TP4 : AM2901

Le but de ce TP est d'utiliser les outils de placement / routage automatique du flot **Coriolis/Alliance?** ainsi que tous les outils de vérification vus dans les TPs précédents, pour générer le dessin des masques du circuit AM2901.

Vous avez dans les TPs précédents appris à utiliser le langage **Stratus** pour décrire des netlists hiérarchiques et des directives de placement manuel.

Vous allez maintenant utiliser **Stratus** pour définir des directives de placement/routage automatiques.

Le routage final sera effectué par l'outil **nero**.

Vous utiliserez également **cougar** pour obtenir une netlist extraite, et **lvx**, pour comparer la netlist extraite à la netlist initiale.

1 Architecture interne du circuit Am2901

La description générale du processeur AM2901 est donnée par : ?ftp://asim.lip6.fr/pub/amd2901/amd2901.pdf.

Nous décomposons le circuit en 2 blocs : la partie contôle, et la partie opérative ou chemin de données.

- Le chemin de données contient les parties régulières de l'Amd2901 c'est à dire les registres et l'unité arithmétique et logique.
- La partie contrôle contient la logique irrégulière, c'est à dire le décodage des instructions et le calcul des "drapeaux" (indicateurs, ou "Flags").

Nous utiliserons la description hiérarchique suivante :

]

Les Fichiers fournis sont les suivants :

- description du comportement de la partie contrôle de l'AM2901
- description logique de la partie chemin de données de l'AM2901
- description logique du coeur de l'AMD2901
- description logique du circuit contenant les plots et le coeur de l'AM2901
- script python de création du circuit AM2901
- le fichier de vecteurs de test de l'AMD2901
- Catalogue des modèles

2 Partie contrôle

2.1 Description comportementale

- Etudiez le fichier amd2901_ctl.vbe fourni (vous pouvez entre autres vérifier qu'il correspond bien aux données founies).
- Générez la vue structurelle de l'AM2901 avec le script python fourni.
- Lancez la simulation avec **asimut** (Vérifiez que le fichier CATAL indique bien au simulateur qu'il faut utiliser la description comportementale (*.vbe*) de la partie controle).

```
> asimut amd2901_chip pattern resultat
```

2.2 Synthèse

On souhaite réaliser la vue structurelle de la partie contrôle de l'Amd2901 à l'aide de la vue comportementale fournie.

- Utilisez les outils de synthèse de la chaîne **Alliance** pour réaliser la synthèse logique avec les cellules pre-caractérisées de **sxlib**.

2.3 Validation du schéma de la partie contrôle

- Utilisez de nouveau **Asimut** pour valider le schéma obtenu en simulant le circuit complet avec les vecteurs de test fournis. Penser à remplacer la vue comportementale de la partie contrôle par la vue structurelle en ôtant le nom **amd2901_ctl** du fichier **CATAL**.

```
> asimut -zerodelay amd2901_chip pattern resultat
```

Notez que l'on réalise une simulation "zero délai" de la netlist.

En cas de problème(s), n'hésitez pas à utiliser **XPAT**.

3 Chemin de données

Le chemin de données est formé de la logique régulière du circuit.

Afin de profiter de cette régularité, on utilise les opérateurs vectoriels de la bibliothèque **Dpgen**. Cela permet d'optimiser le schéma en utilisant plusieurs fois le même matériel. Par exemple, les amplificateurs des signaux de commande d'un multiplexeur sur n bits sont partagés par les n bits ...

3.1 Description structurelle

Le chemin de données de l'Am2901 peut être schématisé par les figures ci-dessous.

- Etudiez Le fichier fourni décrivant le chemin de données.

3.2 Placement

Le fichier fourni comporte non seulement la description de la netlist du chemin de données mais aussi le placement explicite des colonnes représentant les différents opérateurs 4 bits du chemin de données les unes par rapport aux autres.

- Faites appel à la méthode *View* pour visualiser le placement généré.
- Etudiez le placement choisi : vérifiez entre autres que les colonnes ayant un grand nombre d'interconnexions communes sont *proches*

3 Placement / Routage

TODO

4 Rapport

TODO