International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 4, No.1 (Jan-2015)

http://dx.doi.org/10.12785/ijcds/040103

Networks-on-Chip Architecture Customization using Network Partitioning: A System-Level Performance Evaluation

Ahmed A. Morgan[1], Haytham Elmiligi[2], M. Watheq El-Kharashi[3], and Fayez Gebali[4]

[1]Department of Computer Engineering, Cairo University, Giza 12613, Egypt
[2]Department of Computing Science, Thompson Rivers University, Kamloops, BC, Canada, V2C OC3
[3]Department of Computer and Systems Engineering, Ain Shams University, Cairo 11517, Egypt
[4]Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada V8W 3P6

Received 16 Sep. 2014, Revised 10 Oct. 2014, Accepted 6 Nov. 2014, Published 1 Jan. 2015

Abstract: Networks-on-Chip (NoC) design is a trade-off between cost and performance. To realize the best trade-off between these factors, researchers have recently proposed using network partitioning techniques to customize the NoC architecture according to the application requirements. In this paper, the impact of using partitioning on different NoC metrics; namely, power, area, and delay, is analyzed. We present a system-level methodology to evaluate the performance of using partitioning-based architecture customization techniques with NoC. Our methodology is applied onto synthetic traffic as well as a number of real NoC benchmarks with different number of cores. Finally, we mathematically formulate evaluation factors that could be used as measures of the enhancements achieved by using partitioning.

Keywords: Networks-on-Chip (NoC), Network partitioning, NoC architecture, NoC performance analysis

1. Introduction

Modern multicore systems encapsulate an impressive number of modules on a single chip. The performance of these multicore systems is limited by the communication among their modules. Networks-on-Chip (NoC) is proposed to efficiently handle the communication between these modules [1, 2].

NoC designers tried to customize the design of the on-chip network to comply with the application requirements. The efficiency of an NoC design depends primarily on its architecture. Therefore, NoC designers were in need to different techniques that carefully customize the network architecture to meet the application requirements while minimizing both design-time and run-time costs. Run-time cost is dominated by the power consumption, whereas the design area is the key element in determining the design-time cost. Researchers proposed using network partitioning techniques to realize customized network architectures. The use of network partitioning with NoC is proposed on the system-level [3], on the circuit-level [4, 5], and for 3D NoCs [6, 7]. In this paper, we aim at fairly evaluating the performance of using network partitioning for NoC architecture customization at the system-level. More precisely, this paper presents three main contributions:

1) *Analyzing the impact of network partitioning on different NoC metrics; namely, power consumption, area, and delay,*

2) *Presenting an evaluation methodology for partitioning-based topology customization techniques for NoC. Different comparative results are obtained by applying the proposed methodology onto real NoC benchmark applications, and*

3) *Mathematically formulating, based on the results obtained from our methodology, a set of system-level evaluation factors that approximate the enhancements that could be achieved by using partitioning.*

This paper is organized as follows. Related work is discussed in Section 2. Section 3 gives some background from graph theory on how to represent different NoC applications. The power, area, and delay evaluation models, employed in this paper, are represented in Section 4. The analysis of the impact of using network partitioning on NoC area, power, and delay is investigated in Section 5. Section 6 presents the proposed

E-mail: ahmorgan@uvic.ca, haytham@ece.uvic.ca, watheq@ece.uvic.ca, Fayez@ece.uvic.ca

evaluation methodology and discusses the tools and heuristics that could be used in its implementation. Section 7 represents our experimental results for both synthetic and real NoCs traffics. We draw conclusions and give ideas for future work in Section 8.

2. Related Work

The advantages of using custom architectures over standard ones for NoC were discussed in [8]. In [9], Ogras and Marculescu proposed a methodology for NoC custom architecture generation by adding some long-range links to the standard mesh topology. The long-range insertion methodology tried to connect distant nodes that communicate with each other frequently with direct links. Their methodology proved useful in reducing the average latency and improving the throughput. Other researchers tried to use optimization-based techniques, like linear programming, swarm intelligence, and genetic algorithms, to produce an optimum architecture for any NoC-based system [4, 10, 11, 12].

Network partitioning techniques are widely used in many fields, such as parallel computing, power system analysis, and VLSI design to divide a large network into smaller partitions [13, 14]. These techniques could divide the network into exactly two subnets (two-way partitioning [15]), or more than two subnets (multiple-way partitioning [16, 17]). In the literature, the network partitioning problem is usually presented as a graph partitioning one. Such a graph consists of a set of vertices (representing the nodes) and a set of weighted edges (representing the communications between these nodes). However, this graph partitioning problem is known to be an NP-hard one [18]. Therefore, many heuristics and optimization-based techniques were proposed to solve it. For example, an Artificial Intelligence (AI)-based method was used in [19], a stochastic-based method was used in [20], the geometric information of the graph was used in the inertial method in [21], and the eigenvectors of the Laplacian matrix was used in the spectral method in [22].

Simple heuristics, like linear, scattered, and random algorithms [23] were also used to quickly carry out the partitioning without considering the weights of the vertices or the edges. However, the most popular and widely used partitioning heuristic is the Kernighan-Lin (KL) [24] and its derivatives, like Fiduccia-Mattheyses (FM) [25] and Min-Cut [26]. These heuristics aimed at minimizing the total weights of the edges cut due to partitioning. In a nutshell, in [18], Chamberlain surveyed and compared different partitioning techniques focusing on their application onto parallel computing. Finally, many public partitioning software packages, like Chaco [23] and PARMETIS [27] could be employed to carry out network partitioning.

The possibility of using network partitioning techniques for NoC architecture customization was highlighted in [3]. In that study, a tool, OIDIPUS, was proposed to map application cores onto a restricted architecture of two rings connected together, as two partitions. Although the work presented in [3] did not formulate or evaluate the use of network partitioning techniques with on-chip networks, it paved the road for using such techniques in NoC architecture generation. Therefore, different research groups started using network partitioning techniques with NoCs. First, for voltage and frequency islands NoC-based systems, network partitioning techniques were used to divide the whole systems into partitions that were implemented as separate islands [28, 29]. The use of partitioning with voltage and frequency island NoC-based systems was evaluated in [30]. Second, for multicast 2D NoCs routing, network partitioning techniques were used to enhance the bandwidth efficiency and the overall performance of NoC-based systems in [31, 32]. Similar adaptive unicast/multicast routing techniques were used to reduce the packet latency for 3D NoCs based on hamiltonian path partitioning in [6].

For on-chip architecture realization, in parallel to our work, some research groups also advocated using network partitioning to customize the underlying NoC architecture. First, a latency-oriented greedy algorithm was presented in [33] to divide any NoC into subnets that were implemented as ordinary bus-based systems. These buses were then connected together as a mesh architecture to construct the whole network. Second, a circuit-level customization methodology was presented in [5] to reduce the power consumption of 2D NoC using network partitioning. Third, for 3D NoCs, an iterative algorithm was presented in [34] to divide a large network into smaller partitions. Each partition was then implemented on a separate layer. Layers were then connected together to construct the overall 3D network architecture. A tool, SunFloor 3D, was built on this algorithm and presented in [35] to generate and synthesize 3D architectures for NoC-based systems. Different partitioning schemes, which were presented for 3D NoC architecture generation, were surveyed and evaluated in [36]. Fourth, for multimedia and other bandwidth constrained applications, network partitioning techniques were used in [37] to build a low energy tree-based architecture that is suitable for these applications. A similar study was presented in [38] that combines network partitioning with a Rectilinear Steiner Tree (RST) algorithm to reduce the power consumption of NoC-based systems.

The work presented in this paper carries out a fair cost and performance evaluations on using network partitioning techniques for NoC architecture generation at the system-level. The proposed evaluation methodology is completely general and puts no restriction on the number, size, or architecture of partitions.

3. Graph Theory Representation of Applications

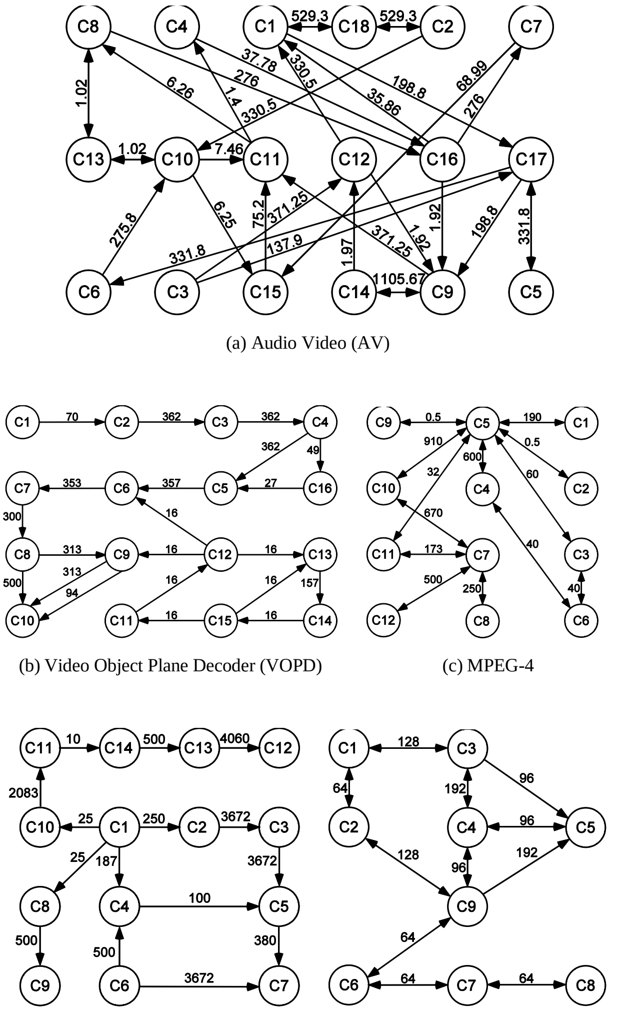
Graph theory concepts could be used to represent NoC applications at the system-level. Given the application description and traffic characteristics, any application could be described by a core graph [39, 40]. A core graph represents the processing elements in the application and the traffic between them. For example, Figure 1 shows sample core graphs of five real NoC benchmark applications. These are an Audio Video (AV) benchmark with 18 cores [41, 42], a Video Object Plane Decoder (VOPD) benchmark with 16 cores [39, 43], an MPEG-4 decoder with 12 cores [40], a 263 decoder MP3 decoder (263DEC) with 14 cores [4], and a Multi-Window Display (MWD) benchmark with 9 cores [41]. As shown in the figure, a core graph is a directed graph G(C, E), where each vertex $c_i \epsilon C$ represents a core in the application, and the directed edge $e_{ij} \epsilon E$ represents the communication between cores c_i and c_j. The weight λ_{ij} of the edge e_{ij} expresses the number of packets/time step sent from core c_i to core c_j . Using NoC terminology, a core graph is usually represented in the form of a traffic distribution matrix (Λ) [44, 45]. For N cores, the dimensions of the matrix is N×N. Any element λ_{ij} in the matrix represents the weight of the edge e_{ij} . That is, the number of packets sent from core c_i to core c_j per time step.

4. Evaluation Models

In this section, we present the cost and performance models used in this paper.

A. PowerModel

NoC power requirements are mainly caused by its routers and links. To model the router power on the system-level, we implemented a library of output queuing routers with different number of ports and buffer sizes [46, 47]. Ports within the same routers have equal buffer sizes and each of them consists of two channels for packets reception and transmission, respectively. A round robin scheduler serves backlogged queues at the output one after another in a fixed order. For each of the routers we implemented, we carried out power simulations at various operating frequencies and target technologies. Power simulations are performed using Synopsys® Design Compiler™, VHDLSIM™, and Power Compiler™ tools. Our approach for router power modeling is presented in details in [48]. For example, for 0.18 μm technology and with an operating frequency of 500 MHz, Table I shows sample results of the power consumptions of 8-flit routers with different number of ports at various average flit arrival rates, in *flit/cycle*. The last row represents the routers leakage power only.



(a) Audio Video (AV)

(b) Video Object Plane Decoder (VOPD) (c) MPEG-4

(d) 263 Decoder MP3 Decoder (263DEC) (e) Multi-Window Display (MWD)

Figure 1. Examples of core graphs for different NoC benchmarks.

TABLE I. POWER CONSUMPTION OF NOC OUTPUT QUEUING ROUTERS WITH DIFFERENT NUMBER OF PORTS AT VARIOUS FLIT ARRIVAL RATES FOR 0.18MM TECHNOLOGY AND WITH AN OPERATING FREQUENCY OF 500MHZ.

Flit Arrival rate (flit/cycle)	**Total Power (P_R) (mW)**				
	4-port	***5-port***	***6-port***	***7-port***	***8-port***
1.000	64.104	96.885	136.044	137.379	234.287
0.400	32.019	48.440	68.041	86.709	117.173
0.200	12.793	19.380	27.229	34.706	46.901
0.100	6.410	9.705	13.635	17.372	23.481
0.050	3.211	4.862	6.832	8.705	11.762

Flit Arrival rate (flit/cycle)	Total Power (P_R) (mW)				
	4-port	*5-port*	*6-port*	*7-port*	*8-port*
0.020	1.293	1.963	2.747	3.505	4.726
0.002	0.135	0.203	0.285	0.380	0.487
0.000	0.008	0.013	0.018	0.025	0.032

From the experiments we conducted, for constant buffer size routers, we notice linear dependencies of both dynamic and leakage powers on the number of ports and the average flit arrival rate. Therefore, we used linear curve fitting to calculate both powers for routers with any number of ports and flit arrival rate. As a result, the power of router *i* is modeled on the system-level as

$$P_{Ri} = P_{RiDynamic} + P_{RiLeakage} \quad (1)$$
$$P_{RiDynamic} = (k_{Dp} \cdot p_i + k_D) \cdot \alpha_{fi} \quad (2)$$
$$P_{RiLeakage} = (k_{Lp} \cdot p_i + k_L) \cdot \alpha_{fi} \quad (3)$$

where $P_{RiDynamic}$, $P_{RiLeakage}$, and P_{Ri} are the dynamic, leakage, and total power of router i, respectively. p_i and α_{fi} are the number of ports and the average flit arrival rate over all the ports of router i, respectively. For any architecture, the average flit injection rate could be calculated for each router by the knowledge of the NoC operating frequency, the employed routing strategy, and the traffic distribution matrix (Λ), discussed in Section 3. Finally, k_{Dp}, k_D, k_{Lp}, and k_L are the dynamic port-dependent, the dynamic port-independent, the leakage port-dependent, and the leakage port-independent power constants, respectively. These constants are technology-dependent and could be obtained by linear curve fitting of the power simulation results of the employed routers. The power simulation should be done once for each router in the designer library. The resultant power constants will then be used in evaluating and generating different NoC architectures. Consequently, for any architecture with N_R routers, the total router power (P_R) is represented as

$$P_R = \sum_{i=1}^{N_R} P_{Ri} \quad (4)$$

For link power modeling, NoC links are either global or local. Global links connect routers, whereas local links connect cores to routers. Global links are usually long and require driving circuits and repeaters [49]. Therefore, it is shown in [50, 51] that the overall link power, area, and delay are completely dominated by those of global links. Accordingly, throughout this paper, the link models proposed and the experimental results presented are to abstract these global links.

To send a certain traffic from core c_i to core c_j, the power consumed in links depends on the amount of traffic transferred, λ_{ij}, the physical characteristics of the links, and the number of hops between the source and the destination cores. Link physical characteristics could be obtained from the corresponding technology library. Moreover, the number of hops between any two cores could be obtained from the architecture connectivity matrix (C) [52]. Any element c_{ij} in the connectivity matrix represents the number of hops between routers R_i and R_j. The connectivity matrix, in turn, is calculated from the adjacency matrix (A) of the architecture [53]. For an architecture with N_R routers, the adjacency matrix is a binary $N_R \times N_R$ matrix, whereas $a_{ij} = 1$ only if there is a direct link connecting between routers R_i and R_j. Figure 2 shows an example for the adjacency and the connectivity matrices of a 6-node ring architecture. In this paper, the connectivity matrix is calculated from the adjacency matrix using the Dijkstra shortest path algorithm [54]. Accordingly, given that every core is connected to exactly one router, NoC link power (P_L) is represented in the system-level as

$$P_L = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_{ij} \,.\, c_{ij} \,.\, P_{\text{unit link}} \quad (5)$$

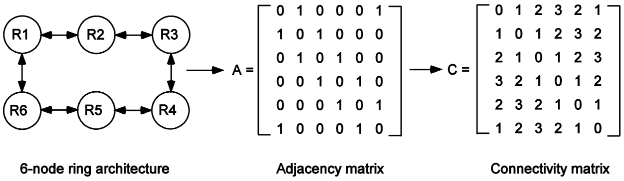
where N is the number of cores in the application. λ_{ij} is the traffic sent from core c_i to core c_j in packet/time step. c_{ij} is the number of hops between the same two cores. $P_{\text{unit link}}$ is the power consumed by a single packet in one unit link. Previous models were proposed to represent this unit link power based on parameters from the targeted fabrication technology. In this paper, we employ the unit link model presented in [55]

$$P_{\text{unit link}} = P_{\text{switching}} + P_{\text{short}} + P_{\text{Leakage}} \quad (6)$$
$$P_{\text{switching}} = 0.5.N_W.V^2_{dd}.(C_S.\alpha_S + C_C.\alpha_C).f_{op} \quad (7)$$
$$P_{\text{short}} = N_W \,.\, \tau_{SC} \,.\, \alpha_S \,.\, V_{dd} \,.\, I_{\text{short}} \,.\, f_{op} \quad (8)$$
$$P_{\text{Leakage}} = N_W \,.\, V_{dd} \,.\, (I_{\text{bias, wire}} + I_{\text{leak}}) \quad (9)$$

where $P_{\text{switching}}$, P_{short}, and P_{Leakage} are the link switching, short circuit, and leakage power, respectively. The summation of $P_{\text{switching}}$ and P_{short} represents the dynamic power consumed in a link. N_w is the number of wires in a link, i.e., channel width, and V_{dd} is the supply voltage. C_S and C_C represent self and coupling capacitance of a wire and with neighboring wires, respectively. Similarly, α_S is the switching activity on a wire and α_C is the switching activity from the adjacent wires. f_{op} denotes the operating frequency and τ_{sc} is the short circuit period during which I_{short} flows between source and ground. Finally, $I_{\text{bias,wire}}$ represents the current flowing from the wire to its substrate and I_{leak} is the leakage current flowing from the source to ground regardless of the gate's state and switching activity. All the parameters for link power calculation could be obtained from the Predictive Technology Model (PTM) [56].

Finally, the total NoC power consumption (P_{NoC}) is represented by

$$P_{NoC} = P_R + P_L \quad (10)$$



$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 3 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 0 \end{bmatrix}$$

Figure 2. An example for adjacency (A) and connectivity (C) matrices of a 6-node ring architecture.

B. Area Model

Similar to NoC power, NoC area consists of the area of its routers and links. As explained in Section 4.1, we are using a synthesized library of pre-designed routers with different number of ports. The area of each router in the library is calculated. Router area is found to be proportional to the number of ports and the buffering within these ports. The total router area (A_R), in μm^2, is expressed as

$$A_R = \sum_{i=1}^{N_R} A_{Ri} \qquad (11)$$

where N_R is the number of routers in the network and A_{Ri} is the area of router i.

The link area depends on the number of wires per each link (i.e., the channel width), the wire length and width, and the spacing between wires. In this paper, we assume a fixed channel width. The wire width and the spacing between wires are technology-dependent and could be obtained from the PTM [56]. As a result, the link area (A_L), in μm^2, is expressed, as in [49], as

$$A_L = N_L \,.\, (N_W \,.\, (w_w + s_w) + s_w) \,.\, l_l \qquad (12)$$

where N_L is the number of links within the network, and N_w is the channel width. w_w, s_w, and l_l are the wire width, the inter-wires spacing, and the wire length for global interconnects, respectively.

Finally, the total NoC area (A_{NoC}), in μm^2, is expressed as

$$A_{NoC} = A_R + A_L \qquad (13)$$

C. Delay Model

The traffic from any source core experiences three types of delays in its way to the destination core. These delays are the arbitration and propagation delays through routers, the propagation delay through links, and the serialization and de-serialization delays through NIs. In this paper, we use the average zero-load delay model presented in [57]. The average zero-load delay model ignores any competition on NoC resources and assumes a contention-free network. It is a fast system-level model to check for the effect of different architectures on NoC delay. This delay model was originally proposed for 3D NoCs. However, it could also be adopted with 2D NoCs that we target in this paper. Accordingly, the overall NoC average delay (D_{NoC}), in seconds, is represented as

$$D_{NoC} = \mu.(t_a+t_r) + (\mu+1).t_l + (N_b/N_w).t_l \qquad (14)$$

where

$$t_a = (5.25.\log_2 (p_{avg}) + (14/12) + 9).\tau \qquad (15)$$

$$t_{l(r)} = 0.377.r_{l(r)}.c_{l(r)}.l^2_{l(r)} + 0.693.(R_{d0}.C_0 + R_{d0}.c_{l(r)}.l_{l(r)} + r_{l(r)}.l_{l(r)}.C_{g0}) \qquad (16)$$

$$l_r = 2.(w_r + s_r).N_W.p_{avg} \qquad (17)$$

$$R_{d0} = 0.98.V_{dd}/I_{d0} \qquad (18)$$

where μ is the average internode distance [9] (i.e., average number of routers between a source node and a destination node). t_a and t_r are the router arbitration and propagation delays, respectively. Similarly, t_l is the link propagation delay. N_b and N_w are the number of bits per packet and the channel width, respectively. p_{avg} is the average number of ports per router and τ is the delay of a minimum-size inverter of the target technology. $r_{l(r)}$ and $c_{l(r)}$ are the per unit length resistance and capacitance of the link (router) wires, respectively. $l_{l(r)}$ is the wire length for the link (router crossbar). R_{d0} and C_{g0} are the equivalent output resistance and the gate capacitance of a minimum-size inverter of the target technology, respectively. C_0 is the total input capacitance of a minimum-size inverter of the target technology, which is the summation of the gate and the drain capacitances. w_r and s_r are the wire width and the inter-wires spacing for router internal interconnects, respectively. V_{dd} is the supply voltage and I_{d0} is the drain current when both the drain and the gate voltages are equal to the supply voltage. The values of all the technology-dependent parameters could be obtained from the PTM [56]. Finally, for an N-cores application, the average internode distance (μ), in hops, could be calculated using the connectivity matrix (C) of the architecture and the traffic distribution matrix (Λ) of the application as

$$\mu = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_{ij} \,.c_{ij}}{\sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_{ij}} \qquad (19)$$

5. NoC Partitioning: Power, Area, and Delay Analysis

Applying network partitioning techniques on NoC affects different design variables, like the number of ports per router, the number of links, and the average internode distance. As a result, using network partitioning changes the values of NoC cost and performance metrics. In this section, we analyze the impact of applying network partitioning techniques on different NoC metrics. In a nutshell, we found that network partitioning always reduces the area, often reduces the power, and sometimes reduces the delay. Subsections 5.A, 5.B, and 5.C discuss the enhancement, or the deterioration, that results from using network partitioning techniques on NoC power consumption, area, and delay, respectively.

A. Power Analysis

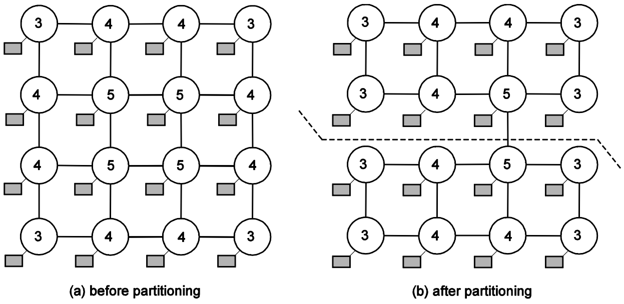
The number of ports within a router constitutes the major factor of the router's power consumption. Routers with different configurations were studied in [58, 59] and their power consumptions were proved to be dominated by the number of ports as well. We showed in Table I that the power of an output queuing router increases significantly as the number of ports increases. From the results in that table, we notice that, for any flit arrival rate, doubling the number of ports results in quadrupling the total router power consumption. Therefore, it is of great importance for any NoC architecture to check the total power consumed by routers that have low number of ports.

Network partitioning results in reducing the number of ports for the routers employed to generate the on-chip network, which in turn reduces the power consumption of the resultant NoC architecture. Furthermore, network partitioning reduces the link power by reducing the total number of links within the generated architecture. However, network partitioning increases the average internode distance taking some traffic through longer routes. For any traffic trace, taking more hops results in increasing the power consumption. Therefore, partitioning outcomes, with respect to power consumption, are indeed conflicting. Therefore, it is dependent on the application whether partitioning results in reducing the overall power consumption or not. By experimentation with many NoC benchmarks, as will be explained in Subsection 7.A, the advantages of reducing the number of ports and links often outperform the deterioration happened by increasing the average internode distance. However, to avoid performing time-consuming power simulation and application traffic routing and to mathematically decide at the system-level whether partitioning will be power beneficial or not, we present a partitioning power factor (η_P). Based on the power equations in Subsection 4.A, the power factor is an approximation of the ratio of the post-partitioning power consumption to the pre-partitioning one. The proposed power factor is proved by experimentation, as will be explained in Subsection 7.E, to be a good approximation to decide whether using partitioning results in reducing the overall NoC power or not. The proposed partitioning power factor is represented as

$$\eta_P = \frac{P_{\text{partitioned}}}{P_{\text{unpartitioned}}} \approx \frac{p_{avgp}\,.\,\mu_p\,.\,p_{avgp}}{p_{avgn}\,.\,\mu_n\,.\,p_{avgn}} \qquad (20)$$

where $P_{\text{partitioned}}$ and $P_{\text{unpartitioned}}$ are the total NoC power consumptions with and without partitioning, respectively, according to (10). p_{avgp} and μ_p are the average number of ports per router and the average internode distance when partitioning is carried out. Similarly, p_{avgn} and μ_n are the average number of ports per router and the average internode distance when partitioning is not carried out and all the nodes in the application are realized with one large non-partitioned network. Consequently, if the value of η_P is less than 1, using network partitioning is useful with respect to the power metric. In other words, the power saved by removing ports and links using partitioning outperforms the power rise by increasing the average internode distance. In contrast, if η_P is greater than 1, the average internode distance is more dominant for this application and partitioning will not result in any power saving. Finally, η_P could be used to approximate the amount of power saving. The percentage of power saving by using partitioning is represented as

$$\% \text{ Power Savings} = 100\,.\,(1 - \eta_P) \qquad (21)$$



(a) before partitioning

(b) after partitioning

Circles represent routers and rectangles represent local processing cores.
Number in each circle represents the number of ports of the router in this node.

Figure 3. Mesh implementation of a 16-core application before and after partitioning. The two partitions are separated by the dotted line in (b).

TABLE II. NUMBER OF ROUTERS AND TOTAL AREA BEFORE AND AFTER PARTITIONING OF A 16-NODE APPLICATION, MAPPED ONTO A MESH ARCHITECTURE, AS SHOWN IN FIGURE 3.

	5-port router	4-port router	3-port router	Total area (μm^2)
Before Partitioning	4	8	4	1,634,400
After Partitioning	2	6	8	1,497,300

B. Area Analysis

The ever-increasing on-chip communication requirements for modern applications force NoCs to replace ordinary shared buses. Nevertheless, NoC area is not supposed to exceed 10% of the whole design area [2]. The area of on-chip networks is mainly determined by routers area [43]. From the implementation results of our output queuing routers, we found that ports with their buffers constitute the major percentage of any router area. Similarly, the areas of routers with different configurations were shown in [43] to be directly proportional to the number of ports as well. Therefore, the less the number of ports of routers used in generating the architecture, the lower the NoC area.

Network partitioning divides a large network into smaller partitions connected together. Smaller partitions require routers with less number of ports than those used in a single unpartitioned network. As a result, partitioning reduces NoC router area. Using network partitioning also removes some of the links and reduces NoC link area. Therefore, using network partitioning techniques in generating NoC architecture guarantees a certain ratio of area reduction.

As an example on the area reduction resulting from using network partitioning, Figure 3 shows a mesh implementation of a 16-node application before and after partitioning. The two partitions are separated by the dotted line in Figure 3(b). On one hand, partitioning results in removing three links in this example. On the other hand, Table II summarizes the total number of routers needed with different number of ports before and after partitioning and the total NoC area. As a conclusion, using network partitioning reduces the NoC area by 9.16% in this example.

As the number of nodes becomes larger, the number of links removed by partitioning increases. By removing a link, the number of ports of the two routers that were connected by this link decreases by one. Consequently, the router area decreases in a nearly linear proportion with the number of ports removed. Therefore, the more cores existing in the application, the more links removed by partitioning, and the more routers area saved. As a result, partitioning becomes more useful in reducing the area cost as the number of cores in the application increases.

C. Delay Analysis

The effect of using network partitioning on NoC delay represents a similar trade-off like that of the power consumption. On one hand, partitioning reduces the routers arbitration and propagation delays by reducing the number of ports. On the other hand, the traffic from some nodes, other than those on the boundary, in a specific partition needs more hops to reach the destination nodes in other partitions. This rise in the average internode distance increases the NoC delay. Accordingly, it is first required to formulate the partitioning problem to minimize the traffic exchanged between different partitions. We decided to use the FM algorithm in building our partitioning methodology (lets call our methodlogy NoC-FM), as it is mainly used to minimize the inter-partition traffic. The advantages of using the FM algorithm with NoC are threefold. First, it mitigates the effect of partitioning on the average internode distance. Second, it avoids creating bottlenecks in the NoC through inter-partition links. Third, it reduces the buffering required at those nodes located at the boundary of partitions.

The trade-off between conflicting design variables makes the use of partitioning an application-dependent with respect to the delay metric. Therefore, we propose our partitioning delay factor (η_D) to approximately quantify the advantage, if any, of using partitioning on NoC delay. Similar to the power factor, the partitioning delay factor is based on the delay equations in Subsection 4.C and decides quickly at the system-level whether the use of partitioning with a specific application will be useful with respect to the delay or not. It is proved by experimentation, as will be represented in Subsection 7.E, to be a good approximation of the ratio between post-partitioning and pre-partitioning delays. Our partitioning delay factor is represented as

$$\eta_P = \frac{D_{\text{partitioned}}}{D_{\text{unpartitioned}}} \approx \frac{\mu_p \cdot (p_{avgp} + \log_2(p_{avgp})}{\mu_n \cdot (p_{avgn} + \log_2(p_{avgn})} \qquad (22)$$

where Dpartitioned and Dunpartitioned are the overall NoC delay with and without partitioning according to (14). p_{avgp} and μ_p are the average number of ports per router and the average internode distance when partitioning is carried out. Similarly, p_{avgn} and μ_n are the average number of ports per router and the average internode distance when partitioning is not carried out and all the nodes in the application are realized with one large non-partitioned network. If the value of the partitioning delay factor is less than 1, using network partitioning is performance-efficient. In other words, the delay reduction by lowering the arbitration and propagation delays resulted from reducing the number of ports per router outperforms the delay rise by increasing the average internode distance. In contrast, if η_D is greater than 1, the average internode distance is more dominant for this application and partitioning will not result in any delay enhancement. Finally, the delay factor could be used to approximate the amount of delay reduction. The percentage delay reduction by using partitioning is represented as

$$\% \text{ Delay Reduction} = 100 \ . \ (1 - \eta_D) \qquad (23)$$

6. PROPOSED METHODOLOGY

In this section, we present our cost-efficient methodology for NoC custom architecture generation using network partitioning. The proposed methodology is shown in Figure 4. The methodology aims at comparing the partitioned and unpartitioned architectures to decide which one is better for the given application. The methodology consists of seven main steps. In the first step, the supply voltage (V_{dd}), frequency of operation (f_{op}), the application core graph, and the targeted fabrication technology are entered to the methodology. Moreover, the required number of partitions, the standard architecture for each partition, and the mapping technique should also be provided in the first step. (The optimum selection of the number of partitions is left for the future work.)

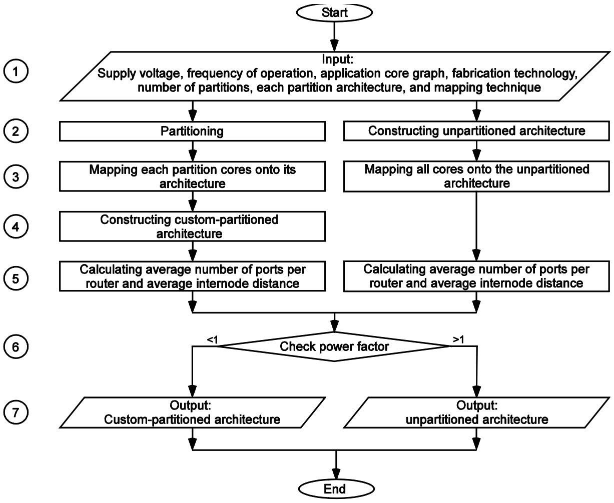

Start
Input:
Supply voltage, frequency of operation, application core graph, fabrication technology, number of partitions, each partition architecture, and mapping technique
1
2
Partitioning
Constructing unpartitioned architecture
3
Mapping each partition cores onto its architecture
Mapping all cores onto the unpartitioned architecture
4
Constructing custom-partitioned architecture
5
Calculating average number of ports per router and average internode distance
Calculating average number of ports per router and average internode distance
6
<1
Check power factor
>1
7
Output:
Custom-partitioned architecture
Output:
unpartitioned architecture
End

Figure 4. Proposed cost-efficient methodology for custom architecture generation usingnetwork partitioning.

In the second step, the core graph is divided into smaller partitions according to our NoC-FM algorithm. Other partitioning schemes could also be used without loss of generality. Public partitioning software packages, like Chaco [23] and PARMETIS [27] could be employed in this step. Simultaneously, in the second step, an unpartitioned standard architecture is constructed. In the third step, all application cores are mapped onto that unpartitioned architecture. In parallel, cores in each partition are also mapped onto the architecture corresponding to that partition. Similar to the partitioning step, any mapping technique could be used with our methodology. Nevertheless, NMAP is proved to outperform PMAP, GMAP, and PBB in [39]. Moreover, NMAP is a general technique that could map cores onto any architecture. In the fourth step, partitions are connected in a one-by-one fashion to construct the whole custom architecture for the application. To establish the connection between any two partitions, cores in each partition are ranked according to the amount of traffic exchanged with cores in the other partition. The router connected to the core with the highest rank is selected as the connecting node. Similar to the previous two steps, other connection schemes maybe used to construct a network of partitions. In the fifth step, the average number of ports per routers and the average internode distance are calculated for the partitioned and unpartitioned architectures. In the sixth step, the partitioning power factor is calculated according to (20) to check for the usefulness of using network partitioning with respect to the power of the application in-hand. As our methodology is proposed as a cost-efficient one, we only consider the power factor. However, the delay factor could also be included according to design requirements and application needs. If the power factor is less than 1, the partitioning is useful for the application and the partitioned architecture is outputted in the seventh step. Nevertheless, if the factor is greater than 1, partitioning only enhances the area not the power of that application and the single unpartitioned architecture is outputted.

7. Experimental Results

The proposed methodology is applied to four of the real benchmark applications shown in Figure 1: VOPD benchmark with 16 cores, 263DEC benchmark with 14 cores, MPEG-4 benchmark with 12 cores, and MWD benchmark with 9 cores. The performance metrics of the architecture resulting from our network partitioning methodology (NP) are compared to that of a standard unpartitioned mesh architecture (MSH). The performance of the custom architecture generated by the long-range link insertion methodology (LR), proposed in [9] and discussed in Section 2, is also considered for the comparison. The three architectures are represented throughout the comparison as NP, MSH, and LR, respectively. As the number of cores in the considered benchmarks is not large, the number of partitions is chosen to be two. NMAP is employed for the mapping and a mesh architecture is used for all partitions. The reason behind the mesh topology selection is twofold. First, grid-based architectures, like mesh, could be easily implemented inside chips. Second, most of practical NoC implementations are done with mesh architectures [60].

A. Power Evaluation

Table III shows the results of evaluating the three architectures in the study with respect to the power metric in values that are absolute and normalized to those of our methodology. Using our methodology, NoC power is reduced by 9.38%, 1.86%, 6.34%, and 1.9% with respect to the standard architecture for the VOPD, 263DEC, MPEG-4, and MWD benchmarks, respectively. Similarly, our architectures are more power-efficient than those of the custom long-range methodology by 11.5%, 4.1%, and 6.46% for the VOPD, MPEG-4, and MWD benchmarks, respectively. For the 263DEC benchmark, our methodology results in the same power consumption like the custom long-range one. Although the two customization methodologies end up with the same power consumption, the power reduction mechanism for both is different. Our methodology mainly reduces the router power with a slight increase on the link power, whereas, the custom long-range methodology mainly reduces the link power with a slight increase on the router power. The total power for both is the same, which is 1.86% less than that of a standard mesh architecture. Finally, on average over the four benchmarks, the power of the architecture resulted from our methodology is less than that of the standard mesh architecture by 4.87% and that of the custom long-range architecture by 5.52%. These results show that our partitioning-based methodology often outperforms other standard and custom architecture realization techniques with respect to the power metric. As network partitioning certainly reduces the area cost of any application, it is an efficient way to reduce the two NoC cost metrics: power and area.

TABLE III. POWER COMPARISON BETWEEN DIFFERENT ARCHITECTURES IN THE STUDY FOR DIFFERENT BENCHMARK APPLICATIONS. (% IS CACLCULATED BY THE NORMALIZATION OF VALUES TO THOSE OF THE NP ARCHITECTURE.)

Benchmark	Architecture					
	NP		*MSH*		*LR*	
	Watts	%	Watts	%	Watts	%
VOPD	0.2600	100.00	0.2844	109.38	0.2899	111.50
263DEC	1.2320	100.00	1.2549	101.86	1.2319	100.00
MPEG-4	0.5219	100.00	0.5550	106.34	0.5433	104.10
MWD	0.1733	100.00	0.1766	101.90	0.1845	106.46
Average		100.00		104.87		105.52

TABLE IV. AREA COMPARISON BETWEEN DIFFERENT ARCHITECTURES IN THE STUDY FOR DIFFERENT BENCHMARK APPLICATIONS. (% IS CACLCULATED BY THE NORMALIZATION OF VALUES TO THOSE OF THE NP ARCHITECTURE.)

Benchmark	Architecture					
	NP		*MSH*		*LR*	
	mm^2	%	mm^2	%	mm^2	%
VOPD	1.4973	100.00	1.6344	109.16	1.7258	115.26
263DEC	1.2473	100.00	1.3387	107.33	1.3844	110.99
MPEG-4	1.0887	100.00	1.1801	108.40	1.2715	116.79
MWD	0.7594	100.00	0.8508	112.04	0.9422	124.07
Average		100.00		109.23		116.78

B. Area Evaluation

Table IV shows the results of evaluating the three architectures in the study with respect to the area reduction efficiency in values that are absolute and normalized to those of our methodology. Using our methodology, NoC area is reduced by 9.16%, 7.33%, 8.4%, and 12.04% with respect to the standard architecture and by 15.26%, 10.99%, 16.79%, and 24.07% with respect to custom long-range architecture for the VOPD, 263DEC, MPEG-4, and MWD benchmarks, respectively. On average over the four benchmarks, the area resulted from our methodology is less than that of the standard mesh architecture by 9.23% and that of the custom long-range architecture by 16.78%. These results prove that our partitioning-based methodology always outperforms other standard and custom architecture generation techniques with respect to the area metric.

C. Delay Evaluation

Table V shows the results of evaluating the three architectures in the study with respect to the delay efficiency in values that are absolute and normalized to those of our methodology. The delay of the architectures generated by our methodology is found to be slightly better than that of standard mesh architectures by 1.88%, 1.97%, and 1.79% for the VOPD, 263DEC, and MPEG-4 benchmarks, respectively. However, for the MWD benchmark, the standard mesh architecture is more delay-efficient than that generated by our methodology by 4.53%. As previously explained in Subsection 5.C, the delay enhancement resulted from our methodology is application-dependent. For our methodology to achieve some delay enhancement, the inter-partition traffic should be low with respect to the total application traffic. Keeping the inter-partition traffic low allows the gains in reducing the router arbitration and propagation delays to overcome the deterioration happened by taking the inter-partition traffic through long routes. In the literature, the traffic locality factor is used to express the ratio of the intra-partition traffic to the total application traffic [61]. As the value of this traffic locality factor goes close to 1, the traffic becomes highly localized and the effect of partitioning on the average internode distance becomes more insignificant. For the MWD benchmark, unlike other benchmarks, this traffic locality factor is found to be 0.784, which means that the inter-partition traffic is 21.6% of the total traffic. This high inter-partition traffic explains why our partitioning-based methodology could not achieve any delay enhancement for the MWD benchmark. In a nutshell, our methodology is expected to enhance the delay only for those applications that exhibit some tarffic locality. Finally, on average over the four benchmarks, the delay resulted from the architecture generated by our methodology is 0.28% less than that of the standard mesh architecture.

The architectures generated by our methodology achieve some delay enhancement over standard mesh architectures for three of the benchmarks. However, they could not outperform those of the custom long-range methodology, except for the 263DEC benchmark and by only 0.62%. The latter methodology is mainly proposed to customize NoC architecture with respect to the delay. Therefore, it outperforms our methodology by 1.18%, 8.39%, and 8.81% for the VOPD, MPEG-4, and the MWD benchmarks, respectively. An analysis of the inter-partition traffic and the traffic locality factor of the 263DEC benchmark explains why our methodology outperforms the custom long-range one. For this benchmark, the traffic locality factor is found to be 0.998, and hence, the inter-partition traffic constitutes only 0.25% of the total traffic. Therefore, the advantages of reducing the arbitration and propagation delays by our methodology dominate the overall delay and allow it to achieve a better delay performance for this benchmark.

Finally, on average over the four benchmarks, the custom long-range methodology outperforms ours by 4.44%.

TABLE V. DELAY COMPARISON BETWEEN DIFFERENT ARCHITECTURES IN THE STUDY FOR DIFFERENT BENCHMARK APPLICATIONS. (% IS CACLCULATED BY THE NORMALIZATION OF VALUES TO THOSE OF THE NP ARCHITECTURE.)

Benchmark	Architecture					
	NP		*MSH*		*LR*	
	µs	%	µs	%	µs	%
VOPD	2.440	100.0	2.486	101.88	2.411	98.82
263DEC	12.160	100.0	12.399	101.97	12.235	100.62
MPEG-4	5.014	100.0	5.104	101.79	4.593	91.61
MWD	1.683	100.0	1.607	95.47	1.535	91.19
Average		100.0		100.28		95.56

The above delay discussion expresses a typical trade-off between performance and cost. On one hand, our methodology is proposed as a cost-efficient one, and hence, it enhances power and area costs more than the delay performance. On the other hand, the custom long-range methodology is a performance-efficient one, and hence, it enhances the delay performance on the expense of more power and area costs of the on-chip network.

D. Partitioning Scheme Evaluation

In this subsection, we verify the efficiency of using the FM [25] partitioning technique in our NoC-FM methodology relative to the heuristic-based scattered and random algorithms [23], as well as the spectral method in [22]. Tables VI and VII compare the power and the delay results, respectively, of our NoC-FM methodology with those of other partitioning schemes in values that are absolute and normalized to those of our methodology. As all schemes result in approximately the same area reduction, area comparison is not included in this subsection. From one hand, the tables show that NoC-FM significantly outperforms random and scattered partitioning schemes with respect to both power and delay. On average over the four benchmarks, our technique is 75.36% and 37.09% more power-efficient than the scattered and random techniques, respectively. It is also 68.35% and 34.44% more delay-efficient than the same two techniques, respectively. On the other hand, NoC-FM is slightly better than the spectral scheme by 3.56% and 3.79% from the power and the delay perspectives, respectively. These results emphasize the efficiency of using NoC-FM with respect to on-chip network power and delay.

Tables VI and VII also show that the power and the delay resulted from partitioning techniques other than NoC-FM are worse than those of the standard unpartitioned mesh architecture, shown in Tables III and V, respectively. This clarifies that the advantages of using network partitioning with NoC is completely dominated by the best selection of the employed partitioning technique. It further emphasizes the efficiency of using NoC-FM in enhancing NoC power and delay.

E. Power and Delay Factors Evaluation

In this subsection, we evaluate the accuracy of the proposed partitioning power (η_P) and delay (η_D) factors represented in (20) and (22), respectively. These factors are used to approximate the ratios of the partitioned power and delay to the unpartitioned ones, respectively. We compare the values of these factors to the exact power and delay ratios. For the four benchmarks, TableVIII shows the percentage of error in the two factors. The table shows that the error resulted from using our approximated factors never exceeds 4%. On average over the four benchmarks, the percentage of error in the power factor is 2.13% and in the delay factor is 2.88%. The use of these factors saves a large amount of time in architecture generation. This time corresponds to simulating the router library, deriving the power constants, and routing all packets from sources to destinations. Since the architecture generation time is a crucial factor for NoC design, the use of these factors is indeed very helpful in the quick evaluation of the efficiency of using partitioning for NoC architecture generation.

TABLE VI. POWER COMPARISON BETWEEN DIFFERENT PARTITIONING SCHEMES FOR DIFFERENT BENCHMARK APPLICATIONS. (W IS WATTS AND % IS CACLCULATED BY THE NORMALIZATION OF VALUES TO THOSE OF THE NOC-FM TECHNIQUE.)

Benchmark	Partitioning Scheme							
	NoC-FM		*Scattered*		*Random*		*Spectral*	
	W	%	W	%	W	%	W	%
VOPD	0.26	100	0.57	218	0.33	126	0.26	100
263DEC	1.23	100	2.57	208	2.11	171	1.34	109
MPEG-4	0.52	100	0.79	152	0.57	109	0.52	100
MWD	0.17	100	0.21	123	0.25	142	0.18	105
Average		100		175		137		104

TABLE VII. DELAY COMPARISON BETWEEN DIFFERENT PARTITIONING SCHEMES FOR DIFFERENT BENCHMARK APPLICATIONS. (% IS CACLCULATED BY THE NORMALIZATION OF VALUES TO THOSE OF THE NOC-FM TECHNIQUE.)

Benchmark	Partitioning Scheme							
	NoC-FM		*Scattered*		*Random*		*Spectral*	
	µs	%	µs	%	µs	%	µs	%
VOPD	2.44	100	5.21	214	3.15	129	2.44	100

Benchmark	Partitioning Scheme							
	NoC-FM		*Scattered*		*Random*		*Spectral*	
263DEC	12.2	100	24.3	200	19.8	163	13.0	107
MPEG-4	5.01	100	6.93	138	5.15	103	5.07	101
MWD	1.68	100	2.06	122	2.41	143	1.80	107
Average		100		168		134		104

TABLE VIII. PERCENTAGE OF ERROR IN PARTITIONING POWER AND DELAY FACTORS COLLECTED FROM THE DIFFERENT BENCHMARK APPLICATIONS USED IN THE STUDY.

Factor	Percentage of error (%)				
	Benchmark				*Average error*
	VOPD	263DEC	MPEG-4	MWD	
Power factor (η_P)	0.077	3.56	2.08	2.8	2.13
Delay factor (η_D)	3.93	3.47	3.79	0.32	2.88

8. CONCLUSION AND FUTURE WORK

This paper evaluated the efficiency of using network partitioning on NoC power, area, and delay. A methodology was presented to reduce the power and area costs of NoC using network partitioning techniques. Smaller partitions require less number of ports and buffering, which constitute the main parts of on-chip power and area. The FM partitioning algorithm was adopted with modification to formulate the partitioning problem. Results prove that network partitioning is an effective way to reduce the NoC power and area, especially for large applications that exhibit locality.

We plan to extend this work in different directions. First, we are going to investigate the optimum number of partitions, the optimum number of cores per partition, and the optimum topology for each partition. Second, we will try to build an integrated tool to carry out the whole topology generation steps. Third, we will address the reliability issues resulting from using network partitioning.

REFERENCES

[1] Benini, L., De Micheli, G. “Networks on chips: A new SoC paradigm,” IEEE Computer Magazine, 2002, 35(1), pp.70–78.

[2] Jantsch, A. “NoCs: A new contract between hardware and software,” In Proceedings of the 2003 Euromicro Symposium on Digital System Design (DSD’03), Belek-Antalya, Turkey, 2003, pp. 10–16.

[3] Ahonen, T., Siguenza-Tortosa, D., Bin, H., Nurmi, J. “Topology optimization for application-specific networks-on-chip,” In Proceedings of the 2004 ACM international workshop on System level interconnect prediction (SLIP’04), Paris, France, 2004, pp. 53–60.

[4] Srinivasan, K., Chatha, K., Konjevod, G. “Linear-programming-based techniques for synthesis of network-on-chip architectures,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2006, 14(4), pp. 407–420.

[5] Yu, B., Dong, S., Chen, S., Goto, S. “Floorplanning and topology generation for application-specific network-on-chip,” In Proceedings of the 15th IEEE Asia and South Pacific Design Automation Conference (ASP-DAC’10), Taipei, Taiwan, 2010, pp. 535–540.

[6] Ebrahimi, M., Daneshtalab, M., Liljeberg, P., Tenhunen, H. “Partitioning methods for unicast/multicast traffic in 3D NoC architecture,” In Proceedings of the 13th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS’10), Vienna, Austria, 2010, pp. 127–132.

[7] Seiculescu, C., Murali, S., Benini, L., De Micheli, G. “SunFloor 3D: A tool for networks on chip topology synthesis for 3D systems on chips,” In Proceedings of the IEEE Design, Automation and Test in Europe Conference and Exhibition (DATE’09), Nice, France, 2009, pp. 9–14.

[8] Benini, L. “Application specific NoC design,” In Proceedings of the 2006 IEEE/ACM Conference on Design, Automation and Test in Europe (DATE’06), vol. 1. Munich, Germany, 2006, pp. 491–495.

[9] Ogras, U., Marculescu, R. “It’s a small world after all: NoC performance optimization via long-range link insertion,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2006,14(7), pp.693–706.

[10] Leary, G., Srinivasan, K., Mehta, K., Chatha, K. “Design of network-on-chip architectures with a genetic algorithm-based technique,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2009;17(5), pp.674–687.

[11] Elmiligi, H., Morgan, A.A., El-Kharashi, M.W., Gebali, F. “Networks-on-chip topology optimization subject to power, delay, and reliability constraints,” In Proceedings of the 2010 IEEE International Symposium on Circuits and Systems (ISCAS 2010), Paris, France, 2010, pp. 2354–2357.

[12] Elmiligi, H., Morgan, A.A., El-Kharashi, M.W., Gebali, F. “Improving networks-on-chip performability: A topology-based approach,” International Journal of Circuit Theory and Applications, 2011, 39(6), pp.557–572.

[13] Li, F., Xiao, Y., Zhang, Q. “Graph partition based traffic balance in industrial network,” In Proceedings of the second IEEE International Conference on Communications and Networking in China (CHINACOM’07), Shanghai, China, 2007, pp. 438–442.

[14] Karypis, G., Kumar, V. “Multilevel algorithms for multi-constraint graph partitioning,” In Proceedings of the 1998 IEEE/ACM Conference on Supercomputing (SC’98), San Jose, CA , USA, 1998, pp.1-17.

[15] Cheng, C., Wei, Y. “An improved two-way partitioning algorithm with stable performance,” IEEE Transactions on Computer-Aided Design, 1991, 10(12), pp.1502–1511.

[16] Sanchis, L. “Multiple-way network partitioning,” IEEE Transactions on Computers, 1989, 38(1), pp.62–81.

[17] Sanchis, L. “Multiple-way network partitioning with different cost functions,” IEEE Transactions on Computers, 1993, 42(12), pp.1500–1504.

[18] Chamberlain, B. “Graph partitioning algorithms for distributing workloads of parallel computations,” Tech. Rep., UW-CSE-98-10-03; University of Washington, WA, USA, 1998.

[19] Saab, Y., Rao, V. “An evolution-based approach to partitioning ASIC systems,” In Proceedings of the 26th Annual IEEE/ACM Design Automation Conference (DAC’89), Las Vegas, NV, USA, 1989, pp. 767–770.

[20] Wan, Y., Roy, S., Saberi, A., Lesieutre, B. “A stochastic automaton-based algorithm for flexible and distributed network partitioning,” In Proceedings of the 2005 IEEE Swarm Intelligence Symposium (SIS’05), California, USA, 2005, pp. 273–280.

[21] Simon, H. “Partitioning of unstructured problems for parallel processing,” Computing Systems in Engineering, 1991, 2(2/3), pp.135–148.

[22] Chan, P., Schlag, M., Zien, J. “Spectral K-way ratio-cut partitioning and clustering,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1994, 13(9), pp.1088–1095.

[23] Hendrickson, B., Leland, R. “The Chaco user’s guide: Version 2,” Tech. Rep. SAND95-2344, Sandia National Laboratories, Albuquerque, NM, USA, 1995.

[24] Kernighan, B., Lin, S. “An efficient heuristic procedure for partitioning graphs,” Bell System Technical Journal, 1970, 49(2), pp.291–307.

[25] Fiduccia, C., Mattheyses, R. “A linear-time heuristic for improving network partitions,” In Proceedings of the 19th Annual IEEE/ACM Design Automation Conference (DAC’82), Las Vegas, NV, USA, 1982, pp. 175–181.

[26] Krishnamurthy, B. “An improved min-cut algorithm for partitioning VLSI networks,” IEEE Transactions on Computers, 1984, 33(5), pp.438–446.

[27] Karypis, G., Schloegel, K., Kumar, V. “PARMETIS: Parallel gragh partitioning and sparse matrix ordering library, Version 3.1,” Tech. Rep., University of Minnesota, Minneapolis, MN 55455, USA, 2003.

[28] Ogras, U., Marculescu, R., Marculescu, D., Jung, E. “Design and management of voltage-frequency island partitioned networks-on-chip,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2009, 17(3), pp.330–341.

[29] Rahmani, A.M., Liljeberg, P., Plosila, J., Tenhunen, H. “Developing reconfigurable FIFOs to optimize power/performance of voltage/frequency island-based networks-on-chip,” In Proceedings of the 13th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS’10), Vienna, Austria, 2010, pp.105–110.

[30] Seiculescu, C., Murali, S., Benini, L., De Micheli, G. “Comparative analysis of NoCs for two-dimensional versus three-dimensional SoCs supporting multiple voltage and frequency islands,” IEEE Transactions on Circuits and Systems II: Express Briefs, 2010, 57(5), pp.364–368.

[31] Wang, L., Jin, Y., Kim, H., Kim, E. “Recursive partitioning multicast: A bandwidth-efficient routing for networks-on-chip,” In Proceedings of the Third IEEE International Symposium on Networks-on-Chip (NOCS’09), San Diego, CA, USA, 2009, pp. 64–73.

[32] Wang, L., Kumar, P., Boyapati, R., Yum, K., Kim, E. “Efficient lookahead routing and header compression for multicasting in networks-on-chip,” In Proceedings of the 2010 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS’10), La Jolla, CA, USA, 2010, pp.1-10.

[33] Tsai, K.L., Lai, F., C.-Y. Pan and, D.S.X., Tan, H.J., Lee, H.C. “Design of low latency on-chip communication based on hybrid NoC architecture,” In Proceedings of the 8th IEEE International Northeast Workshop on Circuits and Systems (NEWCAS’10), Montreal, QC, Canada, 2010, pp. 257–260.

[34] Murali, S., Seiculescu, C., Benini, L., De Micheli, G. “Synthesis of networks on chips for 3D systems on chips,” In Proceedings of the 14th IEEE Asia and South Pacific Design Automation Conference (ASP-DAC’09), Yokohama, Japan, 2009, pp. 242–247.

[35] Seiculescu, C., Murali, S., Benini, L., De Micheli, G. “SunFloor 3D: A tool for networks on chip topology synthesis for 3D systems on chips,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2010, 29(12), pp.1987–2000.

[36] Ebrahimi, M., Daneshtalab, M., Liljeberg, P., Tenhunen, H. “Performance analysis of 3D NoCs partitioning methods,” In Proceedings of the 2010 IEEE Computer Society Annual Symposium on VLSI (ISVLSI’10), Lixouri, Kefalonia, Greece, 2010, pp. 479–480.

[37] Majeti, D., Pasalapudi, A., Yalamanchili, K. “Low energy tree based network on chip architectures using homogeneous routers for bandwidth and latency constrained multimedia applications,” In Proceedings of the Second International Conference on Emerging Trends in Engineering and Technology (ICETET’09), Nagpur, India, 2009, pp. 358–363.

[38] Ezhumali, P., Arun, C. “Low power and area consumption custom networks-on-chip architectures using RST algorithms,” International Journal of Computer Science and Information Security (IJCSIS), 2010, 8(6), pp.107–115.

[39] Murali, S., De Micheli, G. “Bandwidth-constrained mapping of cores onto NoC architectures,” In Proceedings of the 2004 IEEE/ACM Conference on Design, Automation and Test in Europe (DATE’04), vol. 2, Paris, France, 2004, pp. 896–901.

[40] Murali, S., De Micheli, G. “SUNMAP: A tool for automatic topology selection and generation for NoCs,” In Proceedings of the 41st IEEE/ACM Design Automation Conference (DAC’04), San Diego, CA, USA, 2004, pp. 914–919.

[41] Dumitriu, V., Khan, G. “Throughput-oriented NoC topology generation and analysis for high performance SoCs,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2009, 17(10), pp.1433–1446.

[42] Hu, J., Ogras, U., Marculescu, R. “System-level buffer allocation for application-specific networks-on-chip router design,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2006, 25(12), pp.2919–2933.

[43] Shen, W., Chao, C., Lien, Y., Wu, A. “A new binomial mapping and optimization algorithm for reduced-complexity mesh-based on-chip network,” In Proceedings of the First IEEE International Symposium on Networks-on-Chip (NOCS’07), Princeton, NJ, USA, 2007, pp. 317–322.

[44] Elmiligi, H., Morgan, A.A., El-Kharashi, M.W., Gebali, F. “Power-aware topology optimization for networks-on-chips,” In Proceedings of the 2008 IEEE International Symposium on Circuits and Systems (ISCAS’08), Seattle, WA, USA, 2008, pp. 360–363.

[45] Elmiligi, H., Morgan, A.A., El-Kharashi, M.W., Gebali, F. “A topology-based design methodology for networks-on-chip applications.” In Proceedings of the second International Design and Test Workshop (IDT’07), Cairo, Egypt, 2007, pp. 61–65.

[46] Elmiligi, H., Morgan, A.A., El-Kharashi, M.W., Gebali, F. “Performance analysis of networks-on-chip routers,” In Proceedings of the second International Design and Test Workshop (IDT’07), Cairo, Egypt, 2007, pp. 232–263.

[47] Elmiligi, H., Morgan, A.A., El-Kharashi, M.W., Gebali, F. “A delay-aware topology-based design for networks-on-chip applications,” In Proceedings of the fourth International Design and TestWorkshop (IDT’09), Riyadh, Saudi Arabia, 2009, pp.1-5.

[48] Elmiligi, H., Morgan, A.A., El-Kharashi, M.W., Gebali, F. “Power optimization for application-specific networks-on-chips: A topology-based approach,” Journal of Microprocessors and Microsystems, 2009, 33(5-6), pp.343–355.

[49] Kahng, A., Li, B., Peh, L.S., Samadi, K. “ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration,” In Proceedings of the 2009 IEEE/ACM Conference on Design, Automation and Test in Europe (DATE’09), Nice, France, 2009, pp. 423–428.

[50] Lee, K., Lee, S.J., Yoo, H.J. “Low-power network-on-chip for high-performance SoC design,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2006, 14(2), pp.148–160.

[51] Lee, K., Lee, S.J., Kim, D., Kim, K., Kim, G., Kim, J., et al. “Networks-on-chip and networks-in-package for high-performance SoC platforms,” In Proceedings of the first IEEE Asian Solid-State Circuits Conference (A-SSCC’05), Hsinchu, Taiwan, 2005, pp. 485–488.

[52] Agnarsson, G., Greenlaw, R. “Graph Theory: Modeling, Applications, and Algorithms,” Old Tappan, NJ, USA, Prentice Hall; 2006.

[53] Witzke, E.L., Frese, S.D. “Some limitations of adjacency matrices in computer network analysis,” ACM SIGCOMM Computer Communication Review, 1988, 18(5), pp.43–47.

[54] Dijkstra, E.W. “A note on two problems in connexion with graphs,” Numerische Mathematik, 1959, 1(1), pp.269–271.

[55] Bhat, S. “Energy models for networks-on-chip components,” Master’s thesis, Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands, 2005.

[56] Predictive Technology Model (PTM), Arizona State University, Available: http://ptm.asu.edu/; Last accessed: Sep. 2014.

[57] Pavlidis, V., Friedman, E. “3-D topologies for networks-on-chip,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2007, 15(10), pp.1081–1090.

[58] Srinivasan, K., Chatha, K. “A low complexity heuristic for design of custom network-on-chip architectures,” In Proceedings of the 2006 IEEE/ACM Conference on Design, Automation and Test in Europe (DATE’06), vol. 1, Munich, Germany, 2006, pp. 130–135.

[59] Coenen, M., Murali, S., Radulescu, A., Goossens, K., De Micheli, G. “A buffer-sizing algorithm for networks on chip using TDMA and credit-based end-to-end flow control,” In Proceedings of the Third IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS’06), Seoul, Korea, 2006, pp. 130–135.

[60] Benini, L., De Micheli, G. “Networks on chips: Technology and tools,” San Francisco, CA, USA, Morgan Kaufmann, 2006.

[61] Yoo, H.J., Lee, K., Kim, J. “Low-power NoC for high-performance SoC design,” San Francisco, CA, USA, CRC Press, 2008.

Ahmed A. Morgan received the Ph.D. degree in computer engineering from the University of Victoria, Victoria, BC, Canada, in 2011, and the B.Sc. degree (first class honors) and the M.Sc. degree in computer engineering from the faculty of Engineering at Shoubra, Benha University, Egypt in 2005 and 2000, respectively. He got a Diploma in Electronic Design Automation (EDA) and VLSI Design from the Information Technology Institute (ITI), Cairo, Egypt in 2002. His research interests include parallel architectures, multicore systems, digital VLSI design, and Networks-on-Chip (NoC) modeling, optimization, and performance evaluation.

Haytham Elmiligi received the Ph.D. degree in Electrical Engineering from the University of Victoria, Victoria, BC, Canada, in 2011. He is currently affiliated with the IMS research group at the University of Victoria and the Computing Science department at TRU. He is a senior IEEE member and a book series Co-editor with CRC press. His research work strongly relies on combining advanced analytical models and optimization techniques to improve the overall performance of multi-core systems.

M. Watheq El-Kharashi received the Ph.D. degree in computer engineering from the University of Victoria, Victoria, BC, Canada, in 2002, and the B.Sc. degree (first class honors) and the M.Sc. degree in computer engineering from Ain Shams University, Cairo, Egypt, in 1992 and 1996, respectively. He is a Professor in the Department of Computer and Systems Engineering, Ain Shams University, Cairo, Egypt and an Adjunct Associate Professor in the Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada. His general research interests are in advanced system architectures, especially Networks-on-Chip (NoC), Systems-on-Chip (SoC), and secure hardware. He published about 100 papers in refereed international journals and conferences and authored two books and 6 book chapters.

Fayez Gebali received his B.Sc. in Electrical Engineering (first class honors) from Cairo University, his B.Sc. in Mathematics (first class honors) from Ain Shams University, and his Ph.D. degree in Electrical Engineering form the University of British Columbia, where he was a holder of an NSERC post-graduate scholarship. He is a Professor and Chair of the Department of Electrical and Computer Engineering at the University of Victoria. His research interests include parallel algorithms, Networks-on-Chip, 3D-ICs, digital communications, and computer arithmetic.