Technical Report CE-TR-2008-01

Issued: 02/2008

Comparison of custom topology networks against rigid interconnects

Radu Stefan
Ioannis Sourdis
Georgi Gaydadjiev
Kees Goossens

SARC Internal

Concerns: Internal Report

Period of Work: 2007

Author's address
R. A. Stefan radu124@ce.et.tudelft.nl
I. Sourdis sourdis@ce.et.tudelft.nl
G. Gaydadjiev g.n.gaydadjiev@ewi.tudelft.nl
K. Gossens kees.goossens@nxp.com

Title:	Comparison of custom topology networks against rigid interconnects
Author(s):	Radu Stefan; Ioannis Sourdis; Georgi Gaydadjiev; Kees Goossens
Technical Report:	CE-TR-2008-01
Project:	SARC www.sarc-ip.org

Keywords: topology, reconfiguration

Abstract: Recent NoC studies concern the optimization of interconnection networks by generating custom topologies, adapted to the requirements of specific applications. This approach contrasts with the traditional one, that of mapping the nodes to routers connected in standard, well-studied topology. In this study, we have reimplemented and extended an early work in topology generation, by allowing multiple nodes to be connected to the same router. The topologies generated in this way can be used either statically at design time, but we also envision a system where they could be used at run-time, by reconfiguring the interconnection network. In parallel with this study we are also evaluating the cost of reconfiguration.

Contents

1 Introduction **2**

2 Related Work **3**

3 Topology generation **5**

4 Overheads introduced by reconfigurable hardware **16**

5 Conclusions **21**

Chapter 1

Introduction

The traditional approach to communication network optimization in computation systems with multiple processing cores consist of the careful mapping of the processing nodes to routers connected in a fixed topology. Numerous studies debated which topology is more appropriated, and proposed methods of performing this mapping. In general, the mapping is performed at design-time, but re-mapping at run-time is also possible, either by migrating tasks to their desired position in a fixed network, or by modifying the network links in such a way that nodes appear to be connected to the desired router, while the network topology remains the same [5]. A more radical approach consists of generating not only a mapping but a fully custom topology according to the communication needs.

Proponents of standard network topologies point out that the regularity of meshes and tori represents a great advantage, however in practice this may not always be true. Often, the floorplanning is performed before the connecting network is generated and the cores may not be distributed in a regular fashion. Furthermore, mapping the nodes to routers may result in distant cores being mapped to neighboring routers which will result in long links.

If reconfigurable networks are to be considered, it is certainly desirable that the packets travel the largest part of the distance between two nodes using only "blind" switches that do not perform arbitration, or even pass transistors used in FPGAs. In that way, only a small number of packet switching routers that perform buffering have to be traversed. Taking this into consideration, we attempt to generate topologies that are not necessarily regular but which aim to optimize communication parameters of the network like the number of hops, and implicitly the link utilization.

Chapter 2

Related Work

The work in automatic topology generation was pioneered by Smitley and Lee in [**?**, **?**]. Their study is highly theoretic and focuses on optimizing graph parameters like average path length and count of paths of unit length. We use their study as a basis for our current research. Other approaches, targetting large scale telecomunication networks focused on hamiltonian path approximations[1]. Performance and drawbacks of standard topologies were widely studied.

The emergence of NoC was accompanied by a resugence of topology optimization studies [11, 8, 13, 4, 2, 10, 3]. It shall be noted that topology selection or generation is not the only optimization step performed, but it is used in conjunction with mapping, link width selection and other network parameter optimizations.

In [10] the authors propose running a mapping and physical planning flow on multiple topologies selected from a topology library. The solution is further refined by buffer sizing and removing unused edges.

[13] uses linear programming and randomization algorithms to perform the node placement and interconnect optimization formulated as a Steiner arborescence problem. Latency and bandwidth targets are considered, and the flow performs operations from floorplanning to routing. Different core sizes are considered.

[11] proposes a method named k-way merging with two algorithm implementations: divisive clustering and agglomerative clustering. Their approach relies on an economy of scale assumption, which justifies merging multiple communication streams over the same physical communication line.

In [8] the authors propose conflict-free networks optimized for the given application when the traffic is well-known in advance. The study targets permutation and partial permutation traffic patterns. The technique used is recursive bisection. The algorithm starts from a large switch connecting all nodes and dividing it in such a way that communication requirements between nodes is minimized. Simulated annealing and graph coloring are used.

[4, 2] propose flows that include topology optimizations without giving much information about how this optimization is to be achieved. [2] focuses on ring networks and network partitioning.

[3] considers a joint approach for mapping and interconnect synthesis. The author's proposed solution produces a scheduling of tasks on the cores so that a maximum distance between tasks which communicate is obeyed. The topologies are generated using genetic algorithms and the design space is reduced by taking into account graph isomorphism.

Chapter 3

Topology generation

We have based our work on the early studies of Smitley and Lee [Synthesizing Minimum Total Expansion Topologies for Reconfigurable Interconnection Networks,1987]. The original algorithm consists of the following steps:

- the network connections graph is built by applying the Degree Constrained Subgraph algorithm to the Communications Graph;
- if the resulting graph is not connected, edges are replaced in order to connect it;
- a hill-climbing technique is used in order to further refine the result.

We have developed our own implementation for all the mentioned algorithms. In addition to the original algorithm, we evaluate the possibility of connecting two network interfaces to each router. As an effect the number of routers in the network is reduced by half, however the chance of the links between routers becoming saturated increases.

For deciding what nodes should be connected together to the same router we use the maximum matching in general graphs algorithm, which deterministically produces optimal results in terms of maximizing the local traffic between nodes. In addition, within the set of results which obey the previous optimality criterion, we attempt to find a solution which groups nodes that communicate with the same neighbors.

We have performed the tests using synthetic traffic streams in networks of different sizes, with 20,40,60 and 80 nodes, each of them performing read operations (read operations consist of request and replies) from four other random nodes in a reciprocal manner (if A reads from B, then B reads from A).

In order to avoid the problem of deadlock-free routing we have used the GT traffic mode of the AEthereal networks, which preallocates the communication channels. The GT mode represents a form of reconfiguration by using Time-Division-Multiplexing to create on-demand routes. In consequence, the interconnect synthesis algorithm has the role of generating the underlying network that supports the reconfiguration in this case. Some preliminary results were also obtained

using the BE mode (Best Effort traffic), which is in essence Wormhole routing. Wormhole routing will likely be used in the future for Networks-on-Chip, because of its low buffering requirements, but the preallocation strategy may also be very useful when the communication requirements are known.

Power results and BE bandwidth were obtained through SystemC simulation, while area results and slot count were determined analytically.

Evaluation methodology

One random traffic pattern is generated for each of the node counts. For each of the traffic patterns, a custom topology is generated together with a mapping of nodes to routers. The generated topologies are tested against mesh and torus topologies with size twice as large and against torus with the same size (two network interfacesrouter). For the 20 and 80 nodes communication patterns, minimum diameter graph topologies, with two NIs/router were also used. The generated topologies were tested both with the generated mapping, and with the automatic mapping performed by the tools, while for the other topologies only automatic mapping was used.

For each of the evaluated topologies, multiple tests were performed, scaling linearly the required bandwidth of each stream while maintaining the link bandwidth fixed, until the tool set failed to produce a successful allocation of all streams. The AEthereal tool-flow was used for each of the tests as follows:

- mapping (where necessary), slot allocation and routing using the UMARS algorithm [7], were performed; for BE traffic, a deadlock free routing is also generated at this stage;
- if the allocation failed because of too high bandwidth requirements, the test was interrupted;
- the slot allocation is verified and necessary buffer sizes are analytically determined;
- area reports are generated;
- a SystemC simulation is run for the produced configuration;
- bandwidth, latency and power reports are generated from the SystemC simulation.

The maximum bandwidth at which a successful allocation could be performed for each of the topologies is considered as primary result (fig. 3.1), however, statistics are gathered for the lower bandwidths as well. The area used by the routers in each configuration is presented in figure 3.2.

The following graphs present detailed results regarding the variation of network parameters with load. On the horizontal axis, the requested bandwidth in units of 100Mbps is represented. In figures 3.3-3.10 the delivered traffic was equal to the requested traffic as the GT (guaranteed throughput) mode was used. The values measured have the following meaning:

- number of TDM slots used, figures 3.3 and 3.4;

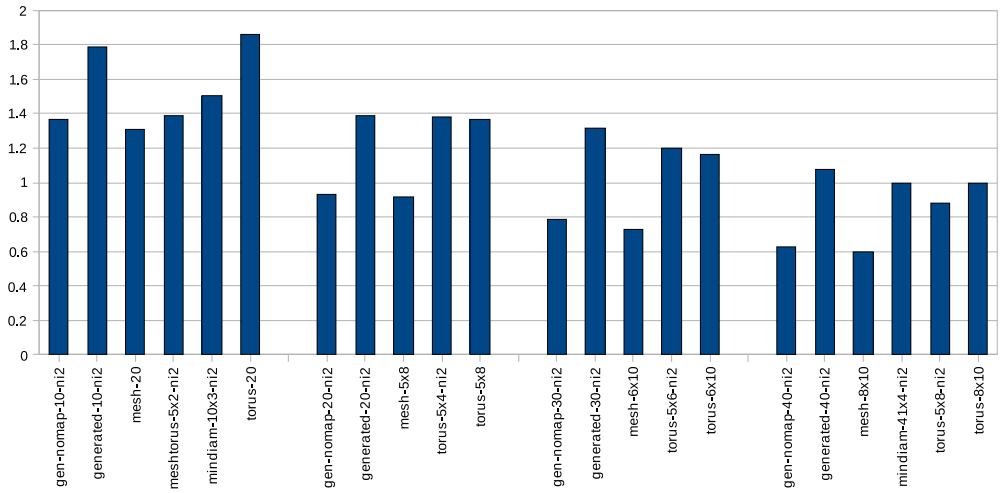

Figure 3.1: normalized bandwidth results, 1 represents a 100Mbps traffic for each independent stream

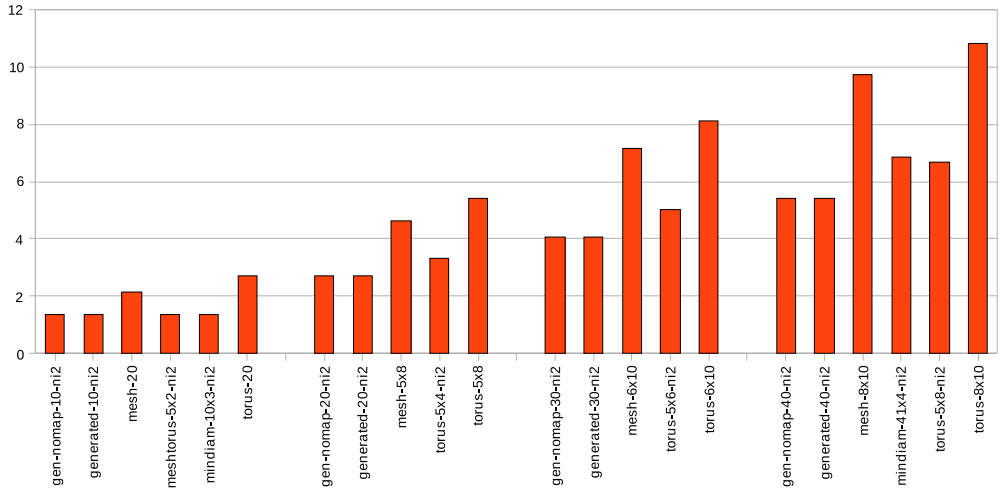

Figure 3.2: network area (routers only, not including network interfaces)

- area including NI buffers that have been sized according to requirements, figures 3.5,3.6,3.7 and **??**;
- average number of hops, figure 3.9;
- power, figure 3.10;
- delivered BE traffic, figures 3.11 and 3.12;

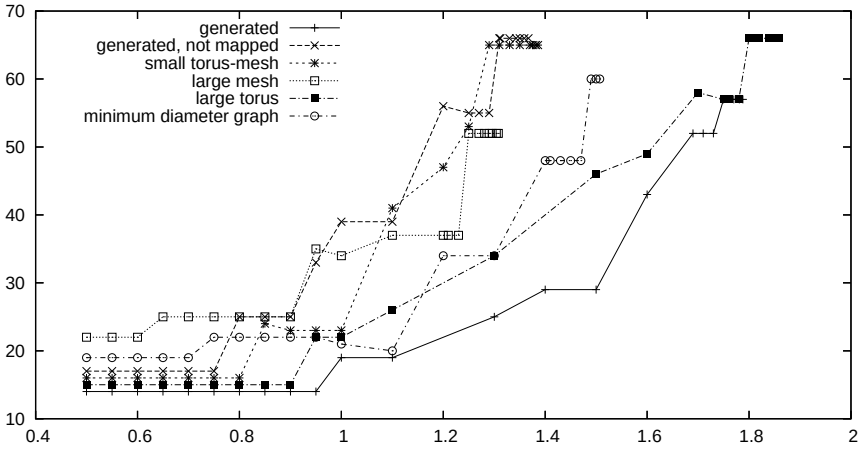

Figure 3.3: number of TDM timeslots vs. requested traffic in 100Mbps units, 20 nodes

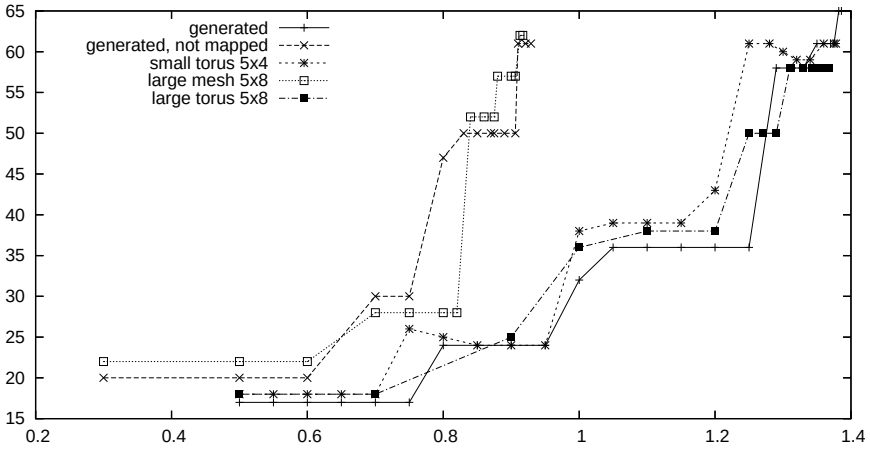

Figure 3.4: number of TDM timeslots vs. requested traffic in 100Mbps units, 40 nodes

The slotcount represents the number of slots used for TDM by AEthereal, and a smaller value is desirable, as it decreases the size of the slot tables and it reduces the wait-delay for transmission. Our generated topologies allow for a better slot allocation and thus a smaller number of slots is necessary, as seen in figures 3.3 and 3.4. Not all graphs were included for reasons of space, but the trends remain the same throughout the entire test set. Some of the power and BE simulation results are missing because of tool limitations.

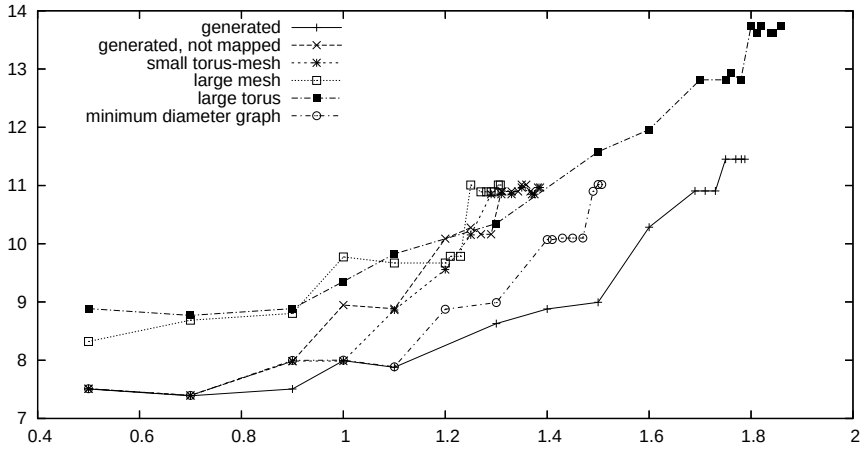

Figure 3.5: area including NI buffers vs. requested traffic in 100Mbps units, 20 nodes

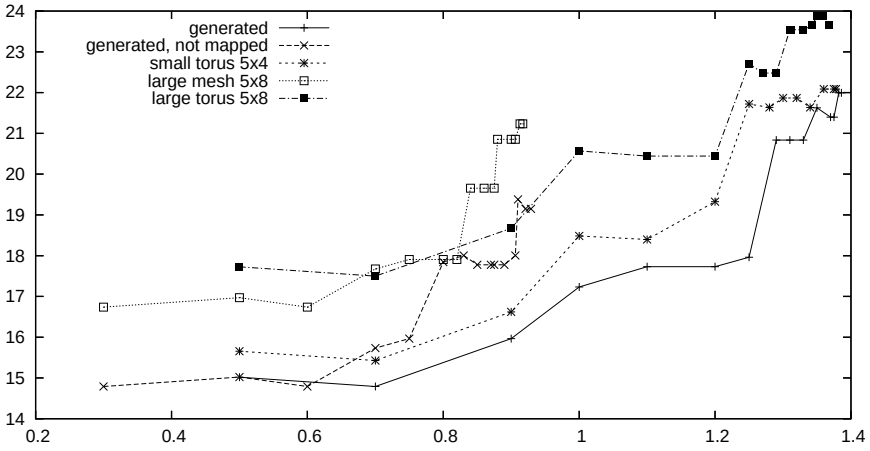

Figure 3.6: area including NI buffers vs. requested traffic in 100Mbps units, 40 nodes

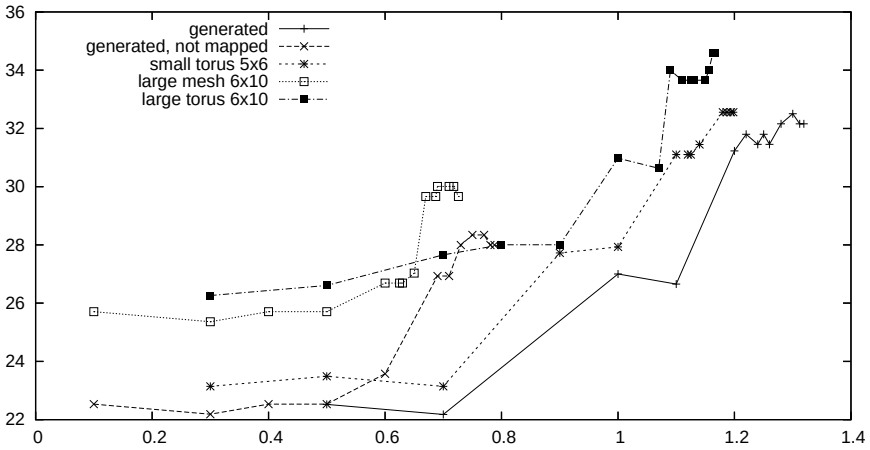

Figure 3.7: area including NI buffers vs. requested traffic in 100Mbps units, 60 nodes

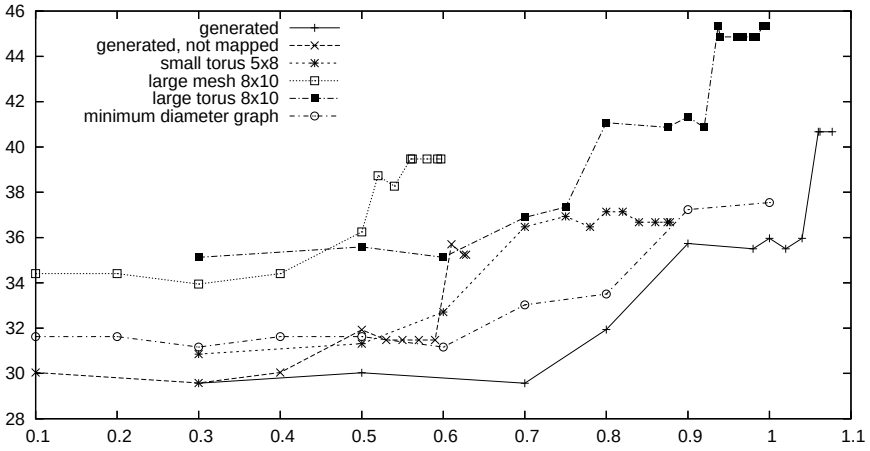

Figure 3.8: area including NI buffers vs. requested traffic in 100Mbps units, 80 nodes

The detailed area reports (fig. 3.5-3.8), present total area of the network, including NIs and NI buffers. The area is dependent on load because the buffers are sized according to traffic requirements. The buffer sizes are calculated analytically based on the round-trip delay and bandwidth overprovisioning.

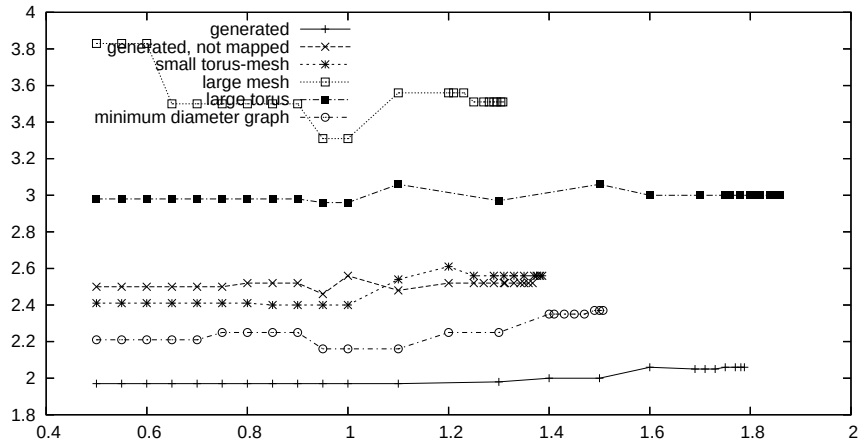

Figure 3.9: average number of hops vs. requested traffic in 100Mbps units, 20 nodes

A large improvement is obtained in the average number of hops (fig. 3.9), which is the target of the Smitley&Lee algorithm [12]. The number of hops is important in reducing the active power consumption as packets generate switching activity in every router they pass through.

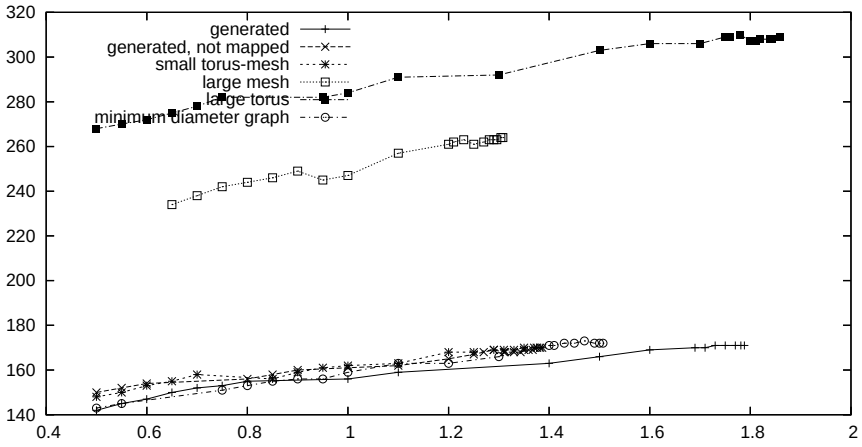

Figure 3.10: power consumption mW, vs. requested traffic in 100Mbps units, 20 nodes

Figure 3.10 presents power consumption versus requested/delivered bandwidth. A large portion of the total power is represented by the static power consumption, and thus the large networks are heavily penalized by the additional hardware used.

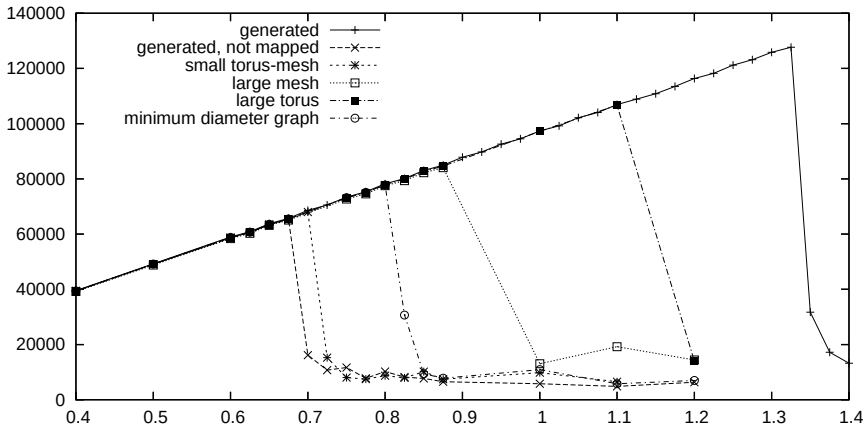

Figure 3.11: delivered vs. requested traffic in Mbps, 20 nodes, in 100Mbps units

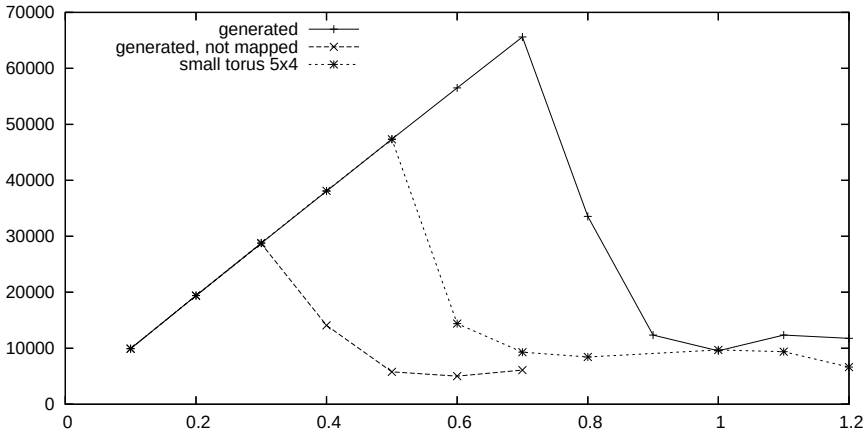

Figure 3.12: delivered vs. requested traffic, 40 nodes, in 100Mbps units

For the BE traffic, the delivered bandwidth is plotted against the requested bandwidth (fig. 3.11,3.12). It is observed that overloading the network results in a sharp decrease in performance. The generated topologies have the highest saturation points, however, if mapping is not performed at the same time with topology generation, the default mapping produced by the tool performs worst among the test-cases.

Chapter 4

Overheads introduced by reconfigurable hardware

Figure 4.1 presents side-by-side a node belonging to the static implementation of a network and the reconfigurable version. The later one consists of a router having each of its connections except for the local one redirected into a reconfigurable mesh. We will use this configuration as a baseline for cost estimation.

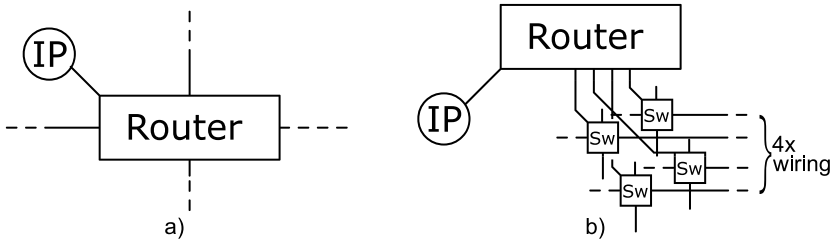

Figure 4.1: Fixed network (a) vs reconfigurable one (b)

Less costly alternatives are represented in figure 4.2 where a smaller reconfigurable mesh is interposed between the processing nodes and the routers, and in figure 4.3 where the topology is in part fixed, but shortcuts between routers may be created across the reconfigurable paths. The first one can be used to achieve the reconfiguration without modifications to the topology as described in [**?**], while the second represents a compromise solution between the degree of reconfigurability offered and the resources used.

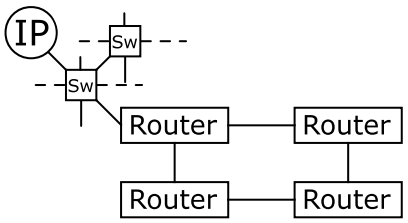

Figure 4.2: Reconfigurable network that allows remapping

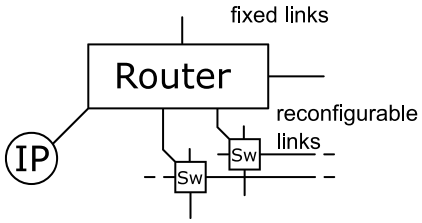

Figure 4.3: Partially reconfigurable topology

Area

The overhead of the proposed architecture in terms of area is highly dependent on the area ratio between the full-size routers and the switch-boxes used for reconfiguration. In order to determine this area ratio we have synthesized both the full router and a switch containing a crossbar, reverse links for transmitting the credits and configuration logic, in the 65nm technology from TSMC, using Synopsys. There are two types of switches, one that buffers its input for one clock cycle and one that does not. For cost evaluation we have considered a configuration in which half of the switches are buffered and half are not, distributed in a checkerboard pattern. The results are presented in figure 4.4 and table 4.1, normalized and side-by-side with other results from the literature. The GT routers are included here as they are similar to the switches we propose for reconfiguration in that they provide no queuing or arbitration. GT routers are also used for providing virtual wires, but using TDM rather than spatial division, and have the advantage of being able to run at twice the speed of a BE router.

Circuit	Area relative to router	Area (mm^2)
Router Area (Synopsys)	100.00%	0.0269
Buffered Switch (syn)	8.91%	0.0024
Unbuffered Switch (syn)	4.46%	0.0012
AEthereal BE	100.00%	0.1500
AEthereal GT	22.00%	0.0250
Intel Router	100.00%	n/a
Intel Crossbar	26.50%	n/a

Table 4.1: Cost of switches used in reconfiguration compared to the cost of routers

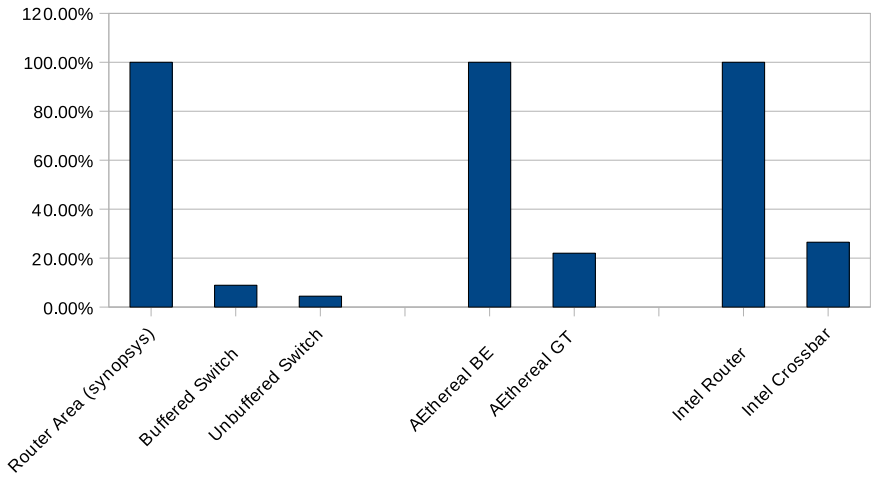

Figure 4.4: Cost of switches used in reconfiguration compared to the cost of routers

In the following discussion, for simplicity, we will denote buffered routers performing packet switching as routers, while the simple crossbars with at most one buffer representing a pipeline level will be called switches.

A significant overhead is expected in terms of wires. As suggested in figure 4.1, the baseline solution would require four times the amount of wires of the non-reconfigurable network. Although this number may seem large it shall be noted that topologies like multi-dimensional meshes use an equivalent number of wires (fig. 4.5). Obviously, mapping such a network on top of the reconfigurable network would require at least the same amount of wires. Further study will be needed to determine efficient underlying topologies and overprovisioning. An alternative would be that of using TDM to allocate only fractions of channels, as it is done by the AEthereal network.

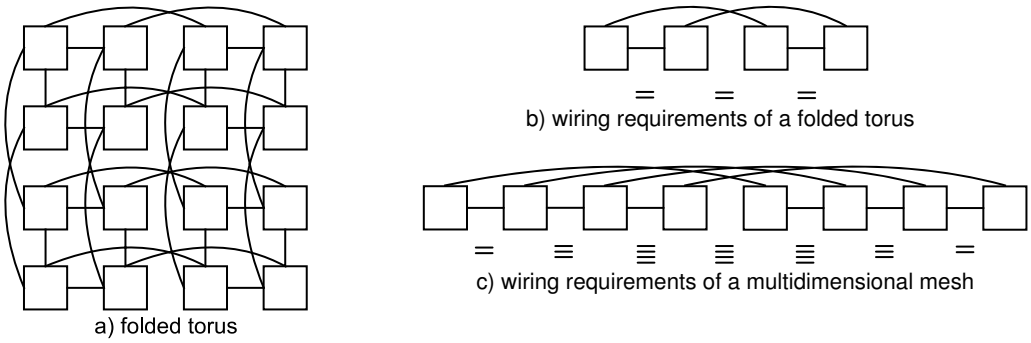

Figure 4.5: Wire cost of different topologies

Power estimation

The reconfiguration technique is especially efficient when a large part of the traffic can be delivered over one single hop (which may be represented by a virtual path through the reconfigurable hardware). In many traffic patterns which are known to be problematic for standard mesh networks, for example matrix transpose, nodes communicate with one single other node. In this highly desirable case, a packet sent on a mesh network, between nodes found at a distance n, has to traverse n links, 2 routers and $n+1$ switches, while in the original network $n+1$ routers and the same n links would have to be traversed (fig. 4.6).

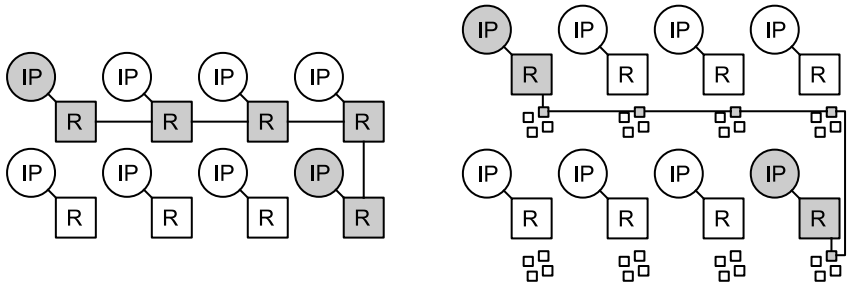

Figure 4.6: Routing in static versus reconfigurable networks

On a 40-node mesh network, with a random communication pattern, the average distance traveled is 4.35. For power estimation purposes, we are interested in the ratio between the power consumed by different components. We consider a passing packet generates switching activity in every component it traverses. We denote P_c, the power drawn by a packet traversing a crossbar, P_l the power for traversing a link and P_b the power of traversing input queues of a router. Power drawn by pipeline stages can be included in these measures.

As an estimation for the power consumed by a packet traversing the network

we will thus have:
$P_{staticnet} = 4.35P_l + 5.35P_b + 5.35P_c$
and
$P_{reconf} = 4.35P_l + 2P_b + 7.35P_c$

Experiments done with AEthereal flow, indicate that the energy used by the GT traffic is 69.2% of the energy used by BE traffic over the same distance between nodes. The 30.8% estimation for energy consumed by the extra buffers is also in line with power estimations Orion power simulator [6]. For the power consumed by links we use a value of 20% based on the estimations of the same study, although this number may be higher, especially for the newer technologies. As only the ratio of the power numbers is relevant we use these normalized results of $P_l = 0.20$, $P_b = 0.308$, $P_c = 0.492$ to calculate $P_{staticnet} = 5.15$ and $P_{reconf} = 5.47$ which corresponds to an increase of 6.2%. We believe these values to be an overestimate in the detriment of the reconfigurable solution and we intend to obtain a more accurate estimation from the physical synthesis of the circuits. The test for estimating the energy in AEthereal was also done with a single stream, in the absence of contention.

Reconfiguration time

For reconfiguration we have considered a simple, centralized solution consisting of linking the entire configuration memory into a shift register or multiple parallel shift registers in a daisy-chain manner. This mechanism was implemented in the synthesized switches. Partial reconfiguration can also be achieved using a similar system, but with the addition of an additional set of shadow configuration registers, or with a mechanism similar to JTAG. The duration of a reconfiguration is much lower than in the case of the FPGAs, as here a single word configures an entire link, while inside an FPGA individual wires are configured separately. Assuming a speed of reconfiguration of one configuration word per clock cycle, the entire network can be reconfigured in $nOfRouters \times nOfSwitchesPerRouter \times nOfPortsPerSwitch$ cycles, which is in the order of hundreds of cycles for a network size of the order of tens of routers.

This evaluation does not account for further improvements and optimizations that can be applied to the proposed architecture.

Chapter 5

Conclusions

We found topology generation to be a promising solution for network optimization in embedded systems when communication characteristics are known beforehand. It is still a question of ongoing research if the benefit offered by this solution can outweigh the cost of reconfiguring the network so that the topology can be chosen dynamically at run-time. We are currently exploring ways of reducing this overhead as well as improving our topology generation algorithms.

The study thus far has been performed using synthetic traffic patterns. For future research we will consider more realistic traffic patterns.

We have also considered a baseline solution for network reconfiguration in order to perform cost estimation. The overheads introduced by the reconfigurable network may be summarized as follows:

- An increase in wiring of approximately 4x, as suggested in fig. 4.1; this number depends on the degree of reconfigurability that is desired, and is the subject of further study.
- An increase in area of 26-44%. The lower value has been obtained considering a number of 4 switches of the type used in the Synopsys synthesis, while the higher value corresponds to 2 GT routers in AEthereal technology.
- No decrease in link bandwidth is expected, as the links can be pipelined to any extent desired
- There will be an increase in router to router latency, however, this effect will be counteracted by a decrease in the number of routers traversed. The delay of router traversal is usually in the range of 2-5 cycles, high throughput routers like the one in Intel's 80 core chip [9] because of aggressive pipelining present delays of 5 cycles or more. In contrast, the delay of a preconfigured switch may be of 0 or 1 cycles.
- We expect an increase in static power consumption, which has yet to be quantified. Active power however may be increased or decreased, depending on the traffic patterns.

- The reconfiguration time will probably be limited by the time taken to drain all the in-flight packets out of the network. The actual time of reconfiguring the switches would be in the order of the hundreds of cycles.

This evaluation does not account for further improvements and optimizations that can be applied to the proposed architecture. For example the baseline implementation assumes a full width link is generated for each point-to-point connection, while in practice this will probably not be necessary. In addition it may also be possible to merge the crossbars used for reconfiguration with the router crossbars thus reducing some of the overhead.

Bibliography

[1] D. Ageyev. Hierarchical telecommunication systems topology synthesis.

[2] T. Ahonen, D. A. Sigüenza-Tortosa, H. Bin, and J. Nurmi. Topology optimization for application-specific networks-on-chip. In *SLIP '04: Proceedings of the 2004 international workshop on System level interconnect prediction*, pages 53–60, New York, NY, USA, 2004. ACM Press.

[3] N. Bambha and S. Bhattacharyya. Joint application mapping/interconnect synthesis techniques for embedded chip-scale multiprocessors. *Parallel and Distributed Systems, IEEE Transactions on*, 16(2):99–112, Feb 2005.

[4] L. Benini. Application specific noc design. In *DATE '06: Proceedings of the conference on Design, automation and test in Europe*, pages 491–495, 3001 Leuven, Belgium, Belgium, 2006. European Design and Automation Association.

[5] J. Garcia, J.M.; Duato. An algorithm for dynamic reconfiguration of a multicomputer network. *Parallel and Distributed Processing, 1991. Proceedings of the Third IEEE Symposium on*, pages 848–855, 2-5 Dec 1991.

[6] S. Hang-Sheng Wang; Xinping Zhu; Li-Shiuan Peh; Malik. Orion: a power-performance simulator for interconnection networks. *Microarchitecture, 2002. (MICRO-35). Proceedings. 35th Annual IEEE/ACM International Symposium on*, pages 294–305, 2002.

[7] A. Hansson, K. Goossens, and A. Rădulescu. A unified approach to constrained mapping and routing on network-on-chip architectures. In *CODES+ISSS '05: Proceedings of the 3rd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, pages 75–80, New York, NY, USA, 2005. ACM.

[8] W. H. Ho and T. Pinkston. A methodology for designing efficient on-chip interconnects on well-behaved communication patterns. In *High-Performance Computer Architecture, 2003. HPCA-9 2003. Proceedings. The Ninth International Symposium on*, pages 377–388, 8-12 Feb. 2003.

[9] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar. A 5-ghz mesh interconnect for a teraflops processor. *IEEE Micro*, 27(5):51–61, Sept.-Oct. 2007.

[10] S. Murali, L. Benini, and G. de Micheli. Mapping and physical planning of networks-on-chip architectures with quality-of-service guarantees. In *Design Automation Conference, 2005. Proceedings of the ASP-DAC 2005. Asia and South Pacific*, volume 1, pages 27–32Vol.1, 18-21 Jan. 2005.

[11] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli. Efficient synthesis of networks on chip. 2003.

[12] D. Smitley and I. Lee. Synthesizing minimum total expansion topologies for reconfigurable interconnection networks. 1987.

[13] K. Srinivasan, K. S. Chatha, and G. Konjevod. An automated technique for topology and route generation of application specific on-chip interconnection networks. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 231–237, Washington, DC, USA, 2005. IEEE Computer Society.