ce type d'interconnexion, nous avons la possibilité ou la flexibilité de connecter des signaux externes (I/O de notre FPGA) ainsi que des feedback locaux (pour connecter 2 clusters ou plus ensemble ou avec une SB) destinés aux entrées de nos blocs logiques.

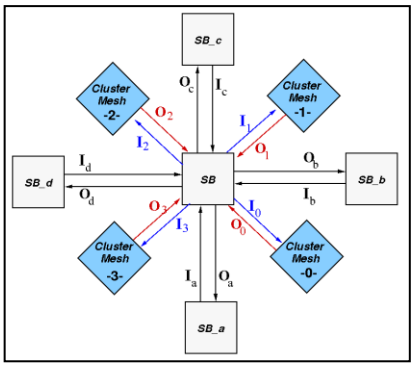

Figure 4 – Réseaux d'interconnexion [3]

4. GENERATEUR DE SB

Le premier élément essentiel à notre FPGA est donc la Switch Box. Toujours dans le cadre de la réalisation de notre générateur, le nombre d'entrées/sorties (lié au nombre d'entrées/sorties du FPGA), la taille des canaux de routage et le nombre de clusters seront les paramètres nous permettant de modifier la structure de nos SBs. Une Switch box doit être composé de 3 blocs spécifiques : un Up Mini Switch Box (sb5) et 2 types de Down Mini Switch Box (sb4 et sb6). Ce sont des blocs d'interconnexion composés de multiplexeurs. Les DMSBs sont utilisées pour connecter une SB à ses quatre voisins ainsi qu'à un DMSB comprit dans la même structure. Pour permettre la connexion entre 2 clusters avec le même SB, un full crossbar programmable upward switch bloc appelé UMSB (sb5) est ajouté. Chaque bloc d'interconnexion (sb4) est relié à chaque sb6 dans un ordre bien précis. Le 1er bloc (le plus à gauche) est connecté à l'entrée A du premier sb6, le bloc suivant à l'entrée A du second sb6, etc.… Pour finir, avec le réseau d'interconnexion de type Butterfly, nous avons accès à tous les blocs logiques présents dans tous les clusters (figure 5).

Pour bien comprendre comment sont organisés nos blocs, le nombre d'entrées est déterminé par le nombre d'I/O de notre FPGA, par la taille des canaux de routage et par le nombre de sorties par clusters. Le nombre de sortie est quant à lui directement lié au nombre d'entrées par clusters sauf pour les blocs sb4 dont les sorties vont directement sur les canaux de routage. Par exemple, si on fixe la taille des canaux de routage à 4, le nombre d'entrées par SBs à 90, le nombre de cluster à 4 avec 24 entrées et 8 sorties par cluster, on obtient une architecture comme celle indiqué ci-dessous.

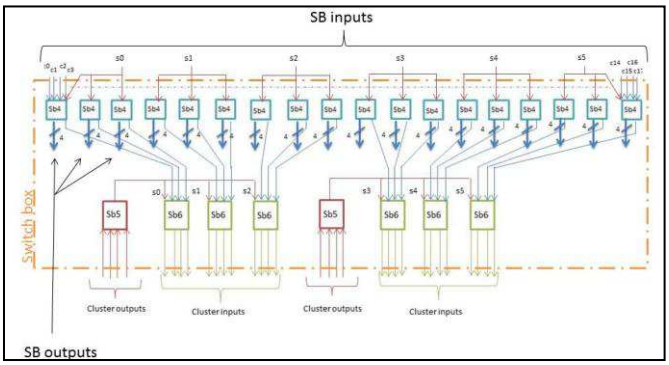

Figure 5 – Architecture d'une SB

5. GENERATEUR DE CLUSTER

Le second bloc essentiel à notre FPGA est donc le cluster. Basé sur une architecture de type « butterfly », chaque Down crossbar (blocs d'interconnections composés uniquement de multiplexeurs) est relié à tous les CLB dans un ordre bien précis. Le bloc le plus à gauche est connecté à toutes les premières entrées de chaque CLB, le bloc suivant à toutes les secondes entrées,… Pour permettre la connexion entre 2 CLBs d'un même cluster, une entité appelée UP a été ajoutée. Elle fonctionne de la même manière que les autres crossbar, on peut ainsi connecter n'importe quelle sortie de CLBs à n'importe quelles entrées. En fait, les sorties du bloc UP sont reliées à chaque Down crossbar ce qui permet d'avoir accès à chaque entrées de nos CLBs. Notre générateur prendra donc comme paramètre le nombre d'entrées/sorties de notre cluster qui sont déterminer par les nombres d'I/O par Sbs et donc par le nombre d'I/O de notre FPGA ainsi que le nombre de blocs logiques (CLBs) voulu par l'utilisateur. Par exemple, si l'on choisit de créer un cluster avec 24 entrées, 8 sorties et 10 CLBs, cela nécessitera 4 Down crossbar ainsi qu'un Up (Figure 6 ci-dessous). Toute l'architecture et toutes les connections étant créées par le générateur pour n'importe quelle combinaison d'entrées/sorties et pour n'importe quel nombre de blocs logiques.

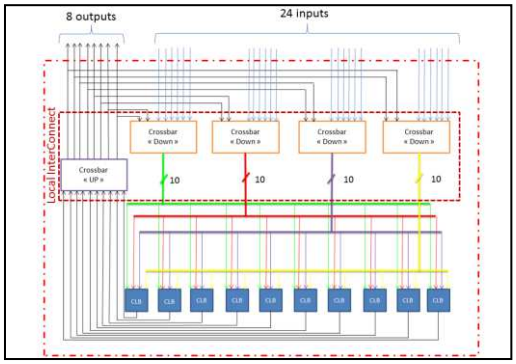

Figure 6 – Architecture d'un cluster

6. CONCLUSION

En conclusion, nous avons réalisé un générateur de FPGA de type mesh of tree. Dans le cadre de notre projet Robust FPGA, qui traite de la tolérance aux défauts de fabrication pour ce genre de circuits, notre générateur va nous permettre de modifier l'ensemble de nos blocs logiques et de nos blocs d'interconnexions. La reconfigurabilité de notre générateur nous permettra donc de tester aisément les différentes solutions existantes pour améliorer la robustesse de notre FPGA.

7. REFERENCES

[1] S. Brown and J. Rose, *Architecture of FPGAs and CPLDs*, University of Toronto, 1996.

[2] Z. Marrakchi, Exploration *et Optimisation d'architecture FPGA arborescentes*, LIP6, 2008.

[3] E. Amouri, *Outils de placement et de routage pour des architectures FPGA sécurisées contre les attaques DPA*, LIP6, 2011.