論文の内容の要旨

論文題目　Instruction-set Extension and Code Generation for Application-specific Processors
（和　　訳　特定用途向けプロセッサのための命令セット拡張およびコード生成に関する研究）

氏　　名　瀬戸　謙修

半導体技術の進歩によって、より複雑な機能をハードウェアおよびソフトウェアとしてシステムLSI上に実装することが年々可能となってきている。このようなシステムLSIを短い納期内に、できるだけ少数のエンジニアで実現するには、設計生産性の向上が必要である。設計生産性の向上には、C言語をはじめとする高位記述からトップダウンでハードウェアおよびソフトウェアを合成することや、プログラム可能で変更容易なアーキテクチャを利用することが有効であると考えられている。本論文では、プロセッサをベースとしたプログラム可能なアーキテクチャに対して、与えられたアプリケーションプログラムの実行性能を向上するような命令セットを自動的に追加し、それらを使用した最適コードを生成する手法を提案する。

提案手法の流れを説明する。与えられたアプリケーションからプロファイリングを利用して性能ボトルネック部分を抽出する。ボトルネック部分をデーターフローグラフ（DFG）として表現し、その部分グラフを拡張命令候補として列挙する。各拡張命令候補に対して、ハードウェアで実現した際の面積など最適化に必要な情報も求めておく。拡張命令候補とプロセッサがもともと持っている命令を使用して、最短サイクルでDFGを計算するようなコードが生成される。なお、拡張命令の個数、拡張命令セットを実現するハードウェアの面積、コード量などに関する制約を課すことも可能である。

本論文では命令セット拡張およびコード生成問題を有限状態機械（FSM）を利用して定式化した。定式化ではマッチとよばれる概念を使用する。DFGの部分グラフgが、命令（あるいは拡張命令候補、以後これらを区別しない）iによって計算可能なとき、その対応関係（g, i）をマッチと呼ぶ。簡単のため命令iは1サイクルで実行されるものと仮定する。部分グラフgを計算するために命令iを実行することを、マッチ（g, i）を実行すると呼ぶ。提案手法では、与えられたDFGと命令セット、拡張命令候補からマッチをすべて列挙し、DFGを最後まで計算するようなマッチの最短実行系列を求める。そのような系列から最短サイクルのコードを抽出可能である。

定式化したFSMの主要部分を説明する。FSMの入力変数として、各マッチmごとに二値変数e_mを用意する。二値変数e_mは、あるサイクルでマッチmを実行するときに1、実行しない場合に0となる変数である。FSMの状態変数として、DFGの各ノードnおよび

各レジスタ l に対して、二値変数 $a_{n,l}$ を用意する。二値変数 $a_{n,l}$ はノード n の計算結果がレジスタ l に格納されているときに１、格納されていないときに０となる変数である。もう一つの状態変数として、各拡張命令候補 c に対して二値変数 u_c を用意する。二値変数 u_c はあるサイクルまでに拡張命令候補 c が実行されたときに１、一度も実行されなかった場合に０となる変数である。他にも入力変数および状態変数が定義されるが、ここでは説明を省略する。FSM の初期状態として DFG の計算前の状態、最終状態として DFG の計算後の状態を設定する。このとき、初期状態から最終状態へ到達するような最短サイクルの入力系列（マッチの実行系列）が最短サイクルのコードを表す。FSM の状態遷移の際には、リソース、データ依存関係、拡張命令の面積やコード量に関する制約を課し、制約を違反するような状態遷移を抑えることができる。

以上のような考え方でコード生成の３つの基本ステップ、すなわち、コード選択、レジスタアロケーション、スケジューリングだけでなく、ソフトウェアパイプライン、SIMD 命令の利用などをすべて同時に考慮した FSM を生成する手法を提案した。FSM を網羅的に解析することで、これらを同時に考慮したうえでの最適解を求めることが可能である。

形式的検証の分野では最近 FSM の解析に充足可能性判定（SAT）がよく利用されている。充足可能性判定（SAT）問題とは、与えられた論理式の値を１にするような変数値の組み合わせを求める問題であり、最近 SAT 問題を高速に解く技術が急速に発達した。FSM の状態遷移関数を一定サイクル分だけ展開し、生成された組み合わせ回路を SAT 問題に変換して解析することで、そのサイクル内で FSM がある初期状態からある最終状態に到達可能か網羅的に調べることができる。本論文ではその技術と、FSM によるコード生成の定式化を組み合わせ、DSP（Digital signal processor）向けのコード生成を行った。数十ノード以内の DFG に対して提案手法を適用したところ、人手で最適化されたコードと同等かそれより良いコードが生成された。

命令セット拡張を ASIC（特定用途向け集積回路）のような専用回路として実現する場合、高い性能や小面積、低費電力といった利点が得られるが、柔軟性に乏しいという欠点がある。本論文では柔軟性が高い命令セット拡張を実現するため、命令セット拡張をリコンフィギュラブルデータパスにマッピングする手法を提案した。リコンフィギュラブルデータパスとは、ALU や乗算器、レジスタなどの機能ユニットが配列状に並んだ構造をとり、機能ユニットの機能や、それらの間の接続をコンフィギュレーションによって変更可能である。提案手法の入力は、DFG およびリコンフィギュラブルデータパスのアーキテクチャであり、与えられた DFG をリコンフィギュラブルデータパスで実行するようなコンフィギュレーションを自動生成する。DFG の配置、概略配線、詳細配線をすべて同時に考慮した充足可能性判定問題の定式化を示し、実験を行った。その結果２０ノード程度までの問題であれば最適にマッピングできることが判明した。