

多段論理合成における時間最適化の一手法

三浦 順子 竹田 信弘 神戸 尚志

シャープ株式会社 コンピュータシステム研究所

論理設計の期間短縮のために、自動論理合成の技術が必要不可欠となっている。当社の論理合成システムLAURELでは、論理式・真理値表等を標準的な論理ゲートを用いて多段化した論理回路を局所変換することにより、所望の製造テクノロジーに応じた回路を作成する。自動合成する回路は、設計者の要求を満たす必要があり、特に遅延時間に対しては厳密さが要求される。本稿では、当社の論理合成システムLAURELのテクノロジー・マッピングにおける時間最適化の手法を紹介する。本手法は、ルールベースと論理式展開を併用することでクリティカルパス上のゲートの段数を削減し、時間最適化を図るものである。

A Method of Timing Optimization of Multi-level Logic Synthesis

Junko MIURA Nobuhiro TAKEDA Takashi KAMBE

Computer Systems Laboratories, SHARP Corporation,
2613-1, ICHINOMOTO-CHO, TENRI-SHI, NARA, 632 JAPAN

"LAUREL" is a logic synthesis system for combinational logic. This system synthesizes multi-level logic from two-level logic and optimizes area and timing for a target technology by local transformation. Timing optimization is applied to satisfy the specification of delay constraint of the circuit.

In this paper, we discuss about a timing optimization method in "LAUREL". We reduce the number of the gate on critical paths and optimize timing condition using rule-based approach and logic equation expansion.

1 はじめに

論理設計の期間短縮のために、自動論理合成の技術が必要不可欠となっている。自動合成により論理設計をおこなう場合には、生成される回路の特性を適切に制御することが重要である。

組み合わせ回路の自動合成では、標準的な論理ゲートを用いて多段化した論理回路を局所変換することにより所望の製造テクノロジーに応じた回路を作成するのが一般的である⁽¹⁾⁻⁽²⁾。

このように自動合成された論理回路は、設計者の要求する性能を満たす必要がある。特に、与えられた制約時間を満足していなければ正しい動作が保証されないため、制約時間を満たす論理合成が必要である。

当社の論理合成システム LAUREL⁽³⁾のテクノロジー・マッピングでは、多段化した論理回路をルールベースを用いた局所的な変換によって面積最適化した後、遅延時間の評価をおこない、制約時間に収まらないクリティカル・パス上の一部の回路を切り出して、論理式に置き換え展開することによってクリティカルパス上のゲートの段数を削減し、時間の最適化を図っている。

以下本稿では、2章において本システムの概要について述べ、3章では本システムのテクノロジー・マッピングの手法について説明し、4章では遅延時間の最適化手法について述べる。最後に実験の結果とその評価について述べる。

2 システム概要

論理合成システム LAUREL は、論理式・真理値表・論理回路を入力とし、特定のテクノロジーに変換して、与えられた面積・遅延時間の制約のもとで、最適化された論理回路を出力する。

本システムでは、組み合わせ回路の2段論理への展開、2段論理の単純化、多段論理合成、テクノロジー・マッピングの順に処理を進めている(図1)。

まず、入力された論理式・真理値表あるいは論理回路から組み合わせ回路の部分を取り出し積和形の論理式に変換する。次に PLA の簡単

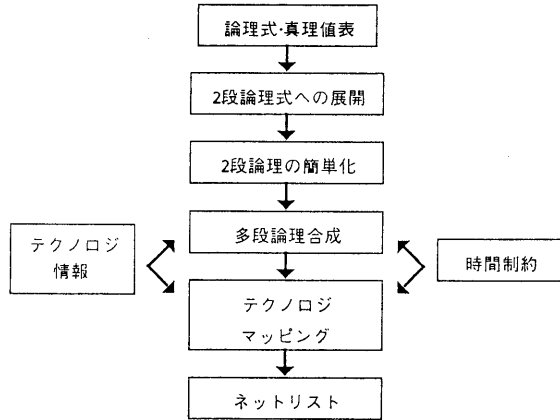


図1 論理合成システム

化で用いる論理単純化アルゴリズム⁽⁴⁾⁻⁽⁵⁾を用いて、冗長な部分を取り除き2段論理の単純化をおこなう。次に、複数の論理式で共通な部分をくりだし、中間変数で置き換えることにより多段の論理式を合成する。論理式から製造テクノロジーとは独立にAND/OR/NOTゲートからなる基本ゲートを用いて構成した初期回路を生成した後、テクノロジー・マッピングにより特定の製造テクノロジーへの変換および面積・時間の最適化をおこなう。

3 テクノロジー・マッピング

テクノロジー・マッピングでは多段論理合成で得られた多段論理式の特定の製造テクノロジーのセルへの割り付けと、最適化の処理をおこなう。

本システムは、知識を if _ then 形式で表現するルールベース手法を用いており、入力された回路に局所的な論理最適化のルールを繰り返し適用していくことで回路全体の最適化を図っている。

処理は、図2に示す手順でおこなう。まず、回路中の各基本ゲートを製造テクノロジーのセルに置き換える。論理ゲートの入力数がライブラリに用意されているセルの入力数を越える場合には、分割して複数のセルに割り当てる。次

に面積の最適化をおこなう。面積最適化のために以下のルール群を用意している。

- ①複合ゲートの割り付け
- ②AND/ORゲートから
NAND/NORゲートへの変換
- ③双対論理を利用した変換
- ④冗長なインバータの削除

①～③のルール群は制御ルールによって順に適用し、回路を変換することで面積最適化をおこなう。④のルール群は、適用可能な場合はただちに実行して回路に冗長な部分を残さないようにしている。

本システムでは、面積最適化後に遅延時間を評価して、あらかじめ指定されている遅延時間を越えるパスがある場合には指定時間内に収まるように時間の最適化を図る。遅延時間を決定する最も大きな要因は外部入力端子から外部出力端子までのゲートの段数であると考えられるため時間最適化ではクリティカルパス上のゲートの段数を減らすことに主眼をおいた処理をおこなう。

時間最適化のために以下の機能を実現した。

- ①ルールによる削減
 - a. ファンアウト調整ルール
 - b. クリティカルパス上のインバータの削減ルール
- ②論理式展開

①の処理は、ルールによる変換であり、②では、ルールによって選り出した部分回路を手続き型の処理で論理式に展開して段数を削減する方法である。まず、①のグループのルールを適用する。①の処理で指定の遅延時間が得られない場合にはクリティカルパス上の部分回路を切り出して②の論理式展開をおこなう。展開された部分回路は基本ゲートに再割り付けされて再び面積最適化の処理へ戻る。

指定の遅延時間が得られるまで①～②の処理を繰り返す。

ルールの構成を図3に示す。

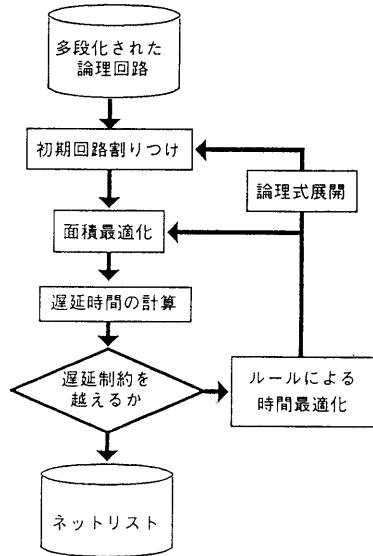


図2 処理フロー

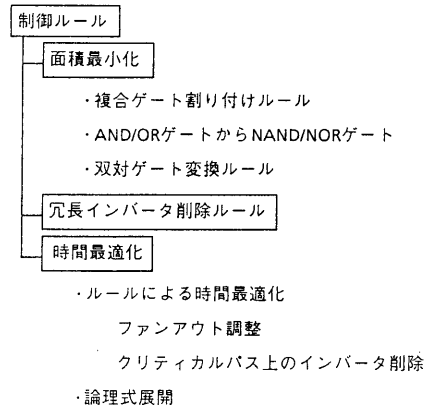


図3 ルールの構成

4 論理式展開による時間最適化

遅延時間の考慮は、多段論理合成の処理でもおこなわれているが、複合ゲートがある場合には論理式の段階で遅延時間をあらかじめ正確に見積もっておくことは非常に困難である。

また、テクノロジー・マッピングで遅延の評価をしながら面積の最適化を図る方法は、処理時間が膨大になり実際的ではない。

本システムでは、はじめに面積最適化をおこなってから時間制約を満たさない信号に対してのみ時間最適化の処理をおこなう。時間最適化のために、回路の一部を切り出して論理式に直した後、手続き型の処理を用いてその論理式中の中間変数を展開することによってクリティカルパス上のゲートの段数を削減する方法を用いた。

これは、局所的な変換で論理の組み換えによって段数削減をおこなうためには複雑なルールが必要となり、さらに適切な時間最適化をおこなうためには様々なセルの組み合わせを考慮した非常に多くのルールを用意しなければならないためである。

4.1 クリティカルパス

本システムでは、ブロックオリエンティッドアルゴリズム⁽⁶⁾を用いて遅延の評価をおこなっている。以下、簡単に概要を述べる。

入力端子に信号が到着する時刻と出力端子に要求される到着時刻はあらかじめ与えられているものとする。

各ゲートの遅延は、内部遅延と接続されている負荷から計算される。

外部入力端子から外部出力端子に向かって順に各ゲートへの到着時刻を計算していくが、そのゲートの入力端子の中で最も遅い信号の到着時刻をそのゲートへの信号の到着時刻として採用する。

同様に外部出力端子から外部入力端子側へ順に要求到着時刻を計算する。要求到着時刻から到着時刻を引いた値がスラックである。

外部出力端子の中でスラックの値が最小のものが最も遅延の大きいものであり、スラックの値が最も小さいものを入力端子側へ順にたどったパスがクリティカルパスである。

4.2 回路の選択方法

回路の一部を切り出して論理式展開によって段数を削減する方法は、展開する部分回路の選択方法が重要になる。

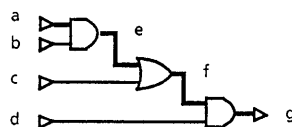
これは、展開処理が効果的におこなわれないときには、他のパスの遅延が必要以上に大きくなり、クリティカルパスが変わる場合があるためである。

本システムでは、クリティカル・パス上のゲートの中でスラックの値が最小のものとそのゲートに接続する前後数段の回路を選び出して、展開の処理へ渡している。スラックの値が小さいほどそのクリティカル・パス全体の遅延に大きな影響を与えると考えられるからである。

4.3 展開処理

論理式展開は、与えられた部分回路を論理式に変換し、クリティカルパス上の論理式を展開する処理であり、次の4段階の処理が順に実行される。

- (step1) 与えられた部分回路中の各ゲートを基本ゲートに分解する。
(複合ゲート処理)
- (step2) 分解された部分回路を多段論理式に変換する。(論理式変換)
- (step3) クリティカルパス上の信号に相当する中間変数を展開する。
(論理式の展開)
- (step4) 展開された論理式に基本ゲートを割り付け、部分回路を構成する。
(基本ゲート割り付け)



クリティカルパス:(a, e, f, g)

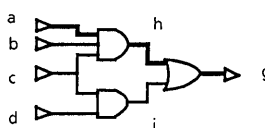
↓ 変換

$$\begin{aligned} g &= d * f \\ f &= c + e \\ e &= a * b \end{aligned}$$

↓ 展開

$$\begin{aligned} g &= d * (c + e) \\ &= d * (c + a * b) \\ &= c * d + a * b * d \end{aligned}$$

↓



論理式展開をおこなうことで、クリティカルパス以外のパスのゲート段数は増加するが、クリティカルパスのゲート段数は減少する。例えば図4では与えられた部分回路のクリティカルパス(b, e, f, g)のゲート段数は3であるが、展開処理後のクリティカルパス(d, h, j)のゲート段数は2に減っている。逆に、クリティカルパス以外のパス、例えばパス(d, g)は、ゲート段数が1段からパス(d, h, j)の2段に増加している。

テクノロジ・マッピングで生成される回路には多くの複合ゲートが含まれる。従って、展開処理においても複合ゲートを扱う必要がある。しかし、与えられた部分回路中の複合ゲートを直接展開すると論理式上ではクリティカルパス上にない部分も展開されるため展開後の回路が必要以上に大きくなってしまふ(図5(a))。そこで、複合ゲートを基本ゲートに分解した後でクリティカルパス上のゲートのみを展開することによる面積増加を防いでいる(図5(b))。また、各テクノロジに固有なゲートを一度基本ゲートに変換することによって、テクノロジが変わっても(step 2)以降の処理を変える必要がない。

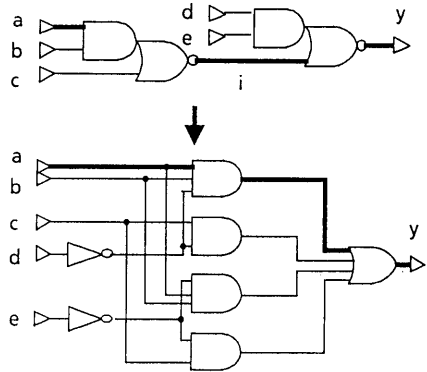
展開された論理式は基本ゲートに割り付けられ、テクノロジ・マッピング本体の処理へ戻される。

5 実験

3章で述べたテクノロジ・マッピングをベンチマーク回路に適用した。使用ライブラリは、MCNCのベンチマークLIBRARY2⁽⁷⁾である。

回路(9sym)に対して面積最適化を行った結果、面積が147ゲート、遅延が20.27nsecであったが、時間最適化後の回路は面積が150ゲート(2%増)、遅延は18.32nsec(10%減)となり、論理式展開によりクリティカルパス上のゲートの段数を削減することにより遅延時間が短縮された。面積は2入力NANDを1としたときの値である。

実験では、SUN3/60上で、ルールベースと手続き型の処理で実現されている。

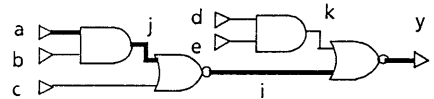


$$y = \neg(d * e + i)$$

$$i = \neg(a * b + c)$$

$$y = a * b * \neg d + a * b * \neg e + c * \neg d + c * \neg e$$

図5 (a)



$$y = \neg(k + i)$$

$$i = \neg(c + j)$$

$$j = a * b$$

$$y = a * b * \neg k + c * \neg k$$

kはクリティカルパス上にないため展開しない

図5 (b)

図5 複合ゲートの論理式展開

6 むすび

本稿では、論理合成システム LAUREL のテクノロジー・マッピングにおける時間最適化の手法を示した。

今後さらに評価を進め、有効性を確認していく予定である。

また、ルールベースではデータ量が増加すると処理時間が極度に遅くなるため大規模回路の分割の検討も進めていく。

参考文献

- [1] A. J. de Gues, et. al, "A Rule-Based System for Optimizing Combinational Logic", IEEE Design & Test, pp.22-32, Aug., 1984.
- [2] R. K. Brayton, et. al, "MIS: A Multiple-Level Logic Optimization System", IEEE Trans. CAD. Vol. CAD-6, pp.1062-1081, Nov., 1987.
- [3] 竹田他、「CMOS用多段論理合成」、信学技報、VLD88-83、1988.
- [4] 笹尾、「PLAの作り方、使い方」、日刊工業新聞社、1986.
- [5] R. K. Brayton, et. al, "Logic Minimization Algorithms for VLSI Synthesis", Kluwer Academic Publishers, 1984.
- [6] R. B. Hitchcock, Sr, et. al, "Timing Verification and the Timing Analysis Program", 19th DAC, 1982, pp.594-604.
- [7] R. Lisanke, "Logic Synthesis and Optimization Benchmarks User Guide Version 2.0", 1988.