Performance and area optimization methods in compiler for a dynamically reconfigurable processor

August 2011

TOI, Takao

主　論　文　要　旨

報告番号	甲）乙　第　　号	氏　名	戸井　崇雄

主 論 文 題 目 :

Performance and area optimization methods in compiler for a dynamically reconfigurable processor

（動的再構成プロセッサ向けコンパイラにおける性能と面積の最適化）

（内容の要旨）

本論文は限定された面積の中で最大性能を発揮するための動的再構成プロセッサ（DRP）向けコンパイラにおける最適化手法を示す。DRP のアーキテクチャでは、多数のプロセッシングエレメント（PE）と自由に接続できるプログラマブル配線からデータパスが構成される。「コンテキスト」と呼ぶこのデータパスを、状態遷移コントローラ（STC）が動的に切り替えて動作することにより、並列化しやすい部分と逐次的な部分の両方を併せ持つ多くのアプリケーションを時空間上で効率的に動かすことができる。このアーキテクチャの長所と短所についてまとめた上で、C 記述からのコンパイルフローについて述べる。このフローでは、動作合成、テクノロジーマッパー、そして配置配線という順番で合成する。特に動作合成は C 記述から高い並列性を持つデータパスと状態遷移マシンを合成する要の部分となる。

次に、動作合成における DRP 向けの３つの新しい最適化手法とその実験結果を示す。一番目は、複数の制御ステップをコンテキストにまとめて PE の使用効率を高める。二番目は、DRP が持つ時空間的な効率性を生かす、ループをパイプライン化するモジュロスケジューリングについて述べる。STC を利用してプロローグとエピローグ向けにパイプラインステージを制御した。三番目は、指定した動作周波数で動かすためのスケジューリング方法について述べる。特に配線スイッチが遅い DRP の短所を逆に活用する。JPEG ベースの画像復号化処理を例として３つの最適化方法を確認した。まずコンテキストの使用数が半分以下になり、さらに演算器の使用効率は 2.5 倍以上に向上した。またパイプライン化によって、PE 数は 2.2 倍の増加に対して、3.6 倍のスループット上の性能向上を確認した。高い遅延制御性により、動作周波数を短期間に最大化した。

最後に、配線混雑を考慮した動作合成から配置配線までの２つの反復合成手法を示す。PE を接続するプログラマブル配線は、複雑なデータパスを構成できる反面、配線の混雑時に遅延が伸びる。一番目の手法では、各コンテキストの配線遅延を動作合成にフィードバックして混雑を分散させる。遅延制約を越えるアプリケーションに適用したところ、クリティカルパス遅延は約 21%短縮した。二番目の手法は、配線の混雑度を考慮して配線遅延を推定して、反復収束までの時間を約 1/3 に短縮する。その結果、遅延短縮効果は平均 17%であり、一番目の手法と比べて 4%の増加で済んだ。

SUMMARY OF Ph.D. DISSERTATION

School Science for Open and Environmental Systems	Student Identification Number	SURNAME, First name TOI, Takao

Title

Performance and area optimization methods in compiler for a dynamically reconfigurable processor

Abstract

This thesis presents compiler's optimization methods to maximize performance in a limited area for a dynamically reconfigurable processor (DRP). At first, the DRP architecture is introduced. It is suitable for both control-intensive and parallelizable parts of an application program since it has a distinctive state transition controller (STC) that switches "contexts" consisting of many processing elements (PEs) and programmable wires. Characteristics of the architecture, each of which has pros and cons, are clearly stated. The compiler from C language consists of a high-level synthesizer (HLS), technology mapping tool, and place and route (place&route) tool. The HLS is the most important tool which extracts both highly parallelized data-paths and finite state machine.

Then, I propose three new optimization techniques in HLS for the DRP. First, several control-steps are combined as a context to maximize PE utilization ratio. Second, modulo scheduling algorithm is described for a loop pipelining, considering both spatial and time efficiencies of the DRP. The STC is used to control pipeline stages for prologue and epilogue. Third, a scheduling technique to improve controllability of clock frequency is explained. It utilizes a disadvantage of the architecture, such as long delay in a routing switch. I have evaluated techniques using a JPEG-based image decoder as one of examples. Experimental results show that the number of contexts is reduced to less than half. The PE usage rate becomes more than 2.5 times higher. Despite an overall increase in performance on pipelining of 3.6 times than that without pipelining, the number of operational units increased by a factor of 2.2. The clock frequency is maximized with accurate delay controllability.

At last, two iterative synthesis methods from HLS to place&route tool for making aware of wire congestion are presented. Although complex data-paths can be synthesized with the programmable wire, its delay is long where wires are congested. First, I feed back wire delays for each context to a scheduler in the HLS. Experimental results showed that a critical-path delay was shortened 21% on average for applications with timing closure problems. Second, the routing is skipped. Wire delays are estimated based on their congestions. The synthesis time was shortened to 1/3 causing delay improvement rate degradation at four points on average compared to the first method.