Varchsyn(4): 演算器合成手法

6N−4

若林 一敏　藤田 友之　前田 直孝

NEC

1　はじめに

論理合成システムVarchsynにおける、VHDLで記述された任意ビットの加算やカウンタ、比較器、パリティ発生器等の演算器を制約条件に沿って合成する手法について述べる。

論理合成の急速な進歩によりランダムロジックの合成に関しては人手を越える回路の合成が可能となってきている。しかしながら、データパス系、特に演算器をロジックから論理合成で最適な回路を合成するのは依然困難な状況にある。即ち、演算器には動作速度や回路規模が異なる様々な回路構成のものがあるが、これらの構成をそれぞれ保持することなく一定の回路構造から論理構造変換で様々なタイプの演算を合成することは依然困難である。例えば、任意ビットのキャリールックアヘッド加算器からタイミング最適化等によってキャリールックアヘッド加算器並の高速加算器を合成することは困難である。また、ゲートアレイやスタンダードセル等のセミカスタムICでは、加算器やカウンタ、比較器、パリティ発生器等の汎用的な演算器に対して、4ビットや8ビットのハードマクロが提供されることが多く、これらの高機能セルをうまく利用することで回路の速度/面積を向上することができるが、従来の一般的な論理合成のテクノロジマッピング技術ではこれらのような高機能なセルをマッピングすることは困難である。

また、ビットスライスタイプのセルを用いたモジュールジェネレータによる演算器の合成システム[1, 2]も提案されているが、これらは合成制約に応じた柔軟な合成が困難である上、ビットスライスにならない演算器の合成には不向きであった。

2　既設計回路を利用した演算器合成

演算器の構成は過去数十年にわたって研究され、高速タイプのもの、低面積なもの等様々な優れた回路が提案されてきており、現在の論理合成の技術に頼った最適化を施すよりも、これらの優れた回路構成の中から適したものを選択してその回路を最適化の出発点とするのが現実的なアプローチと考えた。よって、本稿では、熟練設計者による既存の回路構成や、ハードマクロを演算器部品として利用して、様々な形式の演算器を面積や速度の制約に合わせて選択し、これらを組み合わせて任意のビット数の回路を合成する演算器合成手法の提案を目的とする。

RTレベルのライブラリを利用してより一般的な演算器を合成する手法としては、RTレベルの知的なライブラリの構築法[3]が提案されているが、これらは合成の一歩手前の検索手段を与えているのみである。更に、[4, 5]では、RTレベル部品を用いてルールベース処理により、様々な制約の演算器を合成する手法が提案されているが、演算器の合成ルール記述の一般化が不十分であり、演算器毎に詳細な合成ルールの記述が必要な他、ハードマクロの利用を考慮していない。

本稿では、演算器をできるだけ一般的に扱い演算器の再帰的な構造を入力するだけで様々な演算器を汎用に合成できる演算器合成手法を提案する。また、また、本提案手法は加算器のキャリーインの無い最下位ビットは全加算器より半加算器で構成したほうが有利であるとか、オペランドが定数の部分はハードウェアマクロよりゲートで構成した方が有利である等といった演算器の利用状況をも考慮した演算器合成を行なう。また、加算器や比較器といった汎用回路の他に設計者固有の演算器も、その回路構成を再帰的な構成を登録することにより任意ビットの演算器を合成可能である。

3　演算器合成手法

VHDL記述内の演算は、VHDL入力部[6]でサブモジュールとして回路データに展開される。演算器のタイプやビット幅等はVHDL入力部で決定されサブモジュールにセットされる。本システムはこのサブモジュールをハードウェアマクロまたは、テクノロジマッピング可能な論理式に展開する。例えば、VHDLで加算と減算が記述され、それらが演算器共有部[7]で共有された場合、加減算器のサブモジュールが合成され、本システムがその加減算器の詳細な構成を合成する。

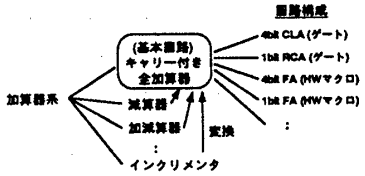

図1. 演算器の分類とテンプレートの階層

3.1　合成対象演算器の分類とテンプレート

本システムでは、演算器は二階層で分類される(図1)。例えば、加算器系という大分類の中に、通常の“a+b”という加算器の他、減算器、加減算器、キャリ付加算器、“a+b+1”、インクリメント等の小分類の回路がある。原則的として、各大分類毎に基本となる小分類回路(基本回路)を設定し、他の小分類回路は基本回路の変形で定義する。例えば、加算器系の基本回路はキャリー付き加算器とすると、減算器は基本回路のキャリーに1をいれ、一方の入力を反転することにより合成される。さらに、小分類各々の演算には様々なタイプの演算器が存在するが、各小分類毎に種々の演算タイプを登録するのはオーバーヘッドが大きいため、各基本回路に対してのみ様々なタイプの演算を登録する。例えば、加算器系の基本回路であるキャリー付きの加算器に対して高速タイプ(CLA等)や小面積タイプ(RCA等)の加算器の構成を登録することにより、減算器等同一大分類に属する他の演算器も各種タイプの演算構成での実現が可能となる。

各タイプの演算はその構成を表現するのに必要な最小限のビット幅のもののみ登録すればよいが、19ビット等の任意のビット長回路の登録も可能である。また、基本回路の1ビットの形式は必ず一種類以上登録が必要である。また、基本回路からの変形では実現できない回路形式の場合直接その回路構成を登録できる。例えば、減算器に対して、加算器と補数器を利用しない直接的な減算器を登録することもできる。

大分類としては、加算器系、比較器(逐次型、並列型)、シフタ/ローテート(右/左、ロード付き)、カウンタ(同期、非同期、アップ、ダウン、アップダウン、データロード付、リセット付)、パリティ発生器(奇数、偶数)、レジスタ、フィリップフロップ、デコーダー等がある。

[テンプレート]

以上の演算器のタイプとその構成方法は、「テンプレート」と呼ばれるテキストファイルに登録する。加算器のテンプレート間の階層の例を図1に示す。これらの階層や基本回路部品の組み上げ法は、3種類のテンプレートを利用して表現される。図の真中にある各小分類演算から基本変換への変換規則を示すのが**メタ変換テンプレート**、基本回路の回路構成を示すのが**基本テンプレート**である。基本構成は、ゲート回路とハードマクロによって示され、任意のビット長の演算を登録できる。基本テンプレートを組み合わせて多ビットの演算を構成する規則は**接続テンプレート**に登録する。例を図3に示す。

3.2　合成アルゴリズム

演算器合成の全体のフローを図2に示す。本節では、この中で特に破線で囲まれた基本回路の合成手法について述べる。多くの演算器は規則的な構造をしており、ビットスライスでは無い場合も、一定のビット幅の構成の再帰的繰り返しで表現できる。本手法は演算器のこの性質に着目し、「演算器はあるビット位置で再帰的に2分割可能であり、分割された部分回路は一

“Varchsyn(4): An Operator Synthesis Method”
Kazutoshi WAKABAYASHI, Tomoyuki FUJITA, Naotaka Maeda
NEC Corporation

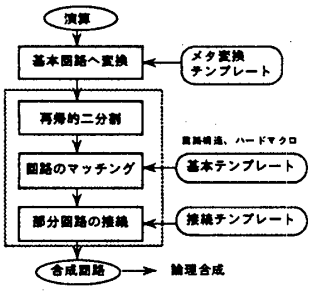

図 2. 演算器合成処理手順

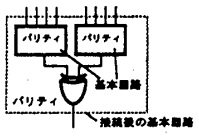

図 3. 接続テンプレートの例: パリティ発生器

定の接続ルールによって接続され、もとのビット数の回路を構成可能である。」という仮定の成り立つ演算器の合成を統一的に扱う。加算器やカウンタ、パリティ発生器、比較器等の一般的な回路はこの仮定を満足する。また、この仮定を満たす回路であれば、ユーザ固有の演算器もテンプレートを用意するだけで同様に合成可能である。図 3にパリティ発生器の再帰的定義の例を示す。

(1) **メタ変換**: 小分類の演算器名が基本回路でなければ、メタ変換テンプレートを用いて、「基本回路 + 付加回路」に変換する。

(2) **回路分割**: 基本回路を基本テンプレートに登録されたビット幅になるまで、再帰的に 2 分割する。一般に多ビットの基本テンプレートを用いる程良い回路が合成できるが、演算器のハードマクロは 2^n ビットとか $4n$ ビットのものが用意されることが多い。よって、できるだけ 2^n ビット、 $4n$ ビットの中で大きいビット数になるように 2 分割する。また、加算器等では最上位ビット、最下位ビットのみ例外的な構造をしているものには、最上位、最下位に余りの端数ビットがくるように分割する。

(3) **部分回路のマッピング**: 分割された回路にマッチするビット幅の基本テンプレートの中から、以下の基準によって最適なものを選択する。

- 設計者に指定された回路部品がある場合はそれを選択。
- 最終的な合成結果の遅延を予測し、その遅延が制約を満たすこと。
- 遅延制約を満たす中で最小面積となるもの。但し、面積の算定は以下の事項も考慮に入れる。
- 入力の一部が定数かどうか。入力の一部が定数の場合は、定数伝搬によりハードマクロよりゲート回路の方が面積的に有利になる場合がある。これは、必要な演算器より基本テンプレートのピンが多い時にもおこる。例えば、加算器の最下位のキャリーがない時や、データロードのないカウンタ等である。これらは、使用しないピンは 0 か 1 にクランプされるから定数入力と考えられる。

(4) **回路の接続**: (2) で分割された回路を接続テンプレートに従って再帰的に 2 つづつ接続し所望のビット長の演算器を得る。

入力が定数でない場合の 15 ビットの加算器で出力も 15 ビットのものを合成した例を図 4に示す。できるだけ大きいハードマクロを使用し、 MSB、 LSB ではゲートを利用して面積・遅延最小の回路を合成している。上の (2) から (4) の基本回路の再帰的な合成アルゴリズム OpeSyn() を以下に示す。

```
OpeSyn( ope ) {  /* ope: 合成したい演算 */
if (ope を合成可能な基本テンプレートがある)
then
   ope を最適な基本テンプレートを用いて合成;
else
  ope を分割規則に従って、上位と下位に分割;
  OpeSyn(上位);
  OpeSyn(下位);
  上位と下位を接続テンプレートに従って接続;
}
```

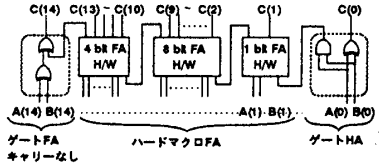

図 4. 15 ビット加算器の合成例

4 合成実験

提案手法で合成した演算器の性能を調べるため、 VHDL で記述された 14 ビットの加算器を NEC CMOS6 のゲートアレイを用いて合成した。基本テンプレートは、ゲート回路で構成した 1 ビットのリップルキャリー加算器と 4 ビットのキャリールックアヘッド加算器、また、 4 ビットと 1 ビットの全加算器のハードマクロのみを用いた。合成結果を図 5に示す。図で CLA と RCA にある矢印は論理最適化によって基本の CLA、 RCA に簡単な最適化を施した結果を示す。図で HW マクロの 4b、 1b は 4 ビット、 1 ビットのハードマクロを使用したことを示す。多ビットのハードマクロを利用する程優れた演算器を合成できる。ハードマクロを用いた加算器は CLA や RCA のゲート回路を最適化した場合に比べ面積・速度共優れており、ハードマクロを利用できる効果は非常に高い。また、図で 4bit 定数、 1bit 定数とは、片方のオペランドの 4 ビット、 1 ビットが定数であることを示し、この場合はその定数部分をゲートで合成することでハードマクロを用いた回路よりさらに面積・速度で優れた回路を合成できたことを示している。合成時間に関しても、従来の論理合成による最適化に比べ、高速に合成することができる。

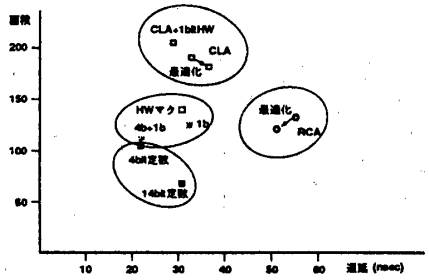

図 5. 14 ビットの加算器の合成結果

5 まとめ

制約に基づいて任意のビット数の最適なタイプの演算器を合成する演算器合成システムの合成手法とシステム構成について述べた。様々な演算器をビット幅に関して再帰的に定義することにより、種々の演算器を統一的に合成する手法を提案した。加算器に関する実験により、従来の論理合成の論理最適化とテクノロジマッピングのみによる方法に比べ、短時間でより制約にあった性能の良い回路が合成可能なことを示した。今後、演算器の最適なタイプの選択手法を機能合成との関連でより洗練していくことが課題である。

参考文献

[1] J.Rabaey, et al., "CATHEDRAL II: A synthesis system for multi-processor DSP systems", in Silicom Compilers(ed. D.D. Gajski), pp.311-360, MA:Addison-Wesley, 1988.
[2] R.Camposano, L.H.Trevillyan, "The integration of logic synthesis and high-level synthesis," ISCAS89, pp.744-747, 1989.
[3] G.D.Chen, D.D.Gajski, "An Intelligent Component Database For Behavioral Synthesis", 27th DAC, pp150-155 1990.
[4] N.D.Dutt, J.R.Kipps, "Bridging High-Level Synthesis to RTL Technology Libraries", 28th DAC, pp526-529, 1991.
[5] J.R.Kipps, D.D.Gajski, "Effects of Mixing Design Styles On The Synthesis of RTL Components", Technical Report91-42 ICS, UCI, 1991.
[6] 河原林政道他、「Varchsyn(2):VHDL からの合成」, 情処 46 回全大, 1993
[7] 伊藤義行他、「Varchsyn(3):VHDL 記述からの演算器シェアリング手法」, 情処 46 回全大, 1993