A New High-Performance Scalable Dynamic Interconnection for FPGA-based Reconfigurable Systems

Slaviša Jovanović, Camel Tanougast and Serge Weber
Laboratoire d'instrumentation et d'électronique de Nancy
Université Henri Poincaré
Vandoeuvre lès Nancy, France
slavisa.jovanovic@lien.uhp-nancy.fr

Abstract

Networks on chip (NoCs) present viable interconnection architectures which are especially characterized by high level of parallelism, high performances and scalability. The already proposed NoC architectures in literature are mostly destined to System-on-chip (SoCs) designs. For a FPGA-based reconfigurable system, the proposed NoCs are not suitable. In this paper, we present a new high-performance interconnection approach destined for FPGA-based reconfigurable system. Our proposed NoC is based on a scalable communication unit characterized by its particularly architecture, an arbitration policy based on the priority-to-the-right rule and high performances. We present the basic concept of this communication approach and we prove its feasibility on examples through the simulations. Implementation results are also detailed.

1 Introduction

FPGA-based systems emerge as a new computation paradigm. The main characteristics of these systems are high level of modularity and flexibility. Several processing units (i.e. IPs) can be implemented at a given time and dynamically replaced (entirely or partially) by means of the reconfiguration [8, 9]. To design a complex system in the FPGA technology, the system which is built of several processing units, especial attention must be paid on communication medium. Actually, the currently used techniques do not take into account all advantages of the FPGA technology, because the system's processing units communicate via bus-based macros. To benefit from all advantages of this reconfigurable technology, a new communication paradigm based on the network-centric approach must be developed. Already presented NoCs are not suitable and appropriated to the FPGA-based systems. The major problem arises when the components need to be dynamically placed on the chip [10].

This paper proposes a new communication architecture called QNoC designed for the FPGA-based systems. The QNoC represents a scalable modular dynamic interconnection whose structure can be reconfigured and adapted depending on the computation needs of the system. In the QNoC, the reconfigurable resources can be used to implement either the routers in the network or processing elements. The QNoC is based on a packet-switched network of intelligent routers called *Q-switches*. These routers are characterized by their particular architecture, *store-and-forward* switching technique, arbitration policy and their specific connection with other switches or with processing elements.

This paper is organized as follows. Section 2 gives an overview of related works on the main network-on-chip architectures. Section 3 describes our communication approach and details its basic element, *the Q-switch*. The Q-switch's architecture and the possible uses in several network structures for different communication needs are both presented. The simulation results are given in Section 4. Implementation results on Xilinx Virtex FPGA technology and performance evaluation are presented in Section 5. Some conclusions and future work are both given in Section 6.

2 Related work

The complexity of a network is described by two parameters: topology and routing algorithm. According to the topology, the NoCs can be classified in two classes: *static* and *dynamic*. Several static NoCs are presented in literature. Among them we distinguish: SPIN [1], CLICHÉ [2], Torus [3], folded 2D Torus [4], Octagon [5, 6] and BFT [7]. The fixed structure of the network is the main reason that the static NoCs are too inflexible for the communication among

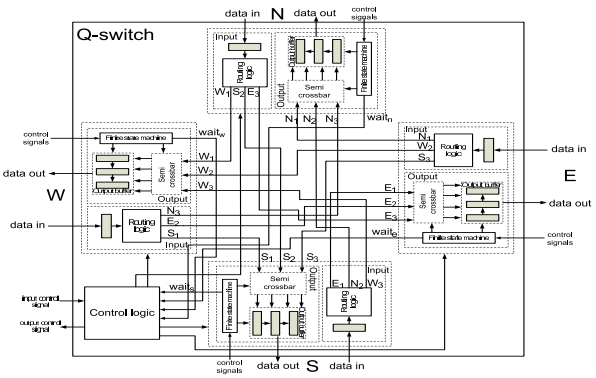


Figure 1. Q-switch's architecture

reconfigurable systems' modules which need a changing network.

The dynamic NoCs are mediums supporting communication among modules which are dynamically placed on a run-time reconfigurable device. Among them we find the DyNoC and CuNoC communication approaches [10, 13]. The dynamically placed modules in DyNoC "cover" the routers which were at their place before placement. The covered routers are deactivated and cannot be used for the communication between modules. However, they can be used as modules' additional logics. The routers reactivate once the function of dynamically placed module has finished and the module is removed from the chip. Significant occupied area by one network element NE (router) [10] and the small area ratio PE/NE (processing/network element) are the main inconveniences for the FPGA-based reconfigurable devices. Another interconnection supporting dynamic placement of module at run-time is the CuNoC [13]. This communication approach is characterized by low area overhead of its basic element, dynamic adaptive routing algorithm and specific connection with the processing elements. The employed routing algorithm in some situations leads to non-minimal paths between processing elements in communication. That causes a need for packet-reordering at destination node. Another drawback of this approach are low performances.

3 QNoC

We propose a new NoC-based communication approach called QNoC for FPGA-based reconfigurable devices. This approach allows communication between modules dynamically placed at the run-time on a chip. The QNoC represents packet-switched network of intelligent independent routers called Q-switches. The Q-switch is characterized by a particular architecture, *store-and-forward* switching technique, an arbitration policy based on the priority-to-the-right rule and specific connection with the processing elements (PEs).

3.1 Message structure

The message used in QNoC is composed of fixed number of packets. The number of packets N_p in a message is not fixed by the network ($N_p \geq 1$). Packet's format is depicted in Figure 2.

Figure 2. Packet's format

The first field denotes the destination address whose size depends on a total number of Q-switches that can be implemented in a reconfigurable area. The second field contains a data information for all packets in the message except for the first one. The first packet in the data field sends the message length information. Thereafter, by the packet's size we consider only the data field's size without destination address field.

3.2 Q-switch

The basic element of our communication approach is a *Q-switch*. The Q-switch's architecture is depicted in Figure 1. The Q-switch can be either used as a stand-alone element to pass the messages between up to 4 processing elements or can be used as a part of the network. The Q-switch has one input register at each port. All packets pass through these input registers. Once the packets are received and stored in these input registers, the blocks named "Routing logic" compute their next directions. According to their next directions, the "Routing blocks" pass on the packets to the "Output logics". It could happen that several packets (up to 3) take the same output direction. In this case, they have to share the same output link. To manage their sending out to a neighbouring Q-switch tagged by the routing logics an arbitration policy have to be adopted. The arbitration logic manages the priority of packet sending. The Q-switch uses an arbitration policy based on the rule of priority to the right. This mechanism is presented in Figure 3. The arbitration policy is built individually for each port of the Q-switch. For example, the port *East* can take only the

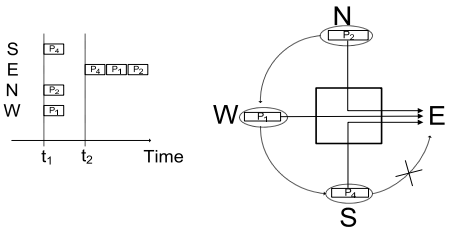


Figure 3. Arbitration policy is based on the priority-to-the-right rule

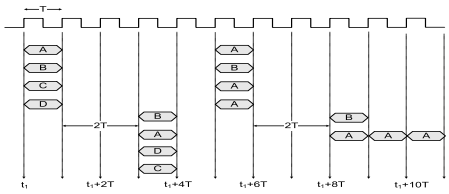


Figure 4. Illustration of the Q-switch latency on example of three received packets

packets arriving from directions *North*, *West* and *South*. If we apply the priority-to-the-right rule to these directions, as a result we have that the packets from *South* direction have the highest priority, than those from *West* and at the end the packets from *North* direction. The same procedure is applied to all other ports. The schedules fixed that way are used in output logics.

The Q-switch uses a store-and-forward switching mode [11]. That means the packet cannot be forwarded until it has been completely received. This fact introduces an additional latency per switch. In our case, the latency is 2 clock periods. Figure 4 presents 2 cases. In the first one, the switch at $t + T$ moment receives 4 packets with 4 different destination addresses. After the latency period of 2 clocks ($t + 3T$), all 4 packets are directed to the links according to their destination addresses. In the second case, at the $t + 5T$ moment, the switch receives 4 packets where 3 have the same destination address. After the latency period at $t + 7T$, the packet having the highest priority among three takes the first the output link. Two others leave the switch in two next successive clock cycles. The QNoC does not use commonly used *wormhole* switching technique because the switches using this technique can sometimes remain blocked till whole message is not transferred to destination node. This fact is not appropriated for QNoC approach, because at each moment all routers must be available for replacing with a processing element if a processing element insertion demand occurs.

3.3 Routing algorithm

In classic XY routing algorithm, the packet is firstly routed in X direction and secondly in Y direction until the final destination. This routing algorithm is not suitable for the dynamically changing networks, because dynamically placed computing module could make maintained communication between two modules worse, or even impossible.

The Q-switch's routing policy is based on modified adaptive XY routing algorithm. The packets are firstly routed in X direction, and secondly in Y direction as in the classic XY routing algorithm. If during the X routing, a packet encounters a processing element, it will surround it by changing temporarily from X to Y routing. For doing this, the Q-switch uses its coordinates and the input control signals that indicate to it the nature of its neighbours (Q-switch or PE). The ping-pong game effect could not arrive in QNoC, because the packets cannot take the directions of arrival [10]. That means if packet arrives from west side, it cannot return to it. This restriction simplifies a bit the routing logic at each port. Moreover, in order to avoid deadlock situations some turns are forbidden. The used routing algorithm allows packets to take always the same path between 2 PEs in the situations where in the meantime between two packet sendings there are no changes in the network. This fact avoids the out-of-order situations at destination node for all packets sent from a source node. However, the received packets at a destination node could be interleaved, if there are several source nodes which send at the same time packets to it.

3.4 Output logic

Output logic consists of a semi crossbar, an output buffer and a finite state machine, Figure 5. The semi crossbar switches between 3 inputs and 4 outputs. The inputs are organized according to the priorities. The input with the number 1 has the highest priority, the input with the number 2 has lower priority than the first one and the input with the number 3 has the lowest priority. The first input of the crossbar can be switched only to the first two outputs. The first output of the crossbar is, if the neighbouring Q-switch is not occupied, linked directly to the output link toward adjacent Q-switch. The second one is linked to the first register of the output buffer, the third one to the second register and the last one to the third register of the output buffer. The second input of the crossbar can be switched to first three outputs and the third input can take all crossbar's outputs.

The output buffer consists of three packet-size registers. The registers are used to store the packets. In the case where more than one packet occurs at the selected output, the packet having the highest priority takes, if there is not occupancy signal (occ_{in}), the direct output link whereas other packets (up to 2) are stored in the output buffer. The output buffer is also used in the case where output link toward neighbouring Q-switch is occupied which is indicated with the occupancy signal. In this case, in output buffer it can be

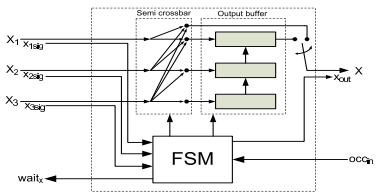


Figure 5. Output logic of the Q-switch

stored up to 3 packets.

The Finite State Machine (FSM) takes care of managing all control signals and allows avoiding of packet collision situations. The FSM of the Q-switch output logic informs also a central logic about occupancy of the neighbouring Q-switch to which its output link is connected ($wait$ signal).

3.5 Control logic

The controls of the outputs are placed in the output logics. These logics manage linking of the routing logics' outputs with the output links. For the cases where there are no occupancy signals from neighboring switches, these logics are sufficient to manage interconnection between all inputs and outputs of the Q-switch.

The Q-switch has at each port input signals which give information of the occupancy of its direct neighbouring Q-switches and output signals that indicate to them its own occupancy. The occupancy situations occur when more than one packet appears at an output logics' input ports. Because all packets (up to 3) choose the same output direction, their delivering to the neighbouring Q-switches is not possible at one clock cycle. The control logic set the output control signal toward Q-switches from which directions the packets have been arrived to inform them that its own Q-switch is temporary unavailable. Once all received packets are transferred, these signals will be reseted.

The central control logic also manages storing of the packets that cannot be transferred to routed directions because of occupancy of neighbouring Q-switches. In this case, all packets that cannot be transferred remain blocked in Q-switch till the occupancy signals are set. The control logic through the output control signals informs neighbouring Q-switches from which directions the blocked packets are arrived that it cannot accept other packets.

3.6 Conditions of placement of modules in mesh QNoC

The starting point of our approach is the valid placement of Q-switches on the reconfigurable area of the chip. They are 3 successive phases of building a mesh QNoC and placement of modules on it. After having placed the network of routers, the next step consists in the placement of the modules. Each module having the area size lower than the Q-switch's size replaces one Q-switch. For all other modules having the area size greater than the Q-switch's size, we replace a zone of several Q-switches having total area size equal or greater than the module's area size. The module which covers at the time of the placement one or several Q-switches inherits all their addresses. For instance, the module which covered 4 Q-switches at the placement time has 4 addresses and more than 4 access points to the network.

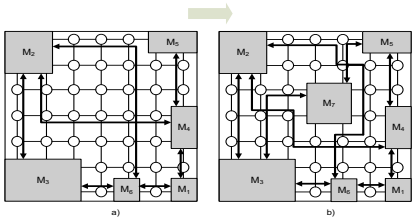


Figure 6. Dynamic placement of modules in QNoC

The QNoC routing algorithm allows the dynamic placement of modules because the path between the source and destination computes at run-time. In fact, for different packets of a message, the path between the source and destination is the same if in meantime a module is not dynamically placed on the communication path between the modules. If we place a module in the middle of the network between two modules having important intercommunication, their communication will not deteriorate. Indeed, the packets sent after the placement of the new module will consider it as an obstacle and will get round it. Let us suppose that we have the following situation: a 7×7 QNoC and 6 computing modules placed on it. Each computing module carries out a function and needs some information exchange with the other modules placed on the chip. After a certain time, communication is established within this network (see Figure 6a). At a given time, a new function demand occurs. This leads to that a new computing module must be inserted into the network. A reconfiguration area controller which decides on which place of the network a module will be inserted, carries out an analysis of the available free reconfigurable area (area which is not covered with other modules) and it decides which Q-switches will be replaced with the new computing element. It "informs" all the Q-switches concerned about the action that will be carried out in order to allow them to empty their buffers and to change their status on "module". The rest of the network will further consider the Q-switch(es) marked to replace as an obstacle and could not take the way(s) containing these Q-switches. This situation is illustrated in Figure 6b. However, a number of rules must be defined and respected in order to ensure an efficient communication between all modules placed statically at compile time or dynamically at run-time on the chip. QNoC use the same set of rules as the CuNoC [13].

The Q-switch has 4 input and output ports and it does not have a link to a processing element. This fact allows mixing of the Q-switches and modules in the network. The Q-switches are well suited for the 2D mesh topology. In Figure 7 are presented some examples of the QNoC structures based on the 2D mesh topology of different sizes (3×3, 4×4 and 5×5). These structures have the maximal

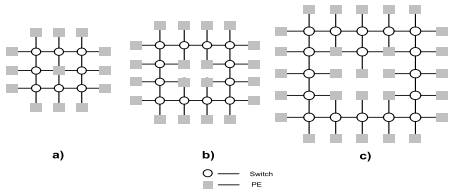


Figure 7. A maximal number of modules that could be connected to a) 3 × 3 b) 4 × 4 and c) 5 × 5 QNoC

number of modules that can be connected to a m × n 2D mesh QNoC. The maximal number of modules that could be connected to the QNoC is calculated for the minimal module's area size. The minimal module's area size is equal to Q-switch's area size. The maximal number of modules of minimal size (size of one Q-switch) which can be connected to 2D m × n mesh QNoC is described with the following equation:

$$N_{mod_{max}} = 3m + 4n - 8 + trunc\left[\frac{(n-4)(m-3)}{2}\right] \quad (1a)$$

$$N_{Q-switch} = m * n - N_{mod_{max}} \quad (1b)$$

where $N_{Q-switch}$ is the number of Q-switches left from initial m x n network. The term $trunc\left[\frac{(n-4)(m-3)}{2}\right]$ is equal to 0 for n and $m \leq 3$.

4 Simulation results

To validate functionally the QNoC communication approach, we implemented an 5×5 QNoC and connected 20 simulation modules "around" this network in the manner described in previous section. The developed simulation modules can generate random-traffic pattern and can be parametrized concerning the interval of the traffic generation. Random traffic pattern means that the simulation modules send packets randomly to other simulation modules. The time period between the sending time of two packets was uniformly distributed between 5 and 10 cycles. In the following figures the x-axis and the y-axis represent the switches in the topology whereas the z-axis represents the time a switch is busy divided by the simulation time.

Figure 8a depicts the network load of a mesh under random traffic for the case of 20 simulation modules connected to the network. All simulation modules in this case are connected all around the 5×5 QNoC. For this case, neither of the switches of the 5×5 network is not replaced with a simulation module. The simulation result shows that the traffic is not uniformly distributed across the entire network. The most traffic cross the switches in the corners and at the edges of the network because all source and destination nodes are connected to the switches in this area of the network. The switches in the middle of the area are less utilized.

Figure 8b depicts the network load of a mesh under random traffic for the case of 21 simulation modules connected to an 5×5. 20 simulation modules are connected all around the 5×5 QNoC whereas the 21th simulation module replaced the switch at position $(2, 1)$. In this case, the simulation module at $(2, 1)$ position does not generate the traffic, it accepts only the packets sent from other simulation modules. The simulation result shows that the traffic gets around this position and as in the first case it is mostly distributed in the corners and at the edges of the network.

The last simulation uses the simulation topology used for the second simulation case. The unique difference is that the simulation module at position $(2, 1)$ generates the traffic. This results are depicted in Figure 8c. It can be seen that the traffic is more dense around the position $(2, 1)$ which is due to the additional traffic from the simulation module at this position. The two last simulations show that the other modules continue to intercommunicate despite the placing of a module in the middle of the network.

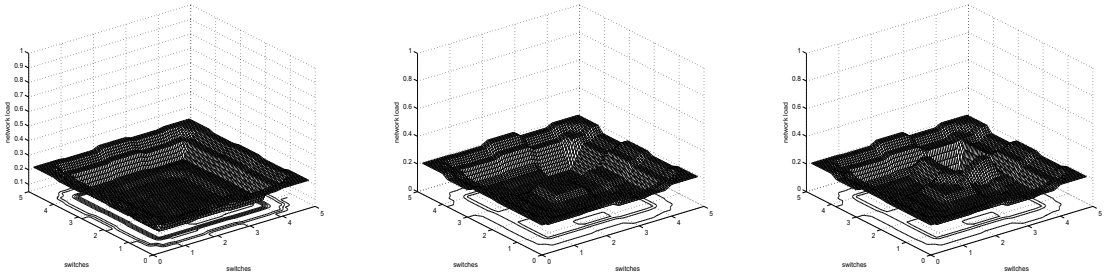
5 Implementations results and performance evaluation

We have synthesized and implemented the Q-switches of various data format in Xilinx Virtex IV technology. These results are given in Table 1 in CLB slices for area occupation and in MHz for maximum operating frequency f. It can be seen that the 8-bit Q-switch takes about 365 CLB slices and has a maximum operating frequency in Virtex IV technology of 249.3 MHz. The implementation results are only given in terms of CLBs, because other FPGA ressources (BRAMs, ...) are not needded for the Q-switch implementation.

The maximum throughput T_{max} of an n data-bit-width Q-switch at frequency f to which the maximum number of processing modules are connected (4 PEs) is given by the

Table 1. Q-switch Statistics

	Q-switch	
data width	CLB Slices	Virtex IV f [MHz]
8 bit	365	249.3
16 bit	434	249.3
24 bit	504	249.3
32 bit	598	249.3



(a) 20 simulation PEs are connected to the network

(b) 21 simulation PEs are connected to the network, all generate and receive the traffic except the 21th at $(2, 1)$ which only receives

(c) 21 PEs are connected to the network, all generate and receive the traffic

Figure 8. Simulation results of a traffic distribution in an 5 $\times$ 5 QNoC under random-traffic pattern

following equation:

$$T_{max} = 4 * n * f \tag{2}$$

For example, for a 8 data-bit-width Q-switch to which 4 processing modules are connected at 100 MHz we have the throughput of 3.2 Gbps.

6 Conclusion and future work

In this paper, we proposed a new dynamic interconnection for run-time reconfigurable modules in FPGAs. We presented the basic concept of this communication approach, its basic router's architecture, possible topologies and structures mixing the network of Q-switches with the processing elements (PEs). Our communication approach represents a scalable network structure which can be used either as a stand-alone unit for communication between up to 4 PEs or as a part of a network. The main advantages of the QNoC are high performances and the possibility of dynamic placement of modules at run-time. From performance evaluation and implementation results we conclude that the QNoC presents a good compromise between the logic area and operating frequency. The QNoC have been tested on 2D mesh topology. Other topologies than mesh will be considered as well as implementation of a given application on the QNoC.

References

[1] P. Guerrier and A. Greiner: *A Generic Architecture for On-chip Packet-Switched Interconnections*, Proc.Design and Test in Europe (DATE), pages 250-256, mar 2000.

[2] A. Andiahantenaina, A. Grenier: *Micro-network for SoC: implementation of a 32-port SPIN network*, in: Design Automation and Test in Europe (DATE'03), Pages 1128-1129, March 2003.

[3] T. Marescaux, A. Bartic, D. Verkest, S. Vernalde and R. Lauwereins: *Interconnection Networks Enable Fine-Grain Dynamic Multi-Tasking on FPGAs*, Proc. of 12th International Conference, FPL, Montpellier, France, sep 2002.

[4] W. J. Dally and B. Towles: *Route Packets, Not Wires: On-Chip Interconnection Networks*,Proc. Design Automation Conf. (DAC), pages 683-689, 2001.

[5] F. Karim et al: *An Interconnect Architecture for Networking Systems on Chips*, IEEE Micro, volume 22, number 5, pages 36-45, mar 2002.

[6] F. Karim et al: *On-chip Communication Architecture for OC-768 Network Processors*,in: 38th Design Automation Conference (DAC'01), Pages 678-683, June 2001.

[7] P. P. Pande, C. Grecu, A. Ivanov and R. Saleh: *Design of a Switch for Network on Chip Applications*, Proc. Int. Symp. Circuits and Systems (ISCAS), pages 217-220, volume 5, may 2003.

[8] S. A. Guccione and D. Levi: *The Advantages of Run-time Reconfiguration*, In John Schewel et al editors, Reconfigurable Technology: FPGAs for Computing and Applications, Proc. SPIE 3844, pages 87-92, Bellingham, WA, sep 1999.

[9] P. Lysaght, J. Dunlop: *Dynamic reconfiguration of FPGAs*, in: W. Moore, W. Luk, (Eds), More FPGAs, Proceedings of the International Workshop on Field-programmable Logic and Applications, pages 82-94, 1993.

[10] C. Bobda, A. Ahmadinia, M. Majer, J. Teich, S. Fekete and J. van der Veen: *DyNoC: A Dynamic Infrastructure for Communication in Dynamically Reconfigurable Devices*, International Conf. on FPL, aug 2005.

[11] L. M. Ni, P. K. McKinley: *A Survey of Wormhole Routing Techniques in Direct Networks*, IEEE Computer Society Press, Volume 26, Issue 2, Pages 62-76, February 1993.

[12] C. Bobda and A. Ahmadinia: *Dynamic Interconnection of reconfigurable modules on reconfigurable devices*, Design & Test of Computers, IEEE, volume 22, issue 5, pages 443-451, sep 2005.

[13] S. Jovanović, C. Tanougast, C. Bobda and S. Weber: *A Scalable Dynamic NoC for Dynamically Reconfigurable FPGAs*, aug 2007,FPL, Amsterdam, Netherlands

[14] S. Jovanović, C. Tanougast, C. Bobda and S. Weber:*A Scalable Dynamic Infrastructure for Dynamically Reconfigurable Systems*, ReCoSoC07, Montpellier, France