汎用高位合成系を用いたバイナリ合成における レジスタジャンプの効率的な実装

Efficient Implementation of Register Jumps in Binary Synthesis Using General-Purpose High-Level Synthesizer

岸本 匠　石浦 菜岐佐　中道 凌
Sho Kishimoto　Nagisa Ishiura　Ryo Nakamichi

関西学院大学
Kwansei Gakuin University

1　はじめに

バイナリ合成 [1] は機械語プログラムからレジスタ転送レベルのハードウェア設計を自動生成する技術であり, 一部または全部がアセンブリやインラインアセンブリで書かれたプログラムからもハードウェア合成が可能である. 文献 [2] は汎用高位合成系を利用したバイナリ合成の容易な実装手法を提案しているが, レジスタジャンプ命令で回路規模やサイクル数が増大するという課題があった. 本稿では, これを解決するレジスタジャンプ命令の効率的な実装法を提案する.

2　汎用高位合成系を用いたバイナリ合成

文献 [2] のバイナリ合成法は, 機械語プログラムを等価な C プログラムに変換し, これを汎用の高位合成系でハードウェア化するというものである. プログラムの変換例を図 1 に示す. 例えば RISC-V 命令の “`addi a1,a4,0`” は C 言語の “`a1 = a4 + 0;`” という文に変換する. レジスタの値をアドレスと解釈してジャンプするレジスタジャンプ命令の分岐先は実行時の値に依存するため, 可能な分岐先を前処理で列挙し, 条件文によって分岐先を選択するようにする. しかし, 分岐先が増えると回路規模やサイクル数が増大してしまう.

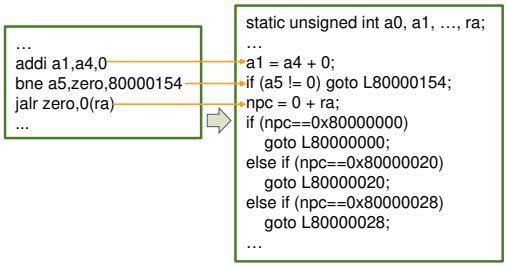

図 1: 機械語から C 言語への変換例

3　レジスタジャンプの効率的な実装

本稿では, レジスタジャンプ命令をハッシュ関数と switch 文で実装するとともに, 同じ処理を一箇所に集約することにより, 回路規模とサイクル数の抑制を図る.

図 2 (b) のように, ハッシュ関数でアドレスを小さな値に変換し, 分岐を switch 文で記述する. ハッシュ関数が衝突した場合は, 同一ケース内で if 文で分岐させる.

プログラム中のレジスタジャンプ命令は全て同じ文に変換されるので, 図 2 (c) のように, 分岐処理は “regjump” というラベルを付した箇所に記述し, 全てのレジスタジャンプ命令はアドレスの計算と “regjump” へのジャンプに変換する.

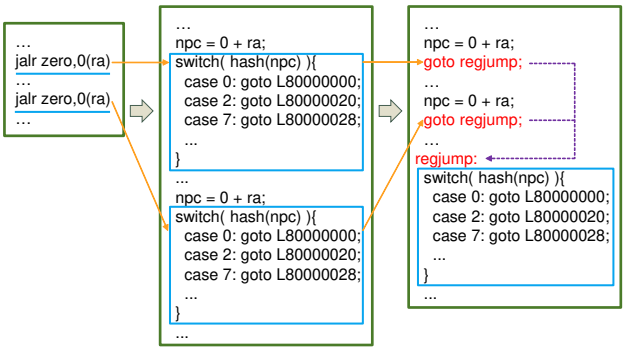

図 2: switch 文の利用および処理の集約

4　実験結果

文献 [2] の RISC-V を対象としたバイナリ合成系に提案手法を追加実装し, 実験を行った結果を表 1 に示す. “DFS” はグラフの深さ優先探索, “fibonacci” はフィボナッチ数列の生成, “heap-sort” は整数配列のヒープソートを行うプログラムである. “jalr” 列の “数” と “分岐” はそれぞれレジスタジャンプ命令の命令数と分岐数, “LUT” は回路規模, “Cycle” は実行サイクル数を示す. 提案手法により回路規模とサイクル数を削減できた.

表 1: 合成結果

	jalr		文献 [2]		提案手法	
	数	分岐	LUT	Cycle	LUT	Cycle
DFS	3	19	20,907	812	16,275 (0.78)	806 (0.99)
fibonacci	4	9	4,201	36	2,685 (0.64)	23 (0.64)
heap-sort	6	18	12,876	19,743	7,361 (0.57)	19,522 (0.99)

High-Level Synthesizer: Xilinx Vivado (2020.1)
Target: Xilinx Artix-7 (xc7a100tcsg324-1)

5　むすび

本稿では, 汎用高位合成系を用いたバイナリ合成におけるレジスタジャンプの効率的な実装法を提案した. メモリアクセスの効率化が今後の課題である.

謝辞 本研究は一部 JSPS 科研費 19H04081 の助成による.

参考文献

[1] G. Stitt and F. Vahid: “Binary synthesis,” *ACM TODAES*, vol. 12, pp. 1–30 (Aug. 2007).

[2] 中道, 石浦, 近藤: “汎用高位合成系をバックエンドとする RISC-V 機械語からのバイナリ合成,” 情処 DA シンポジウム, 6A-1 (Sept. 2021).