Design and Implementation of an FPGA-based NoC for Real Time Systems

Yilian Ribot

Geoffrey Nelissen

CISTER-TR-190510

2019/07/09

Design and implementation of an FPGA-based NoC for Real Time Systems

Yilian Ribot, Geoffrey Nelissen

CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail: ribot@isep.ipp.pt, gnn@isep.ipp.pt

https://www.cister-labs.pt

Abstract

In order to communicate, cores of a multi-core platform traditionally relied on shared busses. However, with the increasing number of computation nodes integrated in multi- and many-core platforms, Network-on-Chips (NoCs) emerged as a new alternative communication medium in Systems-on-Chips (SoCs). Hoplite-RT is a new NoC design that was recently proposed. Hoplite-RT is a compact design easy to analyze and with a low-cost implementation that was specifically tailored for FPGA. In this work, we introduce priority-based routing to Hoplite-RT and change the network topology so as to improve its timing behavior, i.e., its Worst-Case Traversal Time (WCTT).

Design and implementation of an FPGA-based NoC for Real Time Systems

Yilian Ribot González and Geoffrey Nelissen
CISTER Research Centre, ISEP, Polytechnic Institute of Porto, Portugal

***Abstract*—In order to communicate, cores of a multi-core platform traditionally relied on shared busses. However, with the increasing number of computation nodes integrated in multi- and many-core platforms, Network-on-Chips (NoCs) emerged as a new communication medium in Systems-on-Chips (SoCs). Hoplite-RT is a new NoC design that was recently proposed in [1]. Hoplite-RT has a compact design, is easy to analyze, and was specifically tailored to target FPGAs. In this work, we introduce priority-based routing to Hoplite-RT and change the network topology so as to improve packets' Worst-Case Traversal Time (WCTT).**

1. Introduction

SoCs are usually composed of several, possibly heterogeneous, processing elements. In order to communicate, the processing elements used to rely on shared busses. However, due to the large increase of on-chip elements during the last decade, communication through shared busses is not an appropriate solution for such platforms anymore. Indeed, at most one node can take control of a bus and transmit data at each cycle. This causes a bottleneck for the overall system. NoCs have been identified as a good alternative to palliate this issue [2]. NoCs are router-based packet switching networks and hence allow several processing elements to transmit messages in parallel [3]. As discussed in [4], NoCs have remarkable scalability, parallelism, and re-usability properties, and help meet system-wide power and timing constraints.

FPGAs are chips made of reconfigurable elements that can be programmed to implement virtually any digital functionality. Therefore, they are well suited to the development of custom-made SoCs. FPGAs allow to design systems with a high degree of parallelism and high data processing rate at a relatively low cost. However, FPGAs expose a limited number of reconfigurable elements and most FPGAs do not supply enough resources to embed complex NoC designs together with a large number of processing elements.

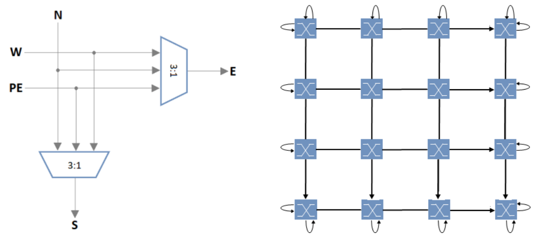
The literature on NoCs is extensive. Most of the proposed solutions that present suitable properties for real-time systems (i.e., bounded worst-case timing behaviors) rely on wormhole switching with virtual channels, buffering, and often some sort of priority-driven routing arbitration [5], [6]. These concepts allow to develop powerful NoC infrastructures with bounded WCTT but suffer from two main drawbacks: (1) they are expensive to implement in FPGA platforms; and (2) their complexity renders their analysis extremely complex as shown by the number of flaws that were recently discovered in existing works [7].

In complete opposition to the solutions mentioned above, Hoplite is a newly proposed NoC infrastructure [8] that reduces the NoC features to their bare minimum and hence decreases considerably the network analysis complexity and implementation cost (in terms of FPGA resources utilization). Introduced by Wasly et al in [1], Hoplite-RT is a modified version of Hoplite that provides an upper bound on the NoC WCTT. Each Hoplite-RT router has three input ports (North (N), West (W) and Programming Element (PE)) and two output ports (South (S) and East (E)) (see Figure 1(a)). Hoplite-RT assumes that packets transmitted through the NoC are composed of a single flit with two fields: the destination address and the transmitted data. Hoplite-RT connects routers in a torus topology (Figure 1(b)) and employs a modified version of X-Y routing (packets first travel horizontally on the X axis, and then vertically on the Y axis). Specifically, the Hoplite-RT routing policy is built upon the concept of deflection to avoid the cost of packet buffering. When two packets coming from the W and N port of a router conflict for the S port, Hoplite-RT gives the highest priority to the packet originating from the W port. The packet originating from the N port is then deflected toward to the E port inconsiderately of its final destination. However, as the deflected packet is now traveling along the X axis, it will have the highest priority when it will require to go south again. Therefore, the maximum number of deflections suffered by a packet can be upper-bounded (see Sec. 2).

Contribution. In Hoplite-RT, a packet may be deflected after each and every hop on the Y-axis, thereby leading to possibly large WCTTs. Furthermore, Hoplite-RT treats all packets identically. It does not allow to associate different priorities, and hence quality of services to different packets. Nonetheless, Hoplite-RT is a compact, easy to analyze, easy to implement, and inexpensive design. For those reasons, we are interested in developing a NoC that keeps the advantages of Hoplite-RT while improving its real-time capabilities. This paper represents the first stages of our research: (1) we propose a solution to limit the number of deflections and hence the WCTT of a packet; and (2) we introduce a notion of quality of service in the routing policy.

2. Background

The architecture of an Hoplite-RT router is shown in Figure 1(a). It is implemented using two multiplexers of three inputs. Hoplite-RT takes advantage of the possibility of *"fracturing"* the Look Up Tables (LUTs) of modern FPGAs to reduce the implementation cost of the expensive crossbar multiplexers. The modern families of Xilinx



(a) Hoplite-RT router (b) Hoplite-RT torus topoligy

Figure 1. Hoplite-RT design.

TABLE 1. Resources utilization in a Virtex-7 485T FPGA

Router	LUTs	FFs
Hoplite-RT router	85	139
Hoplite-RT router + Priorities	86	139
Hoplite-RT router + Priorities + New Topology	88	139

FPGAs present 6-inputs LUTs that can be fractured in two 5-inputs LUTs sharing the same five input signals. Since each 3:1 multiplexer can be implemented with a 5-inputs LUT, the two multiplexers of the router can be implemented with a single 6-inputs LUT. The first row of Table 1 shows the cost in terms of LUTs and flip-flops (FFs) of a 64bits Hoplite-RT router implemented in a Xilinx Virtex-7 485T FPGA after fracturation.

The WCTT wc_{TT} of a packet transmitted over Hoplite-RT between two nodes with coordinates (x_o, y_o) and (x_d, y_d) in a mesh of size $S_x \times S_y$ is given by Equation (1) (in clock cycles)

$$wc_{TT} = h_x + h_y + (h_y \times S_x) + 2, \tag{1}$$

where h_x and h_y are the distances travelled by the packet on the X and Y-axis, respectively, when it does not contend with any other packet (i.e., without any deflection). That is,

$$h_x = (x_d - x_o + S_x) \mod S_x \tag{2}$$

$$h_y = (y_d - y_o + S_y) \mod S_y \tag{3}$$

The term $(h_y \times S_x)$ accounts for potential deflections costs.

3. Preliminary results

In this section, we describe the modifications we made to the Hoplite-RT design in order to: (1) introduce a notion of priority in the routing policy; and (2) decrease the worst-case traversal time of a packet.

3.1. Priority-based routing

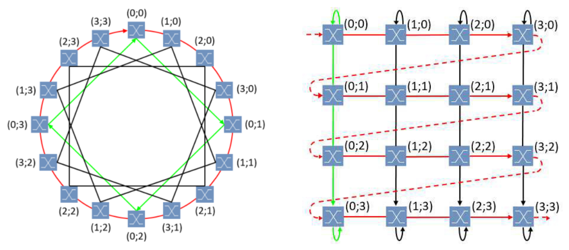
We first modify Hoplite-RT to introduce a two priority levels (Low and High) scheduling scheme. This approach allows us to provide different levels of quality of service to different packets and hence decrease the *average* traversal time of the high priority packets. The WCTT of high priority packets may also be improved depending on the application mapping and the network configuration. However, we leave that analysis for future work.

In Hoplite-RT, the packets coming from the W port always have the highest priority. Instead, in our new design, low priority packets coming from the W port will *never* be permitted to deflect high priority packets coming from the N port. That is, if a high priority packet coming from the N port and a low priority packet coming from the W port conflict for the S port, then the N packet wins the right to use the S port, and the W packet is deflected towards the E port. To support this new routing policy, the packet priority is encoded in its most significant bit.

We observe from Table 1 that this simple modification consumes only one additional 6-inputs LUT in comparison to a normal Hoplite-RT router.

3.2. New topology

Even though it looks beneficial, the new priority-based routing policy described in Section 3.1 is in fact extremely inefficient; the WCTT of high priority packets remains unchanged (only their average-case traversal time is reduced), but more importantly, the WCTT of low priority packets is not bounded anymore. Indeed, a low priority packet may always contend with high priority ones and therefore never be able to use the S port of a router.



(a) Ring representation (b) Equiv. mesh representation

Figure 2. Directional ring with bypasses topology

In this section, we propose a new network topology that, in most cases, reduces the WCTT of high priority packets by a factor of two, and allows to reinstate the same WCTT bound for the low priority packets as in the original design of Hoplite-RT.

In the standard torus topology (see Fig. 1(b)), a deflected packet will hop through S_x routers (where S_x is the number of routers on the X axis), before entering again in the same router where it was initially deflected. That is, the packet did not progress toward its destination after those S_x additional hops. We prevent this issue to happen by changing the network topology to a directional ring with bypasses (see Fig. 2(a)). In that new topology, all routers are connected by a single unidirectional ring (red links). Then, every pair of routers that are S_x positions apart on the ring are connected by a bypass (green and black links). Equivalently, if we look at the network as a mesh (see Fig. 2(b)), it connects the E port of the last router in row number y to the W port of the first router in row number $(y + 1) \mod S_y$. Then, the bypasses correspond to the links on the columns of the mesh.

Thanks to this new topology, when a packet is deflected, it reaches the same router as it would have if it was not deflected, after S_x hops. That is, the packet always progresses toward its destination even when deflected. Consequently, the WCTT decreases.

However, this new topology requires to modify the router design. Indeed, in the particular case where two packets arrive at the same instant in the same router (via the W and N ports) and that router is their destination, Hoplite-RT would solve the conflict by deflecting one of the two packets to the E port. This situation causes a remarkable increase in the WCTT of the deflected packet. We aim at solving this issue by allowing the programming element connected to the router to read both packets simultaneously. This necessarily increases the design cost of the router and the programming element. The cost of implementing a router with priority routing in the new topology is shown in Table 1. It requires only three additional 6-inputs LUTs in comparison to the original Hoplite-RT design.

3.3. Bound on the WCTT

The WCTT of a packet is defined as follow:

$$wc_{TT} = n_{hops} + n_{def} \times c_{def}, \tag{4}$$

where n_{hops} is the number of hops in a network with zero load (i.e., when the packet does not suffer any deflection), n_{def} is the maximum number of deflections suffered by the packet on its route, and c_{def} is the cost of a deflection.

The term n_{hops} is defined as:

$$n_{hops} = h_r + h_b + 2, \tag{5}$$

where h_r and h_b are the number of hops on the ring and bypasses, respectively, and the additional two hops account for the injection (at the source node) and exit (at the destination node) of the packet into and from the network. We prove bounds for each of those terms.

Lemma 1. The number of hops on the ring in a zero-load network is given by $h_r = (x_d - x_o + S_x) \mod S_x$.

Proof: According to our routing policy, each packet travels first through the ring from the origin router at coordinate (x_o, y_o) until it reaches a router with the same X coordinate x_d as the destination. According to the topology presented in Fig. 2(b), the number of hops on the ring is:

$$\begin{aligned} h_r &= \begin{cases} x_d - x_o & \text{when } x_d \geq x_o \\ x_d - x_o + S_x & \text{when } x_d < x_o \end{cases} \\ &= (x_d - x_o + S_x) \mod S_x \end{aligned}$$

□

Lemma 2. The number of hops on a bypass in a zero-load network is given by $h_b = (y_d - y'_o + S_y) \mod S_y$ where

$$y'_o = \begin{cases} y_o & \text{when } x_d \geq x_o \\ y_o + 1 & \text{when } x_d < x_o \end{cases} \tag{6}$$

Proof: Remember that a bypass in Fig. 2(a) corresponds to a column in Fig. 2(b). Let S_y be the number of routers in a column, and y'_o be the Y coordinate of the router at which the packet stops travelling on the ring and starts using bypasses (i.e., the first router with the same X coordinate x_d as the destination). Then, according to the router numbering shown in Fig. 2(b),

$$y'_o = \begin{cases} y_o & \text{when } x_d \geq x_o \\ y_o + 1 & \text{when } x_d < x_o \end{cases}$$

and the number of hops on the Y axis of the mesh is:

$$\begin{aligned} h_b &= \begin{cases} y_d - y'_o & \text{when } y_d \geq y'_o \\ y_d - y'_o + S_y & \text{when } y_d < y'_o \end{cases} \\ &= (y_d - y'_o + S_y) \mod S_y \end{aligned}$$

□

The maximum number of deflections n_{def} that a packet may suffer is different for high and low priority packets. We analyze both cases in Lemmas 3 and 4.

Lemma 3. The maximum number of deflections suffered by a high priority packet is given by $n_{def} = \left\lfloor \frac{h_b}{2} \right\rfloor$.

Proof: Let P denote a high priority packet. When P enters a router from the W port and requests the S port, it cannot be deflected. Then, in the following router of the Y axis, P will be a packet coming from the N port. In this new router, P may be deflected toward the E port by another high priority packet conflicting for the S port. Subsequently, P will travel on the ring until it reaches a router with the same X coordinate as its destination. The whole process will then repeat until P arrives to its destination router. That is, P may be deflected at half the routers it traverses that share the same X coordinate as its destination, i.e., it may be deflected $\left\lfloor \frac{h_b}{2} \right\rfloor$ times. □

Lemma 4. The maximum number of deflections suffered by a low priority packet is given by $n_{def} = h_b$.

Proof: A low priority packet entering from the W port may always be deflected to the E port. Therefore, a low-priority packet may be deflected as many times as it may try to use a bypass, i.e., h_b times. □

The additional cost in terms of hops introduced by each deflection is analyzed in Lemma 5.

Lemma 5. The cost of a deflection is $c_{def} = S_x - 1$.

Proof: When a packet is deflected, it must hop through S_x routers on the ring to reach the same router as it would have if it could have used the bypass instead, i.e., though S_x routers instead of 1, thereby leading to an additional cost of $S_x - 1$. □

4. Conclusion and Future works

In this paper, we presented solutions to improve the timing performance of Hoplite-RT with a marginal increase of the FPGA resource utilization. We first introduced a notion of priority in the routing policy. Then, by changing the network topology to a directional ring with bypasses, we reduced the number of deflections and therefore the WCTT of high priority packets, and maintained the WCTT of low priority packets.

Yet, because each packet may suffer a different number of deflections, Hoplite-RT does not guarantee that packets will be received at the destination router in the same order as they were emitted at the origin router. Therefore, with the current design, long messages can difficulty be split in several flits sequentially injected in the network. As future work, we plan on developing solutions to ensure that the packets arrive in an orderly fashion at the destination router while keeping the advantages of Hoplite-RT and trying to increase its cost as little as possible. We are also working on solutions to map applications on nodes and configure packet injection rates at each node to fully take advantage of the new network design discussed in this paper, e.g., to optimize the link bandwidth usage, or to improve the WCTT of high priority packets at the detriment of low priority ones.

Acknowledgements. This work was partially supported by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UID/CEC/04234), by the Operational Competitiveness Programme and Internationalization (COMPETE 2020) under the PT2020 Partnership Agreement through the European Regional Development Fund (ERDF), and by national funds through FCT within project POCI-01-0145-FEDER-029119 (PReFECT).

References

[1] S. Wasly, R. Pellizzoni, and N. Kapre, "HopliteRT: An efficient FPGA NoC for real-time applications," in *ICFPT 2017*, 2017.

[2] L. Benini and G. De Micheli, "Networks on chip: a new paradigm for systems on chip design," in *Proc. of DATE 2002*, 2002.

[3] A. Agarwal and R. Shankar, "Survey of network on chip (noc) architectures and contributions," *JECA 2009*, vol. 3, 2009.

[4] C. A. Zeferino and A. A. Susin, "Socin: a parametric and scalable network-on-chip," in *Proc. of 16th SBCCI*, 2003.

[5] Y. Huan and A. DeHon, "FPGA optimized packet-switched NoC using split and merge primitives," in *Proc. of FPT 2012*, 2012.

[6] N. Kapre, N. Mehta, M. deLorimier, R. Rubin, H. Barnor, M. Wilson, M. Wrighton, and A. DeHon, "Packet switched vs. time multiplexed fpga overlay networks," in *Proc. of 14th IEEE FCCM*, 2006.

[7] N. Borislav, S. Tobuschat, L.Soares, R. Ernst, and A. Burns, "Real-time analysis of priority-preemptive nocs with arbitrary buffer sizes and router delays," *RTS 2019*, vol. 55, no. 1, 2019.

[8] N. Kapre and J. Gray, "Hoplite: Building austere overlay nocs for fpgas," in *Proc. of 25th FPL*, 2015.