Providing QoS Guarantees in a NoC by Virtual Channel Reservation

Nikolay Kavaldjiev, Gerard J. M. Smit, Pascal T. Wolkotte, Pierre G. Jansen

Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente, the Netherlands
{n.k.kavaldjiev, g.j.m.smit, p.t.wolkotte, p.g.jansen}@utwente.nl

Abstract. Virtual channel reservation is a simple approach for providing guaranteed throughput services in a virtual channel network-on-chip. However, its performance is limited by the number of virtual channels per physical channels. In this paper we explore the limits of the approach and investigate how these limits depend on the routing algorithm, the traffic locality, the network topology and the network size. The results show the the approach can be applied in a network of size 10-by-10 nodes with four virtual channels per physical channel. The traffic locality has strong influence on the performance limits of the approach and can also help in reducing the communication energy cost by 50% to 70%. The type of the routing algorithm does not practically influence the performance limits.

1 Introduction

Multiprocessor System-on-Chip (MPSoC) is an emerging platform for the future mobile devices, e.g. PDAs, media players, mobile phones etc. To meet the functional requirements of these devices, such a platform should provide flexibility together with high performance and low power consumption. A promising approach for satisfying these contradicting requirements is though reconfigurable domain-specific computing. The work presented in this paper is performed as a part of the Gecko project which addresses architectural and design issues in low-power dynamically reconfigurable multimedia systems. The platform we envision for these devices is a MPSoC consisting of a large array of coarse grain reconfigurable processing elements (PEs) and distributed memories. The PEs are heterogeneous and domain specific, performing efficient high performance computation for specific application domains. One of the major issues in such a system is the communication between the PEs. The traditional system bus is not a solution because it is not scalable and cannot sustain the increasing bandwidth demands. The bus easy becomes a bottleneck and hence in the future MPSoC it is replaced by a light weighted communication network built on-chip, also known as Network-on-Chip (NoC) [1]. In this paper we discuss the NoC we propose for our MPSoC.

The network we consider is constructed in the following way. The PEs in our system are arranger in a two-dimensional array. Each PE is equipped with a network router it uses for inter processor communication. The network routers are

connected in a grid by full-duplex channels build by two unidirectional channels - one in each direction. The unidirectional channels are referred to as *physical channels*. In our system we use a virtual channel network [2]. In a virtual channel network, on each physical channel there are several *virtual channels* (VCs). Data in the network is transported over the VCs.

The system we envision is dynamic and reconfigurable at run-time. The applications that will run in the system are not known in advance, but are decided at run-time. A central system authority starts and terminates applications at run-time. When an application is started, the central authority allocates and configures PEs for the application and reserves communication channels in the NoC to carry the data streams between the PEs. When the application is terminated the resources it uses are freed.

Since many of the applications in mobile multimedia devices are real-time, predictable system communications are important. Predictable communications in our network are provided by means of guaranteed throughput (GT) services. The network can provide connections with a guaranteed minimal throughput bound. To guarantee the bound we use a *virtual channel reservation* - the VCs traversed by a connection are reserved and not used by other communications. Such approach is simple, but its potential is limited by the number of VCs in the network. Since there are finite number of VCs on each physical channel, the number of connections that can traverse a physical channel is limited and thus is limited the number of connections that can be opened simultaneously in the network. The number of VCs cannot be increased arbitrary, because it has a strong impact on the router area.

In this paper we explore the limits of the virtual channel reservation approach in a network of size 10-by-10 PEs with four VCs per physical channel. Considering the available chip area and the size of the processing elements this network size is feasible for the today and near future systems. The number of VCs is chosen such that the routers have reasonable size. We also investigate how the limits of the virtual channel reservation approach depend on the network routing algorithm, the network traffic locality and the network topology.

The paper is organized as follows. Section 2 discusses related work. The network is presented in Section 3. Section 4 discusses how the GT traffic is routed in the network and what algorithms are used for that. In Section 5 a model of the GT traffic in the network is constructed. Section 6 describes the performed experiments and Section 7 discussed the simulation results.

2 Related Work

In this section we briefly review the QoS solutions in NoCs. In the Ethereal network-on-chip [3] guaranteed services are based on time-division multiplexing (TDM). The communications on the physical channels are globally scheduled in time slots. A TDM approach is used also in the Nostrum network [4]. Although simple from implementation point of view, the TDM approach is static and not flexible enough for a dynamic system. Small changes in the network configuration

may require complete recomputation of the schedule. The distribution of the new schedule requires reconfiguration of all the routers along the changed paths. Furthermore, the global schedule requires a global notion of time in the system which may become a disadvantage in the near future when systems are expected to be Globally Asynchronous Locally Synchronous (GALS) [5].

A circuit switching NoC is another solution for providing guaranteed services. It benefits from small size and low energy consumption but is restrictive in the number of circuits that can be established. Wolkotte et al [6] overcome the problem by providing more than one physical channel between the neighbour routers. However, an additional network is needed for handling the best effort traffic in the system and for network configuration. The time for establishing a circuit cannot be neglected because all the switches along the circuit have to be reconfigured.

Another approach for providing guaranteed services in a network-on-chip is by introducing priorities. Such an approach is used by Felicijan et al [7] to provide guarantees in a virtual channel network. The VCs over a physical channel have statically assigned priorities. The high priority VCs are used for guaranteed traffic and the low priority VCs are used for best effort traffic. While this approach can guarantee better services for the traffic using higher priority VCs it cannot give exact bound on the provided services.

3 Network Operation

Here we briefly present the on-chip network we propose for interconnecting the PEs in the system. It is a packet switching virtual channel network that provides GT as well as Best Effort (BE) services [8]. The network consists of a grid of routers interconnected by physical channels. Each router is connected to a PE which serves as a source and sink of data. On each physical channel there are 4 VCs , this number being motivated by the trade-off between performance and area of a virtual channel router studied by Dally [2]. The VCs time-share the physical channel but are separately buffered at the router input. The physical channel is shared on a cycle-by-cycle basis in a round-robin fashion, but cycles are only used by the VCs that transmit data; the idle VCs do not use cycles. Since sharing is done in a round-robin fashion, the VCs equally share the physical channel bandwidth. If on a physical channel of bandwidth b there are v VCs currently transmitting data, then each of these v VCs is guaranteed a throughput of

$$TH_{min} = \frac{b}{v} \tag{1}$$

This is the worst case throughput the traffic on the v VCs can experience. Whatever traffic load is applied to the v VCs their throughput will never go below TH_{min}. Therefore, guarantees on the throughput bound of a VC can be given by restricting the number of VCs used on the same physical channel. If a minimal throughput bound TH_R is requested for a VC, then according to Eq. (1) the number v of VCs used on the same physical channel should be

$$v \leq \left\lfloor \frac{b}{TH_{min}} \right\rfloor \quad (2)$$

With four VCs per physical channel in our network we can guarantee throughput of b, b/2, b/3 or b/4.

In traditional virtual channel networks VCs are allocated to packets dynamically by the routers [2]. In such a network the number of currently occupied VCs depends on the current traffic and cannot be determined. Therefore no throughput guarantees can be given for a VC. In contrast, in our network VCs are statically allocated. We use a source routing, which is a technique where the packet destination address describes the exact route in terms of VCs that the packet takes to the destination. The addresses are generated by the central authority and given to the PEs during their configuration. Knowing the routes already in use, the central authority can determine which VCs are used. Therefor, it can predict their throughput and give guarantees.

In our network GT services are provides on a connection basis. A route is found between the source and the destination node and the VCs the route traverses are reserved and not used for other communications. Such a route is called *connection*. The throughput of the connection is determined by the VC with minimal throughput among the traversed. If Eq. (2) holds for all VCs the connection traverses then it guarantees minimal throughput bound of TH_R.

Routes for connections are provided by the central authority using a routing function. The routing function searches for a route traversing only VCs that can satisfy the connection throughput request TH_R according to Eq. (2). Thus, the routing function is in charge of providing GT connections for the application.

4 Routing Function

The task of the routing function is to find routes for the GT connections in the network. The function has the form $\mathrm{R(S,D,TH}_R)$. It takes as input a of a connection description and returns as a result a description of network route. The connection description consists of a source node S, a destination node D and a requested throughput TH_R. The route description is an ordered sequence $<\mathrm{vc}_1, \mathrm{vc}_2, , \mathrm{vc}_n>$ of the virtual channels vc_i traversed by the route. The requested throughput can be a real number but according to Eq. (2) the guaranteed throughput bound is always discrete and takes values b, b/2, b/3 and b/4.

To guarantee the specified throughput the routing function looks for a route traversing only VCs for which Eq. (2) holds. To find such a route, the routing function needs to know the current state of all VCs in the network. The state of a VC is represented by one integer set to 0 when the VC is not used or indicating the throughput that is guaranteed when the VC is used. The states of all VCs form the network state. When searching for a route, the routing function checks the network state and uses only free VCs that satisfy the following two *GT routing criteria*: i) the VC can guarantee the requested throughput according

Eq. (2), ii) the use of the VC will not violate the throughput guarantees already given (if any) by the other VCs on the same physical channel (Eq. (2) will still hold for them). After the route has been constructed, the routing function updates the state of the used VCs. When the route is not needed anymore, e.g. the application using it has terminated, the VCs constructing the route are freed and their state is updated.

Finding a route in a network is equivalent to finding a path between two nodes in a graph - the network topology is represented as a graph and a path searching algorithm is run on it. Among all possible paths the shortest is preferable, because shorter network routes result in less network traffic and less energy for communication. Therefore, the routing function is based on an algorithm for the shortest path search in graphs. However, our routing function runs on a subgraph I=(N,C) of the full network graph, derived by deleting all channels that do not satisfy the two GT routing criteria. Here N represent the set of network nodes and C represent the set on channels.

The routing function is used at run-time and therefore has to be as fast and simple as possible. But a simple algorithm may lead to poor network utilization generating congestions at some parts of the network while other parts stay unutilized. To examine the influence of the routing algorithm, we experiment with two shortest path search algorithms [9]: *Breadth-first search* and *Dijkstra's algorithm*.

Breadth-first search (BFS) is the simplest possible shortest path search algorithm. It works on non-weighted graphs and finds a shortest path in terms of the number of edges. It routes without taking into account the network condition - weakly and heavy loaded physical channels are equally preferred. The computational complexity of the algorithm is $O(N)$. The memory complexity of the algorithm is also linear in the number of network nodes.

Dijkstra's algorithm (DA) is more complex and allows the routing decision to adapt according the current network conditions. The algorithm works on weighted, directed graphs, where all edge weights are nonnegative. It finds shortest paths in terms of the minimal weighted sum. In our network, the weight assigned to an edge is proportional to the load of the corresponding physical channel. The weight equals the number of occupied VCs on the corresponding physical channel plus one, as one stands for a unit physical distance. Thus, if no virtual channel is occupied the weight is one and if all four VCs are occupied the weight is 5. One is added to avoid zero weight and thus to provide that the algorithm always prefers shorter paths. Every time a connection is routed the state of the reserved VCs is changed which increases the weights of the physical channels traversed by the connection. When the connection is deleted, the VCs are released and the weights are reduced. Hence, the weights reflect the current network state and the routing algorithm adapt its decision according to this state. The computational complexity of Dijkstra's algorithm is $O(N^2)$. The memory complexity of the algorithm is linear in the number of vertices.

BFS and DA have the same memory complexity but DA has a higher computational complexity than BFS. We examine how the routing algorithm influences

the performance of the routing function and whether it is profitable to use DA instead of the simpler BFS.

5 Traffic Model

In this section we construct a model of the GT traffic in the network. The model is later used for evaluation of the routing function. We model only the spatial aspects of the traffic, like communication pattern and communication distances, and not the timing aspects. The timing aspects, like data generation rate or data inter arrival time are entirely hidden behind the requested throughput TH_R. The traffic spatial characteristics are determined by two factors - topology of the application graphs and strategy for mapping of the application graphs on the multiprocessor architecture. The application is represented as a graph $G_A = (V_A, E_A)$. The graph vertices V_A represent processes to run on the PEs and the graph edges E_A represent the communication between the processes. To run an application, the vertices of the application graph are mapped on PEs in the system. After the mapping the edges between the processes define the communications between the PEs to be handled by the network.

GT traffic in the system is generated by streaming applications which typically have application graphs with simple pipeline structure [10][11]. At a certain moment in time a number of streaming applications are running simultaneously in the system, hence there are number of pipeline graphs scattered over the PEs. To model such traffic conditions we use a large graph of ring topology which nodes are scatter over the PEs. A large ring graph can be seen as a serial connection of many short pipeline graphs. The number of nodes in the ring graph is equal to the number of PEs in the system and every node is mapped on a separate PE. Thus, we model a system where every PE generates and consumes a data stream.

The mapping decides the actual PEs where the application processes will run and therefore it has a strong influence on the communication locality. To model the effect of traffic locality we use three different strategies for mapping the ring topology graph. The three strategies produce mappings that approximate respectively the best, the average and the worst case of traffic locality. The three mapping strategies use the same algorithm for choosing the PEs, but differ in a parameter given to the algorithm. The algorithm operates on the ring graph as follows. The graph nodes are mapped sequentially in the order they appear in the graph. For every next node a PE is chosen randomly among those which are at distance less than or equal d hops from the PE where the previous graph node was mapped. Here d is a parameter of the algorithm that sets a diameter for the preferred network distance. If there is no free PE within that distance, then a PE is chosen randomly among all free PEs. The three mappings strategies differ only in the value they set the parameter d to. The first strategy tries to maximize the traffic locality; it sets the parameter d to 1 and approximates *best* case locality. The second strategy sets d to 4 and approximates some *average* case locality. The third strategy approximates *worst* case locality. It sets the parameter d to

the diameter of the network (the longest network distance). Hence, the mapping algorithm uniformly scatters the vertices of the ring graph over the PEs and no locality should be expected.

6 Simulation Experiments

To explore the performance limits of the routing function and hence of the virtual channel reservation approach, we perform a number of simulation experiments using the traffic model from Section 5. In an simulation experiment a ring graph of 100 nodes is mapped on a network of size 10-by-10 nodes. The mapping is randomized but generated with specific locality characteristics (worst, average, best case locality). After the mapping, a routing function provides GT connections for the communication channels defined by the edges of the ring graph, all with the same requested throughput TH_R. Thus, 100 GT connections following a ring communication pattern are routed. The routing is considered to be *successful* when all the 100 GT connections are routed. When routes cannot be found for all connections, the routing is considered to fail. If a routing is successful, information is collected about the network distances of the routed GT connections and the utilization of the VCs in the network. Experiments are performed for the three traffic locality conditions (worst, average, best case locality), with two network topologies (mesh and torus), with two routing functions based respectively on the BFS and DA algorithms and with four values for the requested throughput TH_R (b, b/2, b/3 and b/4). All 24 combinations of these factors are explored. To asses the average performance of the routing function, for each of the 24 combinations we perform 1000 experiments, each experiment setting a *sample* in the space of the possible traffic patterns. The number 1000 was chosen empirically such that to provide enough samples for representative average results in acceptable simulation time. Thus in total 24x1000 samples are collected.

7 Simulation Results

In this section we present and discuss the results of the conducted experiments. We compare how the different factors influence the performance of the virtual channel reservation approach in order to decide which of them are of importance and can be used to improve the performance of the approach and which can be neglected.

7.1 Number of successful samples

Figure 1 shows how many of the 1000 samples taken in each of the 24 combination of factors are successful . The three graphs correspond to the three cases of traffic locality, each graph presenting the results for mesh and torus topology. Of interest for us are the cases for which all 1000 samples are successful. We

assume that in these cases the requested GT connections can always be provided; therefore, the virtual channel reservation approach can be safely applied. In the cases when not all samples are successful the routing function cannot always provide all the requested GT connections and the virtual channel reservation approach performs insufficiently for these requirements.

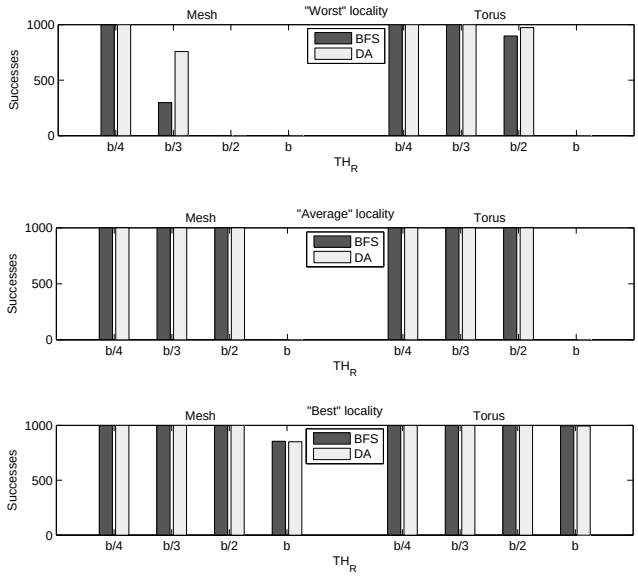

Fig. 1. Number of successfully routed mappings

Figure 1 we see that for worst case traffic locality the virtual channel reservation approach can be safely applied if the requested throughput TH_R is restricted to *b/4* for mesh topology and up to *b/3* for torus topology. The torus topology helps improving the performance in such traffic conditions by increasing the throughput limit from *b/4* to *b/3*. When locality is introduced the performance is improved by increasing the limits on the TH_R to b/2 and b. But for local traffic the improvement achieved by replacing mesh with torus topology is not significant. The routing algorithms do not change the performance of the approach for any traffic conditions. Among the three factors - locality, topology and routing function - the traffic locality has the strongest influence on the performance limits of the virtual channel reservation approach while the routing algorithm does not influence it significantly. The results show also that four VCs per physical channel provide enough network resource for applying the virtual channel reservation in a 10-by-10 network; in all the cases, if TH_R is restricted, the approach can be applied . Restriction on TH_R restricts the maximal through-

put guaranteed to a connection to some fraction of the capacity b of the physical channel. Thus, at network design time an appropriate b has to be chosen (e.g. by choosing the operation frequency of the network and the width of the physical channel).

7.2 Detour cost

The routing function tries to route the requested connections using minimal distance routes, but this is not always possible because some VCs along the minimal distance routes might be occupied. In such a case the routing function takes a detour - a route which is not minimal. *Detour cost* is defined as the difference between the real route distance and the minimal distance. The real distances are the result of the routing, while the minimal distances are idealistic, assuming there is no other traffic in the network. The better routing algorithms manage to route the traffic using shorter paths and therefore with less detour cost.

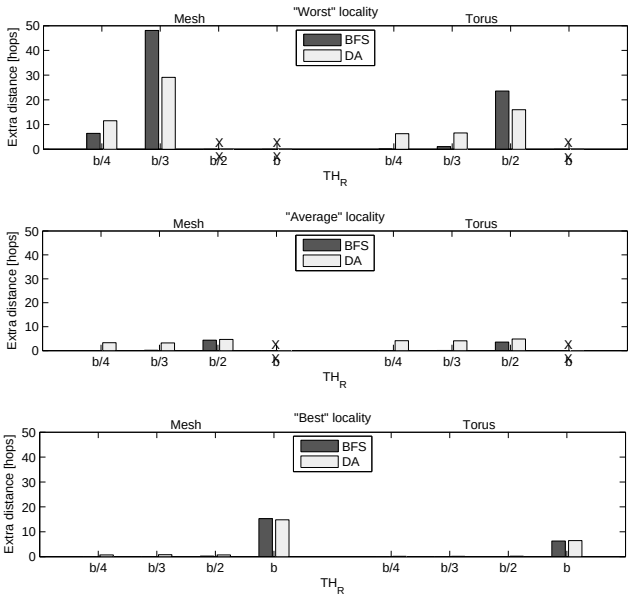

Fig. 2. Average detour cost (sum over 100 connections)

The detour cost is shown in Figure 2. The presented figures give the sum of the detour cost of all 100 connections in the ring graph. In most of the cases the sum detour cost is less than ten hops, which is negligible compared to the sum distances of the 100 connection (at least 100 hops). The detour cost exceeds

10 hops only when the requested GT connection cannot always be provided, therefore the approach cannot be used.

7.3 Communication energy cost

Wolkotte et al [12] performs power analysis of our virtual channel router and derives a energy model of the network. An energy model of a circuit switching network is also derived. We use the two energy models to estimate and compare the average communication energy cost in the proposed virtual channel network and in the circuit switching network proposed by Wolkotte. The energy models estimate the energy cost in [pJ/bit] for transporting a bit in the network

$$E_{ps} = E_R(N_{hop} + 1) + (0.39 + 0.12 l_{wire}) N_{hop} \quad (3)$$

Here l_{wire} is the length of a physical channel in mm. N_{hop} is the network distance in number of hops. E_R stands for the energy cost for traversing a router; for the packet-switching and the circuit-switching network E_R takes values E_{R_PS} = 0.98 and E_{R_CS} = 0.37. The second term in the energy model estimates the energy for traversing the wires between two routers (a physical channel). The model captures only the dynamic energy cost for transporting a bit.

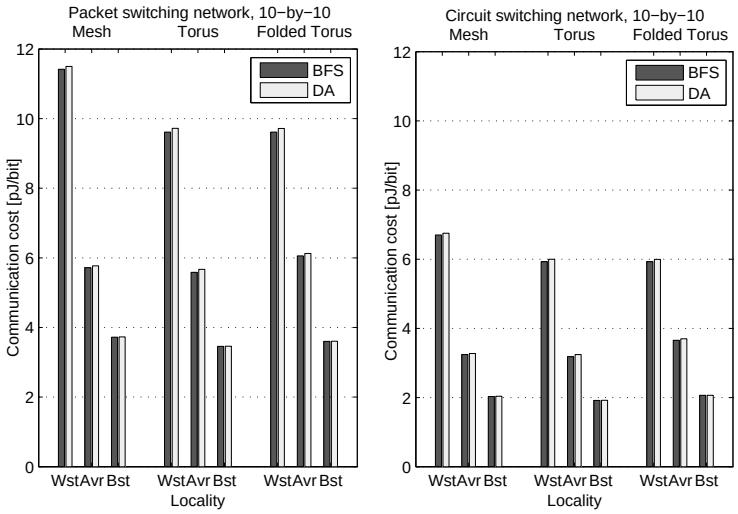

Fig. 3. Average communication energy cost in a packet switching and in a circuit switching network

The energy cost is estimated for three topologies - mesh, torus, and folded torus. A folded torus has the same graph structure as torus, but its nodes are reshuffled in the plane such that the torus wraparound channels are avoided in

expense of regular channels with doubled length [13]. We assume that the size of a PE is 1.5x1.5 mm or 2.25 mm^2 [14]. Thus, the length of the physical channels and therefore of l_{wire} in mesh is 1.5 mm. In the torus topology the wraparound channels cross the entire array of 10-by-10 PEs and hence are 15 mm long. In the folded torus the channels in the middle of the network cross two PEs and are 3 mm long. The network distance N_{hop} in Eq. (3) is substituted with the mean communication distances calculated from the simulation results. To take into account that the wraparound channels in torus has different length, Eq. (3) is modified to contain two terms that capture the wires energy contribution - one term for the regular channels and one for the wraparound channels. During the experiments information is collected about the utilization of the single hop channels and the wraparound channels. This information is used to weight the number of hops N_{hop} in the two terms.

Figure 3 presents the results for the communication energy cost in our virtual channel network (packet switching) and in the circuit switching network proposed by Wolkotte. The results show that by exploiting the communication locality the average communication energy cost can be reduced by 50% to 70% for the different topologies. For worst case locality traffic, the torus topology reduces the communication energy cost compared to mesh because of its smaller network diameter. But since the traffic locality reduces the communication distances, for local traffic the smaller diameter of the torus is not advantageous. The routing algorithm can influence the communication energy cost by the detour cost - higher detour cost entails more energy for communication. But as we saw, the detour cost is negligibly small, which explains the insignificant influence of the routing algorithm on the communication energy.

The energy cost results for the circuit switching network is reduced compared to the results for the packet switching network. This is due to the smaller energy cost for traversing a router in the circuit-switching network - a clear advantage of this approach. Unfortunately, the circuit switching solution is less flexible - it requires external configuration, cannot handle best effort traffic.

8 Conclusion

In this paper we explored the possibility to provide guaranteed throughput services in a virtual channel network by virtual channel reservation. We test the limits of the virtual channel reservation approach for variety of traffic conditions. The results show that the approach is feasible and can be used for providing throughput guarantees in a 10-by-10 network in worst case traffic conditions. For this network size a mesh topology and four virtual channels per physical channel provide enough connectivity for predictable system operation. Amongst the considered factors that influence the performance limits of the approach, the communication locality has the strongest influence. By exploiting communication locality the network performance is improved and, at a certain extend, made independent of the network size. Furthermore, exploiting the locality the communication energy cost can be reduced by 50% to 70%. The routing algorithms,

based respectively on the simple Breath-first search algorithm and the more complex Dijkstra's algorithm, do not show noticeable performance difference. Therefore the Breath-first search is preferred because of its lower computational complexity.

References

1. Dally, W., Towles, B.: Route packets, not wires: on-chip interconnection networks. In: Proceedings of the 38th Conference on Design Automation (DAC'01), ACM Press (2001) 684 – 689
2. Dally, W.: Virtual-channel flow control. IEEE Transactions on Parallel and Distributed Systems, **3**(2) (1992) 194–205
3. Goossens, K., van Meerbergen, J., Peeters, A., Wielage, R.: Networks on silicon: combining best-effort and guaranteed services. In: Proceedings of the Design, Automation and Test in Europe Conference (DATE'02). (2002) 423 – 425
4. Millberg, M., Nilsson, E., Thid, R., Jantsch, A.: Guaranteed bandwidth using looped containers in temporally disjoint networks within the nostrum network on chip. In: Proceedings of the Design, Automation and Test in Europe Conference (DATE'04). Volume 2., IEEE Computer Society (2004) 890 – 895
5. Muttersbach, J., Villiger, T., Kaeslin, H., Felber, N., Fichtner, W.: Globally-asynchronous locally-synchronous architectures to simplify the design of on-chip systems. In: Proceedings of the 12-th Annual IEEE International ASIC/SOC Conference. (1999) 317–321
6. Wolkotte, P., Smit, G., Rauwerda, G.: An energy-efficient reconfigurable circuit switched network-on-chip. In: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05). (2005) 155–161
7. Felicijan, T., Furber, S.: An asynchronous on-chip network router with quality-of-service (qos) support. In: Proceedings of the IEEE International System-on-Chip Conference (SOCC'04), IEEE Computer Society (2004) 274–277
8. Kavaldjiev, N., Smit, G., Jansen, P.: A virtual channel router for on-chip networks. In: Proceedings of the IEEE International System-on-Chip Conference (SOCC'04), IEEE Computer Society (2004) 289–293
9. Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to algorithms. 2 edn. MIT Press, Cambridge, Massachusetts (2001)
10. Rauwerda, G., Heysters, P., Smit, G.: Mapping wireless communication algorithms onto a reconfigurable architecture. Journal of Supercomputing **30**(3) (2004) 263–282
11. Wolkotte, P., Smit, G., Smit, L.: Partitioning of a drm receiver. In: Proceedings of the 9th International OFDM-Workshop. (2004) 299–304
12. Wolkotte, P., Smit, G., Kavaldjiev, N., Becker, J., Becker, J.: Energy model of networks-on-chip and a bus. In: Proceedings of the International Symposium on System-on-Chip (SoC'05). (2005) 82–85
13. Dally, W., Towles, B.: Principles and Practices of Interconnection Networks. The Morgan Kaufmann Series in Computer Architecture and Design. Morgan Kaufmann, San Francisco, CA (2003)
14. Heysters, P., Smit, G., Molenkamp, E.: A flexible and energy-efficient coarse-grained reconfigurable architecture for mobile systems. Journal of Supercomputing **26**(3) (2003) 283–308