Optimizing Collective Communications on the K-port Spidergon Network

Jiri Jaros
Dept. of Computer Systems
Faculty of Information Technology, BUT
Brno, Czech Republic
e-mail: jarosjir@fit.vutbr.cz

Vaclav Dvorak
Dept. of Computer Systems
Faculty of Information Technology, BUT
Brno, Czech Republic
e-mail: dvorak@fit.vutbr.cz

***Abstract*—The paper investigates an impact of using *k* ports in the direct communication model of collective communications on the overall performance of the Spidergon interconnection network. Since the higher number of *k* internal ports can improve performance but increase the cost of interconnection network, the performed analysis introduces the ideal performance-cost tradeoff on slim- and fat-node Spidergon networks.**

Keywords-collective communications; k-port model; Spidergon; slim-nodes; fat-nodes; router usage; latency

I. INTRODUCTION

With an increasing number of processor cores, memory modules and other hardware units in the latest chips, the importance of communication among them and of related interconnection networks is steadily growing. The memory of many-core systems is physically distributed among computing nodes that communicate by sending data through a Network on Chip (NoC) [1].

Communication operations can be either point-to-point, with one source and one destination, or collective, with more than two participating processes. Some embedded parallel applications, like network or media processors, are characterized by independent data streams or by a small amount of inter-process communications [2]. However, many general-purpose parallel applications display a bulk synchronous behavior: the processing nodes access the network according to a global, structured communication pattern.

The performance of these collective communications (CC for short) has a dramatic impact on the overall efficiency of parallel processing. The most efficient way to switch messages through the network connecting multiple processing elements (PEs) makes use of wormhole (WH) switching. Wormhole switching reduces the effect of path length on communication time, but if multiple messages exist in the network concurrently (as it happens in CCs), contention for communication links may be a source of congestion and waiting times. To avoid congestion delays, CCs are necessary to organize into separated steps in time and to put into each step only such pair-wise communications whose paths do not share any links. The contention-free scheduling of CCs is therefore important.

The port model of the system defines the number *k* of PE ports that can be engaged in communication simultaneously. This means that beside $2d$ network channels, there are $2k$ internal unidirectional (DMA) channels, *k* input and *k* output channels, connecting each local processor core to its router that can transfer data simultaneously. Always $k \leq d$, where *d* is a node degree; a one-port model (k=1) and an all-port router model (k=d) are most frequently used. Typically, higher number of ports reduces communication overhead, but on the other hand, increases the complexity of routers and duplicates network interfaces in connected PEs.

In the most common one-port system, a PE has to transmit (and/or receive) messages sequentially (using only one local channel). The messages may block on occupied injection channel, even when their required network channels are free. These systems are very easy to implement and are often used in computer clusters equipped with only one network interface.

Architectures with multiple ports alleviate this bottleneck. In the all-port router architecture, there are as many local PE channels as there are network channels that reduce the message blocking latency during CC operations. On the other hand, an addition of internal ports requires more complex router and makes the system more expensive. Such all-port routers can be often found in systems on a chip. Fig. 1 illustrates the differences between one-port and all-port switches.

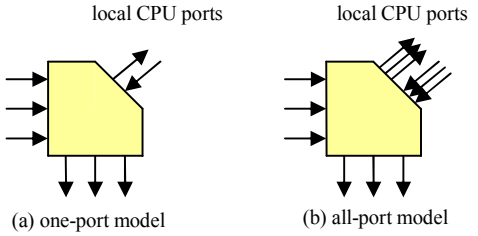


Figure 1. Port models for 3-regular Spidergon network.

The *k*-port model is a generalization of the port models and has been widely used, e.g., in [3] and [4]. An appropriate number of internal ports can boost the performance and keep the router complexity at reasonable level.

One example of successful *k*-port NoCs implementation is presented in [5] and [6]. The authors investigate the speedup of broadcast communication inside the Cell Broadband Engine processor [7], [8] and prove that using multi-ports (up to four) significantly reduces the broadcast latency of short messages. Unfortunately, no idea about other communication patterns was given there.

 ISBN:978-1-61208-114-4

The paper [9] presents a novel analytical model to compute communication latency of multicast as a widely used collective communication operation in wormhole routed Spidergon [10] and Quarc [11] network. This model can predict the latency of broadcast communication in asynchronous multi-port wormhole networks. Unfortunately, the paper does not present any advantages of multi-port model against one-port model.

K-port model can also be effectively used in high-end workstations that are more and more often equipped with two network interfaces. An example implementation of *k*-port model can be found in SuperMicro SuperBlade servers [12]. These servers are equipped with 2-port Gigabit Ethernet connectors and thus can be connected into the interconnection network, e.g., Spidergon, using two ports.

The influence of using *k*-port on CC overhead and complexity of on-chip design has not been deeply investigated as yet. This paper deals with examining the advantages and disadvantages of *k*-port model on Spidergon network [10] with slim and fat nodes. It introduces the optimal number of ports for several Spidergon configurations and discusses their overhead.

The paper is structured as follows. Section II describes the time complexity of CCs and derives their lower bounds. Section III introduces the Spidergon topology and its slim-node and fat-node configurations. The implementation requirements of Spidergon are discussed here. Section IV presents known schedule techniques for CC on the Spidergon network and identifies their weakness. The new CC schedules obtained by means of evolutionary algorithms are presented in section V. The comparison of slim-node and fat-nodes simultaneously with various number of ports are outlined there. The Conclusion summarizes the achieved results and introduces the most suitable configurations.

II. TIME COMPLEXITY OF COLLECTIVE COMMUNICATIONS

A collective communication is usually defined in terms of a group of processes. The operation is executed when all processes in the group call the communication routine with matching parameters. We classify collective operations into three types according to their purpose:

- CCs (OA-One-to-All, AO-All-to-One, AA-All-to-All),
- global computation (reduction AOR or AAR and scan)
- synchronization (barrier).

The CCs are most important, as other collective operations are closely related to them. In a broadcast (OAB), one process sends the same message to every group member, whereas in a scatter (OAS), one process sends a different message to each member. Gather (AOG) is the dual operation to scatter, in that one process receives a message from each group member. These basic operations can be combined to form more complex operations. In all-to-all broadcast (AAB), every process sends a message to every other group member. In complete exchange, also referred to as all-to-all scatter-gather (AAS), every group member sends a different message to every other group member. Permutation operations, such as shift and transpose, are also CCs. Since complexities of some communications are similar (AOG ~ OAS, AOR ~ OAB, AAR ~ AAB), we will focus only on four basic types (OAB, OAS, AAB, AAS). Also, from now on, when we refer to "collective communications", then we will assume only CCs involving the group of all processes.

The simplest time model of point-to-point communication in direct WH networks takes the communication time composed of a fixed start-up time t_s at the beginning (SW and HW overhead of a sender and a receiver), a serialization delay, i.e., the transfer time of m message units (words or bytes), and of a component that is a function of distance h (the number of channels on the route or hops a message has to do):

$$t_{WH} = t_s + mt_1 + ht_r \quad (1)$$

where t_1 is per unit-message transfer time and t_r includes a routing decision delay, switching and inter-router latency. A relatively small dependence on h may be taken into account by including $h_{\max}t_r$ into t_s, so that only two parameters t_s and mt_1 are sufficient.

In the rest of the paper we assume that the CC in WH networks proceeds in synchronized steps. In one step of CC, a set of simultaneous packet transfers takes place along complete disjoint paths between source-destination node pairs. If the source and destination nodes are not adjacent, the messages go via some intermediate nodes, but PEs in these nodes are not aware of it; the messages are routed automatically by the routers attached to PEs.

Complexity of collective communication will be determined in terms of the number of communication steps or equivalently by the number of "start-ups" τ^{CC} (upper bound). Provided that the term $h_{\max}t_r$ is included in t_s and excluding contention for channels, CC times can be obtained approximately as the sum of start-up delays plus associated serialization delays m_it_1 in individual communication steps.

$$t_{CC} = \sum_{i=1}^{\tau^{CC}} (t_S + m_i t_1) = \tau^{CC}[t_S + mt_1] \quad (2)$$

The above expression assumes that the nodes can only re-transmit/consume original messages, so that the length of messages $m_i = m$ remains constant in all communication steps. This is true in the so called direct model of communication; on the contrary, in the combining model the nodes can combine/extract partial messages with negligible overhead. The direct/combining model influences CC performance and either one can outperform the other in some cases. Further on we will consider the direct model only.

Possible synchronization overhead involved in communication steps, be it hardware or software-based, should be included in the start-up time t_s. Let us note that with uniform messages and a single clock signal domain on NoC, one barrier synchronization before CC might be sufficient to synchronize the whole CC. Communication steps would then follow in the lockstep. According to frequency of CCs and an amount of interleaved computation

 ISBN:978-1-61208-114-4

(BSP model) in a certain application, efficiency of parallel processing can be estimated.

Further, the lower bound on the number of steps τ^{CC} depends on a channel type; we have to distinguish between unidirectional (simplex) channels and bi-directional (half-duplex HD, full-duplex FD) channels. Typically τ^{CC} will be twice as large for HD channels than for the FD ones. Further on we will consider FD channels. Finally, the lower bounds depend on number of internal ports k and node degree d.

The number of communication steps is in the first place influenced by the topology of an interconnection network. Generally the lower bounds $\tau_{CC}(G)$ for the network graph G depend on node degree d, number of internal ports k, number of processing elements P, and bisection width B_C, Table I.

TABLE I. LOWER BOUNDS ON THE NUMBER OF COMMUNICATION STEPS τ_{CC} (WH, K-PORT, DIRECT NETWORKS).

CC	τ_{CC} [steps]
OAB	$\lceil \log_{k+1} P \rceil$
AAB	$\lceil (P-1)/k \rceil$
OAS	$\lceil (P-1)/k \rceil$
AAS	$\max(\lceil P^2/(2B_C) \rceil, \lceil \Sigma/(Pd) \rceil, \lceil (P-1)/k \rceil)$

As far as the broadcast communication (OAB) is concerned, the lower bound on the number of steps $\tau_{OAB}(G) = s = \lceil \log_{k+1} P \rceil$ is given by the number of PEs informed in each step, that is initially 1, $1 + 1 \times k$ after the first step, $(k+1) + (k+1) \times k = (k+1)^2$ after the second step, etc.,…, and $(k+1)^s \geq P$ processing elements after step s. Since the broadcast message is the same for all the PEs, each PE once informed can help with distributing of the message in following steps.

In case of AAB communication, since each PE has to accept $P-1$ distinct messages, the lower bound is $\lceil (P-1)/k \rceil$ steps. A similar bound applies to OAS communication, because each PE cannot inject into the network more than k messages in one step.

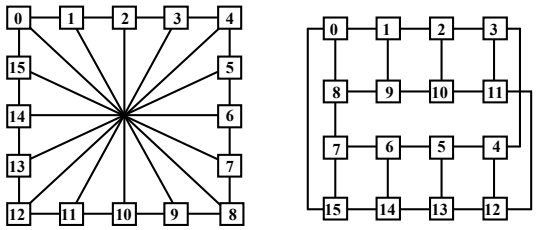
The lower bound for AAS can be obtained considering that half the messages from each PE cross the bisection, whereas the other half do not. There will be altogether $\lceil 2(P/2)(P/2)/B_C \rceil$ of such messages in both ways, where B_C is the channel bisection width [11]. Sometimes a stronger lower bound may be obtained considering the count of channels from all sources to all destinations (Σ) and the limited count Σ_1 of channels available for one step. In regular networks with constant node degree $\Sigma_1 = Pk$. As each PE has to accept $P-1$ distinct messages, $\lceil (P-1)/k \rceil$ bound has to be also obeyed.

Which lower bound takes effect depends on a particular network topology and the port model.

III. SPIDERGON TOPOLOGY AND ITS CONFIGURATIONS

Classical logarithmic diameter networks, e.g., hypercubes, butterflies and fat trees, provide enough bandwidth for all-to-all communications, but do not map well into the two dimensions provided by a silicon chip: the length of some interconnection wires increases proportionally to the number of nodes. This will decrease the clock frequency dramatically and degrade the performance. In this work we therefore restrict our attention to the Spidergon NoC topology with mostly local interconnection among processors.

The Spidergon depicted in Fig. 2 is the novel interconnection network architecture suitable for the on-chip communication demands of SoCs in several application domains [2]. The Spidergon NoC first reported in [10], and later in [11], has been recently adopted by STMicroelectronics [13] with the objective to realize low cost multiprocessor SoC implementation with topology opened for application-specific optimization. Spidergon is somewhere between the ring and mesh topologies: an even number of nodes is connected into a bidirectional ring and pairs of nodes are connected by a cross connection. Each edge in Fig. 2 represents two unidirectional physical links, one for each direction. In order to avoid deadlock, two virtual channels are multiplexed on each physical link. Fig. 2 depicts the 16-node Spidergon topology and its layout on a chip resembling a sparse mesh. Each node represents a router/switch (Fig. 2) and a PE.



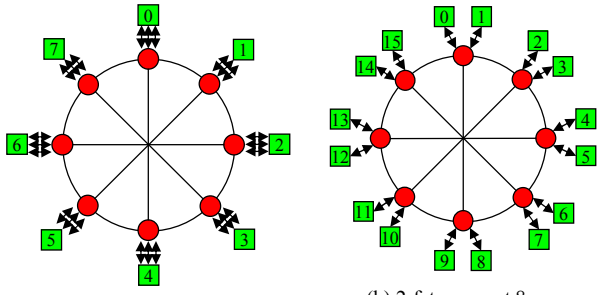
(a) basic spidergon topology (b) the on-chip layout

Figure 2. Isomorphic graphs of the 16-node Spidergon topology.

The nodes placed in direct interconnection networks can be divided into slim and fat ones. Slim nodes contain one PE and a router connecting it into the network. The PE and the router are connected with k internal ports. Slim nodes provide the highest communication performance but lower scalability of the network. Fat nodes with a few PEs connected with separate k internal ports to the router could provide cheaper solution for Spidergon networks of a larger size, similarly as fat hypercubes do. However, it is a trade-off between such measures as cost and performance.

Fig. 3 shows two examples of Spidergon configurations: the slim node all-port Spidergon with 8 nodes; and the 2-fat node one-port Spidergon with 8 nodes carrying 16 PEs.

Finding the optimal ratio between PEs connected to a single router and number of ports used to interconnect them is still an open question. Table II shows the total router port requirements for a few node configurations targeted to Spidergon networks. 1-port, 2-port and 3-port model with slim and 2-fat nodes are compared here. The number of total router ports (including internal and three external ones) is calculated for all configurations. Utilizations of prefabricated 8-port and 12-port routers that could be used for NoC implementation are shown here too.



(a) slim all-port 8-gon (b) 2-fat one-port 8-gon

Figure 3. Two different 8-node Spidergon networks.

From the Table II, it can be seen that the 2-port 2-fat nodes offer the highest utilization of available links in 8-port router. Only one link remains idle here. Very popular 3-port slim nodes utilizing 6 links also offer an acceptable utilization of router resources.

If the bigger routers are available in NoC, the 3-port 2-fat nodes can be used but at the cost of higher complexity of NoC. Let us note that for one-port slim Spidergon, 4-port router suits best because it utilizes all router channels.

TABLE II. ROUTER UTILIZATION FOR TARGET SPIDERGONS NODES

Port model	Node type	Router ports	Utilization 8p	Utilization 12p
1-port	Slim	4	50%	33%
2-port	Slim	5	62.5%	41.6%
3-port	Slim	6	75%	50%
1-port	2-fat	5	62.5%	41.6%
2-port	2-fat	7	87.5%	58.3%
3-port	2-fat	9	--	75%

IV. SCHEDULING CCS ON K-PORT SPIDERGONS

The Spidergon, as well as the bidirectional ring topology, though very simple, is not free from routing deadlock, because the channel dependency graph is not acyclic [13], [14]. This can be seen on a common permutation called the cyclic shift. The problem can be solved by the introduction of virtual channels [13] and by implementing rules on channel use. However, in conflict-free CCs all source to destination paths are disjoint and therefore there is no competition for shared resources, no danger of deadlock and no need for escape virtual channels. When implementing CCs, we therefore use either one of two virtual channels.

The deterministic shortest path routing algorithms proposed for the Spidergon architecture are so called Across First (aFirst) and Across Last (aLast) [13], [15]. Both algorithms are minimal source routing. An analytical performance model has been analyzed in [16] and the average message latency evaluated. Regarding CCs, only the broadcast and multicast CCs on Spidergon were studied in the past[11]. Other CCs, especially all-to-all communications have not been analyzed in the literature as yet.

In this work, we want to improve the performance of Spidergon NoC by designing such communication schedules that prevent any possible link contention. We also want to investigate the influence of the number of internal port k on time complexity of designed schedules and the space overhead of corresponding routers. Optimized CC schedules can be uploaded into switch routing tables and boost the performance of many parallel algorithms. For this reason, four common CC patterns based on broadcast and scatter services will be analyzed.

Further, it should be noted that the lower bounds for fat Spidergon cannot be theoretically estimated like in the case of slim Spidergon, see Table I. This phenomenon can be explained on the 2-fat one-port Spidergon. In this case, neither lower bounds for one-port nor for all-port model apply here. The reason is that we cannot assign 3 network ports of a node explicitly to internal cores (PEs).

Let us also note that the optimal schedules are not known for the k-port fat-node Spidergon networks so far.

The optimization of CC scheduling is based on evolutionary algorithms (EA). These techniques applied already to CC scheduling problem on hypercubes of medium size (tens of nodes) [17] were able to find the already known optimum solutions obtained analytically. A schedule (chromosome) is encoded as a set of pair-wise transfers determined in space and time. The fitness function checks the validity of candidate CC schedules. A valid CC schedule for a given number of communication steps must be conflict-free. There are no shared resources (links/ports) in such a schedule. Valid schedules are either optimal (the number of steps equals the lower bound) or suboptimal. Evolution of a valid schedule for the given number of steps is completed as soon as fitness (number of conflicts) drops to zero. If it does not do so in a reasonable time, the prescribed number of steps must be increased of one step and lunched again.

However, for some CCs studied in this work analytic methods to find optimum schedules do not exist, so that the results can be compared only to theoretical lower bounds. The evolution gives us the upper bounds of time complexity that can be attained. It should be noted, that it is not clear if the lower bound can ever be reached.

V. REACHED UPPER BOUNDS ON TIME COMPLEXITY ON SPIDERGON NETWORKS

In this work, slim-node Spidergons with 6, 8, 10, 12, 14, and 16 nodes were examined. Fat-node Spidergons were represented by 6, 8, 10, 12, 14 and 16 nodes Spidergons with 12, 16, 20, 24, 28, and 32 PEs. The near optimal schedules for varying number of internal ports were sought using evolutionary algorithms. Table III and Table IV summarize the time complexity of designed schedules in terms of communication steps (upper bounds).

Two integers in one cell separated by a slash indicate that the lower bound (a smaller integer) has not been reached. A single integer represents both the lower and the upper identical bounds reached by an EA, or the lower bound cannot be determined.

 ISBN:978-1-61208-114-4

A. Experimetnal Results on Slim-node Spidergons

Table III illustrates the upper bounds of one-to-all CCs are identical with the theoretically derived lower bounds in all cases. The upper bounds are proportional to the number of internal ports *k*. The OAB communication does not depend on *k* too strongly. On the other hand, OAS makes profit from higher number of internal port in full.

A slightly different situation can be seen in the case of all-to-all CCs. The lower bounds were not reached for AAB in all cases, especially for 12- and 14-node Spidergon. There were achieved only suboptimal solutions with one step worse time complexity here. The number of communication steps of AAB depends on number of internal ports significantly, and so, using higher number of port is always better.

The most complex AAS communication shows only a small dependence on number of internal ports. It is given by saturating the network with messages injected even using one port.

B. Experimetnal Results on Fat-node Spidergons

The lower bound on time complexity can be derived only for one-to-all communication in the case of *k*-port fat-node Spidergon. The reason is that we cannot assign 3 network ports of a node explicitly to internal cores. Evolutionary algorithms reached the lower bounds in all cases and designed as fast schedules as possible.

The upper bounds on time complexity are shown in Table IV. The lower bounds of OAB are not very dependent on port model. Further, these values are very close to the results obtained for slim-node spidergons. There are only one step differences in most cases, but with twice more connected PEs. The results reached for OAS show double upper bounds, which is caused by double number of PEs.

The results of evolution produced for AAB show the strong dependency on number of internal ports. The increase of upper bound is more than linear for 2-fat Spidergon than in the case of slim Spidergon.

Finally, Table IV illustrates an insignificant influence of *k* on AAS upper bounds. In most cases, the one-port model is sufficient. Let us note, the lower bound for *k* ports cannot be derived exactly, and so, the upper bounds reached by evolutionary algorithms give us the most accurate estimation.

C. Comparison of Slim-node and Fat-node Spidergons

In this subsection, we would like to mutually compare slim-node 16-Spidergon and 2-fat 8-Spidergon. These topologies connect the same number of PEs but in different manners. In the case of slim-node 16-Spridergon, there are 16 nodes placed around the ring and interconnected using cross links. In addition, each PE holds its own router. In the case of 2-fat 8-Spidergon, there are only 8 nodes around the ring and a router is shared between two PEs.

Looking at Table III and Table IV it is evident that upper bounds for one-to-all CCs are the same. The slim-node Spidergon is slightly better for AAB, but on the other hand, it is outperformed by fat-node one for AAS.

Similar observation can be done comparing slim-node 12-Spidergon and 2-fat 6-Spidergon that shows the fat topology gives the same performance but employing only a half of routers.

Taking into account router utilizations presented in Table II, it can be concluded that the optimal tradeoff between performance and router utilization is represented by 2-port fat-node Spidergons. These configurations bring a utilization of 87.5% of 8-port router with sufficient performance. Usage of 3-port slim-node Spidergons lead to lower router utilization, but can bring desired speed-up. On the other hand, 1-port slim-node Spidergon utilize routers in full, but this solution limit the performance dramatically. Finally, 3-port fat-node Spidergons require more complex routers and thus it is not attractive for NoC.

VI. CONCLUSIONS

We addressed the problems “is it better to use slim-node or fat-node Spidergon and what number of the internal ports should be implemented?” The lower bounds on time complexity cannot be mathematically derived for some Spidergon configurations. For this reason, an evolutionary algorithm was employed to find the lowest possible upper bounds and simultaneously corresponding conflict-free schedules that have not been known so far. The original contribution of the paper is an assessment of upper bounds of CCs on Spidergon network with fat-nodes and *k* internal ports. The assessments done with evolutionary algorithms are presented in Table III and Table IV.

Taking into account router ports utilization and number of interconnection links, fat-node spidergons seem to be more suitable for networks on chip. The performance degradation using fat nodes is very low and even higher for all-to-all scatter CC pattern.

The experimental results also indicate that CCs scale well with the number of internal ports. The only one exception is the AAS communication where the upper bound is given by interconnection network topology.

Considering limited resources on chip and router utilization, the most suitable Spidergon configurations use two PEs in one node, each connected by two internal ports to a shared router. This statement can be generalized to all 3-regular topologies (three output links), e.g., 3D hypercube.

Future research will be oriented toward optimizing CCs on Spidergons with more PEs in a node and also on complex comparison of slim and fat-node Spidergon. Next research will be oriented on investigation of the influence of port model on networks with higher number of external links like K-ring.

ACKNOWLEDGMENT

This research has been partially supported by the research grants "Safety and security of networked embedded systems applications", GA 102/08/1429 of Czech Science Foundation (2008-10), "Natural Computing on Unconventional Platforms", GP103/10/1517, Czech Science Foundation (2010-13), "Secured, reliable and adaptive computer systems", BUT FIT grant FIT-10-S-1 (2010) and the research plan "Security-oriented research in information technology", MSM 0021630528 (2007-13).

 ISBN:978-1-61208-114-4

REFERENCES

[1] D. N. Jayasimha, B. Zafar, and Y. Hoskote, "On-Chip Interconnection Networks: Why They are Different and How to Compare Them", Platform Architecture Research, Intel Corporation, 2006.

[2] A. Jantsch and H. Tenhunen, "Networks on Chip", Kluwer Academic Publ., Boston, 2003.

[3] Q. F. Stout and B. Wagar, "Intensive hypercube communication: prearranged communication in link-bound machines", in Journal of Parallel and Distributed Computing, vol. 10, pp. 167-181, 1990.

[4] J. Bruck, Ching-Tien Ho, S. Kipnis, E. Upfal, and D. Weathersby: "Efficient Algorithms for All-to-All Communications in Multiport Message-Passing Systems", in IEEE Transactions on parallel and distributed systems, vol. 8, no. 11, pp. 1143-1156, 1997.

[5] F. Khunjush and N. J. Dimopoulos, "Characterization of single-port and multi-port collective communication operations on the Cell BE processor", in IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, pp. 624-630, 2009.

[6] Y. Qian and A. Afsahi, "High Performance RDMA-based Multi-port All-gather on Multi-rail QsNet II", in High Performance Computing Systems and Applications, p. 3, 2007,

[7] J. A. Kahle, M. N. Day, H. P. Hofstee, C. R. Johns, T. R. Maeurer, and D. Shippy, "Introduction to the Cell Multiprocessor", in IBM Journal Research and Development, vol. 49, pp. 589-604, 2005.

[8] S. Williams, J. Carter, L. Oliker, J. Shalf, and K. Yelick, "Lattice Boltzmann Simulation Optimization on Leading Multicore Platforms", in International Parallel and Distributed Processing Symposium (IPDPS 2008), USA, pp. 1-14, 2008.

[9] M. Moadeli, and W. Vanderbauwhede, "A Performance Model of Multicast Communication in Wormhole-Routed Networks on-Chip", in IEEE International Symposium on Parallel&Distributed Processing, Italy, pp. 1-8, 2009, ISBN: 978-1-4244-3751-1.

[10] F. Karim and A. Nguyen. "An Interconnect Architecture for Networking Systems on Chips", in IEEE Micro, pp. 36-45, 2002.

[11] M. Moadeli, W. Vanderbauwhede, and A. Shahrabi, "Quarc: A Novel Network 0n-Chip Architecture", in International Conference on Parallel and Distributed Systems, pp. 705-712, 2008.

[12] Super Micro Computer, Inc. Homepage, Blade servers, URL: <http://www.supermicro.com/servers/blade/module/SBI-7226T-T2.cfm>, 2010.

[13] STMicroelectronics. URL: <http://www.st.com >, 2010.

[14] J. Duato and S. Yalamanchili, "Interconnection Networks – An Engineering Approach", Morgan Kaufman Publishers, Elsevier Science, 2003.

[15] N. Concer, S. Iamundo, and L. Bononi, "aEqualized: a Novel Routing Algorithm For The Spidergon Network On Chip", in Design, Automation and Test in Europe, DATE 2009, IEEE CS Press, pp. 749-754, 2009.

[16] M. Moadeli, A. Shahrabi, W. Vanderbauwhede, and M. Ould-Khaoua, "An Analytical Performance Model for the Spidergon NoC", in 21st International Conference on Advanced Networking and Applications (AINA'07), IEEE CS Press, pp. 1014 – 1021, 2007.

[17] J. Jaroš, "Evolutionary Design of Collective Communications on Wormhole Networks", Ph.D. thesis, Brno, CZ, 2010.

TABLE III. ACHIEVED UPPER BOUNDS ON THE NUMBER OF COMMUNICATION STEPS τ_{CC} (WH, K-PORT, DIRECT NETWORKS), SLIM NODES.

CC	OAB	AAB	OAS	AAS
6-gon_1p	3	5	5	5
6-gon_2p	2	3	3	3
6-gon_3p	2	2	2	3
8-gon_1p	3	7	7	7
8-gon_2p	2	4	4	4
8-gon_3p	2	3	3	4
10-gon_1p	4	9	9	9
10-gon_2p	3	5	5	7
10-gon_3p	2	4/3	3	6
12-gon_1p	4	12/11	11	12
12-gon_2p	3	7/6	6	9
12-gon_3p	2	4	4	9
14-gon_1p	4	14/13	13	15
14-gon_2p	3	8/7	7	13
14-gon_3p	2	6/5	5	12
16-gon_1p	4	16/15	15	18
16-gon_2p	3	8	8	17
16-gon_3p	2	5	5	17

TABLE IV. ACHIEVED UPPER BOUNDS ON THE NUMBER OF COMMUNICATION STEPS τ_{CC} (WH, K-PORT, DIRECT NETWORKS), FAT NODES.

CC	OAB	AAB	OAS	AAS
2fat_6-gon_1p	4	12	11	12
2fat_6-gon_2p	3	6	6	10
2fat_6-gon_3p	3	5	4	10
2fat_8-gon_1p	4	16	15	17
2fat_8-gon_2p	3	9	8	16
2fat_8-gon_3p	2	6	5	16
2fat_10-gon_1p	5	20	19	25
2fat_10-gon_2p	3	12	10	25
2fat_10-gon_3p	3	11	7	25
2fat_12-gon_1p	5	24	23	37
2fat_12-gon_2p	3	16	12	36
2fat_12-gon_3p	3	12	8	36
2fat_14-gon_1p	5	29	27	50
2fat_14-gon_2p	4	20	14	50
2fat_14-gon_3p	3	16	9	49
2fat_16-gon_1p	6	27	31	66
2fat_16-gon_2p	4	28	16	66
2fat_16-gon_3p	3	20	11	66

 ISBN:978-1-61208-114-4