Computers and Electrical Engineering 51 (2016) 195–206

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Reconfigurable fault tolerant routing for networks-on-chip with logical hierarchy☆

Gert Schley[a], Ibrahim Ahmed[a,b], Muhammad Afzal[a,c], Martin Radetzki[a,*]

[a] Embedded Systems Group, University of Stuttgart, Pfaffenwaldring 5b, Stuttgart 70569, Germany
[b] Dept. of Electrical and Computer Engineering, University of Toronto, 10 King's College Road, Toronto, ON M5S 3G4, Canada
[c] Altium Europe GmbH, Philipp-Reis-Strasse 3, Karlsruhe 76137, Germany

ARTICLE INFO

Article history:
Received 29 April 2015
Revised 16 February 2016
Accepted 16 February 2016
Available online 7 March 2016

Keywords:
Networks-on-chip
Hierarchy
Routing
Fault tolerance
Reconfiguration

ABSTRACT

This paper presents a reconfigurable fault tolerant routing for Networks-on-Chip organized into hierarchical units. In case of link faults or failure of switches, the proposed approach enables the online adaptation of routing locally within each unit while deadlock freedom is globally ensured in the network. Experimental results of our approach for a 16 × 16 network show a speedup by a factor of almost four for routing reconfiguration compared to the state-of-the-art approach. Evaluation with transient faults shows that a dedicated reconfiguration unit enables successful reconfiguration of routing tables even in case of high error probabilities.

1. Introduction

The ongoing technology scaling allows an increasing number of cores to be implemented on a single chip, e.g. Intel's Xeon Phi Coprocessor [1] or Tilera's Tile-MX multicore processor [2]. As this scaling trend continues [3], future multiprocessor systems will feature hundreds of cores on a single chip.

The increasing size of on-chip systems poses a new challenge to Networks-on-Chip (NoCs). Mechanisms implemented in an NoC, such as table-based routing, work well for small systems but do not scale for bigger systems. A possibility to cope with scalability problems is the introduction of a hierarchical structure to NoCs. A hierarchical NoC can be obtained by constructing its topology using subnetworks or by segmenting a given topology into logical units. Both subnetworks and logical units enable formerly global mechanisms to be applied locally thus reducing their complexity. Compared to subnetworks, logical units have the advantage that they can be applied without changing an existing topology. Typically, network nodes (switch + core) are grouped into a logical unit if they are part of the same task and share a spatial relation.

The downside of technology scaling is the increased probability of occurrence of permanent faults in an NoC due to manufacturing inaccuracies [4] or wear-out effects such as electromigration [5] that emerge during system operation. The failure of links or switches due to permanent faults results in an altered network topology. In such a case, static routing can no longer maintain connectivity between system components. For this reason, it is crucial that the routing is adapted to the new network situation to enable packets to circumvent faulty components.

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. M. Ebrahimi.

* Corresponding author. Tel.: +4971168588270.
E-mail addresses: gert.schley@informatik.uni-stuttgart.de (G. Schley), ibrahimai.ahmed@mail.utoronto.ca (I. Ahmed), muhammad.afzal@outlook.de (M. Afzal), martin.radetzki@informatik.uni-stuttgart.de (M. Radetzki).

http://dx.doi.org/10.1016/j.compeleceng.2016.02.013
0045-7906/

In this paper we present a reconfigurable fault tolerant routing approach based on *Up/Down* routing [6] for large scale NoCs with logical hierarchy. It enables the routing to be adapted locally within each hierarchical unit in case of permanent faults while deadlock freedom is guaranteed globally. Our approach can be applied to any number of hierarchy levels.

The remainder of the paper is organized as follows: In Section 2 related work is discussed. Section 3 contains a formal introduction to NoC topologies as well as an introduction to *Up/Down* routing. In Section 4 we present our hierarchical network concept. Our hierarchical routing is presented in Section 5 and the reconfiguration process in Section 6. Evaluation results are discussed in Section 7. Section 8 concludes the paper.

2. Related work

In this section, we focus on work related to routing reconfiguration. Related work dealing with hierarchical topologies and hierarchical routing is presented in [7].

A reconfigurable scheme for source based routing is presented in [8]. If a source cannot reach a destination, it floods a *path request* through the network. Each node stores the port via which the request was received in a table. When the request reaches the destination, a packet is sent back to the source using the reverse path recording that path by means of the table entries. At the source the recorded path is checked in software if it contains violations of routing restrictions and if so *virtual channels* (VCs) are assigned to prevent deadlocks. Results in [8] show that about 2300 cycles are required to adapt a path of length 10 hops. However, permanent faults usually have an impact on multiple source-destination pairs and thus the path request has to be initiated for each of them. This results in a high time to adapt the routing completely. Furthermore, source based routing induces great costs for implementation of routing tables.

In [9], a centralized routing management system for high-performance networks is proposed. After the occurrence of a fault, the central manager first discovers the remaining topology and creates a topology graph with *Up/Down* directions. This graph then is used to calculate the routing for every network node. However, determining the routing for all nodes by a single manager results in a high calculation time.

A distributed routing recalculation approach for *Up/Down* routing is presented in [10]. For routing adaptation, normal operation in network is stopped and flags are broadcast by each node. *Up/Down* routing directions are assigned to each node port during the initial broadcast. Starting with the node that has detected a permanent fault, each of the n network nodes consecutively broadcasts a flag to all other nodes via dedicated signals. The reconfiguration period is divided into n portions, each with a duration of n cycles. Upon receipt of a flag, a node records the port over which the flag was received and updates its routing table accordingly. In both approaches, [10] and [9], the routing of the entire network has to be adapted. Neither [10,9], nor [8] consider hierarchical network topologies.

A fast online routing reconfiguration algorithm based on *segment-based routing* [11] to compute emergency routes in case of a fault is presented in [12]. The approach allows the adaptation of routing in segments of the network while other segments are not affected. To compute emergency routes the algorithm makes use of routing meta data calculated offline during initial routing computation. Per segment one fault can be tolerated, however, in case of a second fault, the routing and meta data for the whole network has to be recalculated offline in software.

Furthermore, approaches exist that do not require a reconfiguration phase but achieve routing adaptation by means of additional hardware logic, e.g. [13] and [14]. While [13] can tolerate the failure of three switches, [14] provides connectivity for any irregular topology derived from a non-hierarchical 2D mesh. A disadvantage of these approaches is that the hardware logic is prone to faults as well. A permanent fault in this logic leads to a routing failure. In our approach, routing is adapted in software. In general, software-based calculating provides the possibility that in case of a network node failure the calculation task can be migrated to another network node. As in case of [14], our routing provides connectivity as long as the topology is not disconnected by faults.

3. Preliminaries

3.1. NoC topology

The topology of an NoC can be represented by a directed graph $T = (N, C)$ where N is the set of network nodes and C is the set of unidirectional channels. In a typical fault free NoC topology, two connected nodes n_i, $n_j \in N$ have one bidirectional connection, i.e. $c_{i,j}$, $c_{j,i} \in C$. We refer to bidirectional connections as *links* and unidirectional connections are called *channels*. To distinguish between different nodes, each node has a unique ID.

3.2. Up/Down routing

In literature, various approaches (e.g. [10,15,16]) based on *Up/Down* routing can be found. *Up/Down* routing, as originally described in [6], ensures an inherently deadlock free routing for arbitrary network topologies.

For *Up/Down* routing calculation for a given topology T, the spanning tree S of T is determined starting from a root node. Based on spanning tree S, *Up* or *Down* direction is assigned to each channel of C. *Up* direction is assigned to channel $c_{i,j}$ if n_j has a shorter distance to the root node than n_i. If two nodes have the same distance to the root node, then *Up* direction is assigned to the channel leading to the node with the smaller ID. In all other cases, *Down* direction is assigned. To ensure