Allianceチュートリアル
仮想配置エディタGraalの使い方

飯田佳洋
Yoshihiro Iida
3aepm001@keyaki.cc.u-tokai.ac.jp

平成16年5月17日

1 はじめに

本ドキュメントではAlliance CAD System 5.0における仮想配置エディタプログラムGraalの使用方法を解説します。`man graal`も参考にして読み進めてください。

2 起動方法

Graalを起動するにはMBK環境変数やRDS環境変数などを必要としますので、他のAllianceプログラムと同様に`alc_env.sh`(sh, bashの場合)を環境変数に加えてください。

Graalの起動方法は次のように行います。

`> graal -l` *filename*

仮想配置ファイル名は`.ap`であり、ここでは拡張子を略した形で指定します。

3 Graalの基本的な使い方

Graalはエディタまたはビューアとして使用します。エディタとして用いるときに主に使うのは、ツールバーの`create ->`以下になります。

仮想配置では2グリッドが1 λとなり、1 λ=1 μ mのときに2グリッドで1 μ mとなります。

3.1 Instance

インスタンスとして他の仮想配置ファイル(`*.ap`)をインクルードします。

3.2 Abutment Box

仮想配置ファイルの外枠のラインを描画します。Abutment Box ラインは横 5 グリッド、縦 50 グリッドの倍数にしなければならない Alliance 上のルールがあります。

3.3 Segment

Segment サブウィンドウを開き、Nwell、Ndif、Poly や Alu1 といった層ごとに配置します。Segment の各レイヤは Graal 環境設定ファイル[1] で最低描画幅・間隔が指定され、ルールを満たしていないときにエラーを表示します。

CAlu は回路の入出力端子とするときに端子に名前をつけるものです。アルミの各レイヤごとに存在し、CAlu を配置後に名前を入力するウィンドウが開くのでここに任意の端子名を指定します。

3.4 Via

上記の Segment の各レイヤを結ぶための Via を配置します。Via では最低間隔のルールに注意してください。

3.5 Big Via

セル内の配線ではなく、電源リング等の大規模な配線における Via を配置するときに用います。配置は通常の Via と異なり面積で指定し、その面積の中にルールに合わせた Via を配置します。

3.6 Connector

Calu と同様に入出力端子に名前を付けるものですが、既に配線したものに点で配置するところが異なります。Connector は Abutment Box が無い場合には配置できません。

3.7 Transistor

N または P トランジスタを配置します。トランジスタに名前を指定したり、ゲート幅をグリッド単位で指定することができます。

3.8 Reference

Reference は Alliance の他のプログラムから参照するときに点で指定します。Alliance では自動配線を行うときに VDD,VSS に関しては縦 50 グリッドルールに基づいて配線を行い、セル間配線には CAlu と Reference で指定した点を使用します。Connector を用いたときには Reference は使用しません。

[1] 標準では `GRAAL_TECHNO_NAME="/usr/local/alliance/etc/cmos.graal"`

4 回路の抽出

Graal で編集した回路はネットリストを抽出することができ、ここでは Spice ネットリストを抽出する方法を解説します。環境によって異なりますが以下のように環境変数が設定されているものとします。

`MBK_OUT_LO=sp`

`MBK_SPI_MODEL=/usr/share/alliance/etc/spimodel.cfg`

ここで Alliance の回路抽出プログラム `cougar` を用い、以下のように実行します。

`> cougar -t -ar -ac` *filename*

以上で Spice ネットリストの抽出を行うことができます。

5 Spiceシミュレーション

抽出した回路を用いて Spice シミュレーションを行います。`cougar` で抽出した回路はサブサーキットとして作られ、`.subckt` で宣言されています。抽出した Spice ネットリストを図 1 とします。このネットリストを使ったシミュレーションを行うためにシミュレーション用ネットリスト (図 2) を作成し、抽出した回路をインクルードします。図 2 の中の `x1` という記述がサブサーキットをインスタンス化して用いるもので、頭文字が `x` であれば以下は任意の文字を使うことができます。インスタンス生成の書式は以下のようになります。

x*instance_name signal signal ... included_subcircuit*

上記の signal も任意の名前を付けることができ、Spice シミュレーション時に `plot` で指定することができます。

```
* Spice description of test
* Spice driver version 134940126
* Date ( dd/mm/yyyy hh:mm:ss ):  8/05/2004 at 12:42:14

* INTERF a f m_clock p_reset vdd vss


.subckt test 4 6 3 2 5 1
* NET 1 = vss
* NET 2 = p_reset
* NET 3 = m_clock
* NET 4 = a
* NET 5 = vdd
* NET 6 = f
Mtr_00002 6 4 5 5 tp L=1U W=15U AS=30P AD=30P PS=34U PD=34U
Mtr_00001 1 4 6 1 tn L=1U W=10U AS=20P AD=20P PS=24U PD=24U
C6 1 1 2.444e-14
C5 2 1 4.8e-16
C4 3 1 4.8e-16
C3 4 1 3.027e-14
C2 5 1 2.444e-14
C1 6 1 3.032e-14
.ends test
```

図 1: 抽出した Spice ネットリスト

```
test

.include test.sp
* NET 1 = vss
* NET 2 = p_reset
* NET 3 = m_clock
* NET 4 = a
* NET 5 = vdd
* NET 6 = f

x1 a f m_clock p_reset vdd vss test

v0 vdd 0 dc 3.3
v1 vss 0 dc 0
v2 p_reset 0 dc 0
v3 m_clock 0 dc 0
v4 a 0 pulse(0 3.3 10n 0 0 10n 20n)

.tran 1n 100n


.model tp pmos (level=2 vto=-1.0 tox=0.035u nsub=1e16 uo=250
+ ucrit=2.0e4 uexp=0.08 tpg=-1 ld=0.3u xj=0.43u js=6m pb=0.88
+ cj=1.23e-4 cjsw=2.36n cgdo=0.25n cgso=0.25n cgbo=0.02n
+ rd=1 rs=1)

.model tn nmos (level=2 vto=-1.0 tox=0.035u nsub=1e15 uo=500
+ ucrit=4.5e4 uexp=0.11 tpg=-1 ld=0.2u xj=0.29u js=6m pb=0.94
+ cj=3.85e-4 cjsw=1.6n cgdo=0.45n cgso=0.45n cgbo=0.042n
+ rd=1 rs=1)


.end
```

図 2: 抽出した Spice ネットリストを用いたシミュレーション回路