一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
VLD2018-72,CPSY2018-82,RECONF2018-46(2019-01)

一般同期性能を向上させる遅延最適化に関する検討

佐々栄治郎† 佐藤 真平† 高橋 篤司†

† 東京工業大学 工学院 情報通信系 〒152-8550 東京都目黒区大岡山 2-12-1 S3-58
E-mail: †sassa@eda.ict.e.titech.ac.jp, ††{satos,atsushi}@ict.e.titech.ac.jp

あらまし ディジタル集積回路において，クロックの同時分配を前提とせず，異なるクロック遅延を許容する一般同期方式は，クロックの同時分配を前提とする完全同期方式よりも小さなクロック周期で回路を動作させることが可能である．一般同期方式での最小クロック周期は必ずしも回路の最大遅延パスによって決定されないため，完全同期方式の下で論理合成された回路は一般同期方式に適しているとは限らない．したがって，一般同期方式の下で高性能な回路を得るための回路合成手法が求められている．本報告では，任意のクロックタイミングを設定可能であるとの前提のもと，一般同期方式でのクロック周期を定めるクリティカルサイクルに着目した，最小クロック周期削減を実現する回路遅延最適化手法を提案する．提案手法は，クリティカルサイクル上のパスに対して最大遅延を削減，最小遅延を増加させるような遅延最適化を，市販の論理合成ツールにより構成できる．本手法をベンチマーク回路に適用し得られたネットリストを，完全同期方式を前提とした論理合成結果と比較し評価することで，一般同期方式での最小クロック周期削減に有効であることを示す．

キーワード 一般同期方式, 遅延最適化

On Delay Optimization for Improving General Synchronous Performance

Eijiro SASSA†, Shimpei SATO†, and Atsushi TAKAHASHI†

† Department of Information and Communications Engineering, Tokyo Institute of Technology
S3-58, 2-12-1, Ookayama, Meguro-ku, Tokyo 152-8550 Japan
E-mail: †sassa@eda.ict.e.titech.ac.jp, ††{satos,atsushi}@ict.e.titech.ac.jp

Abstract In the digital integrated circuits, a circuit under the general-synchronous framework where zero clock skew is not assumed is expected to achieve a better performance compared with a circuit under the complete-synchronous framework where zero clock skew is assumed. The minimum clock period under general-synchronous framework is not bounded by a maximum delay path but by the delay paths on a critical cycle. It is crucial to synthesize a circuit under the general synchronous framework in order to obtain a better circuit efficiently. In this paper, we propose a circuit synthesis method to obtain a better circuit in general-synchronous framework under the assumption that any clock scheduling can be realized. The proposed method can be easily implemented by a commercial tool. The validity of our proposed method is shown by comparing to the method under complete-synchronous framework.

Key words general-synchronous framework, delay optimization

1. はじめに

従来のディジタル集積回路設計では，各記憶素子に対して大域的なクロックの同時到達を仮定する完全同期方式が広く採用されてきた．さらなる同期回路の高性能化を実現するため，同じく大域的なクロックを用いた同期方式として，各記憶素子に対するクロックの同時到達を仮定しない一般同期方式 [1], [2] が提案されている．一般同期式回路は，各記憶素子に適切なクロックスケジュールを行うことにより，完全同期方式よりも小さなクロック周期で動作する．

本報告では，一般同期方式を前提としない論理合成ツールを用いて, より高速で回路規模の小さい一般同期回路を得るための回路遅延最適化手法について提案する．

一般同期方式には，その最小クロック周期を決定する記憶素子間の経路が存在する．これらは回路の遅延情報から得られる制約グラフ上で閉路を構成する．この閉路をクリティカルサイクルと呼ぶ．クロック周期を改善するためには，クリティカルサイクル上のパスに対して，最大遅延を削減，最小遅延を増加

させるような遅延最適化が必要である．

完全同期方式での最小クロック周期は，記憶素子間の遅延が最大の経路によって決定される．この経路はクリティカルパスと呼ばれる．設計者は論理合成ツール等の支援を受け，クリティカルパスの遅延を小さくするような論理合成を行うことによって，より高速な同期回路を得ることができる．

しかし，完全同期方式を前提とした論理合成は一般同期方式に適しているとは限らない．完全同期方式におけるクリティカルパスは一般同期方式におけるクリティカルサイクルを構成するパスとは必ずしも一致せず，クリティカルパスの遅延最適化が一般同期方式においてクロック周期を改善するとは限らない．また，一般同期方式ではクロック周期の改善のために最小遅延を増加させる必要がある場合があり，完全同期方式を前提とした論理合成ツールはこれらの要因を考慮しない．

提案手法では，一般同期方式でのクロック周期の下限を定めるクリティカルサイクル上のパスに対して最大遅延を削減，最小遅延を増加させるように制約条件を調整する．また，本手法は完全同期方式を前提とした論理合成ツールを用いて，上記のような制約条件のもと，遅延最適化を繰り返し実行することができる．これにより一般同期方式での最小クロック周期の削減を実現する．

提案手法の評価として，ベンチマーク回路への適用を行い，完全同期方式を前提とした論理合成結果と比較する．評価から，提案手法が少ない面積のオーバーヘッドで一般同期方式の最小クロック周期削減に有効であることを示す．

2. 一般同期方式

2.1 制約条件と同期方式

クロック同期方式において，接続されている記憶素子 u，v には以下のようなタイミングに関する制約が課せられる [1]．

セットアップ制約

$$s(u) - s(v) \leq T - (d_{\max}(u,v) + setup(v)) \qquad (1)$$

ホールド制約

$$s(v) - s(u) \leq d_{\min}(u,v) - hold(v) \qquad (2)$$

ここで，$d_{\max}(u,v)$，$d_{\min}(u,v)$ は記憶素子 u，v 間に存在する経路の中で最大の遅延量，最小の遅延量を示している．また，T はクロック周期を表しており，$s(u)$，$s(v)$ は記憶素子 u，v に入力されるクロックのタイミング（スケジュール）を示す．$hold(v)$，$setup(v)$ はそれぞれ記憶素子 v が要求するホールド時間，セットアップ時間を表す．

一般同期方式では，各記憶素子には同時にクロックが到達するという仮定を前提とせず，$s(u) \neq s(v)$ を許容する．一方，完全同期方式では, 各記憶素子には同時にクロックが到達する，すなわち $s(u) = s(v)$ を仮定する．

2.2 クロックスケジュールによるクロック周期の改善

一般同期方式では，$s(u)$，$s(v)$ の値を適切に設定（クロックスケジュール）することで，クロック周期を削減することができる．完全同期回路と, そのクロックスケジュールを設定することで一般同期方式化した例をそれぞれ図 1，2 に示す．なお，

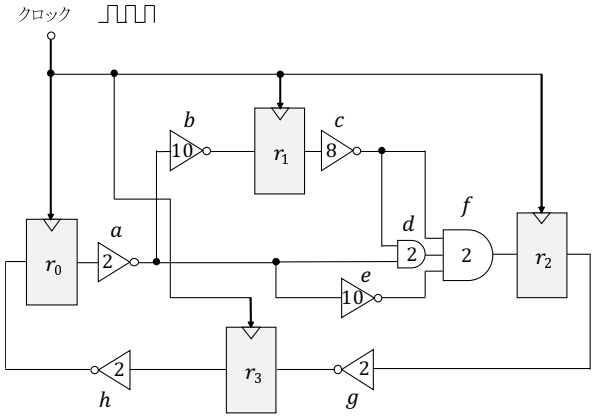

図 1: 完全同期回路

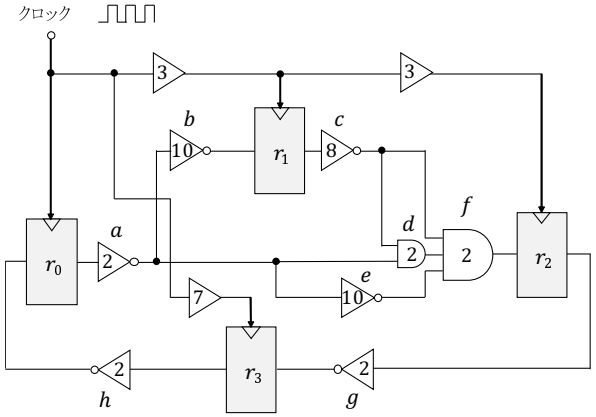

図 2: 一般同期回路

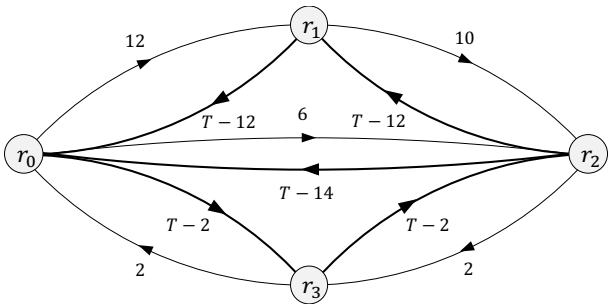

図 3: 制約グラフ

図中の素子につけられた数字はその素子が持つ遅延量を表しており，図中の記憶素子はセットアップ時間，ホールド時間を要求しない．

図 1 の完全同期回路は，記憶素子 r_0，r_2 間に素子 a，e，f を経由する最大遅延 14 を与えるパスを持つ．したがって最小クロック周期は 14 となる．一方，図 2 に示す一般同期回路のクロックスケジュールは $s(r_0) = 0$，$s(r_1) = 3$，$s(r_2) = 6$，$s(r_3) = 7$ である．この同期回路においてクロック周期を 9 とすると，すべてのパスで制約条件 (1)(2) を満たす．したがって，図 2 の一般同期回路は完全同期方式の最小クロック周期 14 よりも小さなクロック周期 9 で動作する．

2.3 制約グラフによる最小クロック周期の導出

一般同期方式によって設計された一般同期回路の最小クロック周期は，回路の遅延情報から定義される制約グラフ $G_T(V,E)$

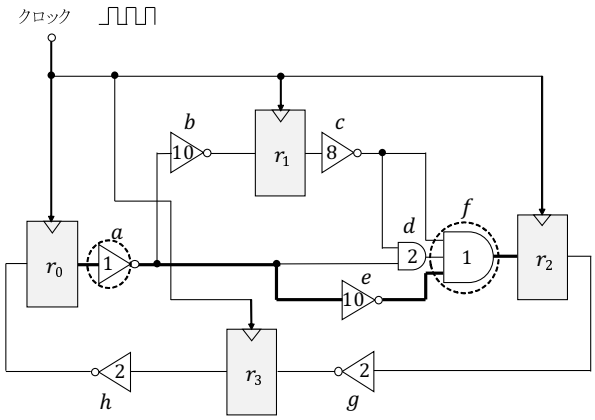

図 4: クリティカルパスを改善

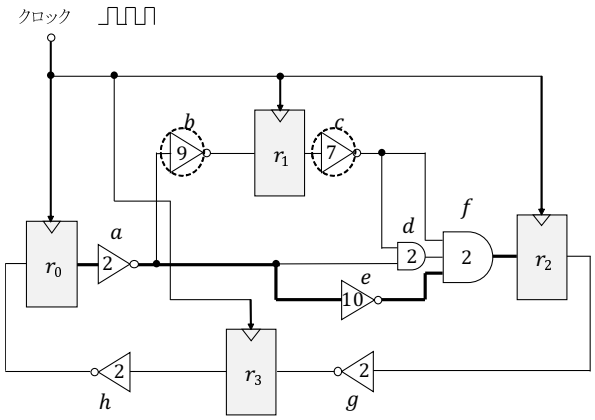

図 5: クリティカルサイクルを改善

によって求めることができる．制約グラフの点 $v \in V$ は回路の記憶素子に対応し，辺 $(u, v) \in E$ はタイミング制約に対応する．セットアップ制約に対応する辺をセットアップ辺，ホールド制約に対応する辺をホールド辺と呼ぶ．それぞれ，辺の重みは $T - (d_{\max}(v, u) + setup(u))$，$d_{\min}(u, v) - hold(v)$ となる．また，クロック周期 t のときの制約グラフを $G_{T=t}(V, E)$ と表現する．

回路の記憶素子に対して任意のクロックスケジュールが設定可能である場合，一般同期式回路の最小クロック周期について以下の定理が成り立つ．

> ［定理 1］([1] [2]) 一般同期式回路 G が正常に動作する最小クロック周期 $T_S(G)$ は，G の制約グラフ $G_{T=t}(V, E)$ が重みの総和が負となる閉路を持たない最小の t である．

任意のクロックスケジュールが設定可能である場合の一般同期回路の最小クロック周期は，制約グラフ $G_{T=T_S(G)}(V, E)$ 上の重みの総和が 0 となる閉路により与えられる．この閉路をクリティカルサイクルと呼ぶ．

クリティカルサイクルの例を挙げる．図 3 は図 1 で示される同期回路の制約グラフである．クロック周期 T を 9 とした場合，制約グラフは重みの総和が負となる閉路をもたない．また，閉路 (r_1, r_0, r_2) は重みの総和が 0 となる．9 未満とした場合，この閉路は重みの総和が負となる．したがって閉路 (r_1, r_0, r_2) は図 3 のクリティカルサイクルとなる．

2.4 遅延最適化

各同期方式によって，クロック周期削減に有効な遅延最適化の手法が異なる例を示す．図 4，5 では，図 1 の同期回路に対して，クリティカルパス，クリティカルサイクルのセットアップ辺を与えるパス上のゲートに遅延量の小さいセルへの置き換えが行われる．この操作により他のゲートの遅延量が変化することはないと仮定する．

クリティカルパス上の遅延最適化は一般同期方式の最小クロック周期を改善しない可能性がある．図 4 では，クリティカルパス (r_0, r_2) 上のゲート a，f に対して，1 だけ遅延の小さなセルへの置き換えを行い，最大遅延を削減している．この操作により，完全同期方式での最小クロック周期は 14 から 12 と改善される．一方，この遅延最適化を一般同期方式の観点から見ると，クリティカルパス (r_0, r_2) はクリティカルサイクルにホールド辺として含まれるが，ホールド辺に対応するパスの最大遅延の削減が最小遅延の減少を引き起こすことで最小クロック周期は改善されず，遅延最適化前の最小クロック周期 9 のまま改善されない．

一方，クリティカルサイクル上の遅延最適化は一般同期方式の最小クロック周期を改善できる．図 5 では, クリティカルサイクル (r_1, r_0, r_2) 上のゲート b，c に対して，1 だけ遅延の小さなセルへの置き換えを行っている．これらのゲートはクリティカルパス上に存在せず，完全同期方式での最小クロック周期は遅延最適化前の最小クロック周期 14 のまま改善されない．しかし，これらはクリティカルサイクルのセットアップ辺を与えるパス上に存在する．したがって，一般同期方式での最小クロック周期は 9 から 8 と改善される．

以上より，図 5 のようなクリティカルサイクルに着目した遅延最適化は，完全同期方式の観点からは積極的に採用されないが，さらなる一般同期性能の改善が期待できる．一般に，セルライブラリには同一の機能を持つ，遅延量や消費電力，面積等の異なるセルが用意されている．文献 [3] では，クリティカルサイクル上のホールド辺を与えるパス上にあるゲートに対して，より遅延量の大きく，消費電力の小さいセルへの置き換えにより，クリティカルサイクルの重みの総和を大きくすることに成功している．これにより最小クロック周期と消費電力が改善される．

2.5 予 備 実 験

より一般的な例として，完全同期方式を前提とした論理合成ツールによる遅延最適化が, 一般同期方式の最小クロック周期削減に対して必ずしも有効でないことを調査する．

RTL 記述されたベンチマーク回路に対して，論理合成時に様々な目標クロック周期を与えることで複数のネットリストを得る．各ネットリストは一般，完全同期方式における 2 つの最小クロック周期を持つ．これらをグラフにプロットすることで各同期方式での最小クロック周期の関係を確認する．

完全同期方式での最小クロック周期は，ネットリストと各記憶素子間の遅延情報により得る．また，一般同期方式での最小クロック周期は，任意のクロックタイミングを設定可能であるとの前提のもと，文献 [4] の手法より得る．

各同期方式における最小クロック周期の組を図 6 に示す．縦軸，横軸はそれぞれ一般同期方式，完全同期方式での最小クロック周期を表している．また，図中の $y = x$ は後述する各同期方式の最小クロック周期における大小関係を説明する．

本実験では，論理合成ツールとして Synopsys Design Compiler(Version L-2016.03-SP4) を使用した．また，ROHM 社の $0.18\mu m$ スタンダードセルライブラリを用いて，IWLS 2005 Benchmarks [5] に含まれる ISCAS Benchmarks の回路 s838_1 に対して論理合成を行った．

図 6 では，完全同期方式の最小クロック周期が小さければ一般同期方式の最小クロック周期も小さい傾向にある．完全同期方式は制約条件 (1),(2) において一般同期方式に包含される．すなわち，一般同期方式の最小クロック周期は完全同期方式のそれより大きくなることはない（図 6 の $y \leqq x$ の領域に存在する）．したがって，完全同期方式の最小クロック周期を削減するような遅延最適化は，一般同期方式においても最小クロック周期を改善することが期待できる．

しかしながら，図 6 は，完全同期方式での最小クロック周期が小さくても，一般同期方式で最小クロック周期の小さい回路が得られるとは限らないことを確認できる．例として，図 6 上の点 A と B に注目する．点 A の示す完全同期方式での最小クロック周期は点 B と比較して 1.9 倍程度高速である．対して，一般同期方式での最小クロック周期は 0.8 倍程度低速となっている．したがって，完全同期方式を前提とした論理合成ツールによる遅延最適化は，一般同期方式の最小クロック周期改善に対して必ずしも有効でない．

完全同期方式を前提とした論理合成ツールは，一般同期方式でクロック周期を決定する幾つかの要因について考慮しない．定理 1 より，一般同期回路の最小クロック周期は制約グラフより与えられるクリティカルサイクルによって決定する．クリティカルサイクルはセットアップ辺，ホールド辺により構成され，辺の重みはそれぞれ式 (1), (2) の右辺より与えられる．これらの閉路の重みの総和を大きくすることで，最小クロック周期を削減することができる．すなわち，クリティカルサイクルを構成するセットアップ辺に対応するパスに対して $d_{\max}(v,u) + setup(u)$ を小さく，ホールド辺に対応するパスに対して $d_{\min}(u,v) - hold(v)$ を大きくすることができれば，一般同期回路の最小クロック周期を削減することができる．しかしながら，クリティカルパスとクリティカルサイクルを構成するパスは必ずしも一致しない．すなわち，完全同期方式でのクロック周期削減に寄与しないパスが一般同期方式での最小クロック周期を決定する可能性がある．さらに，ホールド辺の $d_{\min}$ は最小クロック周期を決定する要因となる．完全同期方式を前提とした論理合成ツールはこれらを考慮しない．したがって，完全同期方式を前提とした論理合成ツールによる遅延最適化は，一般同期方式の最小クロック周期削減に対して必ずしも有効であるとは限らない．

3. 提 案 手 法

提案手法は与えられたネットリストに対して一般同期方式を考慮した遅延最適化を行う．遅延最適化には論理合成ツールの

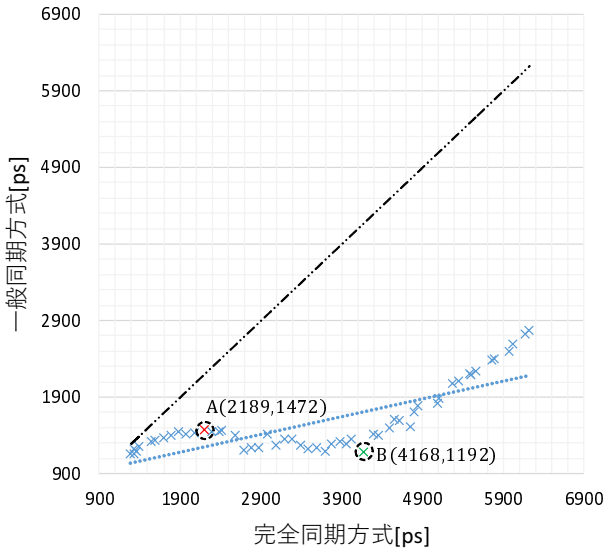

図 6: ベンチマーク回路の各同期方式での最小クロック周期

制約違反修正機能を用いる．論理合成ツールは与えられた目標クロック周期下で制約条件 (1),(2) に違反が発生しないように遅延最適化による修正を行う．提案手法では，後述する設定方法により制約条件と目標クロック周期を調整することで，より厳しい制約条件を論理合成ツールに与える．これにより, クリティカルサイクルのセットアップ辺に対しては辺の重みを削減，ホールド辺に対しては辺の重みを増大させるような遅延最適化を実現する．

提案手法のフローを図 7 に示す．提案手法は入力を回路のネットリストとし，一般同期方式で高速化されたネットリストを出力する．入力されたネットリストの遅延情報を基に制約グラフを作成し, 文献 [4] の手法より負閉路探索を行い，一般同期方式での最小クロック周期 T_{general} とクリティカルサイクルを与えるパスを同定する．また，T_{general} を過去の最適解と比較して解の更新を行う．得られたパスの遅延情報を基に目標クロック周期と制約条件の設定を変更した後，論理合成ツールにより, クリティカルサイクルのセットアップ辺に対しては辺の重みを削減，ホールド辺に対しては辺の重みを増大させるような遅延最適化を行う．結果として新たにクリティカルサイクルが改善されたネットリストを得る．この手続きを有限回繰り返し，最後に現在の最適解を出力する．

3.1 一般同期性能を向上させる遅延最適化設定

論理合成ツールには任意のパスに対して制約を考慮しない設定が可能であり，また，制約条件に対して式 (3),(4) に示すような変数 m_{s}，m_{h} が用意されていると仮定する．

セットアップ制約

$$s(u) - s(v) \leq T - (d_{\max}(u,v) + setup(v)) - m_{\text{s}}(u,v) \quad (3)$$

ホールド制約

$$s(v) - s(u) \leq d_{\min}(u,v) - hold(v) - m_{\text{h}}(u,v) \quad (4)$$

提案手法での論理合成ツールにおける目標クロック周期 T_{target} と制約条件に関する設定方法を以下に示す．

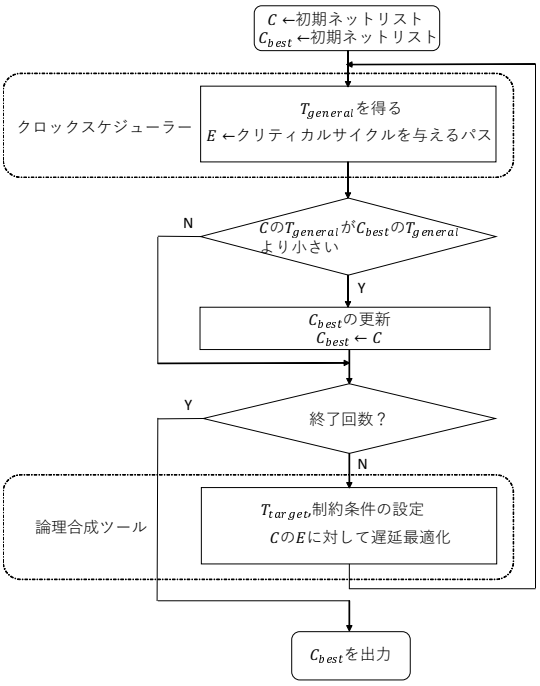

図 7: 提案手法

Step 1: クリティカルサイクルを除いたパスのセットアップ制約を取り払う.

Step 2: $T_{\max} \leftarrow \max_{(u,v)\in E_s} w_s(u,v)$ とする.

Step 3.1: ホールド辺を与えるパス $(u,v) \in E_{\rm h}$ に対し,$m_{\rm h}(u,v) \leftarrow \beta w_{\rm h}(u,v)$, $m_{\rm s}(u,v) \leftarrow -M_{\max}$ とする.

Step 3.2: セットアップ辺を与えるパス $(u,v) \in E_{\rm s}$ に対し,$m_{\rm s}(u,v) \leftarrow T_{\max} - w_{\rm s}(u,v)$ とする.

Step 4: $T_{\rm target} \leftarrow \alpha T_{\max}$ とする.

ここで, $E_{\rm s}, E_{\rm h}$ はそれぞれクリティカルパス上のセットアップ辺, ホールド辺の集合を表す. α, β はセットアップ辺, ホールド辺に対する遅延削減, 増加量の倍率を表しており, $0.0 \leqq \alpha \leqq 1.0$, $1.0 \leqq \beta$ である. また, $w_s(u,v) = d_{\max}(v,u) + setup(u)$, $w_{\rm h}(u,v) = d_{\min}(u,v) - hold(v)$ である. $M_{\max}$ には十分に大きな値を設定することでセットアップ制約を緩和することができる. これによりホールド辺を与えるパスの $w_{\rm h}(u,v)$ 増大によるセットアップ制約の違反を防ぐ.

以上の設定により, 論理合成ツールはクリティカルサイクルのセットアップ辺の与える辺の重みを $(1-\alpha)T_{\max}$ 減少させ, ホールド辺の与える辺の重みを β 倍にするような遅延最適化を実行する.

制約違反が全て取り除かれた場合, 達成できるクリティカルサイクル上でのクロック周期削減量 $\Delta T_{\rm critical}$ は遅延最適化前の $T_{\max}$, $w_{\rm h}(u,v)$ を用いて以下式で表される.

$$\Delta T_{\rm critical} \geqq (1-\alpha)T_{\max} + \frac{\sum_{(u,v)\in E_{\rm h}} (\beta-1) w_{\rm h}(u,v)}{|E_{\rm s}|} \qquad (5)$$

論理合成ツールの仕様により, 制約違反は必ずしも全て取り除かれるとは限らず, 式 (5) の実現は保証されない場合がある. また, クリティカルサイクル上の最小クロック周期削減に成功しても, 新たなクリティカルサイクルが出現する可能性がある. したがって, 最小クロック周期が改善できるとは限らない. しかし, 提案手法はこれを許容して解の探索を行う.

4. 評　　価

提案手法が一般同期方式の最小クロック周期削減を少ない面積のオーバーヘッドで実現することを示す. 有効性を確認するため, ベンチマーク回路への適用を行い, 比較対象として一般同期方式の最小クロック周期を考慮せずに完全同期方式での最小クロック周期を最適化するように論理合成した回路を用意する.

4.1 環 境 設 定

提案手法において, 遅延削減, 増加量を決めるパラメータをそれぞれ $\alpha = 0.9$, $\beta = 1.1$ とする. 繰り返し回数を 100 回と設定し, IWLS 2005 Benchmarks [5] に含まれる ISCAS Benchmarks の 30 回路に適用する. 提案手法を構成する論理合成ツールとして Synopsys Design Compiler(Version L-2016.03-SP4) を使用し, セルライブラリには ROHM 社の $0.18\mu m$ スタンダードセルライブラリを用いる. 初期解として面積最小のセルでマッピングされたネットリストを用いる. 一般同期方式での最小クロック周期は文献 [4] のアルゴリズムで得られた値を用いる. また, 完全同期方式での最小クロック周期は論理合成ツールで評価する. 回路面積については, どちらの方式も論理合成ツールで見積もる. 本評価では, 一般同期回路の最小クロック周期として, 任意のクロックスケジュールが設定可能である場合を仮定する. また, クロック合成による面積の増大を考慮しない.

4.2 結　　果

実験結果を表 1 に示す. $T_{\rm complete}$, $T_{\rm general}$ はそれぞれ完全同期方式, 一般同期方式での最小クロック周期を表している. Area は回路内に存在するセルの面積の総和を表している. また, 実験結果の $T_{\rm complete}$, $T_{\rm general}$ と Area はそれぞれ初期解で用いた回路の完全同期方式での最小クロック周期と Area で正規化されている.

$T_{\rm general}$ については, 30 回路のうち 14 回路で提案手法が比較対象より良い結果となった. $T_{\rm general}$ の平均では, 比較対象が 0.394, 提案手法が 0.416 となり, 提案手法がやや悪い結果となった. しかし, Area の平均では比較対象が 1.581, 提案手法が 1.417 となり, 提案手法が比較対象を上回った. 提案手法は完全同期方式を考慮していないため, すべての回路において $T_{\rm complete}$ が比較対象を上回った. また, 比較対象では 30 回路中 2 回路において Area, $T_{\rm general}$ 両方が提案手法を上回った. 対して, 提案手法では 30 回路中 4 回路において Area, $T_{\rm general}$ 両方が比較対象を上回った.

以上の結果から, 提案手法は一般同期方式のクロック周期削減に有効であると言える.

5. 関 連 研 究

一般同期性能を向上させる回路合成手法として, 与えられた回路に対して遅延を挿入することによって最小クロック周期を削減する手法 [6]〜[8], 記憶素子の再配置を行うことにより最小

クロック周期を削減する方法 [9],[10] が提案されている．また，一般同期方式を考慮した遅延最適化の実例として，セルの置き換えによる最小クロック周期・消費電力の改善が報告されている [3]．これらの手法はゲートのマッピング方法について考慮していない手法であり，提案手法とはアプローチが異なる．テクノロジマッピング手法として，整数計画法を用いた手法 [11] が挙げられる．提案手法は市販の論理合成ツールにより構成することができるため，遅延見積もりに対する誤差を抑えることができる．

6. おわりに

本報告では，任意のクロックタイミングを設定可能であるとの前提のもとで，一般同期方式においてクロック周期の下限を定めるクリティカルサイクルに着目し，最小クロック周期を小さくすることを目標とした回路遅延最適化手法を提案した．また，完全同期方式で高速に動作するような論理合成を施した場合と比較して，提案手法は，一般同期方式の最小クロック周期削減を少ない面積のオーバーヘッドで実現することを示した．

今後の課題として，ホールド辺に対する遅延挿入法の改善が挙げられる．提案手法での遅延挿入は簡易的であり，不要な遅延挿入が行われる可能性がある．これらは最小クロック周期の悪化と不要な回路規模の増大を招く．解決策として，遅延挿入に関して効率的な手法 [6]〜[8] が検討されており，これらを提案手法に取り入れることで，さらなる一般同期性能の向上が期待できる．

謝辞　本研究は東京大学大規模集積システム設計教育研究センターを通し，ローム株式会社，シノプシス株式会社の協力で行われたものである。

表 1: 実験結果

回路名	比較対象			提案手法		
	T_{general}	T_{complete}	Area	T_{general}	T_{complete}	Area
s27	0.692	0.692	1.519	0.737	0.933	1.139
s208_1	0.409	0.454	1.352	0.326	0.995	1.447
s298	0.387	0.447	1.395	0.374	0.630	1.493
s344	0.459	0.465	1.624	0.454	0.728	1.829
s349	0.462	0.482	1.629	0.432	0.650	1.698
s382	0.555	0.576	1.557	0.603	0.823	1.421
s386	0.483	0.486	1.951	0.543	0.840	1.745
s400	0.529	0.558	1.463	0.505	0.824	1.386
s420_1	0.291	0.358	1.553	0.224	1.046	1.625
s444	0.542	0.552	1.488	0.560	0.957	1.319
s510	0.392	0.392	2.462	0.523	0.635	1.332
s526	0.415	0.438	1.484	0.390	0.595	1.546
s526n	0.413	0.460	1.651	0.399	0.627	1.566
s641	0.357	0.377	2.157	0.392	0.606	1.585
s713	0.356	0.368	2.304	0.392	0.606	1.585
s820	0.364	0.364	2.217	0.447	0.487	1.458
s832	0.421	0.421	1.981	0.561	0.652	1.240
s838_1	0.193	0.218	1.502	0.148	0.660	1.614
s1196	0.260	0.337	1.574	0.256	0.561	1.710
s1238	0.288	0.358	1.560	0.276	0.562	1.686
s1423	0.201	0.207	1.785	0.240	0.398	1.761
s1488	0.298	0.298	1.791	0.420	0.457	1.288
s1494	0.306	0.307	1.714	0.441	0.489	1.302
s5378	0.343	0.396	1.134	0.361	0.724	1.173
s9234_1	0.385	0.400	1.400	0.381	0.676	1.259
s13207	0.415	0.435	1.112	0.533	0.735	1.022
s15850	0.540	0.815	0.961	0.405	0.823	1.138
s35932	0.389	0.391	0.938	0.456	0.921	1.009
s38417	0.311	0.336	1.103	0.405	0.650	1.072
s38584	0.373	0.412	1.078	0.296	0.595	1.063
Ave.	0.394	0.427	1.581	0.416	0.696	1.417

文　献

[1] J.P. Fishburn, “Clock skew optimization,” IEEE Transactions on Computers, vol.39, no.7, pp.945–951, 1990.

[2] A. Takahashi and Y. Kajitani, “Performance and reliability driven clock scheduling of sequential logic circuits,” Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC ’97, pp.37–42, 1997.

[3] 川口純樹，小平行秀，“一般同期方式における低電力化のためのテクノロジーマッピング，” 電気関係学会東北支部連合大会講演論文集，vol.2013，pp.33–33，2013.

[4] A. Takahashi, “Practical fast clock-schedule design algorithms,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol.89, no.4, pp.1005–1011, 2006.

[5] C. Albrecht, “Iwls 2005 benchmarks,” 2005 International Workshop on Logic Synthesis, pp. • • – • •, 2005.

[6] T. Yoda and A. Takahashi, “Clock period minimization of semi-synchronous circuits by gate-level delay insertion,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol.82, no.11, pp.2383–2389, 1999.

[7] Y. Kohira and A. Takahashi, “Clock period minimization method of semi-synchronous circuits by delay insertion,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol.88, no.4, pp.892–898, 2005.

[8] Y. Kohira, S. Tani, and A. Takahashi, “Minimization of delay insertion in clock period improvement in general-synchronous framework,” IEICE transactions on fundamentals of electronics, communications and computer sciences, vol.92, no.4, pp.1106–1114, apr 2009.

[9] Y. Kohira and A. Takahashi, “A fast gate-level register relocation method for circuit size reduction in general-synchronous framework,” IEICE Trans. Fundam. Electron. Commun. Comput. Sci., vol.E91-A, no.10, pp.3030–3037, Oct. 2008.

[10] Y. Kohira and A. Takahashi, “Gate-level register relocation in generalized synchronous framework for clock period minimization,” IEICE Transactions, vol.90-A, no.4, pp.800–807, 2007.

[11] J. Kawaguchi, H. Mashiko, and Y. Kohira, “Technology mapping method using integer linear programming for low power consumption and high performance in general-synchronous framework,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol.E99.A, no.7, pp.1366–1373, 2016.