HIGH LEVEL DATAFLOW NETWORK PARTITIONING USING STOCHASTIC ALGORITHMS

WOO YIT WENG

UNIVERSITI TEKNOLOGI MALAYSIA

Replace this page with form PSZ 19:16 (Pind. 1/07), which can be obtained from SPS or your faculty.

*Replace this page with the Cooperation Declaration form, which can be obtained from SPS or your faculty. This page is **OPTIONAL** when your research is done in collaboration with other institutions that requires their consent to publish the finding in this document.]*

HIGH LEVEL DATAFLOW NETWORK PARTITIONING USING STOCHASTIC ALGORITHMS

WOO YIT WENG

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Engineering (Computer and Microelectronic System)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

JUNE 2018

ACKNOWLEDGEMENT

Throughout the duration of conducting my final year project, I have been blessed with assistance from multiple parties. There were many obstacles that needed to be overcome and without the sound advice from these parties, this project would not have come to fruition.

First of all, I would like to express my utmost gratitude to my project supervisor, Dr. Ab Al-Hadi bin Ab Rahman, as a source of guidance and motivation while conducting this project. By generously sacrificing his time, his efforts had stimulated me to find solutions to problems that arose. The suggestions and encouragement that was provided truly made a difference in order for this project to achieve completion. Sincere appreciation is also given to members of the examiners for their constructive comments in order to improve this project even further.

My appreciation also goes to my family members and friends that have provided encouragements throughout the whole journey. Without their direct or indirect support, this project would not have progressed as smoothly. Sincere gratitude is given to all others that have helped me throughout. Apologies to those that have given me assistance but have not been acknowledged by name. Just know that your help and support is greatly appreciated all the same.

ABSTRACT

A dataflow actor network is a method of representing a design, showing clearly how data moves from one actor to another in graph form, suitable to represent designs such as a video streaming application. The design representation is written in the CAL Actor Language and the intent is to eventually implement the design in hardware, more specifically, Field Programmable Gate Arrays (FPGA). Instead of using a large FPGA to fit the entire design, the design is seperated into smaller blocks to be implemented in multiple smaller FPGAs. This has multiple advantages such as savings in cost and time as well as allowing more flexibility according to the design need and available resources. The caveat of this design approach is that the connections between FPGAs would incur some latency and noise. As such, the actors in the design need to be partitioned accordingly to minimize these inter-FPGA connections. This project will be investigating two partitioning algorithms, namely the Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) algorithms, to see which of these stochastic algorithms is better at partitioning the target design. Traditionally, partitioning is done using the cut cost as the optimized metric. While this would lead to less physical wires going across FPGAs, this could result in critical connections being compromised as it needs to traverse FPGAs. As such, this project will also investigate the feasibility of using communication rate as the partitioning criterion to better ensure that the connections between FPGAs are not critical such that the penalty can be tolerated. This project will use the profiles of a basic FIR Digital Filter as well as larger HEVC Decoder and MPEG-4 AVC Decoder test cases. The partitioning algorithms will be written in Java, using information regarding the actors and connections that are in the profile of each design. The results are analyzed to determine which algorithm is more suited to separate the design into balanced partitions as well as whether communication rate is a better partitioning criterion than cut cost for certain applications. The results obtained will also be compared with results obtained using the deteministic Fiduccia-Mattheyses (FM) algorithm.

ABSTRAK

Rangkaian pelakon aliran data adalah kaedah mewakili reka bentuk, menunjukkan dengan jelas bagaimana data bergerak dari satu pelakon ke yang lain dalam bentuk grafik, sesuai untuk mewakili reka bentuk seperti aplikasi streaming video. Reka bentuk ditulis dalam CAL dan tujuannya adalah untuk melaksanakan reka bentuk dalam perkakasan, lebih khusus, FPGA. Selain menggunakan FPGA yang besar untuk keseluruhan reka bentuk, reka bentuk dibahagikan kepada blok yang akan dilaksanakan dalam beberapa FPGA yang lebih kecil. Ini mempunyai banyak kelebihan seperti penjimatan kos dan masa serta memberikan fleksibiliti mengikut keperluan dan sumber yang ada. Kaveat pendekatan ini adalah bahawa sambungan antara FPGA akan menimbulkan latensi dan bunyi gangguan. Oleh itu, pelakon dalam reka bentuk perlu dibahagikan sewajarnya untuk mengurangkan sambungan antara FPGA. Projek ini akan menyiasat dua algoritma pembahagian, iaitu PSO dan ACO, untuk melihat algoritma stokastik mana yang lebih sesuai untuk membahagikan reka bentuk. Secara tradisional, pembahagian dilakukan dengan menggunakan kos potong sebagai metrik yang dioptimumkan. Walaupun ini akan mengurangkan wayar fizikal yang merentasi FPGA, ini boleh mengakibatkan sambungan kritikal dikompromi kerana ia perlu melintasi FPGA. Oleh itu, projek ini juga akan mengkaji penggunaan kadar komunikasi sebagai kriteria pembahagian untuk memastikan bahawa sambungan antara FPGA adalah tidak kritikal supaya penalti boleh diterima. Projek ini menggunakan profil penapis Digital FIR serta penyahkod HEVC dan penyahkod MPEG-4 AVC. Algoritma ditulis dalam Java, menggunakan maklumat pelakon dan sambungan yang ada dalam profil setiap reka bentuk. Datanya dianalisis untuk menentukan algoritma mana yang lebih sesuai untuk membahagikan reka bentuk ke dalam partition yang seimbang serta sama ada kadar komunikasi adalah kriteria pembahagian yang lebih baik untuk aplikasi tertentu. Data yang diperoleh juga dibandingkan dengan data yang diperoleh menggunakan algoritma deterministik FM.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACO	-	Ant Colony Optimization
ASIC	-	Application Specific Integrated Circuit
AVC	-	Advanced Video Coding
CAL	-	CAL Actor Language
CPU	-	Central Processing Unit
FIFO	-	First-In, First-Out
FIR	-	Finite Impulse Response
FM	-	Fiduccia-Mattheyses
FPGA	-	Field Programmable Gate Array
HDL	-	Hardware Description Language
HEVC	-	High Efficiency Video Coding
IC	-	Integrated Circuit
IDE	-	Integrated Development Environment
KL	-	Kernighan-Lin
KPN	-	Kahn Process Network
MoC	-	Model of Computation
MPEG	-	Moving Picture Experts Group
NP	-	Nondeterministic Polynomial time
ORCC	-	Open RVC-CAL Compiler
PSO	-	Particle Swarm Optimization
RTL	-	Register-Transfer Level
RVC	-	Reconfigurable Video Coding
VLSI	-	Very Large Scale Integration
XDF	-	eXtensible Design Format

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Problem Background

Semiconductors have been shaping the modern world with its wide usage throughout multiple industries with its influence growing each day. Its use in electronics means that it is able to be used in virtually any field in the current Internet of Things. Complying with Moore's Law, semiconductors allow the miniturization of electronic devices making electronic devices more powerful as well as more affordable for consumers. Since the first integrated circuit (IC) was invented, technology has not looked back since and we now have very large scale integration (VLSI) circuits built such as the central processing unit (CPU) being used in just the palms of our hands within smartphones.

Starting from simple building blocks, designs have been getting increasingly more complex to satisfy the demands of consumers. As such, proper methods of representing the designs have also become increasingly important. High levels of abstraction are needed to just see the big picture but still being able to look in deatil into the individual building blocks of a design. This gives rise to dataflow actor networks and all its accompanying languages. The added benefit of this representaion is the aspect of parallelism available, exactly as it appears in hardware, making this representation highly suitable for use when designing for hardware.

With competition at an all-time high, companies are looking for ways to produce application specific integrated circuits (ASIC) designs quickly to beat out other competitors, leading to the rapid advancement of use of FPGAs. These devices

are, as its name implies, programmable, meaning that the function that is implemented on them can be changed on a whim by the users. This is done by simply altering the Register-Transfer Level (RTL) code to be downloaded onto the FPGA. This versatile technology has become very prominent in the industry for prototyping designs in order to achieve the quickest possible time-to-market.

Currently, designs are still becoming larger and more complex, such that a single FPGA is unable to sustain the whole design anymore, compromising the ability to prototype. The solution? Use multiple FPGAs for the same design, increasing the number of available resources and allowing flexibility. This then becomes a question of how the design is to be divided in order to be implemented in FPGAs separately. This is where partitioning plays a huge role, breaking down designs in the right way, ensuring that the design itself is not compromised by the connections that need to traverse multiple FPGAs. A good partitioning algorithm will reduce the need for these connections, allowing a smooth implementation in hardware.

1.2 Problem Statement

Partitioning is getting more and more important with the increase in size of designs. In this project, the aim is to partition designs to be implemented in multiple FPGAs. As such the inter-FPGA connections will be penalized with extra latency and external noise. Therefore, it is imperative that the designs are partitioned correctly, so that the latency and noise can be tolerated. Low quality partitions could lead to the design not working as intended, which illustrates the importance of using a good partitioning algorithm to separate the designs.

Reducing cut size is the usual obective of partitioning. However, since this approach does not understand the concept of critical paths, it could lead to undesirable results in some designs. For instance, in data driven designs, doing this could lead to critical paths with high communications rates as one of the paths that needs to traverse FPGAs. The latency and noise incurred on the critical path could render the entire design to be compromised and non-functioning. In order to ensure that this is

less likely to happen, the connections are weighted by its communication rate when partitioning is conducted to optimize this metric between FPGAs.

Partitioning, as with most optimization problems, can be solved or approximated with deterministic or stochastic algorithms. Deterministic algorithms are those that are determined by the parameters set and the initial conditions whereas stochastic algorithms have an inherent randomness. Deterministic partitioning algorithms include the FM algorithm as well as the Kernighan-Lin (KL) algorithm which take an initial state of partitions and perturbs it by swapping if it results in a better partitioning solution. The drawback of this approach is that, due to the greediness of the algorithms and the heavy reliance on initial conditions, it could lead to deterministic algorithms getting locked onto local optimums and this becomes more and more apparent in large test cases due to partitioning being a nondeterministic polynomial time (NP) complete problem.

1.3 Project Objective

The objectives that this project aims to meet are as follows:-

1. To develop, implement and analyse the performance of stochastic partitioning algorithms, Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO).

2. To investigate the use of communication rate as the optimized metric as opposed to the traditional cut size when performing partitioning.

3. To analyse the improvement of using stochastic partitioning algorithms over deterministic algorithms.

1.4 Project Scope

Partitioning is a very large area of study and as such, this project will need its limits clearly defined. First, is that this project is limited to test cases which are part of a video processing design that are written in the CAL Actor Language (CAL). The test cases are an FIR Digital Filter, which is small testcase used more as a proof of concept to test the partitioning algorithms before moving to larger HEVC Decoder and the MPEG-4 AVC Decoder test cases.

The partitioning algorithms used in this project are meant to be stochastic in nature. Therefore, the algorithms that are chosen are the PSO and ACO algorithms. These algorithms were made to be used for different kinds of optimization problems and are not inherently used as partitioning algorithms. As such, these algorithms will need to be adapted to fit the problem at hand. The algorithms are written in the Java programming language.

The partitioning in this project will only be separating the designs into two partitions (bipartitioning). The algorithms will need to include a mechanism in order to balance the size of the partitions produced. In terms of partitioning criteria, this project will only optimize either the traditional cut size or the communication rate to see if data driven designs like the given test case will benefit from this different approach.

1.5 Thesis Organization

This thesis is organized as follows. In chapter one, the project is defined by establishing the problems to be solved as well as exploring the background of these problems. For each problem stated, the project objectives are defined. The scope is given to limit the project within well defined bounds. In chapter two, literature related to the problem are reviewed which include methods and algorithms used for partitioning designs as well as those giving further insight on usage of FPGAs and CAL that the design is written in. The following chapter then illustrates the flow

that this project will undergo to achieve its objective along with the approaches taken throughout the different parts of the project. Chapter four will then go into results obtained from each testcase after first presenting each test case in a more quantitative manner. The results are tabulated and analysed while conclusions are drawn. In the final chapter, the conclusions drawn are consolidated and the contributions of the project are documented. Before the end of the thesis, possible future works that could be done based on or extending this project are given.

REFERENCES

1. Kahn, G. The Semantics of a Simple Language for Parallel Programming. *Information Processing*. 1974. 471 – 475.

2. Ab. Rahman, A. A.-H. *Optimizing Dataflow Programs for Hardware Synthesis*. Ph.D. Thesis. ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE. 2013.

3. Eker, J. and Janneck, J. *CAL Language Report : Specification of the CAL Actor Language*. Technical report. University of California-Berkeley. 2003.

4. Gorin, J., Raulet, M. and Prêteux, F. MPEG Reconfigurable Video Coding: From specification to a reconfigurable implementation. *Signal Processing: Image Communication*, 2013. 28(10): 1224 – 1238.

5. Traskov, B. *Hardware/Software Partitioning of Dataflow Programs : Rapid Prototyping of Computer Systems in the CAL Actor Language*. Master's Thesis. KTH Royal Institute of Technology. 2011.

6. Tai, T. Y. Open RVC-CAL Compiler (ORCC) Guide. http://u1403182299-blog.logdown.com/posts/207201-orcc-guide, 2014. Accessed: 2018-05-24.

7. Bezati, E., Yviquel, H., Raulet, M. and Mattavelli, M. A Unified Hardware/Software Co-Synthesis Solution for Signal Processing Systems. *Design and Architectures for Signal and Image Processing*. 2011. 1 – 6.

8. Wendling, M. and Rosenstiel, W. A Hardware Environment for Prototyping and Partitioning Based on Multiple FPGAs. *Proceedings of the Conference on European Design Automation*. 1994. 77 – 82.

9. Drayer, T. H., King, W. E., Tront, J. G., Connors, R. W. and Araman, P. A. Using Multiple FPGA Architectures for Real-time Processing of Low-level Machine Vision Functions. *Industrial Electronics, Control, and*

Instrumentation. 1995, vol. 2. 1284 – 1289.

10. Jaeger, J. Partitioning an ASIC Design into Multiple FPGAs. https://www.eetimes.com/document.asp?doc_id=1274719, 2010. Accessed: 2018-05-24.

11. Fleming, K. E., Adler, M., Pellauer, M., Parashar, A., Mithal, A. and Emer, J. Leveraging Latency-insensitivity to Ease Multiple FPGA Design. *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*. 2012. 175 – 184.

12. Kernighan, B. W. and Lin, S. An Efficient Heuristic Procedure for Partitioning Graphs. *The Bell System Technical Journal*, 1970. 49(2): 291 – 307.

13. Fiduccia, C. M. and Mattheyses, R. M. A Linear-time Heuristic for Improving Network Partitions. *19th Design Automation Conference*. 1982. 175–181.

14. Tong, Q., Zou, X., Zhang, Q., Gao, F. and Tong, H. The Hardware/Software Partitioning in Embedded System by Improved Particle Swarm Optimization Algorithm. *Fifth IEEE International Symposium on Embedded Computing*. 2008. 43 – 46.

15. Kumar, K. S., Bhaskar, U. P., Chattopadhyay, S. and P., M. Circuit Partitioning Using Particle Swarm Optimization for Pseudo-Exhaustive Testing. *International Conference on Advances in Recent Technologies in Communication and Computing*. 2009. 346 – 350.

16. Comellas, F. and Sapena, E. *Applications of Evolutionary Computing*, Springer, chap. A Multiagent Algorithm for Graph Partitioning. 2006, 279 – 285.

17. Arora, M. and Lall, G. C. Circuit Partitioning in VLSI Design : An Ant Colony Optimization Approach. *International Journal of Advances in Engineering & Technology*, 2013. 6(1): 536 – 541.

18. Omeroglu, N. B., Toroslu, I. H., Gokalp, S. and Davulcu, H. K-Partitioning of Signed or Weighted Bipartite Graphs. *International Conference on Social Computing*. 2013. 815 – 820.

19. Chin, Y. H. *Dataflow Actor Network Partitioning for Multiple FPGAs*. Master's Thesis. Universiti Teknologi Malaysia. 2016.

20. Mohd. Asri, M. F. *Dataflow Actor Network Bi-partitioning Algorithm for*

Hardware Implementation. Master's Thesis. Universiti Teknologi Malaysia. 2017.