

TP3 VLSI - Flot de conception

Roselyne CHOTIN - Ali OUDRHIRI

Préambule

Lors de ce TP on va utiliser des outils *open-source* qui peuvent parfois ne pas fonctionner parfaitement. En cas d'échec d'un outil, on peut essayer une autre configuration. N'hésitez pas à demander de l'aide à l'enseignant si besoin et surtout à bien valider chaque étape en resimulant les *netlists* obtenues et en passant les outils de LVS et DRC.

Sujet

Le but de ce TP est d'obtenir le dessin des masques des circuits modélisés lors des TP précédents et de comparer les différentes architectures et possibilités données par les outils.

Pour cela on utilisera la chaîne Coriolis sur la machine virtuelle `a19` et le fichier `dodo.py` fourni dans le répertoire TP3 du GitLab.

Synthèse

La synthèse permet d'obtenir une *netlist* de portes interconnectées à partir d'une description comportementale en `verilog`.

A chaque synthèse, il faudra vérifier que celle-ci s'est bien passée en resimulant le circuit obtenu avec le *testbench* correspondant. Si vous trouvez des différences, notez-les dans le fichier `notes.md` en essayant de trouver ce qui a pu se passer.

Question 1

Synthétiser un des modèles de produit scalaire du TP précédent sur les différentes technologies fournies dans le répertoire du TP3. Que constatez-vous en termes de cellules utilisées? Que pouvez-vous en conclure?

Question 2

Maintenant pour une technologie donnée, synthétiser les différentes versions du produit scalaire. Que constatez-vous en termes de cellules utilisées? Que pouvez-vous en conclure?

Question 3

Lors des TP précédents, vous avez utilisé les *testbenches* qu'on vous avait fournis qui sont assez succincts. Pour valider de manière plus complète vos circuits, il est nécessaire d'écrire des *testbenches* permettant de les simuler de manière autonome (sans faire appel à `gtkwave`) et plus poussée. Il vous est donc demandé d'enrichir les *testbenches* fournis pour tester davantage de valeurs et vérifier les résultats en comparaison avec la valeur mathématique du produit scalaire. Pour cela vous pouvez par exemple utiliser la bibliothèque `numpy` en Python pour générer des fichiers contenant les entrées et sorties de votre circuits que vous pourrez relire dans le *testbench*.

Placement-routage

Le placement-routage (PNR) permet d'obtenir le dessin en masques du circuit. Il ne faut pas oublier de resimuler le circuit obtenu avec le même *testbench* afin de s'assurer qu'il n'y a pas d'erreurs.

Question 4

Lancer un premier PNR sans trop de contraintes. Quelle surface obtenez-vous ?

Question 5

Essayer d'obtenir la surface minimale en jouant sur le placement des connecteurs, la densité de placement et la taille du circuit. Quelle surface obtenez-vous ?

Validation

Les étapes de LVS (*Layout Versus Schematic*) et DRC (*Design Rule Checking*) permettent de valider le PNR en vérifiant qu'il n'y a pas d'erreurs de connectique et que les règles de dessin sont toutes respectées. Dans cette partie on utilisera une bibliothèque symbolique pour appréhender les principes de validation.

Question 6

Réaliser les DRC et LVS sur un de vos circuits afin de vous assurer que tout s'est bien passé. En cas de problème, reportez-le dans le fichier `notes.md`.

Question 7

Introduisez des erreurs de LVS et DRC dans ce circuit, passez-le à votre voisin afin qu'il essaye de les corriger. Rendez compte de ce que vous avez fait et trouvé dans le fichier `notes.md`.

Analyse de *timing*

L'analyse de *timing* statique (STA) permet d'obtenir les différentes chaînes longues du circuit et le temps de propagation.

Question 8

Déterminer la chaîne longue de votre circuit, combien fait-elle et par où passe-t-elle ? Essayer de la reconstituer à partir de la *netlist* issue de la synthèse. Comment pourriez-vous l'améliorer ?

Analyse des architectures

Lors des TP précédents, vous aviez déterminé de manière empirique la taille et la fréquence de vos circuits. Tracez les mêmes graphes avec maintenant les valeurs réelles. Retrouvez-vous les mêmes ordres de grandeur ? Qu'en concluez-vous ?

Rendus

A l'issue de ce TP, vous devez avoir rajouté dans votre dépôt git :

- les *testbenches* évolués de vos circuits,
- le fichier `doDesign.py` permettant d'obtenir le plus petit circuit

Et bien sûr le fichier `notes.md` mis à jour avec les réponses aux questions et les graphes demandés