1A-02

RTL で設計可能な FPGA 回路のための CAD 開発

豊嶋　真帆†　　佐藤　友暁†

北星学園大学経済学部経営情報学科†

1.　はじめに

FPGA (Field-Programmable Gate Array)は機械学習[1]やネットワーク機器[2]など様々な用途に使用されている。FPGA 上の回路は回路設計者がハードウェア記述言語(HDL: Hardware Description Language)を使用し RTL (Register-Transfer Level)で設計できる。HDL で設計された回路を論理合成し、その結果得られた回路情報を FPGA 上の回路を制御するためのメモリに記録することで、設計者の意図した回路は動作する。

FPGA を使用した処理は CPU を使った処理と比較して高速かつ低消費電力である特徴を有する。この理由は FPGA を使った処理は CPU を使った処理と比較してオーバヘッドが非常に少なく、大量データの並列処理が可能である。加えてパイプライン処理によってスループットの向上を図ることができる。また、FPGA が ASIC (Application Specific Integrated Circuit)より優れている点は、回路の再構成を行うことができることである。

一方、FPGA 上の回路は ASIC 上の回路よりも動作周波数、消費電力、面積の点で劣る問題を有する。この問題は、FPGA 上の回路を再構成可能にするための回路を必要とするために生じる。加えて、従来の FPGA 上の回路の配線制御はトランジスタレベルで行われているため、FPGA 本体の回路開発は RTL で実現できなかった。

これらの問題を解決するために著者らは RTL で設計可能な FPGA の開発を行ってきた[3]。この FPGA は RTL で設計することができるため、ASIC と FPGA の協調設計も可能である[4]。この協調設計によって FPGA が ASIC よりも劣る問題を軽減することができる。この FPGA 上の回路の開発を容易にするために、CAD の開発が不可欠である。本研究では、この FPGA を構成する CB(Connection Block)の CAD ツールアルゴリズムを提案し検証を行う。

Development of a CAD Tool for FPGA Circuits That Can Be Designed with RTL

†Maho Toyoshima and Tomoaki Sato, Dept. of Management Information, Hokusei Gakuen University

2.　RTL で設計可能な FPGA

従来の FPGA は配線の制御にトランスファーゲートが使用されている。したがって、FPGA の開発にはトランジスタレベルでの開発が不可欠である。そこで、トランスファゲートの代わりに RTL で開発が可能なセレクタを使用した FPGA を提案してきた[3]。

この著者らが提案してきた FPGA を図１に示す。従来の FPGA と同様に LB (Logic Block) 、CB (Connection Block) 、SB (switch block)で構成されている。LB は３入力１出力の LUT とフリップフロップで構成されている。

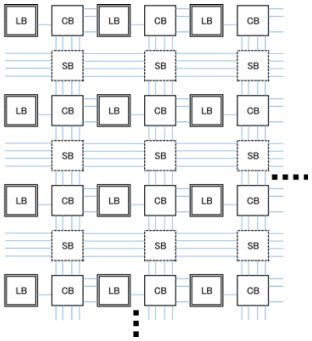

図１　RTL で設計が可能な FPGA アーキテクチャ

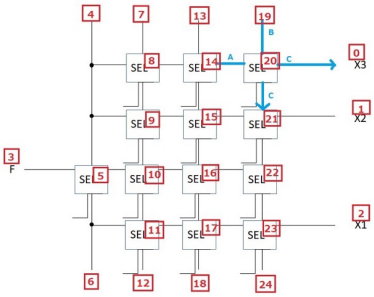

図２　CB の構成と信号の流れ

図２に示す CB と SB はセレクタで構成されている。このため、従来の FPGA と異なる経路の選択や制限が生じる。

3.　CB の CAD ツールアルゴリズム

図２の CB の入出力とセレクタの関係を(1)式に定義する。F は接続されていない状況を表し、F の値は回路規模で変化させる。１は接続されている関係を表す。また、論理合成ツールで得られた遅延時間に置き換えることも可能である。

Copyright ©2023 Information Processing Society of Japan.
All Rights Reserved.

$$
A = \begin{pmatrix}
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & 1 & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & 1 & F & F & 1 & 1 & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & 1 & F & F & F & 1 & 1 & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & 1 & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F & 1 & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F & F \\
1 & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F & F \\
F & 1 & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F & F \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 & F \\
F & F & 1 & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & 1 \\
F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F & F
\end{pmatrix} \tag{1}
$$

CB の経路を求めるアルゴリズムは以下の通りである。

- (1)式を２次元配列で定義
- ダイクストラ法を用いて最短距離求める
- 経路を記録し最短ルートを求める
- 1 度通ったノードを記録
- 1 度通ったノードを除いて再度最短ルートを求める

4.　検証

本研究で提案したアルゴリズムを検証することを目的として、Python によってアルゴリズムを実装した。開発環境を表 1 に示す。その実行結果を図 3 に示す。

表１：開発環境

OS	Windows 10 Home
CPU	Intel　Core i3-5005U 2.00　GHz
Memory	4.00　GB
開発言語	Python 3.9.12
ソースコードエディタ	Visual Studio Code 1.74.1

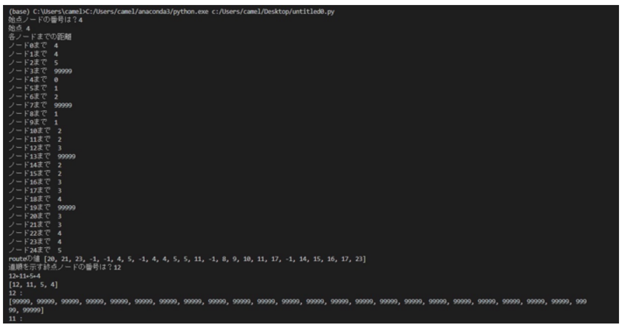

図３　実行結果

5.　まとめ

本研究では、RTL で設計可能な FPGA の回路のための CAD 開発を目的として、CB の経路や遅延時間を求めることが可能なアルゴリズムの提案とこのアルゴリズムの検証を行った。検証の結果、提案アルゴリズムの正当性が確認された。今後の研究においては、FPGA 全体の配線経路アルゴリズムの開発を行う。

謝辞

本研究は、東京大学ＶＤＥＣ活動を通して、日本シノプシス合同会社の協力で行われたものである．また、JSPS 科研費 19K11969 及び JSPS 科研費　22K12032 の助成により実施した．

参考文献

[1] C. Yang, Y. Meng, K. Huo, J. Xi and K. Mei, "A Sparse CNN Accelerator for Eliminating Redundant Computations in Intra- and Inter-Convolutional/Pooling Layers," *IEEE Trans. VLSI Sys.*, vol. 30, no. 12, pp. 1902 - 1915, 2022.

[2] V. Krishnan, O. Serres and M. Blocksome, "Configurable Network Protocol Accelerator (COPA)," *IEEE Micro,* vol. 41, no. 1, pp. 8-14, 2021.

[3] T. Sato, S. Chivapreecha and P. Moungnoul, “Fine-tuning of wave-pipelines on FPGAs developed by the RTL design,” *in Proc.　ECTI-CON 2015*, pp.1230.1-1230.6, 2015.

[4] T. Sato, S. Chivapreecha, P. Moungnoul and K. Higuchi, "An FPGA Architecture for ASIC-FPGA Co-Design to Streamline Processing of IDSs," *in Proc. CTS 2016,* pp.412-417, 2016.

Copyright ©2023 Information Processing Society of Japan.
All Rights Reserved.