UNIVERSITY OF CALIFORNIA, SAN DIEGO

On-Chip Interconnection Architecture Optimization Using a Multicommodity Flow Approach

A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy

in

Computer Science

by

Yuanfang Hu

Committee in charge:

Professor Chung-Kuan Cheng, Chair
Professor Bill Lin
Professor Curt Schurgers
Professor Michael B. Taylor
Professor Dean M. Tullsen

2007

.

The dissertation of Yuanfang Hu is approved, and it is acceptable in quality and form for publication on microfilm:

Chair

University of California, San Diego

2007

To My Parents.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank everyone who helped me during my many years of graduate study in UCSD.

First of all, I would like to thank my advisor, Professor Chung-Kuan Cheng, for his constant support and help. I experienced some very difficult situations twice in my doctoral study. In both cases, it is Prof. Cheng, who gave me his hands. Without his help and encouragement, I could not go through all the difficulties and have such a proud achievement in my life. Besides his help, I am also impressed by his confidence, persistence, diligence, passion, and smartness. It has been a great honor to have the opportunity to work with him and learn from him.

Furthermore, I would like to thank Professor Bill Lin, Professor Curt Schurgers, Professor Michael Taylor, and Professor Dean Tullsen, for serving on my committee, and helping me with my dissertation. I would like to thank my fellow graduate students and colleagues in VLSICAD Lab, Hongyu Chen, Bo Yao, Zhengyong Zhu, Yi Zhu, Ling Zhang, Shuo Zhou, Jianhua Liu, Haikun Zhu, He Peng and Rui Shi, for their invaluable discussion and help on my research. Special thanks go to Hongyu Chen, for his constant help and encouragement. Only with his talented ideas and valuable discussion, am I able to finish my doctoral program. I learned a lot from him. I would also like to have special thanks to Yi Zhu and Ling Zhang, for their kind help, discussion and friendship.

Finally, I would like to thank my family for their support, encouragement and love. Special thanks go to my husband Huaxia Xia. He did all he could to help me during my most difficult time. His encouragement and love has made me capable of weathering all the ups and downs throughout the ordeal of my doctoral study. My gratefulness to him is beyond any words.

Chapter II includes the contents of two published papers. “Communication Latency Aware Low Power NoC Synthesis,” by Y. Hu, Y. Zhu, H. Chen, R. Graham, C.K. Cheng. “Physical Synthesis of Energy-Efficient NoCs Through

Topology Exploration and Wire Style Optimization," by Y. Hu, H. Chen, Y. Zhu, A. A. Chien, C.K. Cheng. The dissertation author was the primary researcher and co-author of both papers.

VITA

1998	B.E. in Computer Science Tsinghua University, Beijing, China
2000	M.S. in Computer Science Tsinghua University, Beijing, China
2007	Ph.D. in Computer Science University of California, San Diego

PUBLICATIONS

S. Zhou, Y. Zhu, Y. Hu, R. Graham, M. Hutton, C.K. Cheng, "Timing Model Reduction for Hierarchical Timing Analysis," *International Conference on Computer Aided Design,* pp. 415–422, November, 2006.

Y. Hu, Y. Zhu, H. Chen, R. Graham, C.K. Cheng, "Communication Latency Aware Low Power NoC Synthesis," *Design Automation Conference,* pp. 574–579, July, 2006.

Y. Hu, H. Chen, Y. Zhu, A. A. Chien, C.K. Cheng, "Physical Synthesis of Energy-Efficient NoCs Through Topology Exploration and Wire Style Optimization," *International Conference on Computer Design,* pp. 111–118, October, 2005.

J. Weinberg, A. Jagatheesan, A. Ding, M. Faerman, Y. Hu, "Gridflow Description, Query, and Execution at SCEC using the SDSC Matrix," *International Symposium on High-Performance Distributed Computing,* pp. 262–263, June, 2004.

S. Narayanasamy, Y. Hu, S. Sair, B. Calder, "An Instruction Scheduling Co-Processor for Adaptive VLIW Schedules," *International Conference on High Performance Computer Architecture,* pp. 210–221, February, 2004.

S. Sair, Y. Hu, T. Sherwood, B. Calder, "Optimized Trace Binaries for Architectural Evaluation," *UCSD Technical report CS2002-0711,* 2002.

Y. Hu, S. Zhang, W. Jiang, "Applying Object-Oriented Method to CSIE System," *International Conference on Technology on Object-Oriented Languages and Systems,* pp. 484–491, September, 1999.

FIELDS OF STUDY

Major Field: Computer Science
Studies in VLSI CAD
Professor Chung-Kuan Cheng, University of California, San Diego

ABSTRACT OF THE DISSERTATION

On-Chip Interconnection Architecture Optimization
Using a Multicommodity Flow Approach

by

Yuanfang Hu
Doctor of Philosophy in Computer Science
University of California, San Diego, 2007
Professor Chung-Kuan Cheng, Chair

Recent technology advent makes the efficient on-chip interconnection architecture critical in modern IC design. First, on-chip communication requirements are dramatically increasing due to the continuously shrinking of the device feature size and integration of billions of transistors on a single chip. Second, the interconnects, rather than devices, become the dominant factors in deciding performance and power consumption of VLSI systems. As a result, we are urged to optimize interconnection architecture to improve the overall system performance.

In this dissertation, we propose a methodology to optimize the power consumption or communication latency of two promising on-chip interconnection architectures, Network-on-Chip (NoC) and FPGA global routing architecture. Our methodology adopts two optimization schemes, topology optimization and wire style optimization, and uses multicommodity flow (MCF) models to perform and evaluate these optimizations. We implement and optimize the MCF solver using polynomial approximation algorithms, which are significantly faster than the commercial linear programming solver CPLEX.

For NoC optimization, our objective is to search for most power efficient NoC topologies and their wire assignments, which at the same time satisfy all communication latency and bandwidth requirements. We use MCF formulations

to model this problem, and build NoC topology library and its power and delay library as inputs to MCF formulations. The solution of MCF formulations indicates the estimated NoC power consumption under a certain NoC topology and wire style assignments, from which we can observe the best NoC optimization. The experimental results show that our methodology can effectively improve the NoC power consumption or communication latency. The results also indicate the importance of power and latency co-optimization in NoC design.

For FPGA global routing architecture optimization, we study segmentation distribution, flexible track assignment and wire style optimization. The objectives are power consumption and switch area density. The design methodology is also based on MCF formulations. We examine our FGPA routing architecture optimization by a set of standard MCNC [57] benchmark circuits. The experimental results show that our optimized FPGA routing architectures achieve average up to 10% to 15% power savings and up to 20% switch area savings when compared to traditional FPGA architecture.

I

Introduction

Recent technology advent makes the efficient on-chip interconnection architecture critical in modern IC design. On one hand, VLSI layout feature size has shrunk into the deep-submicron (DSM) domain. The continued scaling of transistor size allows designers to integrate tens or hundreds of IP blocks on a single chip. These IPs can be CPU or DSP cores, video streaming processors, memory blocks, etc [23]. On the other hand, the shrinking feature size makes interconnect delay and power consumption the dominant factors in VLSI system. As a result, efficient on-chip communication architecture emerges as an important design issue to optimize the performance and power consumption of the modern systems.

On-chip communication architecture design involves many complicated design issues, such as interconnection topology, switching technology, routing schemes, deadlock detection, flow control mechanisms, etc. Our work focuses on the Interconnection Architecture design of on-chip communication systems, and investigate how to effectively construct the interconnections among on-chip communication elements and how to optimize the design according to various design objectives and constraints.

In this dissertation, we develop a methodology to optimize two types of promising on-chip interconnection architectures, Network-on-Chip (NoC) and Field Programmable Gate Array (FPGA) routing architecture, which have regular

interconnection structures in common. This shared desirable feature makes them especially promising in DSM domain, since regular wire structures can improve signal qualities, reduces crosstalk and probabilities of errors, and makes aggressive wiring optimization techniques possible.

Our methodology adopts a multicommodity flow (MCF) approach to optimize the NoC and FPGA routing architectures. MCF has been intensively studied and applied in VLSI global routing since early 90's, and has shown to be an effective approach [2][12]. Different from these existing works, we apply MCF to NoC and FPGA routing architecture design from a unique simultaneous optimization point of view, and explore the new challenges. Furthermore, although NoC and FPGA routing architecture design share much similarity, they have many different design concerns from topology generation to routing characteristics. Hence, we study the optimization of NoC and FPGA routing architecture separately.

Section I.1 explains the motivation of this dissertation work, including impacts of technology trends on on-chip interconnection architecture design, our design metrics, and our approaches. Section I.2 and Section I.3 briefly introduce the background and related researches on NoC and FPGA routing architecture design, respectively. I.4 rounds up with an overview of the dissertation organization.

I.A Motivation

I.A.1 Technology Trends

With the semiconductor technology scaling down, the importance of on-chip interconnect architecture has dramatically risen over the past decade. Prior to the early 1990s, delay and power consumption of logic gates dominated, and on-chip wires are considered as purely capacitive loads of logic gates in chip design. Nowadays, growing wire resistance coupled with shrinking native gate speed and power made wire delays and power consumption increasingly important. Figure I.1 [41] shows wire and gate delays into various technology nodes, which indicates

a clear divergence between gate delays and wire delays. Figure I.2 [38] shows the dynamic power breakdown of an Intel Pentium processor under various technology generations. Interconnect power grows to more than 50% of the total power, and this number keeps growing into the future. Therefore, global on-chip interconnection architecture design becomes more and more critical in chip design.

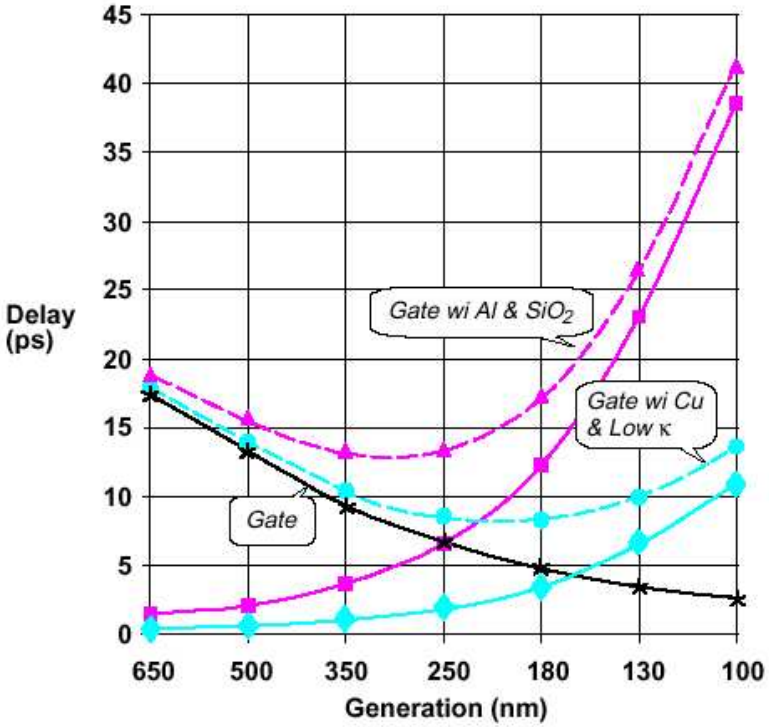


Figure I.1 Source [25]: Wire and Gate Delay Comparison Under Various Technology Nodes

I.A.2 Design Metrics

Unlike most of the traditional interconnection architecture design, which pursues only performance improvement, i.e., reducing communication latency or increasing bandwidth, on-chip interconnection architecture design has additional concerns and design Metrics. In summary, this dissertation adopts following design metrics:

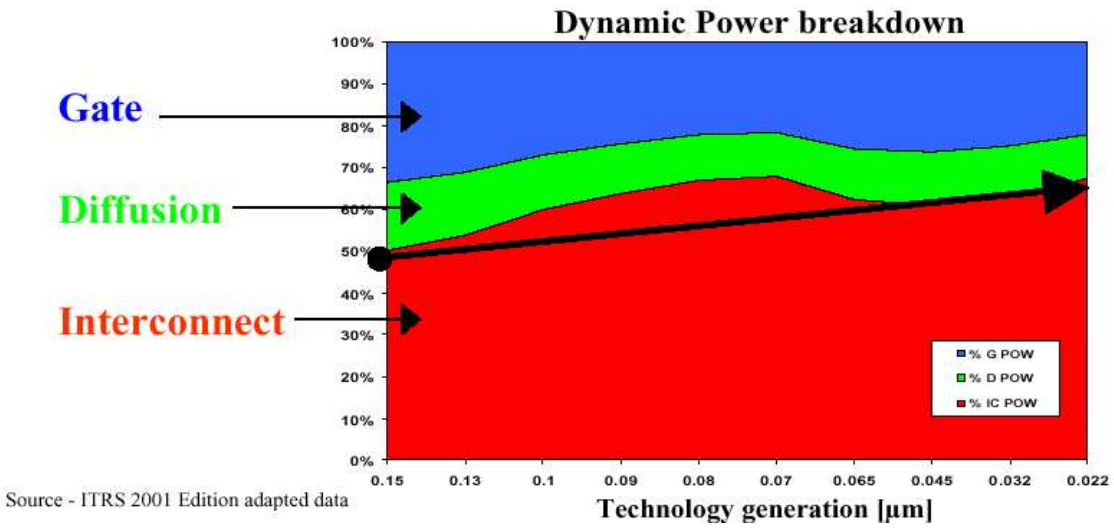


Figure I.2 Source [38]: Power Partition of a Processor

- On-chip area. On-chip interconnection architectures are physical implementation sensitive and should be routable within the limited on-chip wiring resources. Area constraint becomes one of the most critical considerations in NoC and FPGA routing architecture design.

- Power consumption. At the market of mobile products continues expanding and the chip cooling cost keeps increasing, low-power design becomes essential for a success system. Although the power consumption of transistors scales down effectively with the technology scaling, interconnect power consumption does not scale down well [25] and takes more and more occupation of the chip power consumption. Therefore, it is critical in NoC and FPGA routing architecture design to provide efficient communication solution addressing power consumption issues.

- Communication Latency. Undoubtly, performance is as important for on-chip communication architecture as for traditional communication architecture. Chip clock rates are largely affected by performance of communication architectures.

- Communication Bandwidth. Similar to traditional communication architecture, the on-chip interconnection architecture has to satisfy all communication bandwidth requirement with the lest communication congestion.

I.A.3 Optimization Approaches

In this dissertation, we focus on two optimizations, topology optimization and interconnect wire style optimization, to simultaneously optimize power consumption, on-chip area usage and communication latency of on-chip interconnection architecture. At the same time, the design should satisfy given constraints, such as communication bandwidth requirements and physical on-chip area resource constraints.

Topology Optimization

The choices of interconnection topology are important for both NoC and FPGA routing architecture design. Different topologies can dramatically affect the interconnection architecture characteristics, such as number of hops, total wire length, and communication flow distributions. These characteristics in turn determine the latency and power efficiency of interconnection architectures.

Many topologies and their properties have been studied. A large type of topologies are called orthogonal topology, in which nodes are connected in k-ary n-dimensional mesh (k-ary n-mesh) or k-ary n-dimensional torus (k-ary n-cube) formations. Because of the simple connection and easy routing provided by adjacency, mesh and torus networks are widely used in traditional network [20]. One of the merits of orthogonal topology is that it is highly regular and reduces the design efforts greatly. Besides orthogonal topology, other popular topologies include Butterfly topology, Banyan topology, Octagon topology and Fat-Tree topology, and so on. Figure I.3 shows examples of various topologies (Figure I.3(a), (f) Butterfly Fat-Tree topology; Figure I.3(b) Mesh; Figure I.3(c), (d) Torus; Figure I.3(e) Octagon topology).

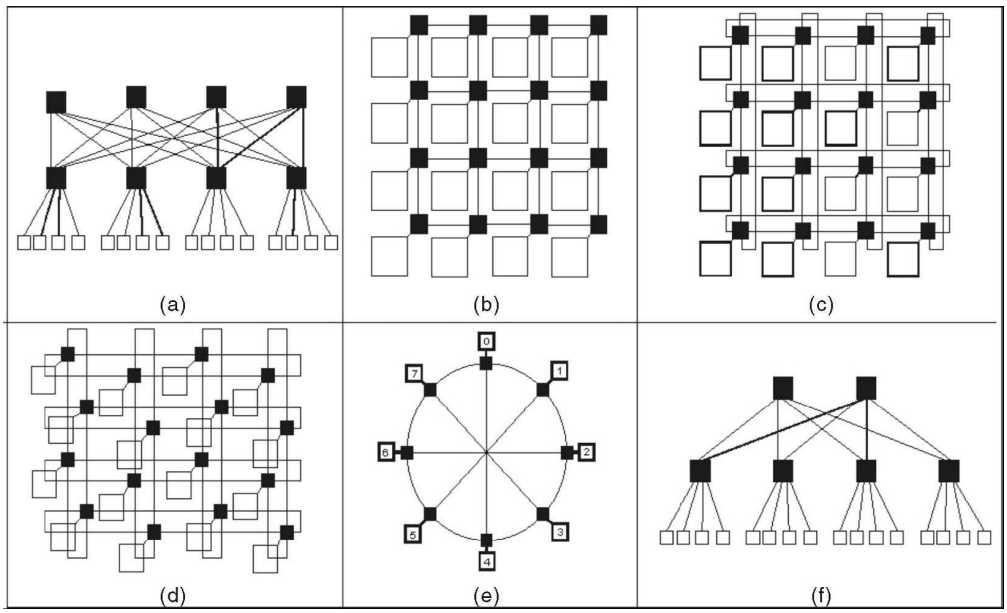


Figure I.3 Examples of Popular Topologies for Interconnection Networks

In this dissertation, we explore a special type of topology, where each row and each column have the same connections. This type of topologies make circuit design and layout easier. They also cover a wide range of popular network topologies described above, such as orthogonal topology, Octagon topology, and twisted cube, etc.

Wire Style Optimization

Wire style optimization is a unique and important optimization for on-chip interconnection architecture, which implements interconnects with the best-fit wiring technologies to optimize the overall system power consumption or communication latency. Optimized wire styles for the critical NoC or FPGA routing links can significantly reduce the interconnect power consumption and latency.

Recent advances in signaling interconnect technologies, such as wave-pipelined RC wires with repeated buffers, low-swing differential pairs, and on-chip transmission lines, provide us wiring schemes to optimize aggressively. These technologies along with traditional minimal separated RC wires display different

Table I.1 Comparison of Different On-Chip Wiring Technologies

Wire styles	**Pros.**	**Cons.**
Buffered *RC* Wire with Minimum Spacing	highest wiring density, most area efficient	highest per-bit energy, longest latency
Buffered *RC* Wire with Large Spacing	shorter latency, lower per-bit energy compared with minimum spaced *RC* wires	more routing area usage than minimum spaced wire
On-Chip Transmission Line	shortest latency, low per-bit energy for long-distance communication	largest routing area, large initial power, largest design effort

tradeoffs between wire resources, wire delay and power consumption.

Table I.1 shows the comparison of three types of popular wiring technologies adopting in this dissertation, regarding to their delay, power and area consumption. RC wire with appropriately spaced repeaters is the most common and simplest means of global interconnect [25]. With inserted buffers, its wire delay is improved to linear with total wirelength, and the wire bandwidth is increased substantially. However, power consumption of RC wires with repeated buffers increases linearly with the total wire length, hence it is not power-efficient for long distance on-chip interconnects. Increasing the spacing between wires can reduce power consumption, but costs more on-chip area resources. Transmission line is appropriate for long distance inductance-dominant high-frequency on-chip interconnects. For length of 20mm, transmission line is five times faster and half power of that of RC wires with repeated buffers [14]. However, due to its complexity at transmitter and receiver circuits, transmission line has a large setup power overhead, hence not suitable for short distance interconnects. Transmission line also requires much wider wire pitch to transfer signals, taking more on-chip area resources.

I.A.4 Multicommodity Flow (MCF) Problems

We integrate both topology and wire style optimizations in our optimization framework to optimize on-chip interconnection architecture. Our methodology is based on *Multicommodity flow* models.

MCF problem is a generalization of maximum-flow problem where, instead of a single commodity, we have several different commodities. Each commodity has an associated source/sink pair and a demand. The goal is to ship all of commodities while satisfying all constraints. More formally, a MCF problem is defined on a directed graph $G = (V, E)$ with capacities $u : E \rightarrow R$ and k source-sink pairs $(s_j, t_j), 1 \leq j \leq k$, where there might be various constraints (edge capacity, flow latency, etc.) and demands (e.g., commodity demands). The problem is to find flows f_j from s_j to t_j that satisfy node conservation constraints and meet some objective function criteria so that the sum of flows on any edge does not exceed the capacity of the edge.

Among various multicommodity flow problems, *maximum concurrent multicommodity flow* and *minimum cost multicommodity flow* are of the most interest of research and applications. For the maximum concurrent multicommodity flow problem, there are demands d_j associated with each commodity j, and the objective is to satisfy the maximum possible proportion of all demands. For minimum const concurrent flow problem, besides of the demands associated with the commodities, there are costs $c(e)$ associated with a unit of flow on edge e. We define the cost of a flow as the cost of sending flow on each edge of the graph summed over all the edges carrying flow. The objective is to find a maximum concurrent flow that satisfies the communication demands and has minimum cost.

Both maximum concurrent flow problem and minimum cost concurrent flow problem are proven to be NP-hard. In recent years, several *fully polynomial time approximation schemes* (FPTAS) are proposed to solve these problems with satisfying accuracy and efficiency. A FPTAS is a family of algorithms that finds an ϵ-approximate solution in time polynomial in the size of the input and $\frac{1}{\epsilon}$. The

algorithms in our optimization framework are similar to these FPTAS algorithms.

MCF model has proven to be an effective tool in global routing optimization. Carden IV *et al.* [12] [13] routed multi-terminal netlists using the approximation MCF algorithm by Shahrokhi and Matula [51]. Garg and Konemann had breakthrough improvements of the algorithm [22], and Albrecht applied the new algorithm to render MCF based global routing practical for full chip design [1]. Albrecht further extended the MCF model for more optimizations in congestion and timing-driven global routing, including buffer insertion, pin assignment, and buffer wire sizing [3]. The basic approach of these previous work is similar to ours. However, we introduce multiple wire styles into the MCF models, and optimally assign the capacities for these wire styles for interconnections, which is shown to have large improvement for on-chip interconnection architecture design.

I.B Review of Network-on-Chip(NoC) Design

I.B.1 Emergence of NoC

Although traditional shared-bus scheme is widely adopted in most existing systems as on-chip interconnect architecture, its performance is inherently not scalable as there can be at most one transaction over the shared bus at any point of time. Therefore, it is no longer satisfying under the new scenario, where on-chip communication demands are orders of magnitudes higher than before. Furthermore, a global bus implies a large capacity load for the bus drivers, which makes large interconnect delay and huge power consumption unavoidable.

Instead, NoC architecture [19][24][35] emerges as a promising on-chip communication solution. It is inspired by scalability and success of packet-based networks in parallel computing and Internet. Although there are many possible NoC implementations, regular tile-based NoC architecture gains most research interest due to its simplicity and popularity. Hence, the work in this dissertation is based on assumption of such NoC architecture. Figure I.4 shows an example of

a simple regular tile-based NoC. In a regular tile-based NoC architecture, a system is composed by placing client logic (e.g., processors, DSPs, peripheral controllers, memory subsystems, etc.) into the regular tiles. A router is embedded within each tile to connect it to its neighboring tiles. In the example, the client logic blocks communicate with one another over network of mesh topology. Therefore, instead of routing design-specific global on-chip wires, the inter-tile communication can be achieved by routing network packets.

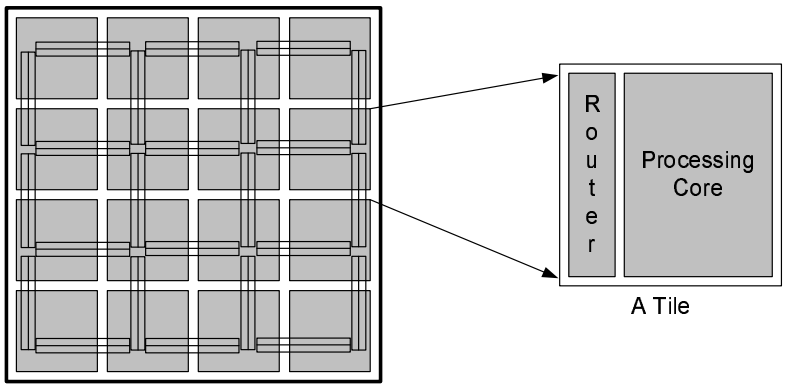


Figure I.4 A Regular 4×4 Tile-Based NoC Architecture with Mesh Topology

As a novel on-chip communication scheme, NoC architecture has many desired properties that distinguish it from other on-chip communication schemes. First, NoC usually adopts very regular structures, and the paths between tiles are precisely defined at the beginning of the design process. Therefore, the network wires can be optimized beforehand to reduce crosstalk and enable the use of aggressive signaling circuits, which reduce power dissipation and propagation delay significantly. Second, on-chip wire resources in NoC can be efficiently shared by many signals, which makes fully use of the critical wiring resources. In comparison, the traditional dedicated wiring scheme has wires be used on average less than 10% of time. Third, with well defied interfaces, NoC can be easily reused in a large number of products, since the switches/routers, the interconnects and lower-level communication protocols can be designed, optimized and verified once and reused for general purposes. At last, compared to the bus architecture, NoC

is scalable. Communication is distributed over the whole network, therefore NoC scheme avoids the severe congestion problems of the bus architecture.

I.B.2 Related Research in NoC Design

Although NoC shares much similarity with the traditional networks, such as flow control mechanisms to avoid network congestion, and efficient routing algorithm to improve performance, etc., NoC architecture has specific features that differentiate it from traditional networks and invokes new design concerns. For example, compared to traditional network designs where there is no foresee traffic pattern available, most NoC are developed specifically for one application or as a platform for a small set of application. Consequently, the designer usually has a good understanding of the traffic characteristics and is able to use this information to customize the NoC design accordingly. Furthermore, in NoC design, how to implement it using limited on-chip resources is one of the important design considerations, while traditional network does not have to worry about this. Traditional network design usually takes performance with the solely targets, but in NoC design, power efficiency is probably more critical.

Because of the above specific features, the existing techniques for traditional network optimization can not be directly applied to NoC design. Thus, there is a need for NoC-specific design methodologies to fully exploit these NoC features.

The basic regular mesh NoC infrastructure is by far the most popular NoC architecture due to its predictability and ease of design. In [19][24][35][37], the possibilities and advantages of using regular NoC for on-chip communication are explored. The use of other regular topologies have also been investigated, such as the octagon topology in [32], fat free topology in SPIN networks [23] and the honeycomb topology in [24].

Routing strategy is another important design issue in NoC design. In [48] and [42], on-chip routers with quality of service (QoS) support are developed,

which allows the NoC to provide both guaranteed throughput service and best-effort service. In [26], Horiguchi et al. present an adaptive routing algorithm for the TESH NoC architecture, which is able to avoid faulty or congested nodes. Routing strategies have also been exploited to control the network peak power consumption [52] and thermal profile [53].

Many works have been done in NoC synthesis and optimization. In [27], Hu et al. proposed a branch and bound algorithm to map the processing cores onto a tile-based mesh NoC architecture to satisfy bandwidth constraints and minimize total energy consumption. In [44][43], Murali et al. designed topology mapping to minimize the average communication delay while satisfying bandwidth constraints, and presented a tool named SUNMAP to select the best topology for a given application. In [30][18], a tool named XpipesCompiler is presented for automatically generating an application specific NoC for heterogeneous multiprocessor SoCs. This is achieved by instantiating a network of building blocks from a library of composable soft macros that are described in SystemC at the cycle-accurate level. In [28], a variety of NoC topologies are designed and the effect of topology on NoC power consumption is studied. In [45], Ogras et al. inserted a few application-specific long-range links to regular mesh based topology to reduce average packet latency.

I.C Review of FPGA Routing Architecture Design

Since introduced in early 80's, FPGA has become one of the most popular implementation for digital circuits. The major advantage of FPGA architectures is their ability to implement any circuit simply by being appropriately programmed, which leads to low non-recurring engineering costs and faster time-to-market. However, FPGA programmability also carries a price. FPGA connects circuits via programmable switches, which incurs additional overheads on power consumption, delay and area. Furthermore, FPGA introduces much redundancy to accommodate various applications on a generic design. As a result, a circuit

implemented in an FPGA is typically 10 times larger and roughly 3 times slower and consumes much more power than the same circuit implemented in ASIC design [9]. Therefore, it is critical to constantly search for better FPGA architectures in order to gain improvement on power and density. In this dissertation, we focus on efficient FPGA routing architecture design, since as we explained earlier, power consumption and latency of routing architecture are dominant in DSM domain.

I.C.1 Basis of FPGA Routing Architecture

Commercial FPGAs adopt one of the three types of routing architectures. The FPGAs of Xilinx, Lucent and Vantis are island-style FPGAs, while Actel's FPGAs are row-based, and Altera's FPGAs are hierarchical [9]. In our dissertation, we will investigate the island-style routing architecture, as shown in Figure I.5. Logic blocks are surrounded by routing channels of pre-fabricated wiring segments on all four sides. A logic block input or output can connect to some or all of the wiring segments in the channel adjacent to it via a connection block of programmable switches. At every intersection of a horizontal channel and a vertical channel, there is a switch block. This is simply a set of programmable switches that allow some of the wire segments incident to the switch block to be connected to others. By turning on the appropriate switches, short wire segments can be connected together to form longer connections.

To design an FPGA routing architecture includes two different issues, global routing architecture and detailed routing architecture. The global routing architecture of an FPGA specifies the relative width of the various wiring channels within the chip that permits their efficient utilization by the largest class of circuits. The detailed routing architecture defines how logic block inputs and outputs can be interconnected. In this dissertation, we investigate the global FPGA routing architecture design.

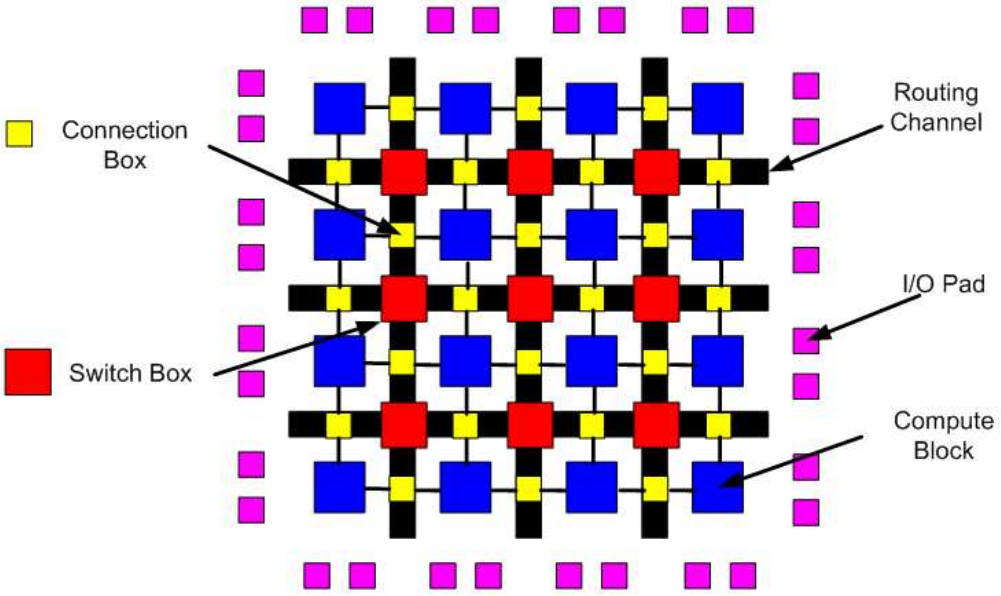


Figure I.5 Island-Style FPGA Routing Architecture

I.C.2 Related Research in FPGA Routing Architecture Design

Despite the large amount of research work in FPGA routing architecture design, we are mainly interested in global routing architecture design for island-style FPGA. Figure I.6 shows a classic CAD flow to evaluate an FPGA global routing architecture. First, SIS [50] is used to perform technology independent logic optimization on a circuit. Next this circuit is technology-mapped by FlowMap [17] into four-input look-up tables (4-LUTs) and registers. We then use VPack [17] to pack 4-LUTs into a bigger block logic element (BLE). Given any global routing architecture, we use VPR [5] to place and route the circuit, and finally we can have the track assignments on each channel of that routing architecture.

There are roughly two categories of research work for island-style FPGA global routing optimization. One investigates the segmentation distribution, the other studies routing track distribution. Segmentation distribution defines the length of routing tracks in each channel. Figure I.7 shows an example of segmentation distribution. The routing wires are of length 1, 2 and 4. 50% of tracks are of

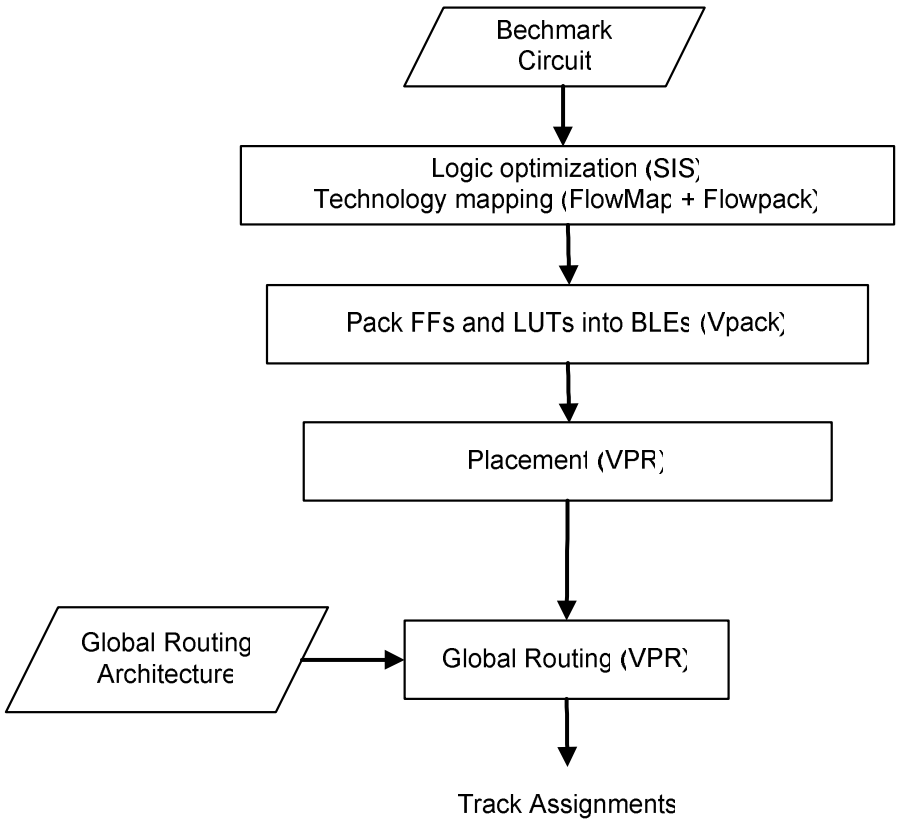


Figure I.6 CAD Flow for FPGA Routing Architecture Evaluation

length 1, 25% are of length 2, and 25% are of length 4. Brown et al investigated differentiation distributions in order to optimize the speed and area of an island-style FPGA [10][11][34]. They placed circuits, then routed them with a global routing followed by a detailed routing procedure, and evaluated the resulting speed and area for each FPGA architecture of interests. Chow et al [16] performed a similar study on the impact of segmentation distributions on circuit routability.

Routing track distribution defines the relative number of tracks contained in each channel of the FPGA. In [6][7], Betz et al investigate the relationship between routing track distribution and area-efficiency for island-style FPGA. They investigate directionally-biased global routing architectures, in which the channels in the vertical direction have a different width than those in the horizontal di-

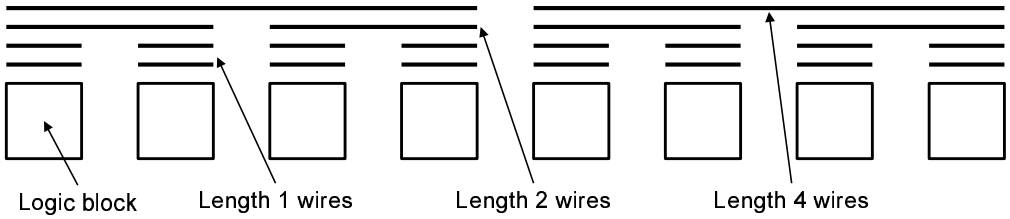


Figure I.7 An Example of Segmentation Distribution

rection. They also examine non-uniform global routing architectures, which have wider channels in some regions of the FPGA than in others. Experiments show these optimizations do not bring much benefit, though.

I.D Dissertation Organization

This dissertation presents a novel methodology that unifies various design metrics in MCF formulations, and explores topology optimization and wire style optimization for NoC and FPGA routing architecture design.

In Chapter II, we present the complete design flow to demonstrate our methodology of applying MCF formulations to NoC design. The objective is to find most power efficient NoC topologies and their wire assignments, which at the same time satisfy all communication latency and bandwidth requirements. We use MCF formulations to model the problem, and build NoC topology library and its power and delay library as inputs to MCF formulations. According to the solution of MCF formulations, which indicates the estimated NoC power consumption under a certain NoC topology and wire style assignments, we can observe the best topology and corresponding wire style assignments to achieve optimal power consumption. Besides that, our experiments also study the power and latency tradeoffs for various topologies in NoC design.

Chapter III studies FPGA routing architecture design. Although the design methodology is the same as that for NoC design, there are several major differences we need to emphasize accordingly. First, communication in NoC is

point-to-point packets, but in FPGA routing architecture, the communication is electronic signals on nets. The routing strategy is essentially different from these two types of on-chip communication architectures. Second, area overheads of network routers are negligible in NoC design, but that of switch box in FPGA is one of the major design metrics. In FPGA, we set area overheads of switch box as one of the design objectives. In experiments, we examine our design by a set of FPGA application benchmarks.

Finally, Chapter IV summarizes the major conclusions for this work and outlines a number of promising future research directions.

II

Communication Latency Aware Low Power NoC Synthesis

In this chapter, we present a NoC synthesis methodology that simultaneously optimizes communication latency and power consumption for NoC interconnect architecture through topology optimization and wire style optimization. Our approach use MCF formulations to model the problem and evaluates the tradeoffs between communication power and latency. The experimental results indicate the importance of power and latency co-optimization in NoC design. The chapter is organized as follows. Section II.1 formulates the latency aware low power NoC synthesis problem. Section II.2 and II.3 describe the design flow and an improved polynomial approximation MCF algorithm. In Section II.4 we give and analyze the experimental results. We summarize our study for NoC interconnect architecture optimization in Section II.5.

II.A Problem Statement

We assume a regular tile based NoC with tiles. Each tile consists of a processing core and a router. Each tile can be regarded as a grid with certain area and dimension, and the total wiring area across a grid cannot exceed grid dimension. Each network link can be implemented with multiple wire styles, which

have varying properties in terms of their area usage, power efficiency, and signaling latency.

Our goal is to select NoC topologies and corresponding interconnect wire style assignments, so that power consumption is minimized subject to average communication latency constraints. Figure II.1 shows an example of such a design optimization for a tile based NoC. The network is linked by various types of wire styles with different capacities (Figure II.1(a)). The topology is a folded torus (Figure II.1(b)). On-chip transmission lines are used for long distance communication to save power and reduce delay, while RC wires with buffer insertion are used for short local connections to make best use of on-chip area resources.

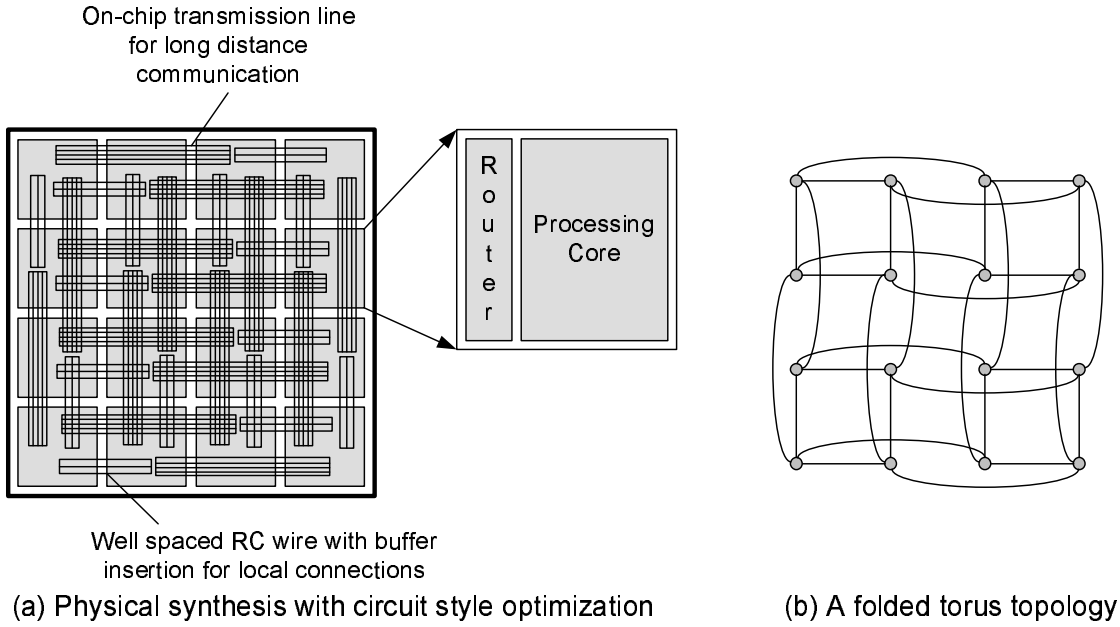


Figure II.1 Tile-Based NoC Architecture with Wire Style Optimization

We define a NoC topology graph as a directed graph $G = (V, E)$, where, each node $v_i \in V$ represents a tile, and each edge $e_{i,j} \in E$ represents a point to point interconnection between tile i and tile j.

An implementation of a NoC topology is a mapping from each edge to a particular wire style, $T(e) : E \rightarrow S$, and a mapping from each edge to the amount

of wiring resources assigned to that edge, $C(e) : E \rightarrow R^+$. For a link with certain length, we are provided a library of interconnect wire styles $S|S_i = (P_i, A_i, D_i)$. A particular wire style $s_i \in S$ has three properties associated with it: the per edge routing area usage A_i, the per bit energy P_i, and the wire delay D_i.

We formulate our problem as the following *communication latency aware minimum power NoC synthesis problem*:

Latency-constrained minimum power NoC synthesis problem:

We have an $n \times n$ array of tiles, a library of interconnect wire components of certain lengths:

Input: *The communication demand matrix between each pair of tiles*

Output: *The most power-efficient NoC topology $G = (V, E)$, and its physical implementation $T(e), C(e), \forall e \in E$*

Constraints: *(1)The communication latency requirements are satisfied; (2)The cross section wiring area can not exceed the grid dimension.*

II.B Design Methodology

As shown in Figure II.2, for a homogenous $n \times n$ NoC, we first automatically generate the topology library. Then, based on power and delay libraries, we use a MCF model to evaluate latency constrained NoC power consumption for each topology in topology library.

Our methodology assumes source routing algorithms to route packets over NoC. Packets belonged to the same message can be routed through different paths to the destination tile, at where software mechanisms take care of resembling packets together to rebuild the original message according to packet header information. When routing data packets to adjacent tiles, routing mechanism may dispatch packets to different types of wires according to their wire capacities.

Since we use network flow models to formulate the NoC synthesis problem, our work is within the static flow scope. For applications with dynamic communications, queuing mechanisms and flow control mechanisms are needed to

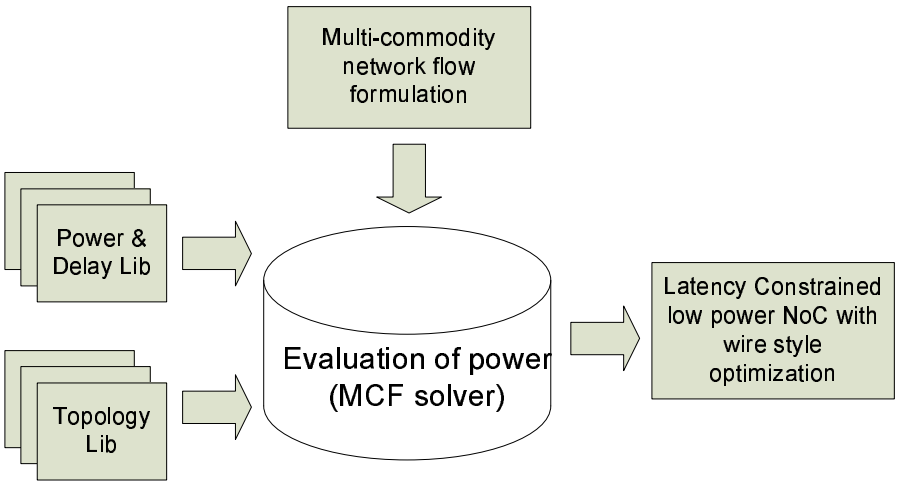


Figure II.2 Design Flow

solve the network contentions, and detailed simulations are required to observe the network dynamic behaviors. Although detailed simulation is a good approach for accurate network behavior estimation, it suffers from the very long evaluation time and high development efforts. On the contrary, though our static network flow model approach bypasses these important network design issues and lose some accuracy, it has merits of providing quick evaluation and useful guidance in the initial stages of NoC design.

In the following subsections, we describe our latency constrained low power MCF model, topology generation, and power and delay models, respectively. An approximation MCF solver will be introduced at section II.C.

II.B.1 Latency Constrained Minimum Power MCF Formulation

For a given NoC topology graph $G = (V, E)$, we construct a flow graph. For each link between any two nodes, it consists of t edges, where t is the number of candidate wire styles of the link. Figure II.3 shows an example flow graph. There are t edges from node v_m to node v_n, and associated with each edge, we have its wire style (P_e, A_e, D_e). P_e, A_e, and D_e are the per-bit energy, routing area usage, and wire delay of edge e, respectively. The estimation of P_e and D_e will be

described in SubsectionII.B.3. A_e is measured for an edge with unit capacity.

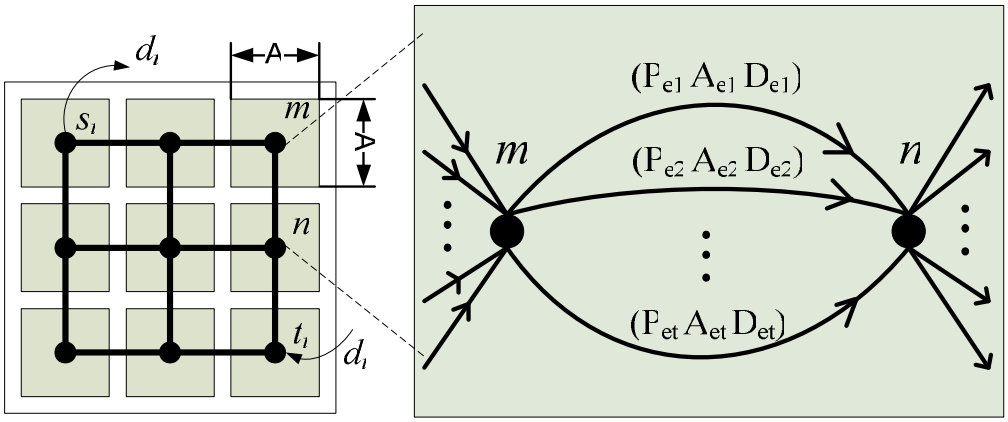


Figure II.3 Flow Graph with Wire Style Optimization

Assume there are k commodities among all pairs of nodes. For each commodity i, which starts at node s_i and ends at node t_i, we are given a communication demand $d_i > 0$. For each vertical or horizontal dimension $Grid(q)$, we are given a grid dimension constraint A so that the sum of all the edge width on this grid is no more than A. Let p_i be the set of paths for commodity i, and let $\mathbf{p} := \cup_i p_i$. Variable $f(p)$ denotes the amount of flow sent along path p, for every $p \in \mathbf{p}$.

In our work, we study the global average latency constraint in NoC design, which is suitable for those applications without critical path timing requirements. We assume the overall latency, which is the sum of the latencies for each flow, to be bounded by LT, so the average latency is $LT/\sum_{i=1}^{k} d_i$. Following is the MCF formulation for global average latency constrained minimum power NoC syntheis.

$$Min: \quad \sum_{j=1}^{k} \sum_{p \in p_j} \sum_{e \in p} f(p) \cdot P_e \tag{II.1}$$

$$s.t. \quad \sum_{j=1}^{k} \sum_{p \in p_j} \sum_{e \in p} f(p) \cdot D_e \leq LT \tag{II.2}$$

$$\forall 1 \leq j \leq k : \sum_{p \in p_j} f(p) \geq d_j \tag{II.3}$$

$$\forall q : \sum_{e \in Grid(q)} A_e \cdot \sum_{p : e \in p} f(p) \leq A \tag{II.4}$$

$$\forall p : f(p) \geq 0 \tag{II.5}$$

The objective is to minimize total NoC power consumption, which is the sum of all the communication flows over per-bit power consumption of all the edges (as in Equation (II.1)). In constraint (II.2), we ensure that global latency requirement is satisfied. NoC average latency can be derived by dividing global latency by total communication demands. Constraint (II.3) guarantees that communication demand for each sender/receiver pair is satisfied. Constraint (II.4) states that the total routing channel dimension is limited by area budget A on every grid area $Grid(q)$ of the routing channel.

II.B.2 Isomorph-Free Exhaustive Topology Generation

The search space of NoC topologies is extremely huge. Even excluding those topologies who are isomorphic, the number of the possible topologies still approaches $O(2^{n^2})$ for n nodes. In our methodology, we restrict the topology search space to *regular topologies*, where each row and column have identical connections. In this way, the number of combinations is significantly reduced, and this regularity makes circuit design and layout easier.

Regular topologies cover a wide range of popular network topologies. We find that any $2n$-dimensional binary hypercube can be mapped to a regular topology, and most of the popular NoC topologies, such as two-dimensional mesh and torus, high-dimensional mesh and torus, octagon, and twisted cube, etc., can be mapped to regular topologies.

We generate connected topologies on n nodes using *nauty* [39][61]. We set MAX_DEGREE as an upper bound on node degree. The maximum node degree limits network router input/output ports, which may dramatically increase the network router area and its power consumption.

Table II.1 lists the number of connected topologies on n nodes with different MAX_DEGREE. For a set of 8 nodes, when MAX_DEGREE equals to 4, there are 1929 distinct topologies.

Table II.1 # of Isomorph-Free Topologies

MAX_DEGREE	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$
3	6	10	29	64	194
4	6	21	78	353	1929

After we generate all connected Isomorph-Free topologies on n nodes, we enumerate all linear placements of them. Figure II.4 shows an example of two mappings of a ring structure onto a row of four tiles. Different linear placements lead to different NoC power consumption and delay time.

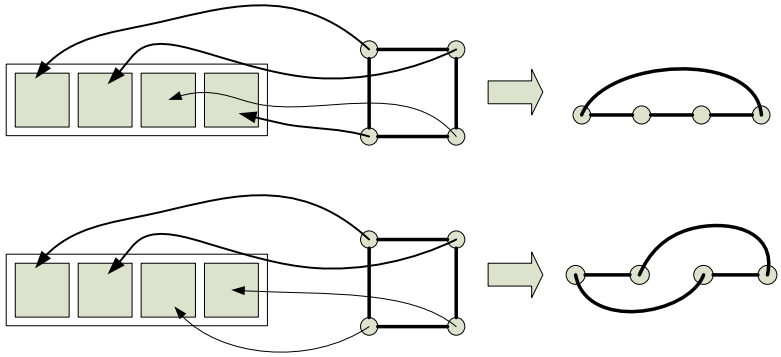


Figure II.4 Example of Linear Placements

Due to symmetry of certain subgraphs, different placements may correspond to the same mapping. We use link bit vector to remove duplicated placements. As shown in Figure II.5, there are six possible links on four nodes, hence a 6-bit link bit vector can represent a placement on four nodes. We then use an array of link bit vectors to keep track of all exist placements and remove those duplicated ones in our mapping algorithm.

Since total wirelength is tightly correlated to NoC power consumption, we only consider the placements with good quality, i.e. with small total wirelength. When generating placements, we set an upper bound for total wirelength. We map a topology to only those placements whose total wirelength is no more than the threshold (as seen in Table II.2) times the minimum wirelength of that topology.

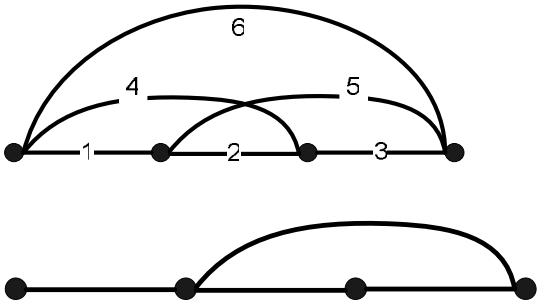


Figure II.5 Link Bit Vector to Represent a Placement

Our experiments show that this heuristic strategy works fine, for a 4×4 NoC, the placements with minimum wirelength always consume minimum power.

Once we generate all the placements on a row/column of on-chip tiles, we can duplicate it to all rows/columns to generate the final NoC topologies. Table II.2 gives the number of final NoC topologies with MAX_DEGREE=3 for one dimensional and maximum node degree of 6 for two dimensional $n \times n$ NoC. Our following power evaluation experiments are all based on these topologies.

Table II.2 Number of Regular Topologies on n by n NoC

size	4×4	5×5	6×6	7×7	8×8
threshold	2.0	1.5	1.2	1.1	1.0
Number of regular topopogies	36	254	534	1306	2092

II.B.3 Power and Delay Models

Interconnects and network routers are two main contributors to NoC power consumption and communication latency. We adopt the concept of *bit energy* proposed in [58] to represent energy consumption when one bit of data is transported through the interconnects or routers. For each network link e, we assume P_e represents bit energy on link e and the corresponding router, and D_e represents delay on link e and the corresponding router.

$$P_e = P_w + P_r$$

$$D_e = D_w + D_r$$

where P_w and P_r are bit energy on interconnects and routers, D_w and D_r are delay of unit flow on interconnects and routers, respectively. When a flow of amount f passes the link and the corresponding router, the power consumption is:

$$P = P_e \cdot f$$

and latency is:

$$D = D_e \cdot f$$

Interconnect Wires

To achieve high performance low power, many wire technologies have been proposed for on-chip interconnects, such as RC wires with repeated buffers, and on-chip transmission lines, etc.

Since different types of interconnect wire styles have different trade-offs on power consumption, communication latency and area resources, we assume that each on-chip network link can be composed of multiple types of wire styles. We assume four types of wire styles are available for interconnects, namely, *RC* wires with repeated buffers with wire pitch varying from 1×, 2×, and 4× minimum global wire pitch, and on-chip transmission line with wire pitch equaling to $16um$.

For RC wires with repeated buffers, we assume P_w and D_w are proportional to wirelength, i.e. P_w = per grid length big energy × wire length, and D_w = per grid length delay × wire length. For on-chip transmission line, comparatively large setup costs should be added to D_w and P_w. We use transmission line model proposed by Chen et *al.* [15] to estimate transmission line bit energy and delay.

Table II.3 lists bit energy and delay per grid length ($2mm$) of these four types of wire styles in $0.18um$ design technology. The supply voltages, wire geometries and device parameters are from ITRS [62]. For RC wires with repeated buffers, the repeaters are inserted to minimize wire delay. Setup costs of 50ps and 4.4pJ/bit are added to D_w and P_w for transmission lines.

Table II.3 Delay Model of Wires

wire type	RC-1x	RC-2x	RC-4x	T-line
P_w (pJ/bit)	2.68	2.15	1.99	0.15
D_w (ns)	0.127	0.112	0.100	0.020

Network Routers

To estimate router bit energy P_r, we use a power simulator called *Orion* [55]. We assume 1GHz frequency, 4-flit buffer size, 128-bit flit size. When the number of router input/output ports increases, P_r increases almost linearly. We use the router delay model proposed by Peh et *al.* [46] to estimate NoC router delay.

Table II.4 shows bit energy and latency of routers in $0.18um$ technology node. When router input/output ports increase, P_r increases almost linearly, and D_r increases in a slower pace.

Table II.4 Power Model of Routers

ports	2	3	4	5	6	7	8
P_r (pJ/bit)	0.22	0.33	0.44	0.55	0.66	0.78	0.90
D_r (ns)	0.599	0.662	0.709	0.756	0.788	0.819	0.835

II.C MCF Approximation Algorithm

We use *polynomial time approximation schemes (PTAS)*, which can obtain $(1+\epsilon)$ optimal solutions in polynomial time, where ϵ is an input accuracy threshold. The PTAS for classic MCF problem has been studied in recent works [22] and [31]. We propose algorithms that are based on the previous framework but adaptive to our NoC synthesis problem. In addition, we propose the interval estimation technique to speed up the process.

II.C.1 Baseline Algorithm

First, our baseline algorithm finds the largest λ such that there is a multicommodity flow which routes at least λd_i units of commodity i, where the wiring area for each grid does not exceed the grid area A, and total power and latency are constrained by the budget PW and LT respectively. The problem formulation is as follows.

$$Primal: \tag{II.6}$$

$$Max: \quad \lambda \tag{II.7}$$

$$\forall j: \quad \sum_{p \in p_j} f(p) \geq \lambda d_j \tag{II.8}$$

$$\forall q: \quad \sum_{e \in Grid(q)} A_e \sum_{p:e \in p} f(p) \leq A \tag{II.9}$$

$$\sum_{i=1}^{k} \sum_{p \in p_i} \sum_{e \in p} f(p) P_e \leq PW \tag{II.10}$$

$$\sum_{i=1}^{k} \sum_{p \in p_i} \sum_{e \in p} f(p) D_e \leq LT \tag{II.11}$$

$$\forall p: \quad f(p) \geq 0 \tag{II.12}$$

The following is the dual problem. Besides a variable X_e for each grid area constraint and a variable Z_j for every commodity demand constraint, the dual problem has another two variables, ϕ_p corresponding to the power budget constraint, and ϕ_d corresponding to the latency budget constraint:

$$Dual: \tag{II.13}$$

$$Min: \quad A \sum_{q=1}^{n} X_q + PW\phi_p + LT\phi_d \tag{II.14}$$

$$\forall j, \forall P \in \rho: \quad \sum_{e \in P} A_e \sum_{e \in Grid(q)} X_q + \sum_{e \in P} P_e \phi_p \tag{II.15}$$

$$+ \sum_{e \in P} D_e \phi_d \geq Z_j \tag{II.16}$$

$$\sum_{j=1}^{k} d_j Z_j \geq 1 \tag{II.17}$$

$$\forall q: \quad X_q \geq 0 \tag{II.18}$$

$$\forall j: \quad Z_j \geq 0 \tag{II.19}$$

Assume the subroutine $mcf(G, d, LT, PW)$ could return such a λ, the power minimization MCF algorithm finds the minimum power that satisfying $\lambda \geq 1$ by recursively binary search, as shown in Algorithm 1, where we use λ_{max} to denote the concurrent value without power budget constraint, i.e. $PW = \infty$.

Algorithm 1 Power Minimization MCF Algorithm

Input: graph G, demand d, latency constraint LT, threshold ϵ
Output: $(1 + \epsilon)$ optimal power
set $\lambda_{max} \leftarrow mcf(G, d, LT, \infty)$
set lower bound $lb \leftarrow 0$
upper bound $ub \leftarrow$ total power under λ_{max}
while $(ub - lb)/ub > \epsilon$ **do**
    $\lambda \leftarrow mcf(G, d, LT, (lb + ub)/2)$
    if $\lambda \geq 1$ **then**
        $ub \leftarrow (lb + ub)/2$
        else $lb \leftarrow (lb + ub)/2$
    end if
end while
Output ub

$mcf(G, d, LT, PW)$ subroutine iteratively updates the primal and dual values till the gap is small enough. The primal value λ is updated by adjusting the flows. To calculate dual values, we define edge length as:

$$l(e) := A_e \sum_{q:e \in Grid(q)} X_q + PW \cdot \phi_p + LT \cdot \phi_d \tag{II.20}$$

So dual is equivalent to:

$$Min : \frac{A \sum_{q=1}^{n} X_q + PW\phi_p + LT\phi_d}{\sum_{j=1}^{k} d_j \cdot dist(j)} \tag{II.21}$$

where $dist(j)$ is the shortest path from the source to the sink of commodity j under the length function $l(e)$. The process is described in Algorithm 2.

Algorithm 2 $(1-\delta)$ Maximum Concurrent Flow Algorithm

Input: graph G, demand d, latency constraint LT, power budget PW, threshold δ

Output: $(1-\delta)$ optimal maximum concurrent value λ

$\forall q$, set $f(e) \leftarrow 0$, $X_q \leftarrow X_0$, $\phi_p \leftarrow X_0$, $\phi_d \leftarrow X_0$

$l(e) \leftarrow A_e \sum_{q:e \in Grid(q)} X_q + PW \cdot \phi_p + LT \cdot \phi_d$

while $A \sum_{q=1}^{n} X_q + PW\phi_p + LT\phi_d \leq 1$ **do**

  for each commodity j **do**

    $rd_j \leftarrow d_j$

    while $rd_j > 0$ **do**

      Route f units of flow from s_i to t_j along the shortest path P

      $f(e) \leftarrow f(e) + f$, $\forall e \in P$

      $X_q \leftarrow X_q(1 + \frac{\delta}{3} \cdot \frac{\sum_{e \in Grid(q)} A_e f(e)}{A})$

      $\phi_p \leftarrow \phi_p(1 + \frac{\delta}{3} \cdot \frac{\sum power}{PW})$, $\phi_d \leftarrow \phi_d(1 + \frac{\delta}{3} \cdot \frac{\sum latency}{PW})$

      $l(e) \leftarrow A_e \sum_{q:e \in Grid(q)} X_q + PW \cdot \phi_p + LT \cdot \phi_d$

      $rd \leftarrow rd - f$

    end while

  end for

  compute primal λ by scaling down all $f(e)$ subject to area, power and latency constraints

  compute dual $D \leftarrow \frac{A \sum_{q=1}^{n} X_q}{\sum_{j=1}^{k} d_j \cdot dist(j)}$

end while

return λ

The above algorithm proceeds in phases and each phase is composed of k iterations. In iteration j of the i^{th} phase we route d_j units of commodity j in a sequence of steps. In each step, a shortest path P from source s_j to sink t_j is computed using the current length function. The dual variables X_q are updated as

$$X_q = X_q(1 + \frac{\delta}{3} \cdot \frac{\sum_{e \in Grid(q)} A_e f(e)}{A}) \tag{II.22}$$

and ϕ_p and ϕ_d are updated in the similar fashion.

Regarding the convergence of Algorithm 2, by carefully choosing the initial values X_0, we have the following theorems:

Theorem 1: When the algorithm terminates, $\frac{\lambda}{D} \geq 1 - \delta$.

Theorem 2: The algorithm runs in $O(\delta^{-2}|E|^2)$.

The first theorem guarantees the $(1 - \delta)$ optimality and the second one shows the efficiency. The proofs are similar to those in [22] and [31]. However, the formulations in the previous works all treat the capacity constraints are on the edges; while here, the constraints are on a set of edges, since multiple wires can be routed through a single grid. Therefore to update the dual variables, we need to modify the original formula in [22] and [31]

$$l(e) = l(e)(1 + \frac{\delta}{3} \cdot \frac{f(e)}{c(e)}) \tag{II.23}$$

where $f(e)$ is the total flow on edge e and $c(e)$ is the edge capacity, to the one as in Equation (II.22). This is from the intrinsic spirit of the dual variable updating scheme: the dual variables reflect the congestion level of the edge or grid, therefore we always update it using the ratio of the flow versus the total available resource. In addition, the power and latency constraints can be viewed as two "pseudo edges" with capacities W and L, so ϕ_p and ϕ_d have the similar update formula.

It is worth noting that the dual variables X_q are associated with a set of edges instead of a single edge, therefore we need to apply formula (II.20) to further compute the edge lengths. Sometimes this results complicated cases. Refer

to Figure II.6, consider a path consisting of two edges (a, e) and (e, c), the lengths should be updated as

$$l((a,e)) = A \cdot (X_1 + X_2 + X_3 + X_4) + W\phi_p + L\phi_d$$

$$l((e,c)) = A \cdot (X_3 + X_4) + W\phi_p + L\phi_d$$

Note that X_3 and X_4 do need to be counted twice in the path, since the path crosses them twice and each time the flow contributes to the congestion individually. This issue should be carefully handled in the implementation.

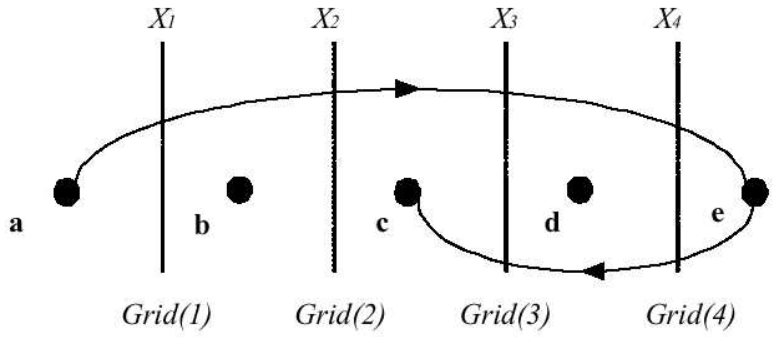


Figure II.6 Length Function on Edge

II.C.2 Interval Estimation Optimization

While Algorithm 1 needs to obtain MCF solutions with $(1 + \epsilon)$ optimal power values, Algorithm 2 returns us $(1 - \delta)$ optimal concurrent flow. Therefore the values of ϵ and δ are associated "pseudo polynomially": δ has to be determined by both the value of ϵ and the unit edge cost P_e, which leads to extremely slow convergence in some pathological cases.

We propose a heuristic interval estimation technique to speedup the process. The idea is to estimate the new lower bound lb' and upper bound ub' while performing the approximation algorithms, and break once $ub' - lb' \leq (ub - lb)/2$ in each step of the binary search scheme.

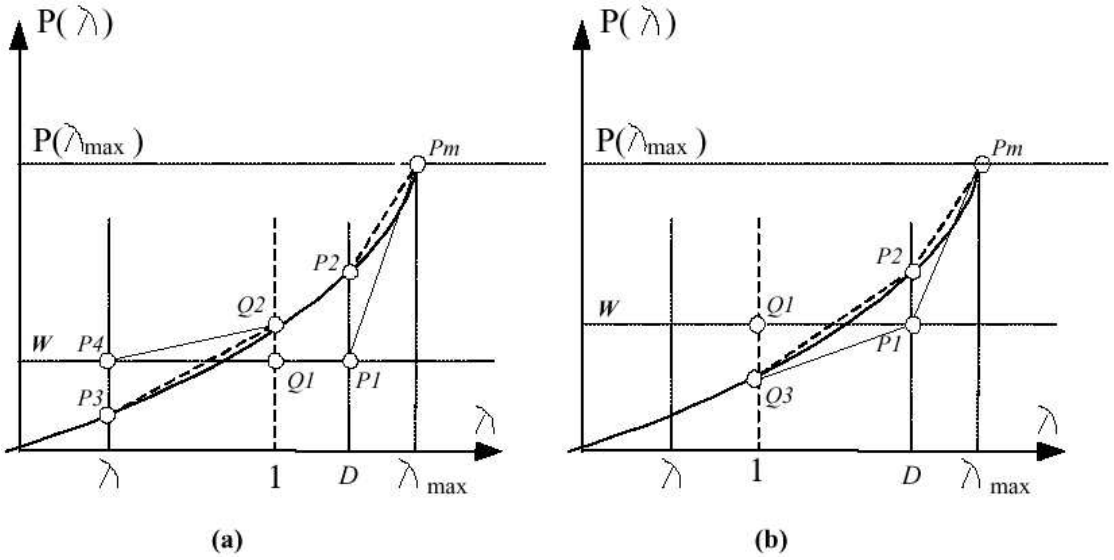


Figure II.7 Interval Estimation

We define a function monotonically increasing $P(\lambda)$, where λ is the concurrent flow and $P(\lambda)$ is the minimum power under this concurrent flow (therefore $P(1)$ is the target optimal value). The curve is shown in Fig II.7. Furthermore, we have the following lemma:

Lemma: $P(\lambda)$ is a convex function.

Proof: For a specific λ_1, the minimum power should be $P(\lambda_1)$; scaling down all the flows by half, the concurrent flow would be $\frac{\lambda_1}{2}$, and the power is $\frac{P(\lambda_1)}{2}$. On the other hand, when the concurrent flow is $\frac{\lambda_1}{2}$, the minimum power should be $P(\frac{\lambda_1}{2})$, therefore we have $P(\frac{\lambda_1}{2}) \leq \frac{P(\lambda_1)}{2}, \forall \lambda_1 \leq \lambda_{max}$. So the function is convex.

We use the following theorem to estimate the lower bound lb' and upper bound ub':

Theorem 3: Given a feasible primal value λ and a feasible dual value D under the power budget PW, we have

$$PW - s \cdot (D - 1) \leq P(1) \leq PW + s \cdot (1 - \lambda) \tag{II.24}$$

where

$$s = \frac{P(\lambda_{max}) - PW}{\lambda_{max} - D} \tag{II.25}$$

Hence, $lb' \leftarrow \max\{lb', PW - s \cdot (D-1)\}$, $ub' \leftarrow \min\{ub', PW + s \cdot (1-\lambda)\}$

We sketch the proof for $P(1) \leq PW + s \cdot (1-\lambda)$ here. Refer to Figure II.7 (a), let the lines $x = \lambda$, $x = 1$, $x = D$ and $x = \lambda_{max}$ intersect the function curve at P_3, Q_2, P_2 and P_m, and $x = 1$, $x = 1$ and $x = D$ intersect $y = PW$ at P_4, Q_1 and P_1 respectively (we use x and y to denote the two axes). We then have

$$P(1) = PW + S_{P_4Q_2} \cdot (1-\lambda) \tag{II.26}$$

where $S_{P_4Q_2}$ is the slope of the line P_4Q_2. And, it is easy to identify that $S_{P_4Q_2} \leq S_{P_3Q_2} \leq S_{P_2P_m} \leq S_{P_1P_m}$, by the property of the convex function. And since $s = S_{P_1P_m}$, we have $P(1) \leq PW + s \cdot (1-\lambda)$. Similarly, $PW - s \cdot (D-1) \leq P(1)$ can be proven by the similar approach, as shown in Figure II.7 (b).

According to Theorem 3, Algorithm 1 and 2 can be improved as Algorithm 3 and Algorithm 4.

Algorithm 3 Modified Power Minimization MCF Algorithm

1: **Input:** graph G, demand d, latency constraint LT, threshold ϵ
2: **Output:** $(1+\epsilon)$ optimal power
3: As in Algorithm 1 Steps 3–5
4: **while** $(ub - lb)/ub > \epsilon$ **do**
5: $\quad (lb', ub') \leftarrow mcf(G, d, LT, (lb + ub)/2)$
6: $\quad lb \leftarrow lb'$; $ub \leftarrow ub'$
7: **end while**
8: Output ub

II.D Experimental Results

Our experiments are on NoC in $0.18um$ design technology. We assume that grid length are $2mm$, and communication demands are evenly distributed, i.e. the bandwidth requirements between every pair of tiles are 1Gb/s. The experiments are based on power/delay parameters described in subsectionII.B.3. We

Algorithm 4 Modified Maximum Concurrent Flow Algorithm

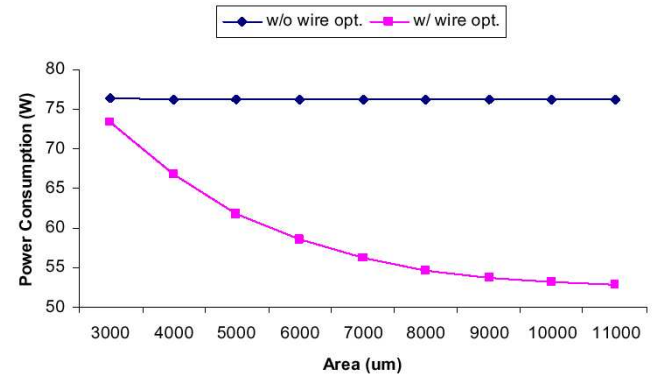
1: **Input:** graph G, demand d, latency constraint LT, power budget PW, threshold δ
2: **Output:** new lower bound lb' and upper bound ub'
3: As in Algorithm 2 Steps 3–4
4: $lb' \leftarrow lb$; $ub' \leftarrow ub$
5: **repeat**
6: As in Algorithm 2 Steps 6–18
7: $lb' \leftarrow \max\{lb', PW - s \cdot (D-1)\}$
8: $ub' \leftarrow \min\{ub', PW + s \cdot (1-\lambda)\}$
9: **if** $ub' - lb' \leq (ub - lb)/2$ **then**
10: **return** (lb', ub')
11: **end if**
12: **end repeat**

use the topology library generated in subsectionII.B.2 as candidate topologies for design selection. In MCF approximation algorithm, we set error tolerance ϵ to 1%. In following experiments, we show the impact of wire style optimization, topology selection and tradeoffs between power/latency optimization in NoC design. Since each grid has the same vertical and horizontal dimension, for convenience, we use only the vertical dimension to represent the area budget. This is why the unit of area in our experiments is um.

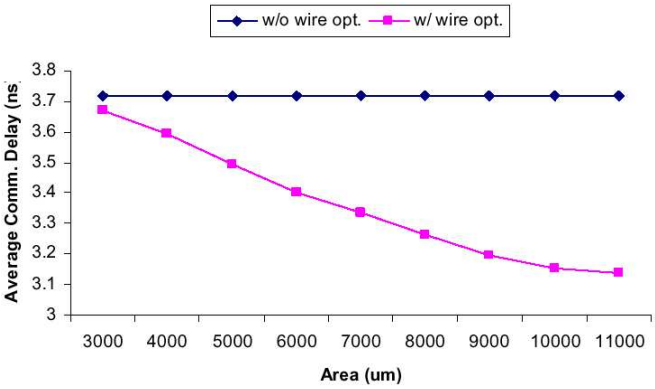
II.D.1 Wire Style Optimization

We first demonstrate the power/latency improvement by wire style optimization. Without wire style optimization, we assume only the basic RC wires with repeated buffers available for on-chip interconnects, whose wire pitch is 1× minimum global pitch. With wire style optimization, other three types of wires (Table II.3) are also available for on-chip interconnects.

For 8x8 torus networks, Figure II.8 shows its power/latency improvement



(a) Power Consumption Comparison w/ and w/o Wire Style Optimizaiton



(b) Communication Latency Comparison w/ and w/o Wire Style Optimizaiton

Figure II.8 Impact of Wire Style Optimization

under various on-chip area resource. When more area is available for wire style optimization, more power/latency savings can be seen. The lower bound of area resource is $3000um$ so that all communication demands be satisfied; when area resource increased to $11000um$, wire style optimization reaches the maximum impact, up to 30.7% power improvement (from 76.2W decreases to 52.9W) and up to 15.6% latency improvement (from 3.72ns to 3.14ns) can be achieved. When area resource decreases to the bottleneck, 3.8% power saving and 1% latency saving are still seen from wire style optimization. We observe that at flow congested area, minimal pitch wire style is used. However, at un-congested on-chip area, we still

can use power efficient wire styles to reduce NoC power consumption and latency.

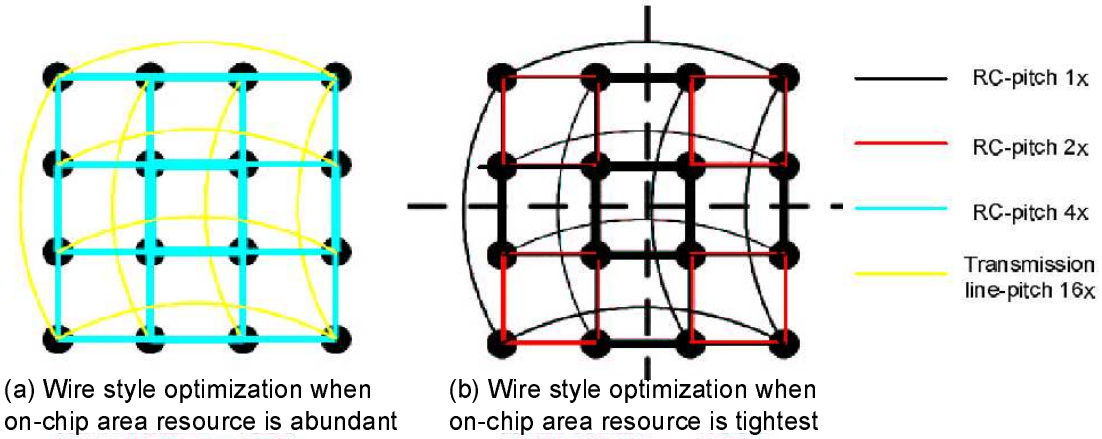


Figure II.9 Details of Wire Style Optimization

Figure II.9 shows details of wire style optimization, e.g. the types of the wire styles for interconnects and their capacities. The different types of lines represent different wire styles, as shown in legend. The line width represents the wire capacity. For a 4×4 torus, Figure II.9(a) shows the wire style assignment under loose area constraints. Due to the relatively large available on-chip area resources, transmission lines are selected for long interconnects, and RC wires with repeated buffers with largest spacing (wire pitch is equal to four times minimum pitch) are selected for short connections. Figure II.9(b) shows the wire style assignment under tightest area constraints. Since the communication reaches the maximum capacity, for all interconnects on the two congested cuts shown by dot lines, only the RC wire with minimal pitch is selected because it provides the highest cross-section bandwidth. For those un-congested links, wider wires with lower power consumption are selected.

II.D.2 Topology Selection

Our design methodology selects the optimal topologies from 8×8 NoC topology library, which consume minimum power and satisfy latency constraints.

In this section, we compare these optimal topologies with traditional topologies, such as mesh, torus and hypercube. Same as in subsection II.D.3, we set the latency constraints by loosening the minimum latency by up to 10%.

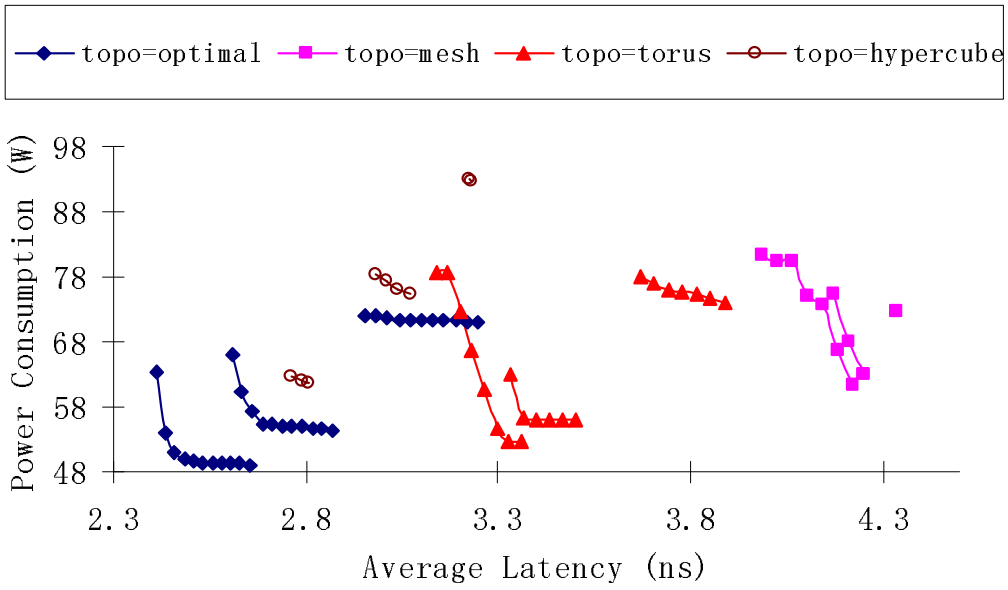


Figure II.10 Power latency tradeoffs among various topologies

Figure II.10 shows comparison among these four types of topologies under different budgets. The x-axis represents average latency. The y-axis represents power consumption. Each group includes 3 curves , 11000um, 7000um, and 3000um, which represents loose, moderate and tight area constraints, respectively, which represent latency constrained minimum power consumption for a certain topology under various area budgets.

For a certain topology, since larger latency constraints lead to smaller power consumption, and vice versa, we pick the point with minimum power latency product for a quantitative comparison, as shown in Table II.5.

The first two columns list area budgets and topologies. Column 3-5 show latency, power consumption, and power latency product for certain topology under given area budgets. The sixth column of the table lists the improvement in terms of power latency production, when compare our selected optimal topology with

Table II.5 Topology Comparison

area (um)	**topo**	**L (ns)**	**P (W)**	**P*L (W*ns)**	**Impv. (%)**
3000	mesh	4.34	72.7	315.2	**26.7**
	torus	3.74	76.1	284.7	**18.9**
	cube	3.23	92.8	299.8	**23.0**
	optimal	3.25	71.1	**230.9**	
7000	mesh	4.25	63.0	267.9	**44.5**
	torus	3.37	56.3	189.6	**21.5**
	cube	3.04	76.0	231.2	**35.6**
	optimal	2.69	55.4	**148.8**	
11000	mesh	4.22	61.2	258.3	**52.1**
	torus	3.33	52.7	175.3	**29.4**
	cube	2.76	62.6	173.1	**28.5**
	optimal	2.48	49.8	**123.8**	

mesh, torus and hypercube.

From the table, we observe that mesh is not a desirable topology for NoC of size 8x8. Compared with other topologies, its latency is quite large, because data packets need many hops to arrive the destinations. Also it lacks of long global links and doesn't make fully use of wire style optimization, so that when area budgets increase, its power consumption is not as good as torus. Torus and hypercube has their own advantages. In general, torus is better in terms of power consumption, since it has simpler network router architecture; hypercube is better in terms of latency, since it has a lot of shortcut links.

Our selected optimal topologies show big advantages over the other three traditional topologies. They have small power consumption and latency. In terms of power latency production, they achieve an improvement up to 52.1% compared to mesh and 29.4% compared to torus (area = $11000um$), and 28.5% compared to hypercube (area = $7000um$). Figure II.11 shows the connections on one raw of our selected optimal topologies under each area budget. Duplicating these connections to every row and column will generate the final topology design.

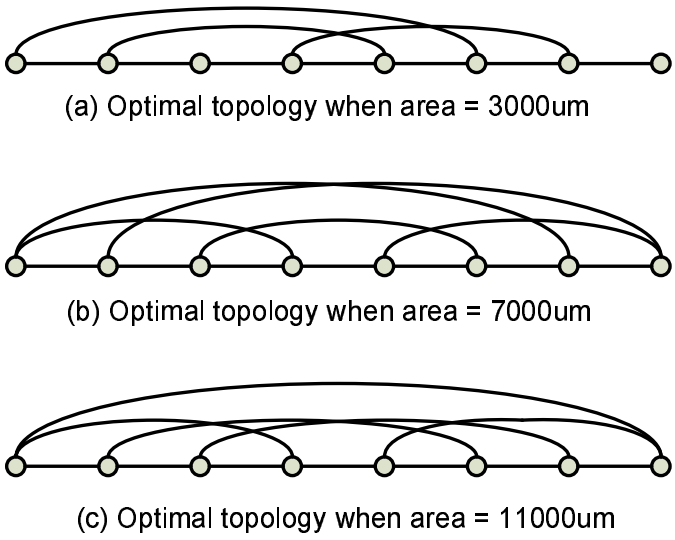


Figure II.11 Optimal topologies under various areas

II.D.3 Power Consumption and Latency Tradeoffs

To demonstrate tradeoffs between power consumption and average latency in 8×8 NoC design, we show power savings when a small amount of communication latency is sacrificed. First, we use MCF model to search the topologies with the minimum latency (no power optimization), then loosing this latency constraint by up to 10% and optimize NoC power consumption. Figure II.12 shows the results. The x-axis represents average latency. The y-axis represents power consumption. Each curve represents latency constrained minimum power consumption under certain area budget.

As area budgets increase, the curves move toward left-bottom due to wire style optimization, because those aggressively optimized but area-consuming wire styles, such as transmission lines, can be adopted to optimize both power and latency. When area increases from $3000um$ to $11000um$, minimum latency drops 18.3%, from 2.95ns to 2.41 ns; average power consumption drops 28.3%, from 71.4W to 51.2W.

The slopes of the curves indicate the power consumption reduce rate when communication latency is increased. Take the curve with area $11000um$ as

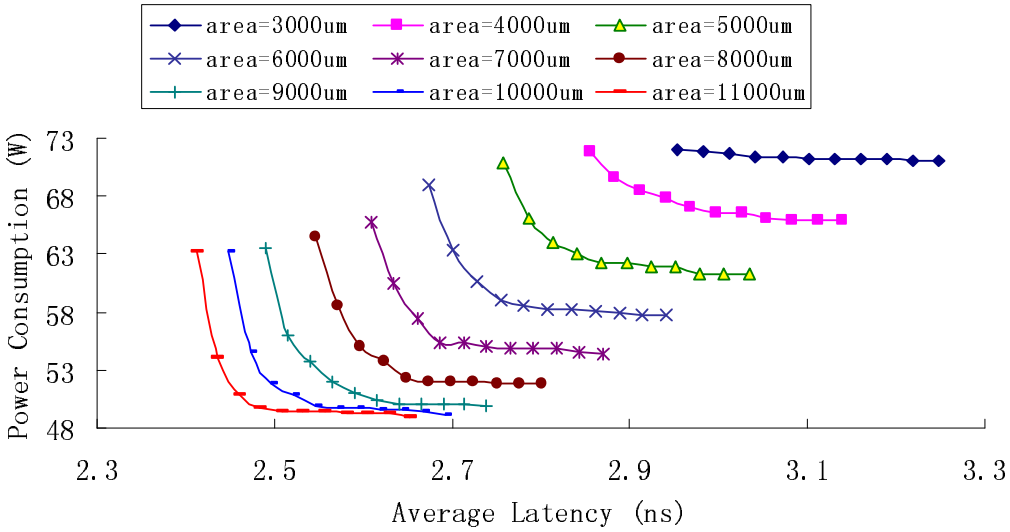


Figure II.12 NoC power and latency tradeoffs

example, when latency constraint is loosen 2%, from 2.41ns to 2.46 ns, the power consumption is reduced from 63.2W to 50.9W, which is a 19.4% improvement. When area is small ($3000um$), the curve is almost flat. This is because area resource becomes bottleneck and flow is congested on chip, so that loosing latency constraint will not bring much benefit.

II.D.4 Video Applications

We use four different video processing applications from [4] to search for the most power-efficient NoC topologies for them. These four high-end video-processing applications include: *Video Object Plane Decoder* (VOPD), *MPEG4 decoder*, *Picture-In-Picture* application (PIP) and *Multi-Window Display* application (MWD). The communications between the cores of these applications are presented in (Figure II.13), the unit is MB/s.

We manually map the four video applications to a 4×4 tile based NoC, and set their communication demands. Figure II.14 depicts the optimal topologies. Torus (Figure II.14(a)) is the most power-efficient for MPEG4. A partially connected topology (Figure II.14(b)) is the best for VOPD. For both PIP and MWD

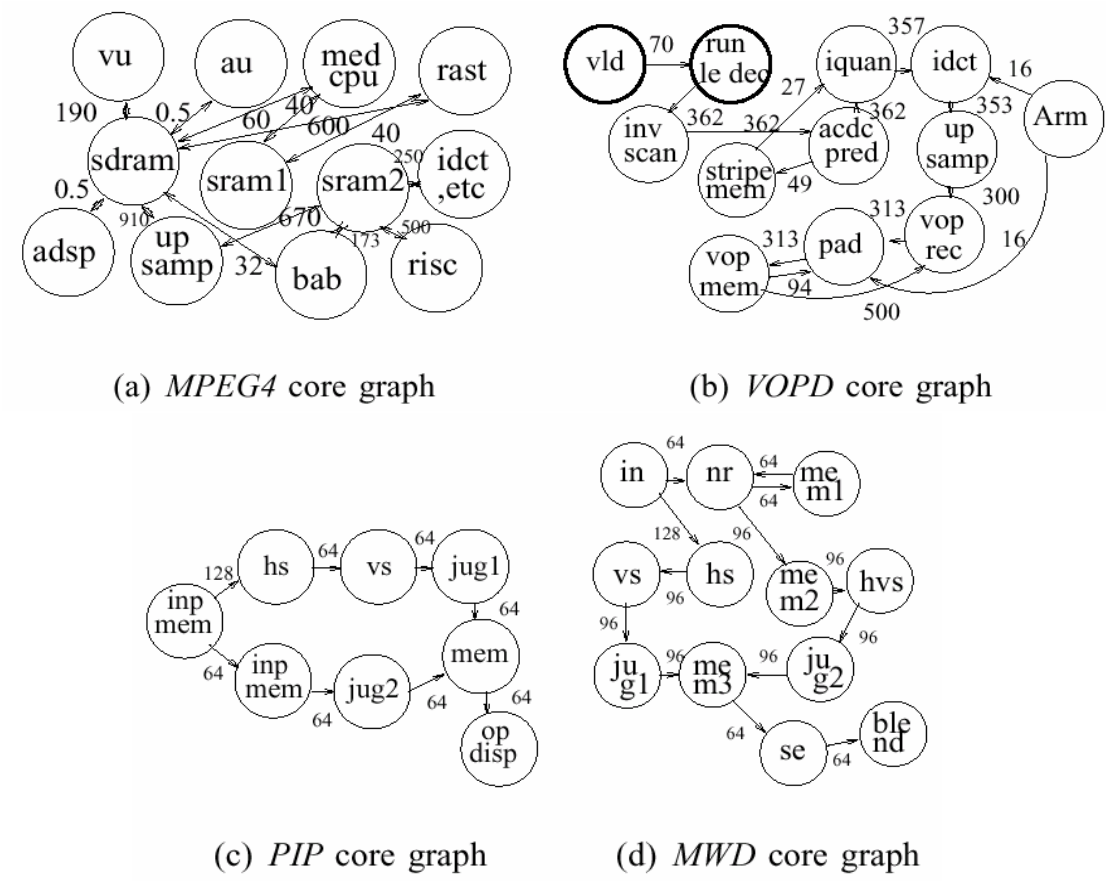


Figure II.13 Communication Graph of Four Video Applications

applications, mesh is the most power-efficient topology. Note that in VOPD (Figure II.13(b)), there are large communication demands of 500MB/s between cores "vop mem" and "vop rec", and 362MB/s between cores "inv scan" and "acdc pred". These particular high bandwidth requirements partially explain the optimality of the un-symmetric topology.

II.D.5 MCF Performance Improvement

To demonstrate the efficiency of the proposed algorithm, we conduct experiments to compare its CPU time with the LP solution produced by CPLEX, a commercial LP solver. We choose torus as the representative topology to make the comparison. We test the performance on 3000, 7000 and 11000 as small, moderate

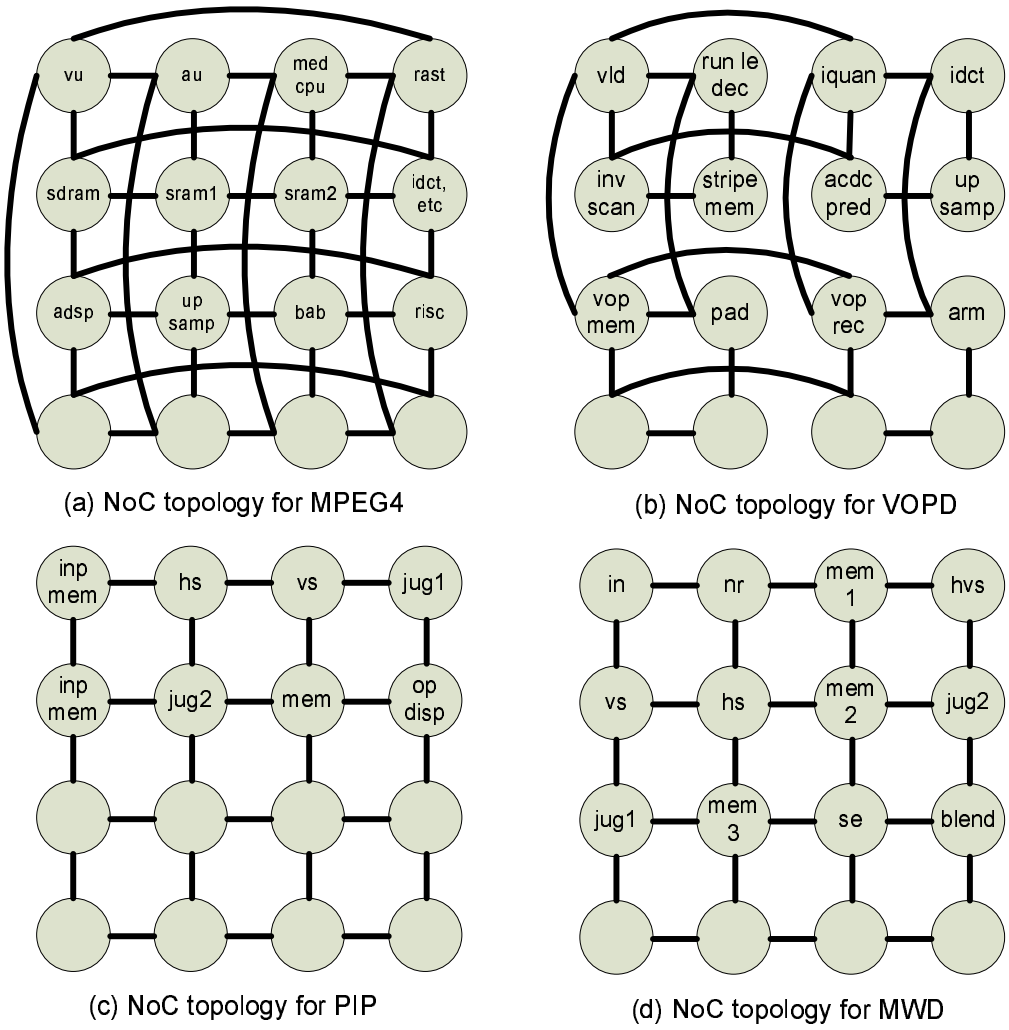


Figure II.14 The Most Power-Efficient Topologies for Four Video Applications

and large grid area, by scaling down them by the factor of $k^4/8^4$ for the $k \times k$ case, by approximating the communication demands to be k^4. All the experiments are conducted in a PC with 2.8 GHz CPU and 784MB memory, and CPLEX 9.1 is used. The detail results are shown in Table II.6, where columns 3–6 show the result values and CPU time (in seconds) of CPLEX and our approximation algorithm respectively, column 7 shows the gap between the approximate results and the optimal solutions, $(col5 - col3)/col3$, and column 8 shows our speedup, $col4/col6$.

The table shows that our proposed algorithm can obtain correct results within the 1% threshold, which is our input settings. Also, it is much faster than

Table II.6 MCF Performance Improvement

Size	Area	CPLEX		Approx.		Err	Speedup
		Obj	CPU	Obj	CPU	(%)	
5×5	473	6611	105	6652	11	0.62	9.55
	1069	5389	104	5430	11	0.76	9.45
	1679	5193	10	5234	12	0.78	0.83
6×6	950	16830	1496	16955	65	0.74	23.02
	2215	13195	1910	13298	29	0.78	65.86
	3481	12580	291	12683	29	0.82	10.03
7×7	1759	36860	9963	37156	78	0.80	127.73
	4104	28405	15040	28641	46	0.83	325.96
	6488	27464	8280	27689	56	0.82	147.86
8×8	3000	N/A	N/A	73315	113	N/A	N/A
	7000	N/A	N/A	56207	48	N/A	N/A
	11000	N/A	N/A	52915	62	N/A	N/A

the LP solver, and becomes more and more significant when the size becomes larger: in the 7×7 cases, it has been more than 100 times faster than CPLEX. In 8×8 cases, the approximation method also runs fast, while CPLEX is too slow to produce any results.

II.E Summary

We study the tradeoffs between NoC power efficiency and average latency. By adopting a MCF formulation, we are able to reduce power consumption of NoC under given latency constraint, through simultaneous optimization of network topologies and wire styles. Experimental results suggest that for NoC of size 8×8 (1) Power and latency co-optimization is critical in NoC design. With 2% latency overhead, up to 19.4% power savings can be seen. (2) compared with mesh, torus and hypercube topologies, our optimized design can improve power latency product by up to 52.1%, 29.4% and 35.6%, respectively.

One of the future research directions is to extend this work to heterogeneous NoC design, where tiles may have different sizes and irregular locations. Heterogeneous NoC architecture enables more efficient design space exploration,

but brings much more computational complexity. We will study the efficient algorithms for such MCF formulations.

II.F Acknowledgement

This chapter includes the contents of two published papers. "Communication Latency Aware Low Power NoC Synthesis," by Y. Hu, Y. Zhu, H. Chen, R. Graham, C.K. Cheng. "Physical Synthesis of Energy-Efficient NoCs Through Topology Exploration and Wire Style Optimization," by Y. Hu, H. Chen, Y. Zhu, A. A. Chien, C.K. Cheng. The dissertation author was the primary researcher and co-author of both papers.

III

FPGA Global Routing Architecture Optimization

In this chapter, we investigate FPGA global routing architecture to reduce its energy and switch area-efficiency. The optimization methodology uses MCF formulations to evaluate the impact of topology optimization and wire style optimization on FPGA routing architecture. Although this approach is similar to what is used in NoC synthesis discussed in the previous chapter, FPGA routing architecture optimization has its own unique design constraints, concerns and metrics. This chapter discusses these specific issues. The chapter is organized as follows. Section III.1 explain our motivation to optimize FPGA global routing architecture. Section III.2 describes our design methodology, including an improved CAD flow we use to generate optimized FPGA routing architecture, and the core components in the flow. Section III.3 presents the experimental results. We summarize our study for FPGA global routing architecture optimization in Section III.4.

III.A Motivation

Low energy and small switch area usage are two of the important design objectives in FPGA global routing architecture design. Compared to ASIC de-

sign, where circuit is interconnected by metal wires, FPGAs connect circuit via programmable switches. Although these programmable switches bring flexibility to FPGA architectures, it pays the price of more energy and on-chip area usage, making FPGAs less favorable in energy-critical applications such as portable devices [8]. Therefore, we study how to effectively reduce energy and switch area usage of FPGA routing architectures through a multicommodity flow (MCF) model based CAD flow.

Traditionally, People adopt Mesh topology for FPGA global routing architecture due to its simplicity and easy to layout, then perform global routing algorithms such as PathFinder [40] to determine number of routing tracks in each routing channel. However, as feature sizes shrink and die sizes grow, interconnect delay and energy consumption may becomes a serious issue for these traditional Mesh-based architectures [47]. Also, as pointed out in [21], Mesh routing schemes suffer from unscalable switching area requirements. Therefore, exploring more complex topology is a promising approach to effectively optimize the energy and switch area of FPGA routing architectures. Segmentation distribution is one of the effective techniques to explore FPGA topology space. Brown *et al.* investigated differentiation segmentation distributions in order to optimize the speed and area [10][11][34]. Chow *et al.* performed a similar study on the impact of segmentation distributions on circuit routability [16]. In [36], Lee *et al.* explored more versatile wire segmentations and richer connections of FPGA routing architecture to improve routability and reduce delay.

Compared with topology optimization which has been widely studied for many years by many researchers, wire style optimization emerges only in recent years as a result of rapid advances in signaling interconnect technologies. A few works explored to introduce multiple signaling technologies to low power network-on-chip (NoC) design [28], and communication latency constrained low power NoC design [29]. Other works studied bus-based connections to improve FPGA switch area density [59][60].

In this dissertation, we address two questions regarding FPGA global routing architecture design. First, what kind of routing topologies should we adopt for an FPGA architecture? FPGA routing topology is closely related to segmentation distribution. Different distribution of long wires leads to different topologies. Therefore, we study how to distribute the long wires to improve FPGA routing architecture regarding to specific design objectives. Second, once FPGA topologies are selected, how many tracks and what wiring technology should be assigned to each channel? Traditional FPGA routing architectures simply adopt single wiring technology and assign uniform number of tracks to all channels. How much benefit we can achieve by allowing more flexible track assignment and wire style optimization?

We have two design metrics in our investigation of FPGA routing architecture. One is interconnection energy consumption, the other is the total on-chip area consumed by switches in switch boxes, which is indicated by the total number of switches in switch boxes. Notice that our methodology is capable of simultaneously optimizing multiple design issues, and can easily be extended to integrate other design metrics, such as communication delay, into the optimization frame.

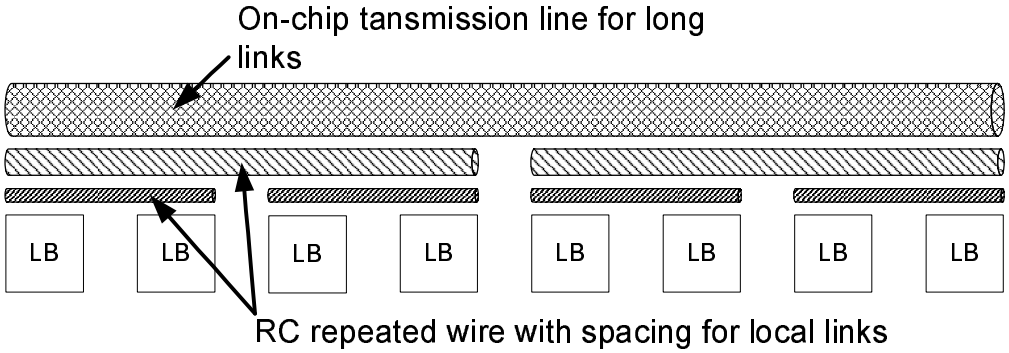


Figure III.1 Topology and Wire Style Optimizations in FPGA Routing Architectures

The same as in NoC optimization, our approaches for FPGA routing architecture optimization are still through topology optimization and wire style optimization. Figure III.1 shows an example of topology and wire style optimiza-

tions in FPGA routing architectures. There are wire segments of various lengths, and different wire segments can be implemented with different wire styles. Our purpose is to schedule these various wire styles and topologies in FPGA routing architecture so that we can make best use of available on-chip area to optimize overall energy consumption and efficient switch area density.

III.B Design Methodology

We generate and evaluate FPGA routing architectures by improving an existing CAD flow similar to that used in commercial FPGAs to automatically implement circuits. The design methodology is mainly to use some representative netlists, which are extracted from existing benchmark circuits, as input workload to MCF formulations to evaluate a set of candidate FPGA topologies, and generate the corresponding channel capacity distribution and wire style optimization.

In the following sections, we first introduce our improved CAD flow. We then describe each major part in the CAD flow, specifically, our MCF formulations for various FPGA routing architecture optimization, the generation of candidate FPGA topologies, and the generation of representative netlists.

III.B.1 An Improved CAD Flow

Figure III.2 summarizes the CAD flow. We have the MCNC benchmark circuits [57] as input. First, we use SIS [50] to perform technology independent logic optimization on each of the circuit. Next, these circuits are technology-mapped by FlowMap [17] into four-input look-up tables (4-LUTs), respectively. We then use VPack [17] to pack every sixteen 4-LUTs into a bigger block logic element (BLE). The netlist of these BLEs are then read into VPR tool, which places the circuits and outputs the new netlist with optimized placements. At this point, we have placed netlist for each benchmark circuit.

We then implement a representative netlist generator to generate some representative netlists by extracting the characters of the input benchmark circuits.

These representative netlists are supposed to reflect the congestion distribution of the benchmark circuits, hence effectively guide the design of FPGA routing architectures to fit for the largest class of the benchmarks. Meanwhile, we use a topology generator to generate a set of candidate FPGA topologies that with reasonable aggressive optimization on segmentation distribution.

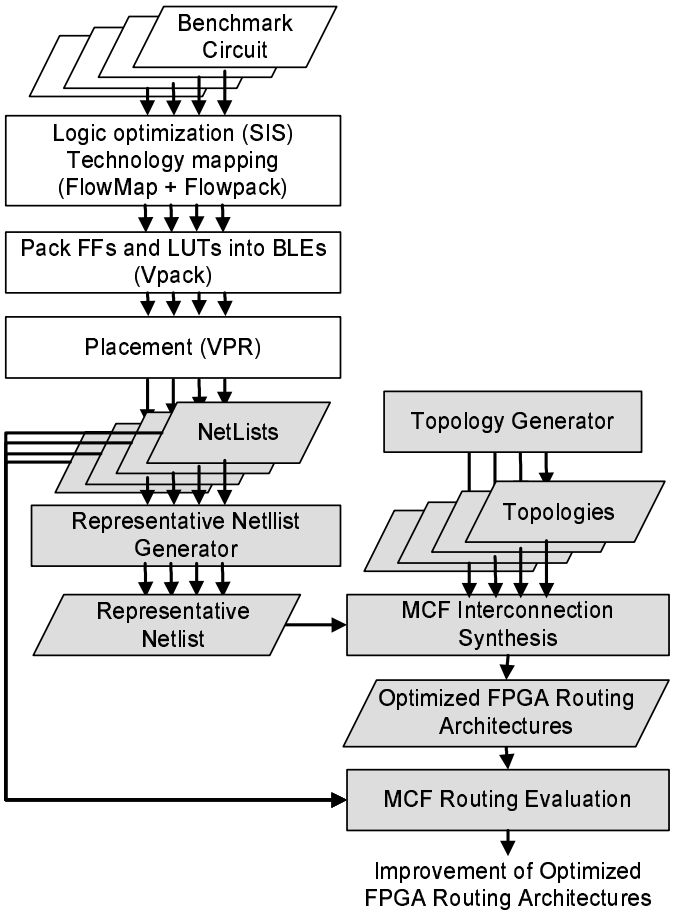


Figure III.2 An Improved CAD Flow for FPGA Routing Architecture Optimization

The representative netlists and the candidate FPGA topologies are then fed into various MCF formulations, which model the FPGA routing optimization problem with specific objectives, such as energy consumption or switch area density. By solving the MCF formulation, we can find out the best topology and the track capacity distribution and wire style assignment, and output these optimized FPGA global routing architectures.

At the last step, we implement the set of benchmark circuits on the optimized FPGA routing architectures to evaluate the actual improvement on energy consumption or switch area density. We compare these results with those from traditional FPGA global routing architecture to evaluate the impact of our optimization methodology.

Compared to traditional CAD flow in FPGA design I.6, our improved design flow is able to automatically generate candidate global routing architectures for evaluation, instead of manually generating the architectures by experienced designers. This largely increases the flexibility of global routing architectures and explores a much larger design space.

In our improved CAD flow, there are four major components. The core components are MCF optimizer for generating optimized routing architecture and MCF evaluator for evaluating the architecture. Besides those, we also have netlist generator for generating representative netlist and topology generator for generating candidate topologies. We describe these components in the following sections.

III.B.2 MCF Interconnection Synthesis and Evaluation

As shown in our improved CAD flow, the cores of our optimization framework are two MCF models. One is MCF interconnection synthesis model, which generates the optimized FPGA routing architectures with topology and wire style optimizations for representative netlist. The other model is MCF routing evaluation, which evaluates the actual performance of these optimized FPGA routing architectures for the targets set of benchmark circuits.

The major difference between these two MCF models lies in their constraints. For MCF interconnection synthesis, the capacity of each routing channel is unknown, hence it regards on-chip area resources of the routing channel as its constraints, and generates optimized capacities for each routing channel. The constraints for MCF routing evaluation are these output routing capacities.

The following subsections describe in details about these two models.

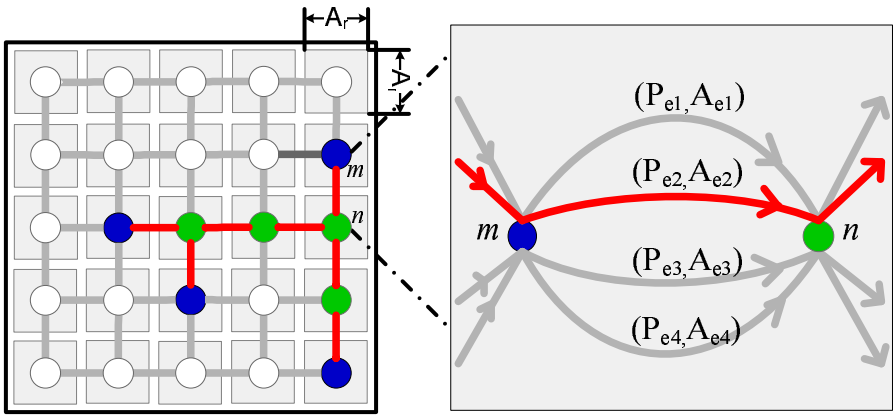


Figure III.3 An Improved CAD Flow for FPGA Routing Architecture Optimization

First, we show how to integrate the wire style optimization into the MCF models. Then, we present the formulations for each MCF model with various design objectives in mind. This demonstrates the flexibility of our methodology to be adapted to a wide range of FPGA routing architecture optimizations. At last, we briefly describe the algorithms to efficiently solve these MCF models.

Integration of Wire Style Optimization

Assume an FPGA chip with $n \times n$ logic blocks. These logic blocks communicate with each other through $n \times n$ switch boxes at the intersection of the channels. A topology is defined as a bi-directed graph $G = (V, E)$, where, each node $v_i \in V$ represents a switch box, and each edge $e_{i,j} \in E$ represents routing tracks between switch boxes i and j. These wire tracks can be implemented with multiple wire styles. Assume there are k nets. For each net i, its communication demand is $d_i = 1$. Let t_i be the set of paths on Steiner trees to connect net i, and let $\mathbf{T} := \cup_i t_i$. Variable $f(t)$ denotes the amount of flow along Steiner tree t, for every $t \in \mathbf{T}$.

Figure III.3 demonstrates an example on how to integrate multiple wire styles into MCF models. In a mesh architecture, we have a net of 4 pins (black nodes) to be routed. We connect these pins using a minimum Steiner tree (grey

nodes are Steiner nodes), as shown in dark lines in left side of the figure. Then we use multiple edges to represent available wire styles, as shown in right side of the figure. For link (m, n), there are 4 edges from node m to node n, which represents 4 types of candidate wire styles. A pair (P_e, A_e) is associated with each edge e. P_e is per bit energy on edge e. A_e is the wire pitch. If there is flow goes through edge 2 (as shown in the dark line), it means wire style 2 is selected for link (m, n), and the capacity of edge 2 equals to the amount of the flow. Therefore, if we solve the MCF formulations and get the flow distribution, we can obtain the optimized global routing architecture with wire style optimization.

MCF Interconnection Synthesis

Different FPGA optimization problems correspond to different MCF interconnection synthesis formulations. MCF model has the flexibility to adapt to various design objectives. In our work, we study three types of optimization problems, focusing on the energy optimization, the switch area optimization, and their co-optimization. These optimization problems are important and of the interests in modern FPGA routing architecture design.

For the first problem, we optimize the energy of FPGA routing architecture. To estimate energy, for each edge e, we assume that P_e represents bit energy on link e and the corresponding switch box.

$$P_e = P_w + P_{sb}$$

where P_w and P_{sb} are bit energy on interconnects and switch box, respectively. When a flow of amount f goes through the edge and the corresponding switch box, the total energy is $P = P_e \cdot f$

P_{sb} can be estimated by

$$P_{sb} = P_s \cdot N_s$$

where P_s is energy for a single switch in a switch box, and N_s is the total number of switches in a switch box. Assume F_s is number of switches connected to each

wire entering a switch box, and f is the amount of flow go through a switch box, we have:

$$N_s = 1/2 \cdot F_s \cdot f$$

The following is the formulations for MCF synthesis on energy optimization. The objective is to minimize total energy of routing architecture, which is the sum of per-bit energy on all routing tracks (as in Equation (III.1)). We have two constraints, routability constraint (III.2), which means all the nets in the representative netlist should be routable, and routing area constraint (III.3), which means that when we route the nets, the routing area usage cannot exceed the available on-chip area resources on the vertical or horizontal dimension A_r.

$$Min: \quad \sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e \in t} f(t) \cdot P_e \tag{III.1}$$

$$s.t. \quad \forall 1 \leq j \leq k : \sum_{t \in T_j} f(t) \geq 1 \tag{III.2}$$

$$\forall q : \sum_{e \in Grid(q)} A_e \cdot \sum_{t:e \in t} f(t) \leq A_r \tag{III.3}$$

$$\forall t : f(t) \geq 0 \tag{III.4}$$

The outputs of the MCF synthesis model are the optimized FPGA global routing architectures with expected design objectives, in this case, expected energy on routing architectures. Notice that the variables in the formulations are $f(t)$. After we solve the MCF formulations, we can obtain the optimized capacity for each edge by calculating the accumulated $f(t)$ on that edge. In this way, we have the optimized FPGA routing architecture for a certain topology. Then we can repeat this process for each candidate topology and generate the optimized FPGA routing architectures with topology and wire style optimizations.

In the second case, we optimize the total switch area of switch boxes. Since the switch area is proportional to the number of switches, we try to minimize the total number of switches in switch boxes as our design objective. The

constraints of this problem are exactly the same as those of the first case, i.e., the routability and routing area constraints. Therefore, we omit the constraints part of the formulations, and give only the objective function as follows, in which energy parameters P_e are simply replaced by switch quantity parameters N_s.

$$Min: \quad \sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e \in t} f(t) \cdot N_s \tag{III.5}$$

Furthermore, such MCF synthesis model can be easily applied to study the tradeoffs between multiple design factors. In our third case, we study switch area constrained energy optimization problem, which means we optimize energy of FPGA routing architecture, while at the same time all requirements on total switch area usage are satisfied. Compared with the first energy optimization problem, this problem has one more switch area constraint (III.6), where A_s is the given switch area budget. The objective function and the other constraints are exactly the same as formulation (III.1), (III.2), and (III.3).

$$\sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e \in t} f(t) \cdot N_s \leq A_s \tag{III.6}$$

MCF Routing Evaluation

As we explained earlier, MCF routing evaluation model differs from MCF interconnection synthesis model only on that one of its constraints is routing channel capacity instead of on-chip routing area resources. Take energy optimization problem as an example, its MCF routing evaluation formulations are as follows, where $c(e)$ represents the capacity of edge e.

$$Min: \quad \sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e} f(t) \cdot P_e \tag{III.7}$$

$$s.t. \quad \forall 1 \leq j \leq k : \sum_{t \in T_j} f(t) \geq 1 \tag{III.8}$$

$$\forall e : \sum_{t : e \in t} f(t) \leq c(e) \tag{III.9}$$

$$\forall t : f(t) \geq 0 \tag{III.10}$$

The outputs of MCF routing evaluation model are the actual design results, such as total energy, for each of benchmark circuits. This evaluation process verifies the effectiveness of MCF interconnection synthesis model.

III.B.3 Representative Netlist Generator

To achieve the best benefits of our FPGA routing optimizations, we need to have a good understanding of the nature of communications of our applications. The performance of both topology and wire style optimizations largely depends on the underlying communication pattern. For example, routing architectures with long distance transmission lines may be able to effectively reduce the energy consumption of the applications which have many global communications, but it may bring negative effects for those applications where most of communications are local.

Therefore, we generate a representative netlist from a set of FPGA benchmark circuits. The generated representative netlist should catch the characteristics of the benchmark circuits. The representative netlist generation is based on the statistical analysis of the candidate application circuits. Three sets of key parameters are to be determined. First, how many nets should the netlist have? Second, what is the size, e.g. the number of pins, of each net? At last, what are the pin locations of each net?

In our work, we set the size of the representative netlist to be the maximum netlist size among all the benchmark circuits. Because a netlist with larger size usually requires more routing capacities to route, it is intuitive to assign the representative netlist with the maximum routing capacity requirements among the benchmarks, so that the optimized FPGA routing architectures can be reconfigurable to accommodate all benchmark circuits.

Assume the number of pins of each application is N_a, the number of pins in the representative netlist is:

$$N = max_{a=1..H} N_a$$

Assume the number of nets of size k in application a is $M_a(k)$, we roughly choose the number of nets of size k in the representative netlist as:

$$M(k) = \frac{\sum_{a=1..H} M_a(k)}{\frac{\sum_{a=1..H} N_a}{N}}$$

However, $\sum_{a=1..H} M_a(k)$ usually cannot be divided evenly by $\frac{\sum_{a=1..H} N_a}{N}$ in practice. We use the scheme shown in Figure III.4 to approximate the applications' net size distribution more accurately. On the left size of the figure, the first column is net size k; the second column is the total number of nets with size k in the all the applications, i.e., $\sum_{a=1..H} M_a(k)$. We also assume $\sum_{a=1..H} N_a/N = 5$ in this example. To decide the number of nets, we go through the table in the order of net size from large to small. We accumulate the counts of number of nets until $\sum_{a=1..H} M_a(k) \geq 5$, then we output a number of nets of average size. For example, we combine the total appearance of net size 90, 78, 72, 72, and 65, and generate one representative net of size $(90 + 78 + 72 * 2 + 65)/5 = 75$; the residual of 2 is left over to the next round of computation. The nets with size of 1 are ignored in the representative netlist.

To determine the size of each net, we first count the net size of all the benchmark circuits, and calculate their distribution. Then we design the size of each net in the representative netlist to match this distribution pattern. For example, if 5% of nets in benchmark circuits have number of pins in the range from 30 to 35, and if the size of our representative netlist is 1000, then we should evenly distribute 50 nets with pins in that range.

Finally, we need to determine the pin locations of each net. Random generation of pin locations may lose the intrinsic communication patterns of the benchmark circuits. Therefore, we analyze the distribution of the frequency of each pin in the candidate circuits, and generate a corresponding "pin pool" with frequency distribution for each pin in the pool. Then for each net, we pick pins

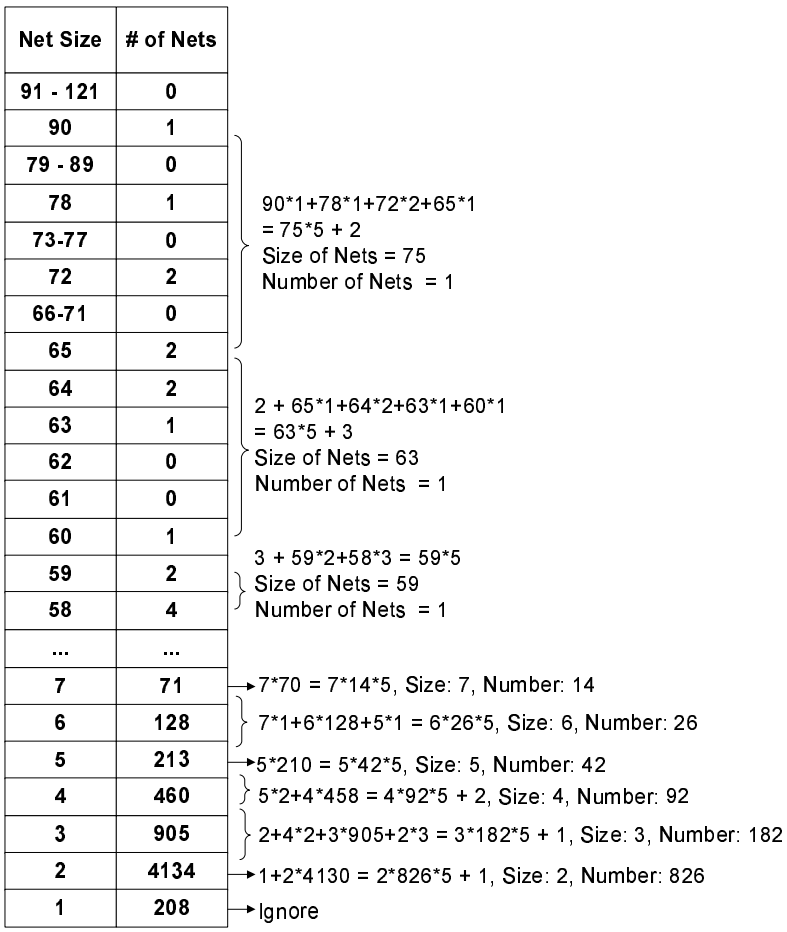


Figure III.4 Example of Net Size Estimation. Assume $\sum_{a=1..H} N_a/N = 5$

from "pin pool" according to their frequency function. In this way, we preserve part of the communication patterns of the benchmark circuits. we determine the distance among pins by a *geometry distribution function*. The function is defined as the probability of the distance between two pins decreases exponentially with increasing distance, i.e., $P(k) = p(1-p)^k, k = 1, 2, ...$ where k is the distance between two pins, p is the probability of links with distance 1, and $P(k)$ is the probability of links with distance k. Figure III.5 shows an example of the process of net generation.

Notice that, in our work, we simplify the problem by generating the representative netlist connecting switch boxes instead of logic blocks, by distributing the pins of nets on logic blocks to the adjacent clockwise switch boxes. By insert-

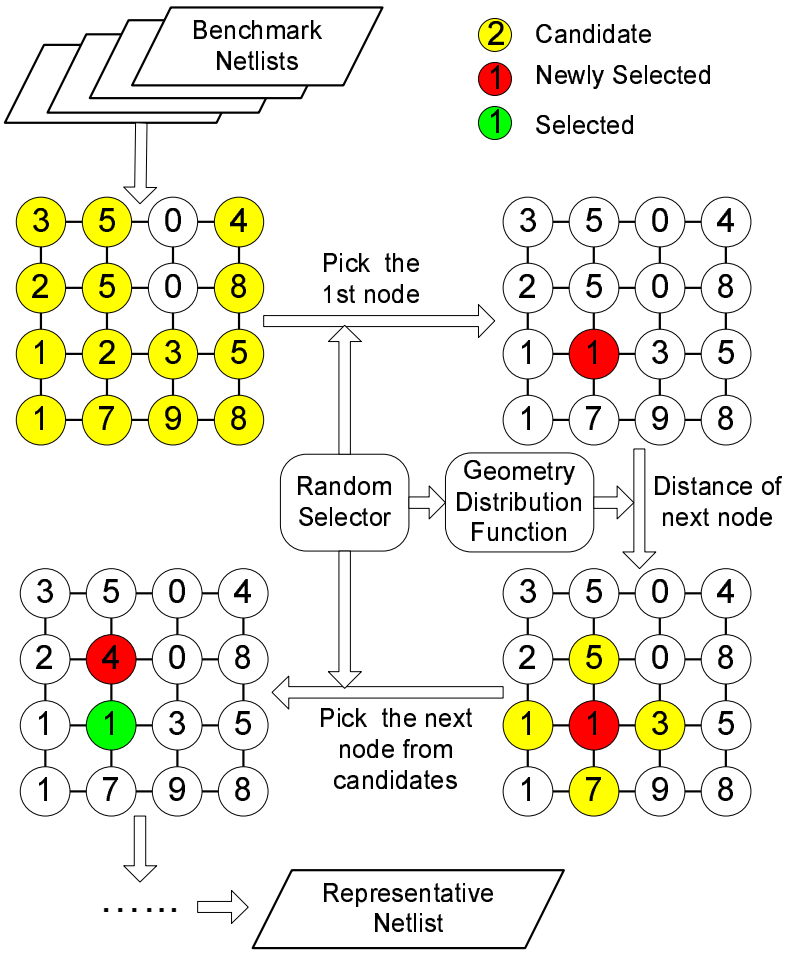


Figure III.5 Example of Netlist Pin Generation

ing logic boxes as extra nodes into our MCF models, we can easily improve the accuracy of our methodology, but with the price of higher computational cost.

III.B.4 Topology Generator

To perform topology optimization, we first generate a set of candidate topologies and put them in the topology library. The topologies of our optimized FPGA routing architectures are selected from the topology library. The library can be easily expanded by importing valuable candidate topologies.

Topology library generation is one of the key issues to the success of our FPGA routing architecture optimization. Even after clustering the look-up

tables (LUTs) into larger logic blocks, there are still a huge number of possible topologies. For example, for an FPGA with 10×10 logic blocks, each row or column has $2^{C(10,2)} = 2^{45}$ different connections, and the whole FPGA chip has $(2^{45})^{20} = 2^{900}$ different connections. It is impossible to explore them exhaustively with the current computation technology.

To reduce the size of topology library and only keep the most valuable and promising topologies, we make a few assumptions without loss of generality. First, we assume all wire segments have lengths of power of two, i.e. there are only wires in lengths of 1, 2, 4, 8, etc. Second, on each row or column, wire segments should repeat themselves consecutively along the whole routing channel. Third, all rows and columns should have identical connections. The reasoning of these assumptions is that: at the stage of FPGA global routing architecture design, the target applications are still unknown, therefore it is reasonable to design relatively regular and symmetric topologies to fit potential applications. Based on the above assumption, we exhaustively generate all qualified candidate topologies.

Based on the above assumption, we exhaustively generate all qualified candidate topologies. The following table III.1 summarize the total number of candidate topologies with various FPGA size.

Table III.1 FPGA Candidate Topology Generation

FPGA size	10x10	11x11	12x12	13x13	14x14	15x15
number of topologies	64	93	93	130	130	176

III.C Experimental Results

In our experiments, we use seven MCNC benchmark circuits [57] with moderate sizes. We first perform technology mapping to map these benchmark circuits to 4-LUTs, and pack every 16 4-LUTs to a larger logic blocks, then place these logic blocks on island-style FPGA chip. Table III.2 shows the size of resulting representative netlists of these seven benchmark circuits. Since the size of switch

box array ranges from 10×10 to 11×11, the representative netlist is of size 11×11. We set p to be 0.1 in the geometry distribution function $f(k) = p(1-p)^k$ in representative netlist generation, because we observe it best match the connection nature of our benchmark circuits.

Table III.2 Size of Representative Netlist of MCNC Benchmark circuits

	alu4	apex4	diffeq	dsip	ex5p	misex3	tseng
size	11x11	10x10	11x11	11x11	10x10	11x11	10x10
# of nets	621	798	945	593	745	771	788

We generate the candidate topologies using the topology generator described in section III.B.4. In our experiments, we assume the available segment lengths are 1, 2, 4, and 8. Segment of length 1 is mandatory, while other three types of segments are optional. Therefore, the candidate topologies include the base mesh topology, plus others that add arbitrary choices of three extra links. Abiding such assumptions, for FPGA of size 11×11, the total number of generated candidate topologies is 93.

We assume 4 types of candidate wires, RC repeated wires with $1\times$, $2\times$ and $4\times$ minimum global pitch and transmission line with $10\times$ minimum pitch. Figure III.6 shows the energy consumption of each wire style under various wire lengths. For RC repeated wires, the energy consumption is proportional to the wire length; for on-chip transmission line, after initial setup energy, its energy keeps almost constant as wire length increase.

In our MCF approximation algorithms, we set error tolerance ϵ to 1%. All of the following experiments are based on $0.18um$ design technology. Since each grid has the same vertical and horizontal dimension, for convenience, we use only the vertical dimension to represent the area budget, therefore the unit of area in our experiments is um.

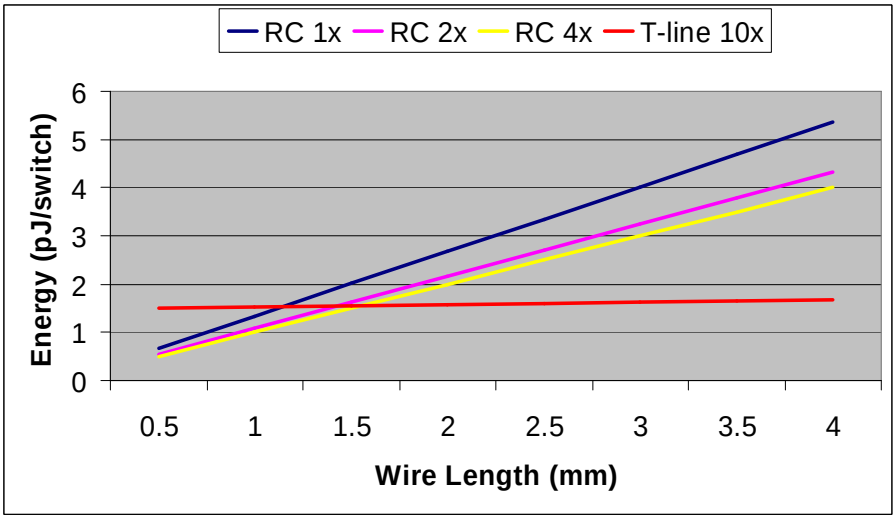


Figure III.6 Energy for Four Types of Wire Styles

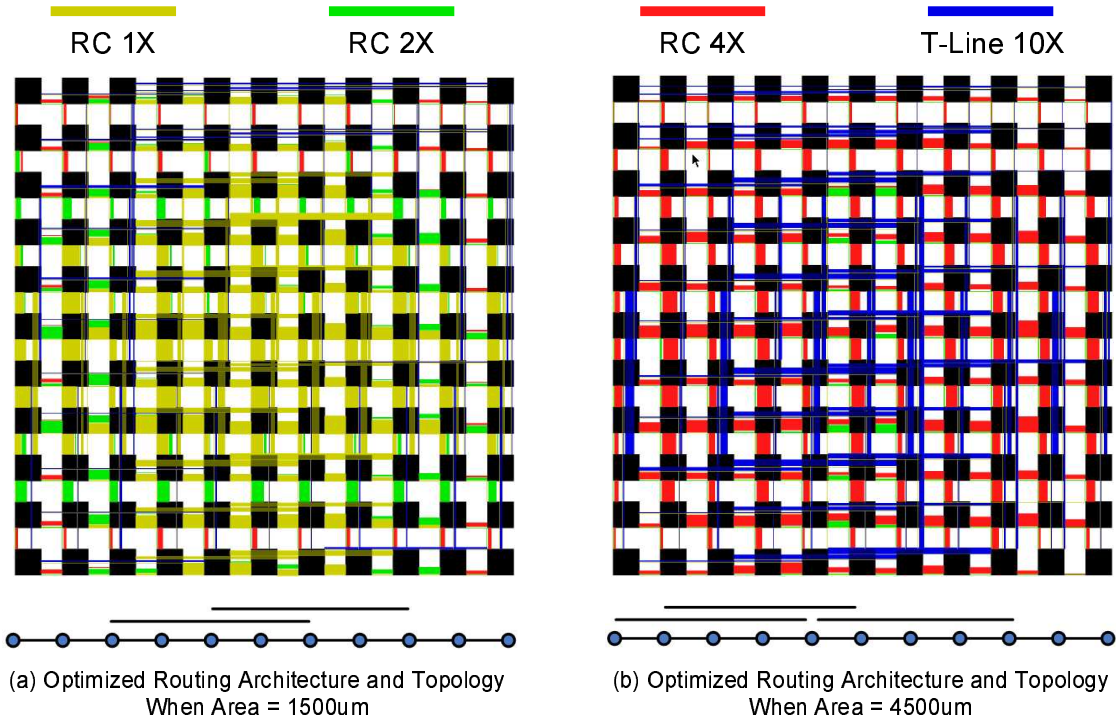


Figure III.7 Optimized FPGA Routing Architectures and Corresponding Topologies (a)When routing area constraint is tight (b) When routing area constraint is loose

III.C.1 FPGA Energy Consumption Optimization

Optimized FPGA Routing Architecture

Figure III.7 shows our optimized energy-efficient FPGA routing architectures, including the corresponding topologies and wire style assignment, under tight and loose routing area constraints. We assume 4 types of wires, RC wires with 1×, 2× and 4× minimum global pitch and transmission line with 10× minimum pitch. These wire styles consume decreasing energy but occupy increasing on-chip routing area. Figure III.7 (a) shows that when the routing area is very tight, e.g. $1500um$, the thinnest 1× RC wires are used for most of the connections. We also notice that at the edge of the chip, we still have some energy-efficient transmission lines used to reduce energy consumption, since at these region the communication flow does not saturate the available routing capacity so that there still is room for wire style optimization. The bottom of the figure shows the topology of the optimized FPGA architecture. As a result of our energy model, we estimate the energy of the optimized architecture is around 6.46×10^3pJ. Figure III.7 (b) shows the optimized architecture when the routing area is loosen to $4500um$. Since now we have abundant routing area for wire style optimization, transmission lines are adopted for all the long links, and RC wire with 4× minimum pitch are adopted for those short links. The resulting energy is 4.63×10^3pJ, which is 28% improvement over that in case (a).

From Figure III.7, we see a clear trend to use wider wires as routing area budget increases. At the same time, narrow wires are used in the center of the chip. Notice that, there is no length 2 wires in our optimized routing architecture. This is because according to our energy model, the transmission line only gain benefits when wire lengths are longer than 3. Therefore, the use of transmission line has profound impact in FPGA routing architecture optimization.

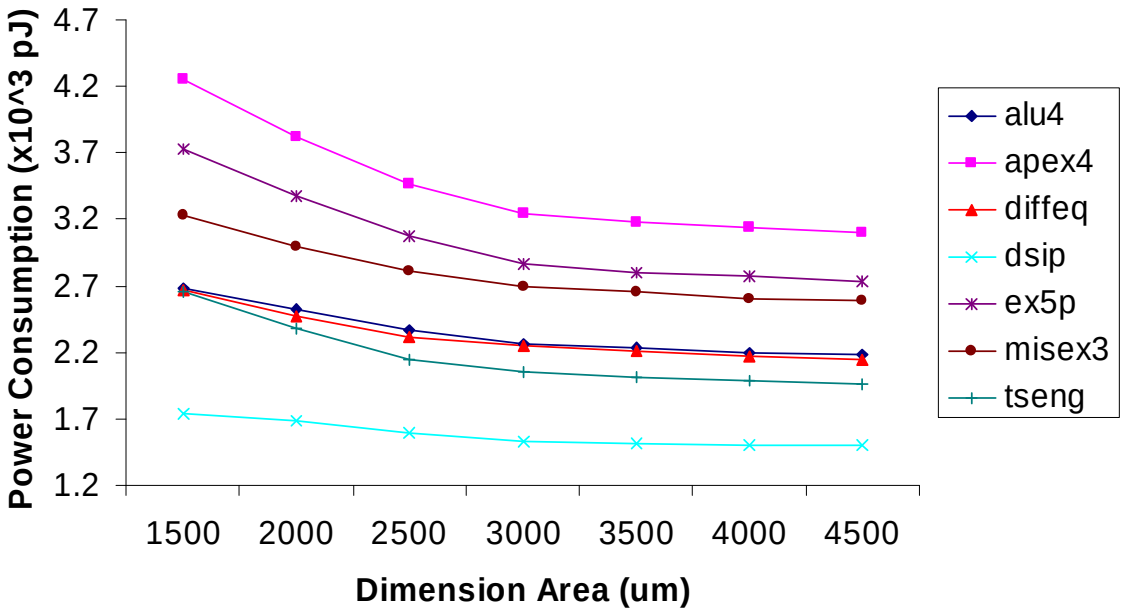


Figure III.8 Energy Consumption of Benchmark Circuits under Various Dimension Area Constraints

Impact of Dimension Area Constraint

Figure III.8 shows the energy estimation of seven of the benchmark circuits on our optimized FPGA routing architecture under various routing area constraints. The x-axis is area constraint from $1500um$ to $4500um$, which represents the area constraint from tightest to loosest. The y-axis is energy consumption in unit $\times 10^3 pJ$. As area constraints become looser, energy consumption of all benchmark circuits improves, where apex4 gains the largest improvement of 27.1% (from 4.26 to 3.11 $\times 10^3 pJ$); dsip gains the smallest improvement of 13.8% (from 2.67 to 2.15 $\times 10^3 pJ$). These results indicate that wire style optimization plays an important role when we have extra routing resources, since as dimension area budgets increase, more advanced but area consuming wires can be used to optimize the overall FPGA energy consumption.

Improvement Over Traditional Mesh Architecture

We compare the above energy consumption of benchmark circuits on our optimized FPGA routing architecture with that of traditional mesh routing architecture. Following the CAD flow described in III.B.1, we first generate a representative netlist with $p = 0.1$, then we solve the MCF formulation for energy optimization III.B.2 to find out the best topology and wire style assignment, and the corresponding energy consumption. We call this *expected* energy consumption, which means that instead of real energy savings by benchmark circuits, it is *expected* energy savings that can be achieved by our representative netlist. We then global route the benchmark circuits on the generated routing architecture, and get the real energy savings for various benchmarks.

Figure III.9 shows the energy improvement in percentage. In x-axis, each group of bars present the energy improvement under various area constraints for a certain benchmark circuit. Circuit *disp* has the smallest improvement, ranging from -3% to 6%; circuit *tseng* has the largest improvement from 5% to 24%. On average, our optimized routing architecture can achieve energy savings of 2% to 15% over mesh architecture, compared to 3% to 23% of expected savings. When area budget is small, such as $1500um$, our optimized routing architecture has no obvious advantages over traditional mesh architecture, which is because that we do not have enough routing area to adopt better wiring technology such as transmission line for energy optimization. When area budget increase from $1500um$ to $2500um$, large improvement can be seen. Further increasing of area budget does not bring too much benefits, though. Therefore, our methodology is able to find out how much on-chip area we should try to add to achieve the best benefit/cost.

Comparison With Optimal Architecture

We also compare the energy consumption of our optimized design with those of the optimal design for each benchmark circuit. By "optimal" energy consumption, we mean that when we design the FPGA routing architecture specifically

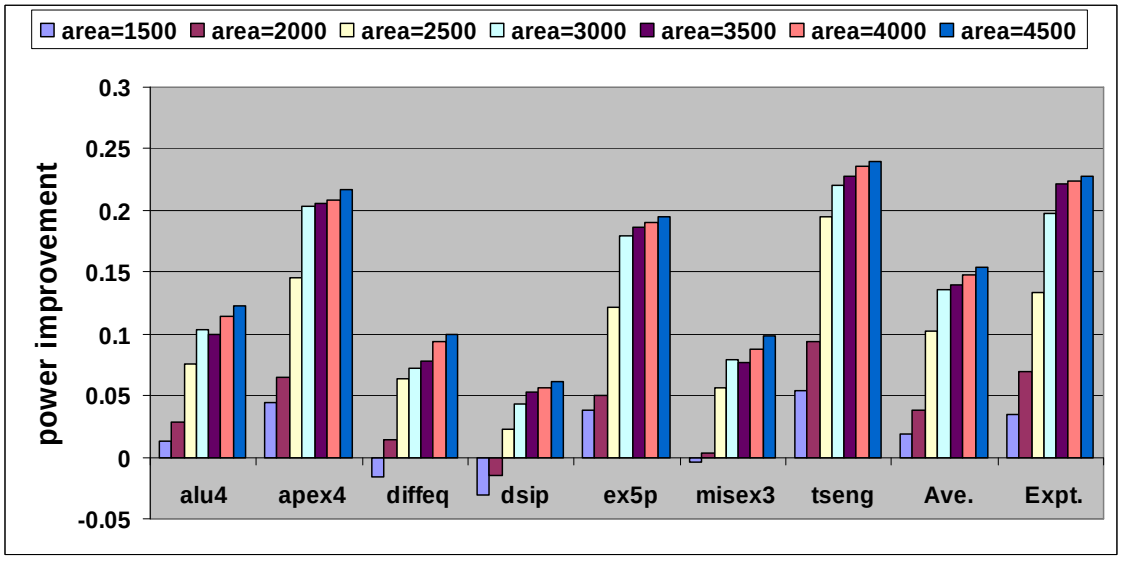


Figure III.9 Energy Improvement of Optimized Architecture vs. Traditional Mesh Architecture when $p = 0.1$

for a certain circuit, which is the lower bound of any optimized schemes. Figure III.10 shows the results. Each group of bars present the energy advantages of optimal design over our optimized design under various area constraints. When area constraints are tight, all benchmark circuits show that energy consumption of our optimized design is much worse than the optimal design, since there is not enough space left for wire style optimization. Once we have abundant area for optimization, our design is only 1% to 3% worse than optimal design. This indicates that since a general FPGA design needs to accommodate various application circuits, it has to assign major of its area to increase channel capacities, instead of optimizing performance. Therefore, if we have more area resources (e.g., extra routing layers), we can expect a significant improvement through intensive wire optimization.

Impact of Representative Netlist Generation

Since our methodology depends on representative netlist to generate the optimized FPGA global routing architecutre, we study the its effect on our optimization results. In representative netlist generation, the main parameter is p.

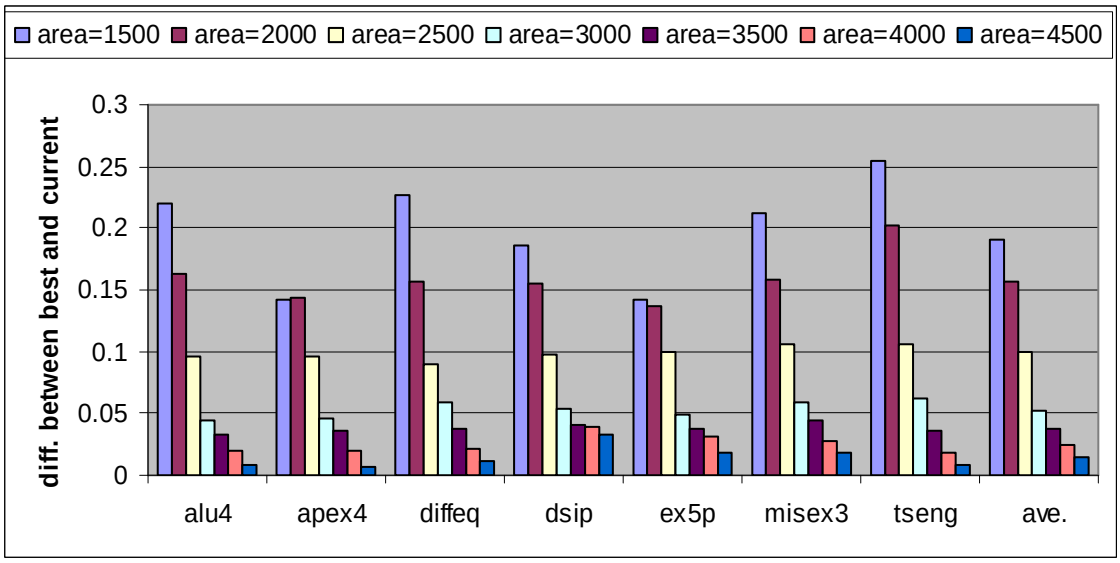


Figure III.10 Energy Difference of Optimized Architecture vs. Optimal Architecture when $p = 0.1$

Therefore, we compare the final energy consumption improvement of our optimized FPGA routing architecture over traditional mesh architecture, when $p = 0.1$, 0.2 and 0.3 in geometry distribution function. Figure III.11 and III.12 shows the results of $p = 0.2$ and $p = 0.3$, respectively. Compared with Figure III.9, although the energy consumption improvement pattern among benchmark circuits differs among different p, all of them have comparable improvement on overall average energy improvement, with maximum benefits ranging from 12% to 15% when comparing to traditional mesh architecture. These experimental results indicate that our methodology is not heavily depended on the characteristics of generated representative netlist.

III.C.2 FPGA Switch Area Density Optimization

As mentioned in III.B.2, our methodology is easy to apply to various design objectives. Besides energy consumption, we also optimize FPGA routing architecture to reduce its switch area density. We use number of total switches in switch box as objective, and have optimization formulations as III.5. Since total

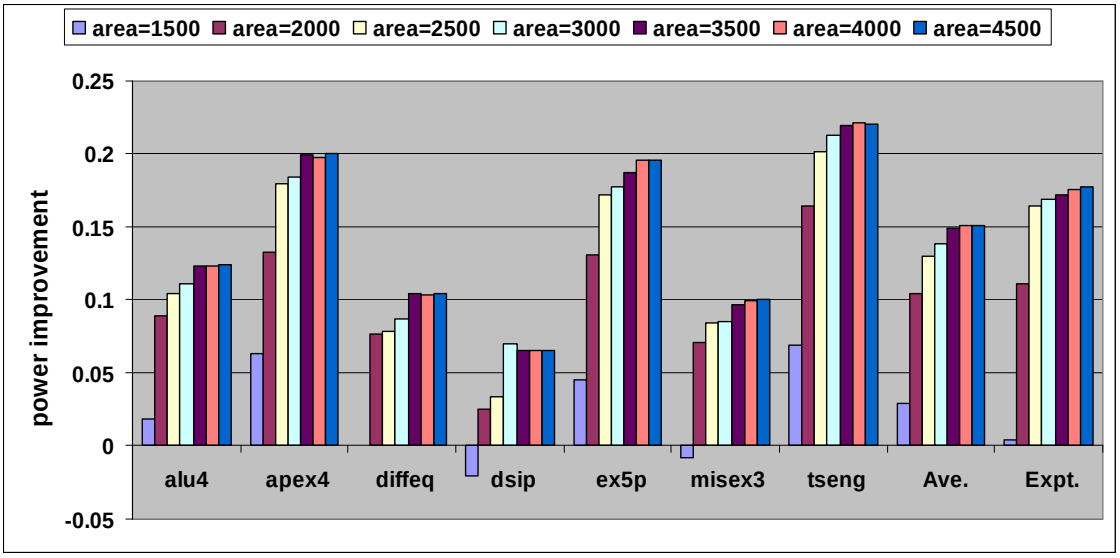


Figure III.11 Energy Improvement of Optimized Architecture vs. Traditional Mesh Architecture when $p = 0.2$

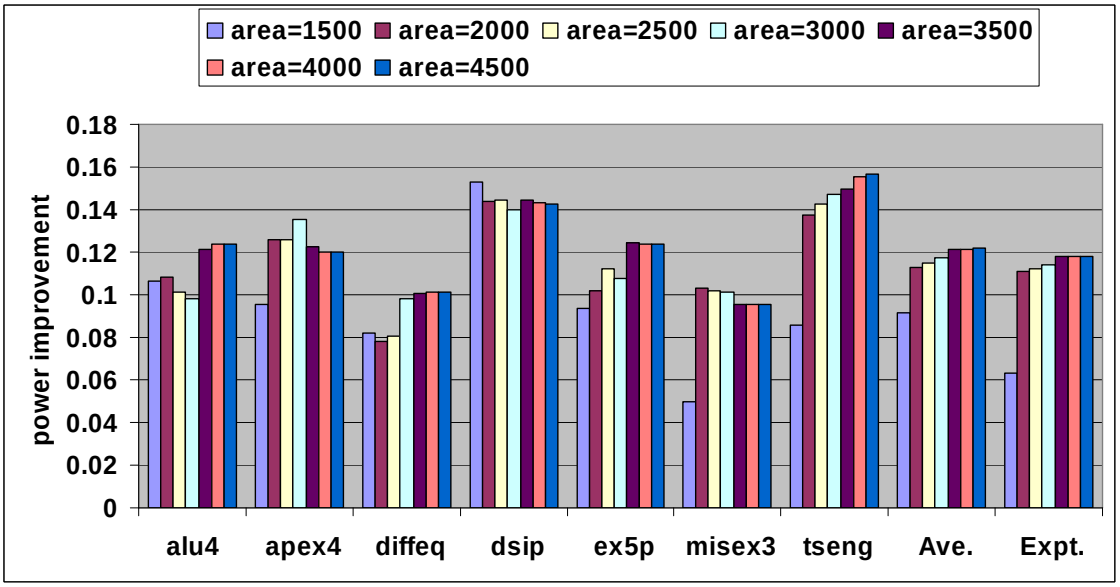


Figure III.12 Energy Improvement of Optimized Architecture vs. Traditional Mesh Architecture when $p = 0.3$

number of switches is not affected by wire styles, dimension area constraint is not a major issue in area density optimization. Therefore, we only compare the results under various p. Figure III.13 shows the switch area improvement when compared to mesh architecture when $p = 0.1$, 0.2 and 0.3. On average, 15% to 20% density improvement can be seen. Again, value of p does not effect the optimization very much.

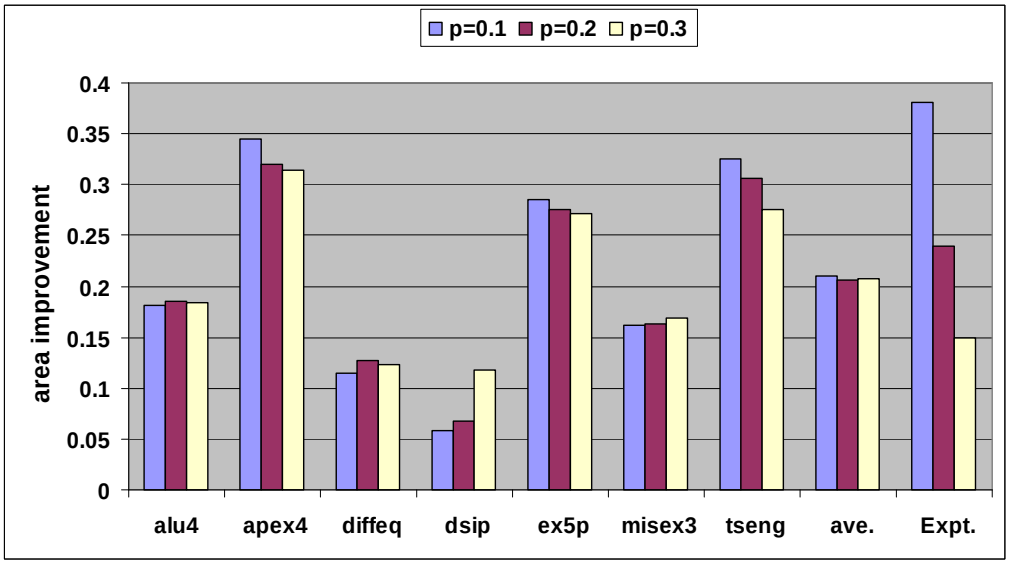


Figure III.13 Improvement of Number of Switches of Optimized Architecture vs. Mesh Architecture

III.C.3 FPGA Switch Density Constrained Low Energy Optimization

We have studied the FPGA routing architecture optimization on energy consumption and switch area density, respectively. In this section, we study the case that combines these two factors in unified optimization framework. We show energy savings when a small amount of switch area density is sacrificed. First, we use MCF model to search the topologies with the minimum number of switches (without energy optimization constraint), then loosing this switch area density constraint by up to 10% and optimize NoC energy consumption.

Figure III.14 depicts the optimized energy estimation under various

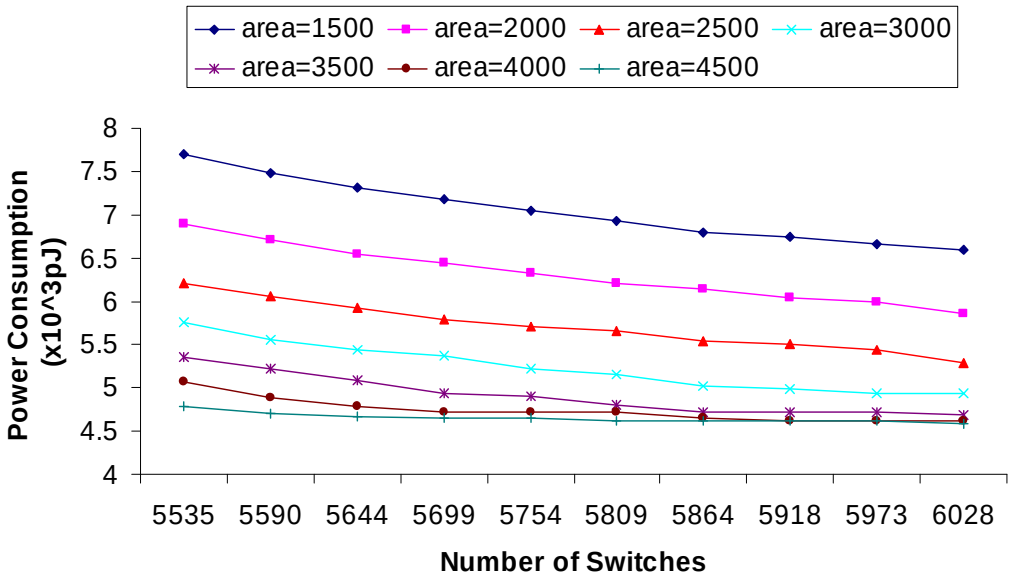


Figure III.14 Switch Density Constrained Low Energy Optimization

switch area constraints. The x-axis is the switch number constraints, from the minimum required amount of switches to 10% more than the minimum. The y-axis is energy consumption in unit $\times 10^3 pJ$. Each curve shows the changes of energy with a certain area budget. As the area budget increases, energy consumption of all benchmark improves because more energy-efficient wires can be used. As the budget of switch number increases, energy consumption also improves because the communication flow can be routed to more energy-efficient paths which may have high switch costs, for example, consisting of high-degree nodes. An interesting observation is that when the area budget is less, changing the switch number budget has larger impact on the optimal energy consumption. For example, when changing number of switches from 5535 to 6028, the energy consumption changes by 16.7% for area=$1500um$, which it is 4.6% for area=$4500um$. This is because that with tighter area budget, we have to use narrow but energy-costly wires, leaving a larger space for wiring optimization. When the area budget is abundant, the energy is already quite optimized no matter the switch density constraints.

III.D Summary

In this paper, we present an improved MCF model based CAD flow to perform aggressive optimizations, such as topology and wire style optimizations, to reduce the energy and switch area of FPGA global routing architectures. The experiments show that when compared to traditional mesh architecture, our optimized architectures achieve up to 10% to 15% power savings and up to 20% switch area savings in average for a set of seven benchmark circuits. As future work, we can apply the methodology to other design objectives, such as interconnect delay in FPGA global routing architectures.

IV

Conclusions and Future Work

With the develop of modern semiconductor technology, on-chip interconnection architecture will become the bottleneck of system performance, power consumption and routing resources, and will play a more and more important role in chip design. This places increasing demands on methodologies to effectively optimize on-chip interconnection architecture.

IV.A Dissertation Contributions

In this dissertation, we propose a methodology to optimize the power consumption or communication latency of two promising on-chip interconnection architectures, NoC and FPGA global routing architecture. Our methodology adopts two optimization schemes, topology optimization and wire style optimization, and uses MCF models to perform and evaluate these optimizations. We implement and optimize the MCF solver using polynomial approximation algorithms, which are significantly faster than the commercial linear programming solver CPLEX. The primary contributions of the dissertation are:

- We introduce a variety of wire styles into on-chip interconnection architecture optimization, and propose a methodology that adopts unified MCF models to optimize power, latency and area of on-chip interconnection architecture through topology optimization and wire style optimization.

- We develop approximation algorithm and apply the interval estimation technique to speedup MCF solver. Experiments show that our solver is faster than the commercial linear programming solver CPLEX by order of hundreds.

- For NoC optimization, we find the best NoC implementations for any given average node to node communication latency requirement. The experiments show that for NoC, an optimized design can improve the power-latency product by up to 52.1%, 29.4% and 35.6%, if compared with mesh, torus and hypercube topologies, respectively. Furthermore, by sacrificing 2% of latency constraints, power consumption of that optimized design can be improved by up to 19.4%. Carefully balancing between NoC power efficiency and communication latency is important in NoC design.

- For FPGA routing architecture optimization, we propose an improved CAD flow to perform aggressive optimizations, such as segmentation distribution, flexible track assignment and wire style optimization. Our methodology allows flexible number of routing tracks in each routing channel. The experiments show that given available routing resources, when compared with mesh architecture, the optimized architectures achieve average up to 10% to 15% power savings and up to 20% switch area savings for a set of seven benchmark circuits. The methodology can easily be applied to optimize FPGA switch area density, and to power/switch area co-optimization.

IV.B Future Directions

IV.B.1 Timing Analysis in On-Chip Interconnection Architecture Optimization

One of the future work is to improve the timing analysis capability of our optimization methodology. In our current NoC optimization, we assume the global average latency as the timing constraint, which is the sum of latency of all

the communication pairs. This assumption is invalid for many applications that have tight timing requirements on critical paths. For such applications, the design of on-chip interconnection architecture need to guarantee that all critical paths are routed within timing constraints, and at the same time, the power consumption is optimized. The problem can be formulated as following if we use the same notations as in II.B.1, and assume LT_i is latency constraint on critical commodity i.

$$Min: \quad \sum_{j=1}^{k} \sum_{p \in p_j} \sum_{e \in p} f(p) \cdot P_e \tag{IV.1}$$

$$s.t. \quad \forall j \in C : \sum_{p \in p_j} \sum_{e \in p} f(p) \cdot D_e \leq LT_j \tag{IV.2}$$

$$\forall 1 \leq j \leq k : \sum_{p \in p_j} f(p) \geq d_j \tag{IV.3}$$

$$\forall q : \sum_{e \in Grid(q)} A_e \cdot \sum_{p : e \in p} f(p) \leq A \tag{IV.4}$$

$$\forall p : f(p) \geq 0 \tag{IV.5}$$

The formulation is very similar to that of global average latency constraint in II.B.1, except for IV.2, which indicate the pairwise critical path timing constraint. How to efficiently solve the above formulation using approximation algorithm is for the future research work.

For FPGA routing architecture optimization, currently we do not have timing constraint included. In future research, we will add timing analysis to our optimization framework, hence integrate three key design metrics in FPGA routing architecture, power consumption, timing and area density, all in a unified optimization framework. This will be an ambitious research attempt for FGPA routing architecture optimization.

IV.B.2 Large Scale On-Chip Interconnection Architecture Optimization

We also hope to improve our methodology to solve the larger scale optimization problems of on-chip interconnection architectures. Currently, we can optimize up to 8×8 NoC, and 12×12 FPGA routing architecture. When the problem scale is beyond this, solving MCF formulations will take too long time to explore a large design space. But as the technology shrinking down, the on-chip communication architecture size will increase fast. Therefore, it is critical to effectively improve our methodology to optimize large scale on-chip interconnection architecture.

One approach is to improve the algorithms inside MCF solver. Heuristic algorithms can be introduced to speed up the flow routing procedure in MCF solver. Another approach is to apply our methodology to hierarchical on-chip interconnection architecture design, which is a natural and effective solution when problem size becomes very large. Finally, we can use parallel techniques to speed up the computation. Our algorithm searches through a large number of interconnection topologies, so it is easy to be processed in parallel.

Looking forward, we believe that on-chip interconnection architecture optimization will play key role in future VLSI design. Our research is one of the valuable studies to explore challenges and opportunities of the new paradigm the on-chip communication.

Bibliography

[1] C. Albrecht, "Provably Good Global Routing by A New Approximation Algorithm for Multicommodity Flow," *International Symposium on Physical Design,* pp.19-25, 2000.

[2] C. Albrecht, "Global Routing by New Approximation Algorithms for Multicommodity Flow," *IEEE Transactions on Computer-Aided Design of Integreted Circuits and Systems,* vol. 20, no. 5, May, 2001.

[3] C. Albrecht, A.B. Kahng, I.I. Mandoiu, and A.Z. Zelikovsky, "Multicommodity Flow Algorithms for Buffered Global Routing," *Approximation Algorithms and Metaheuristics,* Chapter 80, 2007.

[4] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. De Micheli, "NoC Synthesis Flow for Customized Domain Specific Multi-Processor System-on-Chip," *IEEE Trans. on Parallel and Distributed System,* Feb. 2005.

[5] V. Betz, and J. Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," *International Workshop on Field-Programmable Logic and Applications,* pp.213-222, 1997.

[6] V. Betz, and J. Rose, "Directional Bias and Non-Uniformity in FPGA Global Routing Architectures," *ICCAD* pp.652-659, 1996.

[7] V. Betz, and J. Rose, "Effect of the Prefabricated Routing Track Distribution on FPGA Area-Efficiency," *IEEE Transactions on VLSI* pp.445-456, September, 1998.

[8] V. Betz, J. Rose, and A. Marquardt, "Architecture and CAD for Deep-Submicron FPGAs," *Kluwer Academic Publishers,* February, 1999.

[9] S. Brown, R. Francis, J. Rose and Z. Vranesic, "Field-Programmable Gate Arrays," *Kluwer Academic Publishers,* 1992.

[10] S. Brown, M. Khellah, and G. Lemieux, "Segmented Routing for Speed-Performance and Routability in Field-Programmable Gate Arrays," *Journal of VLSI Design,* vol. 4, no. 4, pp.275-291, 1996.

[11] S. Brown, M. Khellah, and Z. Vranesic, "Minimizing FPGA Interconnect Delays," *IEEE Design and Test Magazine,* pp.16-23, 1996.

[12] R.C. Carden IV, and C.K. Cheng, "A Global Router Using An Efficient Approximation Multicommodity Multiterminal Flow Approximation," *Design Automation Conference,* pp.316-321, June, 1991.

[13] R. Carden, J. Li, and C.K. Cheng, "A Global Routing with a Theoretical Bound on the Optimum Solution," *IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems,* vol. 15, pp.208-216, 1996.

[14] R. Chang, N. Talwalkar, C.P. Yue, and S. S. Wong, "Near Speed-of-light Signaling Over On-Chip Electrical Interconnects," *IEEE J. of Solid-State Circuits,* vol. 38, no. 5, pp. 834-838, May 2003.

[15] H. Chen, R. Shi, C.K. Cheng, and D. Harris, "Surfliner: A Distortionless Electrical Signaling Scheme for Speed of Light On-Chip Communications," *IEEE Intl. Conf. on Computer Design,* pp.497-502, Oct. 2005.

[16] P. Chow, S. Seo, J. Rose, K. Chung, G. Paez and I. Rahardja, "The Design of an SRAM-Based Field-Programmable Gate Array, Part I: Architecture," *IEEE Transactions on VLSI*

[17] J. Cong, and Y. Ding, "FlowMap: An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table Based FPGA Designs," *IEEE Transactions on CAD,* pp.1-12, January, 1994.

[18] M. DallOsso, G. Biccari, L. Giovannini, D. Bertozzi, and L. Benini, "xPipes: A Latency Insensitive Parameterized Network-on-Chip Architecture for Multiprocessor SoCs," *International Conference on Computer Design,* pp. 80-85, October, 2003.

[19] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," *ACM/IEEE Design Automation Conf.*, pp. 684-689, June 2001.

[20] W.J. Dally, "Performance Analysis of a k-ary n-cube Interconnect Networks," *IEEE Transactions on Computers,* pp. 775-785, June, 1990.

[21] A. DeHon and R. Rubin, "Design of FPGA Interconnect for Multileverl Metallization," *IEEE Trans. on VLSI*, Vol. 12, No. 10, pp. 1038-1050, Oct., 2004.

[22] N. Garg, and J. Konemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," *the 39th Annual IEEE Symp. on Foundations of Computer Science,* pp. 300-309, Nov. 1998.

[23] P. Guerrier, and A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnections," *Design Automation and Test in Europe,* pp.250-256, 2000.

[24] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Oberg, M. Millberg, and D. Lindqvist,"Network on a Chip: An Architecture for Billion Transistor Era," *IEEE NorChip Conference,* pp. 166-173, November, 2000.

[25] R. Ho, K. W. Mai, and M. Horowitz, "The Future of Wires," *Proc. of IEEE,* vol. 89, no. 4, pp. 490-504, Aprail, 2001.

[26] S. Horiguchi, and Y. Miura, "Performance of Deadlock-Free Adaptive Routing for Hierarchical Interconnection Network TESH," *IEEE Symposim on Defect and Fault Tolerance in VLSI Systems,* pp.275-283, November, 2002.

[27] J. Hu and R. Marculescu, "Energy-Aware Mapping for Tile-based NoC Architectures Under Performance Constraints," *Asia and South Pacific Design Automation Conf.*, pp. 233-239, Jan. 2003.

[28] Y. Hu, H. Chen, Y. Zhu, A. A. Chien, and CK. Cheng, "Physical Synthesis of Energy-Efficient Networks-on-Chip Through Topology Exploration and Wire Style Optimization," *Intl. Conf. on Computer Design,* pp. 111-118, Oct. 2005.

[29] Y. Hu, Y. Zhu, H. Chen, R.L. Graham, and C.K. Cheng, "Communication latency aware low power NoC synthesis," *Design Automation Conference*, pp.574-579, June, 2006.

[30] A. Jalabert, S. Murali, L. Benini, and G.De Micheli, "xPipesCompiler: A Tool for Instantiating Application-Specific NoCs," *Design Automation and Test in Europe,* February, 2004.

[31] G. Karakostas, "Faster approximation schemes for fractional multicommodity flow problems", *the 13th Annual ACM/SIAM Symp. on Discrete Algorithms*, pp. 166-173, 2002.

[32] F. Karim, A. Nguyen, S. Dey, and, R. Rao, "On-Chip Communication Architecture for OC-768 Network Processors," *ACM/IEEE Design Automation Conf.*, pp.678-683. June, 2001.

[33] S.W. Keckler, D. Burger, C.R. Moore, R. Nagarajan, K. Sankaralingam, V. Agarwal, M.S. Hrishikesh, N. Ranganathan, and P. Shivakumar,"A Wire-Delay Scalable Microprocessor Architecture for High Performance Systems," *Intl. Solid-State Circuits Conf.*, pp. 1068-1069, February, 2003.

[34] M. Khellah, S. Brown, and Z. Vranesic, "Minimizing Interconnection Delays in Array-Bassed FPGAs," *CICC* pp.181-184, 1994.

[35] S. Kumar, A. Jantsch, M. Millberg, J. Oberg, J. P. Soininen, M. Forsell, K. Tiensyrja and A. Hemani, "A Network on Chip Architecture and Design Methodology", *IEEE Symp. on VLSI*, pp. 117-124, April 2002.

[36] S. Lee, H. Xiang, D.F. Wong, and R.Y.Sun, "Wire Type Assignment for FPGA Routing," *International Symposium on Field Programmable Gate Arrays,* pp.61-67, 2003.

[37] J. Liang, S. Swaminathan, and R. Tessier, "aSOC: A Scalable, Single-Chip Communications Architecture," *IEEE International Conference on Parallel Architectures and Compilation Techniques,* pp.37-46, October, 2000.

[38] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-power dissipation in a microprocessor," *International Workshop on System Level Interconnect Prediction,* pp.7-13, 2004.

[39] B. D. McKay, "Isomorph-free Exhaustive Generation," *J. Algorithms,* 26, pp.306-324, 1998.

[40] L. McMurchie, and C. Ebeling, "PathFinder: A Negotiation-Based Performance-Driven Router for FPGAs," *International Symposium on Field Programmable Gate Arrays,* pp.111-117, 1995.

[41] J. Meindl, comments at SRC/Marco Workshop, held at the Center for Integrated Systems, Stanford University, May 1999.

[42] M. Millberg, E. Nilsson, R. Thid, and A. Jantsch, "Guaranteed Bandwidth Using Looped Containers in Temporally Disjoint Networks Within the Nostrum Network on Chip," *Design Automation and Test in Europe,* pp. 890-895, February, 2004.

[43] S. Murali and G. De Micheli, "Bandwidth Constrained Mapping of Cores onto NoC Architectures", *Asia and South Pacific Design Automation Conf.*, pp. 896-901, Vol.2, Feb. 2004.

[44] S. Murali and G. De Micheli, "SUNMAP: A Tool for Automatic Topology Selection and Generation for NoCs", *ACM/IEEE Design Automation Conf.*, pp. 914-919, June, 2004.

[45] U.Y. Ogras and R. Marculescu, "Application-Specific Network-on-Chip Architecture Customization Vis Long-Range Link Insertion", *Intl. Conf. on Computer Aided Design*, pp. 246-253, Nov., 2005.

[46] L.S. Peh, "Flow Control and Micro-Architectural Mechanisms for Extending the Performance of Interconnection Networks", *Ph.D. Thesis, Stanford University*, August, 2001.

[47] K. Poon, S. Wilton and A. Yan, "A Detailed Power Model for Field Programmable Gate Arrays," *ACM Trans. on Design Automation of Electronic Systems*, Vol. 10, Issue 2, pp. 279-302, April, 2002.

[48] E. Rijpkema, K.G. Gossens, A. Radulescu, J. Dielissen, J. Meerbergen, P. Wielage, and E. Waterlander, "Trade Offs in the Design of A Router with Both Guaranteed and Best-Effort Services for Networks on Chip," *Design Automation and Test in Europe,* March, 2003.

[49] C. Seitz, "Let's Route Packets Instead of Wires", *Advanced Research in VLSI: Proceedings of the Sixth MIT Conference,* pp. 133-138, 1990.

[50] E.M. Sentovich, et al, "SIS: A System for Sequential Circuit Analysis," *Technical Report No. UCB/ERL M92/41, University of California, Berkeley,* 1992.

[51] F. Shahrokhi, and D.W. Matula, "The Maximum Concurrent Flow Problem," *Journal Associate Computer Mathematics,* vol.37, no.2, pp.318-334, 1990.

[52] L. Shang, L. Peh, and N.K. Jha, "PowerHerd: Dynamic Satisfaction of Peak Power Constraints in Interconnection Networks," *International Conference on Supercomputing,* pp. 98-108, 2003.

[53] L. Shang, L. Peh, A. Kumar, and N.K.Jha, "Thermal Modeling, Characterization and Management of On-Chip Networks," *International Symposium on Microarchitecture,* pp. 67-78, 2004.

[54] M. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, et. *al*, "The Raw Microprocessor: A Computational Fabric for Software Circuits and General Purpose Programs," *IEEE Micro*, Mar./Apr.2002.

[55] H. Wang, L. S. Peh, and S. Malik, "Orion: A Power-Performance Simulator for Interconnection Network," *Int. Symp. on Microarchitecture,* pp. 294-305, Nov. 2002.

[56] H. Wang, L. S. Peh, and S. Malik, "A Technology-Aware and Energy-Oriented Topology Exploration for On-Chip Networks," *Design, Automation and Test in Europe,* pp.1238-1243, Vol.2, Mar. 2005.

[57] S. Yang, "Logic Synthesis and Optimization Benchmarks, Version 3.0," *Technical Report, Microelectronics Center of North Carolina,* 1991.

[58] T. T. Ye, L. Benini, and G. De Micheli, "Analysis of Power Consumption on Switch Fabrics in Network Routers," *ACM/IEEE Design Automation Conf.*, pp.524-529, June, 2002.

[59] A. Ye, J. Rose, and D. Lewis, "Architecture of Datapath-Oriented Coarse-Grain Logic and Routing for FPGAs," *Custom Integrated Circuits Conference,* pp.61-64, 2003.

[60] A. Ye, and J. Rose, "Using Bus-Based Connections to Improve Field-Programmable Gate Array Density for Implementing Datapath Circuits," *International Symposium on Field Programmable Gate Arrays,* pp.3-13, 2005.

[61] http://cs.anu.edu.au/ bdm/nauty

[62] http://public.itrs.net