社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2003－ＳＬＤＭ－112 (13)
2003／11／28

2線2相論理回路方式ASDDL/ASD-CMOSの論理合成手法

田中 義則† 森本 薫夫† 永田 真†† 瀧 和男†††

† 神戸大学大学院自然科学研究科
† 神戸大学工学部情報知能工学科
††† エイ・アイ・エル株式会社

E-mail: †{y.tanaka,morimoto}@cs26.scitec.kobe-u.ac.jp, ††nagata@cs.kobe-u.ac.jp, †††taki@ailabo.co.jp

あらまし 2線2相論理回路方式を用いて，信号の立上り遷移と立下がり遷移に要する時間を意図的に非対称とすることで高速化を図った新たな回路方式，ASDDL/ASD-CMOSの論理合成手法を提案し評価した．ASDDL/ASD-CMOSは独特の回路方式であるため，これまでは全てをフルカスタムで設計してきた．しかし独自に提案したセルライブラリと変換ツールを用いることにより，CMOS用の論理合成環境をそのまま適用した論理合成を可能にした．0.18μmプロセスにて16ビット乗算器を論理合成した結果，論理合成回路の遅延値はフルカスタム回路の約126%，CMOS回路の45%であり，フルカスタムで設計した回路に近い性能の回路を論理合成できることを確認した．

キーワード 2線2相論理回路, ASDDL, ASD-CMOS, 論理合成

A Logical Synthesis Methodology for Two-phase and Differential Logic Technology ASDDL/ASD-CMOS

Yoshinori TANAKA†, Masao MORIMOTO†, Makoto NAGATA††, and Kazuo TAKI†††

† Graduate School of Science and Technology, Kobe University
† Department of Computer and Systems Engineering, Kobe University
††† AIL Co.,Ltd.

E-mail: †{y.tanaka,morimoto}@cs26.scitec.kobe-u.ac.jp, ††nagata@cs.kobe-u.ac.jp, †††taki@ailabo.co.jp

Abstract We propose a logical synthesis methodology for a high speed logic technology ASDDL/ASD-CMOS, which are two-phase and differential logic circuit. Because of its originality, it was unable to synthesize a ASDDL/ASD-CMOS circuit as yet. But we have been able to synthesize a its circuit by proposing a original cell library and transforming tool to apply a logical synthesis environment for CMOS circuit to ASDDL/ASD-CMOS circuit. The delay time of synthesized ASDDL circuit was 126% of full customed.

Key words two-phase and differential logic circuit, ASDDL, ASD-CMOS, logical synthesis

1. はじめに

近年の微細化加工技術の著しい進歩により，LSIはその性能を飛躍的に発展させている．それに伴い膨大なデータをより短時間で処理することを求められるようになっており，LSIへの高速処理の要求はとどまることがない．

本稿では高速な回路方式としてASDDL（Asymmetric Slope Differential Dynamic logic）/ASD-CMOS（Asymmetric Slope Differential CMOS）回路方式[1][2]を扱う．これらは2線2相論理回路方式というCMOSとは全く異なる回路方式であるため，回路をHDLで記述して論理合成する方法では設計できない．そのため回路設計は常に回路図を直接入力してネットリストを作成し，自分で最適化するというフルカスタムの設計であった．このような人間の手による設計であるため多くの時間がかかってしまう上，誤った回路を設計してしまう危険性があるという問題があった．

そこで，これまでは自動設計されていなかったASDDL/ASD-CMOS回路を論理合成によって自動設計する手法を提案する．自動設計するに当たっては，ASDDL/ASD-CMOS用の論理合成ツールそのものを作成するのではなく，CMOS用の論理合成環境をうまく適用して論理合成結果を得るというアプローチをとる．ASDDLはダイナミック・ロジックという特徴も持っているため，本来ならスタティック回路であるCMOS用ではなくダイナミック・ロジック用の論理合成ツールを使用するの

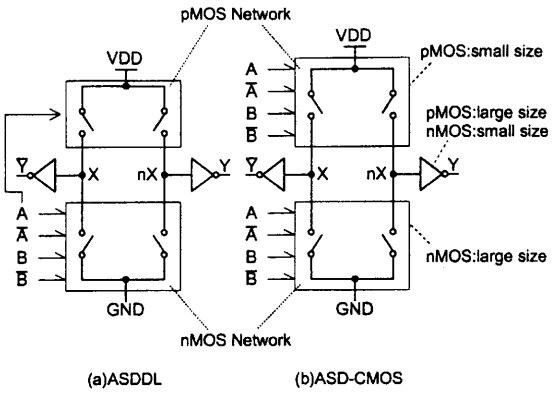


図 1 ASDDL/ASD-CMOS basic circuit structure

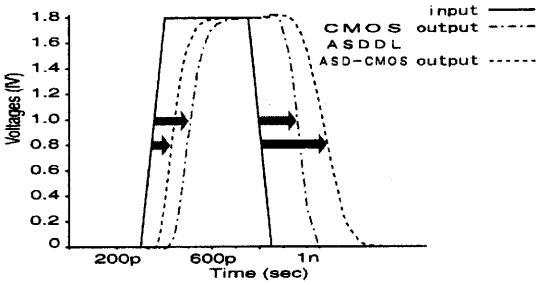


図 2 Delay time of ASDDL/ASD-CMOS and CMOS

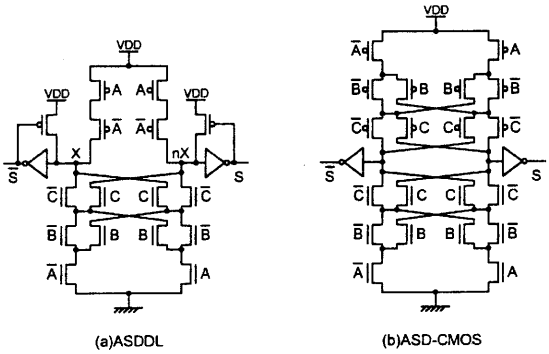


図 3 Circuit of FullAdder S output

が妥当である．しかしダイナミック・ロジックはレーシングや雑音等の問題があり，設計や自動化が非常に困難であるため，そのようなツールはほとんど存在しないといってよい状況にある [3]．また CMOS 用の論理合成環境を適用する事は，CMOS 用のハイレベルな論理合成の環境を ASDDL/ASD-CMOS にもそのまま適用できるという点でも有用である．

以下，2. では ASDDL/ASD-CMOS 回路方式の特徴，回路構成と動作について説明し，3. で論理合成の実現法を提案する．4. でライブラリと変換ツールの説明をした後，5. では実際に ASDDL 回路を論理合成して評価し，最後に 6. でまとめる．

2. ASDDL/ASD-CMOS 回路方式

2.1 特　　徴

ASDDL/ASD-CMOS の概念図を図 1 に示す．以下に特徴を述べる．

- 2 線 2 相論理回路方式

2 線とは信号が正論理，負論理のペア $\{x,\overline{x}\}$ で表されることを意味する．常に両論理を有しているため，複雑な論理式も少ないゲート段数で構成可能であり，高速化に適している．2 相とは信号が有効値（effective value）と休止値（reset value）の 2 状態を有することを意味する．有効値は論理 0（{0,1}）と論理 1（{1,0}），休止値は（{0,0}）である．初期値を休止値とし，有効値を入力に与え順次後段に伝搬させることで動作する．休止値→有効値→休止値にて 1 動作サイクルを形成する．

- 非対称な信号遷移

信号の立ち上がり遷移（有効値への遷移）時間と，立ち下がり遷移（休止値への遷移）時間を意図的に非対象とし，立ち上がり遷移を高速化するように回路が設計されている．これは主にトランジスタサイジングで実現し（図 1），これにより有効値の伝搬を高速化している（図 2）．

2.2 回路構成と動作

ASDDL/ASD-CMOS の FullAdder の S 出力の 回路図を図 3 に示す．同図に示すように，nMOS Network の構成は ASDDL，ASD-CMOS 共に共通であり，pMOS Network のみが異なる．ASDDL の pMOS Network は pMOS トランジスタの直列接続で構成され，ダイナミック回路におけるプリチャージ回路として機能する．ASDDL は「クロックレス・ダイナミック回路」である．ASD-CMOS は pMOS Network に nMOS Network と双対の構造を持つ，スタティック回路である．両者に共通の nMOS Network は，全ての入力が有効値になった時に図 1 の内部ノード X，nX のいずれかを排他的にグランドに接続するように動作する．この様なネットワークは BDD（Binary Decision Diagram）を用いて容易に設計できる．また ASDDL ではダイナミック動作時に出力"0"を安定して出力するために，pull-up pMOS が必要になる．

ASDDL は以下のように動作する．入力が休止値の時，pMOS Network は導通し，nMOS Network は切断するので内部ノード X，nX がチャージされる．よって出力は休止値となる．その状態から入力が有効値へ変化すると pMOS Network が電源と内部ノードを切断させ，回路がダイナミック動作に入る．一方 nMOS Network は内部ノードのどちらか一方をグランドと導通させて蓄えられていた電荷を引き抜く．これによって有効値が出力される．ASD-CMOS の場合は全入力が有効値になると pMOS Network は内部ノードのどちらか一方を切断，他方は電源と導通させたままという状態になり，有効値を出力する．よって，ASD-CMOS は動作の安定性において ASDDL に勝る．一方 pMOS Network が小さくなる ASDDL は，回路の静電容量が低いことから高速性が期待され，また回路規模の点でも有利である．

3. 論理合成の実現

3.1 2線2相回路方式の合成

ASDDL/ASD-CMOS 回路方式が CMOS 回路方式と主に異なる点は，2線回路であること，2相回路であること，の2点である．以下ではこのような特徴を持つ回路を論理合成するためのアプローチと，2線回路の論理合成フロー，2相回路の論理合成フローについて説明する．

3.2 アプローチ

2線2相論理回路方式という独特の回路方式用の論理合成ツールの実用化例はほとんど存在しない．しかし論理合成の環境を我々でゼロから構築することは多大なリソースを必要とし，現実的ではない．そこで既存の CMOS 用の論理合成環境をうまく適用するというアプローチをとる．CMOS 用の論理合成環境を適用するために，ダミーライブラリという CMOS 回路と ASDDL/ASD-CMOS 回路の中間にあたる回路のセルライブラリを作成する．これを用いて CMOS 用論理合成ツールで論理合成し，その結果に後処理を施すことで ASDDL/ASD-CMOS 回路が直接論理合成されたのと同等の結果を得る．

3.3 2線回路の論理合成フロー

まず論理合成用のセルライブラリとして，2線回路のセルライブラリそのものではなく，ダミーライブラリと呼ぶセルライブラリを作成する．ダミーライブラリは以下のような特徴を持つ．

- CMOS 用論理合成ツールのセルライブラリファイルで定義されている
- セルは通常の CMOS セルの出力のみが2線 ($\{Y, \overline{Y}\}$) に拡張された形である
- セルの遅延時間，面積等のパラメータは本来の ASDDL/ASD-CMOS セルの値である

次にこのダミーライブラリを用いて CMOS 用論理合成ツールで合成する．ダミーライブラリで論理合成を行うと各セルの出力端子は，Y のみが使用される，$\overline{Y}$ のみが使用される，もしくは Y と $\overline{Y}$ が共に使用されるという3通りのパターンができる．最後にこのパターンを利用して，ダミーライブラリの論理合成結果に変換ツールで2線回路へ拡張する後処理を施し，ASDDL/ASD-CMOS の論理合成結果を得る．

このように，先に論理合成後の後処理を見据えたダミーライブラリを作成し，それを用いて論理合成したネットリストに独自の変換ツールで後処理を施すことであたかも2線回路が直接論理合成できたのと同等の結果を得ることができる．ダミーライブラリや変換ツールに関しては 4. 内で詳しく説明する．

3.4 2相回路の論理合成フロー

ASDDL/ASD-CMOS は演算をする前に休止値を伝搬して回路を初期化する必要がある．2相回路の論理合成はこの機能を付加して合成することにあたる．休止値の伝搬に関しては，”有効値の伝搬は高速であるが，休止値の伝搬が遅いためサイクルタイムが長くなる”という問題がある．しかしその問題の解決策としてサイクルタイム短縮手法 [4] が提案されているため，この手法を適用した回路構成を合成する．

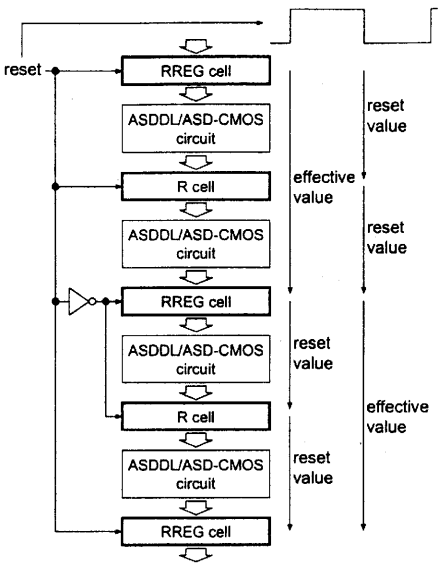


図 4 ASDDL/ASD-CMOS circuit applied the cycle time shortening architecture

サイクルタイム短縮手法を簡単に説明する（図4）．まず回路のクリティカルパスの中央段に位置するセルを休止値投入&レジスタ機能付きセル（RREG セル）に置き換える．更にその RREG 間の中央段のセルを休止値投入機能付きセル（R セル）に置き換える．これに図4のように適切な reset 信号を入力すると休止値伝搬を有効値伝搬の裏に隠すことができる．

この機能を実現するためには回路の全てのパス上に R セルや RREG セルを適切に配置する必要がある．以下にサイクルタイム短縮手法を適用した回路を論理合成するためのフローを示す．まず入力となる論理合成の対象回路を，マルチサイクルでパイプライン動作する回路に意図的に変更する．これは後処理で R セルや RREG セルを正しく挿入する際の目印として，レジスタを予め合成回路内に組み込んでおくためである．次にそれを論理合成する．論理合成ツールにレジスタを挿入させることにより，全てのパス上に漏らすこと無くレジスタを挿入することができる．最後にそれらを変換ツールで適切に R セル，RREG セルで置き換える．これにより全てのパスに対して正しく R セル，RREG セルを挿入することができ，サイクルタイム短縮手法の構成を組み込んだ論理合成が実現できる．

4. ライブラリと変換ツール

4.1 ASDDL/ASD-CMOS ライブラリ

ASDDL/ASD-CMOS は，常に正負両論理を有しており，どのような論理についても BDD から比較的簡単に回路を設計できるため，XOR の様な CMOS では複雑になる論理でも簡単に設計することができる．今回設計する論理セルは，3入力以下で実現可能なあらゆる論理のうち MCNC ベンチマーク回路を合成した際によく使われた論理を選択した．さらに使用頻度の高い論理に関しては駆動力の異なるセルを3種類用意した．各

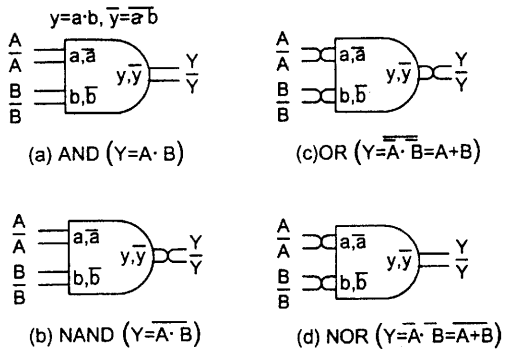


図 5 Variation of Pin connection

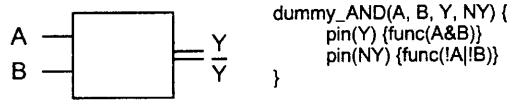


図 6 Example of dummy cell (Y = A・B)

セルは立ち上がり遅延時間と立ち下がり遅延時間の比が 1 対 2 になるように設計する．これはサイクルタイム短縮手法において RREG セルの間に配置する R セルの数を 1 つだけにするためである．

ASDDL/ASD-CMOS は 2 線の信号を逆に入れ替えることでもとの論理の否定論理を表現できる．これを利用することで 1 つのセルからたくさんの論理を作成することができる．例えば 2 入力で考えられるあらゆる論理は，2 種類のレイアウトで全て賄うことができる．図 5 は入力や出力の 2 線を入れ替える（Pin 名をつけ変える）ことで，1 つのセルから (a) AND, (b) NAND, (c) OR, (d) NOR 等の論理を得られることを示している．(a) AND の出力のみを入れ替えると AND の否定論理を表すことになり，(b) NAND 回路となる．さらに 2 つの入力のそれぞれを入れ替えることで $Y = \overline{\overline{A}\cdot\overline{B}} = A + B$ となり (c) OR 回路になる．また OR の出力を元に戻すと (d) NOR 回路になる．この特徴により，ASDDL/ASD-CMOS のセルライブラリはセルの種類が少なく，代わりに駆動力の種類が多いというライブラリ構成になる．

4.2 合成用ダミーライブラリ

ダミーライブラリは以下に示すことを基本として記述する．

- 図 6 に示すように出力のみが 2 線になるように記述する．入力は正論理のみであるが，負論理の入力も存在するものとして出力の論理を定義する．
- 遅延時間や面積といったセルライブラリに必要な情報は本来の ASDDL/ASD-CMOS セルの値をそのまま使用する．
- 実際のセルは立ち上がり遅延と立ち下がり遅延の値が大きく異なるが，ダミーセルライブラリではどちらにも立ち上がり遅延の値を使用する．
- ASDDL/ASD-CMOS は入力が全て有効値になった後にはじめて出力が有効値になるため，入力ピンから出力までの遅延時間はあらゆるピンに関して同じ値にする．

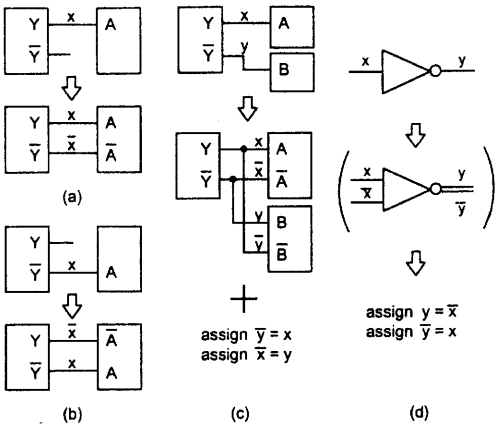


図 7 Conversion to differential logic

各セルが正負両論理を持っているので論理面からみれば論理合成にインバータは必要ない．また，たとえ使用されていたとしても 4.3 で述べるようにインバータは最終的には削除される．しかしインバータが駆動力確保等の目的で使用されていても削除されてしまうため，駆動力の計算に大きな違いを生じてしまう可能性がある．よってダミーライブラリの論理合成ではインバータの使用は望ましくない．インバータは市販の論理合成ツールではライブラリ内に欠かせないため，予め除外しておく事はできない．そこで論理合成ツールにできるだけインバータを使わせないようにするために，ダミーライブラリ中のインバータの面積や遅延を実際以上に大きく設定したり，最適化に使用しない属性を付加するなどの工夫が必要である．インバータの使用を抑える代わりに，駆動力確保等のために実際のセルライブラリ中のバッファの種類を多く用意したり，セルの駆動力の種類を増やしたりすることで対応する必要がある．

4.3 変換ツール

変換ツールはダミーライブラリで論理合成した Verilog ファイルを入力することで，それを 2 線 2 相に拡張した Verilog ファイルを出力する．

ダミーライブラリを論理合成するとセル間の配線には 3.3 で述べた 3 種類のパターンが存在する．変換ツールはそのパターンをもとに 2 線回路へと拡張する．変換方法を図 7 を用いて説明する．同図 (a) は正論理の出力 (Y) のみが次のセルの入力 (A) に x というネット名で接続されている形である．ツールは自動的に $\overline{A}$ という入力ピンと $\overline{x}$ というネットを作成して 2 線へ拡張し，それらを使用して $\overline{Y}$ と $\overline{A}$ を $\overline{x}$ で新たに接続する．同図 (b) は負論理の出力 ($\overline{Y}$) のみが次のセルの入力 (A) に x というネット名で接続されている形である．(a) と同様に，ツールは自動的に $\overline{A}$ という入力ピンと $\overline{x}$ というネットを作成して 2 線へ拡張し，それらを使用して Y と $\overline{A}$ を $\overline{x}$ で新たに接続する．同図 (c) は正論理の出力 (Y) と負論理の出力 ($\overline{Y}$) がそれぞれ次のセルの入力 (A, B) にそれぞれ x, y というネット名で接続されている形である．ツールは自動的に $\overline{A}$, $\overline{B}$ という入力ピンと $\overline{x}$, $\overline{y}$ というネットを作成して 2 線へ拡張し，それ

表 1　Logic cells of ASDDL cell library

name	logic	rank
AD2A	$A \cdot B$	x2 x3 x4
AD2B	$A \oplus B$	x2 x3 x4
AD3A	$A \cdot B \cdot C$	x2 x3 x4
AD3B	$A \oplus B \oplus C$	x2 x3
AD3C	$(A \oplus B) \cdot C$	x2 x3 x4
AD3D	$A + B \cdot C$	x2 x3 x4
AD3E	$A \cdot B \cdot C + \overline{A} \cdot \overline{B} \cdot \overline{C}$	x2
AD3F	$A \cdot B + \overline{A} \cdot C$	x2 x3 x4
AD3G	$A \cdot B + B \cdot C + C \cdot A$	x2
AD3H	$A \cdot B \cdot C + \overline{A} \cdot \overline{B} + \overline{A} \cdot \overline{C}$	x2

らを使用して $\overline{Y}$ と $\overline{A}$ を $\bar{x}$ で，Y と $\overline{B}$ を $\bar{y}$ で，新たに接続する．さらにこの時，新たに作成した $\bar{x}$，$\bar{y}$ というネットをそれぞれ y，x に接続することが必要になる．それを実現するために，assign $\bar{y} = x$，assign $\bar{x} = y$ という記述を追加する．同図 (d) は入力のネット名が，x，出力のネット名が y のインバータの変換を示している．ツールは自動的に $\bar{x}$，$\bar{y}$ というネットを作成して 2 線へ拡張する．そして入力の 2 線と出力の 2 線を入れ替えることによって否定論理を実現する．よって回路に使用されたインバータは assign $y = \bar{x}$，assign $\bar{y} = x$ という配線の情報のみに変換され，合成後の回路からは削除されることになる．

5. 適用事例と評価

提案した論理合成手法の有効性を検証をするために，ASDDL セルライブラリとそのダミーライブラリを設計し，0.18μm ターゲットプロセスにおいて 16 ビット乗算器を論理合成し，変換ツールで変換した．論理合成の対象回路を組み合わせ回路としたためフリップフロップ，ラッチ等を除いたセルライブラリを設計した．また今回は R セル，RREG セルを用いた 2 相回路としての論理合成の検証を行わず，2 線回路として ASDDL を論理合成し，検証を行った．

設計したセルライブラリの論理セルの種類を表 1 に示す．セルの駆動力の種類を小さい順に x1，x2$\cdots$x16 で表す．バッファは x1 から x16 のうちの 9 種類の駆動力を用意した．また駆動力確保を目的としたインバータの使用を避けるため，ライブラリ内のインバータは駆動力を x1 とし，面積，固有遅延値はセルライブラリの中で一番大きくした．セルライブラリの遅延モデルは CMOS 線形遅延モデルである．

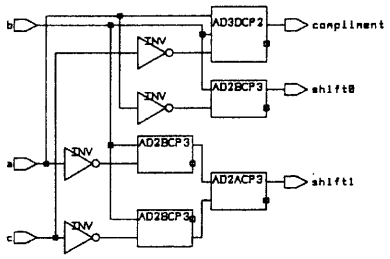
5.1 合成内容の評価

提案した論理合成手法を用いて 16 ビット乗算器（以下 MULT16_ASDDL）を設計した．遅延最適を制約として論理合成を行った．アーキテクチャ等は特に指定しなかったが，論理合成結果を解析すると CMOS 用論理合成ツールの機能により booth デコーダや加算器の機能を持つ module が自動的に利用され，高速な回路として合成されていることが確認できた．

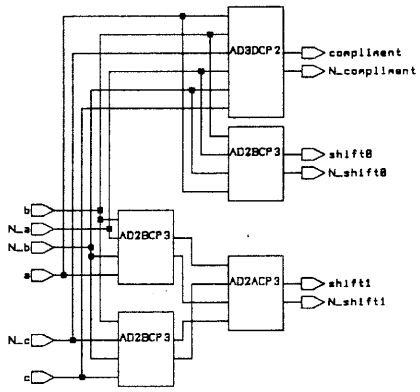
次に論理合成に使用された論理セルとその個数を表 2 に示す．多く使用されているのは AD2A（論理積）や AD3B（排他的論理和）といった代表的な論理セルである．しかし AD3H のような一見しただけでは用途が分からないようなセルも多く

表 2　Logic cells used in MULT16_ASDDL

name	rank	number	total	name	rank	number	total
AD2A	x2	527	616	AD2B	x2	19	93
	x3	89			x3	74	
	x4	1			x4	0	
AD3A	x2	3	39	AD3B	x2	133	133
	x3	0			x3	0	
	x4	36				0	
AD3C	x2	0	0	AD3D	x2	40	83
	x3	0			x3	0	
	x4	0			x4	43	
AD3E	x2	0	0	AD3F	x2	1	1
					x3	0	
					x4	0	
AD3G	x2	93	93	AD3H	x2	227	227
BUF	x0	10	22				
	x1	10					
	x3	2					



(A)



(B)

図 8　A module of booth decoder

使用されていることが分かる．この AD3H のように負論理を多く使用した複雑なセルは CMOS では容易に実現できないが，ASDDL では簡単に実現できる．よってこのような論理セルを他の代表的な論理セルと同様に数多く論理合成に使用できることは ASDDL を論理合成に使うことの大きなアドバンテージであるといえる．また逆に，論理合成は ASDDL の「常に正，負，両論理を有している」という特徴を最大限利用できるともいえ，ASDDL 回路の論理合成が大変有用であることがわかる．

変換ツールによる，ダミーライブラリの論理合成回路から

ASDDL 回路への変換の様子を図 8 で示す．同図は論理合成結果における booth デコーダ内のある module を viewer で表示させたものである．(A) はダミーライブラリでの論理合成結果であり，インバータで入力信号の負論理を生成していること，各セルの 2 つの出力のうち，どちらかが使用されていることが分かる．(B) は (A) の回路を変換ツールで変換した後の回路であるり，インバータが削除され，完全な 2 線回路になっていることが分かる．

変換ツールによって削除されたインバータの総数は 275 個であった．これらのインバータは図 8 に示すような，論理合成中に使用された booth デコーダなどの各 module 内でのみ使用されており，セル間の配線上には全く存在しないことを確認した．インバータは各 module の入力信号の負論理を生成するためだけに使用され，駆動力確保のためにはインバータではなく表 2 のようにバッファが使用されたため，インバータ削除による遅延増加の問題を避ける事ができた．

以上の結果より，ダミーライブラリと変換ツールを使用することにより，CMOS 用のハイレベルな論理合成環境をそのまま適用しながら ASDDL 回路の特徴を十分適用した論理合成が可能であることを確認できた．

5.2 フルカスタム回路，CMOS 論理合成回路との比較評価

ASDDL 回路を論理合成した MULT16_ASDDL に対し，

- 従来通りフルカスタム設計した ASDDL 乗算器
(以下 FC_MULT16_ASDDL)
- 市販の論理合成ツールにより論理合成した CMOS 乗算器
(以下 MULT16_CMOS)

の 2 つを設計対象として，論理合成した ASDDL 回路の性能を評価した．

FC_MULT16_ASDDL は従来の設計手法通り回路全体の回路図を直接入力する方法で設計した．booth デコーダは使用せず，部分積を生成して wallace tree で加算し，最後に BLCA (Binary Look-ahead Carry Adder) で桁上げ吸収加算を行う．またレイアウトの際は論理合成用に設計した ASDDL セルライブラリのセルを使用し，不足するセルに関しては新たに追加してレイアウトを行った．使用した 8 種類のセルの内，新たに追加したのは 1 種類であった．一方 MULT16_CMOS は市販の論理合成ツールを使用し，booth デコーダと BLCA を使用し，遅延最適の条件を与えて論理合成して回路を得た．

MULT16_ASDDL を含む全 3 回路に対し，自動配置配線後に RC 抽出し，電源電圧を通常の 1.8V より 10%低下させた 1.62V，プロセス条件を slow として遅延時間を測定した．MULT16_ASDDL，FC_MULT16_ASDDL はランダムに 2000 パターンの入力を与えてダイナミック解析を行い，MULT16_CMOS においてはスタティック解析を行った．測定結果を表 3 にまとめる．

結果より，論理合成した回路はフルカスタム設計した回路に比べ遅延で約 26%，面積で約 42%増加した．論理合成ツールで最適化されているにもかかわらずフルカスタム設計に及ばないのは，ライブラリ自体の設計に不十分な点があるためだと考えられる．セルのトランジスタサイジングの最適化，キャラクタライズの精度を向上させることによって，フルカスタム設計の回路と同等の回路が論理合成できるようになると考えられる．

表 3 Simulation result

circuit	Delay[nsec]	Area[mm^2]
MULT16_ASDDL	3.13 (1.00)	0.081(1.00)
FC_MULT16_ASDDL	2.48 (0.79)	0.057(0.70)
MULT16_CMOS	4.73 (1.51)	0.041(0.47)

CMOS 回路と比較すると遅延時間は約 65%，面積は約 2 倍となっている．この数値は従来からの ASDDL 回路と CMOS 回路の遅延時間と面積の関係にほぼ一致している．よって booth デコーダと BLCA を使用した CMOS 回路とほぼ同等の構成の回路が論理合成できたと考えられる．

6. おわりに

2 線 2 相論理回路方式を用いて，信号の立上り遷移と立下がり遷移に要する時間を意図的に非対称とすることで高速化を図った新たな回路方式，ASDDL/ASD-CMOS の論理合成手法を提案し評価した．

ダミーライブラリと変換ツールを用いる手法を適用することで，2 線 2 相論理回路方式であるにもかかわらず CMOS 用に構築されたハイレベルな論理合成環境をそのまま適用できる論理合成を可能にした．また，論理合成が ASDDL のような 2 線回路には大変有効になることも確認できた．

謝辞 シミュレーション用データを提供下さった関係各社に感謝致します．

文　献

[1] 瀧和男, 八木幹雄, 森本薫夫, 尾形俊郎, 池見憲一, 北村清志．高速低消費電力回路方式 ASDDL/ASD-CMOS とその評価．DA シンポジウム 2001 論文集, pp.113–118, Jul. 2001.

[2] 池見憲一, 瀧和男, 北村清志, 尾形俊郎, 八木幹雄．低電圧動作を視野にいれた高速演算用回路方式 ASDL の提案．電子情報通信学会論文誌, VLD200-75, pp35–40, Nov. 2000.

[3] 日経マイクロデバイス，"歩留まりを上げる設計，設計効率を上げる製造"，日経 BP 社 pp. 53–60 (2003)

[4] 八木幹雄, 森本薫夫, 瀧和男, 北村清志, 高速低消費電力論理回路方式 ASDL のパイプライン化手法とその評価デザインガイア 2001