SDIモデルに基づいた非同期式パイプライン・データパスの論理合成

今井 雅[†] 中村 宏[†] 南谷 崇[†]

本論文では，相対的遅延変動に着目した新しい遅延モデルであるSDI（Scalable-Delay-Insensitive）モデルに基づいて非同期式パイプライン・データパスを構成する手法を提案する．データパスの論理合成では，LUTベースの多段論理合成アルゴリズムを適用する．また，SDIモデルで規定する相対遅延変動率の上限値によってパイプラインステージの制御回路が異なることを示し，それぞれの遅延モデルの下で構成した回路の遅延および回路量を比較した結果を示す．

Logic Synthesis of Pipelined Asynchronous Datapaths Based on the SDI Model

MASASHI IMAI,[†] HIROSHI NAKAMURA[†] and TAKASHI NANYA[†]

In this paper, we present a synthesis method of pipelined asynchronous datapaths based on the SDI (Scalable-Delay-Insensitive) model. For datapath synthesis we use a LUT-based logic synthesis algorithm. We also show that the synthesized control circuits for the pipeline stages differ according to the value of the relative variation ratio, which is defined in the SDI model. We evaluate the delay and size of synthesized circuits under different delay models.

1. はじめに

VLSI技術の微細化が進むにつれ，ゲート遅延が小さくなる一方，配線抵抗の増大で配線遅延が絶対的にも相対的にも増加している．そのため，システム全体へ位相差なくクロック信号を分配する必要のある従来の同期式デジタルシステムでは，高速素子の性能を十分に活用できないことが指摘されている[1]．また，同期式システムではある時点で使用されていない回路にもクロック信号が供給されるため，不必要な電力消費が生じる．

高速素子の性能を有効に活用し，消費電力を低減するための方法の1つにシステムを非同期式で構成する方法がある．非同期式回路は高速性，低消費電力，高い拡張性などの多くの利点を持っており，近年様々な研究がなされている．

我々は非同期式設計スタイルの実用化を目指して，32ビットの非同期式プロセッサTITAC-2[2]を0.5 μmルールのプロセスで設計・試作し，Dhrystoneベンチマークで54 MIPSの性能を実現した．

† 東京大学先端科学技術研究センター
Research Center for Advanced Science and Technology, University of Tokyo

非同期式システム設計では，その前提となる遅延モデルが重要な役割を果たす．遅延モデルとは，論理ゲートや配線における遅延に関して設ける仮定のことである．これまでの非同期式回路に関する理論的研究でしばしば用いられてきたDelay-Insensitive（DI）モデル[3]やQuasi-Delay-Insensitive（QDI）モデル[4]と呼ばれる遅延モデルに基づいた設計では，現実には起こりそうもない遅延変動に対しても正しい動作を保証する必要がある．そのため，実用的観点からは効率の良い回路とはいえず，十分な速度性能が得られない．そこで，TITAC-2の設計ではより現実的な仮定として，回路要素の絶対的な遅延変動の大きさに上限はないが相対的な変動率には上限があると考え，新たにScalable-Delay-Insensitive（SDI）モデル[2]を導入した．しかし，SDIモデルの下での組織的な回路設計方法についてはいまだ研究段階である．

QDIモデルやSDIモデルのように，回路要素の遅延の上限値を未知とする遅延モデルに基づいてデータパス論理回路を実現するためには，当該論理動作がいつ完了したかを検知する仕組みが必要である．その実現手法の1つとして，2線2相式[1]がある．この方式では，1ビットのデータを2本の信号線対 $(x, \overline{x})$ を用いて表現し，$(x, \overline{x}) = (1, 0)$ を論理“1”に，$(x, \overline{x}) = (0, 1)$

を論理 "0" に対応させ，これらを符号語と呼ぶ．また，状態 $(x,\bar{x}) = (0,0)$ で "0" でも "1" でもない中立的状態を表し，これをスペーサと呼ぶ．この符号語とスペーサを交互に送ることでデータ転送を繰り返す方式が2線2相式である．機能ブロックの入力がスペーサから符号語に遷移し，そのブロックで何らかの処理が行われて出力がスペーサから符号語へ遷移する期間を稼働相，入力が符号語からスペーサへ遷移し，出力が符号語からスペーサへ遷移する期間を休止相と呼ぶ．

2線2相式のデータ転送において，休止相は次の符号語を処理するために回路の初期化を行う期間であり，システムの性能上クリティカルなパスにこのオーバヘッドがかかると全体の性能を制限することになる．この問題を解決する方法の1つは，組合せ回路をパイプライン化することである[5]．パイプライン化することにより，稼働相と休止相を別のパイプラインステージで並列に実行することができ，休止相のオーバヘッドを隠すことができる．

本論文では，SDIモデルの下でパイプライン化された2線2相式データパス回路の論理合成手法を提案し，ベンチマーク回路の合成結果を示す．この論理合成では，DCVSL（Differential-Cascode-Voltage-Switch-Logic）回路[6]が互いに同族なすべての論理関数を同一の2線式論理で実現することに着目し，ルックアップテーブルを前提とした論理合成アルゴリズムを適用する．本手法に基づく合成システムでは，入力として，実現したい論理式（あるいは実現したい回路のゲート記述）と要求されるサイクルタイム，およびSDIモデルで規定する遅延変動率の比の上限値を与えると，DCVSL回路を利用した2線2相式パイプライン化組合せ回路を出力する．

本論文の構成は以下のとおりである．2章では，回路設計の前提とするSDIモデルの概念について述べる．3章では，DCVSL回路を用いた2線2相式多段組合せ回路の構成手法を述べる．4章では，SDIモデルに基づいてDCVSL回路による多段組合せ回路をパイプライン化する手法について述べる．5章では，パイプライン・データパスを生成するアルゴリズムについて述べる．6章では，本論文で提案する構成手法をベンチマーク回路に適用した例を示し，評価を行う．7章では，まとめを述べる．

2. SDIモデル[2]

DIモデルは，遅延の大きさは有限であるが上限値は未知であると仮定したモデルであり，QDIモデルは，配線の分岐先の信号遷移はほぼ同時に起こるという等時分岐の仮定をDIモデルに加えたものである．これらのモデルの下で設計された回路は，現実的には起こりそうもない遅延変動に対しても正しい動作を保証する．そのため，遅延変動に対する信頼性は高いが，回路量が多くなり，十分な速度性能が得られない．一方，SDIモデルでは回路要素の絶対的な遅延変動の大きさに上限はないが，互いに他の要素の遅延に対する相対的な変動率には上限があると考え，遅延に関して次のように仮定する．

『ある回路要素 C に対して，設計者が設計段階で予測した遅延を De とし，システムの生涯を通じて起こりうる実際の遅延を Da とする．このとき，$R = Da/De$ は回路要素 C の時刻 t における遅延変動率を表す．

任意の2つの回路要素 $C1$ と $C2$ の時刻 t における遅延変動率をそれぞれ $R1$, $R2$ とすると，$V = R2/R1$ は時刻 t における相対遅延変動率を表す．このとき，回路には定数 K $(K > 1)$ が存在し，任意の2つの回路要素の間の相対遅延変動率 V に関して，システムの生涯を通じて $1/K < V < K$ がつねに成り立つ．』

SDI回路とは適切に設定された相対遅延変動率の上限値 K のモデルの上で正常に動作する回路である．

回路中の信号遷移 t_1 が信号遷移 t_2 よりも早く生じるように仕様で定められている場合，DIモデルの下では遅延に上限値がないため，図1 (a) に示すように t_1 を原因としなければ t_2 は起きないように回路を構成しなければ正しい動作が保証されない．QDIモデルの下では，分岐配線に遅延差はないため，図1 (b) に示すように，原因となる遷移を t_1 と共有する経路によって t_2 が起きてもよい．これらに対してSDIモデルの下では，図1 (c) に示すように，t_1, t_2 の共通原因となる信号遷移 t からの各経路の遅延をそれぞれ d_1, d_2 としたとき，$K \cdot d_1 < d_2$ が成り立つような回路構成を行えば，信号遷移 t_1 と t_2 の順序関係は保

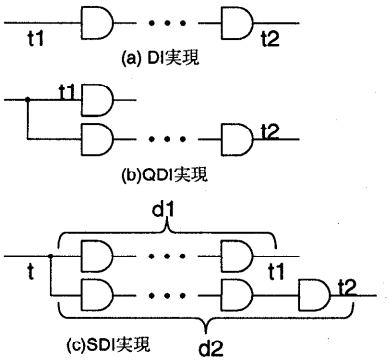

図1 遅延モデルの比較
Fig. 1 Implementations on unbounded delay models.

証される．なお，DIモデルおよびQDIモデルはSDIモデルの特別なケース（$K=\infty$）と見なすことができる．

SDIモデルに基づく回路実現例を図**2**に示す．要求仕様として与えられた相対遅延変動率の上限値が$K=\infty$であるとき，図2(a)に示すように，データパスユニット内部が安定状態に達したことを確認し，かつ出力遷移の完了を確認してから完了信号を生成しなければ，正しい動作が保証されない．これはすなわちQDIモデルに基づく実現と同じであり，完了信号を生成するための論理が大きいため，速度性能を得られない．しかし，現実的には$K=\infty$ということはありえない．したがって，図2(b)に示すように，早いパスの遅延（d_1）と遅いパスの遅延（d_2）が$K\cdot d_1 < d_2$を満たす限り，完了信号生成回路の入力をデータパスユニット内部の信号のみからとるように回路を構成することで，速度性能を向上させることができる．

また，図2(c)に示すように，完了信号をデータパスユニット内部の構成とはまったく無関係に，遅延素子を用いて生成する束データ方式を適用することも考えられる．束データ方式とは，データパスを1線式で構成し，データパスの当該動作が完了したことを示す1ビットのストローブ信号を付加する方式である．回路要素の遅延変動率Rが非常に小さい場合，この方式をSDIモデルとして適用し，速度性能を得ることができる．しかし，遅延変動率Rが大きくなる可能性がある場合，相対遅延変動率の上限値Kが小さい場合でも，ストローブ信号生成回路を固定遅延素子で実現する限り，速度性能が得られない．それはラッチおよびデータパスユニットの遅延変動の最悪値をもとに，遅延素子の遅延の大きさを決めなければならないためである，速度性能を得るためには，ラッチおよびデータパスユニットの遅延変動とほぼ同じように変動する回路を構成しなければならないが，そのような遅延回路を実現することは困難である．

本論文では，回路要素の遅延変動率Rは任意に大きくなりうるが，相対遅延変動率は上限値Kでおさえられるという仮定のもとでパイプライン回路を構成する一手法を提案する．

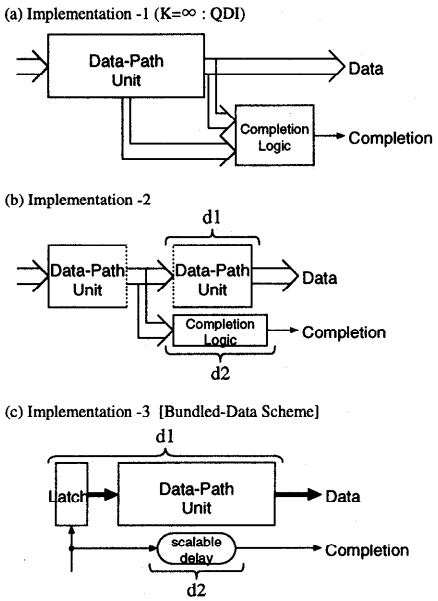

図2 SDIモデルに基づく回路実現
Fig. 2 Circuit realization based on the SDI model.

3. DCVSL多段組合せ回路の論理合成

DCVSL回路[6]はCMOSによる2線式論理設計の一手法であり，ダイナミック論理のDCVSL回路（**図3**）は非同期式2線2相式回路設計に適している[5]．ダイナミック論理のDCVSL回路は，プリチャージを行って出力が$(F_p, F_n)=(0,0)$となる期間と，NMOSツリーにより実現された関数の評価を行って出力が$(F_p, F_n)=(0,1)$あるいは$(F_p, F_n)=(1,0)$となる期間があり，それぞれ2相式の休止相と稼働相とに対応付けられるからである．

DCVSLでは，NMOSを使用したツリー構造により任意の関数が実現できる．一般にNMOSを直列接続すると遷移遅延が比例増加する．また，電荷が抜けきらないために出力が変化しないこともある．そのため，このNMOSツリーの実現においては，NMOSの直列接続段数に3段あるいは4段の制限が課される．したがって，5変数以上の多変数論理関数を実現するためには，NMOSツリーの入力数に制限が加えられたDCVSL回路を多段接続しなければならない．

一方，DCVSLはある信号に対して肯定線と否定線を入力とする2線式論理回路であり，入力信号に対

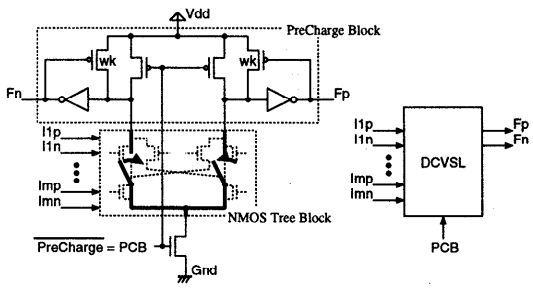

図3 DCVSLの基本構成
Fig. 3 Basic structure of DCVSL.

するNOT演算は肯定線と否定線を交換することで得ることができる．また，論理関数自体に対するNOT演算も出力の肯定線と否定線を交換することで得ることができる．したがって，ある論理関数を実現したDCVSL回路は同族変換（2値変数の否定，2値変数の置換，論理関数の否定）を有限回行うことによって得られる同族な論理関数をすべて実現していることになる．

互いに同族な論理関数を集めて同族類としたとき，2入力の場合2種類，3入力の場合10種類，4入力の場合224種類の同族類しか存在しない．すなわち，これらすべての論理関数のNMOSツリーをライブラリとして用意すれば，N ($N=2\sim4$) 入力の任意の論理関数をDCVSLで実現することができる．

このような同族類ライブラリを用いてDCVSL多段組合せ回路を構成する場合，LUT（Look-Up Table）ベースのFPGA用に開発された論理合成アルゴリズム[7]を適用することができる．LUTベースのFPGAはN入力までの任意の論理関数を実現できるLook-Up Table（N-LUTと略記）を使用したものであり，N-LUTは一般的にSRAMにより実現されている．このN-LUTをSRAMではなくDCVSL回路による同族類ライブラリを用いて実現することで，同期式FPGA用に開発された既存の様々な論理合成アルゴリズムをそのままDCVSL回路を用いた非同期式データパスの論理合成に適用することができる．

4. SDIモデルに基づくパイプライン・データパスの構成方法

4.1 DCVSLを使用したパイプラインの基本構成

QDIモデルの下で，DCVSL回路を使用してパイプライン・データパスを構成することはMartinら[5]により提案されており，その基本構成は**図4**のように表すことができる．

QDIモデルに基づくパイプライン・データパスのステージ制御回路は，図4において破線で囲われた部分，すなわちDCVSL回路以外すべての回路である．この制御回路では，パイプラインの各ステージごとにデータの有効性を示すために，すべての信号線対の遷移完了をNORゲートで検知し，その結果をC素子によりまとめている．QDIモデルの下ではこのようにすべての信号線対の遷移完了を確認する必要がある．一方，SDIモデルの下では，図1(c)に示されるように，早いパスの遅延（d_1）と遅いパスの遅延（d_2）が相対遅延変動率の上限値Kに対して$K\cdot d_1 < d_2$を満たすように構成することができれば，必ずしもすべての信号線対の遷移完了を検知しなくても正しいデータ転送を保証することができる．

初めに図4において，レジスタからの出力が初段のDCVSL回路へ入力される部分に着目する．この回路では，DCVSL回路への入力の遷移（t_1）が完了信号生成回路の出力の遷移（t_2）よりも早く生じることが仕様で定められている．そこで，これらの遷移の共通原因となる遷移を求めると，レジスタからの出力を制御するANDゲートへの入力が共通原因（t）となる（**図5**）．したがって，完了信号生成パスの遅延（図5. d_2）とデータパスの遅延（図5. $d_{1p}, d_{1n}, \cdots, d_{np}, d_{nn}$）を比較し，$K\cdot d_{1p} < d_2$, $K\cdot d_{1n} < d_2$, $\cdots$, $K\cdot d_{np} < d_2$, $K\cdot d_{nn} < d_2$を満たすように構成することでSDIモデルの下でのパイプラインステージ制御回路を実現することができる．

次に，2段目以降のDCVSL回路への入力部分に着目すると，先の例と同様に，各段のDCVSL回路への入力の遷移（t_1）は完了信号生成回路の出力の遷移（t_2）よりも早く生じることが仕様で定められている．これらの遷移の共通原因を求めると，それぞれ前段のDCVSL回路へのプリチャージ入力（PCB）が出力遷移の共通原因（t）であることが分かる（**図6**）．DCVSL回路は，稼働相の場合NMOSツリーへの入

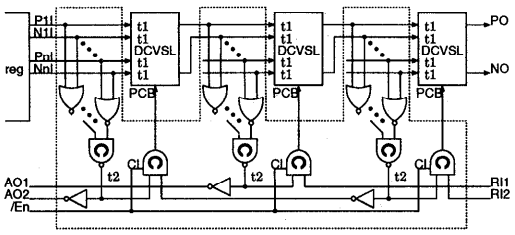

図4 QDIモデルに基づいたDCVSLによるパイプラインの基本構成

Fig. 4 Basic structure of pipelined asynchronous datapaths using DCVSL cells based on the QDI model.

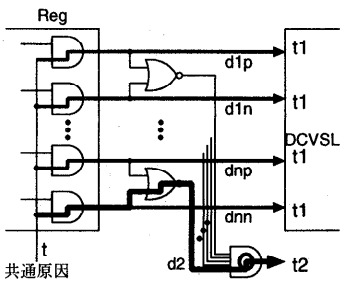

図5 第1ステージの完了信号生成回路の構成

Fig. 5 Structure of the completion signal generation circuit in the first stage.

力が符号語の状態になり，かつPCBが1になったときに出力の遷移が開始される回路である．図4で示された構成はPCBが1になるときにはNMOSツリーへの入力は符号語となっているため，PCBが出力遷移の共通原因となる．また，休止相に関しては，DCVSLはNMOSツリーへの入力にかかわらずPCBが0になることで初期化が行われるため，同じくPCBが出力遷移の共通原因となる．すなわち，稼働相，休止相ともにプリチャージ入力が共通原因となっているため，図6において完了信号生成パスの遅延（図6. d_2）とデータパスの遅延（図6. $d_{1p}, d_{1n}, \cdots, d_{np}, d_{nn}$）が，レジスタからの出力と同様に $K \cdot d_{1p} < d_2$, $K \cdot d_{1n} < d_2$, $\cdots$, $K \cdot d_{np} < d_2$, $K \cdot d_{nn} < d_2$ を満たすように構成することでSDIモデルの下での制御回路を実現することができる．

4.2 SDIモデルに基づくパイプラインステージ制御回路の構成

SDIモデルの下で回路設計を行うには，回路要素（ゲートまたは配線）の遅延の推定値を用いて，仕様で互いの順序関係の定義された経路遅延どうしの比較を行う必要がある．しかし，一般に論理設計の段階では十分な精度で遅延を推定することは困難である．そこで，次のような近似を行う．

一次近似　ゲート遅延（ANDなどの基本ゲートのほか，DCVSL回路の遅延も含む）をすべてユニットディレイとし，配線遅延を0とする．

二次近似　ゲート遅延を，使用するライブラリあるいはテクノロジで規定されている遅延とし，配線遅延は一定の負荷（配線長，配線容量）をかけたときの遅延とする．この配線遅延はゲートの種類によって異なるのでそれぞれのゲートごとに設定する．

図5および図6に示した完了信号生成回路を実現する方法の1つとして，信号線対の遷移完了を示すNORゲートの出力を，2入力C素子のツリー構造回路でまとめる構成が考えられる．このとき，監視する信号線対の数を 2^l 組とすると，完了信号生成までに通過するゲート段数はNORゲートも含めて l 段となる．監視する信号線対の数をそれぞれ1組, 2組, 4組, $\cdots$, $L(2^l)$ 組としたときの完了信号生成回路を**図7**に示す．

監視する信号線対の数により遅いパスの遅延（d_2）が変化するため，図7に示す各完了信号生成回路で実現される相対遅延変動率の上限値 K は異なる．相対遅延変動率の上限値 K は遅いパスの遅延と早いパスの遅延を比較することで求められる．そこで，一次近似および二次近似を適用して，早いパスの遅延（d_1 (d_{1p}〜d_{nn})）と遅いパスの遅延（d_2）を比較すると**表1**となる．ただし，表1では，二次近似としてNEC CBC10ライブラリの値を用いている．監視する2線対の数が1組の場合，早いパスはゲート1段，遅い完了信号生成パスはゲート2段を通過することになるため，一次近似では $K = 2/1 = 2$，二次近似では $K = 0.68/0.42 = 1.62$ の K のもとで実現した回路となる．以下同様に，監視する信号線対の数に従って，その回路で実現される相対遅延変動率の上限値 K

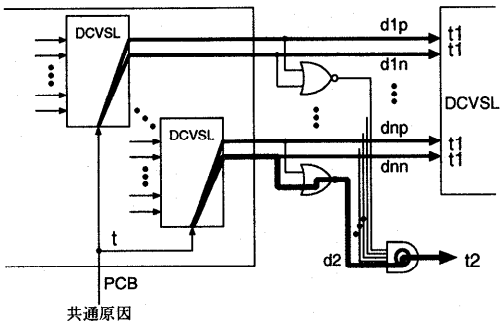

図6 第2ステージ以降の完了信号生成回路の構成
Fig. 6 Structure of the completion signal generation circuit after the second stage.

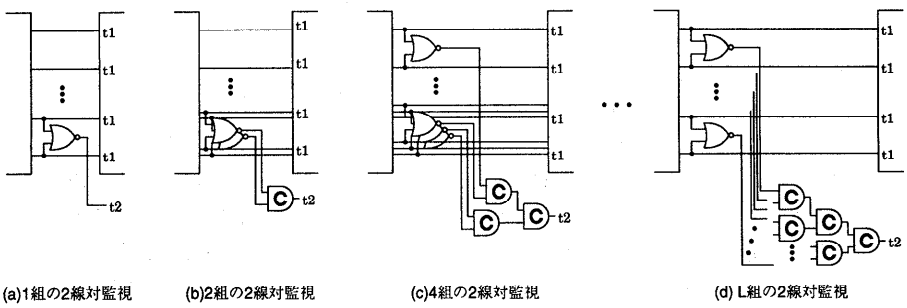

図7 監視する信号線対の数を少なくした完了信号生成回路の構成
Fig. 7 Simple structure of the completion signal generation circuit.

表1 完了信号生成回路における遅延と K の関係
Table 1 The relative variation ratio "K".

監視する	一次近似			二次近似（遅延：ns）		
2線対の数	d_1	d_2	K	d_1	d_2	K
1	1	2	2	0.42	0.68	1.62
2	1	3	3	0.42	0.90	2.14
4	1	4	4	0.42	1.12	2.67
8	1	5	5	0.42	1.34	3.19
16	1	6	6	0.42	1.56	3.71
32	1	7	7	0.42	1.78	4.24

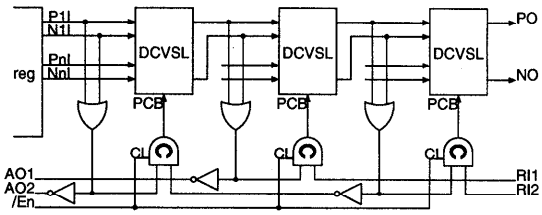

図8 SDIモデルの下で構成したパイプライン
Fig. 8 Pipelined asynchronous datapaths based on the SDI model.

の値が変化する．

SDIモデルに基づく非同期式パイプライン・データパスを構成する際には，表1に示された値を用いて，要求される相対遅延変動率の上限値 K を満たす完了信号生成回路の構成を選択することができる．

図7(a)で示された制御回路をQDIモデルの下で設計された図4のパイプライン・データパスに適用すると，一次近似では $K=2$，二次近似では $K=1.6$ に対するSDIモデルの下で構成された回路として**図8**を得ることができる．

与えられたDCVSL回路に対して，図8のように各ステージごとに制御回路を付加することにより，QDIモデルの下で設計されたものと比較して明らかに制御回路のオーバヘッドが小さいパイプラインを構成することができる．

4.3 DCVSL多段組合せ回路の分割

図8ではDCVSL回路1段を1パイプラインステージとして回路を構成しているが，DCVSL回路 m $(m>1)$ 段ごとに1ステージを構成することも可能である．DCVSL回路2段ごとにパイプラインステージを構成した場合の例を**図9**に示す．

図8と図9を比較すると明らかなとおり，DCVSL回路 m $(m>1)$ 段ごとにパイプラインステージを構成することで，1段ごとに構成したときよりもステージ制御回路の量を減少させることができる．また，次々ステージ以降へ信号を転送するためのDCVSL回路（5章参照）の量を減少させることもできるため，DCVSL

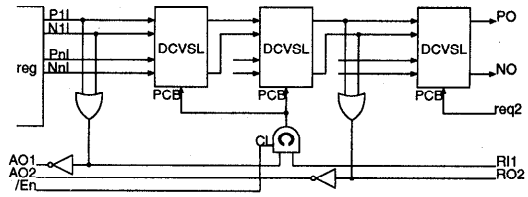

図9 SDIモデルで構成したパイプライン（DCVSL2段）
Fig. 9 Pipelined asynchronous datapaths based on the SDI model (a pipeline stage contains 2 steps of DCVSL).

回路を1段ごとではなく多段ごとにパイプライン化した方が回路量の面では有利である．しかし，稼働相においてDCVSL回路 m 段を通過する遅延は比例増加するため，パイプラインのサイクルタイムは増加することになる．

非同期式パイプラインにおける各ステージのサイクルタイムは前後のステージの処理にかかる遅延やパイプラインへのデータ供給およびパイプラインから出力されたデータの消費にかかる遅延により大きく変化するため，簡単に求めることはできない．しかし，次の近似により1ステージに含まれるべきDCVSL回路の最適な段数を求めることができる．

要求されるサイクルタイムを D_{cycle} とする．DCVSL回路1段の稼働相の遅延を d_w，休止相の遅延を d_i とし，制御にかかる遅延を d_c とする．1ステージに m 段のDCVSL回路が含まれるとすると，稼働相は m に比例した遅延となるため $m \times d_w$，休止相はいっせいに行われるため d_i，制御遅延は稼働相および休止相とも必要で $2d_c$ となる．したがって，そのステージのサイクルタイムはおおよそ $m \times d_w + d_i + 2d_c$ となる．休止相の遅延を $d_i = d_w$，制御にかかる遅延を $d_c = (K-1) \times d_w$ と仮定すると，サイクルタイムは $(2K+m-1) \times d_w$ となる．このとき，$(2K+m-1) \times d_w < D_{cycle}$ を満たすように，$m < D_{cycle}/d_w - 2K + 1$ $(m \geq 1)$ となる整数 m の最大値が1ステージに含まれるDCVSL回路の最適な段数である．

5. パイプライン生成アルゴリズム

DCVSL回路を使用したパイプライン・データパスの論理合成を行うアルゴリズムを以下に示す．このアルゴリズムでは，DCVSL回路による多段組合せ回路の論理合成とパイプライン化を同時に行う．アルゴリズムを適用する前の段階として，要求されるサイクルタイムとDCVSL回路の遅延を用いて，1ステージに含まれるDCVSL回路の段数（m）をあらかじめ求め

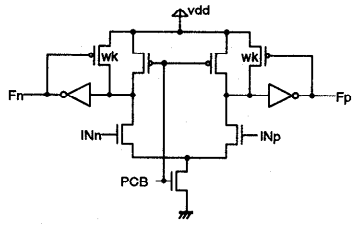

図10 信号転送用DCVSL回路
Fig. 10 A DCVSL circuit for the signal transfer.

ておく.

アルゴリズムは基本的に3つのフェーズから成り立っている.

初めに,与えられた論理式あるいは回路を入力数が2までの基本ゲート(INV, AND, OR, NAND, NOR, XOR, XNOR)により構成する.これは多入力の素子を排除し,マッピングアルゴリズムを適用させるための前処理を行うものである.

次に,初めのフェーズで得られた2入力までのゲートによる回路に対して,すべてのゲートのラベル付けを入力側から行う.得られたラベルは入力からの通過ゲート段数を表している.すなわち,DCVSL回路1段ごとにパイプライン化した場合,そのラベルが何ステージ目のパイプラインになるかを表すことになる.

最後に前のフェーズで付けられたラベルを基に,出力側から N-LUT を適用して回路を構成する.この際,得られた N-LUT を DCVSL 回路に変換するとともにパイプライン制御回路を付加する.DCVSL 回路1段ごとにパイプラインステージを構成する場合,各ラベルごとにまとめられたものそれぞれにパイプライン制御回路を付加する.また,DCVSL 回路2段ごとにパイプライン化するときはこのラベルを基に $(0,1),(2,3),(4,5),\cdots$ を組にしてパイプラインステージを構成することになる.同様に DCVSL 回路 m 段ごとにパイプラインステージを構成する場合,$(0,1,\cdots,m-1),(m,m+1,\cdots,2m-1),\cdots$ を組にしてパイプラインを構成し,それぞれ制御回路を付加する.制御回路を付加する際には,SDIモデルで規定する K の値に従って,表1を利用して監視する2線対の数を決定する.

また,DCVSL 回路の出力が次々ステージ以降のパイプラインで使用されていた場合,次ステージからその出力が使用されるステージの前のステージまで,DCVSL 回路の出力を伝播するための DCVSL 回路(**図10**)を挿入する必要がある.

N 入力までの任意の論理関数を実現できる N-LUT を使用する場合のアルゴリズムの詳細を以下に示す.

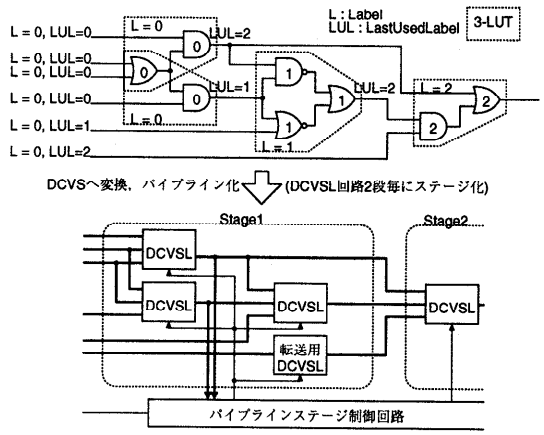

図11 アルゴリズム適用例
Fig. 11 An example of the algorithm.

また,$N=3$ として,アルゴリズムを回路に適用した例を**図11**に示す.

(1) 2入力までの基本ゲートによる回路に変換する.
- DMIG (Decompose-Multi-Input-Gate) アルゴリズム[7]を適用し,論理関数を2入力までの基本ゲート(INV, AND, OR, NAND, NOR, XOR, XNOR)を用いて実現する.

(2) 入力からトポロジカルな順でゲートにラベル付けを行う.
- Primary Input (PI) のラベルを0, LastUsedLabel を0とする.
- 注目するゲート v の入力の最大ラベルを p とする.このとき,$S_p(v)$ を,ゲート v から入力へとたどったときに通過するゲートのうち,ラベル p を持ったゲートの集合とする.
- 集合 X に含まれるゲートの入力信号に対し,その信号をドライブするゲートが集合 X に含まれない入力信号の数を $input(X)$ とする.このとき,$input(S_p(v)\cup\{v\}) \leq N$ となる場合,ゲート v のラベルを p とし $(\mathrm{Label}(v)=p)$,それ以外のときは $p+1$ とする $(\mathrm{Label}(v)=p+1)$.
- ゲート v の各入力信号に対し,その入力信号をドライブするゲート v' の LastUsedLabel とゲート v のラベルを比較する.このとき,$\mathrm{LastUsedLabel}(v') < \mathrm{Label}(v)$ となる場合,$\mathrm{LastUsedLabel}(v') = \mathrm{Label}(v)$ とする.
- Primary Output (PO) になるまで繰り

返す.

(3) 出力から順に N-LUT を適用する.

- 出力から順に，関数を実現するように同じラベルを持ったものをまとめて1つの N-LUT を作成し，DCVSL に変換する.
- SDI モデルで規定する遅延変動率の比 K の値に従ったパイプライン制御回路を付加する．$K-2$ 組の2線対の OR をとり，それらを2入力 C 素子でまとめる.
- DCVSL 回路1段ごとにパイプラインステージを構成する場合はラベルごと，DCVSL 回路 m 段ごとにパイプラインステージを構成する場合は $(0,1,\cdots,m-1),(m,m+1,\cdots,2m-1),\cdots$ を組としたパイプラインを構成する.
- DCVSL 回路 m 段ごとにパイプラインステージを構成する場合，
  LastUsedLabel(v) > Label(v) + m のとき，次ステージ以降，
  LastUsedLabel(v) の DCVSL 回路が含まれるステージの前の段まで信号伝搬用の DCVSL 回路（図 10）を加える.
- ゲートの入力がすべて PI になったとき終了.

6. 論理合成結果とその評価

論理合成用ベンチマーク回路 LGSynth89[7),8)]に前述のアルゴリズムを適用して DCVSL 回路によるパイプライン・データパスを構成した結果を**表2**，**表3**，**表4** に示す.

各表において，回路の遅延を求めるときはゲート遅延として NEC 社の CBC10 ライブラリの遅延を使用している．また，非同期式回路は事象駆動型論理回路であるため，最悪遅延ではなく平均遅延で回路が評価される．そのため，表2〜表4 のサイクルタイムは論理合成されたパイプラインの平均サイクルタイムを示している（サイクルタイムはそのパイプラインステージの稼働相が開始されてから休止相を経て次の稼働相が開始されるまでの遅延である）．平均サイクルタイムを求めるときは，パイプラインへのデータ供給およびパイプラインから出力されたデータの消費がそれぞれ 1 (ns) で処理されるものとし，2^{16} 回のランダムな入力信号パターンを入力したときの平均を求めている.

また，制御信号の分配にはファンアウト数16の制限を設けている．16以上の DCVSL 回路に同一の制御信号を分配するときはバッファを通し，すべての DCVSL 回路に対し，信号がフラットに伝達されるようにしている.

表2 3入力同族類ライブラリを用いたときの合成結果 ($K=2$)
Table 2 Synthesis results using 3 input library ($K=2$).

Bench-mark	最大段数	DCVSL1 段ごと		DCVSL2 段ごと	
		セル数	Cycle Time (ns)	セル数	Cycle Time (ns)
5xp1	6	81	4.51	69	5.91
9sym	7	48	4.53	38	5.80
alu4	16	635	4.60	599	6.22
apex1	13	1042	4.62	978	6.58
apex2	15	243	4.62	219	6.35
bw	6	129	4.55	113	5.97
clip	9	91	4.50	78	5.92
con1	4	23	4.48	16	5.87
duke2	9	267	4.61	238	6.34
e64	32	143	4.57	111	6.32
misex1	5	54	4.55	43	5.93
misex2	7	87	4.56	72	5.73
sao2	9	120	4.59	106	6.20
vg2	9	76	4.55	64	6.08

表3 4入力同族類ライブラリを用いたときの合成結果 ($K=2$)
Table 3 Synthesis results using 4 input library ($K=2$).

Bench-mark	最大段数	DCVSL1 段ごと		DCVSL2 段ごと	
		セル数	Cycle Time (ns)	セル数	Cycle Time (ns)
5xp1	4	58	4.58	46	5.98
9sym	6	35	4.54	27	5.82
alu4	13	456	4.72	408	6.36
apex1	9	730	4.78	687	6.80
apex2	9	177	4.66	155	6.46
bw	3	97	4.60	87	6.01
clip	6	69	4.58	55	6.03
con1	3	18	4.49	13	5.89
duke2	6	213	4.72	180	6.42
e64	17	80	4.64	59	6.36
misex1	3	41	4.55	30	5.95
misex2	3	56	4.58	42	5.78
sao2	5	87	4.61	74	6.28
vg2	7	66	4.58	52	6.16

表2 は3入力までの任意の論理関数を実現できる DCVSL ライブラリを使用して一次近似による $K=2$ の SDI モデルの下で回路を構成したときの値である．DCVSL 回路の最大段数，および1段ごとにパイプライン化，2段ごとにパイプライン化したときそれぞれの回路量と平均サイクルタイムを示している．表3 は同様に4入力までの任意の論理関数を実現できる DCVSL ライブラリを使用して論理合成を行ったときの結果である.

表2，表3 より，DCVSL 回路1段ごとにパイプラインを構成した場合は DCVSL 回路2段ごとにパイプラインを構成したときよりも回路量が増加していることが確認できる．また，各パイプラインに含まれる

表4 遅延モデルを変化させた場合の合成結果
Table 4 Synthesis results when change the delay model.

Benchmark	最大段数	SDI ($K=2$) セル数	SDI ($K=2$) Cycle Time (ns)	SDI ($K=3$) セル数	SDI ($K=3$) Cycle Time (ns)	QDI セル数	QDI Cycle Time (ns)
5xp1	6	81	4.51	95	4.94	113	5.99
9sym	7	48	4.53	62	4.97	74	5.77
alu4	16	635	4.60	683	5.06	788	6.38
apex1	13	1042	4.62	1130	5.32	1785	6.68
apex2	15	243	4.62	285	5.22	400	6.22
bw	6	129	4.55	145	5.12	188	5.98
clip	9	91	4.50	110	5.08	137	5.79
con1	4	23	4.48	31	4.98	35	5.63
duke2	9	267	4.61	287	5.08	430	6.02
e64	32	143	4.57	210	5.10	228	5.92
misex1	5	54	4.55	64	4.98	72	5.54
misex2	7	87	4.56	102	5.04	123	5.98
sao2	9	120	4.59	140	5.08	183	6.02
vg2	9	76	4.55	98	4.98	110	5.72

DCVSLの段数を1段から2段にしても，サイクルタイムは2倍にはならず，1.5倍程度でおさえられることが分かる．これは，DCVSLの段数が増えると稼働相の遅延は比例増加するが，プリチャージ（初期化）を行う休止相はパイプラインステージごとにいっせいに行われるためである．

表2と表3を比較することにより，3入力の任意関数を実現できるライブラリを使用したときの方が4入力の任意関数ライブラリを使用したときよりも，ほとんどの場合サイクルタイムが短くなっていることが確認できる．これは4段のNMOSツリーよりも3段のNMOSツリーで構成されたDCVSL回路の方が高速であることと，制御信号を各DCVSL回路に分配するときに必要とするバッファの数が3入力の任意関数ライブラリを使用した方が少ないためである．また，セル数を比較すると当然のことながら4入力の任意関数ライブラリを使用した方が少ない．

表4は3入力までの任意関数を実現できるライブラリが利用可能であるとして，遅延モデルとしてQDIモデルの下で構成した場合，SDIモデル（一次近似$K=2$）の下で構成した場合，SDIモデル（一次近似$K=3$）の下で構成した場合それぞれのセル数とサイクルタイムを求めたものである．また，各パイプラインステージに含まれるDCVSLの段数は1段として構成している．

表4より，SDIモデルの下で構成したときには，QDIモデルの下で構成したときと比較して，サイクルタイムが約70〜95%に短縮できることが確認できる．また，SDIモデルの下で構成した方がセル数が少なくなり，回路規模が大きいほどこの傾向が大きくなることが確認できる．

7. まとめ

SDIモデルの下で，DCVSL回路を利用した非同期式パイプライン・データパスの論理合成を行う手法を提案した．データパスの論理合成においては，DCVSL回路が互いに同族なすべての論理関数を同一の2線式論理で実現できることを示し，LUT（Look-Up Table）を前提としたアルゴリズムを適用できることを示した．また，ステージ制御回路の構成においては，相対遅延変動率の上限値によって異なる制御回路とその構成方法を示した．

論理合成用ベンチマーク回路に対し，提案するアルゴリズムを適用してパイプライン・データパスを構成し，評価を行った．また，QDIモデルの下で構成した場合との比較を行い，SDIモデルの下で構成した方がセル数が少なく，サイクルタイムが最大約70%まで短縮できることを確認した．

このアルゴリズムは現在開発中の高速版32ビット非同期式プロセッサTITAC-3のデータパス設計に適用される．

謝辞 本研究の遂行にあたり，日本学術振興会の特別研究員制度（「非同期事象駆動型VLSIシステムの構成に関する研究」）および文部省科学研究費補助金基盤研究（B）09480049，研究者別科学研究費補助金特別研究員奨励費10-04682のご支援をいただいた．また，本研究の一部は（株）半導体理工学研究センターとの共同研究によるものである．

参考文献

1) 南谷 崇：非同期式マイクロプロセッサの動向，情報処理，Vol.39, No.3, pp.181–186 (1998).
2) Takamura, A., Kuwako, M., Imai, M., Fujii, T., Ozawa, M., Fukasaku, I., Ueno, Y. and Nanya, T.: TITAC-2: An asynchronous 32-bit microprocessor based on Scalable-Delay-Insensitive model, *Proc. International Conf. Computer Design (ICCD)*, pp.288–294 (1997).
3) Udding., J.T.: A formal model for defining and classifying delay-insensitive circuits., *Distributed Computing*, Vol.1, pp.197–204 (1986).
4) Martin, A.J.: The Limitations to Delay-Insensitivity in Asynchronous Circuits, *Advanced Research in VLSI*, Dally, W.J. (Ed.), pp.263–278, MIT Press (1990).
5) Martin, A.J., Lines, A., Manohar, R., Nystroem, M., Penzes, P., Southworth, R. and Cummings, U.: The Design of an Asynchronous MIPS R3000 Microprocessor, *Advanced Research in VLSI*, pp.164–181 (1997).
6) Heller, L.G. and Griffin, W.R.: Cascode Voltage Switch Logic: A Differential CMOS Logic Family, IEEE International Solid-State Circuits Conference, Vol.Digest of Technical Papers, pp.16–17 (1984).
7) Chen, K.-C., Cong, J., Ding, Y., Kahng, A.B. and Trajmar, P.: DAG-Map: Graph-Based FPGA Technology Mapping for Delay Optimization, *IEEE Design and Test of Computers*, pp.7–20 (1992).
8) ftp://ftp.mcnc.org/pub/benchmark/

(平成10年 9 月21日受付)
(平成10年12月 7 日採録)

今井 雅

平成7年東京工業大学工学部電気電子工学科卒業．平成9年同大学院情報理工学研究科計算工学専攻修士課程修了．現在東京大学大学院工学系研究科先端学際工学博士後期課程に在学中．非同期式VLSIシステムの構成に関する研究に従事．

中村 宏（正会員）

昭和60年東京大学工学部電子工学科卒業．平成2年同大学院工学系研究科電気工学専攻博士課程修了．工学博士．同年筑波大学電子・情報工学系助手．同講師，同助教授を経て，平成8年より東京大学先端科学技術研究センター助教授．計算機アーキテクチャ，ハイパフォーマンスコンピューティング，計算機の上位レベル設計支援，非同期式計算システムの研究に従事．本会平成5年度論文賞，平成6年度山下記念研究賞各受賞．電子情報通信学会，IEEE，ACM各会員．

南谷 崇（正会員）

昭和21年生．昭和44年東京大学工学部計数工学科卒業．昭和46年同大学院修士課程修了．日本電気（株）中央研究所勤務を経て，昭和56年東京工業大学情報工学科助教授，平成元年同電気電子工学科教授．平成7年東京大学計数工学科教授．平成8年東京大学先端科学技術研究センター教授．論理システムの物理的実現に関する諸問題に興味を持つ．工学博士．昭和62年電子情報通信学会論文賞，平成6年大川出版賞，平成10年ASP-DAC Best Paper賞，Outstanding Design賞受賞．著書「順序機械」(岩波書店)，「フォールトトレラントシステムの構成と設計」(槙書店)，「フォールトトレラントコンピュータ」(オーム社) 等．IEEE，ACM，電子情報通信学会各会員．