Journal of Circuits, Systems, and Computers
Vol. 23, No. 9 (2014) 1450120 (21 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0218126614501205

EFFICIENT SCHEME FOR CONGESTION CONTROL IN NETWORK-ON-CHIP WITH QoS CONSIDERATION*

ADEL SOUDANI[†], AHMED ALDAMMAS[‡]
and ABDULLAH AL-DHELAAN[§]

College of Computer and Information Sciences,
King Saud University, Kingdom of Saudi Arabia
[†]*asoudani@ksu.edu.sa*
[‡]*bindammas@student.ksu.edu.sa*
[§]*dhelaan@ksu.edu.sa*

Received 24 December 2013
Accepted 9 May 2014
Published 25 June 2014

Embedded distributed multimedia applications based on the use of on-chip networks for communication and messages exchange requires specific and enhanced quality of service (QoS) management. To reach the desired performances at the application level, the network-on-chip (NoC) router should implement per flit handling strategy with wide granularity. This purpose requires an enhanced internal architecture that ensures from one hand a specific management according to a service classification and from the other hand, it enhances the routing process.

In this context, this paper proposes a new mechanism for QoS management in NoC. This mechanism is based on the use of central memory where flits are in-queued according to their class of service. This scheme enables an optimal flit scheduling phase and provides more capabilities to drop low important flits when the router shows congestion state symptoms. The paper presents, also, a protocol structure that fills with this architecture and introduces a signaling mechanism to make efficient the QoS management through the proposed architecture. The circuit performances and its adaptability to achieve QoS with low power processing and high bandwidth in on chip multiprocessor systems will be studied in this paper.

Keywords: Network-on-chip; congestion control; WRED algorithm; quality-of-service; performances.

1. Introduction

The design of high-performance networks-on-chip (NoC) used for the interconnection of sophisticated multi-processor system-on-chip (MP-SoC) with billions of transistors requires a careful combination of different metrics related to different issues. In fact, in addition to already existing constraints related to the embedded hardware and real-time system co-design, other constraints related to protocol

*This paper was recommended by Regional Editor Tongquan Wei.

efficiency and reliability should be, also, considered to evolve the NoC in order to face the challenges related to quality-of-service (QoS).

Actually, with the 45 nm application-specific integrated circuit (ASIC) integration technology the design of more complex protocol management tasks in the on chip architecture is being thinkable. The additional cost associated to these tasks, look to be acceptable in order to enhance the provided QoS to the application level.

A big interest at the research community is granted to the design of embedded micro-architecture with specific scopes to optimize both required physical resources and networking constraints.[1–5] One of the most interesting issues that limit the use of these systems for efficient communication is the congestion control and efficient way for its resolution.[6–9] In fact, these systems with a high bandwidth suffer from the leak of memory that represents the costly critical part in the design of the embedded system. The flit's losses might heavily affect the ensured QoS at the application level that in turn questions the efficiency of these systems for reliable communication.

We think that an efficient strategy for flits queuing and an enhanced scheme for internal memory organization will contribute to reduce the congestion occurrences in NoC and to ensure better delivery of the data while avoiding distortion in QoS for the communication processes. To achieve this goal, we believe that a substantial research effort is yet required to define an enhanced router architecture that enables internal per-flit management as well as the interaction with the neighbors. This paper is a contribution in this area. Its main goal is to propose a new scheme to avoid congestion, in NoC routers, based on per-flit management. In-depth, flits are managed according to their data type (flit-type) and their importance in their associated communication process. Low priority flits will be discarded using a WRED-like[10] technique for flow control mechanism.

The paper evaluates the efficiency of the proposed solution to avoid congestion and to ensure QoS in NoC. It also presents the main characteristics of the ASIC circuit of the specified router and discusses it adequacy for low power achievements.

2. Overview of Related Works

Congestion control mechanisms, in NoC, were proposed to avoid saturation and improve the throughput in NoCs. Congestion-aware adaptive routing is one of the most adopted methods to balance the traffic over the network.[7] The routing process might be enhanced with the information's related to internal buffer status and the link utilization so that the router can optimize the choice of the next hope for incoming flits.

Adaptive routing algorithm was also proposed for congestion avoidance. In Ref. 11, the authors proposed an algorithm based on dynamic XY (DyXY), is called enhanced dynamic XY (EDXY). In the proposed scheme, every router shares two dedicated wires for congestion information in every direction for all the neighbors. This approach avoid direct XY routing problem and makes possible to share the traffic in the network. This method showed a good performance for packet latency (number of clock cycle) compared to other method like direct XY routing. The hardware prototyping showed also attracting features. However, the proposed method does not deal about optimal congestion resolution when it occurs in order to keep acceptable QoS at the application level.

An approach based on fully adaptive routing scheme (EBAR) was proposed in Ref. 12 to distribute thermal energy over all the NoC. They proposed a new model for thermal distribution in the NoC while keeping acceptable communication performances. The thermal state of the router was then used as condition in the routing table in order to avoid high temperature point in the NoC.

Proximity congestion awareness technique, based on signaling scheme between neighboring routers, was also proposed to avoid congested areas.[8,13] Broadcasting the congestion information's and the buffer status requires extra wires or a communication overhead. Other scheme for congestion management based on the use of virtual channel as a congestion metric of the router was also proposed.[14]

In Ref. 15, authors proposed a method to utilize both local and non-local network information to determine the optimal path to forward a packet. This method is the base of a routing scheme called congestion aware trapezoid-based routing algorithm (CATRA). The idea defended in this work is to involve the congestion status of the nodes that are more likely to be selected in the routing path and to process flits routing accordingly. The paper deals also about congestion information distribution among the network by adding dedicated wiring for that purpose. The implementation of this approach was studied at the hardware level to show that the overhead cost is acceptable in term of circuit area and power consumption. However, it was not shown in this paper the improvement of the QoS at the communication level.

Becker *et al.* introduced in Ref. 16 a new scheme to control the utilization of router input buffers by the observation of the traffic. This mechanism heuristically limits credit availability for each VC based on its observed performance characteristics. The implementation of this approach showed interesting performances. However the power consumption cost of this scheme was not studied. Besides, the impact of this method to the communication process at the application level also needs to be evaluated.

In Ref. 9, the authors introduced the flit-reservation flow control in which the buffer space and channel bandwidth are reserved for the data flits before they arrive. In the proposed scheme a per-flit control is processed when data are delivered to the network as a flow control mechanism. Advance scheduling for buffers in this method, increases the utilization of buffer space in a comparison with other methods. It is clear that in flit-reservation flow control, the buffer is reserved during the actual transmission resulting in the re-usage of the buffer immediately. As result of flit reservation, the latency will be minimized due to the elimination of many cycles that make the routing and arbitration decisions.

The paper presented in Ref. 17, proposed a congestion control strategy for on-chip networks to control the network load. The proposed strategy introduces congestion controlled best-effort (CCBE) as a new service level. CCBE connections trade bandwidth for constant and reduced latency. Link utilization is used as congestion metric. The centralized model predictive controller (MPC) gathers the link utilization as congestion metric and determines the load of the connection. The measurements obtained by hardware analysis probes are sent to a MPC that decides CCBE loads based on this information and model-based predictions.

While many efforts were carried in the context of congestion avoidance and load balance,[18,19] few of them were per-flit oriented schemes. It seems clear that a research effort is yet required to achieve per-flit memory management with an adequate mechanism for flow control to guaranty an acceptable QoS when the network is in congestion state. In order to achieve that goal, enhanced internal hardware structure of the router has to be designed.

In the following part of this paper, we present our proposed approach for the enhanced internal micro-architecture of the NoC router. We show its capacity to delay and avoid congestion while enhancing the QoS for the end to end communication process.

3. Specification of the Internal Hardware Architecture

The proposed on-chip router architecture is summarized in Fig. 1. The main novelty of this architecture is to allow the network to perform on the best effort while pushing the capability to manage congestion in the internal functions of the router itself. For that purpose, the designed architecture is specified in order to provide per-flit management according to the processed data type. In fact, the structure of flits

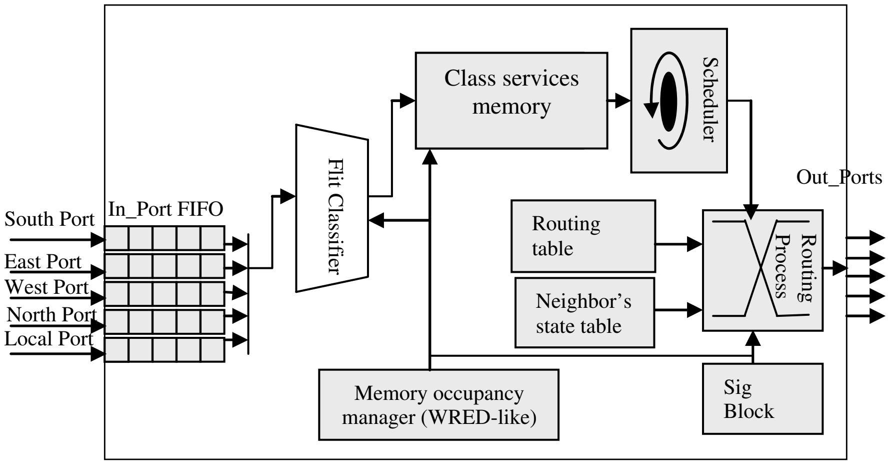


Fig. 1. Internal hardware architecture of the router.

crossing the router defines a set of fields making possible a handling according to the class service identifier (id_serv). When a symptom of congestion is detected a specific mechanism is started to unload the memory according to the importance of the flits in their communication process. An identifier is then associated to the importance of the flit payload in packet (priority tag: pr_tag).

At their arrival to the router, the flits are first classified into class of services and then in-queued in the internal memory. The output scheduling and congestion avoidance mechanism will be then performed based on the information carried by the flits. This strategy gives more autonomy to routers to get local decision for congestion avoidance and reduces the overhead communication in the network.

The specified architecture is composed of different blocks ensuring the internal tasks required during the flit's processing. All of these internal components are synchronized by an external clock signal.

- **The input FIFO:** These input queues are used to save asynchronous incoming flits from neighbor's routers. Their depths define the capabilities of the routers to absorb a burst of incoming flits. In the structure of the router, these input FIFO(s) are physically independents and they are managed separately.
- **The Flit classifier:** This block is responsible of flits identification according to their class of service identifier (id_serv). It extracts, every clock cycle, flits from the input FIFO(s) that connect the router to the neighbors. The identifier of the class of service (id_serv) is then used to in-queue the received flits in their corresponding virtual in-queue in a common memory used for this purpose. This classification helps later to perform their output delivery according to the service constraint. The state of the memory is controlled by the *memory occupancy manager* that is responsible of congestion prediction and avoidance by selecting the flits to be dropped according to their importance tags when the memory is about to be congested.
- **The central memory:** This memory is used to store flits as different class service in-queues. This memory interfaces, from its output, the *routing process* through a scheduling mechanism (Figs. 1 and 2). The output scheduling mechanism extracts flits from the in-queues proportionally to the weight of the class of services and send them to be routed to their output ports. *The memory occupancy manager* has to permanently control the depth of the in-queues in the memory to estimate it's occupancy state. Above a certain occupancy threshold, the memory is considered to be loaded and this process start selecting from the in queues tails of this memory where less significant flits from each process to be dropped in order to avoid congestion in the router.
- **The Scheduler process:** This block performs the link between the memory and the routing process. Its main function is to extract flits from the memory outputs and forward them to the routing process. Its function is based on weighted-round

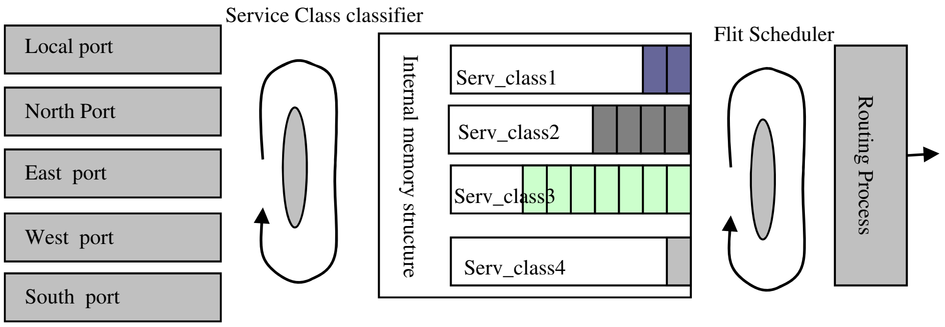


Fig. 2. General approach for service classification.

Robin (WRR)-like algorithm. In fact, it associates a weight (Wi) for each service_class_i. The flit's extraction from the four classes of services is performed proportionally to the weight (Wi). The idea in the use of this scheduling approach is to handle the incoming flits according to their QoS constraints while maximizing the output rate of the router.

In the proposed architecture, the routing block is able to process at maximum 5 flits having different destination addresses during one clock cycle. So as explained in Fig. 2, the scheduling mechanism aims to maximize the number of flits processed in a clock cycle.

Let us consider that, during a scheduling cycle, the number of flit having different destinations in-queued in class service (i) is defined by Flit_class_i, that is defined by Eq. (1):

$$\text{Flit\_class\_}i = \sum_{j=0}^{(\text{MaxClass}_i)-1} \text{flit}(j)\,. \tag{1}$$

In this Eq. (1), the MaxClass_i for the class (i), is defined by Eq. (2)

$$\text{MaxClass}_{\_i} = \omega_i; \quad \text{where } i \in \{1\ldots4\}\,. \tag{2}$$

In this proposed router architecture, the output scheduler process is triggered by the routing block. So when it is being triggered, the number of flits that will be transferred to the routing block is expressed by Eq. (3) during one scheduling cycle. In this equation, all the selected flits should get different output ports:

$$\text{NbrFlit} = \sum_{i=1}^{4} \sum_{j=0}^{(\text{MaxClass}_i)-1} \text{flit}(j) \le 5\,. \tag{3}$$

As expressed in this equation, in one output scheduling cycle, all the class services can be concerned. So, while this approach, based on WRR-like mechanism, allows sending out variable number of flits form classes of services according to theirs weights (Wi), it maximizes the total number of flit that will be managed by the router in one routing cycle. As a consequence, it will minimize the total in-queue length of flits waiting in the memory that in turn reduces the transaction time and avoid congestion.

- **The Routing block:** It is the main block in the NoC router. It ensures the routing process of the different incoming flits. It is based on the wormhole technique and allows channel multiplexing with the concept of virtual circuit (VC) that improves bandwidth allocation with the TDMA concept.[19,20] A communication process starts with the transmission of the *head-data flit* that contains informations about the process identifier (id_proc), the source and the destination addresses. The informations in this flit are used to establish the communication path for the remaining data flits. Each router has an internal representation about traffic load state of all the neighbors' routers. The routing algorithm will, then, favorites the application of a direct $X–Y$ routing scheme to forward the header flit to the next router in the path. If the next hope is in congestion, the adaptive $X–Y$ scheme based on the state of the neighbors will be used to find the next hope. The current router will update the *routing table* with the couple of informations (id_proc, output_port). All the next coming data_flits with the same id_proc will be forwarded through the same channel using the information updated in the *routing table.* At the reception of the end_data_flit this created virtual channel will be aborted.
- **The Signaling block:** This block provides some tasks related to the signaling process in the network. It generates and diffuses asynchronously signaling flits to update the neighbors' routers with a new traffic load state to be considered in the routing process. At the reception of a new signaling flit, this block analyzes the incoming signaling-flit in order to update the internal table with the neighbors' router's states. These information's stored in this table are required for the management of a head-flit during the path establishment process. This block interacts with the memory controller to get the updated state of the memory.
- **Memory occupancy manager block:** This block uses an approach based on weighted random early detection to drop the incoming flits having low-importance in order to avoid memory congestion when required. It controls the memory occupancy and when the occupancy is above a maximum threshold, it starts dropping flits with low importance from the memory and also from the output of the flit classifier based on their service classes (id_serv) and importance tags (pr_tag). The priority is scaled out of four for all the class of services. It is attributed by the IP when it generates the application data stream. *The Memory occupancy manager* starts dropping flits with the lowest priorities (pr_tag = 0). If after finishing all the flits with low-priorities and the router is still congested, the concerned priority will be increased to involve more dropped incoming flits which reduces the congestion resolution time.

4. The General Protocol Structure and Flit Types

In the proposed communication architecture, data exchanged between distributed IPs in the network are structured in different flit types. We can roughly distinguish tow flit family-types: Data flit and signaling flit. To maintain scalability and to match the hardware data bus and internal router-buffer constraints, we have chosen a fixed flit size for all flit-types (32 bits).

Data flit: It is dedicated to the transport of data between IPs. During the communication process, three types of data flit are required:

- **Head-data flit:** The wormhole commutation technique is based on route establishment (with physical multiplexed channel)[21] between source and destination IPs. In the proposed architecture, this step should take into account some information related to the expected process such as class service identifier (Id_serv). Furthermore, it is mainly enhanced by the study of the physical states of the expected in-path NoC-routers. These informations, quantifying the available memory space and the length of in-queue waiting flit in service class, are provided by the signaling process between neighbors' routers. The head data flit used in our approach is presented in Fig. 3.
- **Continuation-data flit:** This flit type (Fig. 3) is processed after the establishment of the communication path between the source and the destination during packet transmission. It transports the payload data. In this flit, the process identifier Id_proc will be used to identify the output port at the crossed routers using a local switching table that must be updated at the initiation and the end of each process.

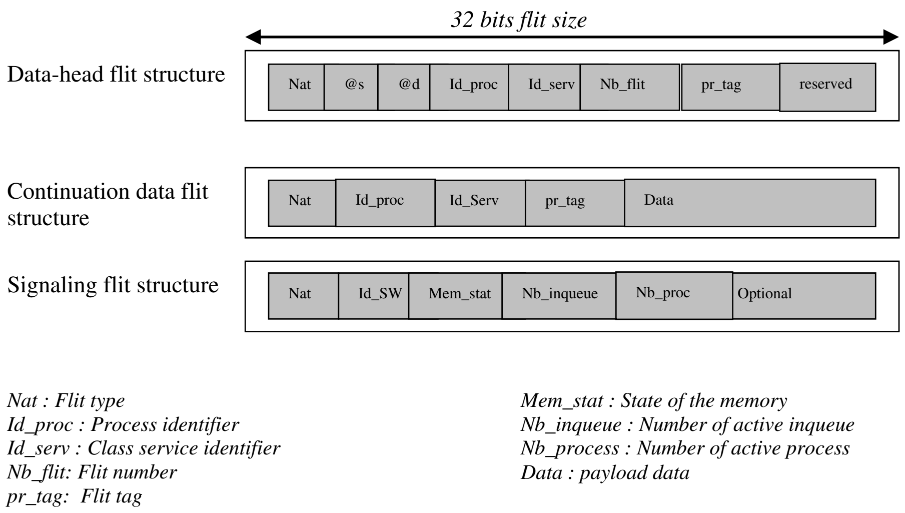


Fig. 3. Flit structures in the proposed communication architecture.

- **End-data flit:** It has the same structure as the data flit but it is used to free reserved resources for current communication process.

Signaling flit: This flit is generated by a router whenever the parameters quantifying its state get changed. It carries useful information between neighbors' routers in order to provide them with a virtual representation of their neighbors.

5. Service Classes Management and Memory Structure

The end-to-end QoS provided by the NoC is mainly depending on the performances of the internal router to provide granularity when handling flits of a given communication process. The way on which is based the processing of incoming flit, before its output delivery, affects the flit time transaction and also the depth of waiting in-queue flits. Therefore, we have proposed a flit management approach based on service differentiation that enhances the scheduling to meet the application requirements.

The internal memory is designed to classify and to store flits in four service classes based on their application-QoS requirements. We think that any communication process can belong to one of four set of class of services that are:

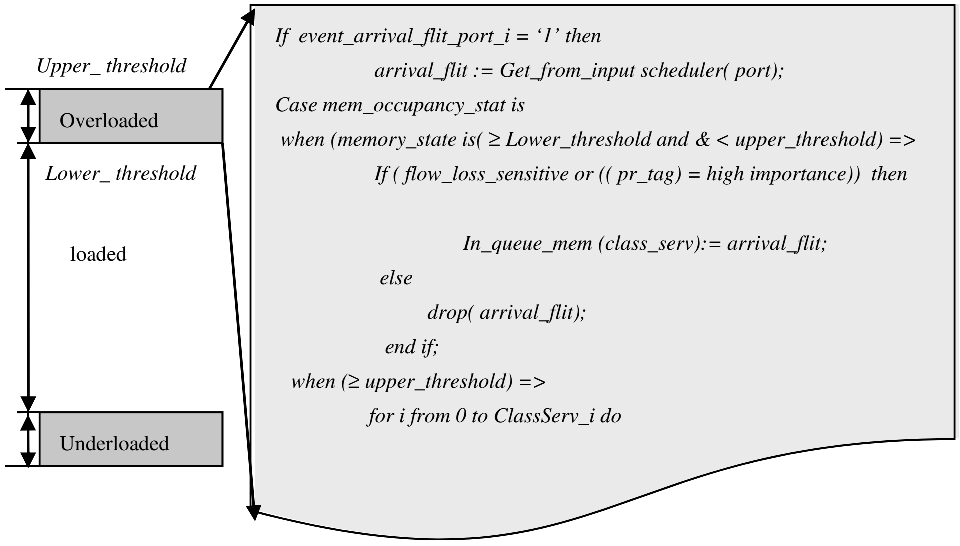
- *Signaling flits*: Dedicated to transfer short messages between IP-cores such as control, acknowledgment and interruption messages. The highest priority in scheduling is associated to this service level.
- *Real-time flits*: This service level is associated to critical time applications such as multimedia applications.
- *Short data flits*: Concerns short memory data and register access.
- *Block memory transfer flits*: It is dedicated to the transfer of huge amount of data from IP to another one.

At the input, the Service_class_classifier extracts the incoming flits from the input FIFO of the different in-ports of the router using round robin (RR) way. Referring to their service identifier (id_serv), it classifies them into four service classes in-queued in a common memory. For that purpose, and as explained, the available memory space is organized into four in-queues maintained with a dedicated set of pointers.

At the output, the Scheduler process takes into account each service class with the consideration of a corresponding weight (Wi) during the management of their

Table 1. Service classes and scheduling weight specification.

Service class	Id_serv	Output scheduling weight (Wi)
Signaling flits	Class_1	4
Real-time flits	Class_2	3
Short data flits	Class_3	2
Block memory transfer flits	Class_4	1



```
If  event_arrival_flit_port_i = '1' then
        arrival_flit := Get_from_input scheduler( port);
Case mem_occupancy_stat is
 when (memory_state is( ≥ Lower_threshold and & < upper_threshold) =>
        If ( flow_loss_sensitive or (( pr_tag) = high importance))  then

               In_queue_mem (class_serv):= arrival_flit;
        else
            drop( arrival_flit);
        end if;
  when (≥ upper_threshold) =>
        for i from 0 to ClassServ_i do
```

Fig. 4. Memory control strategy and general management pseudo-code algorithm.

waiting in queue flits (WRR). Table 1 sums up the class service defined for this approach and their weight for the output scheduling process.

During the communication process, the occupancy state of memory is controlled by the *Memory occupancy manager* in order to avoid congestion and to generate a signaling flit when required. The management approach associated with this memory is based on the WRED-like technique.[10,22] In fact, it is higher than the lower_threshold waiting in queue-flits, the router will inform its neighbors with a signaling flit to avoid using it as a next hope node in their routing processes. A flit dropping process will, then, start according to the flit importance tag (pr_tag). This parameter is provided by the IPs to each flit according to the importance of the processed data. It will be identified and used when congestion occurs. Figure 4 summarizes the memory control strategy and its general management approach.

6. Performances Evaluation of the Router

To check the performances of the router architecture, designed to enhance QoS and to solve congestion in NoC, we adopted a co-simulation environment based on Simulink-ModelSim environment that allows checking out the behavior of the hardware circuit while evaluating the QoS at the application level. We built up for simulation a 3*3 NoC using the router structure that we presented. A VHDL description of the router function was used for all the routers in the network. Different communication processes were created to load the network and testify the performance of the router. Among these processes, we defined a specific application, through Matlab link, for the transmission of image and its reconstruction at the receiver IP. We also designed the IP adapter to the network that generates flits from the data stream with required format.

In the performance evaluation of the proposed on-chip router, we studied the capacity of the WRED-like algorithm to avoid and to delay congestion in the internal memory of NoC. So we have introduced to one router heavy input processes, coming through different input ports, in order to check out it's resistivity against congestion. Figure 5 shows up the occurrence instant of congestion at the internal memory for two different input rates. In this figure, T1 and T2 represent the instant when the congestion occurs respectively for 0.23 flits/clk and 0.19 flits/clk input rates without the application of the WRED-like algorithm. T3 and T4 represent the occurrence instants for the same event when the WRED-like algorithm is applied. In both of these cases, the congestion will be delayed when we apply the proposed WRED-like algorithm. Figure 5 shows that the congestion was delayed of more than 180 clk cycles in the case of with the case of 0.23 flits/clk input rate and more than 200 clk cycles with 0.19 flits/clk input rate. The delay is bigger in the case of low incoming rate because it gives more time to the output scheduler and the router process to send out more flits. This gives more chances to the network to avoid congestion if a specific signaling protocol is implemented such as the signaling scheme that we explained in this paper.

In Ref. 23, a study addressing the implementation of a new scheme of globally-synchronized frames (GSF) for QoS guaranty in multi-hop on-chip networks was

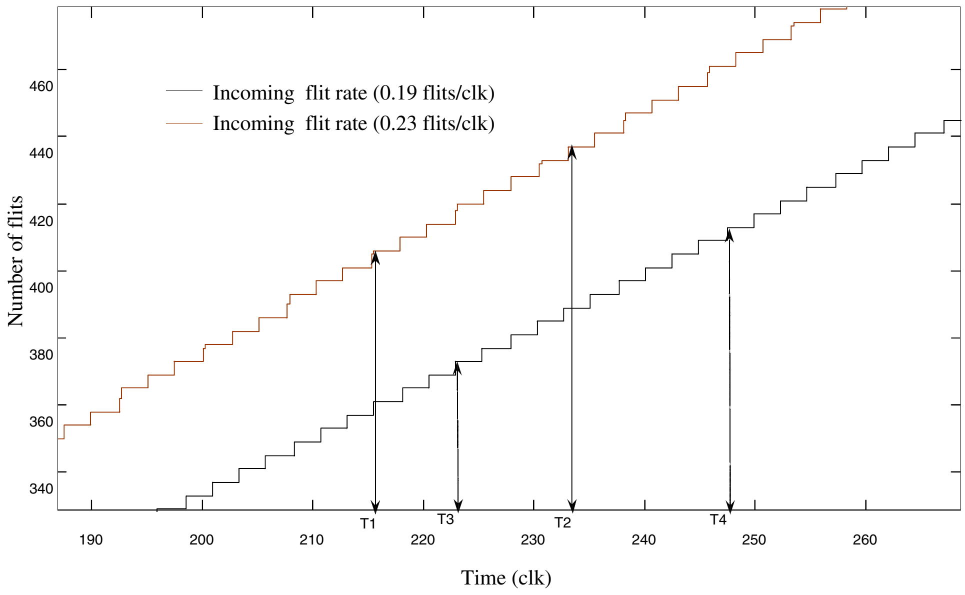


Fig. 5. Congestion delay with WRED-like algorithm.

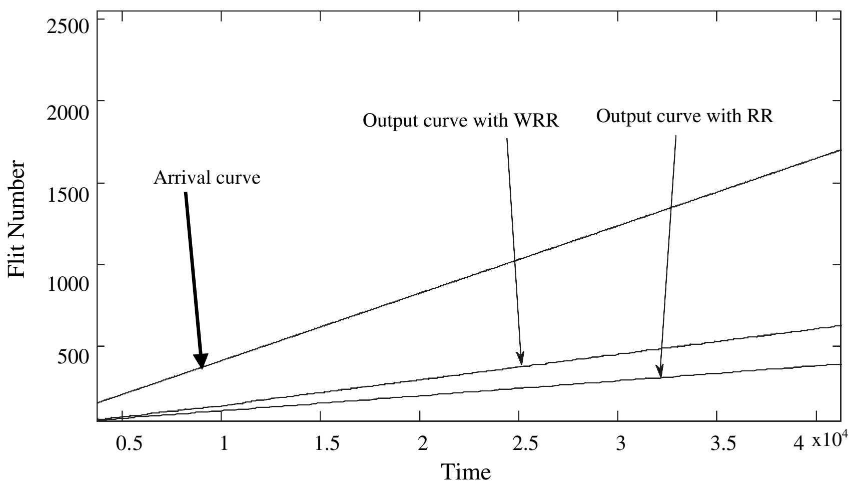


Fig. 6. Arrival and output curves of the router using RR and WRR scheduling algorithms.

presented. This study has reported that the congestion occurs for different sink nodes at an input injection of around 0.22 flit/clk for different arbitration schemes. However this study had not addressed the capability of the router to delay the congestion or to resolve it. Our proposed router gets also congested at the same input injection ratio (0.23 flits/clk, but the WRED algorithm is able to delay in time the occurrence of this congestion, that might enhance the QoS at the reception level.

We also studied the efficiency of our output scheduling scheme performed through the proposed architecture. Figure 6 shows the reactivity of the router in time, for a real time process that is expected to be managed with high priority. This figure contains three curves: the arrival flit curve, the outputs of the router when it performs round Robin (RR) and the WRR-based scheduling schemes. We note that the arrival curve includes all the processes of different class services. We can see from this figure that using the WRR-based scheduling algorithm, the router is able to send out more real time flits, in time, to their destination when compared to the use of RR scheme. We can also note that, in time, the amount of extracted flits from the real time process is increased compared with the RR scheme. This attests about the scalability of this scheme. In fact, when we maximize the output rate, we minimize the probability of congestion and we increase the adaptability of the router to meet real time constraints.

We evaluated the capability of this scheme to enhance the QoS at the end to end application level. Therefore we transmitted an image of 512*512 8 bpp transformed with Haar wavelet transform through a 3*3 NoC architecture and we checked the quality of the image at the reception level according to the number of lost flits. Table 2 sums up the main characteristics of the data streams used for communication during simulation.

Table 2. General description of the data flows.

Id of the communication process	Class of service	Input injection rate in the network (flits/clk)	Source-destination (X–Y)	General description
1	1	Variable from 0.02 to 0.1	(00,00) to (01, 10)	Haar wavelet image data stream.
2	2	Variable from 0.015 to 0.1	(00,01) to (10,01)	These communication processes are shifted in times during simulation. They are used to load the network.
3	3	0.02	(01,00) to (01,10)	
4	4	0.02	(10,00) to (00,10)	
5	1	Variable from 0.015 to 0.1	(10,00) to (01,10)	

The use of the Haar wavelet transform is to get, in the same communication process, data with different importances for the application level (Fig. 7). Other traffics were carried simultaneously in order to create congestion in the network routers. Our algorithm for flit's discarding was applied in the transition nodes, in order to check out its capability to improve the performances of the NoC under congestion state. The idea is to show that the algorithm is able to discriminate low-importance flits and discard them to give more buffer space for flits with high-importance.

The transmitted data that represents the image is the output of the Haar wavelet transform with one decomposition level. After the decomposition we obtain, as shown in Fig. 7, four sub-bands (LL, LH, HL and HH).

The data in the sub-band LL represents the low frequencies and has the highest importance in the image retrieving process. The two sub-bands LH and HL contain data with less importance, for the image reconstruction, compared to the LL sub-band data. The least important data is located in the HH sub-band. The data stream is, then, framed into flits according to the flit structure, specified in the previous section, before being sent into the network. The main concerned parameter after the

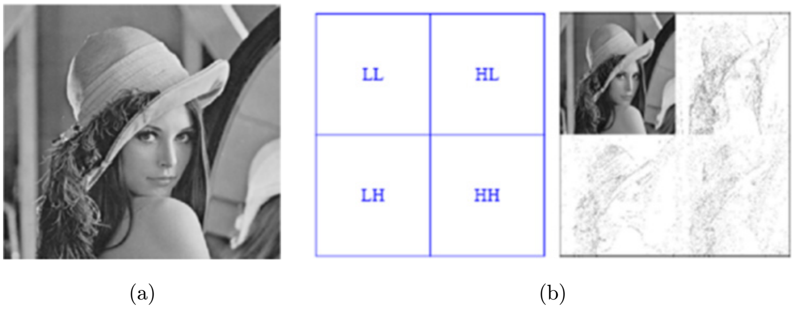


Fig. 7. One level decomposition of Haar wavelet transform applied to image. (a) Lena original image and (b) frequency sub-bands after wavelet transform.

Table 3. Statistics of discarded flits with and without the application of the WRED-like algorithm.

			Without the application of the WRED-like algorithm				With the application of the WRED-like algorithm			
			B1	B2	B3	B4	P1	P2	P3	P4
Discarded flits	Sub-band	LL	0	2	8	25	0	0	0	0
		LH	0	55	91	84	0	0	0	35
		HL	41	20	0	98	0	10	35	40
		HH	0	50	43	1	56	100	75	99

Note: The set of points {Pi, Bi} are those illustrated in Fig. 8.

Haar wavelet decomposition is to show that the algorithm WRED-like is able to distinguish the weight of the flit in the communication process and to resolve the congestion in the routers accordingly. In that manner the algorithm when discarding flits from memory will check out the importance of the flit using their weight in the communication process tag (pr_tag). Table 3 sums up the statistics of discarded flit numbers from the image communication process after crossing the network.

When applying WRED-like algorithm, the discarded flits belong to less important coefficients of the wavelet transform. The algorithm was able to avoid flit losses from the LL sub-band that contain as explained the most interesting part of data for the image reconstruction. The number of discarded flits, with the application of WRED-algorithm, is mainly from less significant part of the image. This number is inversely proportional to the importance of the wavelet sub-band data. This explains well the difference in the PSNR of the reconstructed image with and without the application of the WRED-like algorithm (Fig. 8). Without the application of the WRED-algorithm, Table 3 shows that flits are discarded regardless their importance for the application level.

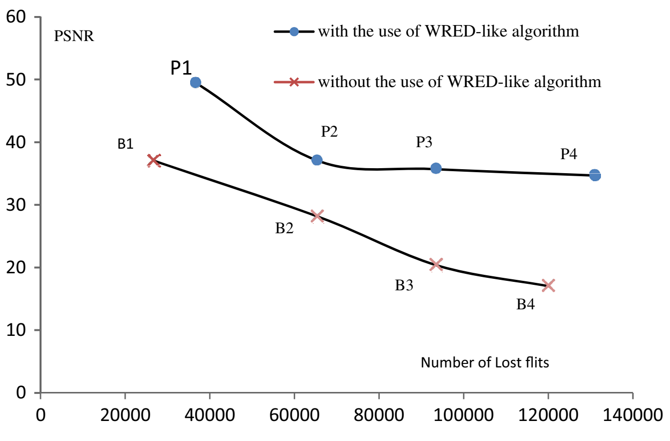


Fig. 8. Quality of the image at the reception level with and without the use of WRED-like scheme.

In Fig. 8, we illustrate the PSNR of the received image after being transmitted in the network under the same traffic load with and without the WRED-like algorithm. The number of lost flits represents the number of whole lost flits in the network including the other communication processes. This figure shows that the WRED algorithm can enhance well the quality of the image at the reception level. We can see that, when the network is heavily congested, the gain in the PSNR may exceed 50% that indicates the efficiency of this approach.

The router performance depends also on the input FIFO depth. As deeper will be the input FIFO as it gives more time for the scheduler to process more flits. As a consequence the router gets more capability to avoid congestion. However, we have to remind that memory, represents the critical part of on chip system. When it increases, it drives up the cost of the circuit and the power consumption. We studied the impact of the memory depth on the QoS at the application level. For that purpose, we followed through similar experience to what has been described previously, the QoS at the output of one router in the NoC. We measured then the PSNR of the image at the output of the router for different input-FIFO depth. Figure 9 illustrates the quality of the image for different input throughputs and for variable sizes of the input-FIFO. It shows that interesting qualities of image are obtained with a FIFO size greater or equal to six. Even with high throughput, the quality of the image will remain good at the reconstruction that attests well the efficiency of the WRED-algorithm to ensure quality for loss sensitive applications. The suitable input FIFO size will be more discussed in the hardware prototyping section to point out the suitable size for low-power consumption.

The performance evaluation of the router, shown in this section, to ensure high QoS for real-time and loss sensitive processes, attests of the efficiency of our approach. In literature review, there were no comparative studies that addressed a similar aspect for router performance's evaluation. We will focus in the hardware characteristics for comparison with other similar solutions.

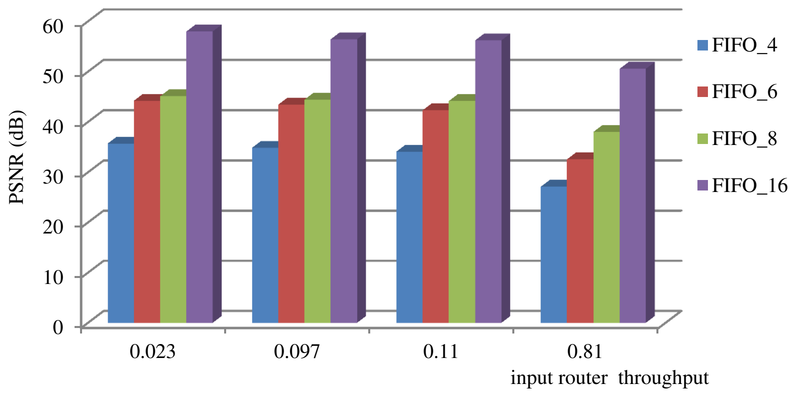


Fig. 9. Quality of the image at the output of NoC router with different FIFO depth.

7. Hardware Prototyping and Circuit Characteristics

To check out the features of the proposed approach, we have designed the NoC Router, as the basic element for the communication in NoC, for CMOS ASIC circuit. We have described on VHDL the functionalities of the different blocks as sets of extended finites states machines (EFSMs) in the RTL level.

For the design of the router circuit, we have used Design Vision for synthesis with CMOS cell libraries and Encounter at the placement and routing steps when performing the layout of the circuit. We have used the Silicon Encounter tool with different integration technologies Standard cell libraries in order to analyze the characteristics of the circuit. In fact we have studied the circuit area and the power consumption of the NoC router using 130 nm, 65 nm and 45 nm standard cell libraries.

Figures 10 and 11 show respectively the circuit area and the dynamic power consumption according to the use of different integration technologies for frequencies

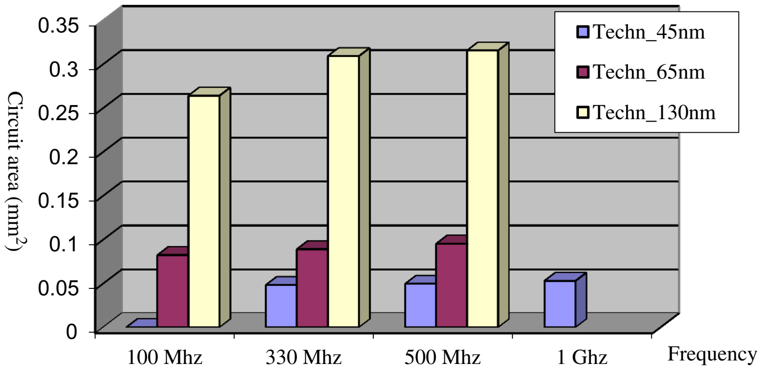


Fig. 10. NoC router area and frequency.

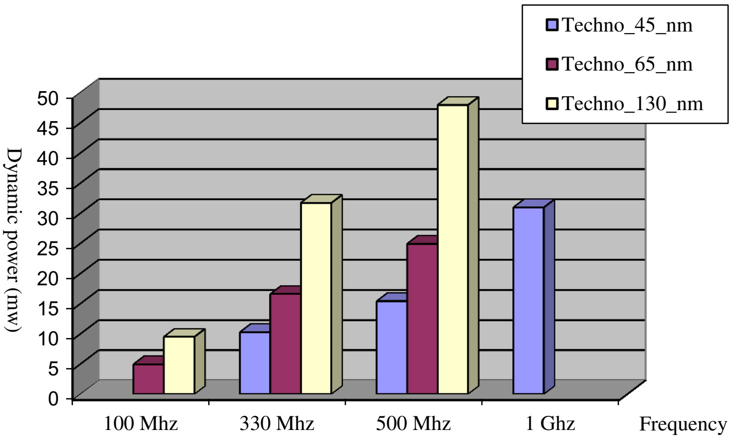


Fig. 11. Power consumption in regard to frequency.

ranging between 100 MHz and 1 GHz. It was impossible to reach 1 GHz with 130 and 60 nm, therefore the power and circuit area missed for 1 GHz in both figures.

These results are obtained with the consideration of central memory depth of 1024 bytes which represents a capacity of four internal service classes in queues of eight flits each one. We see in Fig. 10 that circuit area increases slightly with the increase of the frequency, using the same technology. In fact, the increase is about 0.007 mm^2 between 330 and 500 MHz using 65 nm integration technology. This increase in the router area is not very significant. As a consequence, the router can be used for high-bandwidth communication for on chip communication without reducing the area reserved for applications (IPs) in the system on chip.

For the circuit power consumption, it is mainly driven by the dynamic part. Thus, we have considered in Fig. 11 only this part of the power consumption.

The power consumption is an increasing function with the frequency. So, as we want to increase the bandwidth-link between NoC neighbor's routers in the SoC architecture, in order to allow higher IP possible throughput and short end to end packet-time transaction, we have to remind that the power consumption will increase too.

With 45-nm technology, the power consumption is significantly reduced compared to the use of the 130 nm and 65 nm. In fact, with 45-nm technology, for a frequency of 500 MHz the power consumption is reduced with 67.8% compared to 130 nm and with 38.1% compared to 65-nm technology. It is possible to reach 1 GHz frequency with low power consumption compared to old integration technologies (130 nm and 65 nm). At this frequency, the NoC router will get per link-available bandwidth of about 320 Gbits (frequency * link size). As a general note, the proposed router-circuit is able to achieve high-communication bandwidth while keeping low-power consumption in the NoC. This characteristic elects this solution to be very suitable for power constraint system on chip.

We have studied, with 45-nm integration technology, the relationship between the physical characteristics of the circuit for the proposed router architecture and some internal parameters that manage the QoS such as the depth of the input in queue FIFO and the depth of the internal central memory size. Tables 4 and 5 illustrate respectively the router circuit area and its power consumption in regard to the FIFO depth (unit = flit) and to the internal memory size.

Table 4. Router area (mm^2) in regard to the input FIFO depth and the central memory size (45-nm technology).

		Internal memory size		
		8 flits	16 flits	32 flits
Input-FIFo depth	4 flits	0.025	0.0274	0.329
	6 flits	0.028	0.029	0.354
	8 flits	0.031	0.032	0.384
	10 flits	0.034	0.036	0.417
	16 flits	0.042	0.043	0.495

Table 5. Dynamic power consumption (mw) according to input FIFO and central memory depths (45-nm technology).

		Internal memory size (flits)		
		8 flits	16 flits	32 flits
Input-FIFo depth	2 flits	5.95	6.801	8.2249
	4 flits	7.01	7.8684	9.2813
	6 flits	7.86	8.7308	10.1573
	8 flits	9.01	9.8852	11.2987
	10 flits	10.05	10.9047	12.3182
	16 flits	13.10	13.9641	15.4242

Table 4 shows that for a central memory size of 32 flits (1 Ko), the circuit area is the biggest. The size of the circuit increases as well as the depth of the input FIFO increases, but the difference in circuit area is very high between a router with a central memory size of either 8 flits or 16 flits compared to 32 flits. Although, we have to keep in mind that the router capability to avoid congestions and to ensure QoS depends mainly on the central memory size. In depth, the memory manager will drop out less flits when the router is under congestion with a memory size of 32 flits. This highly impacts the end-to-end QoS for the application level. The size of the input FIFO enhances this capability too. It allows a high throughput flits with lossless-exchange between neighbors. Therefore, for power consumption critical application, the optimal central memory size should be calculated considering the input FIFO depth and also the associated power consumption (Table 5).

Based on these Tables 4 and 5, we can conclude that the optimal characteristics of the NoC router could be a central memory size of 16 or 32 flits and an input FIFO depth of 6 or 8 flits. These hardware's characteristics ensure a tradeoff between the circuit area and the dynamic power consumption associated to these configurations from one side and the router capabilities to manage the QoS from another side.

In Ref. 6, the authors used 90-nm technology to synthetize the router, it clocks up to 425.5 MHz and it occupies around 0.143 mm^2 when using 8 flits as the FIFO(s) size. The power consumption is relatively low in their design (around 2 mw) because they are using the clock boosting concept and they are not using internal memory. Their approach was oriented to reduce the power consumption, but they did not showed at which level it may avoid congestion for heavy traffic data load.

In Ref. 11, Lotfi *et al.* designed a solution based on EDXY routing scheme to load the traffic through the network and to avoid congestion. They showed the router with EDXY routing scheme is able to enhance the latency for high traffic load. The architecture of the router has the same bus size (32 bits). The synthesis results showed that the circuit occupies an area of 0.089 mm^2 and consumes a dynamic power consumption of 26.1197 mw. When compared to our solution, we can see that with 65 nm and 45 nm, our scheme has more attractive characteristics in terms of area and power consumption.

As physical resources occupancy, the NoC router has a circuit area with the same range as the Ethereal NoC router (0.26 mm^2 with input FIFO of 4 flits) when performed with 130 nm integration technology. However, with 45 nm our circuit, of the NoC router circuit has a smaller area than Ethereal, and offer more bandwidth with 1 GHz frequency.

When compared to the XHiNoC[4] with the same integration technology, our solution has a bigger circuit area. In fact, the XHiNoC circuit area is 0.108 mm^2 with 130-nm technology for 4 flits of 38 bits. The maximum frequency of the XHiNoC router is 453 MHz. The main reasons for this difference in circuit area are related to the use of a central memory dedicated for different flit class services management. We think that in addition to the bandwidth, our router offers more granularities in QoS management.

The SoCBuS NoC[24] designed for real time service (1.2 GHz) clocks at the same range of our solution. That means that this latter can also meet high bandwidth for real time applications.

FPGA solution, for embedded multiprocessor systems have been addressed in the CuNoC solution presented in Ref. 25. However, the QoS related to the flit scheduling and congestion management had not been addressed. This explains the high throughput of CuNoC router (more than 500 Mbits for 16 bits data width).

8. Conclusion and Future Work

In this paper, we have addressed the problem of congestion control and QoS insurance in NoC. We proposed an enhanced architecture that allows per-flit differentiation that offers more granularities for in-routers management. The proposed architecture uses an enhanced approach for memory organization that helps to apply per class of service differentiation.

A WRED-like algorithm was proposed for flow control and congestion resolution to get less distortion on the QoS at the reception level. The performances of this approach were checked out through simulation to prove its efficiency. The prototyping of this solution was carried out for ASIC circuit with different integration technologies (130, 65 and 45 nm). The obtained results show that our approach outperforms many similar solutions for on chip communication while ensuring attractive hardware characteristics.

As future work, we think that the proposed architecture can be enhanced by specific cooperative flow-control mechanisms between routers to balance the load and to avoid congestion. We think also that a control of active processes in the router, based on clock gating mechanism, will also enhance the power consumption of the proposed router.

Acknowledgments

This work is supported by the Research Center (project RC121275) of the College of Computer and Information Sciences, King Saud University.

References

1. A. Kumar and R. N. Mahapatra, An integrated scheduling and buffer management scheme for input queued switches with finite buffer space, *Comput. Commun.* **29** (2005) 42–51.
2. E. Bolotin, I. Cidon, R. Ginosar and A. Kolodny, QNoC: QoS architecture and design process for network on chip, *J. Syst. Architect.* **50** (2004) 105–128.
3. E. Rijpkema, K. Goossens, A. Rădulescu, J. Dielissen, J. van Meerbergen, P. Wielage and E. Waterlander, Trade-offs in the design of a router with both guaranteed and best-effort services for networks on chip, *IEE Proc. Comput. Digital Techniq.* **150** (2003) 294–302.
4. F. A. Samman, T. Hollstein and M. Glesner, Networks-on-chip based on dynamic wormhole packet identity mapping management, *VLSI Design* **2009** (2009) 2.
5. Y.-J. Chen, C.-L. Yang and Y.-S. Chang, An architectural co-synthesis algorithm for energy-aware network-on-chip design, *J. Syst. Architecture* **55** (2009) 299–309.
6. S. E. Lee and N. Bagherzadeh, A variable frequency link for a power-aware network-on-chip (NoC), *Integr VLSI J.* **42** (2009) 479–485.
7. C. Wang, W.-H. Hu and N. Bagherzadeh, Scalable load balancing congestion-aware Network-on-Chip router architecture, *J. Comput. Syst. Sci.* **79** (2013) 421–439.
8. C. I. Aci and M. F. Akay, A new congestion control algorithm for improving the performance of a broadcast-based multiprocessor architecture, *J. Parallel Distributed Comput.* **70** (2010) 930–940.
9. L.-S. Peh and W. J. Dally, Flit-reservation flow control, *Proc. Sixth Int. Symp. High-Performance Computer Architecture, 2000. HPCA-6* (2000), pp. 73–84.
10. M. Barbera, A. Lombardo, G. Schembra and A. Trecarichi, Improving fairness in a WRED-based DiffServ network: A fluid-flow approach, *Performance Evaluation* **65** (2008) 759–783.
11. P. Lotfi-Kamran, A.-M. Rahmani, M. Daneshtalab, A. Afzali-Kusha and Z. Navabi, EDXY–A low cost congestion-aware routing algorithm for network-on-chips, *J. Syst. Architect.* **56** (2010) 256–264.
12. J. Wang, H. Gu, Y. Yang and K. Wang, An energy-and buffer-aware fully adaptive routing algorithm for Network-on-Chip, *Microelectron. J.* **44** (2013) 137–144.
13. M. L. Kaddachi, A. Soudani and R. Tourki, Signaling approach for NOC quality of service requirements, *2nd Int. Conf. Signals, Circuits and Systems, 2008. SCS 2008* (2008), pp. 1–5.
14. M. Daneshtalab, M. Ebrahimi, P. Liljeberg, J. Plosila and H. Tenhunen, A systematic reordering mechanism for on-chip networks using efficient congestion-aware method, *J. Syst. Architect.* **59** (2013) 213–222.
15. M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila and H. Tenhunen, CATRA-congestion aware trapezoid-based routing algorithm for on-chip networks, *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012* (2012), pp. 320–325.
16. D. U. Becker, N. Jiang, G. Michelogiannakis and W. J. Dally, Adaptive Backpressure: Efficient buffer management for on-chip networks, *2012 IEEE 30th Int. Conf. Computer Design (ICCD)* (2012), pp. 419–426.

17. J. W. van den Brand, C. Ciordas, K. Goossens and T. Basten, Congestion-controlled best-effort communication for networks-on-chip, *Proc. Conf. Design, Automation and Test in Europe* (2007), pp. 948–953.
18. A. K. Mishra, A. Yanamandra, R. Das, S. Eachempati, R. Iyer, N. Vijaykrishnan and C. R. Das, RAFT: A router architecture with frequency tuning for on-chip networks, *J. Parallel Distributed Comput.* **71** (2011) 625–640.
19. G. Ascia, V. Catania, M. Palesi and D. Patti, A new selection policy for adaptive routing in network on chip, *Proc. 5th WSEAS Int. Conf. Electronics, Hardware, Wireless and Optical Communications* (2006), pp. 94–99.
20. M. Palesi, S. Kumar and R. Holsmark, A method for router table compression for application specific routing in mesh topology NoC architectures, *Embedded Computer Systems: Architectures, Modeling, and Simulation* (Springer, 2006), pp. 373–384.
21. L. M. Ni and P. K. McKinley, A survey of wormhole routing techniques in direct networks, *Computer* **26** (1993) 62–76.
22. L. David, M. Sood and M. K. Kajla, Router based approach to mitigate DOS attacks on the wireless networks, *Proc. 2011 Int. Conf. Communication, Computing & Security* (2011), pp. 569–572.
23. J. W. Lee, M. C. Ng and K. Asanović, Globally synchronized frames for guaranteed quality-of-service in on-chip networks, *J. Parallel and Distributed Comput.* **72** (2012) 1401–1411.
24. D. Wiklund and D. Liu, SoCBUS: Switched network on chip for hard real time embedded systems, *Proc. Int. Parallel and Distributed Processing Symposium, 2003* (2003), p. 8.
25. S. Jovanovic, C. Tanougast, S. Weber and C. Bobda, Cunoc: A scalable dynamic noc for dynamically reconfigurable fpgas, *Int. Conf. Field Programmable Logic and Applications, 2007. FPL 2007* (2007), pp. 753–756.