TP5 : Dessin de cellules précaractérisées

Introduction :

Le but de ces quatre séances de TP est de présenter quelques outils de la chaîne ALLIANCE ainsi que du flot back-end CORIOLIS, dont :

Le langage de description procédural STRATUS

- L'éditeur de layout GRAAL
- Le vérificateur de règles de dessin DRUC
- L'extracteur de netlist COUGAR
- L'outil de placement du flot CORIOLIS MISTRAL
- L'outil de routage de la chaîne ALLIANCE NERO

La première séance portera sur le dessin sous GRAAL d'une cellule inverseuse et d'un buffer instanciant cet inverseur.

Les notions de cellules précaractérisées, de gabarit et de hiérarchie de cellules seront introduites.

1-Dessin d'un inverseur et d'un buffer sous GRAAL

Introduction

Dans les TP précédents nous avons utilisé des cellules d'une bibliothèque. Cette biblioth

èque peut être enrichie de nouvelles cellules grâce à l'éditeur GRAAL.

GRAAL est un éditeur de layout symbolique intégrant le vérificateur de règles de dessin DRUC.

La première partie de cette séance a pour objectif de dessiner une cellule inverseuse inv_x1 sous la forme d'une cellule précaractérisée sxlib en respectant règles de dessin fournies.

Cette cellule sera instanciée pour concevoir un buffer.

1.1 Environnement technologique

Certains outils utilisent un environnement technologique particulier. Il est désigné la variable d'environnement RDS_TECHNO_NAME qui doit être positionnée à

opt/alliance/etc/cmos.rds :

```
> export RDS_TECHNO_NAME=/opt/alliance/etc/cmos.rds
```

1.2 GRAAL

L'éditeur de layout GRAAL manipule six types d'objets différents que l'on peut créer avec le menu CREATE :

- Les instances (importation de cellules physiques)
- Les boîtes d'aboutement qui définissent les limites de la cellule
- Les segments : DiffN, DiffP, Poly, Alu1, Alu2 ...
- Le CAluX est utilisé pour désigner une portion possible pour les connecteurs.
- Les VIAs ou contacts : ContDiffN, ContDiffP, ContPoly? et Via Metal1/Metal2.
- Les Big VIAs
- Les transistors : NMOS ou PMOS

GRAAL utilise la variable d'environnement GRAAL_TECHNO_NAME. Elle doit

être positionnée à /opt/alliance/etc/cmos.graal.

```
> export GRAAL_TECHNO_NAME=/opt/alliance/etc/cmos.graal
```

1.3 COUGAR

L'outil COUGAR est capable d'extraire la netlist d'un circuit aux formats .vst ou .al à partir d'une description au format .ap.

Pour extraire au niveau transistor, la commande à utiliser est :

```
> cougar -t file1 file2
```

COUGAR utilise les variables d'environnement MBK_IN_PH et MBK_OUT_LO suivant les formats d'entrée et de sortie. Par exemple pour générer une netlist au format al à partir d'une description ap il faut écrire :

```
> export MBK_IN_PH=ap

> export MBK_OUT_LO=al
```

```
> cougar -t file1 file2
```

2 Schéma d'un inverseur

Le schéma théorique d'un inverseur est présenté

3 Schéma d'un buffer

Le schéma théorique d'un buffer et la hiérarchie utilisée sont présentés

4 Le gabarit sxlib

- Les cellules sxlib ont toutes une hauteur de 50 lambdas et une largeur multiple de 5 lambda
- Les alimentations Vdd et Vss sont réalisées en Calu1 ; elles ont une largeur de 6 lambdas et sont placées horizontalement en haut et en bas de la cellule
- Les transistors P sont placés près du rail Vdd tandis que les transistors N sont placés près du rail Vss
- Le caisson N doit avoir une hauteur de 24 lambdas
- Les segments spéciaux CAluX (CAlu1, Calu2, CAlu3...) forment l'interface de la cellule et jouent le rôle de connecteurs "étalés". Ils doivent obligatoirement être placés sur une grille de 5x5 lambdas et peuvent se trouver n'importe où à l'intérieur de la cellule
- Les segments spéciaux TAlux (TAlu1, TAlu2, ...) servent à désigner les obstacles au routeur Lorsque vous voulez protéger des segments AluX, il faut les recouvrir ou les entourer de TAlux correspondant (même couche). Les TAluX sont placés sur une grille au pas de 5 lambdas
- La largeur minimale de CAlu1 est de 2 lambdas, plus 1 lambda pour l'extension
- Les caissons N et P doivent être polarisés. Il faut donc les relier respectivement à Vdd et à Vss

5 Travail à effectuer

5.1 Réalisation d'un inverseur

- Décrire le comportement de la cellule inverseuse dans un fichier .vbe
- Dessiner le "stick-diagram" de l'inverseur inv_x1 dont le schéma en transistors

  est représenté
- Saisir sous GRAAL le dessin de la cellule en respectant le gabarit spécifié
- Valider les règles de dessin symbolique en lançant DRUC sous GRAAL
- Extraire la netlist de l'inverseur au format .al avec COUGAR

5.2 Réalisation d'un buffer

Le buffer est réalisé sous GRAAL à partir de l'instanciation de deux inverseurs. La hiérarchie ainsi créée est représentée sur la Le schéma en transistors est

représenté sur la

Décrire le comportement de la cellule buffer dans un fichier .vbe

Saisir sous GRAAL le dessin de la cellule en respectant le gabarit spécifé.

Vous utiliserez pour cela la fonction d'instanciation de GRAAL. La cellule à instancier est bien sûr l'inverseur créé précédemment, que vous relierez

(routerez) manuellement

Valider les règles de dessin symbolique en lançant DRUC sous GRAAL

Extraire la netlist du buffer au format .al avec COUGAR

Extraire le VHDL comportemental avec YAGLE

Effectuer la preuve formelle entre le fichier .vbe extrait par YAGLE et le fichier

.vbe de la spécification initiale

Automatisez la vérification en écrivant un Makefile.

N'oubliez pas que les mans existent