Nano Communication Networks 26 (2020) 100321

Contents lists available at ScienceDirect

Nano Communication Networks

journal homepage: www.elsevier.com/locate/nanocomnet

Traffic-aware performance optimization in Real-time wireless network on chip

Mohammad Baharloo [a,*], Ahmad Khonsari [a,b], Mahdi Dolati [b], Pouya Shiri [c], Masoumeh Ebrahimi [d], Dara Rahmati [a,e]

[a] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
[b] School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran
[c] School of Electrical and Computer Engineering, University of Victoria, BC, Canada
[d] KTH Royal Institute of Technology, Sweden
[e] School of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran

ARTICLE INFO

Article history:
Received 27 February 2020
Received in revised form 14 July 2020
Accepted 25 August 2020
Available online xxxx

Keywords:
Wireless network on chip
media access control
Real-time communication
Traffic distribution
Load balancing

ABSTRACT

Network on Chip (NoC) is a prevailing communication platform for multi-core embedded systems. Wireless network on chip (WNoC) employs wired and wireless technologies simultaneously to improve the performance and power-efficiency of traditional NoCs. In this paper, we propose a deterministic and scalable arbitration mechanism for the medium access control in the wireless plane and present its analytical worst-case delay model in a certain use-case scenario that considers both Real-time (RT) and Non Real-time (NRT) flows with different packet sizes. Furthermore, we design an optimization model to jointly consider the worst-case and the average-case performance parameters of the system. The Optimization technique determines how NRT flows are allowed to use the wireless plane in a way that all RT flows meet their deadlines, and the average case delay of the WNoC is minimized. Results show that our proposed approach decreases the average latency of network flows up to 17.9%, and 11.5% in 5 × 5, and 6 × 6 mesh sizes, respectively.

1. Introduction

The wireless network on chip (WNoC) paradigm facilitates communication among a large number of cores in an embedded system, where, traditionally a wired network known as NoC was the only means of communication. Previous studies (*e.g.*, [1,2]) have demonstrated the implementation viability and the effectiveness of the WNoC technology, however, this promising approach is not adequately employed to address the ever-increasing challenges in the design of embedded systems. On the other hand, the emergence of complex multi/many-core processors enables the integration of a broad range of functionality on an embedded system, which imposes new challenges to satisfy the desired quality of service (QoS) level for each of them. There are two main category of real-time (RT) and non real-time (NRT) dataflows in these systems, where, the RT flows are time-critical and extremely sensitive to the worst-case delay and bandwidth. NRT flows, on the other hand, do not have stringent communication constraints. Still, it is possible to improve the QoS of NRT flows by reducing their average delay and allocating more bandwidth to them. Therefore, performance parameters, including the average and worst-case metrics, are crucial design targets in multi-core systems.

Most of the proposed designs in the domain of WNoCs [3–9] have focused on average performance of the network. These approaches cannot cover RT constraints, which are crucial for real-time systems; hence they do not cover all the challenging aspects of WNoC performance. On the other hand, the few works that deal with QoS in WNoC [10–14] do not consider average parameters simultaneously with their worst-case counterparts. These approaches have concentrated on the RT flows and have not considered the performance parameters of NRT flows in the network.

To the best of our knowledge, this is the first work that considers the average-case and the worst-case performance evaluation for WNoC simultaneously. Our proposed approach focuses on performance parameters of both RT and NRT flows. This approach reduces the average network latency while ensuring that the worst-case delay does not exceed its predetermined limit. To this end, we design an optimization model to determine the exact fractions of NRT flows which can use wireless links in

* Corresponding author.
E-mail addresses: m.baharloo@ipm.ir (M. Baharloo), ak@ipm.ir (A. Khonsari), mahdidolati@ut.ac.ir (M. Dolati), pouyashiri@uvic.ca (P. Shiri), mebr@kth.se (M. Ebrahimi), dara.rahmati@ipm.ir, d_rahmati@sbu.ac.ir (D. Rahmati).

https://doi.org/10.1016/j.nancom.2020.100321
1878-7789/

a way that all RT flows meet their timing deadlines. The proposed optimization model investigates the overall status of the network regarding average-case and worst-case timings of both RT and NRT flows. The main contributions of the paper can be summarized as follows:

(1) We design a scalable centralized arbiter to efficiently perform the medium access control in a wireless-enabled NoC with mesh topology.
(2) We consider a general workload model with realtime (RT) and non-realtime (NRT) traffic flows, where RT flows have higher priority and lower data generation rate. Then, we rigorously characterize the analytical worst-case delay model of the proposed arbiter architecture, when the RT and NRT flows, respectively, use the wireless and wired planes.
(3) We design an optimization model to distribute RT and NRT traffic flows between the wireless and wired planes to improve the resource utilization and average-case delay. The proposed model handles two common scenarios that are not considered in the routing scheme where the RT flows always use the wireless plane: (1) The injection rate of the RT flows exceeds the capacity of the wireless plane, which leads to the over-utilization of the wireless links and excess latency and (2) The injection rate of the RT flows is lower than the capacity of the wireless plane, which leads to the under-utilization of the wireless links and waste of resources. Our proposed model is computationally efficient and is able to split the network flows optimally between the wired and the wireless plane based on the average-case and the worst-case delays. Through the optimization problem, we minimize the *maximum average latency* of NRT flows while respecting the deadlines (*i.e.,* the worst-case delay) of the RT flows. In this way, we guarantee that the delay of any flow will not exceed the minimum value obtained through the optimization process.
(4) We use real-world applications to extensively evaluate the latency improvement and area-overhead of our proposed architecture and optimization model.

1.1. Paper organization

The remainder of the paper is organized as follows: Section 2 describes the related works. We describe a generalized network model in Section 3 and a model that characterizes the worst-case performance of a WNoC in Section 4. In Section 5, we propose our optimization model for improving the worst and average delays. Section 6 describes our evaluations, and finally, Section 7 concludes the paper.

2. Related works

We categorize the existing works in this domain into two groups: (1) Modern architectures that consider the shortcomings of the wired-only NoC with an emphasis on those that address the applicability and implementation of WNoCs; and (2) Mathematical performance optimization techniques that try to analytically analyze and optimize the NoC performance.

Modern NoC architectures. In response to the ever-growing integration levels and increased number of cores on a chip, the area of scalable and high-performance on-chip communication has become an active field of research. Optical interconnects, three-dimensional integrated circuits (3D ICs), and wireless interconnects are the most important alternatives to the traditional planar metal interconnects which cause multi-hop communications between distant blocks on a chip [15,16]. Optical architectures, though, suffer from the high design complexity and high area overhead that are imposed by the modules that convert an electrical signal to light and vice versa. 3D architectures are CMOS compatible. However, severe issues such as low yield, high temperature, and alignment complications limit their performance gain [17,18]. Wireless interconnects, on the other hand, provide a viable solution to the latency and power inefficiency of multi-hop communication employed in current wired-based NoCs [19–22]. Moreover, due to the emergence of data-intensive applications, the need to support multicast on-chip traffic to meet the required higher performance levels has become one of the principles of many-core chip design [10,23]. Indeed, a little increment in multicast traffic, as it imposes unicast traffic and consequently creates congestion, may lead to a substantial throughput loss in on-chip communications. For improving the throughput of multicast communications, network coding has shown as (NC) has shown a viable solution. In this scheme, by the cooperation of on-chip nodes and a coding unit, multiple packets can be combined into a new one. In this way, the network load is reduced, leading to improved link utilization and network throughput. Authors in [23] have proposed a cooperative NC approach, which realized through a corridor routing algorithm (CRA) and adaptive flit dropping (AFD) scheme to support NC-based multicast, avoiding congestion, and saving power. To further improve the throughput of on-chip collective communication (i.e., broadcast and multicast), wireless NoC seems to be a right option, which inherently is broadcast/multicast friendly communication technology. Toward this end, authors in [10], by integrating the congestion-aware routing protocol and network coding techniques, provide a multicast-aware wireless NoC, that could handle the high volume of multicast traffic along with the reduction of energy dissipation of the chip.

Among the existing solution proposals for the implementation of wireless interconnects, wireless RF paradigm is a scalable, simple, and flexible option with broadcast capability [16,24]. This capability is a crucial factor for applications like cache coherence protocols, which can significantly benefit from multicast or broadcast communications [3]. It is possible to realize On-chip wireless communication at the core level through a cost-effective RF paradigm by employing non-intrusive solutions and eliminating the transmission lines, which reduce the chip floorplanning costs. On the other hand, QoS is a crucial issue for many application domains wherein traffic flows with real-time requirements exist. In [25], authors try to realize guaranteed services through best-effort mechanisms in NoCs. The architecture proposed in [26] exploits the TDMA technique in packet-switched networks to provide real-time guaranteed services. In [27], an analytical method for calculating worst-case traffic delay in best-effort NoCs without special hardware provisioning is presented. Many other works have been proposed to provide real-time guaranteed services [3,28–30]. However, none of them calculate the worst-case delay and bandwidth in hybrid wire/wireless NoCs. Recent works have demonstrated that RF circuits operating at 100 GHz and above are realizable [31]. As well, they confirm that constructing small-footprint antennas and other components that operate at high frequencies are achievable [32,33].

Media Access Control (MAC) Protocol is one of the main components of wireless NoC that control the access of competing wireless nodes to the wireless channel. Prior works demonstrated that the use of conventional MAC protocols such as FDMA, CDMA, and token passing are inefficient in terms of channel utilization, area, and energy dissipation [34]. In [35], the combination of TDMA and FDMA at the expense of multiple wireless transceivers in each node is reported. In [34] a distributed MAC protocol has been proposed for millimeter-wave smallworld wireless Network-on-Chip (mSWNoC). In mSWNoC architecture, only a few numbers of distant nodes equipped to

the antenna to construct long-range wireless shortcuts. So, the Authors provide a MAC scheme that benefits from the fairness of token-passing protocol, which utilizes a simple orthogonality code for requesting wireless channels. In this scheme, the grant transmission is not needed due to the distributed processing of requests at each wireless node. A distributed priority multicast MAC protocol is published in [2,35]. In this scheme as the number of wireless interconnects increases, the channel access time is increased, so it is not suitable for a large NoC with a large number of nodes equipped with a wireless antenna.

One of the essential factors in the performance of wireless NoC is determining the location of antennas [36]. Indeed, if there are a limited number of wireless interconnects, the placement of these links must be optimized concerning traffic patterns. In this case, adopting a dead-lock free routing protocol is challenging and imperative. However, in this work, as each on-chip router is equipped with an antenna, the problem of finding the optimum locations for deploying antennas will not exist. Also, as packets in wireless plane travel to their destinations through one hop, the frequent shuttling of packets between the wired and wireless planes will not occur. In this way, utilizing the traditional on-chip routing protocols is a proper option for providing a deadlock-free routing protocol.

Mathematical performance optimization. Optimization techniques are commonly used for increasing the performance of NoCs [37]. Manna et al. [38] formulated the thermal-aware application mapping in NoCs as an Integer Linear Program (ILP) to reduce the peak temperature and the sum of Manhattan distances between each pair of source and destination. Coskun et al. [39] utilized an ILP model to compute an NoC task scheduling that achieves the best possible temperature profile while meeting the deadline and dependency constraints of the tasks. An optimization program for minimizing the peak NoC temperature that considers the applications hard deadlines was proposed in [40]. None of these works consider the wireless connections in their NoC models. A makespan minimization model for task scheduling in wireless NoCs was designed in [41]. However, they consider a load-independent delay model for the wireless and wired routers. Kim et al. [42] modeled the problem of Multiple Voltage Frequency Island (VFI) clustering in wireless NoCs as a $0-1$ quadratic programming. They use two different constant values to distinguish the cost of inter- and intra-cluster communication delay, which is assumed to be proportional to the number of hop counts. Authors in [43] have demonstrated that the traditional analytical approaches fail to capture some critical characteristics of traffic which emerge in heterogeneous networks. So, they proposed a model that was enabled to analyze the multi-fractal and non-stationary behavior of on-chip traffics. Bogdan et al. have revealed that the multi-fractality and non-stationary behavior of the on-chip computation and communication workloads have a profound impact on temperature regulation, voltage/frequency-island partitioning, and dynamic power management of a chip [44]. Consequently, he has provided a complex dynamics model for data-center-on-a-chip (DCoC) platforms for regulating chip dynamics as a function of frequency.

3. The network model

In this section, we elaborate our router model which is the key determinant component to characterize network behavior. We employ the generic architecture shown in Fig. 1 that is integrated with the wireless transceiver to provide wireless communication for each core [3]. Considering RF research, applicable transceivers can be achieved by down-scaling power and area characteristics along with communication frequency [45,46]. Studies show that quadratically reduced passive elements can be used as transceivers in fully WNoC platforms with 1 GHz frequency [45,47,48]. In [47], a transceiver is presented in 65 nm CMOS technology operating at 1.7 GHz with 0.34 mm^2 and 98 mW area and power consumption characteristics, respectively. Transceivers serialize and modulate processor communications at a predefined frequency in the sender side. The reverse operation occurs at the receiver side.

Table 1
Network model parameters and symbols [27].

Parameter	Description
$Freq$	Clock rate of the system
a	Number of registers used to segment NoC links
b_1	Input buffer depth
b_2	Number of pipeline stages in the router (0 if combinational)
b_3	Output buffer depth
b	$:= b_1 + b_2 + b_3$
B_d	Buffer depth, $:= a + b1 + b2 + b3 = a + b$
ts_1	Packet injection latency at the source
ts_2	Packet ejection latency at the destination
$FlitWidth$	Links width of NoC (in bytes)
SW_j	The Switch with index j

For generality, we consider a router with optional input and output buffering schemes where for each physical link some virtual channels are associated to multiplex the available bandwidth of the wired connection. As in most of the on-chip applications, packet dropping is not tolerable, we exploit the flow control mechanism that applies back-pressure to avoid packet dropping.

Authors in [46] proposed a wireless many-core architecture called Replica, for speed-up communication-intensive ordinary data. In Replica, a dynamic MAC protocol has been introduced that can dynamically switch between two different schemes, *i.e.*, *carrier sensing* and *token passing* based on the wireless network utilization pattern which could vary across different applications and also within an application. However, in this architecture, by considering approximation techniques, packets could be selectively dropped for compromising accuracy for reduced communication size.

To provide a fair, reliable and deterministic medium access for all the cores on the chip, we consider a centralized controller, shown in Fig. 2, to perform the media access control function in a contention-free manner. When a core wants to send packets through the wireless network, the MAC module should send a *request* signal to the *C-MAC* through the wired control plane and wait for a *grant* signal. The wired control plane is constructed as follows: A pair of wires are routed from each router to the C-MAC module, one for the *request* signal and the other for the *grant* signal. The feasibility of routing such metallic wires to drive signals across the chip within a single cycle is comprehensively explained in [49].

Table 1 shows the network parameters that are required to describe the network model, which are adopted from [27] and extended towards our analysis. We define the parameter *Freq* and *FlitWidth* as the operating frequency of all the cores and the width of all the on-chip data links, respectively. As shown in Fig. 1, the buffer depth parameter (*i.e.*, B_d) represents the aggregate number of buffers and registers between the arbitration points of switch j and switch $j+1$. For simplicity, throughout the paper, we assume the intermediate buffers and registers between the arbitration points of two adjacent switches to be lumped, so we refer to the output and input buffers of two adjacent switches equivalently. The parameters b_1 and b_3 define the input and output buffer depth of the switch, respectively. Note that the register depth of the input buffer is assumed to be at least one. The number of pipeline stages (registers) in the crossbar is denoted by b_2 (if pipelined). For generality, we assume that data links between two adjacent switches can be pipelined, so we define parameter

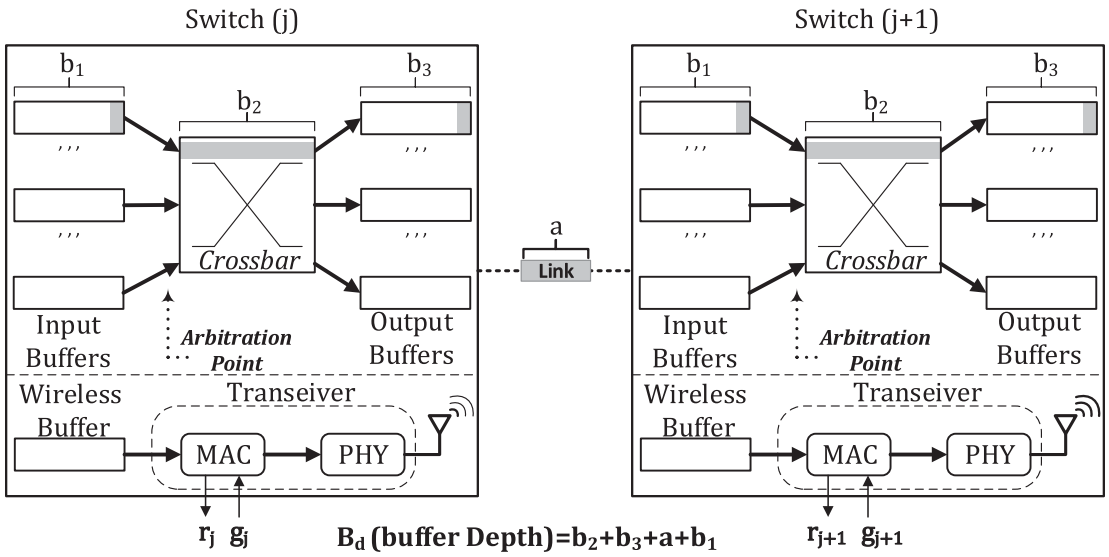


Fig. 1. Switch model and parameters [27].

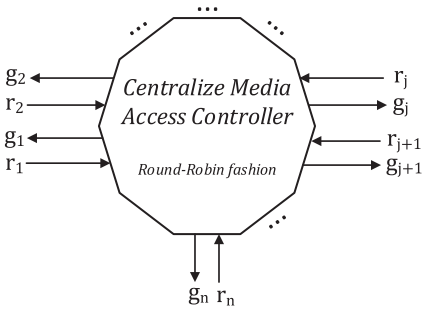


Fig. 2. Centralized Media Access Controller (C-MAC) based on Round-Robin arbitration for wireless communications.

a as the number of registers along with each on-chip data link. By this consideration, the propagation delay of the wires can be compensated to boost the operating frequency of the NoC.

It is important to note that packets will always experience the latencies characterized by parameters a and b_2, and the latencies characterized by parameters b_1 and b_3 are seen by the packets only in the case of congestion. Note that, in the absence of congestion, input and output buffers can be traversed in a single cycle. We index the switches in the path of a flow by $j = 1 \ldots m$, while for source conflict modeling (*i.e.,* sending more than one flow from the source) we also use $j = 0$ which represents a virtual switch inside the source node. To model the latency overhead of injecting a packet into the network at source and ejecting it at the destination, we use parameters ts_1 and ts_2, respectively. To use finite parameters, we assume that the receiving nodes are able to accept incoming data at any required rate.

4. Analytic model of worst-case delay

We use the parameters listed in Table 2 to describe the NoC flows, while the parameters that are required to model the performance of such flows are summarized in Table 3. We assume the path of all flows are predefined and deterministic, so packets do not experience deadlock and livelock along their routes from a source to a destination. Switches perform arbitration through the Round-Robin fashion, so we have a confined maximum delay, *i.e.,* there is no starvation despite the assumption that the current flow served last. The upper bound delay for a packet of a specific flow F_i is denoted by UB_i. Also, parameter MI_i denotes the maximum guaranteed interval after which the output buffer of a specific switch becomes free for subsequent injection. Similar to UB_i, we define the parameter wUB_i as the upper bound delay for transmitting a packet of flow F_i through the wireless network.

Table 2
Traffic model parameters.

Parameter	Description
F_i	ith traffic flow in the network
L_i	Length of the packets in F_i, in flits
S_i	Source node of F_i
D_i	Destination node of F_i
p_i	A packet in F_i
h_i	Number of switches (hops) along the path of F_i

4.1. Wired delay model

Here we calculate the parameter UB_i (worst-case network traversal latency) and MI_i (maximum injection interval in worst-case situation) while three cases is considered: (a) $B_d = L$, (b) $B_d < L$, and (c) $B_d > L$.

4.1.1. Case $B_d = L$

By considering $B_d = L$ we assume that one packet can fill up the buffering space between two adjoining switches. For visualizing the worst-case situation, we assume the network is fully loaded. By this assumption, all switches arbitrate every L cycles simultaneously. In this case, packets follow each other and fill up the buffers just as they become free.

To be more specific, when packet P_i is generated by S_i, in the worst-case condition, this packet experiences congestion along its route assuming that all intermediate buffers are filled by packets of other flows. In this situation, all these packets must go through and free their buffers, before P_i can proceed to the next router. For instance, when P_i reaches switch j, it experiences $z_c(i, j)$ times arbitration losses, due to conflicts with other contending flows at output channel c of switch j. In other words, while we consider round-robin arbitration, all other contending flows must send a packet prior to P_i for realizing worst-case analysis. Evidently, the order in which contending flows transmit their packets does not affect the worst-case delay calculation of P_i. So once a flow

Table 3
Performance model parameters [27].

Parameter	Description
UB_i	Upper bound delay for a packet in flow F_i
MI_i	Maximum inter-packet-injection time of flow F_i
mI_i	Minimum inter-packet-injection time of flow F_i
u_i^j	The time needed for P_i to go from the input buffer of SW_j ($j > 0$) or from generating process in S_i ($j = 0$) to the input buffer of SW_{j+1} ($0 \le j \le h_{i-1}$) or D_j ($j = h_i$)
U_i^j	The time needed for P_i to go from the output buffer of SW_j ($j > 0$) or from the output buffer of S_i ($j = 0$) to the output buffer of SW_{j+1}
$z_0(i, 0)$	Number of flows contending with F_i at S_i
$z_c(i, j)$	Number of flows competing with F_i at SW_j to access output channel c
$I(x)$	Index of the xth flow contending with F_i at S_i (or SW_j)

wins the arbitration it sends a packet. While the packet goes through, the buffer space is freed flit by flit and the buffer content is smoothly replaced with the flits of a packet from another contending flow. Eventually, P_i can also make one hop progress.

The parameter u_i^j represents the latency for the traversal of P_i from the input buffer of switch j to the input buffer of switch j + 1, except for the last switch. Due to considering source conflict (originating more than one flow from a source), we define the parameter u_i^0 which represents the latency overhead of injecting P_i by S_i to be placed in the input buffer of the first switch of F_i. At the destination (last switch), P_i should be ejected. So we have a latency overhead of ejecting P_i, *i.e.,* latency of putting P_i into the input buffer of its destination D_i. Considering the fixed latency for creating and ejecting P_i, we can calculate the UB_i from Eq. (1).

$$UB_i = ts_1 + ts_2 + \sum_j u_i^j \quad , j = o \dots h_i \tag{1}$$

The time period in which S_i can inject packets is the latency overhead of creating a packet, plus the latency overhead for this packet to make progress to the input buffer of the first switch. This time is represented by MI_i, and described as:

$$MI_i = ts_1 + u_i^0 \tag{2}$$

To model the latency overhead of traversing a packet from the output buffer of switch j to output buffer of switch $j+1$, we define the uppercase U_i^j parameter.

Now, let us calculate the latency overhead for the traversal of a packet P_i from its generating source S_i to the input buffer of the first switch along its route. This progress can occur after all existing packets leave the input buffer of the first switch. such existing packet either belongs to the current flow F_i or contending flows at the output port of S_i. The worst-case latency for traveling any existing packet to free up the buffer space is given by $MAX_x(U_i^0, U_{I(x)}^0)$, where $I(x)$ is the index of flows contending with F_i at the output channel of S_i. Of course, all contending flows with F_i must send a packet before P_i in a worst-case analysis. So the delay for P_i to reach the input buffer of the first switch along its route is given by:

$$u_i^0 = MAX_x(U_i^0, U_{I(x)}^0) = \sum_x U_{I(x)}^0 \quad , x = 1 \dots z_0(i, 0) \tag{3}$$

We can calculate u_i^0 for subsequent hops similarly:

$$u_i^j = MAX_x(U_i^j, U_{I(x)}^j) = \sum_x U_{I(x)}^j \quad , x = 1 \dots z_c(i, j), 1 \le j \le h_i \tag{4}$$

Note that, in the case where there is no contending flow with F_i, then the above equation can be reduced to $u_i^j = U_i^j$. This results from the fact that packets move based on a pipeline fashion through the network.

For calculating U_i^j values, we consider a packet P_i traverses from the output buffer of switch j to the output buffer of switch $j + 1$. This traversal can occur if any existing packets at output buffer of the switch $j + 1$ move to the output buffer of switch $j + 2$. Similar to the above calculation, The latency overhead of such a movement in worst-case situation can be obtained from $MAX_x(U_i^{j+1}, U_{I(x)}^{j+1})$. In this equation, we can see that the upper bond delay at switch j depends on the delay values at switch $j + 1$. Thus, we can calculate the U_i^j values through the following equation:

$$U_i^j = MAX_x(U_i^{j+1}, U_{I(x)}^{j+1}) = \sum_x U_{I(x)}^{j+1} \quad , x = 1 \dots z_c(i, j + 1),$$
$$1 \le j \le h_i - 1 \tag{5}$$

At the last switch, the packet is ejected from the output buffer. By assuming one flit per cycle ejection rate, a packet can be ejected in L_i cycles. Thus,

$$U_i^{h_i} = L_i \quad , \quad U_{I(x)}^{h_{I(x)}} = L_{I(x)} \tag{6}$$

4.1.2. Case $B_d < L$

The second case considers $B_d < L$. In this case, each packet occupies more buffering space than buffering space between the arbitration points of two adjacent switches. Therefore, a new parameter δ_i^j is introduced which shows the worst-case delay for a packet to leave a router. Indeed this parameter represents the number of cycles between the moment that header flit of P_i enters the arbitration point of switch $j + 1$ to the moment that the tail flit leaves the arbitration point of switch j. This parameter, as is presented in Eq. (7), can extend relations as mentioned earlier in Eqs. (1) to (6). The δ_i^j parameter would be zero for the case $B_d = L$.

$$\delta_i^j = \begin{cases} \sum_{k=1}^{s} u_i^{j+k} & \text{when } j + s \le h_i \\ \sum_{k=1}^{h_i - j} u_i^{j+k} + (j + s - h_i) \times B_d & \text{when } j + s \ge h_i \,,\ j = 0 \dots h_i \end{cases} \tag{7}$$

Considering the $B_d < L$ assumption, when the header flit of the packet resides in the arbitration point of switch $j+1$, the rest of flits of that packet reside in $S = L - B_d - 1$ latter switches and has not passed the arbitration point of switch j. Therefore, the header must at least traverse S switches to ensure that the tail flit has passed the arbitration point of switch j. The $j + s \le h_i$ condition in δ_i^j occurs when remaining switches of the flow i is smaller than S.

In the case $B_d < L$, the U_i^j parameter is calculated from Eq. (8). It shows the worst-case delay between entering the header flit of P_i to the arbitration point of switch $j + 1$ and the time when the tail of the packet leaves that point.

$$U_i^j = MAX_x(U_i^{j+1} - \delta_i^{j+1}, U_{I(x)}^{j+1} - \delta_{I(x)}^{j+1}) = \sum_x U_{I(x)}^{j+1} + \delta_i^{j+1} \quad ,$$
$$x = 1 \dots z_c(i, j + 1), \ 1 \le j \le h_i - 1 \tag{8}$$

and

$$U_i^{h_i} = L \tag{9}$$

4.1.3. Case $B_d > L$

The third case considers $B_d > L$ and buffering space can be filled with multiple packets. This situation can be imagined by putting the proper number of dummy switches between the

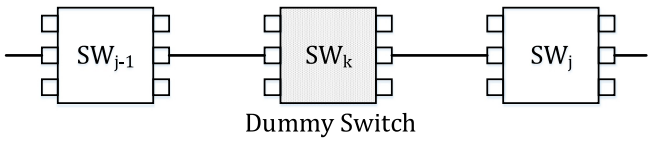


Fig. 3. Dummy switch insertion in the case $B_d = 2L$.

Table 4
Wireless network model parameters and symbols.

Parameter	Description
wUB_i	Upper bound delay for sending a packet P_i of F_i through the wireless network
ART	Worst-case latency for acquiring a grant from C-MAC
t_r	Latency overhead for *request* signal propagation
w_g	Worst-case waiting time for grant at C-MAC
t_g	Latency overhead for *grant* signal propagation
t_p	Latency overhead for transmitting a packet through wireless network
n	Number of real-time (RT) flows

buffering space of adjacent switches. For example $B_d = 2L$ shows that the buffering space between two adjacent switches can be occupied with two packets of flow F_i. This case is depicted in Fig. 3 while switch k is a dummy switch at the input buffer of switch j and the buffering space between switch k and switch j is L. So, independent of packet flows in the buffer, the $u_i^k = u_i^j$ equation always holds.

Generally, when $B_d = m \times L + k$ where $0 < k < m$, demonstrates that one switch can be divided into $m+1$ switches, i.e., one real switch with m dummy switches (by rounding up B_d to $(m+1) \times L$). So, UB_i and MI_i become:

$$UB_i = ts_1 + ts_2 + (m+1) \times \sum_j u_i^j, \ j = 0..h_i \tag{10}$$

$$MI_i = ts_1 + u_i^0 \tag{11}$$

4.2. Wireless delay model

In order to characterize the performance of wireless communications in our WNoC architecture, we define some parameters which are listed in Table 4. t_r is the latency overhead for propagating the *request* signal from a node to the C-MAC module, t_g is the latency overhead for propagating *grant* signal from C-MAC module to applicant node, w_g is the elapsed time between the arrival instant of the *request* signal and the preparation instant of the *grant* signal by C-MAC, and t_p is the time to transmit a packet through the wireless network. t_p represents the effective delay that is observed in the physical environment. Thus, this value incorporates information about delays that arise from transmission error and packet corruption. Note that, due to the central arbitration, medium access is contention-free. Therefore, the wireless interference-related delay does not apply to our model.

We define the parameter ART which represents the arbiter response time that calculates the time interval between the instant of sending a request to the C-MAC and the instant of returning the *grant* signal. For calculating ART, we calculate the worst-case waiting time for a grant at C-MAC which can be achieved through Eq. (12).

$$w_g = (n-1)t_g + (n-1)t_p \tag{12}$$

Based on Eq. (12), the value of ART is achieved through Eq. (13),

$$ART = t_r + w_g + t_g \tag{13}$$

The worst-case time needed to acquire the *grant* signal is the summation of three factors; The time to send a grant by the C-MAC to all applicant nodes which have sent their requests beforehand, the time needed for sending their packets, *i.e.*, w_g and one request propagation latency and one grant propagation latency for the current node. According to Eq. (13), the calculation of wUB_i can be done through Eq. (14).

$$wUB_i = ts_1 + ART + t_p + ts_2 \tag{14}$$

The upper bound delay for sending a packet through the wireless network consists of the time for injecting the packet to the wireless buffer at the source (ts_1), the worst-case latency for acquiring a grant from C-MAC (ART), the latency overhead for transmitting the packet through wireless network (t_p), and the time needed to eject the received packet from the wireless buffer at the destination (ts_2).

5. Delay optimization model

The analysis presented in the previous section provides the insight required for guaranteeing the worst-case delay of RT packets. Accordingly, it is possible to achieve a pre-specified worst-case delay by limiting the number of cores that are allowed to send their packets through the wireless plane. Worst-case delay performance, however, indicates very little about the average-case delay performance of the system. Particularly, it is desirable to consider the packet generation rate as well as the distance between the source and destination of the flows, and accordingly determine the optimal packet routing option which not only respects the worst-case delay threshold but also attains the best average-case performance.

In this section, we focus on the problem of minimizing the average delay of the system, under the assumption of stationary Markovian RT and NRT traffic flows. Furthermore, we assume that the service rate of the arbitration module and wired routers, which follow a deterministic distribution, are known. This assumption along with a deterministic routing algorithm (*i.e.*, X-Y routing) allows us to efficiently compute the latency in different parts of the system. To this end, we provide a method to distribute RT and NRT traffic flows between the wireless and wired network planes. Specifically, we distribute RT and NRT traffic flows in proportion to their volumes between wireless and wired network plane in a way that the average latency of the wired network is minimized, while we meet all deadlines of the RT flows. Here, another underlying assumption is that we are able to split a flow into two arbitrary portions, to be sent through wired and wireless networks, respectively. In practice, the precision of achieving a split portion is a function of number of packets and their lengths. Although we can get very close to the desired portion value, a difference between theory and practice is expected which might adversely affects the performance. We further discuss the limitations and restrictions of the model at the end of the section.

5.1. Problem definition

In this sub-section, we formulate the problem of optimizing the distribution of traffic between the wired and the wireless network planes. Our primary goal is designing an optimization process to reduce the average latency of the network flows while the average-case and worst-case delay of any single flow does not exceed a pre-specified and application-dependent maximum threshold. This objective is achieved by minimizing the maximum average delay of the network flows. In addition to the physical constraints of the problem, which include the service rate of routers in the wired plane and the capacity of wireless links, the

Table 5
Optimization problem parameters and symbols.

Parameter	Description
F_{RT}	Set of all RT flows
F_{NRT}	Set of all NRT flows
$F = F_{RT} \cup F_{NRT}$	Set of all flows
L_{F_i}	Set of all links in the path of flow F_i
F^{L_j}	Set of all flows using link L_j
λ_i	Injection rate of flow F_i
μ_w	Service rate of wired routers
μ_{CMAC}	Service rate of C-MAC module
X_i	Portion of flow F_i transmitted via wired plane
a_j	Workload of the link L_j
a_{CMAC}	Workload of the C-MAC module

deadline of RT flows is a part of the optimization constraints as well. In this way, the deadline of any RT flow will not be lost. The average-delay and worst-delay threshold are assumed to be given, which are determined according to the application and factors such as the number of RT flows, packet generation rate of RT flows, and the real-time requirements of the applications that affect their value.

The problem is formally defined as follows. Suppose we have an NoC where each router is connected to neighboring nodes through a number of wired links. Also, assume that there is a wireless antenna at each router through which a router can send packets to any desired destination directly. The wired plane uses the XY routing scheme which is a deterministic routing protocol that allows us to pre-calculate the path of any flow in the wired plane based on its source and destination. The data transmission through the wireless plane is done according to the request and grant mechanism described in the previous section.

The parameters and symbols used in the optimization problem are listed in Table 5. There are a number of data flows in this network, each with a specific source, destination, and packet injection rate. Let us show all RT and NRT flows with F_{RT} and F_{NRT}, respectively, such that the set of all flows is $F = F_{RT} \cup F_{NRT}$. For each flow F_i, we show the packet injection rate with λ_i. If the flow F_i is sent through the wired plane, we will show the set of all the links through which flow F_i passes with L_{F_i} (this information is available due to the XY routing mechanism). As discussed earlier, our goal is to distribute the flows between the wired and wireless plane to minimize the network latency. To this end, we define the decision variable X_i, which represents the portion of flow F_i sent through the wired network. For example, if $X_i = 0.7$, then 70% of packets from flow F_i are sent through the wired plane while 30% of packets sent through the wireless plane.

To formulate the optimization problem, we perform the following steps. In the first step, according to the decision variable X_i, which will be obtained from the optimization problem, we determine the traffic volume of each link as follows.

$$a_j = \sum_{\forall F_i \in F^{L_j}} X_i \lambda_i \tag{15}$$

Any amount of a flow that is not sent through the wired plane must be sent via the wireless plane. Thus, the $1 - X_i$ portion of flow F_i will be sent through the wireless plane. Since for sending a packet via the wireless plane, a *request* signal must be sent to C-MAC, the request rate or workload of the C-MAC module can be calculated as follows.

$$a_{CMAC} = \sum_{\forall F_i \in F} (1 - X_i)\lambda_i \tag{16}$$

According to the traffic volume of each link (Eq. (15)) and the workload of the CMAC module (Eq. (16)), the link and the C-MAC utilization factor can be written as follows.

$$\rho_{L_j} = a_j / \mu_w \tag{17}$$

$$\rho_{CMAC} = a_{CMAC} / \mu_{CMAC} \tag{18}$$

where μ_w and μ_{CMAC} are the service rate of a wireline router and C-MAC module, respectively.

Given that each flow might be divided between the wired and wireless paths, to calculate the latency of each flow, we calculate the latency of sending each flow wirelessly and wired individually.

To calculate the flow latency when a flow F_i is sent through the wired plane, we sum up the latency of all wired links along the flow path. We use the M/D/1 queue formula to calculate the latency of each link. In this way, the latency of flow F_i on the wired plane is calculated as follows.

$$d_{F_i}^{wire} = \sum_{\forall L_j \in L_{F_i}} \frac{1}{\mu_w} + \frac{\rho_{L_j}}{2\mu_w(1 - \rho_{L_j})} \tag{19}$$

Moreover, we use the following constraint to ensure that the buffer length of each individual router in the wired network is respected,

$$\Gamma \times \frac{\rho_L}{2\mu_w(1 - \rho_L)} \leq B_d \times \frac{1}{\mu_w}, \quad \forall L \tag{20}$$

where, the left-hand-side of equation computes the maximum queuing delay, which is ensured to be less than or equal to maximum time that a buffer with depth B_d can accommodate for. Here, Γ is a coefficient to relate the average delay $\frac{\rho_L}{2\mu_w(1-\rho_L)}$ to the maximum delay, which is determined from experiment.

The latency of sending a flow through the wireless plane consists of two parts. The first part involves waiting time for a *grant* signal from the C-MAC module which is proportional to the workload of C-MAC. We formulate this waiting time in both of the average and worst-case to calculate the average-case and the worst-case latency of wireless plane. Finally, we will apply a deadline to the delays, and try to meet the deadline through the optimization problem. To formulate the average waiting time for a grant, we utilize the M/D/1 queuing model, so the waiting time is written as follows.

$$d_{avg}^{CMAC} = \frac{1}{\mu_{CMAC}} + \frac{\rho_{CMAC}}{2\mu_{CMAC}(1 - \rho_{CMAC})} \tag{21}$$

To calculate the worst-case waiting time of a flow for a grant from C-MAC, initially, we calculate the number of flows which sent through the wireless plane. Since the C-MAC module works according to the Round-Robin fashion, the worst-case waiting time of a flow to reach the head of C-MAC queue is equal to the total waiting time to grant all prior flows. Regarding the division of traffic between wireless and wired networks, and based on the decision variable X_i, the number of flows sent from the wireless network is as follows.

$$\sum_{F_i \in F} \lceil (1 - X_i) \rceil \tag{22}$$

Due to the latency overhead for propagating *grant* signal from C-MAC module to applicant router (t_g), and the time to transmit a packet through the wireless plane (t_p), the worst-case waiting time will be as follows.

$$d_{worst}^{CMAC} = (t_g + t_p) \times \sum_{F_i \in F} \lceil (1 - X_i) \rceil \tag{23}$$

Like the average-case analysis of the waiting time for the *grant* signal, we apply a deadline to the worst-case waiting time for a grant. To this end, we have restricted the number of flows that can be sent through the wireless plane. We set the upper bound for the number of flows that can be transmitted wirelessly, which is calculated based on the real-time constraints imposed by the applications. In this way, by applying restriction on the number

of flows that transmit through the wireless plane, we meet the application's deadline.

The second part of the latency of sending a flow through the wireless plane consists of the latency of propagating the *request* signal to the C-MAC module (t_r), the latency of propagating *grant* signal from the C-MAC module (t_g), and the time to transmit a packet through the wireless link (t_p). So we can write the second part as follows.

$$d^{wl} = t_r + t_g + t_p \tag{24}$$

Through Eqs. (21) to (24), we calculate the average-case and the worst-case latency of transmitting a flow through the wireless plane as follows.

$$\textit{Average latency of wireless plane} = d^{CMAC}_{avg} + d^{wl} \tag{25}$$

$$\textit{Worst-case latency of wireless plane} = d^{CMAC}_{worst} + d^{wl} \tag{26}$$

We define the parameters *MTAL* and *MTWL* which represent the *maximum tolerable average latency* and *maximum tolerable worst latency*, respectively, and apply them as follows:

$$d^{CMAC}_{avg} + d^{wl} \leq MTAL \tag{27}$$

$$d^{CMAC}_{worst} + d^{wl} \leq MTWL \tag{28}$$

Using this parameter, it is possible to determine how close the wireless network is to the saturation point and how much the average delay in the wireless network is acceptable. Clearly, the higher tolerable latency on the wireless network, *i.e.,* the greater *MTAL*, the load in the wireless network increases, which results in more latency. On the other hand, if the *MTAL* is reduced, the number of packets sent through the wireless network will be reduced.

To give the RT flows a higher priority for transmitting through the wireless plane, we add another constraint to the problem. In this way there is only a possibility of transmitting an NRT flow wirelessly where all RT flows are sent through the wireless plane. To address this, we apply the following constraint.

$$1 - X_i > 0 \rightarrow \sum_{\forall F_i \in F_{RT}} (1 - X_i) = |F_{RT}| \qquad , \forall i : F_i \in F_{NRT} \tag{29}$$

So, we differentiate between RT and NRT flows, and give higher priority to RT flows for using the wireless plane. In this way, we use the wireless network capacity as best as possible. If wireless network capacity is less than RT traffic volume, then RT flows are sent wirelessly, which will ultimately have the least effect on the latency of wired plane. Otherwise, with respect to the deadlines of RT flows, a proper portion of the NRT flows will also be sent through the wireless plane, which again will lead to the maximum reduction in the latency of wired plane. It is noteworthy to mention that all the constraints here are second-order cones or linearizable equations and therefore existing commercial optimization solvers, like Gurobi [50], can efficiently solve the problem.

Finally, as the goal of our optimization problem, we minimize the maximum average latency of the wired plane, while we are confident in meeting deadlines on the wireless plane. This assurance is achieved through the constraints (27) to (29). The optimization problem is shown in Table 6.

Discussion. The optimization model presented in Table 6 assumes stationary traffic flows. However, in some applications (*e.g.* [51]) the traffic may exhibit a non-stationary behavior as the application changes its phase. In these situations, one can extend the proposed model with the concept of application phase and time, where every input value and decision variable is indexed by phase and time. Moreover, a series of modifications becomes necessary to connect the constraints of successive time steps. Also, it is interesting to investigate the extension of proposed model with the non-stationary Poisson processes and queuing models like [52,53]. Our proposed model, nevertheless, can serve as a template to employ these more sophisticated packet arrival models. In our model, we used M/D/1 queuing model to compute the delays and used constraint (20) for modeling the effect of buffer size. However, it is possible to consider the existence of buffers more accurately by using a M/D/1/K queue model (see [54]). However, this model has a more complex analytical equation for the delay, which complicates the solution of optimization problem.

Table 6
Optimization problem of network load distribution.

Optimal wireless load formulation summary		
min. $\max\{d^{wire}_{F_i} : \forall F_i \in F\}$		(15)
$a_j = \sum_{\forall F_i \in F^{L_j}} X_i \lambda_i$	$\forall j \in E$	(15a)
$a_{CMAC} = \sum_{\forall F_i \in F} (1 - X_i)\lambda_i$		(15b)
$\rho_{L_j} = a_j / \mu_w$	$\forall j \in E$	(15c)
$\rho_{CMAC} = a_{CMAC} / \mu_{CMAC}$		(15d)
$d^{wire}_{F_i} = \sum_{\forall L_j \in L_{F_i}} \frac{1}{\mu_w} + \frac{\rho_{L_j}}{2\mu_w(1-\rho_{L_j})}$	$\forall F_i \in F$	(15e)
$d^{CMAC}_{avg} = \frac{1}{\mu_{CMAC}} + \frac{\rho_{CMAC}}{2\mu_{CMAC}(1-\rho_{CMAC})}$		(15f)
$d^{CMAC}_{worst} = (t_g + t_p) \times \sum_{F_i \in F} \lceil (1 - X_i) \rceil$		(15g)
$d^{CMAC}_{avg} + d^{wl} \leq MTAL$		(15h)
$d^{CMAC}_{worst} + d^{wl} \leq MTWL$		(15i)
$1 - X_i > 0 \rightarrow \sum_{\forall F_i \in F_{RT}} (1 - X_i) = \lvert F_{RT} \rvert$	$\forall i : F_i \in F_{NRT}$	(15j)

6. Results and analysis

In this section, we evaluate the average-case and the worst-case performance parameters simultaneously on different networks as a result of the proposed method. For this purpose, multiple real or synthetic workloads are applied in three different scenarios. The scenarios are as follows:

- In scenario 1 (SC1), we utilize a simple wired mesh NoC as the baseline architecture. So, the whole traffic, including RT and NRT are routed through the wired plane.
- Scenario 2 (SC2) is a hybrid viable [24,55] wired/wireless NoC structure wherein each switch is equipped with an antenna, as described in Section 3. In this scenario, the RT flows are routed through the wireless network while NRT flows use the wired network.
- In Scenario 3 (SC3), the network structure is the same as SC2, but the RT and NRT traffic flows are distributed between the wired and wireless planes based on the decision of an optimization problem described in Section 5.

6.1. Average performance evaluation

In order to show the scalability of the proposed approach in terms of the network size, we have considered mesh sizes 4×4, 6×6, and 8×8. For this purpose, extensive simulation has been carried out to evaluate the average performance metrics using the Booksim2.0 [56] simulator. The simulation exploits full mesh traffic distribution, in which the destinations of the flows are distributed uniformly among all the nodes. For SC1, in which

all the traffic is routed in the wired plane, a standard implementation of a best-effort credit-based wormhole NoC is used. For SC2, employing the wired network for NRT and the wireless for RT traffic, the C-MAC is integrated inside the simulator. This integration facilitates extracting accurate average performance parameters for the packets of RT flows sent through the wireless plane.

As depicted in Fig. 4, the baseline wired network (SC1) is compared to two different examples implemented on the wired/wireless network (SC2) in which 10% and 30% of the traffic flows are considered to be RT. We should bear in mind that 10% is reasonable for real-time applications [27], and 30% is considered as an extreme condition for evaluation purposes. The figure shows the saturation point of the wireless plane outperforms the wired plane by far when comparing c2 with c1 (in SC2 and SC1, respectively. This is true also for c4 compared to c1 (in SC2 and SC1), when comparing the whole packets in the network. We have considered a viable wireless network configuration in terms of feasibility of physical implementation [3,55] and also the overhead (as reported in Section 6.4) incurred by adding the wireless plane to a baseline wired network. For the case of SC2 when 30% of the flows are RT, due to limited capacity of the wireless plane, the average delay of the RT flows (c5) is worse than the average-case of SC1 (c1). We have extracted experimentally for the 4×4 network the turning point in which the RT flows will exhibit better average performance than sending the RT flows on the wired plane in SC1 is when 18% of the flows are RT. In other words, the hybrid wireless/wired network will not worsen the average delay of the RT flows in SC2 compared to SC1, if at most 18% of the (RT) flows use the wireless plane for our viable network setup. It is evident, in any case the average delay for NRT flows exhibits better results in SC2 compared to SC1. We have examined this situation for 6×6 and also 8×8 networks and have observed similar behavior (Figs. 5 and 6). The turning point for these networks is 17% and 15% respectively. This evaluation confirms the fact that by using a wireless plane for RT traffics the average-case performance is not deteriorated and even improved. At the same time, we have provided extremely better hard and tight worst-case performance metrics for RT flows by employing the wireless plane as is discussed in the following subsection. Figs. 4, 5, and 6 also demonstrate that the zero-load latency for RT flows in SC2 outperforms SC1 due to the fewer hop counts in the wireless plane compared to the wired counterpart in which low zero-load latency is crucial for such networks.

6.2. Worst-case performance evaluation

Figs. 7a and 7b illustrate the performance speedup in terms of worst-case latency and guaranteed bandwidth for SC2 compared to SC1 for different variants of full mesh networks (4×4 with 240, 6×6 with 1260 and 8×8 with 4032 flows). Fig. 7a shows the worst-case delay speedup (reduction) parameter. In this figure, the horizontal axes show the number of RT flows in different cases ranging from zero to the number of flows. The full range is covered for evaluation purposes. Although, as discussed earlier in real-world RT applications at most 10% of the flows are RT, but exploring the results in wider range helps to clarify the subject and also the robustness of the proposed methods. For this purpose, we have considered all the combinations in which a specific number of traffic flows are RT and evaluated the flows' average worst-case latency speedup. As seen for the case of 4×4 network, in case 10% (24 out of 240) of the flows are RT, the delay is decreased by a factor of 58x for RT flows, in which such extremely low delay is essential in real-time applications. In this case, the NRT flows exhibit a worst-case latency decrease by a factor of 3x, although not an important metric for them as it is for their RT counterparts. These numbers for 6×6 and 8×8 networks are 111x and 2860x. It should be noted although very dense scenarios may never happen in real-world, but we have evaluated them to show the scalability of the proposed approach. Fig. 7b shows the guaranteed bandwidth improvement, in which for the 10% case, the average bandwidth is increased by a factor of 75x, 127x, and 3156x for RT flows in 4×4, 6×6, and 8×8 counterparts, respectively.

We have also applied the proposed idea to a real-world multimedia application named D26-Media from [27] with 25 IP-cores and 67 traffic flows in which 7 of them are RT (specified as filled circles in Fig. 8). The architectures are selected from both 4×4 and 5×5 hybrid mesh networks. The horizontal axes show the index of traffic flows. The RT flows' average worst-case latency and bandwidth metrics improve by 80% and 301% for implementing D26-Media application on a 4×4 mesh respectively. The results for implementing on a 5×5 mesh are 98% and 3666%. Moreover, the improvement of these parameters for the NRT flows are 12% and 20% on the 4×4 mesh and 10% and 28% on the 5×5 mesh networks.

6.3. Average vs. worst-case performance evaluation

In this section, we evaluate the proposed optimization problem which is described as SC3. For this purpose, the results of the optimization problem for two real-world applications are compared with SC1 and SC2. These applications, which are referred to as *36core-4* [57] and *D26-Media* [58], are mapped on a 6×6 and a 5×5 meshes, respectively. In *36core-4*, 24 flows out of 144 and in *D26-Media*, 7 flows out of 67 are the RT flows. In this section, we evaluate the average performance of the system and hence make sure that all the packet delivery deadlines are guaranteed by setting *MTWL* to the possible maximum latency which is 144 and 100 cycles for the meshes of size 36 and 25, respectively. This allows the optimization program to send traffic from any core to the wireless plane.

Evaluation criteria. We consider packet and flow latency as evaluation criteria to compare and evaluate three proposed scenarios. The latency of a flow is the average delay of all packets belonging to that flow. After calculating the flow latency, we report the average latency of all flows, denoted by *FAVG*, as well as the maximum observed latency of any flow and any packet, denoted by *FMAX* and *PMAX*, respectively. Remember that the objective of the optimization program is to minimize the *FAVG* subject to the upper-bound constraint on the values of *FMAX* and *PMAX*, specified by the *MTAL* and *MTWL* parameters defined in Section 5.

***36core-4* latency.** The total size of the RT flows in this application is rather higher than the capacity of the wireless channel, and therefore, we expect to observe that the naive approach of SC2, *i.e.,* sending all RT flows via the wireless channel, cause high delays due to saturation. Since wireless network capacity is limited and rapidly approaches the state of saturation, such undesirable conditions are observed if the amount of traffic sent to the wireless network is greater than its capacity. Indeed, we can see, in Table 7, that the maximum packet latency is 131 cycles which is close to the maximum possible latency of 144 cycles. As expected, the wired traffic density under SC2 is lower compared to SC1, and average flow latency is reduced from 20 cycles to 17.9 cycles. However, the maximum flow latency is not changed and is equal to 33 cycles for both of these scenarios. The proposed optimization model, *i.e.,* SC3, accounts for the volume of RT flows and avoids sending all of them via the wireless channel to prevent overloading the C-MAC. Since the SC3 uses the *MTAL* parameter to control how close the C-MAC can be to the saturation state, we, for a closer examination of the effect of *MTAL* on the performance

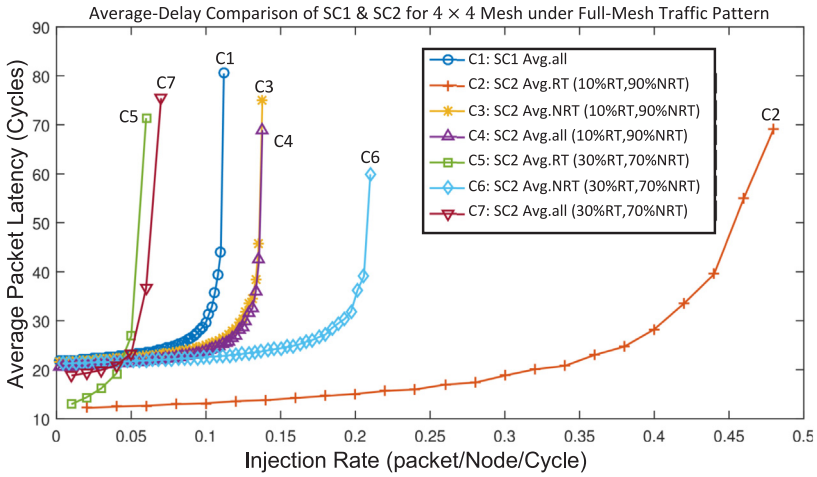


Fig. 4. Average-case analysis for a 4 × 4 mesh.

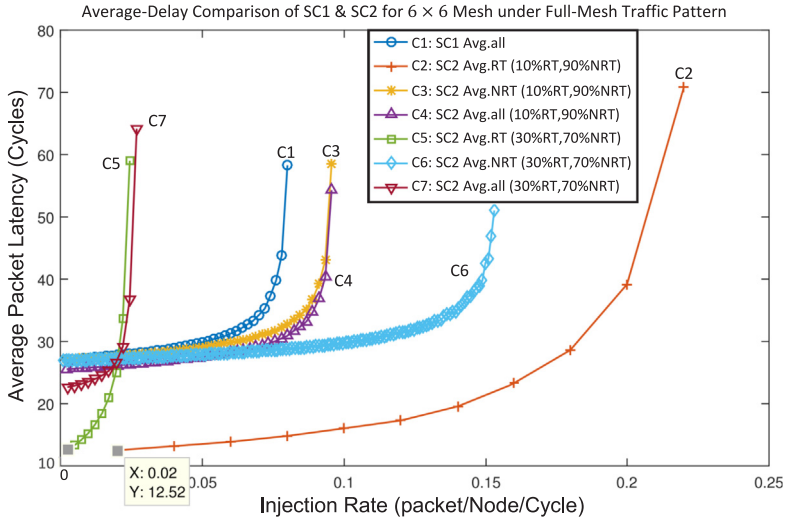


Fig. 5. Average-case analysis for a 6 × 6 mesh.

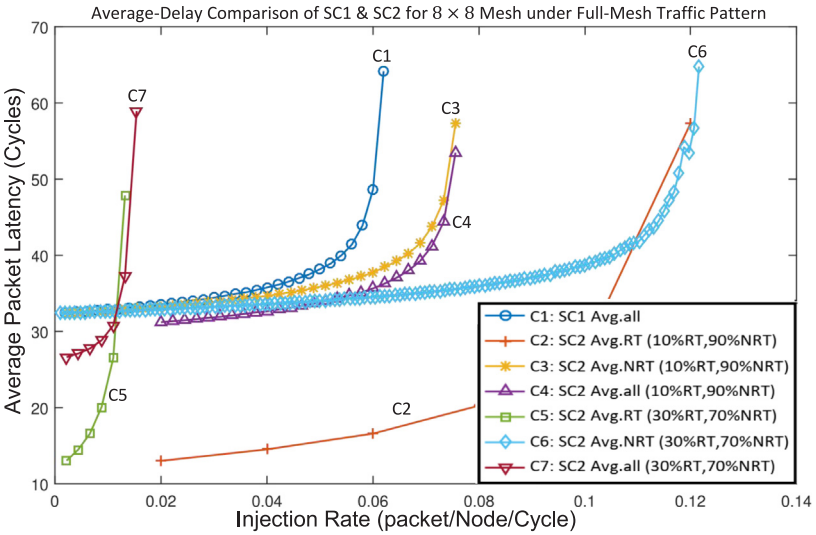
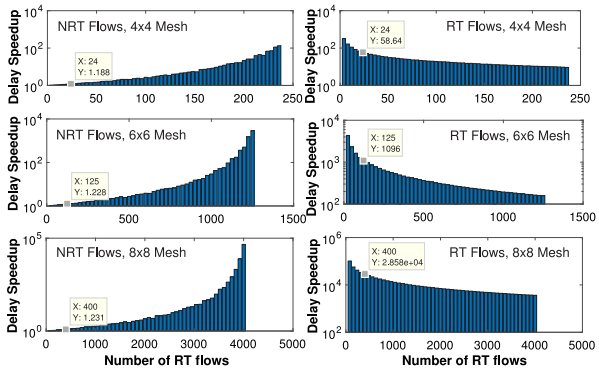
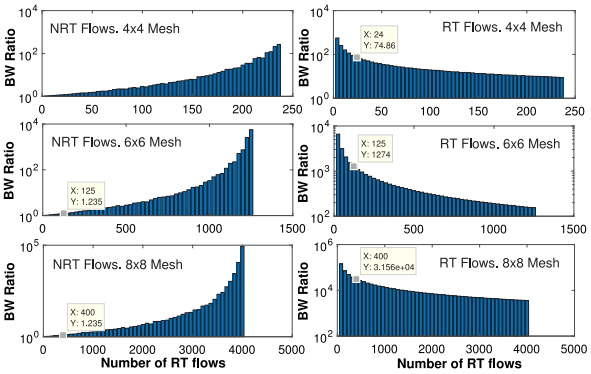


Fig. 6. Average-case analysis for an 8 × 8 mesh.



(a) Delay speedup in worst-case analysis for RT and NRT flows



(b) Bandwidth ratio in worst-case analysis for RT and NRT flows

Fig. 7. Worst-case delay and bandwidth analysis.

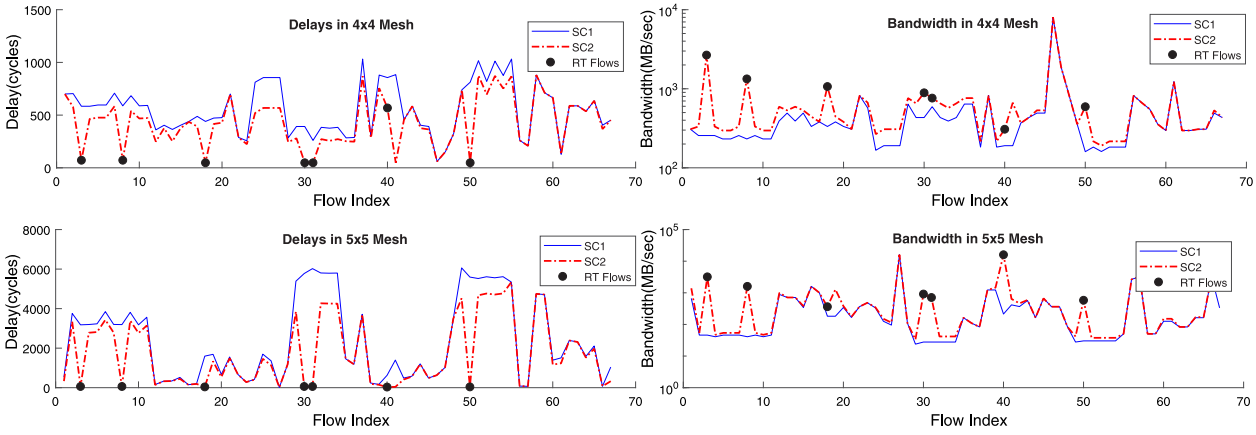


Fig. 8. Worst-case analysis of D26-Media application applied on different mesh sizes.

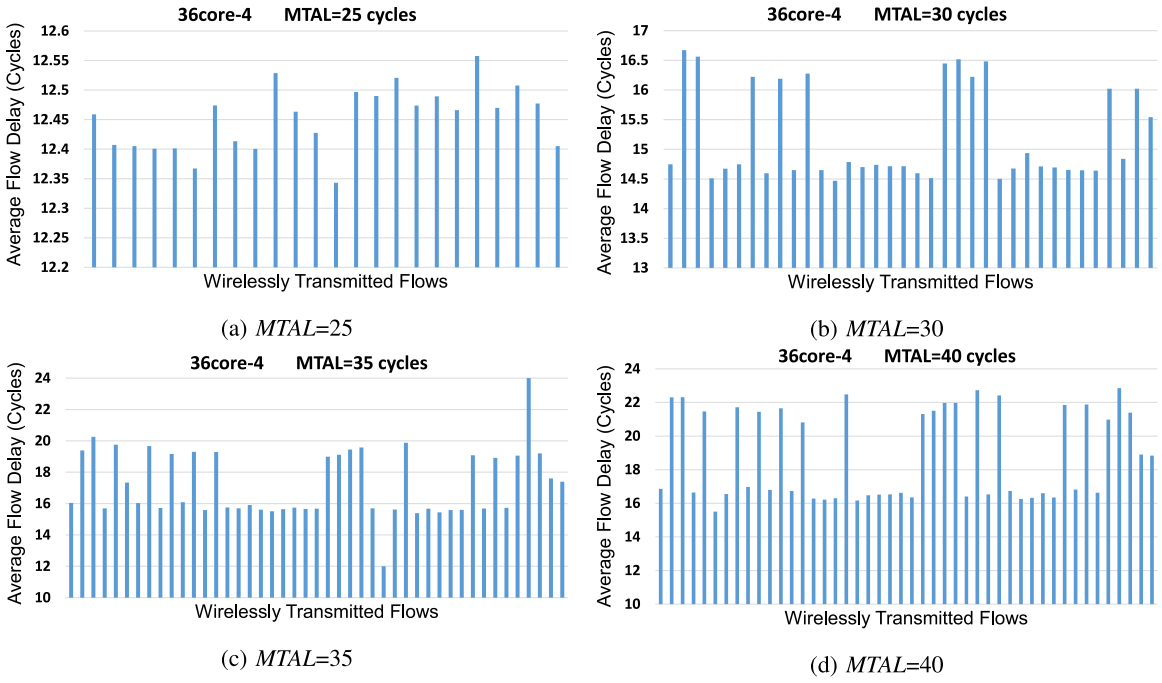
Table 7
36core-4.

	Wired					Wireless					All
	FAVG	**FMAX**	**PMAX**	**PKT#**	**F#**	**FAVG**	**FMAX**	**PMAX**	**PKT#**	**F#**	**FAVG**
SC1	20.1	33.2	45	–	–	–	–	–	–	–	20.1
SC2	18.7	33.2	43	1 200 000	120	13.6	13.8	131	240 000	24	17.9
SC3(25)	19.9	33.2	42	1 364 759	142	12.4	12.5	47	75 241	24	18.8
SC3(30)	18.9	33.2	42	1 252 673	117	12.7	12.9	92	187 327	24	18.1
SC3(35)	18.0	33.2	40	1 066 471	113	17.3	21	126	373 529	45	17.8
SC3(40)	18.0	33.1	39	1 038 896	113	18.7	29	140	401 104	46	18.2

of the optimization problem, examine the values of $MTAL = 25$, $MTAL = 30$, $MTAL = 35$, and $MTAL = 40$.

We see that SC3(25) avoids undesirable packet delays like what is observed under SC2, *i.e.,* 131 cycles, by reducing the number of packets that use the wireless channel by about 69% which results in the maximum packet latency of 47 cycles and 64% reduction. This suggests that the proposed optimization method effectively prevents the wireless network from being saturated, and has never injected it over the capacity of the wireless network. Moreover, the average delay in the wireless network has fallen between 13.6 cycles in SC2 and 12.4 in SC3, and the maximum flow latency has fallen between 13.8 cycles in SC2 and 12.5 in SC3. Sending more packets through the wired plane increases the average flow latency in the wired plane, from 18.7 cycles to 19.9 cycles. The maximum flow latency in the wired network remains unchanged at 33.2 cycles. Furthermore, Fig. 9a shows the average latency for all flows sent through the wireless plane. We see that the latency of all flows is less than the *MTAL* parameter.

By setting the *MTAL* to 30 cycles, SC3 is allowed to increase the traffic load of the wireless channel, compared to $MTAL = 25$. We can see that, in Table 7, the number of packets in the wireless network is roughly increased by a factor of 2.4, still less than what is observed in SC2. Because the number of packets that use the wireless channel increases and it has lower latency compared to the wired network (on average), the overall average flow latency is reduced, compared to SC3(25), and reaching 18.1 cycles. However, the average flow latency of the wireless network increases from 12.5 cycles to 12.7. In this case, the maximum flow latency of the wireless plane increases from 12.5 cycles to 12.9. For the wired plane, since it routes a smaller number of packets, the average flow latency is reduced around 6% compared to SC3(25); however, it is still slightly above what is observed in SC2. Another noteworthy point is that the maximum packet latency of the wireless network increase from 47 to 92 cycles. This means that the optimizer has allowed taking the wireless



(a) *MTAL*=25 (b) *MTAL*=30

(c) *MTAL*=35 (d) *MTAL*=40

Fig. 9. Latency of *36core-4* flows.

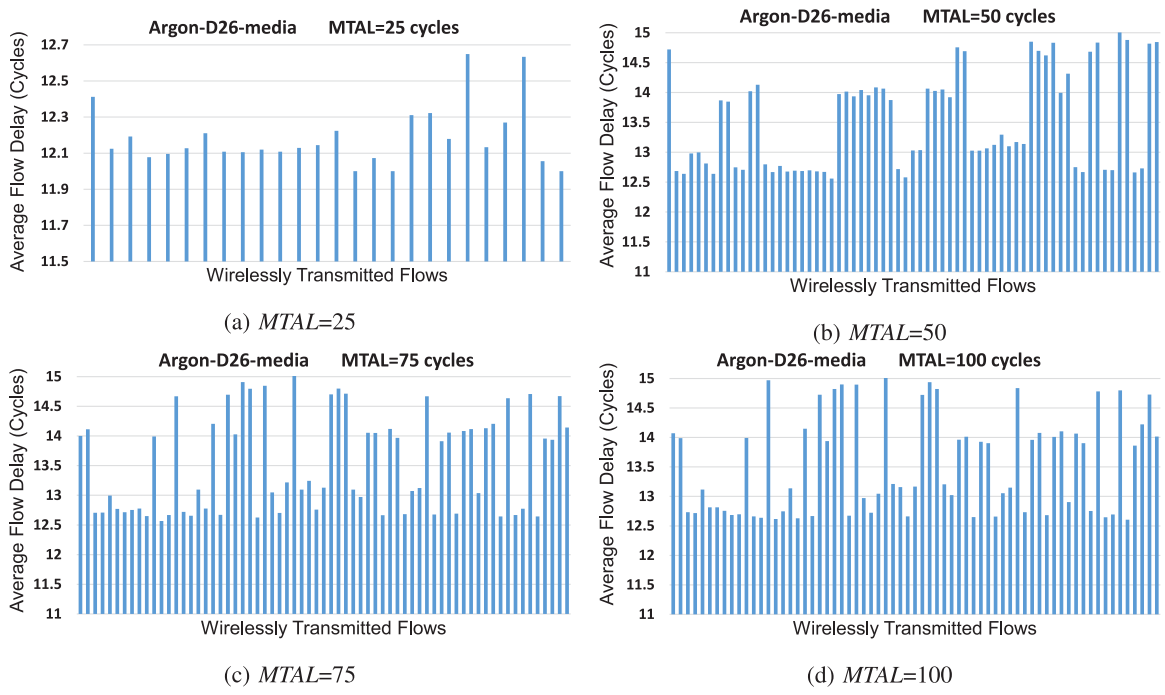
Table 8
D26-Media.

	Wired					Wireless					All
	FAVG	**FMAX**	**PMAX**	**PKT#**	**F#**	**FAVG**	**FMAX**	**PMAX**	**PKT#**	**F#**	**FAVG**
SC1	17.9	37.0	43	–	–	–	–	–	–	–	17.9
SC2	17.8	37.0	43	600 000	–	12.1	12.2	35	70 000	–	17.2
SC3(25)	16.8	29.0	33	554 993	–	12.4	24	45	115 007	–	15.1
SC3(50)	16.6	29.0	33	215 724	–	13.5	25.2	78	454 276	–	14.7
SC3(75)	16.7	29.0	33	213 354	–	13.5	33.0	83	456 646	–	14.7
SC3(100)	16.9	29.0	32	212 599	–	13.5	42.1	87	457 401	–	14.8

network to saturation point so that such a large delay has occurred and since the wireless network is very sensitive to the traffic load, it rapidly approaches the saturation point. However, this delay has a considerable distance from 131 cycles which is observable under SC2. The next point is that for SC3(35) the value of *MTAL* is large enough to allow the optimizer to load the wireless channel to the point that the number of packets transmitted via the wireless channel exceeds what is achieved by SC2. Consequently, the load on the wired network reduces significantly, by about 1.9 cycles compared to SC3(25), and the overall average flow latency reaches 17.8 which is lower than SC2, SC3(25), and SC3(30). This trend is observed in SC3(40) where the increased number of packets transmitted through the wireless channel causes a considerable delay at the wireless channel and the overall average flow latency reaches 18.2 cycles. Furthermore, Figs. 9b, 9c, and 9d show that the average latency of all flows is less than the respective value of the parameter *MTAL* and the optimizer is able to guarantee this maximum latency for them.

***D26-Media* latency.** The number of RT flows is low in this application compared to the capacity of the C-MAC (, in contrast to the *36core-4*). Therefore, to fully utilize the wireless channel and achieve an optimal average delay a portion of NRT flows should be sent through the wireless plane. Consequently, the naive approach of the SC2 is not sufficient to achieve the best performance. Again, for a deeper analysis, we present the results of running SC3 with four different values of *MTAL*, *i.e.,* 25, 50, 75, and 100. Specifically, Table 8 shows that SC3(25) sends about 64% more packets through the wireless channel which leads to a significantly lower delay in the wired network while not exacerbating the state of the wireless channel. Using SC3(25), the average delay of the wired network reaches 16.8 cycles (from 17.8 cycles of SC2). Furthermore, the maximum flow latency and the maximum packet latency of 37 and 43 cycles in SC2, is reduced to 29 and 33 cycles in SC3, respectively. Moreover, the C-MAC still remains in an unsaturated state and only shows a less than 0.3 cycles increase of average flow latency. Considering all the packets transmitted through a wired and wireless planes we see a 13% decrease in the average flow latency, achieving 15.1 cycles from 17.2 cycles under SC2. Specifically, the maximum flow latency increases from 12.2 cycles to 24 cycles and maximum packet latency increases from 35 cycles to 45 cycles. Remember that, in a 5×5 mesh network, the maximum packet latency is 100 cycles, therefore, the network still is far from being saturated. The average delay for all the flows that are transmitted via the wireless plane is depicted in Fig. 10a which shows that all of them are significantly lower than the tolerable delay parameter, *MTAL*.

As we increase the value of *MTAL*, at first, the overall delay decreases. The reason is that the wireless channel still has capacity and sending more packets through it reduces the delay. However, beyond a certain threshold (50 cycles here), the delay does not change since the optimizer avoids saturating the C-MAC and does not send more packets to it. Specifically, we can see that when we set $MTAL = 50$, the overall delay decreases, from 15.1 cycles, to 14.7 cycles. However, setting $MTAL = 75$ or $MTAL = 100$ does not change the overall delay significantly. Table 8 and Figs. 10c and 10d show that the number of packets and flows that are transmitted through the wireless plane by setting $MTAL = 75$ and $MTAL = 100$ are relatively equal to what we observed under



(a) *MTAL*=25 (b) *MTAL*=50

(c) *MTAL*=75 (d) *MTAL*=100

Fig. 10. Latency of Argon 5 × 5 flows.

Table 9
Area overhead of C-MAC module compared to router's area.

Module type ↓	Area (μm^2) ↓			
Baseline router (BR)	108 805			
Baseline router with RF transceiver (HB)	468 805			
RF transceiver	360 000 [3]			
NoC Mesh size →	4 × 4	6 × 6	8 × 8	10 × 10
C-MAC area (μm^2)	453	1115	1997	3146
C-MAC area per router (μm^2)	28.3	31.0	31.2	31.4
C-MAC area overhead compared to BR	0.42%	1.03%	1.84%	2.9%
C-MAC area overhead per hybrid router	0.006%	0.007%	0.007%	0.007%

MTAL = 50. Table 8 shows that the average flow latency in wired plane is close to 16.7 for all three values of *MTAL* parameter, and the average flow latency in wireless plane is equal to 13.5 for all of them. Again, Figs. 10b, 10c, and 10d show the average latency for all flows sent through wireless plane which demonstrates that the delay of all flows under the specified *MTAL* parameters and the optimization model is able to guarantee the maximum average latency.

6.4. Area overhead

Table 9 shows the area overhead of the proposed C-MAC arbiter compared to the area of hybrid router (HB) in SC2 for different network sizes. Furthermore, to show the area overhead of the C-MAC module, we report the C-MAC area per router and the per-router area overhead of C-MAC module in a hybrid NoC for different mesh sizes. As it is shown in the table the C-MAC area per router is approximately 30 μm^2 which is around 0.007% of a HB area. These results extracted using our VHDL implementation of the router, applied to 45 nm VLSI Technology and Synopsys Prime-Power synthesis tool.

To further clarify the scalability of the proposed architecture, it is critical to note that it is feasible to create multiple distinct on-chip frequency channels through current CMOS technology [2,35]. By increasing the number of on-chip processing cores, the chip can be divided into different zones with a dedicated frequency to realize wireless communication within a zone and a CMAC module to implement the media access control process. In this way, all the wireless transmissions within a specific zone can be carried out without any interference from other zones. Also, the area overhead of the CMAC module is kept proportional to the limited number of cores in each region. In this case, to further facilitate the communication process and minimizing the inter-communication between different zones, it will be helpful that during the task mapping process, all the tasks belong to an application are mapped within a single zone.

7. Conclusions

This paper proposed the idea of a hybrid wireless/wired router mapped on variable-sized mesh NoCs. It also proposed the structure of an arbitration unit for the wireless section, in which the worst-case performance metrics are improved significantly compared to a baseline best-effort wired NoC employing different scenarios. The first scenario (SC1) is a base-line wired mesh network, which is used to evaluate the other two scenarios. The second scenario (SC2) suggests to send the real-time traffic through the wireless plane, while the remaining portion to be routed through the wired plane. Employing this scenario results in significantly better worst-case performance parameters for the real-time portion and slightly better worst-case parameters for the non real-time traffics. This is true when both the real-time and non real-time traffics exhibit better average-case performance parameters for known viable structures and applications. As a result the main goal in this scenario has been significantly improved the worst-case performance parameters for the real-time traffic. The third scenario (SC3) aims at improving the average-case performance parameters while the required worst-case performance is not violated using a hybrid traffic

distribution approach through the planes along with an optimization problem solving. The result is significant worst-case and average-case parameters improvement over the base-line approach. The analytical model to calculate the worst-case parameters for the network is provided and the proposed method has been evaluated employing different real or synthetic applications.

CRediT authorship contribution statement

Mohammad Baharloo: Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing. **Ahmad Khonsari:** Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing. **Mahdi Dolati:** Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing. **Pouya Shiri:** Acquisition of data, Analysis and/or interpretation of data, Writing - original draft. **Masoumeh Ebrahimi:** Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing. **Dara Rahmati:** Conception and design of study, Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

All authors approved the version of the manuscript to be published.

References

[1] J. Lin, H. Wu, Y. Su, L. Gao, A. Sugavanam, J.E. Brewer, K.K. O, Communication using antennas fabricated in silicon integrated circuits, IEEE J. Solid-State Circuits 42 (8) (2007) 1678–1687.

[2] S. Deb, K. Chang, X. Yu, S.P. Sah, M. Cosic, A. Ganguly, P.P. Pande, B. Belzer, D. Heo, Design of an energy-efficient CMOS-compatible NoC architecture with millimeter-wave wireless interconnects, IEEE Trans. Comput. 62 (12) (2013) 2382–2396.

[3] S. Abadal, A. Mestres, M. Nemirovsky, H. Lee, A. González, E. Alarcón, A. Cabellos-Aparicio, Scalability of broadcast performance in wireless network-on-chip, IEEE Trans. Parallel Distrib. Syst. 27 (12) (2016) 3631–3645.

[4] S.H. Gade, M. Sinha, S.S. Rout, S. Deb, Enabling reliable high throughput on-chip wireless communication for many core architectures, in: 2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2018, pp. 591–596.

[5] S. Abadal, J. Torrellas, E. Alarcón, A. Cabellos-Aparicio, OrthoNoC: A broadcast-oriented dual-plane wireless network-on-chip architecture, IEEE Trans. Parallel Distrib. Syst. 29 (3) (2018) 628–641.

[6] P. Mitra, B. Sharma, V.K. Chandna, V.S. Rathore, Design and performance evaluation of hybrid wired-wireless network on chip interconnect architectures, in: X.-S. Yang, S. Sherratt, N. Dey, A. Joshi (Eds.), Third International Congress on Information and Communication Technology, Springer Singapore, Singapore, 2019, pp. 191–199.

[7] S. Abadal, A. Mestres, J. Torrellas, E. Alarcon, A. Cabellos-Aparicio, Medium access control in wireless network-on-chip: A context analysis, IEEE Commun. Mag. 56 (6) (2018) 172–178.

[8] B. Bahrami, M.A.J. Jamali, S. Saeidi, A novel hierarchical architecture for wireless network-on-chip, J. Parallel Distrib. Comput. 120 (2018) 307–321, [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0743731518301102.

[9] A. Dehghani, K. RahimiZadeh, Design and performance evaluation of mesh-of-tree-based hierarchical wireless network-on-chip for multicore systems, J. Parallel Distrib. Comput. 123 (2019) 100–117, [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0743731518306592.

[10] K. Duraisamy, Y. Xue, P. Bogdan, P.P. Pande, Multicast-aware high-performance wireless network-on-chip architectures, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 25 (3) (2017) 1126–1139.

[11] M. Opoku Agyeman, W. Zong, A. Yakovlev, K.-F. Tong, T. Mak, Extending the performance of hybrid nocs beyond the limitations of network heterogeneity, J. Low Power Electron. Appl. 7 (2) (2017) 8.

[12] Y. Peng, L. Guo, Q. Gai, Cross-layer QoS-aware routing protocol for multi-radio multi-channel wireless mesh networks, in: Communication Technology (ICCT), 2012 IEEE 14th International Conference on, IEEE, 2012, pp. 197–201.

[13] D. Zhao, Y. Wang, SD-MAC: Design and synthesis of a hardware-efficient collision-free QoS-aware MAC protocol for wireless network-on-chip, IEEE Trans. Comput. 57 (9) (2008) 1230–1245.

[14] A. Rezaei, M. Daneshtalab, D. Zhao, CAP-W: Congestion-aware platform for wireless-based network-on-chip in many-core era, Microprocess. Microsyst. 52 (2017) 23–33, [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0141933116302381.

[15] A. Shacham, K. Bergman, L.P. Carloni, Photonic networks-on-chip for future generations of chip multiprocessors, IEEE Trans. Comput. 57 (9) (2008) 1246–1260.

[16] M.F. Chang, J. Cong, A. Kaplan, M. Naik, G. Reinman, E. Socher, S.W. Tam, CMP network-on-chip overlaid with multi-band RF-interconnect, in: 2008 IEEE 14th International Symposium on High Performance Computer Architecture, 2008, pp. 191–202.

[17] M.O. Agyeman, A. Ahmadinia, N. Bagherzadeh, Performance and energy aware inhomogeneous 3D networks-on-chip architecture generation, IEEE Trans. Parallel Distrib. Syst. 27 (6) (2016) 1756–1769.

[18] M. Chrzanowska-Jeske, J. Becker, Tutorial 2A: 3D integration - challenges and advantages, in: 2016 29th IEEE International System-on-Chip Conference (SOCC), 2016, pp. 1–3.

[19] C. Wang, W.H. Hu, N. Bagherzadeh, A wireless network-on-chip design for multicore platforms, in: 2011 19th International Euromicro Conference on Parallel, Distributed and Network-Based Processing, 2011, pp. 409–416.

[20] M.S. Shamim, N. Mansoor, R.S. Narde, V. Kothandapani, A. Ganguly, J. Venkataraman, A wireless interconnection framework for seamless inter and intra-chip communication in multichip systems, IEEE Trans. Comput. 66 (3) (2017) 389–402.

[21] A. Samaiyar, S.S. Ram, S. Deb, Millimeter-wave planar log periodic antenna for on-chip wireless interconnects, in: The 8th European Conference on Antennas and Propagation (EuCAP 2014), 2014, pp. 1007–1009.

[22] H.K. Mondal, S.H. Gade, M.S. Shamim, S. Deb, A. Ganguly, Interference-aware wireless network-on-chip architecture using directional antennas, IEEE Trans. Multi-Scale Comput. Syst. 3 (3) (2017) 193–205.

[23] Y. Xue, P. Bogdan, User cooperation network coding approach for noc performance improvement, in: Proceedings of the 9th International Symposium on Networks-on-Chip, in: NOCS '15, Association for Computing Machinery, New York, NY, USA, 2015, [Online]. Available: https://doi.org/10.1145/2786572.2786575.

[24] S. Abadal, B. Sheinman, O. Katz, O. Markish, D. Elad, Y. Fournier, D. Roca, M. Hanzich, G. Houzeaux, M. Nemirovsky, E. Alarcón, A. Cabellos-Aparicio, Broadcast-enabled massive multicore architectures: A wireless RF approach, IEEE Micro 35 (5) (2015) 52–61.

[25] K. Goossens, J. Dielissen, A. Radulescu, Aethereal network on chip: concepts, architectures, and implementations, IEEE Des. Test Comput. 22 (5) (2005) 414–421.

[26] H. Kopetz, A. Damm, C. Koza, M. Mulazzani, W. Schwabl, C. Senft, R. Zainlinger, Distributed fault-tolerant real-time systems: the Mars approach, IEEE Micro 9 (1) (1989) 25–40.

[27] D. Rahmati, S. Murali, L. Benini, F. Angiolini, G.D. Micheli, H. Sarbazi-Azad, Computing accurate performance bounds for best effort networks-on-chip, IEEE Trans. Comput. 62 (3) (2013) 452–467.

[28] R. Mullins, A. West, S. Moore, The design and implementation of a low-latency on-chip network, in: Proceedings of the 2006 Asia and South Pacific Design Automation Conference, in: ASP-DAC '06, IEEE Press, Piscataway, NJ, USA, 2006, pp. 164–169, [Online]. Available: https://doi.org/10.1145/1118299.1118348.

[29] C. Paukovits, H. Kopetz, Concepts of switching in the time-triggered network-on-chip, in: 2008 14th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, 2008, pp. 120–129.

[30] Z. Shi, A. Burns, Real-time communication analysis for on-chip networks with wormhole switching, in: Second ACM/IEEE International Symposium on Networks-on-Chip (Nocs 2008), 2008, pp. 161–170.

[31] D. DiTomaso, S. Laha, A. Kodi, S. Kaya, D. Matolak, Evaluation and performance analysis of energy efficient wireless NoC architecture, in: 2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS), 2012, pp. 798–801.

[32] S. Abadal, M. Iannazzo, M. Nemirovsky, A. Cabellos-Aparicio, H. Lee, E. Alarcón, On the area and energy scalability of wireless network-on-chip: A model-based benchmarked design space exploration, IEEE/ACM Trans. Netw. 23 (5) (2015) 1501–1513.
[33] O. Markish, O. Katz, B. Sheinman, D. Corcos, D. Elad, On-chip millimeter wave antennas and transceivers, in: Proceedings of the 9th International Symposium on Networks-on-Chip, in: NOCS '15, ACM, New York, NY, USA, 2015, pp. 11:1–11:7, [Online]. Available: http://doi.acm.org/10.1145/2786572.2789983.
[34] K. Duraisamy, R.G. Kim, P.P. Pande, Enhancing performance of wireless NoCs with distributed MAC protocols, in: Sixteenth International Symposium on Quality Electronic Design, 2015, pp. 406–411.
[35] A. Ganguly, K. Chang, S. Deb, P.P. Pande, B. Belzer, C. Teuscher, Scalable hybrid wireless network-on-chip architectures for multicore systems, IEEE Trans. Comput. 60 (10) (2011) 1485–1502.
[36] M. Baharloo, A. Khonsari, A low-power wireless-assisted multiple network-on-chip, Microprocess. Microsyst. 63 (2018) 104–115.
[37] M. Ruggiero, A. Guerri, D. Bertozzi, F. Poletti, M. Milano, Communication-aware allocation and scheduling framework for stream-oriented multi-processor systems-on-chip, in: Design Automation Test in Europe Conference, Vol. 1, 2006, p. 6.
[38] K. Manna, P. Mukherjee, S. Chattopadhyay, I. Sengupta, Thermal-aware application mapping strategy for network-on-chip based system design, IEEE Trans. Comput. 67 (4) (2018) 528–542.
[39] A.K. Coskun, T.v. Rosing, K.A. Whisnant, K.C. Gross, Static and dynamic temperature-aware scheduling for multiprocessor SoCs, IEEE Trans. Very Large Scale Integr. Syst. 16 (9) (2008) 1127–1140.
[40] T. Chantem, X.S. Hu, R.P. Dick, Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs, IEEE Trans. Very Large Scale Integr. Syst. 19 (10) (2011) 1884–1897.
[41] M. Sacanamboy, L. Quesada, F. Bolanos, A. Bernal, B. O'Sullivan, A comparison between two optimisation alternatives for mapping in wireless network on chip, in: IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2016, pp. 938–945.
[42] R.G. Kim, W. Choi, G. Liu, E. Mohandesi, P.P. Pande, D. Marculescu, R. Marculescu, Wireless noc for VFI-enabled multicore chip design: Performance evaluation and design trade-offs, IEEE Trans. Comput. 65 (4) (2016) 1323–1336.
[43] P. Bogdan, R. Marculescu, Workload characterization and its impact on multicore platform design, in: 2010 IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2010, pp. 231–240.
[44] P. Bogdan, Mathematical modeling and control of multifractal workloads for data-center-on-a-chip optimization, in: Proceedings of the 9th International Symposium on Networks-on-Chip, in: NOCS '15, Association for Computing Machinery, New York, NY, USA, 2015, [Online]. Available: https://doi.org/10.1145/2786572.2786592.
[45] S. Abadal, A. Cabellos-Aparicio, E. Alarcon, J. Torrellas, WiSync: An architecture for fast synchronization through on-chip wireless communication, in: Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems, in: ASPLOS '16, ACM, New York, NY, USA, 2016, pp. 3–17, [Online]. Available: http://doi.acm.org/10.1145/2872362.2872396.
[46] V. Fernando, A. Franques, S. Abadal, S. Misailovic, J. Torrellas, Replica: A wireless manycore for communication-intensive and approximate data, in: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, in: ASPLOS '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 849–863, [Online]. Available: https://doi.org/10.1145/3297858.3304033.
[47] P. Guerrier, A. Greiner, A generic architecture for on-chip packet-switched interconnections, in: Proceedings Design, Automation and Test in Europe Conference and Exhibition 2000 (Cat. No. PR00537), 2000, pp. 250–256.
[48] D. Sanchez, G. Michelogiannakis, C. Kozyrakis, An analysis of on-chip interconnection networks for large-scale chip multiprocessors, ACM Trans. Archit. Code Optim. 7 (1) (2010) 4:1–4:28, [Online]. Available: http://doi.acm.org/10.1145/1736065.1736069.
[49] C.-H.O. Chen, S. Park, T. Krishna, S. Subramanian, A.P. Chandrakasan, L.-S. Peh, SMART: A single-cycle reconfigurable NoC for SoC applications, in: Proceedings of the Conference on Design, Automation and Test in Europe, in: DATE '13, EDA Consortium, San Jose, CA, USA, 2013, pp. 338–343, [Online]. Available: http://dl.acm.org/citation.cfm?id=2485288.2485371.
[50] L. Gurobi Optimization, Gurobi optimizer reference manual, 2018, [Online]. Available: http://www.gurobi.com.
[51] P. Bogdan, M. Kas, R. Marculescu, O. Mutlu, QuaLe: A quantum-leap inspired model for non-stationary analysis of noc traffic in chip multi-processors, in: 2010 Fourth ACM/IEEE International Symposium on Networks-on-Chip, 2010, pp. 241–248.
[52] R.G. Askin, G.J. Hanumantha, Approximate queueing network analysis for nonstationary demand, IFAC-PapersOnLine 49 (12) (2016) 1502–1507, 8th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896316310655.
[53] W.-P. Wang, D. Tipper, S. Banerjee, A simple approximation for modeling nonstationary queues, in: Proceedings of IEEE INFOCOM '96. Conference on Computer Communications, Vol. 1, 1996, pp. 255–262.
[54] J. Wang, Y. Li, C. Wu, An analytical model for network-on-chip with finite input buffer, Front. Comput. Sci. China 5 (1) (2011) 126–134, [Online]. Available: https://doi.org/10.1007/s11704-010-0117-0.
[55] S. Abadal, A. Cabellos-Aparicio, E. Alarcón, J. Torrellas, WiSync: An architecture for fast synchronization through on-chip wireless communication, 2016.
[56] N. Jiang, G. Michelogiannakis, D. Becker, B. Towles, W. Dally, Booksim interconnection network simulator, 2016, Online, https://nocs.stanford.edu/cgibin/trac.cgi/wiki/Resources/BookSim.
[57] D. Rahmati, S. Murali, L. Benini, F. Angiolini, G.D. Micheli, H. Sarbazi-Azad, A method for calculating hard qos guarantees for networks-on-chip, in: Computer-Aided Design - Digest of Technical Papers, 2009, pp. 579–586.
[58] C. Seiculescu, S. Murali, L. Benini, G. De Micheli, Sunfloor 3D: A tool for networks on chip topology synthesis for 3-D systems on chips, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 29 (2009) 9–14.

Mohammad Baharloo received the PhD degree in computer engineering from the University of Tehran, Iran, in 2019, the MS degree in computer engineering from the Sharif University of Technology, Iran, in 2008, and the BSc degree in computer engineering from the Bahonar University, Iran, in 2006. He is currently postdoc researcher at the Institute for Research in Fundamental Sciences (IPM), Tehran, Iran. His research interests include chip-scale wireless communications, performance modeling/evaluation of network-on-chip, high-performance and low-power multi/many-core processors, and hardware accelerators for machine learning.

Ahmad khonsari received the BS degree in electrical and computer engineering from Shahid-Beheshti University, Iran, in 1991, the MS degree in computer engineering from the Iran University of Science and Technology, Iran, in 1996, and the Ph.D. degree in computer science from the University of Glasgow, United Kingdom, in 2003. He is currently an associate professor in the Department of Electrical and Computer Engineering, University of Tehran, Iran. He has been with School of Computer Science, Institute for Research in Fundamental Sciences (I.P.M.), Iran as a resident researcher since 2004 and also as a member of Scientific Council since 2012. His research interests are performance modeling/evaluation of computer and communication systems, wired and wireless networks, mobile and ubiquitous computing, distributed systems and high performance computer architecture.

Mahdi Dolati received his B.Sc. and M.Sc. degrees, both in software engineering, from the University of Tehran and Sharif University of Technology, respectively. He is currently a Ph.D. student with the department of electrical and computer engineering at the University of Tehran. He was a Ph.D. Visiting Student with the department of computer science at the University of Calgary from 2018 to 2019. His research interests include resource allocation and performance optimization in the virtualized software-defined networks.

Pouya Shiri received his B.Sc. in Electrical Engineering and his M.Sc. in Computer architecture from Shahid Beheshti University and University of Tehran respectively. He is currently a Ph.D. student in ECE department of University of Victoria, studying Deep Learning. His research interests include machine learning and parallel and high performance computing.

Masoumeh (Azin) Ebrahimi received a Ph.D. degree with honors from University of Turku, Finland in 2013 and MBA in 2015. She is currently a senior researcher (docent) at KTH Royal Institute of Technology, Sweden and an adjunct professor at University of Turku, Finland. Her scientific work contains more than 100 publications including journal articles, conference papers, book chapters, edited proceedings, and edited special issue of journal. The majority of work has been performed on interconnection networks, fault-tolerant methods, and neural network accelerators. She actively acts as a guest editor, organizer, and program chair in different workshops and conferences. She is an IEEE senior member.

Dara Rahmati (M'08) received the B.Sc. and M.Sc. degrees in computer engineering from the University of Tehran, Tehran, Iran, in 1998 and 2001, respectively, and the Ph.D. degree in computer engineering from the Sharif University of Technology, Tehran, in 2012. He is currently Assistant Professor with the Computer Science and Engineering Department, Shahid Beheshti University, Tehran, Iran. He is also the head of the High Performance Computing (HPC) center in the Institute for Research in Fundamental Sciences, Tehran, Iran. His current research interests include computer architecture, performance evaluation of networks-on-chip, hardware accelerators for machine learning and test of digital systems. Dr. Rahmati is a member of IEEE.