九州大学学術情報リポジトリ
Kyushu University Institutional Repository

高位合成技術の基礎

松永，裕介
九州大学大学院システム情報科学研究院

https://hdl.handle.net/2324/6069

出版情報：電子情報通信学会ソサイエティ大会講演論文集，2003-09. IEICE
バージョン：
権利関係：

高位合成技術の基礎

Basic techniques for high-level synthesis

松永裕介 [1]
Yusuke Matsunaga

九州大学大学院システム情報科学研究院
Graduate school of information science and electorical engineering, Kyushu University

1　はじめに

半導体の集積技術のたゆまない進歩によってLSI1チップに実装可能な回路規模は増加の一途をたどっている．一方，そのような大規模かつ複雑な回路を短期間で誤り無く設計するための技術として，高位合成技術が注目されている．高位合成とは回路の動作レベルの記述 (ハードウェア記述言語や C 言語等が用いられる) を入力として，回路の生成を行うものであり，設計の初期の段階でアーキテクチャや実現方式を決めるための重要な手段となっている．本稿ではこの高位合成技術の基礎の概説を行う．

2　高位合成の流れ

図 1 に高位合成の一般的な流れを示す．

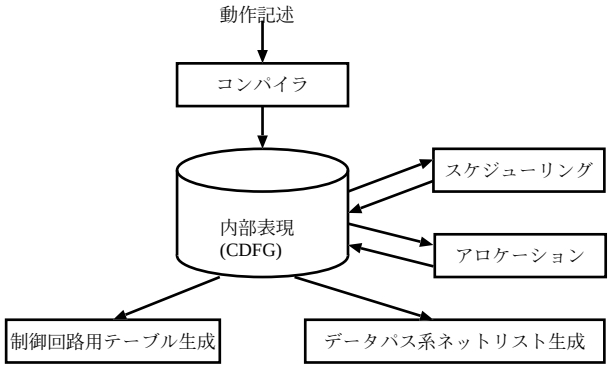

図 1　高位合成の流れ

まず，入力として与えられた動作記述を解析して内部表現に変換し，内部表現の最適化を行なう．この内部表現に対して，スケジューリングとアロケーションという 2 つの処理を行なう．スケジューリングとアロケーションの処理が終了すると，レジスタ，機能ユニット，マルチプレクサ等を回路要素としたデータパスのネットリストと，各クロックで実行する動作を制御する信号と順次実行させるための状態遷移の情報を示すテーブルが生成される．

こののち，制御回路部はランダムロジックやマイクロコードで実現され，データパスの構成要素は，それぞれ具体的な構造へと変換される．実際にはスケジューリングと制御回路合成，アロケーションとデータパス構成要素の実現とは互いに関連があり，それらを考慮に入れたスケジューリングやアロケーションも検討されている．

3　内部表現生成，最適化

内部表現としては，動作記述に現れたデータの流れ (データフロー) と各演算の実行順序の制御の流れ (コントロールフロー) を表した，コントロールデータフローグラフ (CDFG : Control Data Flow Graph) が用いられることが多い．CDFG の表わし方は後の処理のしやすさ等に影響を与えるため，さまざまなバリエーションが提案されている [3],[4]．図 2(a) の記述に対する CDFG の例を (b),(c) に示す．CDFG は，演算を表す節点 (◯)，制御を表す節点 (△, ▽)，および，それらの間の値や制御の流れを示す枝から構成される．(b) はコントロールフローとデータフローを分けた形でもつ手法 (disjoint CDFG) で，(c) は両方を統合した形でもつ手法 (hybrid CDFG) である．図中，△ の節点は条件分岐を表わし，x の値によって，その値がラベルづけされた枝に制御が流れる．▽ の節点は分岐からの結合，四角いブロックは条件分岐を含まない基本的なブロック (一連の代入文) を示している．(b) では，基本的なブロックごとにデータフローが表され，(c) ではデータフローの中に制御のための情報が埋め込まれた形になっている．

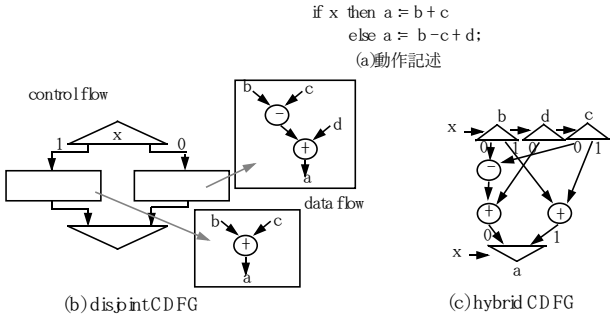

図 2　CDFG の例

入力記述から直接生成された CDFG から，言語の構文に依存した冗長性を取り除くため，CDFG 上でいくつかの変換処理が行なわれる．これには，定数伝搬，共通演算除去，デッドコード除去等のコンパイラの最適化でおこなわれるのと同様な処理と，ハードウェア記述に内在する並列性を引き出すためにフローグラフの構造を変換する処理等がある．

4　スケジューリング

スケジューリング問題とは，全体のハードウェア量や，総制御ステップ数などの時間制約を考慮しながら，動作記述中に現れる各演算を，具体的にどの制御ステップで実行するかを決定するものである．これは，演算を表し

た CDFG 中の各節点に対して，データの依存関係と与えられた制約をみたすように，制御ステップを割り当てることに対応する．

スケジューリングアルゴリズムはその制約の与え方によって，時間制約のもとでのスケジューリングと資源制約のもとでのスケジューリングに分類される．時間制約のもとでのスケジューリングでは，全体の制御ステップ数が制約として与えられ，その元でハードウェア量が小さくなるようにスケジューリングを行なうもので，DSP のようなリアルタイムシステムを対象とした設計によく用いられる．一方，資源制約のもとでのスケジューリングでは，使用できるハードウェア量の制約が与えられ，そのもとで制御ステップ数が最小になるようにスケジューリングを行なうもので，制御系の回路のスケジューリングに用いられることが多い．また，この両方を考慮に入れたスケジューリングも研究されている．

資源制約に基づくスケジューリングとしてはリストスケジューリングと呼ばれるアルゴリズムが提案されている．このスケジューリングでは，各演算に対して優先順位を計算し，各ステップで割り当て可能な演算をその優先順位にしたがって割り当てる．優先順位の基準としては，その節点を通るパスの終端節点まで距離，その演算をスケジュール可能なステップ荷関する自由度，等が提案されている [5].

一方，時間制約に基づくスケジューリングとしては，ヒューリスティックを用いて逐次割り当ていく方法 (Force-Directed Scheduling[6])，整数計画法を持ちいて解く方法 [7] 等がある．

5 アロケーション

データパスアロケーションとは，必要となるハードウェア量を考慮しながら，与えられた動作を実現するレジスタ転送レベルの回路構成を決定するものである．レジスタ転送レベルの回路は，機能ユニット (加算器等)，記憶ユニット (レジスタ等)，及び，通信ユニット (マルチプレクサ，バス等) から構成されるので，これら 3 種類の構成要素に対するアロケーションを行う必要がある．例として図 3(a) に示す CDFG に対するアロケーションを考える．同時に使用しない資源は共有できるため，2 つの加算に対して共通の加算器を割り当て，各変数に対しては 3 つのレジスタを (r1, r2, r3) を割り当てるとする．ここで，変数 a,d,e に r1，b に r2，c に r3 を割り当て，共有した加算器の入力の値を選択するためにマルチプレクサを割り当てると，(b) のような構成が決定される．ここで，d に r2 を割り当てると (c) のようになり，通信のコストが大きくなる (マルチプレクサ 2 つ)．したがって，資源の共有とそれによる通信コストの増加のトレードオフを考慮しながら，全体としてハードウェア量が小さくなるように割り当てていく必要がある．

アロケーションアルゴリズムには，データパスの回路要素を逐次割り当ててデータパスを構築する方法，3 つのアロケーション問題を，それぞれ独立にグラフ理論の問題に定式化して解く方法等がある．ただし，3 種のアロケーション問題は互いに関連しあっているため，それぞれのアロケーションを最適に行なうことができたとしても，全体として最適になるとはいえない．

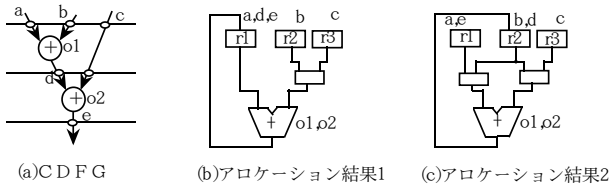

図 3 アロケーション

6 おわりに

高位合成に関する研究においても，個々のアルゴリズムの高度化に関する研究とともに，それらを統合したシステムの研究や試作，実用化に向けた検討が活発になってきており，実用化への動きは今後も続いていくと思われる．高位合成システムが広く受け入れられるためには，高位合成を有効に活用できるための環境の整備や，設計の方法論に関する研究も重要であろう．特に，高位レベルでの検証や性能見積もりの技術は不可欠であると思われる．

参考文献

[1] Daniel D. Gajski, Nikil D. Dutt, Allen C-H Wu, and Steve Y-L Lin, “HIGH-LEVEL SYNTHESIS Introduction to Chip and System Design”, Kluwer Academic Publisher, 1992.

[2] Michael C. McFarland, Alice C. Parker, and Raul Camposano, “The High-Level Synthesis of Digital Systems”, in Proceedings of the IEEE, vol.78, no.2, Feb. 1990

[3] A. Orailogulu and D. D. Gajski, “Flow Graph Representation,” Proceedings of the 23th Design Automation Conference, pp. 503-509, 1986.

[4] G. DeJong, “Data Flow Graphs: System Specification with the Most Unrestricted Semantics,” Proceedings of the European Design Automation Conference, 1991.

[5] M. C. McFarland, “Using Bottom-Up Design Techniques in the Synthesis of Digital Hardware from Abstract Behavioral Descriptions”, Proceedings of the 23th Design Automation Conference, 1986.

[6] P.G. Paulin and J.P. Knight, “Force-Directed Scheduling for the Behavioral Synthesis of ASIC’s”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol.8, no.6, pp.661-679, June 1989.

[7] J. Lee, Y. Hsu, and Y. Lin, “A New Integer Linear Programming Formulation for the Scheduling Problem in Data-Path Synthesis”, in Proceedings of the International Conference on Computer-Aided Design, pp.20-23, 1989.