

制御のタイミングスキューおよびストールに基づく LSI チューニング

上原 八弓[†] 金子 峰雄[†]

[†] 北陸先端科学技術大学院大学 情報科学研究科 〒 923-1292 石川県能美市旭台 1-1
E-mail: †{yayumi,mkaneko}@jaist.ac.jp

あらまし LSI の微細化に伴って、遅延量などのばらつきが相対的に大きくなってきており、最悪値評価に基づく設計では性能向上が難しくなっている。この問題に対して、製造後の LSI の一部チューニングによって性能を確保することが考えられる。本稿ではデータパス回路を対象として、制御タイミング・スキューと制御ストールによって性能劣化を最小限に止めて回路を正しく動作させる手法を提案する。特にここでは与えられたデータパス回路 (構造記述と制御記述) と遅延情報とから、スキュー調整を許してストール数を最小化する問題を提起し、混合整数線形計画法に基づく解法を提案する。なお提案手法が最も効率的に機能するようなデータパス合成は今後の課題となっている。
キーワード スキュー, ストール, チューニング可能 LSI, 混合整数線形計画法

A Tunable LSI Based on Timing Skew and Stall Adjustments

Yayumi UEHARA[†] and Mineo KANEKO[†]

[†] School of Information Science, Japan Advanced Institute of Science and Technology
1-1, Asahidai, Nomi-shi, Ishikawa 923-1292 Japan
E-mail: †{yayumi,mkaneko}@jaist.ac.jp

Abstract With the advance of process technologies, delay variations become relatively larger. As a result, it becomes difficult to improve a performance such as a clock frequency by a conventional worst case based design. To overcome this serious problem, we introduce a tunable LSI in which a part of LSI can be tuned after fabrication to meet timing constraints and performance requirement. In this paper, we propose a datapath which is tuned by timing skew and stall adjustments. We formulate a novel stall minimization problem with timing skew optimization, and derive a mixed integer linear programming (MILP) formulation. The tuned datapath can be expected to operate correctly with a minimum degradation. To develop a datapath synthesis which is appropriate to our approach is one of our important future works.

Key words skew, stall, tunable LSI, mixed integer linear programming

1. はじめに

近年、半導体微細加工技術の進歩に伴って、トランジスタの性能や配線抵抗などのばらつきによる LSI 中の信号伝搬遅延のばらつきが遅延量に対して相対的に拡大している [1]。これにより、信号伝搬遅延の最悪値を見積もり、得られた遅延量を前提にマージンをとりつつ回路設計を行うような従来手法では、所望の性能を確保することが困難になりつつある。このような遅延ばらつきの問題に対して、製造後に回路のクロック周期を調整することで対処することが考えられる。しかしながら、注目している回路ブロックと他の回路ブロックとのインターフェースを考えたとき、回路ブロック毎のクロック周期調整は、クロック信号の共通化の妨げとなるのみならず、回路ブロック間のスムーズなデータの授受の妨げともなることから、回路のクロック

周期を変更せずに、遅延ばらつきの問題に対処する必要がある。また、ある一部分の信号伝播遅延の値のみが悪化したとき、その箇所に対処するためだけに回路ブロック全体へのクロック周期の値を大きくする必要があり、回路全体の処理速度が低下する。これらの問題に対し、統計的な遅延解析を行い、製造後の遅延量を分析し、歩留まりの改善を行う手法などが提案されている [2] [3]。また、各レジスタへの制御信号の到着時刻のばらつきに対しては、ばらつきにロバスタなクロック配線手法などが提案されている [4]。これに対して本研究ではデータパスの信号伝搬遅延のばらつきへの対応として、各チップの遅延のばらつきの状態に応じて、LSI 製造後に一部の回路部品を調整することにより、LSI を高速に、かつ正常に動作させるための回路方式と回路設計手法の開発を目的としている。特に本稿ではレジスタ等への制御信号のタイミングスキューと制御コント

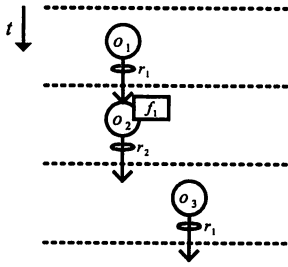


図1 データフローグラフと資源割り当て

ロールステップのストールによって性能劣化を最小限に抑えて回路を正しく動作させる手法を提案する。

これ以降、本稿は次のように構成されている。まず第2章にて、データパス回路が正しく動作するためのセットアップ条件、ホールド条件を説明し、次いで第3章にて、提案手法である制御のスキューとストールを併用してデータパス回路を正しく動作させる方式について説明する。第4章では、スキュー調整の下でストール数を最小化する問題を混合整数線形計画問題として定式化している。第5章にて、幾つかのスキュー・ストール調整実験について報告し、第6章にて本稿をまとめる。

2. セットアップ・ホールド条件

はじめに、レジスタに正しく演算結果が書き込まれる条件であるセットアップ条件とホールド条件について説明する。

図1に、実装対象となる演算と演算間のデータの依存関係を表わしたデータフローグラフに、各演算をどの時刻に行うかを定めるスケジューリング、各演算を行う演算器を決める演算器割り当て、各演算の演算結果のデータを書き込むレジスタを決めるレジスタ割り当てを行った一例を示す。すなわち、演算 o_2 は、演算 o_1 の演算結果を入力として、演算器 f_1 の上で実行される。また、演算 o_1 、 o_3 の演算結果をレジスタ r_1 に、演算 o_2 の演算結果をレジスタ r_2 に書き込んでいる。これらのタイミング図を図2に示す。演算 o_1 、 o_2 、 o_3 の演算結果のデータをレジスタに書込むクロックと同期したタイミングを $\sigma(o_1)$ 、 $\sigma(o_2)$ 、 $\sigma(o_3)$ 、クロック周期を T_c 、レジスタ r_1 から演算器 f_1 を経てレジスタ r_2 へ信号が伝搬する最大遅延を $D_{r_1 \rightarrow f_1 \rightarrow r_2}^{max}$ 、最小遅延を $D_{r_1 \rightarrow f_1 \rightarrow r_2}^{min}$ とする。演算 o_2 の演算結果をレジスタ r_2 へ書き込むには、演算 o_1 の結果を書き込んだレジスタ r_1 から信号が出力され、演算器 f_1 を経て、レジスタ r_2 に信号 (o_2 の演算結果) が到達した後に r_2 にデータを書き込まなければならない。この条件をセットアップ条件といい、式 (1) にて表わされる。

$$\sigma(o_1) \cdot T_c + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{max} \leq \sigma(o_2) \cdot T_c \quad (1)$$

一方、レジスタ r_1 にある演算 o_1 の演算結果は演算 o_3 の結果によって上書きされる。この状況において、演算 o_2 の結果がレジスタ r_2 に正しく書き込まれるためには、レジスタ r_1 から出力される o_3 の結果の信号が、演算器 f_1 を経てレジスタ r_2 に到達する以前に演算 o_2 の結果をレジスタ r_2 に書き込まなければならない。この条件をホールド条件といい、式 (2) で表わさ

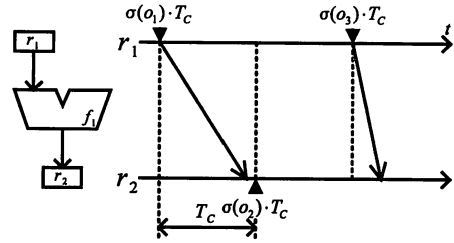


図2 セットアップ条件とホールド条件

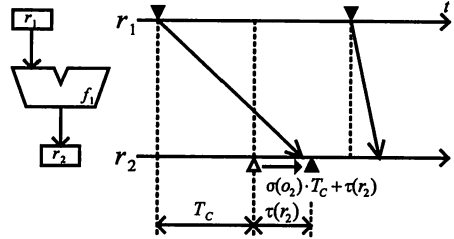


図3 スキューを用いたセットアップ違反回避

れる。

$$\sigma(o_3) \cdot T_c + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{min} > \sigma(o_2) \cdot T_c \quad (2)$$

3. 本研究で提案する信号伝搬遅延のばらつきに 適応する回路方式

LSIの製造ばらつきにより、信号伝搬遅延がばらつく。この為、各レジスタ間の最大遅延、最小遅延がばらつき、設計時に見積もった値を外れ、レジスタに正しいデータが書き込まれる条件であるセットアップ・ホールド条件を満たさなくなる可能性がある。本研究では、製造ばらつきによる信号伝搬遅延のばらつきに適応する為、以下に説明するタイミングスキュー調整とストール操作を製造後に行い、すべての演算がセットアップ・ホールド条件を満たすようにする。

3.1 タイミングスキュー

レジスタの書き込み制御信号の到着時刻は、クロックと同期している。このレジスタの書き込み制御信号の到着時刻にレジスタ毎のズレを持たせることにより、信号伝搬遅延ばらつきによって生じるセットアップ・ホールド条件違反を回避することが考えられる。本稿ではレジスタごとに独立にスキューの値を設定することができるものとし、レジスタ r_i のスキューの値を $\tau(r_i)$ とする。

図3は、製造ばらつきにより、レジスタ r_1 から演算器 f_1 を経て、レジスタ r_2 へ到達するまでの最大遅延の値が大きくなり、 r_1 を入力レジスタ、 r_2 を出力レジスタとする o_2 の演算に対するセットアップ条件が満たされなくなった場合に、レジスタ r_2 の書き込み制御信号のタイミングを $\tau(r_2)$ だけ遅らせることにより、セットアップ条件を満たすことができる例を示している。

図4はスキューを考慮したタイミング図を表しており、スキューを考慮したセットアップ条件式は式 (3)、ホールド条件

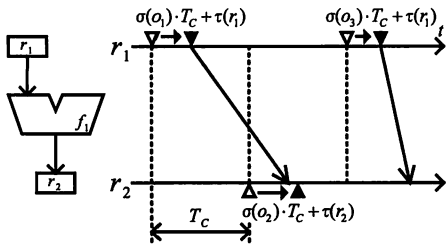


図4 スキューを考慮したセットアップ・ホールド条件

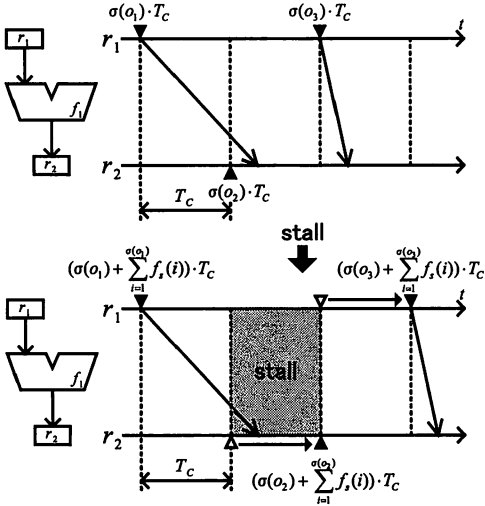


図5 ストール操作によるセットアップ条件違反の回避

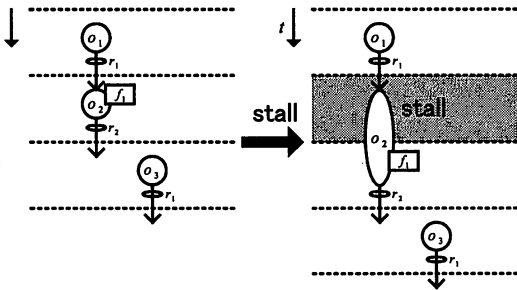


図6 スケジュール上でのストール操作

式は式(4)のように表わされる。

$$\sigma(o_1) \cdot T_c + \tau(r_1) + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{max} + \leq \sigma(o_2) \cdot T_c + \tau(r_2) \quad (3)$$

$$\sigma(o_3) \cdot T_c + \tau(r_1) + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{min} + > \sigma(o_2) \cdot T_c + \tau(r_2) \quad (4)$$

3.2 ストール

ここで考えるストールは、本来刻まれるべき制御ステップの刻みを見送ることで、以降のタイミングを時間軸方向で平行移動する操作である。図5は、レジスタ r_1 のデータを入力とし、演算器 f_1 にて実行される演算 o_2 の結果のレジスタ r_2 への書

き込みについてのセットアップ条件違反を、ストールにより回避する様子を示している。また、これに対応するスケジュール図を図6に示す。

ストールされたコントロールステップ ($\sigma(o_1)$) 以降の演算は、すべてストールしたステップ数だけ後にスケジュールが変更されることになる。本研究で提案する回路方式では、データバス制御部に、あらかじめストールを行うための機構を設けておき、製造後に各コントロールステップのストール数を調整できるようにするものとする。コントロールステップ i のストール数を $f_s(i)$ とするとき、ストールを考慮したセットアップ・ホールド条件式は以下のようになる。

$$\begin{aligned} (\sigma(o_1) + \sum_{i=1}^{\sigma(o_1)} f_s(i)) \cdot T_c + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{max} \\ \leq (\sigma(o_2) + \sum_{i=1}^{\sigma(o_2)} f_s(i)) \cdot T_c \quad (5) \end{aligned}$$

$$\begin{aligned} (\sigma(o_3) + \sum_{i=1}^{\sigma(o_3)} f_s(i)) \cdot T_c + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{min} \\ > (\sigma(o_2) + \sum_{i=1}^{\sigma(o_2)} f_s(i)) \cdot T_c \quad (6) \end{aligned}$$

3.3 ストール数最小化問題の定式化

ここでは、製造後に各レジスタのスキューの値と各コントロールステップのストール数を調整できる回路方式のもので、製造後に回路全体のストール数を最小化問題を考える。

実装する計算が、演算の集合を O 、演算間のデータの依存関係を有向辺の集合 D としたデータフローグラフ $G = (O, D)$ にて与えられ、更に、クロックに同期した演算結果の書き込みタイミングのスケジュール $\sigma : O \rightarrow \mathbb{N}$ 、演算器の集合 F 、レジスタの集合 R 、演算器割り当て $\rho : O \rightarrow F$ 、演算結果のデータを書き込むレジスタを特定するためのレジスタ割り当て $\xi : O \rightarrow R$ 、レジスタ r_i から演算器 f_x を経てレジスタ r_j へデータの信号が到達する最大遅延 $D_{r_i \rightarrow f_x \rightarrow r_j}^{max} \in \mathbb{R}$ 、最小遅延 $D_{r_i \rightarrow f_x \rightarrow r_j}^{min} \in \mathbb{R}$ 、クロック周期 $T_c \in \mathbb{R}$ が与えられ、レジスタ r_i のスキューの値 $\tau(r_i) \in \mathbb{R}$ と、コントロールステップ $i \in \mathbb{N}$ のストール数 $f_s(i) \in \mathbb{N}$ を求める問題とする。

[入力] データフローグラフ (G)、スケジュール (σ)、演算器割り当て (ρ)、レジスタ割り当て (ξ)、遅延情報 (D^{max}, D^{min})、

クロック周期 T_c

[出力] 各レジスタのスキューの値 (τ)、各コントロールステップのストール数 (f_s)

[制約条件] 全ての演算に対するセットアップ条件、ホールド条件

[最適化目標] ストール数最小化

3.4 提案手法を用いた例

提案方式の有用性を、図7に示すスケジューリングと資源割り当てが行われたデータフローグラフを例に説明する。図7で

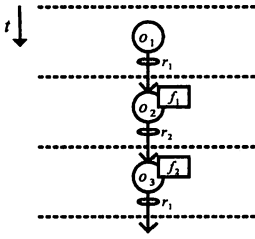


図7 スケジューリング・資源割り当て済みのデータフローグラフ

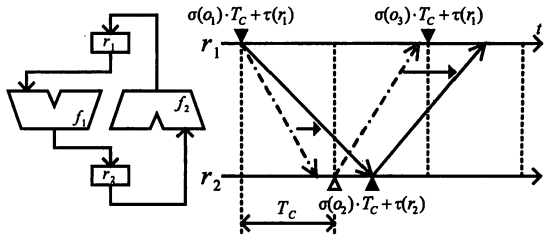


図9 スキューのみの場合

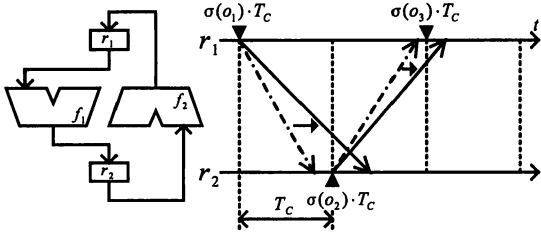


図8 従来方式の場合

は、演算 o_2 を演算器 f_1 の上で実行し、 o_2 の演算結果をレジスタ r_1 に書き込み、次にレジスタ r_2 に書き込まれた演算 o_2 の結果を用いて演算 o_3 を演算器 f_2 の上で実行して、その o_3 の演算結果をレジスタ r_1 に書き込んでいる。このとき、レジスタ r_1 から演算器 f_1 を経てレジスタ r_2 へ到達する最大遅延 $D_{r_1 \rightarrow f_1 \rightarrow r_2}^{max}$ とレジスタ r_2 から演算器 f_2 を経てレジスタ r_1 へ到達する最大遅延 $D_{r_2 \rightarrow f_2 \rightarrow r_1}^{max}$ の設計時見積もりは、共にクロック周期 T_c 未満とする。

製造ばらつきにより、 $D_{r_1 \rightarrow f_1 \rightarrow r_2}^{max}$ と、 $D_{r_2 \rightarrow f_2 \rightarrow r_1}^{max}$ の値がそれぞれ増加し、演算 o_2 、 o_3 の演算結果を書き込むタイミングよりも演算結果の到着が遅くなったとき、スキュー・ストールがない従来の回路方式において、それぞれの演算においてセットアップ条件を満たすことができず、レジスタ r_1 、レジスタ r_2 に正しい演算結果が書きこまれない。また、各レジスタのスキューの値のみ調整可能な場合では、スキューは各レジスタごとに固有の値をとるため、一方の演算(たとえば、 r_1 から r_2 への遅延によって支配される演算 o_2)にあわせてスキューを調整してセットアップ条件を満たせたととしても、他方の演算(r_2 から r_1 への遅延によって支配される演算 o_3)については、セットアップ条件違反を回避できない。

ストール調整のみが可能な場合では、図10のようにストールを行うことで、セットアップ条件を満たし、正しい演算結果がレジスタに書き込まれるが、全体のコントロールステップ数は2ステップ増加する。

これらに対し提案手法では、スキューとストールを共に調整することによって、セットアップ条件を満たし、全体のコントロールステップ数の増加を1ステップにとどめることができ、ストール調整のみの構成と比べて、コントロールステップ数を1ステップ削減することができる。

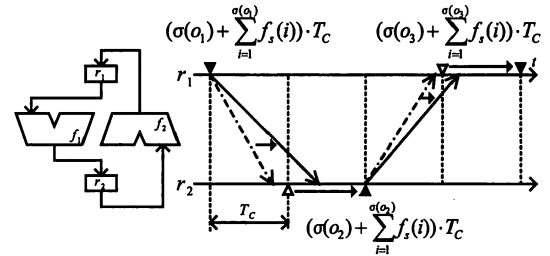


図10 ストールのみの場合

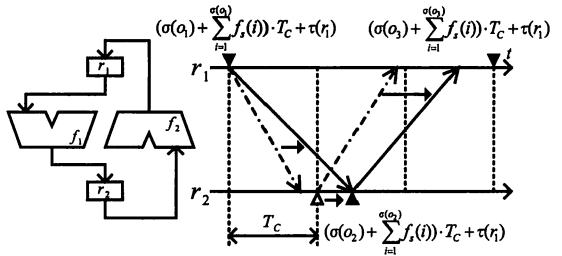


図11 スキュー・ストールを用いた場合

4. MILP による問題の定式化

ここでは、スキューとストール調整の下でのストール数最小化問題を混合整数線形計画法にて解く為の定式化を行う。

クロックに同期した演算 o_i の書き込みタイミングを $\sigma(o_i) \in \mathbb{N}$ 、レジスタ r_i のスキューの値 $\tau(r_i) \in \mathbb{R}$ 、コントロールステップ $i \in \mathbb{N}$ のストール数を $f_s(i) \in \mathbb{N}$ 、クロック周期を $T_c \in \mathbb{R}$ 、レジスタ r_i から演算器 f_x を経てレジスタ r_j に信号が到達する最大遅延を $D_{r_i \rightarrow f_x \rightarrow r_j}^{max} \in \mathbb{R}$ 、最小遅延を $D_{r_i \rightarrow f_x \rightarrow r_j}^{min} \in \mathbb{R}$ としたとき、 $\sigma(o_i)$ 、 T_c 、 D^{max} 、 D^{min} が定数変数となり、各レジスタのスキューの値 $\tau(r_i)$ と、各コントロールステップのストール数 $f_s(i)$ が未知変数となる。これら変数を使って、セットアップ・ホールド条件は以下のように書き表わされる。

$$\begin{aligned}
 & (\sigma(o_1) + \sum_{i=1}^{\sigma(o_1)} f_s(i)) \cdot T_c + \tau(r_1) + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{max} \\
 & \leq (\sigma(o_2) + \sum_{i=1}^{\sigma(o_2)} f_s(i)) \cdot T_c + \tau(r_2) \quad (7)
 \end{aligned}$$

表 1 JWF

手法	平均総実行時間	総実行時間差の最大値	総実行時間差の最小値	平均ストール数	ストール数差の最大値	ストール数差の最小値
提案手法	11.86	0	0	0.86	0	0
クロック周期調整	11.26	0	-4.12	-	-	-
ストールのみ調整	13.88	15	0	2.22	6	0

表 2 IDCT

手法	平均総実行時間	総実行時間差の最大値	総実行時間差の最小値	平均ストール数	ストール数差の最大値	ストール数差の最小値
提案手法	24.5	0	0	0.56	0	0
クロック周期調整	21.17	0	-9.2	-	-	-
ストールのみ調整	27.38	13	0	7.38	13	0

表 3 16-FFT

手法	平均総実行時間	総実行時間差の最大値	総実行時間差の最小値	平均ストール数	ストール数差の最大値	ストール数差の最小値
提案手法	26.18	0	0	5.18	0	0
クロック周期調整	22.29	0	-12.16	-	-	-
ストールのみ調整	32.1	13	0	11.1	13	0

$$\begin{aligned}
 & (\sigma(o_3) + \sum_{i=1}^{\sigma(o_3)} f_s(i)) \cdot T_c + \tau(r_1) + \tau(r_1) + D_{r_1 \rightarrow f_1 \rightarrow r_2}^{\min} \\
 & > (\sigma(o_2) + \sum_{i=1}^{\sigma(o_2)} f_s(i)) \cdot T_c + \tau(r_2) \quad (8)
 \end{aligned}$$

最適化目標である回路全体のストール数最小化は以下のようになる。

$$\sum_{i=0}^{\max_j(\sigma(o_j))} f_s(i) \rightarrow \min \quad (9)$$

5. 評価実験

提案手法の有用性を評価するために、いくつかのベンチマークに対し、提案手法とクロック周期のみを調整する手法、ストールのみを調整する手法との比較を行った。MILPの解法には、CPLEXバージョン11.0.0を用い、プロセッサ:2.4GHz AMD(R)Dual Opteron, メモリ8GB RAMの上で実行した。ベンチマークとして、fourth-order Jaumann wave digital filter(JWF), inverse discrete cosine transform(IDCT), 16-point Fast Fourier Transform(16-FFT)を用いた。クロック周期の値を1とし、各演算器の遅延の値は、各演算器の標準の遅延の値(加減算器:最大遅延0.95, 最小遅延0.5 乗算器:最大遅延1.95, 最小遅延0.95, シフト演算器最大遅延0.9, 最小遅延0.45)を中心値とし、分散0.1の正規分布で与えた。各ベンチマークに対し、50組の遅延値設定を用意し、ストール数最小化実験を行った。ベンチマークに対するスケジューリングはリストスケジューリング、レジスタ割り当てはレフトエッジアルゴリズムにて行っている。表1, 2, 3に各ベンチマークに対する実験結果を示す。提案手法、クロック周期のみ調整、ストールのみ調整の3つの手法について、回路の総実行時間(クロック周期×ステップ数)の平均値、提案手法を基準とした同じ遅延値の設定での総実行時間の差の最大値, 最小値, ストール数の

平均の値(提案手法, ストールのみ調整), 提案手法を基準とした同じ遅延値の設定でのストール数の差の最大値, 最小値について評価を行った。

提案手法とストールのみを調整する手法とを比較したとき、すべてのベンチマーク対して、提案手法の平均総実行時間の値が小さく、またストール数も少なくなっている。これはスキューによる調整が有効に働いているためである。またストール数の差の最大値については、提案手法とストールのみ調整において、最小6(JWF), 最大13(IDCT, 16-FFT)の差が出たが、差の最小値については、与えられた遅延値が、ストールなしでセットアップ・ホールド条件を満たしているケースがあり、ストール数が共に0となって、差の最小値も0となっている。

一方、提案手法とクロック周期のみを調整する手法を比較した場合、すべてのベンチマークにおいて平均実行時間の値が提案手法において悪くなっている。今回の実験では最大遅延のばらつきによる伸びが比較的小さく、わずかなクロック周期の増大でタイミング条件が満たされること多かったために、クロック周期 T_c 単位で総実行時間が延びるストールが劣る結果となった。また制御タイミングスキューはレジスタ毎に設定されるため、入力と同じレジスタに出力を書き戻すような演算に対しては、スキューは機能せず、必ずストールが発生してしまうという問題もある。こうしたことからデータのレジスタへの割り当てによって、必要となるストール数が変化することが容易に想像でき、提案手法が有効に機能するようなデータのレジスタへの割り当て手法は今後の重要な課題となっている。

今回の実験では、演算器ごとに遅延の値を割り当て、配線遅延を考慮していないが、本来はレジスタ間の配線遅延も考慮する必要があり、今後、配線遅延を考慮した実験を行う必要がある。

6. ま と め

本稿では, LSI 製造後に制御タイミングスキューとストールを調整する LSI の回路方式を提案し, その回路方式においてストール数を最小化する手法を提案した. これにより, 信号伝播遅延の製造時ばらつきに対して, クロック周期の値を変更することなくかつ実行時間の伸びを低く抑えて, 回路を正しく動作させることが可能となる. 本稿にて提案した回路方式が最も効率的に機能するようなデータベース合成は今後の重要な課題となっている.

謝 辞

本研究に関して本学情報科学研究科岩垣剛助教, 博士課程小畑貴之氏, 井上恵介氏から多くのご助言をいただいた. 厚く感謝する. また, 本研究の一部は日本学術振興会科学研究費補助金(基礎研究(C)19560340, 2007-2008)の支援を受けたものであり, 関係各位に感謝する.

文 献

- [1] Paul S. Zuchowski, Peter A. Habitz, Jerry D. Hayes, Jeffery H. Oppold, "Process and Environmental Variation Impacts on ASIC Timing", Proc. International Conference on Computer Design(ICC'D), pp.336-342, 2004.
- [2] Jongyoon Jung and Taewhan Kin, "Timing Variation-Aware High-Level Synthesis", In Proc.ICCAD, pp.424-428, 2007.
- [3] 岡田健一, 藤田智弘, 小野寺秀俊, "トランジスタ製造ばらつきにおけるチップ内特性変動を考慮した統計遅延解析手法", 信学技報, pp.7-12, 2001.
- [4] K.Inoue, M.Kaneko, and T.Iwagaki, "Safe clocking register assignment in datapath synthesis", Proc. International Conference on Computer Design(ICC'D), pp.120-127, October 2008.
- [5] Jose Luis Neves and Edy G. Friedman, "Optimal Clock Skew Scheduling Tolerant to Process Variations", Conference on Design Automation(DAC), pp.623-628, 1996.