Application-Specific Network-on-Chip with Link Aggregation

Ievgen Korotkyi, Oleksandr Lysenko
Department of Design of Electronic Digital Equipment
National Technical University of Ukraine “Kyiv Polytechnic Institute”
Kyiv, Ukraine
korotkiy.eugene@ieee.org, o.lysenko@kpi.ua

Abstract— **A method is proposed to reduce the hardware costs of networks-on-chip (NoC) with link aggregation by uneven distribution the number of physical links in aggregated logical connections. Created application-specific NoC, hardware cost of which is more than two times lower (by 65%), and the maximum operating frequency is by 41% higher than that of the network with homogeneous architecture. As a result of simulation in ModelSim investigated the transport latencies of the proposed solution.**

Keywords – network on chip; NoC; link aggregation; LAG; router; synthesis; application specific;

I. INTRODUCTION

Increasing complexity of systems-on-chip (SoC) reveals the problem of scalability for such ways to organize the communication subsystem, as a "common bus", "fully linked topology" and the connection type of "point-point" [1]. In order to increase scalability, throughput, and maximum operating frequency for communication between computational modules in VLSI is proposed to use the concept of a network-on-chip (NoC) [2]. The relevance of the given direction is confirmed by a large number of publications in international peer reviewed journals [3-5].

In contrast to the macro networks, the design of NoC imposes tighter restrictions on the number of hardware resources required to implement it, that makes impractical use large amounts of memory and as a consequence, buffering the entire packet before it sending. Application of small input buffers in NoC routers, while preserving the length of the packets, becomes possible in case of transmitting data with the aid of "wormhole" technology [6], when packets partitioned into atomic flow control units (flits) that are passed continuously one after another. The flits are advancing whenever possible, without waiting for arrival of followers, which permits to soften requirements to the buffer space volume. Such an approach increases the probability of head of the line blocking (HOLB) phenomenon, which can lead to decreasing in NoC throughput by up to 42% of its capacity [7].

To reduce the probability of HOLB and increase throughput of NoC, some researchers are using virtual channels (VC) [7]. Such an approach allows to slightly increase the saturation threshold of the network, but does not eliminate HOLB completely, since the virtual streams are multiplexed over a single physical connection between the routers, which adversely affects the transport delay and the saturation threshold of the NoC [8].

In [8] we proposed structural and hardware solutions for implementation of link aggregation (LAG) in NoC, when neighboring routers are joined with the aid of multiple physical links (PL). This approach eliminates HOLB and significantly (4 times or more) increases saturation threshold of the wormhole network. The disadvantage is a large amount of hardware resources required for implementation of the NoC with LAG compared to its classical wormhole counterpart .

In the present paper we investigate a method of reducing the hardware costs of the NoC with LAG by creating application-specific NoC with not uniform distribution the number of the physical links in aggregated logical connections (trunks), depending on the amount of network traffic that is passed through them.

The paper is organized as follows. Section II provides information about architecture of the NoC with LAG. In section III we consider the design of the router, which allows setting an arbitrary amount of PL in each trunk. Here the analysis of hardware cost for different configurations of such a router is performed. In the next section described and investigated a method for creating an application-specific NoC, in which the number of physical links inside each trunk proportional to the quantity of data streams flowing through it. In this section is also performed a comparative analysis of transport delays and hardware costs between created application-specific NoC and its homogenous counterpart. The last section contains conclusions and proposals for further investigations.

II. ARCHITECTURE OF THE NOC WITH LAG

In [7] Dally proposed to solve the problem of HOLB by associating the several buffers with each of the router’s inputs. Every of such buffers correspond to the virtual channel (VC). In case of blocking one of the input VCs, incoming packets can be transmitted through any another VC, which is free, bypassing the blocked one. This approach allows to increase the saturation threshold of the network by amount within 20% of NoC's capacity, but has such disadvantages as poor scalability and high hardware costs. Studies [9-10] have shown that increasing the number of VCs from two to four (by 100%) leads to growth of saturation threshold by only 2%, which characterizes poor scalability of the approach. According to authors of [8], this behavior is explained by the fact that the throughput of each of the router's inputs is distributed between the VCs, involved. With increasing number of VCs, the period between transferring flits of virtual flows is growing, which entails rising the transport latency and restriction the saturation threshold of the NoC.

As an alternative to the VC, for reducing HOLB, authors of [8] proposed to use link aggregation (LAG), where the topologically neighboring routers are connected by a set of physical links (PL), aggregated into logical channel (trunk). Through each trunk can be transmitted simultaneously N data packets, where N is the number of PL in trunk. This reduces the probability of HOLB and increases the throughput of the NoC. If we know spatial distribution of data flows in the network, it is possible to choose the number of PL for each trunk in such a way as to completely eliminate HOLB. Fig.1 illustrates how the aggregation of three links eliminates HOLB at the western input of router R_{11}.

The structure and principle of operation for the NoC router with LAG considered in detail in [8]. The results that are listed there show the increasing of saturation threshold by 100% with doubling the number of PL in trunks. According to [8], the saturation threshold for the NoC with LAG increased by 126% compared to Netmaker NoC [11] and by 152% compared to HERMES NoC [9] (both analogs are using VC technology). Compared with the classical wormhole architecture, without VC, the saturation threshold increased in four times (by 300%). The hardware costs of the NoC with LAG turned out comparable with requirements to VC NoC, thus the problem of reducing the amount of hardware resources needed to implement NoC with LAG is still open.

III. APPLICATION-SPECIFIC ROUTER FOR THE NOC WITH LAG

To reduce the hardware costs of the NoC with LAG we can tune the parameters of each router according to traffic passing through it. For example, if a particular input trunk takes *N* data streams, and through the rest of input trunks flowing $M<N$ data streams, it is inexpedient to choose the width of the each input trunk equal to *N* PL. Enough to do this only for the most "loaded" trunk and choose width of all remaining trunks according to the number of streams flowing through them. Similar conclusions are also valid for the trunks with outgoing traffic.

To make it possible to perform such an adjustment required a parameterized design that allows specifying an arbitrary number of PL for each of the router's trunk. Such a design is developed by the authors on the basis of the router from [8] and described using synthesizable subset of System Verilog. The block diagram of the proposed solution includes *N* input trunks, *N* output trunks, crossbar switch and the control module (Fig. 2). The number of PL in the *i*-th input trunk is MIN_i units, and in the *i*-th output trunk – $MOUT_i$ units. The control module contains block of XY routing, block of PL allocation and block of credit-based flow control. The operating principles of the referred blocks described in detail in [8]. The main difference between this solution and the router from [8] is the ability to define parameters MIN_i and $MOUT_i$. Technically it is implemented using two-dimensional dynamic array as a parameter of the router's module.

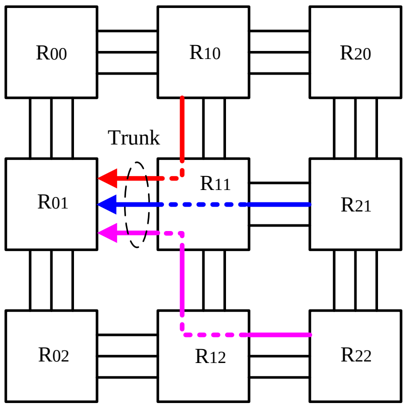

Figure 1. Eliminating HOLB in NoC by aggregation of 3 physical links

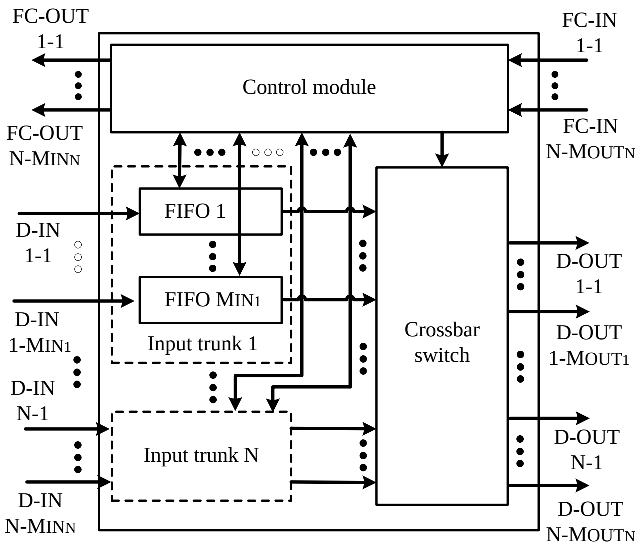

Figure 2. Block diagram of application-specific router for the NoC with LAG; FC – flow control; D-IN – data in; D-OUT – data out

In order to demonstrate the benefits of parametric adjustment of our application-specific router according to the traffic passing through it, we synthesized in Quartus II v.9.1 several configurations of device with different distribution of PL in it trunks. As target FPGA was selected EP4SGX230KF40C2 from Stratix IV family. Synthesis results are shown in table I. Although the router proposed by the authors can be used in NoC with any topology, the considered configurations of device are intended to use in network with mesh topology, so they contain five pairs of trunks. To ensure bidirectional data exchange, each pair is composed of input and output trunks. The four pairs are used for connection with the neighboring routers and marked in accordance with notation of the compass. The fifth pair is used to communicate with computational module and called as “local”. Numbers in cells of table I correspond to the amount of PL in trunks of mentioned pairs, but nothing prevents specify different values for each trunk in a pair.

Table I shows that in case of unequal number of data streams flowing through the trunks of the router, applying uneven distribution of PL allows considerably reduce the hardware costs of device (on average by 56%) and increase its maximum operating frequency by 19%, compared to homogeneous architecture comprising of 3 PL in each trunk.

TABLE I. SYNTHESIS RESULTS FOR CONFIGURATIONS OF ROUTER WITH LAG

Number of PL in each trunk of the specific pair					**Hardware costs**		
North pair	***East pair***	***South pair***	***West pair***	***Local pair***	***LUT***	***Reg***	***Fmax, MHz***
1	1	1	1	1	887	555	250
1	1	**3**	1	1	1581	833	195
1	**3**	1	**3**	1	2190	1098	185
2	1	**3**	2	1	2014	981	190
3	**3**	**3**	**3**	**3**	4533	1722	160

IV. METHOD OF CREATING APPLICATION-SPECIFIC NOC WITH LAG

Let us show how on the basis of proposed above parameterized router we can synthesize an application-specific NoC with heterogeneous architecture, in which the number of PL in each trunk is proportional to the amount of data streams flowing through it. This possibility follows from the fact that in NoC, connecting components of application-specific SoC, data flows can be distributed not uniformly that creates prerequisites for parametric optimization. The term "application-specific SoC" implies not a universal system designed to perform only one task (such as high-definition video codec [1]). Since state-of-the-art NoCs use static routing algorithms [5], if we know task graph of the system, mapping of computing cores onto the NoC and utilized routing algorithm, we can easily evaluate spatial distribution of traffic in network.

In current investigation, as the test SoC we selected the computing system, task graph of which is shown in Fig.3. The nodes of the graph correspond to the computing modules (CM) of the SoC. Edges of the graph correspond to data streams that flow from source CM to destination CM. The arrows indicate direction of data exchange. The weight of each edge corresponds to intensity of data transmission for the flow in number of data words per clock cycle. Through each edge, per clock cycle, can be transmitted no more than one informational word of size 16 bit.

As basic topology for our application-specific NoC we select matrix (mesh) method of connection [6]. Its advantages are good scalability, uniform utilization of hardware resources and the same order of length of connections between the network nodes. The latter contributes to the high clock frequency and reliability of the NoC [3]. The mapping of CM onto nodes of the NoC is shown in Fig.4. Routers marked with circles, and CM – with squares. Each router is addressed by two integer indices ij, where i specifies the horizontal position in the matrix and j – the vertical position (from upper-left node). The width of the flit in our NoC is equal to the length of informational word in SoC (16 bit). To determine the packets pathways is used simple dimension ordered XY routing. The compounds in Fig.4 correspond to full-duplex connections and composed of two oppositely directed trunks. To indicate the trunk we use the following notation: $T_{ij\to kl}$, where the pair of integers ij indexes the transmitter node, and kl – indexes the receiver node. Sometimes, instead of kl can be the name of CM.

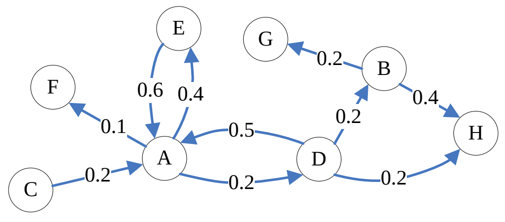

Figure 3. Task graph for the test system-on-chip

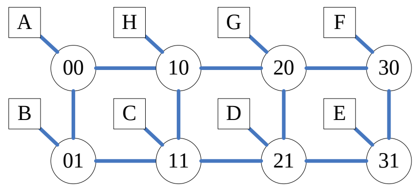

Figure 4. Block diagram of NoC that interconnects computing modules of test system-on-chip

The mapping of CM onto NoC (Fig. 4) is not an optimal and chosen so to simulate a large and unevenly distributed load on the nodes and connections of the NoC, occurring in application-specific SoC. Comparing task graph of the SoC (Fig. 3) and structure of the NoC (Fig. 4), we can conclude that the most "loaded" trunks are: $T_{21\to11}$,$T_{11\to01}$, $T_{01\to00}$ and $T_{00\to A}$. Utilization rates for these trunks exceed unity: $\gamma_{21\to11}=\gamma_{11\to01}=1{,}5$ and $\gamma_{01\to00}=\gamma_{00\to A}=1{,}3$. Under utilization rate γ we understood the sum of transmission intensities of all data streams flowing through the trunk. The value of $\gamma>1$ means that through the particular connection it is necessary to transfer more than one flit per clock cycle. Such a task isn't solvable for all of currently existing NoC architectures except of our proposed approach that emphasizes the advantage of employing LAG in NoC.

The task of creation of an application-specific NoC can be formulated as follows: to ensure that the transport delay of all data streams isn't greater than 150 clock cycles with minimum hardware costs. To solve this task we created a model of NoC in accordance with the structure shown in Fig.4 using a synthesizable subset of System Verilog. For each trunk in created NoC exists an opportunity to determine arbitrary

TABLE II. TRANSFER LATENCIES OF DATA FLOWS FOR DIFFERENT CONFIGURATIONS OF THE NOC WITH LAG

Number of PLs in the trunks of the NoC	**Transfer latencies of data flows, clock cycles (maximum values)**									
	A–>F	***A–>E***	***A–>D***	***B–>G***	***B–>H***	***C–>A***	***D–>H***	***D–>B***	***D–>A***	***E–>A***
All the trunks contain 1 PL	73	79	74	63	65	63	386	380	414	415
All the trunks contain 2 PL	73	79	74	57	59	33	172	170	173	72
All the trunks contain 3 PL	73	79	74	57	59	17	137	136	139	40
$T_{21\to11}=T_{11\to01}=T_{01\to00}=T_{00\to A}=3$ PL $T_{11\to10}=T_{10\to H}=2$ PL All another trunks contain 1 PL	73	79	74	57	59	17	139	138	140	40

number of PL. Technically it is implemented by means of a multidimensional array. Transport delays of data streams are obtained by simulation of the proposed NoC in ModelSim 6.5. The simulation process is as follows. Computational modules that are connected to the routers with the intensity of λ injected into the network the flits of data packets. The length of each packet is five flits and the values of λ are shown in Fig.3. During the simulation run, each CM creates 2100 packets (10 500 flits). Simulation ends when all the generated packets reach their destinations. To ensure transition of the NoC in steady state, the receiving side of CM starts calculation of the transport delays after registering the first 100 packets. To estimate the hardware costs of different configurations of our application-specific NoC we synthesized it System Verilog model in Quartus II v.9.1 with different values of design parameters. As target FPGA was selected EP4SGX230KF40C2 from Stratix IV family.

First consider the case when all trunks of the NoC contain one PL. This architecture is homogeneous and therefore non application-specific. Delays of data streams for such configuration of a NoC are presented in the first row of Table II and its hardware costs is shown at the beginning of Table III. Note that in Table II are given only maximum delays, while their average values can be several times lower. As shown in Table III the synthesis of this configuration implies minimum hardware costs, but the requirements to transport delay for the flows D–>H, D–>B, D–>A and E–>A are not met.

Increasing the number of PL in trunks up to two allows to achieve acceptable value of transport latency for the stream E–>A and reduce delays of the remaining "problem" streams on average by 55% bringing them to 172, 170 and 173 clock cycles. The hardware costs of the NoC in this case increases by 125%. From the Table II it follows that for a homogeneous structure of the NoC ensuring acceptable delays for all of the streams is possible in configuration with three PL per trunk. Such a decision would be the most hardware expensive, requiring for its realization by 300% more FPGA resources than the initial configuration.

In the fourth row of Table II presented the configuration of application-specific NoC that meets requirements to transport delays of all data streams. The number of PL in trunks was determined using the following method proposed by the authors. Initially, all the trunks are initialized by a single PL. During the first stage are estimated the transport delays of traffic streams and utilization rates of the trunks. Estimation is carrying out with the aid of simulation in ModelSim. Then is performed the normalizing of the obtained utilization rates dividing them by the number of streams flowing through corresponded trunks. As candidate to incrementing the quantity of PL is selected the aggregated connection corresponding to maximum value of the normalized utilization rate on condition that such a trunk transmits the data stream that does not comply with the requirements to transport delay. Described actions are repeated iteratively until the delays of data streams will not meet the specified requirements. Thus, at each iteration of our method the quantity of PL increments only for the most loaded trunk. In present study the method described above was implemented manually. The synthesis results for created application-specific NoC are given in the last row of Table III and show that the hardware costs has been reduced by 65% (more than twice) compared with homogeneous NoC containing 3 PL per each trunk, and similar in traffic delays. The maximum clock frequency of application-specific NoC has grown by 41%.

TABLE III. HARDWARE COSTS FOR CONFIGURATIONS OF NOC WITH LAG

Number of PLs in the trunks of the NoC	***LUT***	***Reg.***	***Fmax, MHz***
All the trunks contain 1 PL	4 410	3 080	231
All the trunks contain 2 PL	11 048	5 704	180
All the trunks contain 3 PL	21 176	8 976	145
$T_{21\to11} = T_{11\to01} = T_{01\to00} = T_{00\to A} = 3$ PL $T_{11\to10} = T_{10\to H} = 2$ PL All another trunks contain 1 PL	6 535	4 038	205

V. CONCLUSIONS

In this paper we proposed and investigated the construction of application-specific router for a NoC with LAG, which allows setting an arbitrary amount of PL in each trunk of the router. It is shown that in case of uneven quantity of data streams flowing through aggregated logical connections of device, such an approach can significantly (by 56%) reduce its hardware cost.

We also proposed and investigated a method of creating an application-specific NoC with LAG in which the amount of PL in each trunk proportional to quantity of data streams flowing through it. Using proposed method we developed the application-specific NoC, hardware costs of which more than twice lower (by 65%) compared with its homogeneous counterpart, and the maximum clock frequency is 41% higher.

The vector of further research aims at creating an algorithm that automates the proposed method of synthesis an application-specific NoC with LAG.

REFERENCES

[1] H.G. Lee, N. Chang, U.Y. Ogras, R. Marculescu, “On-chip communication architecture exploration: A quantitative evaluation of point-to-point, bus and network-on-chip approaches,” *ACM Trans. on Design Automation of Electronic Systems*, vol. 12, no. 3, pp. 1-20, 2007.

[2] W. Dally, B. Towles, “Route packets, not wires: on-chip interconnection networks,” *in Proceedings of the 38th annual Design Automation Conference*, Las Vegas, USA, June 2001, pp. 684-689.

[3] T. Bjerregaard, S. Mahadevan, “A survey of research and practices of network-on-chip,” *ACM Comp. Surveys*, vol. 38, no. 1, pp. 1-51, 2006.

[4] D. Atienza, F. Angiolini, S. Murali, A. Pullini, L. Benini, G.D. Micheli, “Network-on-chip design and synthesis outlook,” *Integration The VLSI journal*, vol. 41, no. 3, pp. 340-359, 2008.

[5] R. Marculescu, U.Y. Ogras, L.S. Peh, N.E. Jerger, Y. Hoskote, “Outstanding research problems in NoC design: system, microarchitecture, and circuit perspectives,” *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 1, pp. 3-21, 2009.

[6] W.J. Dally, “Performance analysis of k-ary n-cube interconnection networks,” *IEEE Trans. Comput.*, vol. 39, no. 6, pp. 775-785, 1990.

[7] W.J. Dally, “Virtual-channel flow control,” *IEEE Trans. on Parallel Distrib. Syst.*, vol. 3, no. 2, pp. 194-205, 1992.

[8] I. Korotkyi, O. Lysenko, “Hardware implementation of link aggregation in networks-on-chip,” *in Proc. of World Congress on Information and Communication Technologies*, Mumbai, India, Dec.2011, pp.1112-1117.

[9] A. Mello, L. Tedesco, N. Calazans, F. Moraes, “Virtual channels in networks on chip: implementation and evaluation on Hermes NoC,” *in Proceedings of 18th Symposium Integrated Circuits and System Design*, New York, USA, 2005, pp. 178-183.

[10] R. Mullins, A. West, S. Moore, “Low-latency virtual-channel routers for on-chip networks,” *in Proceedings of 31-th International Symposium on Computer Architecture*, Munich, Germany, June 2004, pp. 188-197.

[11] Netmaker, http://www-dyn.cl.cam.ac.uk/~rdm34/wiki