Alliance CAD System を用いたLSIチップ試作の研究

情報理工学部 ソフトウェア開発工学科

6ADF2119 樋口拓哉

指導教員　清水尚彦

平成 22 年 1 月 31 日

目 次

図 目 次

表 目 次

1 はじめに

1.1 まえがき

高集積化の親展の伴い、1チップ上に実現するシステムLSIの必要性が高まっている。システムLSIにはメモリ、プロセッサなどシリコン上せ実現できるすべての機能素子の集積化が要求される。したがって従来の設計手法で主流であったゲートアレイからがらスタンダードセル手法に変わった。スタンダードセルは設計済みの機能ブロックを配置し、これらの間の配線層を作るため、小面積にでき高性能化が可能となる。しかし、下地から作る分試作期間が長くなる。そのため、CADを駆使した設計期間の短縮が必要となる。

1.2 研究目的

LSI設計に着手し実際にデバイスが完成するまでには、多くの設計工程とそれに伴う開発コストが必要となる。設計には複数の設計自動化ツール(EADツール)を使用するが。EADツールには、Cadence ,Synopsys ,Mentor Graphics ,Celoxlca 社などのCADツールがあるが、使用するためには多額のライセンス料を払う必要がある。近年では、ライセンスフリーのEADツールも開発されているが、開発事例は少ない。今回使用する、Alliance CAD Systemは、フランスのPierre et Marie Curie Universityで開発されたライセンスフリーで配布されているLSI統合開発環境で、VHDLやネットリストを入力し、GDSまたはCIFフォーマットを実配置として出力することができる。本研究では、Allianceを使用しLSIチップ試作を試みた。

2 設計環境

2.1 Alliance CAD System

Allianceでは、付属されているセルライブラリを使用し、サブミクロンプロセスの中でスケーラブルに設計が可能である。また、論理合成時の回路規模、遅延時間の論理最適化やネットリストの抽出、レイアウトエディタ・ビュアなどのツールが統合されている。

2.2 記述ファイル

sfl(.sfl)

SFLはRTL(Register Transfer Level)での記述が可能で、完全同期式の回路の設計ができるハードウェア記述言語。

vhdl(.vhdl)

VHDLはハードウェア記述言語の一種で幅広い分野の記述が可能。

vbe(.vbe)

VBE(VHDL Behaviour）は、データフローレベルの記述。

vst(.vst)

VST(VHDL structural)は、ネットリストの拡張子。

仮想配置ファイル (.ap)

仮想配置ファイルは、各仮想レイヤの配置が座標として記述されている。

実配置ファイル (.cif,.gds)

実配置ファイルは、各実配置レイヤの配置が座標として記述されている。

RDS ファイル (.rds)

RDS(Rectangle Data Structure)ファイルは、Allianceで使うテクノロジーファイルで、仮想配置から実配置の変換にはこれを使う。

PAD リングファイル (.rin)

PADリングの配置場所が記述されている。

2.3 Alliance ツール

今回LSI試作を行う上で、使用したプログラムである。

vasy

vasy は VHDL Analyzer for SYnthesis の略で、VHDL の合成可能な動作記述から、Alliance データフロー記述への変換を行う。

boom

boom は Boolean Minimaization の略で、Alliance データフローレベルでの遅延時間等の最適化を行う。

boog

boog は Binding and Optimaizing On Gates の略で、sxlib を用いた論理合成を行い、ネットリストとして出力する。ツールに通す際に、オプションで回路規模、遅延時間、どちらの最適化を有線するか決めることができる。

ocp

ocp は Standard Cell Placer の略で、ネットリストもとにスタンダードセルの配置を行う。処理後は、セルのみが配置された仮想配置ファイルが出力される。

nero

nero は Negotiating Router の略で、セルの配置された仮想配置ファイルに配線を行う。

ring

ring は PAD RING router という名前で、PAD リングの配置パラメータ、PAD のネットリスト、コアの仮想配置ファイルをもとに、padlib の配置配線を行う。

s2r

s2r は Symbolic to Real layout の略で、レイヤーの最小幅、間隔などのパラメータの記述されたテクノロジーファイルを使い、仮想配置ファイルから実配置ファイルへの変換をう。

xsch

xsch は、ネットリストファイルの情報をグラフィカルに表示することができる。

graal

graal は Graphic Layout editor の略で、仮想配置ファイルの情報をグラフィカルに表示でき、また仮想レイヤーを記述することもできる。テクノロジーファイルを環境変数に設定することで、実配置に似たレイアウトを見ることができる。

dreal

dreal は Design REAL layout の略で、実配置ファイルをグラフィカルに表じすることがで、また実レイヤーを記述することができる。

cougar

cougar は、仮想配置、実配置から寄生容量等の RC 成分を含む SPICE ネットリストを抽出することができる。RC 成分は、テクノロジーファイルに記述する。

3 設計手順

ここでは、Alliance ツールの作業工程を説明していく。例として入力 (a,b,enable)、出力 (s,c) の enable 付き 4bit 加算器 (adder4.vhdl) を用いる。

3.1 Alliance 設計手順

3.2 論理合成

最初に VHDL から vasy を使い Alliance データフロー記述 (.vbe) への変換を行う。

vasy -aopI vhdl halfadder.vhdl

オプションの-p は電源の追加を意味する。入力する VHDL が階層構造の場合は-H を付け加える。実行後に adder4.vbe が出力される。
出力された.vbe ファイルのデータフロー最適化には boom を使う。今回は最適化前とファイルの区別するために別のファイル名で出力する。

boom -i 3 -l 0 adder4 adder4_o

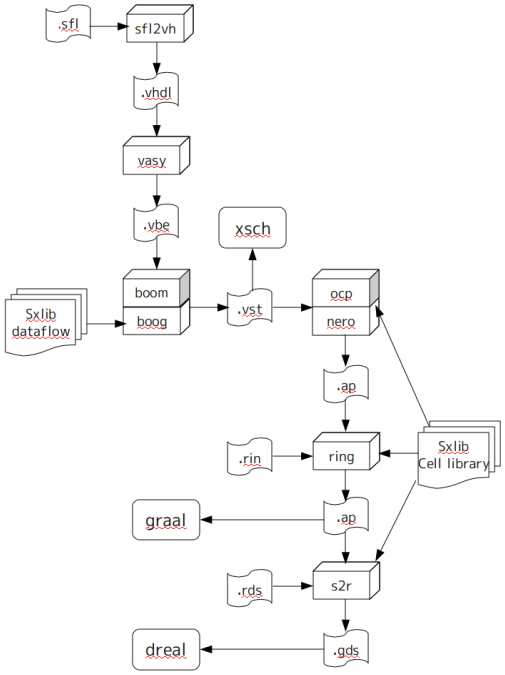

図 1: Alliance 設計手順

-i は最適化を実行する回数の指定、-l は (0-3) で最適化レベルの指定ができる。実行後に adder4_o.vbe が出力される。
論理合成を行うには boog を使う。boog では、sxlib をターゲットに VHDL ネットリストを合成する。実行の際に、sxlib を用いたときの回路規模、遅延時間の見積りを算出される。

boog -m 0 adder4_o adder4

-m は最適化の優先度の指定で、(0-3) までの指定ができ、少なければ規模優先、多ければ遅延優先となっている。出力ファイルは、ネットリストファイルと.xsc ファイルが出力され、xsch で回路を見ることができる。実行後に、adder4.vst、adder4.xsc が出力される。
xsch でネットリストを参照する場合は、以下のコマンドを実行する。

```
Controlling lib '/opt/alliance/cells/sxlib'...
Preparing file 'halfadder_o.vbe'...
Capacitances on file 'halfadder_o.vbe'...
Unflattening file 'halfadder_o.vbe'...
Mapping file 'halfadder_o.vbe'...
Saving file 'halfadder.vst'...
Quick estimated critical path (no warranty)...627 ps from 'n_a' to 'n_s'
Quick estimated area (with over-cell routing)...5750 lambda
Details...
        nao22_x1: 1
        na3_x1: 1
        inv_x2: 1
        an12_x1: 1
        na2_x1: 1
        Total: 5
Saving critical path in xsch color file 'halfadder.xsc'...
End of boog...
```

xsch -l adder4

3.3 配置配線

論理合成で出力されたネットリストを元に、ocp を使ってゲートセルの配置を行う。

ocp -ring -v adder4 adder4

出力された仮想ファイルをさらに nero を使い自動配線する。

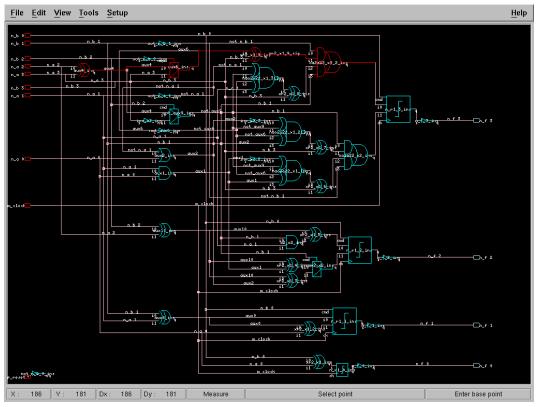

図 2: adder4 ネットリスト

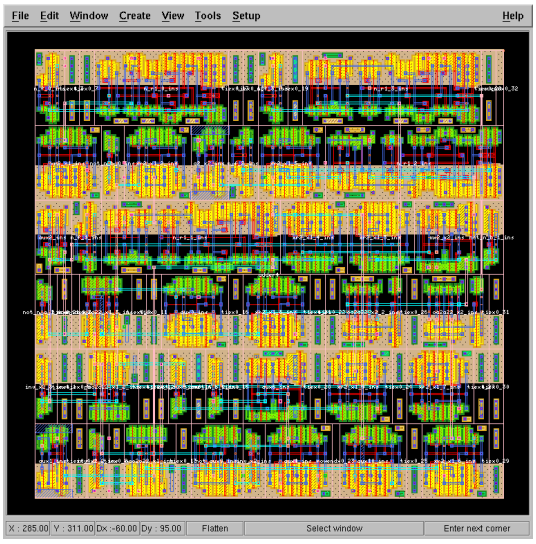

図 3: adder4 仮想配置

nero adder4 adder4

仮想配置ファイルを参照したい場合は以下のコマンドを実行する

graal -l adder4

graal の表示は、テクノロジーファイル (.rds) ファイルを参照するため。実が位置に近いレイアウトを表示させることができ。また、環境変数 GRAAL_TECHNO_NAME に graal 設定ファイルを指定すればレイヤの色を任意に指定できる。

3.3.1 仮想配置ファイル

仮想配置ファイルには、各レイヤの幅、座標が記述されている。今回は sxlib の inv_x1.ap を使い解説する。

3.4 実配置

s2r を使い、仮想配置ファイルからテクノロジファイルにしたがって実配置ファイルへの変換を行う。テクノロジーファイルについては以下で詳しく説明する。

s2r -v -r adder4 adder4

実配置ファイルを参照したい場合は以下のコマンドを実行する

dreal -l adder4

3.5 電源・クロックリング、IO パッドの配置配線

3.6 SPICE 検証

4 Alliance を用いた LSI 設計試行

Alliance では、スタンダードセル方式で設計する。そのためレイアウトパターンの配置配線の後に実配

表 2: .ap ファイル

```
V ALLIANCE : 6
H inv_x1,P, 8/ 6/2002,100
A 0,0,1500,5000
R 1000,4000,ref_ref,nq_40
R 1000,3500,ref_ref,nq_35
R 1000,3000,ref_ref,nq_30
R 1000,2500,ref_ref,nq_25
R 1000,2000,ref_ref,nq_20
R 1000,1500,ref_ref,nq_15
R 1000,1000,ref_ref,nq_10
R 500,1000,ref_ref,i_10
R 500,1500,ref_ref,i_15
R 500,2000,ref_ref,i_20
R 500,2500,ref_ref,i_25
R 500,3000,ref_ref,i_30
R 500,3500,ref_ref,i_35
R 500,4000,ref_ref,i_40
S 1000,4300,1000,4800,300,*,DOWN,NTIE
S 1000,1000,1000,4000,200,*,DOWN,ALU1
S 1000,2800,1000,3700,300,*,DOWN,PDIF
S 700,2600,700,3900,100,*,UP,PTRANS
S 1000,800,1000,1200,300,*,UP,NDIF
S 700,600,700,1400,100,*,DOWN,NTRANS
S 0,300,1500,300,600,vss,RIGHT,CALU1
S 0,4700,1500,4700,600,vdd,RIGHT,CALU1
S 400,2000,700,2000,300,*,RIGHT,POLY
S 700,1400,700,2600,100,*,UP,POLY
S 500,1000,500,4000,100,*,DOWN,ALU1
S 0,3900,1500,3900,2400,*,RIGHT,NWELL
S 350,400,350,1200,400,*,UP,NDIF
S 350,2800,350,4600,400,*,DOWN,PDIF
S 1000,1000,1000,4000,200,nq,DOWN,CALU1
S 500,1000,500,4000,200,i,DOWN,CALU1
V 1000,3000,CONT_DIF_P,*
V 400,4500,CONT_DIF_P,*
V 1000,3500,CONT_DIF_P,*
V 400,500,CONT_DIF_N,*
V 1000,1000,CONT_DIF_N,*
V 500,2000,CONT_POLY,*
V 1000,4700,CONT_BODY_N,*
EOF
```

置へ変換する必要がある。ファウンドリから指定されるルールファイルには、各レイヤの最小幅、最小間隔、面積等の指示があり、それに沿った実配置のレイアウトを作成する必要がある。

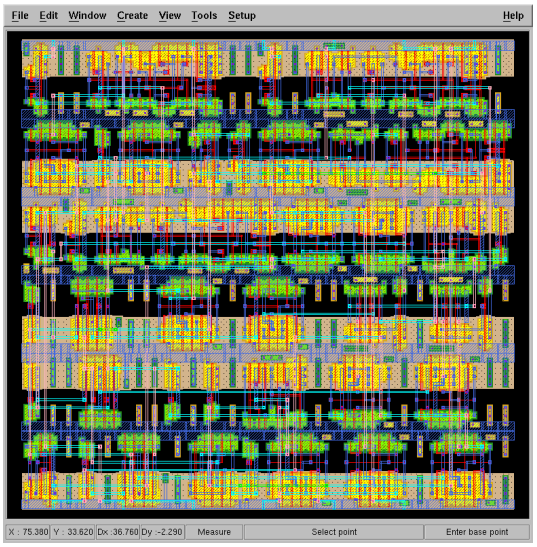

図 4: adder4 実配置

表 3: Alliance 付属ツール

ツール名	使用用途
sfl2vh	SFLl から論理合成可能な動作記述へ変換
vasy	VHDL を Alliance VHDL データフロー記述に変換
boom	データフローレベルの論理最適化
boog	sxlib を用いた論理合成
ocp	セルの自動配置
nero	自動配線
ring	電源・クロックリング、IO パッドの自動配置配線
s2r	仮想配置から実配置への変換
xsch	ネットリストファイル・ビューア
graal	仮想配置ファイルエディタ・ビューア
dreal	実配置ファイルエディタ・ビューア
cougar	ネットリスト自動抽出

4.1 テクノロジーファイルの適合

4.1.1 LAMBDA

RDS ファイルでは、まず最初に基準値にあたるλの値をμm単位で決定する。例えば、LAMBDA 0.5 と宣言した場合、1 λ=0.5 μmとなる。inv_1.ap ファイルで、A 0,0,1500,5000 と宣言されているため、このセルは幅 5 λ、高さ 50 λとなっている事が分か

る。この場合、実配置にした時、幅 2.5 μm、高さ 25 μmに変換される。基準値の設定は、適応するプロセスルールのゲート幅になるべく近づける事が望ましい。

DEFINE LAMBDA *value*

4.1.2 PHYSICAL_GRID

PHYSICAL_GRID は、実配置での最小グリッドを設定でする。各レイヤ幅の最小値はこれより小さくすることができない。

DEFINE PHYSICAL_GRID *value*

4.1.3 MBK_TO_RDS_SEGMENT

MBK_TO_RDS_SEGMENT は、各レイヤの最小幅を設定するパラメータである。MBK には仮想レイヤ名が入り、RDS には実レイヤ名が入力される。各仮想レイヤに対応する最小幅は決められており、それを元に DLR、DWR を設定する。各仮想レイヤの最小幅の値は、/opt/alliance/etc/cmos.graal の GRAAL_SEGMENT_VALUE に記述されている。

TABLE MBK_TO_RDS_SEGMENT
MBK RDS
name name Type DLR DWR OFFSET MODE
name Type DLR DWR OFFSET ...

表 4: 仮想レイヤの最小幅

TABLE GRAAL_SEGMENT_VALUE

MBK layer	Ws	Ls	RDS layer
NWELL	4	4	RDS_NWELL
PWELL	4	4	RDS_PWELL
NDIF	2	2	RDS_NDIF
	2	2	RDS_ACTIV
	2	2	RDS_NIMP
	2	2	RDS_PWELL
PDIF	2	2	RDS_PDIF
	2	2	RDS_ACTIV
	2	2	RDS_PIMP
	2	2	RDS_NWELL
NTIE	2	2	RDS_NTIE
	2	2	RDS_ACTIV
	2	2	RDS_NIMP
	2	2	RDS_NWELL
PTIE	2	2	RDS_PTIE
	2	2	RDS_ACTIV
	2	2	RDS_PIMP
	2	2	RDS_PWELL
NTRANS	1	4	RDS_POLY
	1	4	RDS_NDIF
	1	4	RDS_NDIF
	1	4	RDS_NDIF
	1	4	RDS_ACTIV
	1	4	RDS_NIMP
	1	4	RDS_PWEEL
PTRANS	1	4	RDS_POLY
	1	4	RDS_PDIF
	1	4	RDS_PDIF
	1	4	RDS_PDIF
	1	4	RDS_ACTIV
	1	4	RDS_PIMP
	1	4	RDS_NWEEL
POLY	1	1	RDS_POLY
ALU1	1	1	RDS_ALU1
ALU2	2	1	RDS_ALU2
ALU3	2	1	RDS_ALU3
ALU4	2	1	RDS_ALU4
ALU5	2	1	RDS_ALU5

DLR

DLR は、実レイヤの長さ (Lr) 設定するパラメータで、以下の公式でそれぞれ求めることがでる。また、DLR を設定する際は、最小幅にする必要があるので Lr の値は GRAAL_SEGMENT_VALUE の値を使って求める。

$$Lr = Ls\lambda + 2DLR \quad (1)$$

$$DLR = \frac{Lr - Ls\lambda}{2} \quad (2)$$

DWR

DWR は、実レイヤの幅 (Wr) を設定するパラメータで、実レイヤの長さは以下の公式より求めることができる。DWE の値を求める際は、DLR の時と同様 Ws は最小幅を用いる。

$$Wr = Ws\lambda + DWR \quad (3)$$
$$DWR = Wr - Ws\lambda \quad (4)$$

Type

Type には VW,LEW,RCW の三種類あり、VW(Variable Width) は、幅が一定でない事を宣言します。Transistor 意外はほとんどこのパラメータが入る。
LCW(Left Constant Width)、RCW(Right Constant Width) はそれぞれ左右の Transistor 上の Diffusion 幅の宣言に使う。ネットリストを抽出した際、Source と Drain をつなげてはいけないため、左右の宣言を別々に行う。

MODE

MODE には ALL、DRC、EXT の三種類あり、Dreal、Graal を実行した際、変換公式を用いてレイヤを表示させるかを設定する。ALL は両方表示させ、DRC は Dreal、EXT は Graal で表示させることを意味する。

OFFSET

OFFSET の値は、レイヤの軸をμm単位で動かす値である。仮想配置ファイルに記述されている、UP、DOWN、RIGHT、LEFT は始点から終点への方向を意味しており、それぞれの軸の方向へ移動させる。

4.1.4 MBK_TO_RDS_CONNECTOR

4.1.5 MBK_TO_RDS_REFERENCE

4.1.6 MBK_TO_RDS_VIA

4.1.7 S2R_POST_TREAT

4.1.8 S2R_OVERSIZE_DENOTCH

4.1.9 S2R_BLOC_RING_WIDTH

4.1.10 S2R_MINIMUM_LAYER_WIDTH

4.1.11 CIF_LAYER,GDS_LAYER

s2r 変換公式

VW

UP

$$Xr = Xs\lambda - (Ws/2)\lambda - DWR/2 + OFFSET \quad (5)$$
$$Yr = Ys\lambda - DLR \quad (6)$$
$$dXr = Ws\lambda + DWR \quad (7)$$
$$dYr = Ls\lambda + 2DLR \quad (8)$$

DOWN

$$Xr = Xs\lambda - (Ws/2)\lambda - DWR/2 + OFFSET \quad (9)$$
$$Yr = Ys\lambda - Ls\lambda - DLR \quad (10)$$
$$dXr = Ws\lambda + DWR \quad (11)$$
$$dYr = Ls\lambda + 2DLR \quad (12)$$

RIGHT

$$Xr = Xs\lambda - DLR \quad (13)$$
$$Yr = Ys\lambda - (Ws/2)\lambda - DWR/2 + OFFSET \quad (14)$$
$$dXr = Ls\lambda + 2DLR \quad (15)$$
$$dYr = Ws\lambda + DWR \quad (16)$$

LEFT

$$Xr = Xs\lambda - Ls\lambda - DLR \quad (17)$$
$$Yr = Ys\lambda - (Ws/2)\lambda - DWR/2 + OFFSET \quad (18)$$
$$dXr = Ls\lambda + 2DLR \quad (19)$$
$$dYr = Ws\lambda + DWR \quad (20)$$

LCW

UP

$$Xr = Xs\lambda - (Ws/2)\lambda - WR - OFFSET \quad (21)$$
$$Yr = Ys\lambda - DLR \quad (22)$$
$$dXr = WR \quad (23)$$
$$dYr = Ls\lambda + 2DLR \quad (24)$$

DOWN

$$Xr = Xs\lambda + (Ws/2)\lambda + OFFSET \quad (25)$$
$$Yr = Ys\lambda - Ls\lambda - DLR \quad (26)$$
$$dXr = WR \quad (27)$$
$$dYr = Ls\lambda + 2DLR \quad (28)$$

RIGHT

$$Xr = Xs\lambda - DLR \quad (29)$$
$$Yr = Ys\lambda - (Ws/2)\lambda + OFFSET \quad (30)$$
$$dXr = Ls\lambda + 2DLR \quad (31)$$
$$dYr = WR \quad (32)$$

LEFT

$$Xr = Xs\lambda - Ls\lambda - DLR \quad (33)$$
$$Yr = Ys\lambda - (Ws/2)\lambda - WR - OFFSET \quad (34)$$
$$dXr = Ls\lambda + 2DLR \quad (35)$$
$$dYr = WR \quad (36)$$

RCW

UP

$$Xr = Xs\lambda - (Ws/2)\lambda + OFFSET \quad (37)$$
$$Yr = Ys\lambda - DLR \quad (38)$$
$$dXr = WR \quad (39)$$
$$dYr = Ls\lambda + 2DLR \quad (40)$$

DOWN

$$Xr = Xs\lambda - (Ws/2)\lambda - WR - OFFSET \quad (41)$$
$$Yr = Ys\lambda - Ls\lambda - DLR \quad (42)$$
$$dXr = WR \quad (43)$$
$$dYr = Ls\lambda + 2DLR \quad (44)$$

RIGHT

$$Xr = Xs\lambda - DLR \quad (45)$$
$$Yr = Ys\lambda - (Ws/2)\lambda - WR - OFFSET \quad (46)$$
$$dXr = Ls\lambda + 2DLR \quad (47)$$
$$dYr = WR \quad (48)$$

LEFT

$$Xr = Xs\lambda - Ls\lambda - DLR \quad (49)$$
$$Yr = Ys\lambda - (Ws/2)\lambda + OFFSET \quad (50)$$
$$dXr = Ls\lambda + 2DLR \quad (51)$$
$$dYr = WR \quad (52)$$

5 まとめ

参考文献

[1] 清水尚彦, コンピュータ設計の基礎知識-ハードウェア・アーキテクチャ・コンパイラの設計と実装-, 共立出版,2003.

[2] IP ARCH,Inc. http://www.ip-arch.jp

表 1: Alliance 環境変数

環境変数	内容
MBK_WORK_LIB	出力するディレクトリ
MBK_IN_LO MBK_OUT_LO	vst 拡張子 (cougar の時は sp に変更)
MBK_TARGET_LIB	使用するセルライブラリの vst ファイルがあるディレクトリ
MBK_CATA_LIB	使用するセルライブラリの ap ファイルがあるディレクトリ
MBK_IN_PH MBK_OUT_PH	ap 拡張子
RDS_TECHNO_NAME	テクノロジーファイル
GRAAL_TECHNO_NAME	GRAAL 設定ファイル
DREAL_TECHNO_NAME	DREAL 設定ファイル