Microprocessors and Microsystems 000 (2016) 1–14

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Power- and performance-efficient cluster-based network-on-chip with reconfigurable topology

Pooyan Mehrvarzy[a], Mehdi Modarressi[b,*], Hamid Sarbazi-Azad[a,c]

[a] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
[b] School of Electrical and Computer Engineering, College of engineering, University of Tehran, Theran, Iran
[c] Department of Computer engineering, Sharif University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 1 November 2015
Revised 5 February 2016
Accepted 3 March 2016
Available online xxx

Keywords:
Network-on-chip
Topology
Reconfiguration
Power
Performance
Mapping

ABSTRACT

Topology is widely known as the most important characteristic of networks-on-chip (NoC), since it highly affects overall network performance, cost, and power consumption. In this paper, we propose a reconfigurable architecture and design flow for NoCs on which a customized topology for any target application can be implemented. In this structure, the nodes are grouped into some clusters interconnected by a reconfigurable communication infrastructure. The nodes inside a cluster are connected by a mesh to benefit from the interesting characteristics of the mesh topology, i.e. regular structure and efficient handling of local traffic. A reconfigurable inter-cluster topology then eliminates the major shortcoming of the mesh by providing short paths between remotely located nodes. We then present a design flow that maps the frequently communicating tasks of a given application onto the same cluster and exploits the reconfigurable infrastructure to set up appropriate inter-cluster connections. The experimental results show that by efficiently handling local and long-distance traffic flows, this structure is scalable, and power- and performance-efficient.

1. Introduction

With the advances of semiconductor technology in recent years, current application-specific multi-core system-on-chips (SoCs) have rapidly grown in size and complexity. Future SoCs will consist of a large number of complex components communicating with each other at very high-speed rates. The microprocessor industry is also moving from single-core and multi-core to many-core architectures containing tens to hundreds of identical cores arranged as chip multiprocessors (CMPs) [1]. The lack of scalability in bus-based systems and large area overhead and unpredictability of electrical parameters of point-to-point dedicated links have motivated researchers to move toward packet-switched Network-on-Chip (NoC) architectures in order to overcome common on-chip communication problems [2].

Power consumption is considered as a major design concern in current and future digital systems. Since NoCs rely on routers to handle on-chip traffic, they impose some power overhead for packet buffering, routing, and arbitration. In [1], it has been projected that in future CMPs and Multicore SoCs, the power consumption of the NoC designed by the current methodologies will be about 10 times greater than the power budget that can be devoted to on-chip communication. Application-specific optimization, in which the NoC parameters are customized for a target application, is one of the most effective approaches to bridge the existing gap between the current and ideal NoC power/performance metrics in application-specific multi-core SoCs, where the application and its traffic characteristics are generally known at design time.

Since topology is one of the most important NoC attributes, a large body of research has been devoted to topology customization methods. Topology deals with the physical placement of network nodes and inter-node connections. When a topology is generated for a NoC, it will remain unchanged during the system lifetime and cannot be modified once the chip is fabricated. This is a problem in today's multi-core SoCs (in particular, programmable multi-core SoCs) into which several different applications (often unknown at design-time) are integrated. Since the inter-core communication characteristics can be very different across different applications, a topology that is designed based on the traffic pattern of one application does not necessarily meet the design constraints of some other application. In [3], over 1500 different NoC configurations (topology, buffer size, etc.) are investigated and it has been shown that no single NoC can be found to provide optimal performance across a range of applications.

The problem of dynamic topology adaption has been addressed in several previous works [3–6]. In our previous works [4,6], we

* Corresponding author. Tel.: +982161119756.
E-mail addresses: p.mehrvarzy@ipm.ir (P. Mehrvarzy), modarressi@ut.ac.ir (M. Modarressi), azad@sharif.edu (H. Sarbazi-Azad).

http://dx.doi.org/10.1016/j.micpro.2016.03.004
0141-9331/

proposed a reconfigurable NoC to dynamically change the network topology by changing the connectivity among network nodes. The reconfiguration of this architecture is achieved by inserting several simple switches in the network. These simple and programmable switches allow the network to dynamically change inter-node connections and implement a customized topology that best matches the communication pattern of the running application. For a fixed physical mapping, NoC tries to reduce the hop count (or number of routers) between the endpoint nodes of high volume communication flows by bypassing some intermediate routers via switches. This can lead to considerable performance improvement since the latency (power) of the router pipeline stages has a significant contribution to the total NoC latency (power). The experimental results show a considerable power and performance improvement over a conventional architecture when a new application is introduced to a multi-core SoC after synthesis.

We then extend the NoC by a more efficient reconfigurable structure [7]. The new structure is a cluster-based reconfigurable NoC that supports both short- and long-distance traffic flows by revising the placement strategy of the reconfigurable resources of the baseline reconfigurable NoC. To achieve this goal, we group the processors into some mesh clusters and use the reconfigurable infrastructure to connect the clusters in a hierarchical fashion. This hierarchy consists of several clusters with fixed topology at the first level and a reconfigurable topology at the second level. Consequently, this structure not only benefits from the advantages of hierarchical topologies (high scalability and efficient traffic management), but also can outperform the existing hierarchical structures, as it uses an adaptable topology for the higher level of the hierarchy.

In addition, this architecture realizes application-specific topologies over structured and regular components and benefits from both worlds: it can be designed, optimized, and reused like regular NoCs, while its connections can be dynamically reconfigured to adapt to the traffic pattern of the currently running application.

Early results show that the cluster-based reconfigurable NoC can potentially outperform the baseline. However, the main challenge in using the new cluster-based scheme is to develop a design flow to exploit its potential capabilities. This involves finding methods to allocate clusters to different parts of the workload, map the workload tasks to the cluster nodes, and configuring the inter-cluster switches to adapt the reconfigurable connections to the workload traffic pattern.

The proposed reconfigurable NoC helps to reduce average packet hop count for applications that are running on an unoptimized mapping. This may occur in two cases: first, when multiple applications with different traffic patterns are integrated on the same NoC and finding a mapping that is optimal for all applications is not possible. The second case occurs when a new application is developed for a system after design time, when the physical mapping is already done and each core has a fixed place in the topology. In the former case, our design flow finds a mapping based on the entire set of input applications (that is not necessarily optimal for each individual application) and then calculates a topology for each individual application. In the latter case, the mapping is fixed and the problem reduces to topology reconfiguration for the new application.

In our previous work, we focused more on exploring different architectural aspects of the NoC, but used a simple heuristic algorithm for design flow and reconfiguration. Although the algorithm works very fast, it has limited capabilities to explore optimal and near optimal solutions.

In particular, the most challenging part of the design flow is topology reconfiguration. We compare the quality of the solutions given by that mapping and topology reconfiguration algorithms with the optimal results obtained by exhaustive search method. Experiments are done under 10 random communication task graphs in a 6×6 mesh NoC and show that the results are 26% far from optimal (nonetheless, they improve the baseline NoC latency by 18–30%).

In this paper, we extend our previous work by developing a new design flow for the cluster-based reconfigurable NoCs in order to bridge the existing gap between the optimal and current solution qualities and find near-optimal topologies for each input application in an acceptable time. The topology reconfiguration algorithm, as the core of this design flow, can also be extended to be used in solving the routing problem in FPGAs.

The rest of the paper is organized as follows. Section 2 provides a short survey on related work. Section 3 overviews the baseline and cluster-based reconfigurable NoC architectures. Section 4 develops a design flow to map a set of target applications onto the proposed cluster-based architecture and find a suitable topology for them. Section 5 evaluates the proposed architecture and design flow, and finally, Section 6 concludes the paper.

2. Related work

As current multi-core SoCs have been rapidly growing in size and complexity, many studies have focused on the necessity of scalable on-chip communication architectures [7–8] and various optimization methods for NoCs are proposed to reduce the message latency and power consumption.

Most NoC proposals (including all commercial NoC implementations) adopt regular topologies that can be laid out on a 2-dimensional plane and can offer an average performance for a large set of applications [23–25]. Despite the benefits of meshes for circuit-level implementation, some packets may suffer from long latencies due to the lack of short paths between remotely located nodes.

The best way to decrease NoC power and latency is to decrease the average number of hops messages taken in the network. This is mainly achieved by using low-diameter networks including high-radix networks [9,10] or equipping mesh topologies with extra irregular links [11] that are inserted randomly to shorten diameter. NoC-Out is a hybrid high-radix interconnection scheme for CMPs used in server applications and benefits from both high-radix flattened butterfly topology to reduce hop count and low latency simple routers to reduce per-hop latency [12]. In NoC-Out CMP, LLC slices are connected by the flattened butterfly. Each flattened butterfly router, in addition to its connections to LLC, is also connected to a cluster of cores connected by simple and fast routers.

Single-cycle multi-hop traversal proposed by SMART NoC (Single-cycle Multi-hop Asynchronous Repeated Traversal) [13] reduces the hop count virtually. SMART implement bypass paths on regular NoC links and crossbars without adding any physical channels. A multi-hop path that bypasses several intermediate routers is set up in a per cycle basis by a control network; setting up bypass paths involves adding one cycle to the router pipeline stages. Leveraging the waiting time of packets to proactively set up multi-hop paths, our method eliminates the extra clock cycle needed by SMART and can potentially achieve better performance. Efficient routing algorithms can also reduce NoC latency by directing packets to less congested paths, thereby reducing packet blocking latency. Among them, those methods that leverage both local and global congestion metrics [14,15] can make proper routing decisions.

Hop count reduction can also be done in networks with known application traffic patterns, by application-specific mapping, routing, and topology selection mechanisms. The communication traffic characteristics of multi-core SoCs used in embedded systems are non-uniform and can usually be obtained statically. As a

Please cite this article as: P. Mehrvarzy et al., Power- and performance-efficient cluster-based network-on-chip with reconfigurable topology, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.03.004