Vol.2017-ARC-224 No.13
Vol.2017-SLDM-178 No.13
2017/1/24

北九州学術研究都市CMOSプロセスに対応した高位合成デジタルLSI開発フローの構築

古川 拓実[1]　河村 まりや[2]　冨山 修平[2]　清水 尚彦[2]

概要：北九州学術研究都市では手作業でチップを試作する手法により、安価なCMOSチップ製造環境を提供している。機材設備のグレードや手作業によるズレを考慮するため、プロセスルールが厳しくなる。このCMOSプロセスは商用EDAが対応しておらずデジタルLSIの開発手法が確立されていない。我々は、フリーのEDAツールとスケーラブルなセルライブラリによる高位合成デジタルLSI開発フローを構築し低コストの開発環境を提供する。本論文では、本デジタルLSI開発フローと北九州学術研究都市CMOSプロセスへのツールの対応について述べる。

キーワード：デジタルLSI設計, EDA, スケーラブルセルライブラリ

The Method of High-Level Digital LSI Design with Free EDA Tools for FAIS 2um Process

Takumi Furukawa[1]　Mariya Kawamura[2]　Syuhei Tomiyama[2]　Naohiko Shimizu[2]

1. はじめに

デジタルLSIは大規模化し、効率の良い開発のためにEDA(Electronic Design Automation)ツールは必要不可欠なものとなっている。この問題に対し、我々はこれまでに高位合成言語NSLとフリーのEDAツールセットAlliance[1]ならびにスケーラブルなセルライブラリを用いたLSI開発手法を確立してきた。スケーラブルなセルライブラリを用いることで一つの仮想レイアウトから複数のプロセスに対応した実レイアウトを作ることができる。スケーラブルなセルライブラリに広く用いられているλ-based rule[2]はパラメータλに依存するため線形的なスケールしかできない。我々はこの問題に対しAllianceのシンボリックルールを用いて、Onsemi1.2um, Rohm0.35um, Rohm0.18umのプロセスのチップを試作してきた[3][4]。試作してきたチップを**図 1**に示す。

今回我々は、北九州産業学術推進機構（FAIS）共同研究開発センター[5]で行っているCMOS 2umプロセス(以降、FAIS2umプロセス)を対象にデジタルLSIチップの開発フローを構築し、チップ試作と検証を行った。[6] FAIS2umプロセスではデジタルLSI開発手法が確立されておらず、利用者は自ら設計ツールを用意しなければならない。我々はFAIS2umプロセスに対応したデジタルLSI開発フローを構築することによって利用者に対し、新たなデジタルLSI開発ツールセットを提供する。FAIS2umプロセスへの対応の問題点として、チップ面積が最小となるようにシンボリックルールのパラメータを作成した場合、ボンディングパッド等のチップ外部仕様に合わせることが出来ず最終的な調整を手作業で行っていることがあげられる。[6] また、レイアウト可能領域に対してI/Oセルが小さく、領域を十分に使用できていなかった。

この問題に対し、チップ外部仕様も満たすことが可能なパラメータへの変更やI/Oセルの再設計を行うことで要求を満たすEDAツールを開発した。これにより本手法の利用者は論理設計のみでLSIチップを開発することが可能となった。

本論文では、高位合成言語NSLとフリーのEDAツールAllianceを用いた高位合成デジタルLSI設計フローとFAIS2umプロセスへのツール対応について述べる。

[1] 東海大学　情報通信学研究科　情報通信学専攻
[2] 東海大学　情報通信学部　組込みソフトウェア工学科

Vol.2017-ARC-224 No.13
Vol.2017-SLDM-178 No.13
2017/1/24

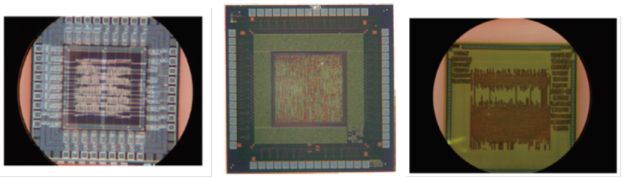

図 1　過去に試作を行ったチップの顕微鏡拡大写真
左:Onsemi 1.2um, 中央:Rohm 0.35um, 右:Rohm 0.18um

2. 設計フロー

論理回路設計からレイアウトの生成、検証のフローを図 2 に示す。大きく分けて高位合成言語 NSL[7] による設計、フリーの EDA ツールセット Alliance を用いたレイアウトの生成、各種検証がある。Alliance はフランスの UPMC が開発したオープンソースの EDA ツールセットである。このツールセットは回路エントリに VHDL の小さなサブセットを用いる。NSL コンパイラである NSL Core は、本サブセットに合わせた VHDL を生成可能である。

まず、NSL にて論理回路設計を行い高位合成して VHDL を生成する。NSL は同期設計で抽象度の高い記述が可能である。本ツールを用いるデジタル LSI チップ開発者 (以下、利用者) はインターフェースと動作を記述し、下位記述に触れずに設計ができるため短時間で大規模な LSI 設計を行うことができる。高位合成時に回路の最適化によって回路規模の削減をする。

Alliance は NSL が生成した VHDL を論理合成してセルライブラリに対応したネットリストを抽出する。この時点で論理圧縮して更に回路の最適化をする。抽出したネットリストを入力として配置配線を行い、仮想レイアウトを生成する。仮想レイアウトをプロセスルールごとに用意した変換ルールである Technology Mapping File を用いて実レイアウトへ変換する。この実レイアウトはストリーム形式 (GDSII) で出力される。

UPMC が開発した検証ツールには、配置配線した仮想レイアウトがネットリストと一致しているか確認する LVS ツール、レイアウトと VHDL の形式検証ツール、デザインルールチェック (DRC) ツールがある。形式検証ではレイアウトから SPICE モデルを抽出しバックアノテーションすることで VHDL を生成する。NSL が生成した元となる VHDL とバックアノテーションした VHDL を形式検証する。DRC はプロセスごとのデザインルールをまとめたファイル (DRC Deck) を読み込み、レイアウトのデザインルール違反を確認する。これらの検証ツールによって利用者は本手法で作成したチップレイアウトにエラーが無いことを確認することが可能である。

3. 実レイアウトへの変換ルール

スケーラブルなセルにおいて λ-based rule が広く用いら

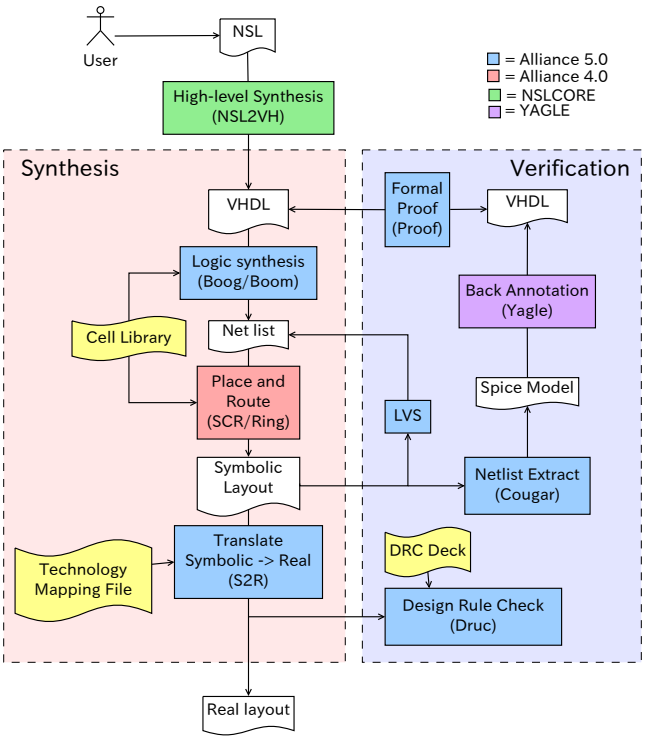

図 2　デジタル LSI 設計フロー

れている。λ-based rule は Stick Diagram のグリッド幅を 1λ としてセルを設計し、λ を変更することで様々なプロセスに対応することが可能な手法である。パラメータ λ 一つでサイズを変更し各オブジェクトの比率が一定の線形的な変換になる特徴がある。このため、細かいルールへの対応ができない。一方、Alliance で用いるシンボリックルールでは、λ-based rule と同様に Stick Diagram のグリッドの単位を λ とするが、Alliance のシンボリックルールは実レイアウトへ変換時に変換テーブルを用いてセグメント長、幅、オフセットを修正することで λ-based rule より細かい調整を行うことができる。

仮想レイアウトはレイヤではなく λ を単位とする長さ Ls、幅 Ws のセグメントで表される。Technology Mapping File 内には各セグメントが実レイアウトに変換された際に配置されるレイヤ情報と各レイヤのオフセット値 DLR, DWR を記述する。変換後のレイヤ長 Lr は式 (1)、レイヤ幅 Wr は式 (2) で決定される。

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

$$Wr = Ws * \lambda + DWR \quad (2)$$

仮想レイアウトの各セグメントは λ を単位とするグリッド上に配置される。レイアウト中の各オブジェクトはセグメントの種類、グリッド座標、長さ、幅の情報を持つ。配置配線終了後まではすべて仮想レイアウトで処理を行い、図 2 中央下の仮想レイアウトから実レイアウト (GDS) への変換時に Technology Mapping File を適用する。表 1 に Technology Mapping File 内の変換ルール

Vol.2017-ARC-224 No.13
Vol.2017-SLDM-178 No.13
2017/1/24

表 1 Technology Mapping File に記述されている変換ルールの一部 (λ=3um)
1 つのセグメントに複数のレイヤを割り当てることができる。

Segment	Layer	DLR(um)	DWR(um)
NDIFF	NSELECT	4	2
	VTH-N	4	2
	ACTIVE	2	0
NTRANS	POLY	0	-1.0
	ACTIVE	-2	14
	NSELECT	0	18
	VTH-N	0	18

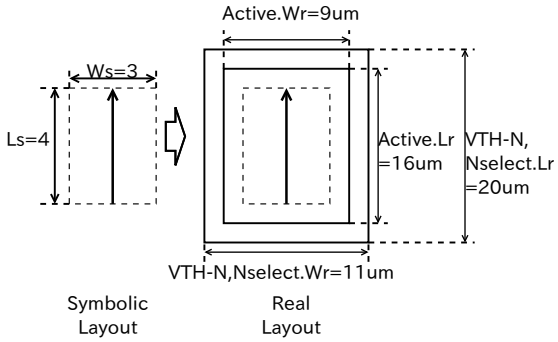

図 3 N-Diffusion 変換例
λ=3um の場合、図 1 を用いると Active レイヤは $Wr = 3*3+0 = 9um$, $Lr = 3*4+2*2 = 16um$ となる。

例を示す。Alliance の変換ルールは、1 つの仮想セグメントから複数の実レイヤへ変換することができる。このルールを用いると**図 3** のように Ls=4, Ws=3 の NDIFF セグメントは Lr=4*3+2*4=20um, Wr=3*3+2=11um の NSELECT レイヤ、Lr=4*3+2*4=20um, Wr=3*3+2=11um の VTH-N レイヤ、Lr=4*3+2*2=16um, Wr=3*3+0=9um の ACTIVE レイヤへ変換される。

図 4 に λ-based rule と本手法で作成した Diffusion と Active レイヤの比較を示す。プロセスルールとして Active レイヤは Nselect レイヤ, Vth 調整用レイヤ Nth-N の 2um 内側になければならないものとする。この時 λ-based rule では Active レイヤ、Nselect レイヤ、Vth-N レイヤすべてを別のオブジェクトとして座標データを持つ。Alliance では、オブジェクトは仮想の Diffusion セグメントのみを持ち、実レイヤへの変換時に Active レイヤ、Nselect レイヤ、Vth-N レイヤを Technology Mapping File に則った適切なサイズで変換する。例では、表 1 に示す変換ルールを用いて NDIFF セグメントを各レイヤへ変換している。

プロセスルールによっては複数のレイヤを重ねて適切な配置を行わなければならないため、λ-based rule ではレイヤ作成時の人的エラーが発生しやすい。本手法はレイヤをまとめてセグメントと扱うことに寄ってエラーを最小限に抑えることができる。

図 5 にトランジスタの変換例を示す。トランジスタも同様に 1 つの仮想セグメントとして扱う。例では Polysilicon, Active, NSelect, Vth-N を 1 つにまとめてトランジスタを構成している。オフセット DLR、DWR はマイナス値も設定することができ、この例では Polysilicon のルールが幅 2um に対して 1λ が 3um なので DWR に-1.0 を指定している。

Object List (λ-based rule)

Layer	XY1	XY2
NSELECT	(0,0)	(6,10)
VTH-N	(0,0)	(6,10)
ACTIVE	(1,1)	(5,9)

Object List (Symbolic rule)

Segment	XY1	XY2	Width
NDIFF	(3,2)	(3,6)	3

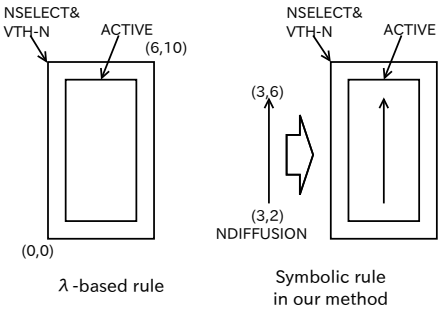

図 4 Diffusion と Active レイヤを作成する際の λ-based rule と本手法の比較
λ-based rule では必要なレイヤをそれぞれ作成するが、本手法では Diffusion というセグメントにまとめて 1 つのオブジェクトにできる。

Symbolic Object

Segment	Ls(λ)	Ws(λ)
NTRANS	6	1

Real Object

Layer	Lr (um)	Wr (um)
POLY	18	2
ACTIVE	14	17
NSELECT	18	21
VTH-N	18	21

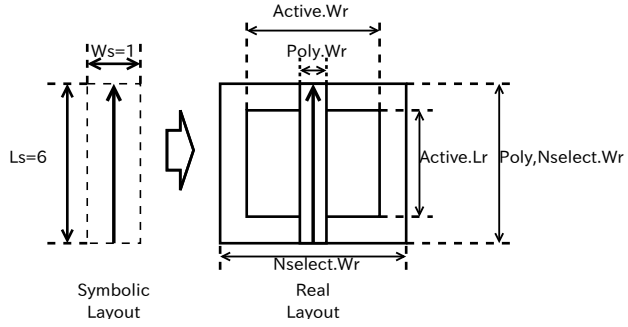

図 5 トランジスタ変換例
仮想レイアウトではトランジスタはオブジェクト NTRANS のみで表される。表 1 と式 (1), 式 (2) に基づいて仮想オブジェクト NTRANS から実レイアウト {POLY,ACTIVE,NSELECT,VTH-N} へ変換する。
ACTIVE の例では Lr は $6*3+2*(-2) = 14um$, Wr は $1*3+14 = 17um$ となる

Vol.2017-ARC-224 No.13
Vol.2017-SLDM-178 No.13
2017/1/24

Symbolic Rule

Segment	Layer	Witdh(um)
POLY_CONT	METAL1	6.0
	CONTACT	2.0
	POLY	7.0

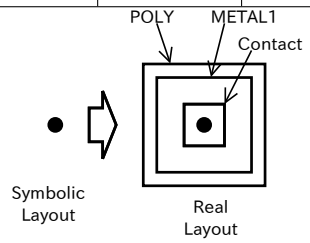

図 6 コンタクト・VIA 変換例
コンタクト、VIA は一辺の幅のみを持ち、仮想オブジェクトを中心とした正方形のレイヤへ変換される。

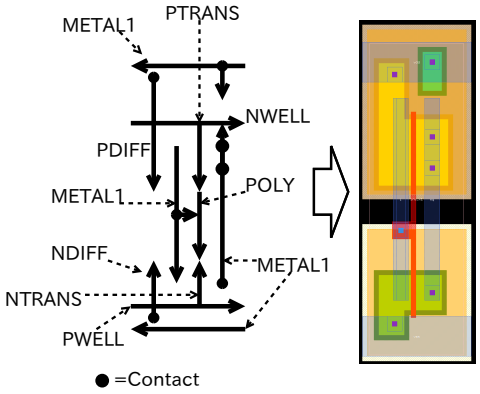

図 7 インバータ変換例

VIA やコンタクトは仮想レイアウト上に VIA・コンタクトの中心点で表される。実レイアウトでは**図 6** のようにオフセット値を一辺とした正方形のレイヤへ変換される。

Alliance で扱うスケーラブルなセルライブラリはこのシンボリックルールを用いて作成する。**図 7** にインバータセルの仮想レイアウトから実レイアウトへの変換例を示す。仮想レイアウトでセルライブラリが揃っているため、プロセスルールの変更時にはセルライブラリを大きく変更することなく、プロセスに合った Technology Mapping File を読み込むことで対応できる。

4. FAIS2um プロセスへの適用

新たなプロセスへ本手法を対応させるには、変換ルールである Technology Mapping File とレイアウト検証のための DRC Deck を作成する必要がある。今回は FAIS2um プロセスルールを満たすようにこれらのファイルを記述した。チップサイズを決める上で初めにベースとなるグリッドサイズ λ を決める。FAIS2um は商用プロセスと大きく異なる点として、手作業でのチップ試作や機材設備の都合から METAL-METAL 間と VIA-VIA 間が非常に大きく必要である。Alliance で用いる配線ツール SCR(図 2 中部) は、各セル間を接続する METAL を 5λ 間隔で配線する。この時

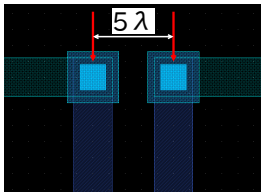

図 8 配線ツール SCR の配線した VIA 拡大図
青:METAL1, 青緑:METAL2, 水色:VIA
SCR は配線を 5λ 間隔で配線するが、VIA が置かれた際に METAL-METAL 間を 4um 離すには VIA-VIA 間 (中心-中心) が 12um 以上離れていなければならない。

表 2 λ=2.4um とした場合の Technology Mapping File の一部
ゲート幅 2um なので Polysilicon は DWR を-0.4 にすることで 2.4+(-0.4)=2.0um となる。METAL も同様に Ws=2λ 時に幅 6um を満たすように 2.4*2+(1.2)=6.0um としている。

Segment	Layer	DLR(um)	DWR(um)
POLY	POLY	1.0	-0.4
ALU1	METAL1	3.0	1.2

に VIA を配置すると**図 8** のようになる。FAIS2um プロセスでは VIA-VIA 間が 12um 必要となるため、5λ 間隔の配線でエラーを起こさない λ の最小値は 2.4um である。一番のボトルネックである VIA-VIA 間に合わせたグリッドサイズ λ=2.4um として Technology Mapping File を作成した。作成した Technology Mapping File の一部を**表 2** に示す。

従来対応してきたプロセスとは大きく異なる FAIS2um プロセスでは Technology Mapping File では対応できない以下のような問題が発生した。

- POLY 上に VIA を置くことが出来ないため、すでにセルライブラリ中にある POLY 上の VIA が違反してしまう。
- チップサイズに対して既存の I/O セルが大きすぎるため、レイアウト領域が非常に小さくなってしまう。

これらの問題に対し、セルライブラリの新規開発を行い対応した。

4.1 セルライブラリの開発

セルライブラリは大きく分けてロジックごとのスタンダードセルと電源リングを含む I/O パッドセルがある。Alliance のシンボリックルールではセグメントはすべてグリッド上に存在し、Technology Mapping File では座標の変更は行われない。スタンダードセルには、POLY 上に VIA が配置されているものがあり、変換ルールのオフセット調整では違反を防げない。**図 9** のようにセル内の POLY と VIA の位置を調整することでデザインルール違反が起こらないように変更した。

チップサイズが小さいため既存の I/O パッドセルではレイアウト可能領域を圧迫してしまう。最大限レイアウト

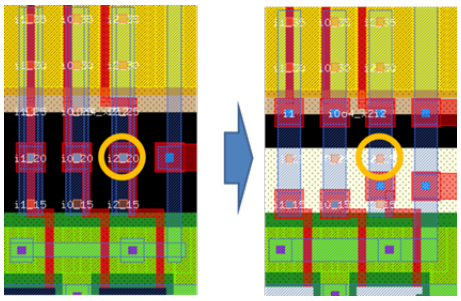

図 9　OR セル内の POLY 位置修正

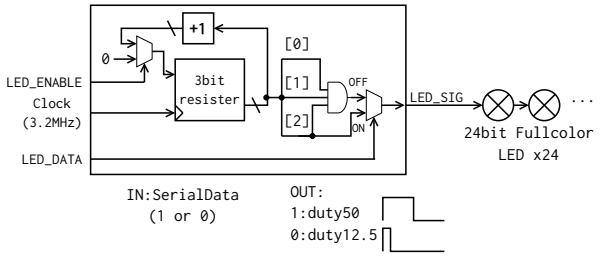

図 10　マイコン LED 制御信号用デジタル-パルス変換回路

領域をとることができるように I/O パッドセルを設計し、この問題に対処した。作成したセルは、Input・Output・VDD・VSS・Clock の 5 つである。ボンディングパッドを含む外部仕様は今回利用したシャトル指定のレイアウトがあり、I/O パッドセルはボンディングパッドに接続する部分までを設計した。

5. チップ試作

パラメータを λ=2.4um に調整した本手法を用いて試作チップを作成した。ターゲット回路としてマイコン LED 制御用デジタル-パルス変換回路を設計し、本手法の試作回路とした。**図 10** にターゲット回路のブロック図を示す。デジタル信号の入力に応じて 1 ならば duty50、0 ならば duty12.5 のパルスを生成する回路である。本手法を用いて作成したレイアウトを**図 11** に示す。作成したレイアウトは一般財団法人ファジィシステム研究所が主催する「第 7 回ユニーク・自作チップ・コンテスト in ひびきの」へ応募し、筆者らが手作業でのチップ試作を行った。このチップを評価したところ、設計した全セルの動作と、デジタルーパルス変換回路の正常動作を確認できた。

図 12 に試作チップの拡大写真を示す。手動で外部回路と組み合わせたため、ボンディングパッド部分と I/O パッドセルの間に無駄なスペースができていることがわかる。

6. チップ外部仕様への対応

第 4 章ではチップ面積が最小となるようにパラメータ λ=2.4um とした。しかし、ボンディングパッドやチップ特性測定用回路は FAIS 指定のレイアウトがあるため本手法で作成したレイアウトを手動で組み合わせた。本研究の手法は利用者が論理回路を記述後はテープアウトまで自動

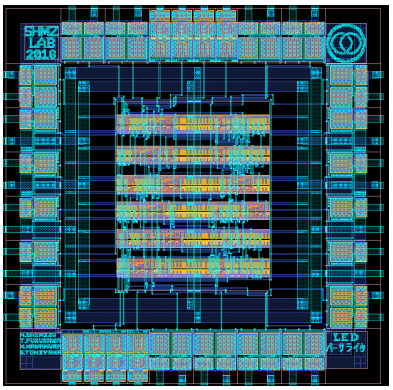

図 11　本手法を用いて生成した試作チップマスクデータ (λ=2.4um)

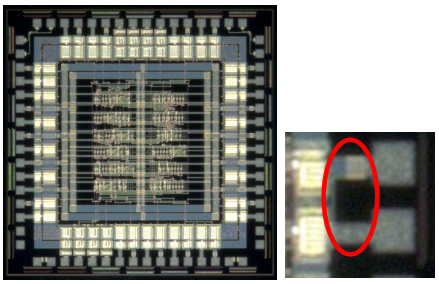

図 12　試作チップ顕微鏡拡大写真
右:パッド接続部分拡大写真、赤丸部分の領域が空いてしまっていることがわかる。

表 3　λ=2.5um とした場合の Technology Mapping File の一部
2.5um に変更したことにより、表 2 と比較して DWR へ影響が出ている。
Polysilicon は DWR を-0.5 にすることで 2.5+(-0.5)=2.0um となる。METAL も同様に Ws=2λ 時に幅 6um を満たすように 2.5*2+(1.0)=6.0um としている。

Segment	Layer	DLR(um)	DWR(um)
POLY	POLY	1.0	-0.5
ALU1	METAL1	3.0	1.0

で行うことを目標としているため、我々は新たにツールの外部仕様への対応を行った。

FAIS2um プロセスではボンディングパッドは 100um 四方の METAL2 を 160um 間隔で配置しなければならない。しかし、λ=2.4um では割り切れないためパッドを均等に配置することが出来ない。我々は、λ を 2.5um に変更しボンディングパッドを含む外部仕様回路を作成した。λ はレイアウトサイズへ直接影響が出るため 2.4um から 2.5um に変更することで回路面積が 8.5%増加する。

λ を変更したことにより 2.5um に対応する Technology Mapping File を新たに作成した。**表 3** に変更したルールの一部を示す。

パラメータ λ の変更の影響を大きく受ける I/O パッドセルも同様に新規作成した。その際に第 5 章で述べたボンディングパッドと I/O パッドセル間の無駄なスペースを

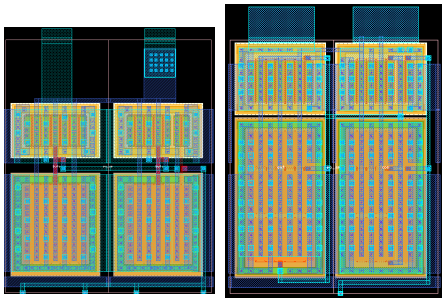

図 13 Output パッドセルの比較
左:λ=2.4um, 右:λ=2.5um
ボンディングパッド位置までの空いている領域を利用してドライブ能力を向上させている。

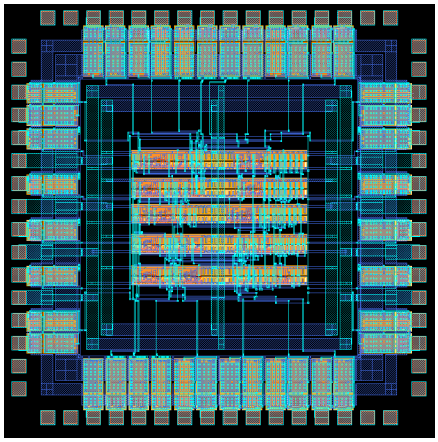

図 14 λ=2.5um に変更した本手法を用いて生成したチップマスクデータ
図 11 では生成されなかったボンディングパッド部分が生成されている。

解消している。**図 13** に新規作成したパッドと従来のパッドの比較を示す。領域を十全に使うことで電流ドライブ能力は旧パッドでは約 70mA に対して新たなパッドでは約 100mA となり 40%ほど向上している。

配置配線したコア回路に新規作成した I/O パッドセル、ボンディングパッドの外部仕様回路を自動合成することで本手法は論理回路設計からレイアウトまでのフローをすべてサポートすることが可能となった。

7. 評価

λ を 2.5um へ変更したことによる影響の評価のため、改良したツールを用いて図 10 の回路からレイアウトを生成した。**図 14** に生成した回路を示す。図 11 ではできていなかったボンディングパッド部分が正確な位置に生成されている。

2.5um に変更した影響の比較のため、2.4um, 2.5um それぞれ 10 回配置配線行い、コア回路面積を計測した。**表 4** に変更前と後の面積を示す。λ を 2.5um にしたことで面積は 8.2%の増加となり、期待値とほぼ同等の結果となった。

表 4 コア回路の面積比較
配置配線をそれぞれ 10 回行い、平均値を比較する。λ=2.4um と比較して 2.5um は 8.2%増えている。

λ[um]	Area of Core circuit [um^2]
2.4	1.496
2.5	1.619

しかし、図 14 を見てわかるように、2 層メタルではコアの周りに大きな配線領域を必要となるし、配置配線ツールのルーティングによっては 2.5um より 2.4um の面積が大きくなる場合があった。配置配線による面積変動より λ 変更による面積増大の影響が少ないことから、レイアウトサイズ増大の問題は無いと言える。

8. まとめ

我々は高位合成言語 NSL とフリーの EDA ツールセット Alliance を用いて FAIS2um プロセスのデジタル LSI 開発フローの構築を行った。従来のツールセットでは FAIS2um プロセスの外部仕様回路へ対応しておらず、手作業での修正を必要としていた。この問題に対して、仮想レイアウトのグリッドサイズを変更し、それに伴うセルライブラリの修正と新規開発を行った。その結果、グリッドサイズ変更による回路面積への影響を最小限に抑えつつ外部仕様回路を含むすべてのプロセス要求仕様を満たすことが可能となった。これにより本手法を用いることで FAIS2um プロセスにおいてデジタル LSI 開発フローを確立した。

謝辞 本研究の LSI チップは, 北九州学術研究都市の共同研究開発センターで開催された「第 7 回ユニーク・自作チップ・コンテスト in ひびきの」において試作したものである。

参考文献

[1] http://www-asim.lip6.fr/recherche/alliance/
[2] c.Mead and L.conway, Introduction to VLSI Systems, Addison-Wesley, 1980
[3] 樋口拓哉, 大金淳一郎, 清水尚彦, ”スケーラブルなセルライブラリを用いた LSI 試作のデザインフロー設計試行 (ON-SEMI 1.2um)”, 東海大学紀要情報通信学部 vol.3, No.1, pp.25-30, 2010
[4] 樋口 拓哉, 細川 達也, 今井 紘士, 清水 尚彦, ”λ ルールスケーラブルセルライブラリの 0.18u m プロセスへの実装試行と TEG チップ開発”, 東海大学紀要情報通信学部 Vol.4, No1, pp.19-25, 2011
[5] http://www.ksrp.or.jp/shisetsu/semicon1.html
[6] 古川拓実, 河村まりや, 冨山修平, 清水尚彦, ”フリー EDA ツールによる高位合成デジタル LSI 開発手法の研究”, 電子回路研究会, ECT-16-103, IEEJ, 2016
[7] http://www.overtone.co.jp/