CuNoC: A Scalable Dynamic NoC for Dynamically Reconfigurable FPGAs

Slavisa Jovanovic, Camel Tanougast, Serge Weber, Christophe Bobda

CuNoC: A scalable dynamic NoC for dynamically reconfigurable FPGAs

Conference Paper · September 2007

DOI: 10.1109/FPL.2007.4380761 · Source: IEEE Xplore

CITATIONS
36

READS
153

4 authors:

Slavisa Jovanovic
University of Lorraine
43 PUBLICATIONS **223** CITATIONS
SEE PROFILE

Camel Tanougast
University of Lorraine
161 PUBLICATIONS **825** CITATIONS
SEE PROFILE

Serge Weber
University of Lorraine
111 PUBLICATIONS **464** CITATIONS
SEE PROFILE

Christophe Bobda
University of Florida
158 PUBLICATIONS **1,575** CITATIONS
SEE PROFILE

CuNoC: A Scalable Dynamic NoC for Dynamically Reconfigurable FPGAs

S. Jovanovic, C. Tanougast and S. Weber

LIEN
University Henri Poincaré - Nancy 1
54506 Vandoeuvre lès Nancy, France
e-mail: slavisa.jovanovic@lien.uhp-nancy.fr

C. Bobda

Department of Computer Science
University of Kaiserslautern
67653 Kaiserslautern, Germany
e-mail:bobda@informatik.uni-kl.de

ABSTRACT

In this article, we present CuNoC, a new paradigm for intercommunication between modules dynamically placed on a chip for FPGA-based reconfigurable devices. The CuNoC is based on scalable communication unit called CU which allows the simultaneous communication between several processing elements placed on the chip. We present the basic concept of this communication approach, its main advantages and drawbacks with regards to the other main NoC approaches already proposed.

1. INTRODUCTION

Nowdays, the most frequently used on-chip interconnect architecture is the shared medium arbitrated bus. The advantages of the shared-bus architecture are simply technology, low area cost and extensibility. On the other hand, the relatively long bus line increase significantly propagation delay which may result in unsuitable functioning of the system.

An alternative to the bus-based interconnection architectures are a network-centric approaches [1, 2]. The NoCs are characterized by explicit parallelism, high modularity and high performances. These approaches are especially suited for high-performance parallel computing. Several papers address this subject [3, 4, 5, 6, 7, 8].

In the last decade, the FPGA-based systems emerge as a new computation paradigm. Several processing units can be implemented at a given time and dynamically replaced (entirely or partially) by means of the reconfiguration. The designing of a complex FPGA-based system which takes all benefits of FPGA technology, needs an adapted NoC. Already presented NoCs are not suitable and appropriated to the FPGA-based systems. The major problem arises when components need to be dynamically placed on chip [9].

This paper proposes a new intercommunication architecture called CuNoC designed for the FPGA-based systems. The CuNoC represents a scalable modular dynamic communication infrastructure. The basic element of the CuNoC communication medium is the router called *Communication Unit - CU*. It is charachterized by novel arbitrary policy based on priority-to-the-right rule, specific architecture and low-overhead implementation compared with other already presented routers.

This paper is organized as follows. Section 2 gives an overview of related work on the main NoC architectures. Section 3 describes the CuNoC, our communication approach, and details its basic element, the communication unit (CU).The CU's architecture and the possible uses in several network structures for different communication needs are both presented. The simulation and implementation results on Xilinx Virtex-II FPGA technology are given in Section 4. The conclusion and future work are given in Section 5.

2. RELATED WORK

Several NoCs are presented in literature. Among different interconnection NoC architectures we distinguish : SPIN [3], CLICHÉ [4], Torus [5], Octagon [6] and BFT [7]. Each of these architectures is characterized by different topology, routing algorithm, number of processing elements (PEs) which may be connected and performances (i.e. transfer data rate) [8]. In these static NoCs, the PEs are placed in rectangular tiles on the chip and communicate with each other via fixed network structure. These fixed networks are composed of intelligent switches. Each switch is connected to the tile with an interface over input and output ports. The output port is used to send packets from the tile whereas the input port delivers packets to the tile. Each switch is characterized by used *switching technique*: Circuit Switching, Packet Routing and Wormhole Routing [8].

A static NoC presents a viable communication infrastructure with many advantages in regard to bus-based platforms. However, it is too inflexible for communication in FPGA-based systems which need a changing network.

The DyNoC was presented in [9], [10] as a medium supporting communication among modules being dynamically placed on a run-time reconfigurable device. The dynamically placed modules in the 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 after removing of the dynamically placed modules. That makes the network dynamic.

The DyNoC is a logical choice for dynamic changing networks. On the other hand, for communication between PEs in the FPGA-based systems, despite all the advantages, it is not very suited communication medium. Significant occupied area by one router (network element - NE) [9] and the small area ratio PE / NE (processing/network element) are the main inconveniencies for the FPGA-based reconfigurable devices.

3. CuNoC

We propose a new communication approach called CuNoC for FPGA-based reconfigurable devices. This communication approach

allows communication between modules dynamically placed on the chip. The CuNoC represents packet-switched network of intelligent independent routers called *Communication Unit - (CU)* . The CU is characterized by novel arbitrary policy based on priority-to-the-right rule, unique architecture and specific connection with processing elements.

3.1. Communication Unit (CU)

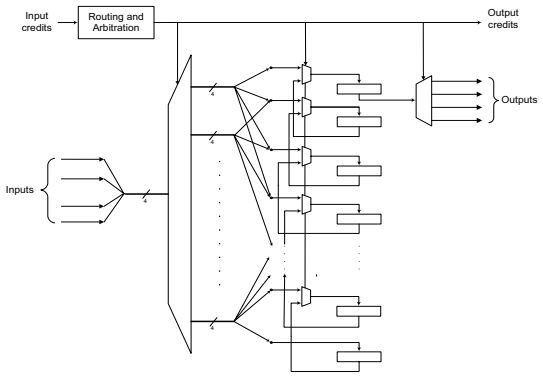

Fig. 1. CU's architecture

The basic element of the CuNoC is the CU - *the Communication Unit*. Its architecture is presented in figure 1.The CU does not have input channel buffers which are used to store temporarily the packets on entries in bottleneck situations, but it has one for all 4 inputs. This fact significantly reduces the need for the memory ressources on the chip. The CU uses an arbitrary policy based on the priority-to-the-right rule. By analogy with a car traffic, we can consider the packets behave like vehicles arriving at one intersection with no traffic lights or other signalization. To get priority, the priority-to-the-right rule should be applied.

Once the packets occur at the CU's ports (up to 4, from 4 directions), the CU "receives" them all at the same time, and treats them, one by one, according to the scheduling imposed by the priority-to-the-right policy. The packet coming from one direction and which had not other packets to its right at the arrival time, has the highest priority and will be transferred as first. This packet is placed in the first register of the buffer. The packet having lower priority in the next internal buffer register and so on. The packet transfer (CU's latency) from input to output port takes 2 clock cycles. For example, for 2 packets received at the same time, their transfer is carried out in 4 clock cycles. After this delay, the CU is ready to receive new packets. The CU's routing policy is based on modified XY routing algorithm. To transfer the received packet, the CU takes into account the network conditions, that means occupations of its direct neighbouring CUs. The packet sent via the CuNoC alternates between the horizontal and vertical routing depending on network conditions. We can say that the packet "searches" a free and available way to the final destination by crossing CUs which are on its way. The used routing mechanism does not allow to the packet to take the same direction of arrival. This avoids, among others, the ping-pong game effect between the two CUs which occurs in the case of an obstacle between actual packet's position and its final destination.

3.2. Types of CU

We distinguish two types of CU: the classic CU and to-give-way CU (CUgw). All CUs behave in the same way in normal circumstances, which means in situations where bottleneck does not occur.

The control flow graphs (CFG) of the packet treatment for classic and to-give-way CU are depicted in the Figure 2. The following scenario is applied:

1. The CU receives all packets (up to 4) at the given time and puts on the status "occupied",

2. According to the priority-to-the-right policy, the CU determines the transfert schedule of the received packets.

3. The CU examines the destination address field of the packet and compares it with its own address. The result of this phase is the affection of the direction signals: *right*, *left*, *down*, *up*, *stay-x* or *stay-y*.

4. Then the CU tests its neighbouring CUs (adjacent nodes), especially those being marked by the direction signals (logic "1"), one by one following the order defined by arbitrary policy. If the adjacent CU's access is not occupied, the CU transfers the packet to its direction. The CU carries out the same procedure and tests other free accesses until the packet is delivered. If all routing possibilities are exhausted, bottleneck situation occurres. That means, the situation when all neighboring CUs are in mode "occupied". In this case, the to-give-way CU (CUgw) passes in action. The concerned CUgw puts on the status "free" and receives all packets of the adjacent classic CUs. The block labelled CUgw (dashed line) in Figure 2 describes this procedure. Once the packets are received from the adjacent classic CUs, the CUgw can transfer its packets to adjacent CUs according to their destination address field. On that way, the bottleneck situation is solved.

5. The CU carries out the same tasks, from step 3 to the end of the CFG for all packets being received and scheduled in step 1.

6. Once all packets are transferred to the neighbouring CUs or to the modules, the CU puts on the status "free" and waits for the other packets to be routed.

In order to ensure efficient communication between all modules connected to the CuNoC, each classic CU must be connected and surrounded only by the CUgw and vice versa, Figure 3.

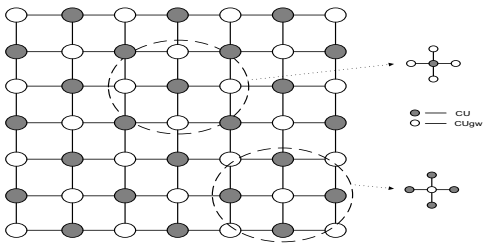

Fig. 3. Valid CUs' position in the CuNoC

3.3. Dynamic module placements in mesh-based CuNoC

The CuNoC is specially suited for dynamic placement of modules because the path between the source and destination calculates at the run-time. In fact, for different packets of one message, the path between the source and destination is not always the same.

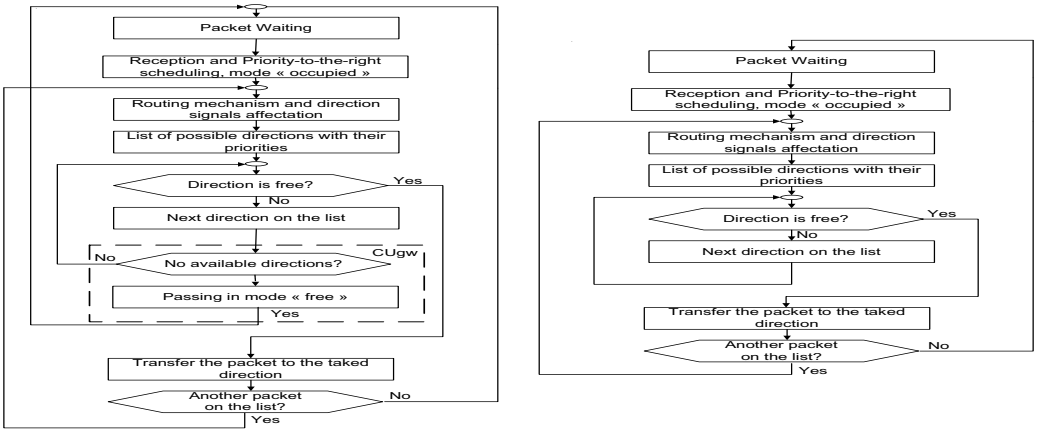

Fig. 2. Control flow graph (CFG) of CU's packet handling : a) CUgw (left) b) classic CU (right)

The network situation evolves at the time. Each packet manages to achieve its final destination. The CU which has the status "free" becomes "occupied" because it receives some packets, another CU which has the status "occupied" becomes "free" and so on. Thus, by placing a module in the middle of the network between two modules having important intercommunication, their communication will not deteriorate. The packet will consider the dynamically placed module as a temporary obstacle and will try to find another way to its destination getting round the placed module. However, a certain 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:

Rule 1. Each module (i.e. PE) has the access to the network by at least one port of the CU which surround it.

Rule 2. All PEs communicate via the CUs, even the adjacent elements.

Rule 3. The CUs are either connected to the adjacent CUs or directly to the modules via their input/output ports. Each CU is connected at least to one CU of different type (the classic CU to the CUgw and vice versa (see figure 3).

Rule 4. Each PE is surrounded by other PEs on maximum 3 sides.

Rule 5. The DyNoC rule: All PEs dynamically or staticaly placed being surrounded on all sides by CUs, are always reachable [9].

Rule 6. Between all PEs there should be at least one path allowing their intercommunication.

In Figure 5 some possible placements of modules at run-time are presented. Figure 5a presents non valid placement (Rule 6 is not respected) whereas Figure 5b presents a valid placement of modules in the CuNoC. The all placement constraints presented for the DyNoC in [9, 11] fit well with the CuNoC network.

Valid placement of the CUs presented in Figure 3 is the starting point of our approach. After having placed the network of CUs, the next step consists in the placement of the modules. Each module having the area size lower than the CU's size replaces one CU. For all other modules having the area size greater than the CU's size, we replace a zone of several CUs having total area size equal or greater than the module's area size. Figure 4 presents a possible placement of modules in the CuNoC.

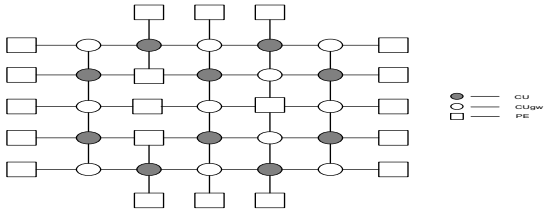

Fig. 4. A possible placement of modules in the CuNoC: each module has at least one CU connected only to it

4. SIMULATION AND IMPLEMENTATION RESULTS

We have simulated two possible structures based on the 4 x 4 mesh CuNoC : the first 4 x 4 CuNoC is used for the communication of 12 modules (see Figure 6a) and the second one for the communication of 4 modules (see Figure 6b). In the first case, 12 modules send at the same time packets and after latency period they receive them from other modules. An inconvenience of this approach is that certain packets do not arrive in the sending order. That imposes to each module to have a packet re-ordering interface. In the second case, 4 modules intercommunicate at the same time. For the both cases, all sent packets successfully reach their destination.

We have synthesized and implemented various packet data format in Xilinx Virtex II FPGA technology (Virtex II - 1000 and Virtex II - 6000). The results are given in Table 1 in terms of the Area (A) occupation for different data bit-widths (destination address field is not counted in this size), memory (M) usage and maximum operating frequency (F) in MHz.

The theorical data bit rate for the case 4x4 Network-12 modules at operating frequency of 100 MHz for 8 data width is 3.12 Gbps and maximum packet latency is 38 clock cycles. For the second case, the theorical data bit rate at 100 MHz for 8 data width is 1.04 Gbps and maximum latency is 42 clock cycles.

We show that CuNoC gives a good compromise between the logic area and operating frequency. Indeed, the CuNoC proves good performances and much less significant area overhead com-

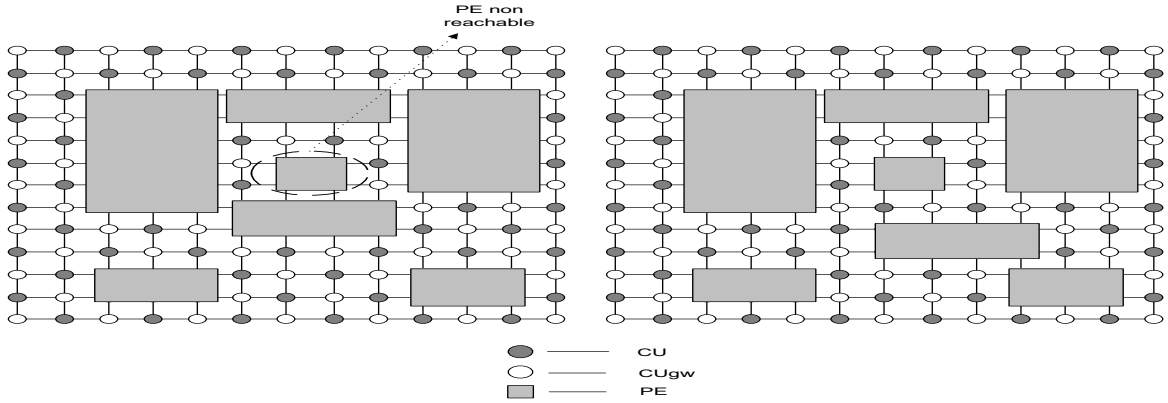

Fig. 5. CuNoC placement: a) non valid (left) b) valid (right)

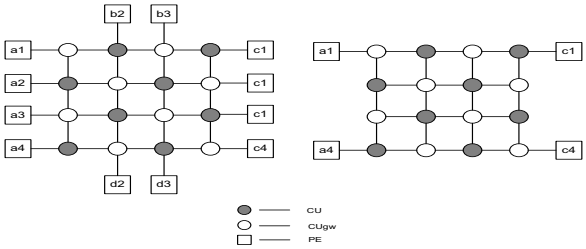

Fig. 6. Simulation structures: a) Communication of 12 modules by 4 x 4 CuNoC (left) b) Communication of 4 modules by 4 x 4 CuNoC (right)

Table 1. CU Statistics

	VirtexII-1000	VirtexII-6000
A / M / F (8 bit)	1.39 % / 0 % / 335.6	0.21 % / 0 % / 335.6
A / M / F (16 bit)	2.56 % / 0 % / 320.7	0.39 % / 0 % / 320.7
A / M / F (32 bit)	4.90 % / 0 % / 272.1	0.74 % / 0 % / 272.1
A / M / F (64 bit)	9.59 % / 0 % / 272.1	1.45 % / 0 % / 272.1

pared with DyNoC's network element.

5. CONCLUSION AND FUTUR WORK

In this article, we proposed a new paradigm for dynamic intercommunication between run-time reconfigurable modules. Our communication approach CuNoC represents an infrastructure which is namely adapted and suited to FPGA-based reconfigurable devices. We presented the basic concept of this communication approach, its basic routers architecture, possible topologies and structures mixing network CUs with processing elements (PEs). The main advantages and drawbacks as well as the implementation results on for chosen topology are also presented and discussed. The future work are to apply the CuNoC on a given applications and to test the CuNoC on other topologies than mesh.

6. REFERENCES

[1] L. Benini and G. D. Micheli, "Networks on chips: a new SoC paradigm," in *IEEE Computer*, Jan. 2002.

[2] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Oberg, and D. Lindqvist, "Network on chip: An architecture for billion transistor era," in *Proc. of the International NorChip Conference*, Sept. 2000.

[3] P. Guerrier and A. Greiner, "A generic architecture for on-chip packet-switched interconnections," in *Proc. Design and Test in Europe (DATE)*, Mar. 2000, pp. 250–256.

[4] S. K. et al, "A network on chip architecture and design methodology," in *Proc. IEEE Computer Society Annual Symp. on VLSI*, 2002, pp. 117–124.

[5] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proc. Design Automation Conf. (DAC)*, 2001, pp. 683–689.

[6] F. Karim and et al, "An interconnect architecture for networking systems on chips," *IEEE Micro*, vol. 22, no. 5, pp. 36–45, Mar. 2002.

[7] P. P. Pande, C. Grecu, A. Ivanov, and R. Saleh, "Design of a switch for network on chip applications," in *Proc. Int. Symp. Circuits and Systems (ISCAS)*, vol. 5, May 2003, pp. 217–220.

[8] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures," *IEEE Trans. Comput.*, vol. 54, no. 8, pp. 1025–1040, Aug. 2005.

[9] 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," in *International Conf. on FPL*, Aug. 2005.

[10] M. Majer, C. Bobda, A. Ahmadinia, and J. Teich, "Packet routing in dynamically changing networks on chip," in *Proc. of 19th IEEE International Parallel and Distributed Symposium*, Apr. 2005.

[11] C. Bobda and A. Ahmadinia, "Dynamic interconnection of reconfigurable modules on reconfigurable devices," in *Design & Test of Computers, IEEE*, vol. 22, Sept. 2005, pp. 443–451.