Engineering Science and Technology, an International Journal 50 (2024) 101607

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

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

Full length article

AS-Router: A novel allocation service for efficient Network-on-Chip ☆

Monika Katta [a,*], T.K. Ramesh [a,*], Juha Plosila [b]

[a] Department of Electronics & Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Bengaluru, India
[b] Autonomous Systems Laboratory, Department of Future Technologies, Faculty of Science and Engineering, University of Turku, 20014 Turku, Finland

ARTICLE INFO

Keywords:
Network-on-Chip
End-point congestion
Switch allocation

ABSTRACT

Assigning input ports to output ports and allowing flits to pass through the switch without colliding is accomplished by Switch Allocation (SA), a crucial pipeline stage in the Network-on-Chip (NoC) router. Earlier research improved the efficiency of matching to enhance SA processes. The studies in question, however, failed to account for the importance of information sharing between the various phases of the router's pipeline. To improve NoC routers' allocation, this article introduces the Allocation Service (AS) concept. By adding a new pipeline stage called Request Finalization (RF) before the SA stage, the AS can be optimized. The RF stage's objective is to complete the uniform request and Endpoint Congestion Causing (EPC) requests for SA before sending them on their way. These results inform the proposal of a new design, the CUE-Router, to increase AS by the integration of an RF stage and inter-pipeline communication between routers. Two new types of router architecture form the basis for CUE-Router. The first thing we do in this work is introduce URR-Router, a router that only allocates EPC requests if there are no uniform requests in the SA stage, effectively giving uniform requests a priority. The second example is the EPR-Router, which alleviates endpoint congestion by favoring EPC requests over uniform ones. In this work, we take a fresh look at the SA process optimization that underpins the low-latency CUE-Router architecture by cycling through a series of prioritization changes between uniform requests and EPC requests. CUE-Router greatly improves SA efficiency and network performance by facilitating communication between the various stages of the pipeline. Based on the results of the analysis, our approach has the potential to significantly enhance performance with minimal additional effort.

1. Introduction

With the involution in the applications of Network-on-Chip (NoC), its efficient design remains a challenge, as it deals with interconnecting more than hundreds of cores [1]. Network latency is largely determined by router micro-architecture, which has been included in many existing works to improve the design of NoC [2]. The Input-Queued (IQ) router is the most well-known router design [2,3] which deals with five stages of the pipeline, namely the Routing Computation (RC) stage, Virtual-Channel Allocation (VCA), Switch Allocation (SA), Switch Traversal (ST) and Link Traversal (LT). The majority of current research focuses on improving one of the IQ router's pipeline stages. The most crucial phase of the IQ router pipeline is SA, as it determines a router's throughput by guaranteeing a conflict-free flit transmission in the ST stage, with the assignment of output ports to input ports. Many existing works focus on improving the matching efficiency in the SA stage [4–9]. To maximize the matching effectiveness, authors have advised treating the incoming requests using time series during the SA stage [3,8]. Other works have added dedicated circuitry to bypass the routers [1,9]. However, none of these works have suggested maximizing the Allocation Service (AS), while implementing SA strategies.

We define AS as the router's ability to improve or enhance further functions during allocation of packets in the SA stage. AS can be enhanced by incorporating the allocation results into later pipeline stages to improve router features like adaptive routing choices and congestion control. Prior research that aimed to maximize matching solely considered the SA stage's efficiency. The majority of current work bases its allocation decisions only on data from SA requests, rarely taking into account data from other pipeline stages or efforts to boost the efficiency of other pipelines. If taken into account, AS can be enhanced to give increased matching efficiency in SA and hence enhance the performance of NoCs. AS is the association between the pipeline stages to have informed decisions while allocating ports in the SA stage, which is the crucial point to improving NoC's performance. As a result, co-designing between router pipelines should be done

☆ Peer review under responsibility of Karabuk University.

* Corresponding authors.
E-mail addresses: er.monikakatta@gmail.com (M. Katta), tk_ramesh@blr.amrita.edu (T.K. Ramesh), juplos@utu.fi (J. Plosila).

https://doi.org/10.1016/j.jestch.2023.101607
Received 18 January 2023; Received in revised form 14 November 2023; Accepted 30 December 2023
Available online 12 January 2024

to enhance the AS in SA. The following techniques can be used, in particular, to enhance AS through the co-design of router pipelines.

To begin with, endpoint congestion can be lessened by using the information about congestion from the RC stage in SA. Endpoint congestion and network congestion are a part of on-chip congestion. When a network channel or a network is unable to transfer a packet promptly, this is referred to as network congestion. Adaptive routing, which aims to boost routing adaptability, may help to lessen the effects of network congestion by sending packets over congested network channels [10–14]. Authors of recent research studies [10] seek to lessen endpoint congestion in the RC stage by restricting the resources utilized by Endpoint-Congestion-Causing (EPC) packets during the RC stage. Several packets that have the same destination are referred to as EPC packets because they can result in endpoint congestion. Other packets, such as those sent at random and those with diverse destinations that may cause network congestion, are not EPC packets. When an EPC flit advances, the other one becomes an ordinary flit (if there are no more flits to the same destination node). Traditional switch allocators, on the other hand, can weaken resource limitation because they treat EPC and normal requests equally. In addition, endpoint congestion brought on by tree saturation [2] and output port congestion may limit the number of requests that make it to the switch allocator, lowering matching efficiency. As a result, the SA should take the impact of EPC into account.

Second, by using SA stage results to better manage other pipeline stages, the switch allocator will receive more requests that are conflict-free. Routing algorithms have traditionally been created to relieve network congestion rather than improve the efficiency of other pipeline stages such as SA. The outcome of RC can have a direct impact on the SA matching efficiency. In-depth, if the RC process selects an output port that is already in use by other requests, conflicts are bound to happen in that particular output port, thereby limiting the performance of the SA stage. Consequently, the results of the SA stage need to be considered during the RC stage to ensure that the matching efficiency is maximized in the SA stage.

This paper proposes a novel category of routers that we refer to as AS-Routers, with the goal of maximizing the AS. The AS factor is improved by using the two methods described in the previous paragraphs and following which three instances of routers are proposed. In order to limit the distribution of EPC requests during the SA stage and mitigate the effects of endpoint congestion, we first develop a Uniform Request Recognize (URR) Router. In URR-Router, we suggest using the data from the requests during the RC stage, contributing to the EPC. To implement the design of URR-Router, the strategy is giving the highest priority to the uniform requests in the SA stage. URR-Router anticipates endpoint congestion and seeks to mitigate its impact in the SA, resulting in higher SA performance over long periods. We then design a second router namely, Endpoint-Congestion-Causing Request (EPR) Router, which reduces the endpoint congestion by giving priority to EPC requests over uniform requests in the SA stage. In EPR-Router, EPC requests are allocated using a special priority mechanism along with the other uniform requests, whose output ports are different than the output ports of the EPC request selected. This helps in getting further improvement in matching efficiency in the SA stage over some time. Finally, we propose Combining URR EPR (CUE) Router, based on URR-Router and EPR-Router, to make efficient use of information received from the new pipeline stage, namely Request Finalization (RF), to optimize the SA process. URR-Router and EPR-Router are partial mechanisms used in the design of CUE-Router. CUE-Router is designed by combining both URR-Router and EPR-Router to optimize the SA process by switching the priority between uniform requests and EPC requests one after another. Because low-priority requests would have little possibility of being allotted for a long time, we have carefully constructed the SA process to reduce the impact of unfair allocation. Since the NoC flit size is relatively wide, our method works well for two-dimensional (2D) mesh. We favor 2D mesh topology in this study due to its scalability and regularity. The proposed design is also implemented for Flattened butterfly (FBFly) topology; however, it is not suited for 3D. The lightweight implementation of our idea allows for effective latency reduction with minimal hardware overhead.

The remaining portions of this paper are structured as follows. Related research is introduced in Section 2, including endpoint congestion and the origins of SA in NoC. We give a scheduling example in Section 3 to show how the SA procedure works for our suggested architecture. A thorough router architecture of the proposed design is presented in Section 4. To show how the proposed design's SA procedures outperform earlier allocation algorithms, a SA example is also given. Section 5 discusses the implementation of our design and results. Finally, in Section 6, the article is concluded.

2. Related work

For on-chip routers, SA is the most crucial pipeline stage since it influences the latency of packets in NoCs. To facilitate packet transmission over the switch, the input port's packets are now mapped to the output port. The main purpose of the SA stage is to allow the smooth movement of packets from the input port to the corresponding output port without any conflicts in the mapping process. Every flit undergoes SA, and a period is set up for each flit to travel through the switch. To get efficiency in SA, previous popular works maximize the matching number in one cycle such as iSLIP, augmenting paths, and wavefront [3,7]. However, these works either need repeated cycles or are too complex to achieve desired performance. Many prior works have used the time-series method of allocation for achieving efficiency in SA namely, Packet-Chaining [8] and Pseudo-circuit [4], these methods make use of allocation information from the previous cycle to the current SA stage. Other works like TS-Router [3] uses future request allocation to take current SA decision.

Other works add extra hardware to increase the matching efficiency in the SA stage. Multiple input VCs in any input port can pass across the switch in one cycle thanks to the virtual input crossbar [9]. To facilitate packet transmission during the SA, SB-Router [15] arranges the input buffer as a swapped buffer. To reduce router frequency, authors in [16,17] break the SA step into numerous sub-stages and pipeline them. However, these works rarely take AS or information from other pipeline stages into account when improving SA to improve NoC performance. The benefit of our proposed methodology is to lessen the impact of endpoint congestion. Endpoint congestion can be exacerbated by a large number of short packets. Most NoC packets only contain a control instruction or a memory address, and these packets are known as short packets. The vast majority of these packets have the same output port, resulting in endpoint congestion. Endpoint congestion can harm SA matching efficiency due to occupancy in output ports and, as a result, a decrease in the count of SA requests over time. To the greatest of our knowledge, no previous work has been done to relate endpoint congestion details to the SA technique. Previous works have designed various types of traffic control strategies to alleviate the influence of EPC. Authors in [18] have suggested a reservation protocol using a short message to target the minimization of EPC initiated by short messages. Authors in [19] have discussed using the congestion information from the intermediate routers to recognize EPC at the particular node. Authors in [20] strive for achieving efficient reservation, especially for short packets by chaining them flexibly using reservation granularity. However, these methods are complex for adoption in NoC to manage congestion. Another approach is congestion notification [21,22] that relives EPC by using a hardware approach. A method for reducing traffic injection rate based on network congestion is the Explicit Congestion Notification (ECN) [23]. However, due to the time required to detect and throttle congestion-causing traffic, ECN responds slowly. Furthermore, ECN's performance is significantly impacted by its throttling parameters. Speculative Reservation Protocol (SRP) uses a handshake reservation between the starting and

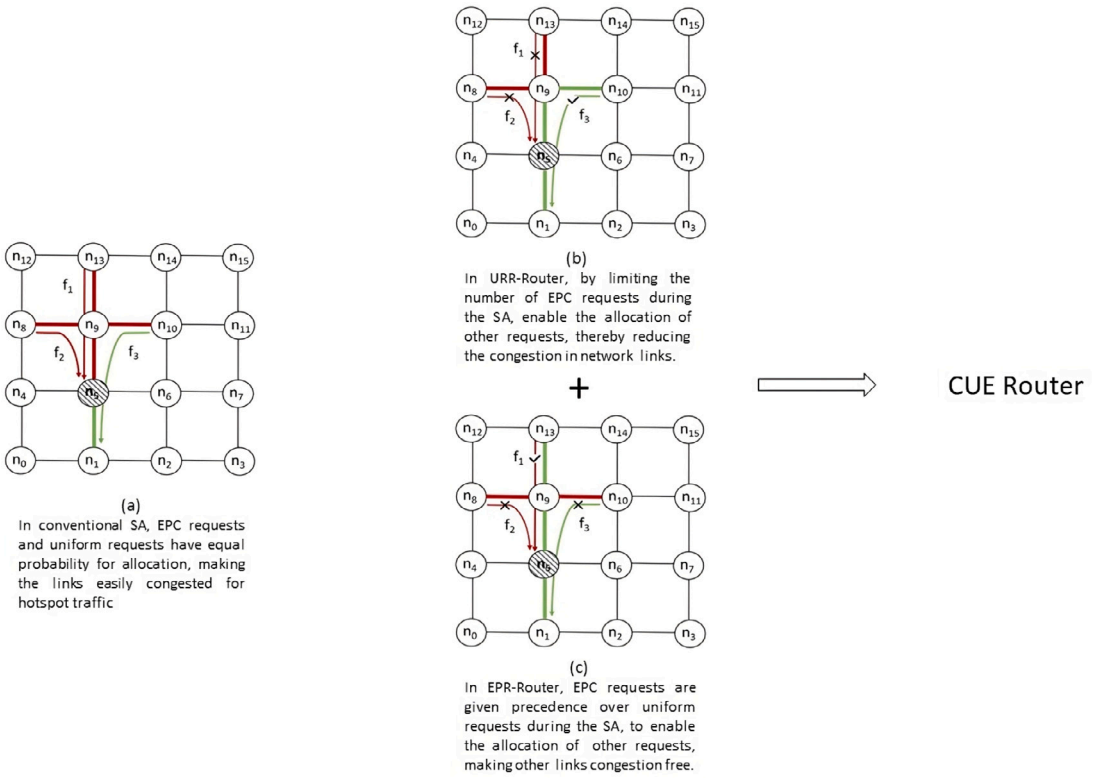

Fig. 1. An instance showing the benefits of URR-Router and EPR-Router, the red links showcase the congested links, those can be reduced and the packets having uniform traffic can be sent through free links showcased by green links.

intended nodes to prevent overloading in network endpoints [24]. In oversubscribed networks, some SRP-based studies employ multiple resource reservation protocols [25]. These methods, however, are too complicated to be used in NoC.

Recent research focuses on mitigating the effects of endpoint congestion through adaptive routing optimization [10]. By offering more path or port options to avoid the crowded link, adaptive routing suitably mitigates network congestion-induced performance loss [10–14]. However, even though these adaptive routing techniques can improve network performance in the face of adverse traffic conditions, they cannot reduce endpoint congestion and may even make matters worse when combined with congestion regulation techniques [26]. According to CBCM [26], it is crucial to distinguish between network congestion and endpoint congestion since adaptive routing might be challenging to handle when faced with endpoint congestion. By restricting the number of hot-spot packets that are queued on the footprint VC, footprint [27] is proposed to reduce Head-of-Line (HoL) blocking [28] brought on by endpoint congestion. The influence of improved routing algorithms, however, may mitigate the negative consequences of endpoint congestion in the SA process. The switch allocator may prioritize EPC requests over uniform requests during the SA process. Therefore, it is necessary to completely remove the impact of endpoint congestion from the SA process. Fortunately, this can be done by including additional links, as we suggest in our design.

3. Motivation

Limiting the allocation of EPC requests during the SA stage is the goal of URR-Router. Endpoint congestion occurs when a hotspot node receives a large number of packets having the same destination. In light of this, packets that are moving through the network sharing the same destination might be categorized as EPC packets. As a result, if URR-Router senses that various arriving packets share the same destination, these packets are classified as EPC packets. By carefully restricting the EPC request allocation in the SA stage, output port congestion can be relieved, which results in an increase in the free links in the network over some time. As shown in Fig. 1, there are many advantages of using the proposed technique in 16-node 2-D mesh NoC. In Fig. 1, the red link (dark color) showcases the congestion and the green color (light color) denotes the congestion-free link. As illustrated in Fig. 1(a), permutation traffic is observed for the traffic flows shown below:

$$f_1 = n_{13} -> n_5$$
$$f_2 = n_8 -> n_5$$
$$f_3 = n_{10} -> n_1$$

When the conventional SA strategy is applied in the network, using Dimension-Order-Routing (DOR), node n_5 will be overcrowded by f_1 and f_2, resulting in endpoint congestion. This endpoint congestion in n_5 also propagates back-pressure up to the source nodes (n_8, n_{10} and n_{13}) forming a congestion tree, causing congestion in four links. As illustrated in Fig. 1(a), DOR fails to tackle endpoint congestion in such situation. However, Fig. 1(b) shows an illustration that by restricting the EPC requests during the SA step, the number of clogged links can be reduced. During the RC process, node n_9 of the router chooses the south port as the output port for f_1, f_2, and f_3. This shows that at the SA stage, there are three requests competing for the output port, out of which two are EPC requests and one is uniform request. Within node n_9 in the router, by limiting the allocation of EPC requests during the SA process, the congestion situation of the south output port can be mitigated. As a result, the links from n_{10} to n_9 and n_9 to n_5 would not be congested and f_3 can be transmitted to the node n_1 using a congestion-free link.

Endpoint congestion could be dealt with at the RC pipeline stage, but the SA stage is equally crucial for dealing with this problem. Instead of focusing on the endpoint node, URR-Router accomplishes this by identifying endpoint congestion at intermediate routers. When compared to RC-based optimizations, this technique detects endpoint

congestion earlier. Because endpoint congestion always leads to congestion in the intermediate router, this strategy is effective. Endpoint congestion is observed in node n_5, as shown in Fig. 1(a), but contention can be found within node n_9 during the router's SA stage. By optimizing the SA process in intermediate routers, we may therefore detect endpoint congestion and enhance NoCs performance. Contention requests can specifically be identified at the SA stage of the intermediate router, and their corresponding destinations can specifically be identified in the RC stage. Using this information in SA, the endpoint congestion can be determined and its negative effects on network performance can be alleviated.

Further, the design is modified for EPR-Router, by giving EPC requests precedence over uniform requests. This is done to assure fairness and boost the matching effectiveness of SA. To implement the design of EPR-Router, a novel technique is among EPC requests, a special priority mechanism is used to select any one request for allocation. Fig. 1(c) illustrates that at the SA stage, there are three requests competing for the output port, out of which two are EPC requests namely, n_{13} to n_5 and n_8 to n_5 and one is uniform request i.e. n_{10} to n_1. During the SA stage, one EPC request namely, n_{13} to n_5 is selected for allocation based on the priority mechanism, as a result the links from n_{13} to n_9 and n_9 to n_5 would not be congested and f_1 can be transmitted to the node n_5 using a congestion-free link.

Finally, we propose CUE-Router, based on URR-Router and EPR-Router, by combining both URR-Router and EPR-Router to optimize the SA process by switching the priority between uniform requests and EPC requests one after another. This combination ensures that the network will not be biased by prioritizing only one type of requests, rather by switching the priority between uniform requests and EPC requests one after another, the matching efficiency of SA will be boosted and the request allocation will not be biased. This also provides congestion free links and output ports over a period of time. We have efficiently designed the SA process to alleviate the influence of unfair allocation, where low-priority requests would not have any chance of being allocated for an extended period.

3.1. Benefits of URR-router, EPR-router and CUE-router

It is to be noted here that in conventional SA strategy as shown in Fig. 1(a), the endpoint congestion in n_5 propagates back-pressure up to the source nodes (n_8, n_{10} and n_{13}) forming a congestion tree, causing congestion in four links. When the same network is dealt with URR-Router, it gives three green links i.e. the links from n_{10} to n_9 and n_9 to n_5 would not be congested and f_3 can be transmitted to the node n_1 using a congestion-free link as shown in Fig. 1(b). Further, the EPR-Router allocates the requests in such a way that the links from n_{13} to n_9 and n_9 to n_5 would not be congested and f_1 can be transmitted to the node n_5 using a congestion-free link as shown in Fig. 1(c). The CUE-Router combines both URR-Router and EPR-Router by switching the priority between uniform and EPC requests one after another, thus making sure that links congested by different types of requests become congestion-free, and over some time the network performance will improve as the number of congestion-free link will increase.

Unlike URR-Router and EPR-Router, other measures (such adaptive routing and congestion management) can dynamically estimate endpoint congestion, but they only start to function once the congestion has started to harm network performance. Alternatively, CUE-Router detects endpoint congestion in intermediate routers rather than at the endpoint nodes and thereby provides a novel approach to solve endpoint congestion more quickly. It should be made clear that our solution only does local observation based on the specifications of the router where SA is performed. Our system, however, is unable to detect congestion in remote routers. For instance, in Fig. 1(a), in case the router links in node n_1 are congested, the router in node n_9 will not be able to recognize this congestion, making it impossible to properly determine whether f_1, f_2 or f_3 should have priority. The information from the remote router may be kept in the present router, which would address the issue. But getting information on the remote router does not compensate for hardware complexity. In our proposed design, we do not consider the integration of the remote router information in the decision-making of the current router.

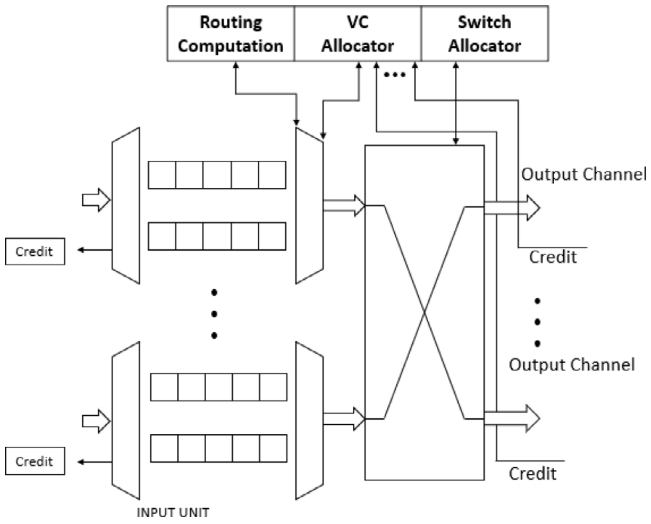

Fig. 2. Baseline Router Architecture for URR-Router, EPR-Router and CUE-Router.

4. Router architecture in detail

4.1. Baseline router

The baseline architecture under consideration is a cutting-edge router [29] as shown in Fig. 2. There are five pipeline stages in the baseline router: RC, VCA, SA, ST, and LT. It employs traditional VC flow control [30], with multiple VCs on each port at the input. Each VC associated with the input port has a personal buffer that can hold packets. The entire input port's buffer is made up of the cumulative buffer of all VCs that are present in it, and all of the VCs in it can share the input port's bandwidth. A VC serves as a kind of flit-buffering FIFO queue. We can gain a better grasp of the basic router design by seeing a packet's journey through the router. A packet is written into the input buffer as soon as it enters the input port; however, when the packet enters the VC's head, RC occurs. The VC allocator would then assign a free VC to the input buffer of the next-hop router. After then, a switch path would be assigned by the switch allocator for this packet to follow as it traveled from the input to the output port. The packet is routed to the output channel connection and after passing through the switch, it is sent to the closest reachable router. Each pipeline stage in our design requires only one cycle. This architecture serves as the foundation for Packet Chaining, URR-Router, EPR-Router, CUE-Router, TS-Router and other routers considered for comparison of results.

4.2. SA technique in URR-router

Endpoint congestion is reduced by URR-Router by restricting EPC request allocation during the SA stage. To execute the architecture of URR-Router, the technique is providing the highest priority to the uniform requests while enabling EPC requests to be allotted primarily when there are no uniform requests in the SA stage. Three requests are vying for output port S, as shown in Fig. 3, two of which are EPC requests (N→S and W→S) and one of which is a uniform request (E→S). The uniform request will be given priority (E→S) under the proposed method in the SA, and EPC requests will not be taken into account for allocation. The allocation of EPC requests will only occur in the absence of consistent requests.

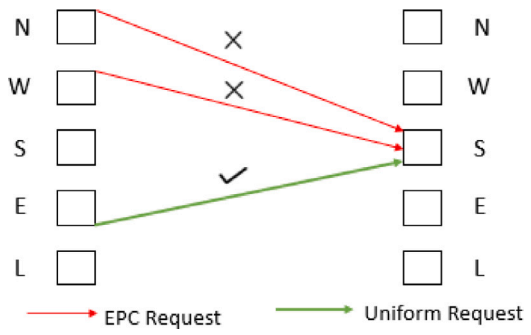

Fig. 3. A Switch Allocation Technique developed for URR-Router that provides the highest priority to the uniform requests while enabling EPC requests to be allotted primarily when there are no uniform requests in the switch allocation stage.

Although this approach can lessen the effects of endpoint congestion, it also introduces additional issues to the network. First, there is no assurance of justice during the SA stage. With this method, EPC requests will not be awarded for a while if there are uniform requests. This causes the starving of EPC requests. This can lead to an unfair degradation in performance in the matching efficiency and can be seriously degrading performance, especially in the routers close to the destination of EPC requests. This is a result of EPC requests being held for a long time, which might lower the number of packets that reach the endpoint node and lower the efficiency of the routers attached to such nodes. In addition to this, if temporary congestion is improperly detected as endpoint congestion, such requests will be restricted from allocation in the SA stage, causing poor matching efficiency in the SA stage.

4.2.1. Fairness-guaranteed using a priority mechanism in SA strategy

To enhance the efficiency of matching in SA and to ensure fairness, a novel SA strategy is proposed in URR-Router. In this technique, EPC requests can be allocated along with uniform requests. To deal with EPC requests, a new stage called RF is added before the SA stage. In actuality, the RF process and router pipelines can run simultaneously. Yet, in this case, we treat RF as an independent pipeline stage for a better explanation. The RF stage picks one or more of the EPC requests depending on a priority mechanism after detecting all of the EPC requests. Following the priority assigned to them, VCs are prioritized at the input port. When a flit enters the router, the input port priority is presumed to be an integer, and the input port with priority n (n = 0, 1, 2, …, 5) belongs to the *n*th class. A given input port's priority is indicated by the *n*th class of the input port. A lower index indicates a higher priority for an input port. The output port is allocated to the lowest index one, which receives priority based on the index of an input port.

The volume of chosen queries is independent of network design or traffic. In actuality, all EPC requests have an equal number of unique destinations as there are requests that were selected. The RF process gathers and evaluates each request's destination to determine whether it is an EPC request; if many requests have the same destination, they are all classified as EPC requests. After identifying all of the EPC requests, the RF stage will choose the EPC requests depending on the suggested priority strategy. It is important to note that the RF method compares the destinations of each request individually rather than all requests with the same output. This is because, in order to reduce the effects of endpoint congestion on time series, URR-Router strives to identify all EPC requests. While other non-chosen EPC requests are deleted, the switch allocator will receive and process the chosen EPC requests as normal requests. The SA will also be informed of the additional uniform requirements. The number of EPC requests that were chosen is the same as the variety of destinations in the EPC requests. The standard SA stage will be carried out after the RF stage.

A sample of the RF and SA stage of the URR-Router is shown in Fig. 4. For this situation, the SA process adopts an output-first separable instance [31]. The SA process comes after the RF procedure, as shown in Fig. 4(a). The three EPC requests are N→S, W→S, and W→E and the remaining requests are uniform requests. As can be seen in the Fig. 4(a), the input port E is simultaneously receiving two requests. This is a result of the potential for numerous VCs in a single input port to simultaneously compete for switch traversal. Each VC is given a chance to compete for the switch. Along with all the uniform requests, the EPC requests selected in the RF process, based on the priority mechanism, are added for SA. The priority mechanism is explained at the beginning of this section. The number of output ports chosen is the same as the variety of destinations in the EPC requests waiting for their allocation. As a result, along with the other standard requests, the EPC requests N→S, and W→E are chosen and approved for SA. In the output arbitration of SA, as illustrated in Fig. 4(b), the EPC requests N→S, and W→E are selected while discarding the uniform requests E→S, and L→E because these requests contradict with the EPC requests on output port S and E respectively. Finally, the switch allocator makes decisions on three requests during the SA input arbitration, N→S, W→E, and E→L, including two EPC requests as shown in Fig. 4(c). As illustrated in Fig. 4, the URR-Router may provide fairness in the SA stage by adopting a priority method to distribute the EPC requests while assuring fairness. Restricting the distribution of EPC requests, this tactic can lessen the effects of endpoint congestion.

4.3. SA technique in EPR-router

By giving EPC requests precedence over uniform requests, EPR-Router lowers endpoint congestion. To implement the design of EPR-Router, a novel technique is among EPC requests, a special priority mechanism is used to select the request for allocation. Fig. 5 illustrates an instance of the SA strategy using this technique. In this technique, EPC requests are allocated using a special priority mechanism along with the other uniform requests, whose output ports are different than the output ports of the EPC request selected. Three requests are vying for output port S, as shown in Fig. 5, two of which are EPC requests (N→S and W→S) and one of which is a uniform request (E→S). With the proposed technique in the SA, the EPC requests will be prioritized (N→S) and among EPC requests based on the priority mechanism, one or more requests will be selected and the total number of requests selected is equal to the number of different output ports. In this case, all the EPC requests are fighting for the same output port i.e. S, and one of them, N→S will be selected for allocation.

An instance showcasing EPR-Router is illustrated in Fig. 6. This strategy can prioritize the EPC request and relieve endpoint congestion but brings new problems. First, a fair chance for all the incoming requests to appear for SA is not guaranteed. In addition to this, the uniform requests destined to the output ports same as the output ports of the EPC request selected will be kept on hold, resulting in low matching efficiency in SA.

4.4. CUE-router: A fair SA technique for both uniform requests and EPC requests

URR-Router reduces the impact of endpoint congestion by prioritizing uniform requests over EPC requests and EPR-Router reduces the endpoint congestion by prioritizing the EPC request over uniform request. This is done by collecting information in RF stage and sending the same to SA stage. By merging URR-Router and EPR-Router, CUE-Router is created to enhance the SA process by sequentially altering the priority between uniform requests and EPC requests in order to further boost SA performance. The rationale for the alternating policy is to reduce the effects of unfair allocation, wherein low-priority requests would have little possibility of being allocated for an extended period of time. This ensures a fairness in giving opportunity to the requests to

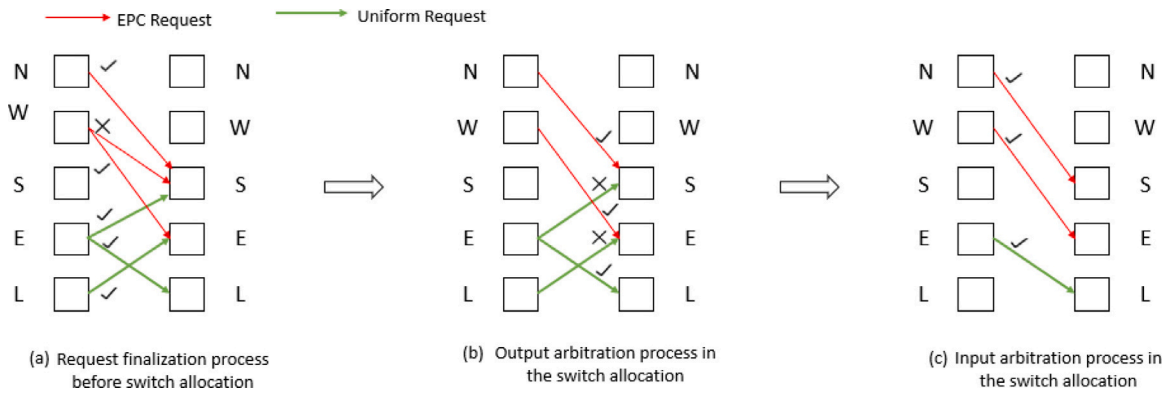

Fig. 4. A specific instance of the switch allocation strategical procedure for the URR-Router. Before switch allocation, the RF process is performed to choose requests from among all EPC requests (highlighted in red), and the chosen EPC requests as well as all uniform requests (highlighted in green) would be used as input requests for the switch allocator. URR-Router's switch allocator is a conventional output-first separable allocator, and both the chosen EPC requests and other uniform requests have an equal chance of being allocated during the SA. A certain probability is given to EPC requests in this system, which limits the allocation of EPC requests while assuring fairness.

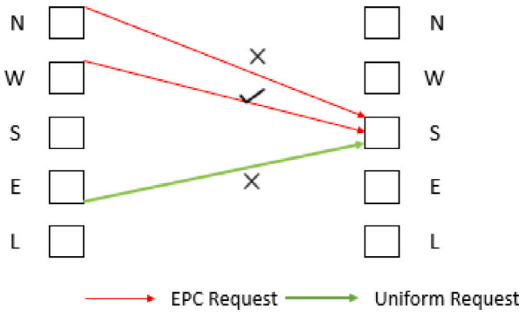

Fig. 5. A Switch Allocation Technique developed for EPR-Router, wherein EPC requests are allocated using a special priority mechanism along with the other uniform requests, whose output ports are different than the output ports of the EPC request selected.

be sent for SA. Moreover, neither uniform nor EPC requests will go in starvation state by using this technique.

Fig. 7 depicts an illustration of the RC process and the SA process in the CUE-Router, along with a comparison of the analogous processes in the baseline router. The baseline router likewise uses a conventional routing approach, but it is unable to make use of data from other pipelines to provide routing outcomes that are better suited for SA. CUE-Router's RC procedure allows users to choose an output port that will not be contended with additional SA requests in the ensuing cycle. The packet at input port S has two possible options for output ports, namely, *N* and E, as depicted in Fig. 7, and these two output ports have an equal chance of being chosen as the final output port. The switch allocator will receive a new SA request S→N in the cycle after an output port, let us say N, is selected (in this case, we use speculative SA [15], where SA is executed concurrently with VCA, causing a one-cycle delay between RC and SA stages). To lessen conflicts that the RC stage produces in the following cycle of SA, certain SA data must be acquired and applied. These details cover both the freshly created SA requests that will be submitted in the following cycle (L→E in the RC stage in Fig. 7) and the SA requests that were not allocated in the present cycle (W→S, W→E, and E→S in the RC stage in Fig. 7). These two sorts of requests will proceed to the SA stage in the subsequent cycle. Conventional adaptive routing algorithms fail to take this information into account when picking a route. Without this knowledge, a conventional routing approach will often randomly choose a potential output port (let us say, S→E), as shown in Fig. 7 (a). If the chosen output port conflicts with upcoming SA requests, the switch allocator will receive fewer conflict-free requests in the following cycle, which will result in fewer matchings being possible in the following cycle SA. However, if information regarding SA requests is taken into consideration during the RC process, then more SA matches can be made since the switch allocator is provided more conflict-free requests. To prevent adding more conflicts to the SA in the subsequent cycle, CUE-Router selects the routing path S→N rather than S→E, as shown in Fig. 7 (b). By making such a routing decision, CUE-Router enhances the number of matches in the SA in the subsequent cycle.

4.5. Pipeline design of URR-router, EPR-router and CUE-router

To optimize the SA process, the URR-Router, EPR-Router, and CUE-Router gather data during the RC stage. The simplest option is by incorporating an additional pipeline stage (RF) ahead of the SA stage to process RC information and finalize uniform and EPC requests to be sent for SA. However, the low latency properties of the on-chip router may be harmed as a result of such a design, which can cause a one-cycle increase in router latency. To alleviate the performance loss, we separate the RF process into two parts.

RF1 — The RF1 stage analyses the packets' destination information to identify EPC packets, because the input data (packet destinations) for the RF1 and the RC stage are the same, they can be run in parallel.

RF2 — The RF2 stage is executed in first phase (selection of EPC requests) and second phase (elimination of marked EPC requests).

RF2 first phase — simple selection logic can be used to implement the selection of EPC requests, and it can be carried out concurrently with the first two stages before SA.

RF2 s phase — During the SA stage, the marked EPC request can be directly disregarded. Because of this, the second phase of the second stage of RF can be embedded with the SA stage without significantly delaying the SA stage.

As discussed above, the first stage analyses the packets' destination information to identify EPC packets, and the second stage determines which EPC requests should be selected and disregarded before the SA operation. Because the input data (packet destinations) for the first stage and the RC stage are the same, they can be run in parallel. Selection and elimination of EPC requests is necessary for the second stage. Simple selection logic can be used to implement the selection of EPC requests, and it can be carried out concurrently with the first two stages before SA stage. By marking the EPC request before the SA process begins, the elimination of marked EPC requests can be easily implemented. During the SA stage, the marked EPC request can be directly disregarded. Because of this, the second phase of the second stage of RF can be embedded with the SA stage without significantly delaying the SA stage, even though it cannot be carried out in parallel with router pipelines. We also use the Cadence Encounter RTL compiler to calculate the critical path delay of the suggested router design to investigate the overheads in router frequency. The operating voltage is 1.8 V and the operating frequency is 1.0 GHz. 180-nm design libraries

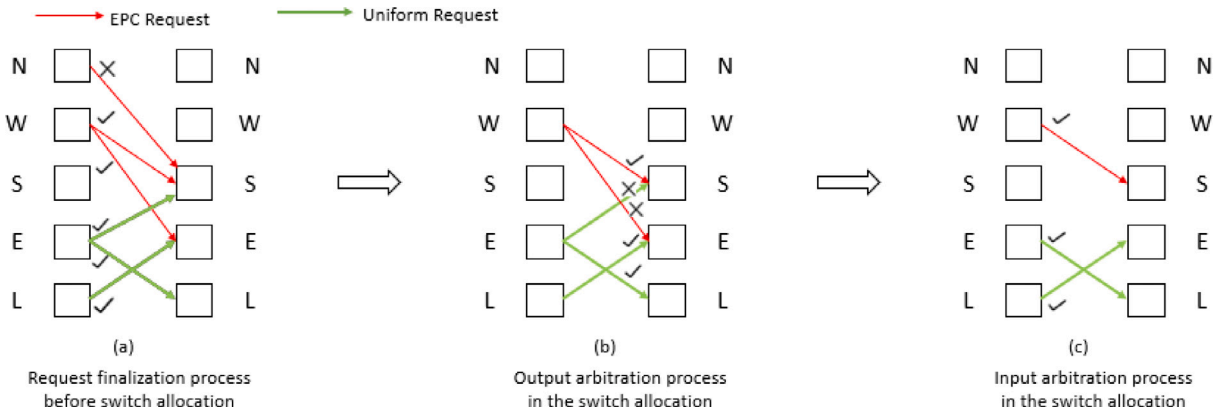

Fig. 6. A specific instance of the switch allocation strategical procedure for the EPR-Router. A novel technique is among EPC requests, a special priority mechanism is used to select the request for allocation (highlighted in red), and the chosen EPC requests as well as all uniform requests (highlighted in green) would be used as input requests for the switch allocator. This strategy can prioritize the EPC request and relieve endpoint congestion.

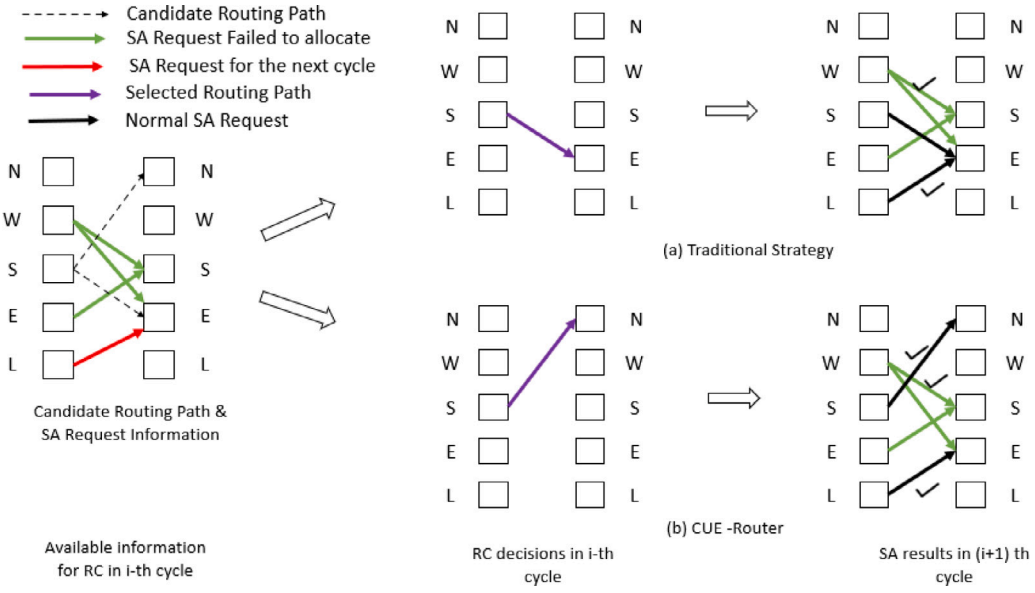

Fig. 7. Pipeline comparisons of the CUE-Router with the baseline router, which employs the conventional RC strategy. Before RC, certain data will be gathered, including the SA requests that have not yet been allotted for the current cycle and those that will arrive in the following cycle (these requests will be utilized as requests for the following SA cycle). Based on this data, CUE-Router can choose a routing path with the least amount of competition, allowing the SA process to get more requests without conflicts and, as a result, more matches in the following cycle. The chosen routing method may compete with incoming SA requests to reduce the matching efficiency because traditional RC algorithms, on the other hand, do not take advantage of this information.

are used to perform behavioral RTL. The outcome demonstrates that our design produces a router that runs up to 885 MHz, so it is 11.6 percent slower.

The datapath of the URR-Router, EPR-Router, and CUE-Router in a five-stage pipeline router is shown in Fig. 8. The figure depicts the addition of a single forwarding connection from the RC stage to the SA stage for the transmission of the EPC request data in the CUE-Router. The initial stage of RF's identification of the EPC request has been combined with the RC process. URR-Router and EPR-Router are the design model for CUE-Router. CUE-Router now has one more link. The one from the SA stage is fed back to the RC stage to transfer the information about unassigned SA requests back to the RC stage.

Since AS-Router's design is independent of specific router architectures or routing algorithms, it may be extended to other IQ routers as well. CUE is basically an allocation policy that can be applied to various routers within an NoC. It determines how communication requests are assigned to resources within the router, thus influencing congestion management and overall network performance. The router which applies this allocation policy is referred to as CUE-Router in our paper. The design of CUE-Router architecture is orthogonal to majority of prior SA techniques, because RF and SA stages are independent. As a result, the performance of the SA process of CUE-Router can be further enhanced by using previous state-of-the-art SA methodologies. Other adaptive routing techniques can be implemented with CUE-Router as well. With the aid of this design, we can select one of several possible routing paths with the same likelihood of success during the RC stage. Even though the routing paths are selected differently by different routing algorithms, the likelihood of selecting two paths at the same time is still quite high. In this way, the CUE-Router can be extended to the majority of other IQ routers that have a capable adaptive routing algorithm. However, CUE-Router cannot be combined with DOR. In DOR, the routing path is determined by sequentially traversing the dimensions of the grid and hence CUE-Router cannot be used to select an output port to reduce SA conflicts.

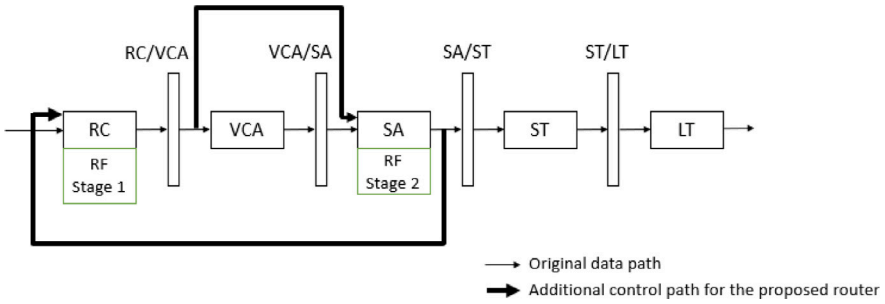

Fig. 8. The design can be implemented simply by inserting a new pipeline stage divided into two stages (RF stage 1 and RF stage 2). RF stage 1 is added ahead of the SA stage. Because the input data (packet destinations) for the RF stage 1 and the RC stage are the same, they can be run in parallel. For the second stage, simple selection logic can be used to implement the selection of EPC requests, and it can be carried out concurrently with the router's pipeline stage. By marking the EPC request before the SA process begins, the elimination of marked EPC requests can be easily implemented. During the SA stage, the marked EPC request can be directly disregarded. Because of this, the second phase of RF stage 2 can be embedded with the SA stage without significantly delaying the SA stage, even though it cannot be carried out in parallel with router pipelines. One extra link is added to CUE-Router. The one from the SA stage is fed back to the RC stage to transfer the information about unassigned SA requests back to the RC stage.

Table 1
Network parameters.

Network parameters	Values
Topology	2D-Mesh and Flattened Butterfly (FBFly) [32]
Routing Algorithm	Footprint [27]
Virtual Channel	2, 4, 8 per physical channel
Traffic	Transpose, Tornado, Uniform Random, Bitrev, Hotspot, PARSEC benchmarks
Size of packet	single-flit, 2,3,4.....16- flit packets
Flow Control	Wormhole (credit-based)
Virtual Channel Allocator	Round-Robin
Switch Allocator	Packet Chaining [8], iSLIP, TS-Router [3], URR-Router, EPR-Router, CUE-Router
Speedup	1.0

5. Performance evaluation

5.1. Methodology used for experimental evaluation

Booksim, a cycle-accurate simulator, is used to evaluate the design [33] for an interconnection network. The design is simulated for both the synthetic traffic and traces from realistic traffic from PARSEC. The network configuration is listed in 1. The network consists of an 8x8 two-dimensional (2-D) mesh as well as Flattened Butterfly (FBFly) [32] having 64 nodes. The proposed design is also simulated for 4x4 and 16 × 16 2-D mesh to check the scalability. These are the most popular topologies used to test the design and evaluate the performance of NoCs. Each router is connected to one router in 2-D mesh and four other routers in FBFly. All the channels are having a delay of one cycle. Conventional VC flow control [30] is employed. Each port contains 4 VCs. Every VC can accommodate 8 flits in a buffer and the proposed design is also evaluated by varying the number of VCs. Footprint [27] routing algorithm based on Duato's theory [34] has been chosen for both the mesh and the FBFly topology as it is a deadlock-free and efficient adaptive routing algorithm Section 4 describes the detailed router architecture. The credits can be transferred to the upstream router in 2 cycles. The baseline allocator for Packet Chaining, TS-Router, and the proposed router design is one-iteration iSLIP. In the network evaluation, Packet Chaining uses the same port but different VCs.

The design is evaluated for single-flit packets unless stated specifically. Four types of synthetic traffic patterns are considered for the design, namely, uniform, bit rev, transpose, and tornado. Moreover, the performance of the design is evaluated by designing hot-spot traffic for endpoint congestion conditions i.e. 10% nodes are chosen at random that accept traffic from other nodes and these are known as hotspot receivers. In the remaining nodes, there is a 20% chance that packets will be forwarded to hotspot endpoint nodes and an 80% chance that they will be forwarded to a randomly chosen node. The design is evaluated for the realistic traffic traces taken from PARSEC benchmark [35]. The traces are from 64-node network full-system simulations. We received a thorough analysis of the design from the power, area, and time analyses. Cadence Encounter is used to analyze the dynamic power of the detailed Booksim factors.

5.2. Synthetic traffic network performance

The design of the URR-Router and EPR-Router is evaluated under various traffic patterns for the mesh topology. The design is evaluated by comparing it to other popular SA techniques. Fig. 9 depicts the average packet latency observed and plotted. Fig. 9 shows that URR-Router and EPR-Router outcompete most other routers, and CUE-Router shows further performance improvement. The proposed design shows lower latency and higher saturation throughput for all the injection rates less than the saturation injection rate. In comparison to TS-Router, URR-Router improves saturation throughput by 4.34%, 2.85%, 3.70%, and 3.03% for uniform, transpose, tornado, and bit rev traffic, respectively. Under uniform, transpose, tornado, and bit rev traffic patterns, the EPR-Router improves throughput by 5.37%, 5.55%, 5.45%, and 4.47%, respectively. For each traffic pattern, URR-Router and EPR-Router show a reduction in packet latency, as the congestion is alleviated in the SA stage, as a result of which packet latency is reduced. CUE-Router shows further improvement in saturation throughput for each of the four traffic patterns. The results are 6.38%, 8.10%, 7.14%, and 5.88% improvement in saturation throughput for the four traffic patterns mentioned previously. To get higher performance than using URR-Router and EPR-Router separately, CUE-Router is built on the foundation of these two routers. The reason is the requests reaching the SA stage are conflict-free now.

Fig. 10 shows the plot of average packet latency measured in cycles v/s injection rate measured in flits/cycle/node under FBFly topology for tornado and transpose traffic. The improvement in latency for URR-Router is 17.5%, for EPR-Router is 30%, and for CUE-Router is 45% as compared to TS-Router at the injection rate below the saturation injection rate for tornado traffic. The average latency improvement is 21.05% for URR-Router, 26.31% for EPR-Router, and 31.57% for CUE-Router as compared to TS-Router for transpose traffic at the injection rate below the saturation.

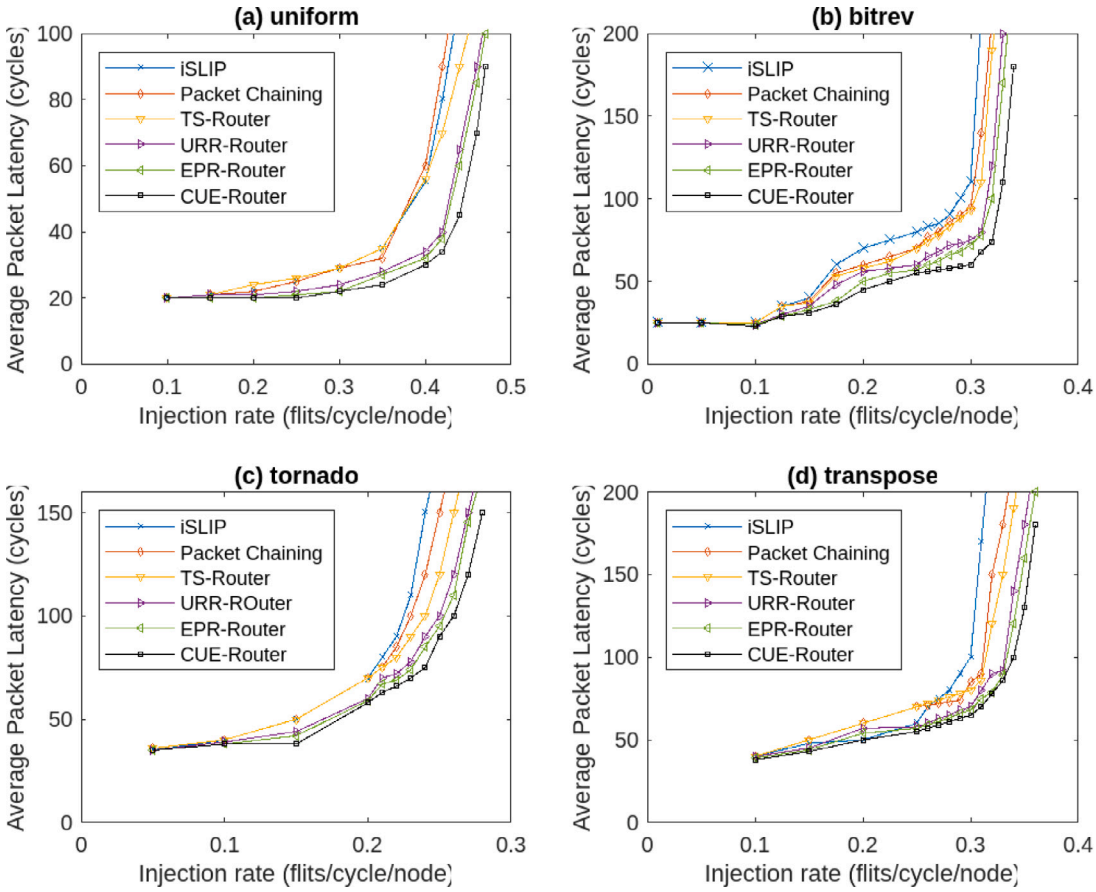

Fig. 9. Plot of average packet latency measured in cycles v/s injection rate measured in flits/cycle for (a) Uniform Random traffic (b) Bit Reverse traffic (c) Tornado traffic and (d) Transpose traffic under mesh topology.

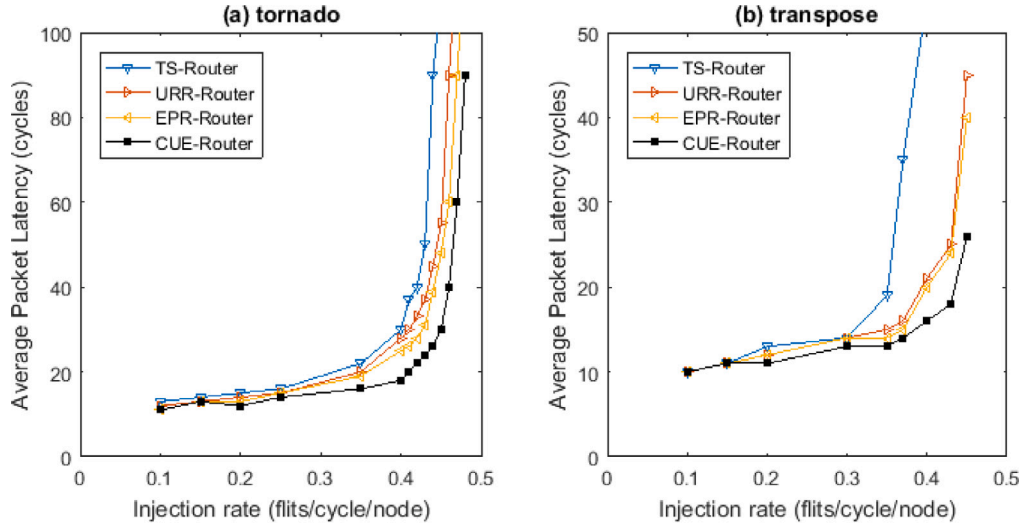

Fig. 10. Plot of average packet latency measured in cycles v/s injection rate measured in flits/cycle/node under Flattened Butterfly topology for (a) Tornado traffic (b) Transpose traffic.

It should be highlighted that the improvement in overall performance is insufficient to make up for the evaluation's decreased frequency (11.6%). This is due to a flaw in our architecture that will cause the complex SA procedure to lower router frequency. Yet, our solution offers an innovative way to boost SA's performance, and we can get even better performance by making further optimizations that balance out the router frequency overhead. Yet, our solution can deliver great performance in specific traffic scenarios or with bigger NoCs, which is sufficient to balance out the router frequency's overhead. The next paragraphs will give specific experimental findings.

Then we evaluate how many VCs affect NoC performance, as shown in Fig. 11. We start comparing our design to a popular router, the TS-Router, by changing the number of VCs. To avoid deadlock conditions in the RC stage, the input buffer should have at least two VCs [36]. We have chosen 2 VCs, 4 VCs, and 8 VCs in the design of the input buffer. The amount of requests turning up at the switch allocator increases

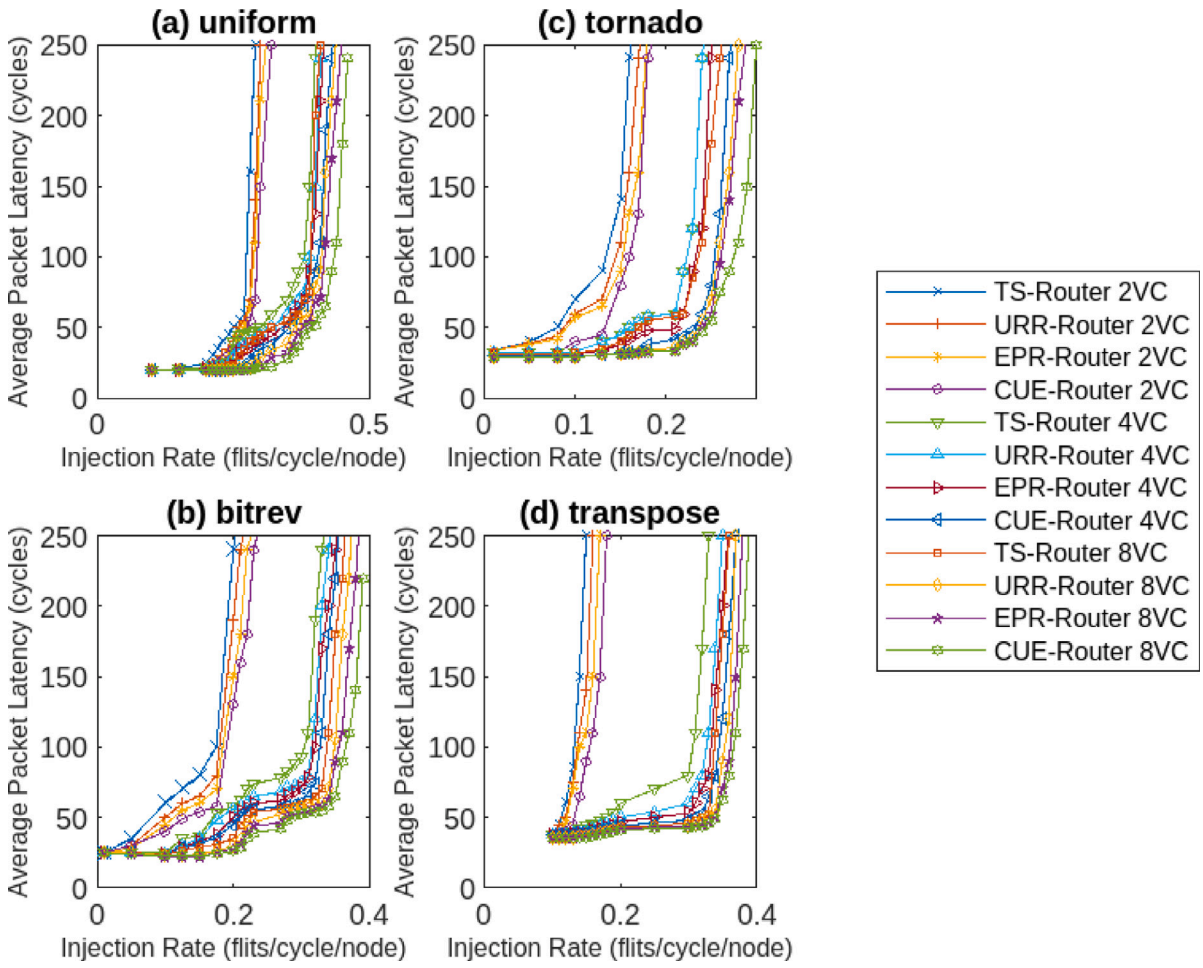

Fig. 11. Comparison of average packet latency for URR-Router, EPR-Router, CUE-Router, and TS-Router for various traffic patterns by varying the count of VCs.

as the amount of VCs increase, this increases the number of matching during the SA process. The injection rate is kept below saturation injection rate i.e., 0.25 flits/cycle/node for tornado traffic and 0.37 flits/cycle/node for transpose traffic under the mesh topology. From Fig. 11, it is observed that for various SA strategies, performance can be improved with the increase in the number of VCs. The SA method performs better as we increase the number of VCs. However, if we keep adding VCs, the performance bottleneck shifts from the volume of requests to the switch allocator's efficiency in allocating resources. This makes it abundantly evident that, as the number of VCs rises, the performance of various SA approaches will vary. Fig. 11 shows that when using the same count of VCs, URR-Router, EPR-Router, and CUE-Router outcompete the existing TS-Router in patterns of traffic. URR-Router gives performance improvement for saturation throughput by 3.33% for two VCs and 4.65% for 8 VCs, the corresponding saturation throughput improvement for uniform traffic patterns for EPR-Router are 3.33% and 6.81% respectively, and for CUE-Router 9.37% and 10.8% respectively.

For AS-Router, the most prominent reason for being efficient is to be able to remove the influence of endpoint congestion. To that end, we compare the proposed design's performance with that of other popular SA techniques in hotspot traffic and uniform mixed traffic. Hotspot traffic produces HoL blocking, which clogs other uniform flow and builds a congestion tree. This has the potential to reduce network throughput. Fig. 12 depicts the throughput latency curve, which demonstrates the benefits of our architecture in attempting to mitigate endpoint congestion's effects. Fig. 12 showcases that as compared with other SA strategies, CUE-Router can achieve quite a low packet latency for the injection rates less than the saturation point. When the injection rate is kept at 0.2 flits/cycle, CUE-Router reduces average packet latency by

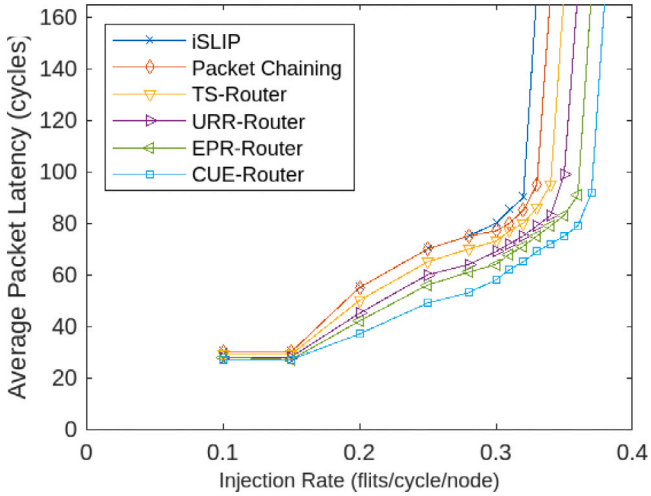

Fig. 12. Comparison of average packet latency measured in cycles for various routers under hotspot traffic.

26%, in comparison to TS-Router and 17.77% and 11.9% in comparison to URR-Router and EPR-Router.

Fig. 13 shows the outcome of the comparison of our solution with the TS-Router utilizing various network scales. The TS-Router is used to compare our design, and the results are presented in Fig. 13. For each traffic pattern, the throughput of the URR-Router, EPR-Router, and

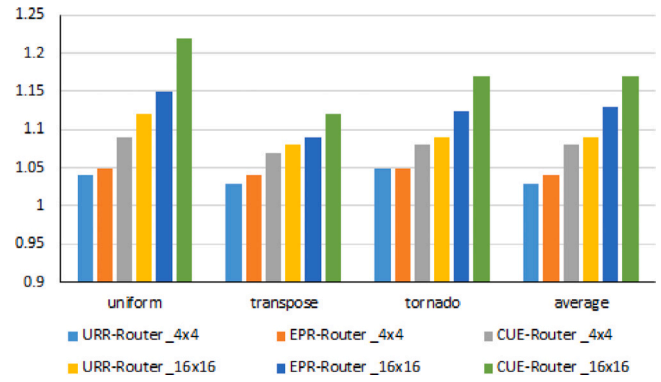

Fig. 13. Comparison of throughput of URR-Router, EPR-Router, CUE-Router, and TS-Router. The throughput of other routers is normalized to that of the TS-Router.

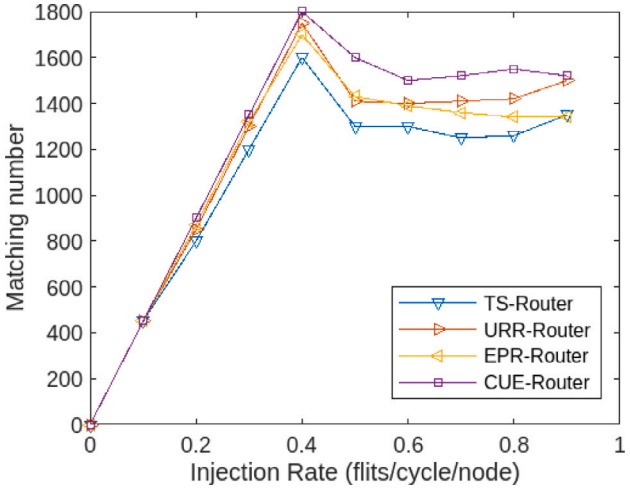

Fig. 14. Matching number during SA stage comparison for URR-Router, EPR-Router, CUE-Router, and TS-Router for the mesh network under uniform traffic.

CUE-Router is normalized to the TS-Router. As shown in Fig. 13, the increase in performance for our architecture is greater for the 16 × 16 mesh than for the 4 × 4 mesh; because a bigger network can aggravate the congestion.

In 4 × 4 and 16 × 16 mesh topologies, URR-Router outperforms TS-Router by 3.84% and 10.71%, respectively. EPR-Router achieves 4.76% and 13.04% improvement for these two meshes, while CUE-Router achieves 8.25% and 18.03% improvement. Since larger NoCs can more easily handle the router frequency overhead, the performance advantage of our solution is more apparent for them. We do not increase the number of VCs and the depth of VCs as well with the increase in the network size. As the number of VC and the depth of VCs increase, the performance of NoC shows improvement. The higher buffer size and the complex scheduling logic are difficult to adopt in the NoC router, even though adding more VCs or deeper VCs promises better results in NoCs. A NoC router's input buffer consumes a lot of power and space, making it difficult to include a large buffer [37–41].

The matching number refers to a notion involving routing algorithms. In NoC systems, routing entails choosing the best path or route for data packets to travel from a source node to a destination node. The matching number serves as a gauge of a NoC architecture's routing adaptability or capacity. Each node in a NoC has a network of communication links connecting it to a group of surrounding nodes. The matching number represents the most discontinuous paths that can be created between any two network nodes at once. A higher matching number suggests more flexibility in developing distinct communication

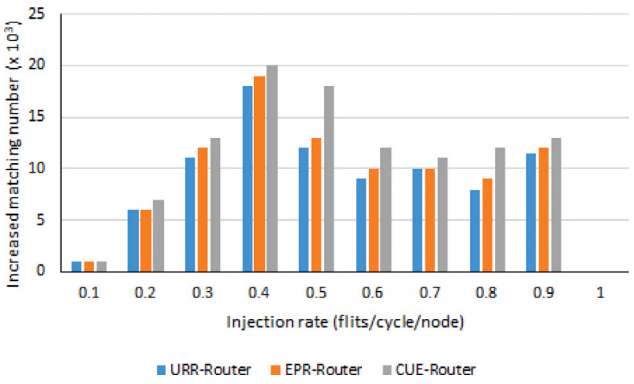

Fig. 15. For the mesh topology with a uniform traffic pattern, the improvement in the matching number during SA stage for URR-Router, EPR-Router, and CUE-Router over TS-Router.

pathways, allowing for better use of the network resources and possibly lowering congestion. On the other side, a lower matching number restricts the number of concurrent pathways and could cause more congestion or longer routing delays. As it affects the network's overall performance, scalability, and efficiency, the matching number is a crucial parameter in the design and analysis of routing algorithms for NoC systems. Based on their matching numbers, several routing algorithms and network topologies can be compared to determine which is best for a given application. In an evaluation, the number of matching for URR-Router, EPR-Router, CUE-Router, and TS-Router is compared as the injection rate is increased from 0 to 0.9 flits/cycle/node. Fig. 14 depicts the outcome of 1000 consecutive stable cycles for a mesh topology with a uniform traffic pattern. The network becomes saturated for the injection rate exceeding 0.4 flits/cycle/node, as shown in Fig. 14. As we further boost the injection rate after reaching saturation, the matching number falls instead of rising. The cause of this is that when there is output port congestion, fewer requests in total are received by the switch allocator, which lowers the number of SA requests. However, when the injection rate is close to or above the saturation injection rate, CUE-Router, EPR-Router, and URR-Router fulfill more matching due to the advantages of the SA process. Fig. 15 illustrates the result of our design's benefits in raising the number of matchings. Fig. 15 shows the increase in the matching count for URR-Router, EPR-Router, and CUE-Router over TS-Router. In an 8 × 8 mesh network, the matching amount of switch allocators is collected in 40,000 cycles for URR-Router, EPR-Router, and CUE-Router injection rates ranging from 0.1 to 1 flits/cycle/node.

For each injection rate, URR-Router, EPR-Router, and CUE-Router achieve greater matching than TS-Router as illustrated in Fig. 16. CUE-Router can generate more matching than URR-Router and EPR-Router. For the cases, when the packets are injected at less than 0.5 injection rate, the matching number increases with the increase in the injection rate. This is due to the fact that the matching number depends on the total number of requests; when more requests arrive at the switch allocator, the matching number grows as well. When the packets are injected at an injection rate greater than 0.5, the matching number starts to decline. The cause of this is the heavy load that runs the risk of blocking the router's output port. As a result, the number of valid requests decreases, as does the switch allocator's allocation efficiency.

URR-Router reduces the effect of endpoint congestion on average packet latency. This reduces packet waiting latency caused by endpoint congestion. This reduces average packet latency. Although CUE-Router alters how the routing algorithm is executed, performance is unaffected. The latency distribution of the TS-Router, URR-Router, EPR-Router, and CUE-Router are evaluated and the results for the Cumulative Distribution Function (CDF) are shown in Fig. 16. The test is carried out in a mesh 8 × 8 network with packets injected at

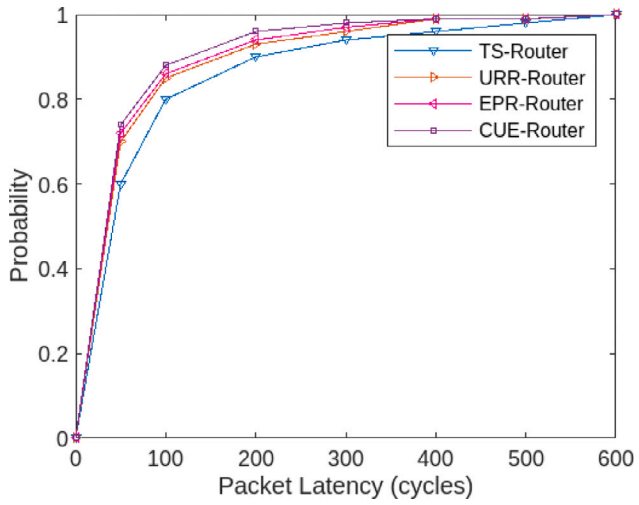

Fig. 16. Plot of cumulative distribution function of the packet latency for URR-Router, EPR-Router, CUE-Router, and TS-Router.

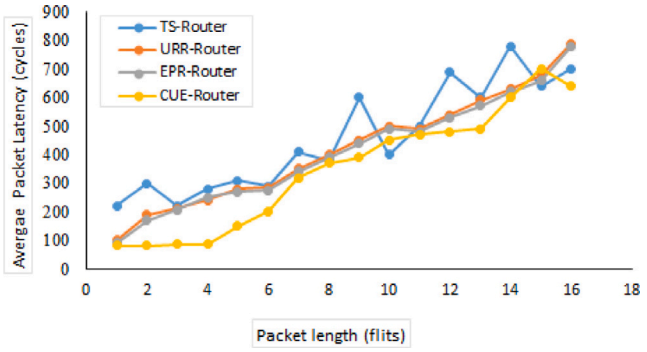

Fig. 17. Variation of packet length and measuring average packet latency for TS-Router, URR-Router, EPR-Router, and CUE-Router for mesh topology under uniform traffic.

the saturation injection rate. As shown in Fig. 16, our design has a higher proportion of low-latency packets than TS-Router. Furthermore, there is no significant tail latency in our design. In other words, when confronted with endpoint congestion, our design can avoid a long time waiting that causes long packet latency.

Next, we have also analyzed the impact of the length of the packet on the latency of the proposed NoC. It is possible to determine the average packet latency of the URR-Router, EPR-Router, and CUE-Router by varying the length of the packet between 1 and 16 flits. The outcomes are depicted in Fig. 17. For each test, the injection rate is set at saturation. As can be observed from the graph, URR-Router and EPR-Router performs better than TS-Router when the packet length is significantly less than 8 flits, and performs worse when the packet length is greater than 8 flits. As packet length rises, the URR-Router and EPR-Router performance suffers. Large packets limit the number of queries the SA process can send to the switch allocator, limiting its optimization. After the packet length exceeds 8 flits, TS-Router undergoes a mutation. The reason for this is that each VC's capacity is limited to 8 flits. Two packets having the same VC are highly unlikely to occur when the packet length is more than 8 flits. When the packet length exceeds 8 flits, the chances of two packets being in the same VC are extremely low. As a result, the possibility of obtaining information about subsequent requests is reduced, resulting in lower TS-Router performance. Most of the time, CUE-Router performs better than the other two approaches, particularly whenever the packet length is kept under 7. This shows that NoCs with a large percentage of short packets are better suitable for CUE-Router [11].

The proposed CUE-Router is compared to two other cutting-edge routers, the RoB-Router, and the SB-Router [15], as shown in Fig. 18. The injection rate is kept below the saturation injection rate i.e. 0.44 flits/cycle/node for tornado traffic and 0.37 flits/cycle/node for transpose traffic under Flattened Butterfly topology. The channel count between routers in FBFly is higher within the same dimension. Lower latency improvement is accomplished as a result of shorter queues waiting in the input port. In contrast to current routers, our suggested router further reduces latency, as demonstrated in Fig. 18.

5.3. Application-level network performance

The performance of our design has been evaluated and compared the results with TS-Router. The mesh network was built using traces from PARSEC 2.0 workloads [35]. Based on the application scenarios, eight PARSEC benchmarks have been selected to analyze the performance of our proposed design. For instance, fluidanimate, ferret, channel, and bodytrack are used for the application domains like animation, similarity search, engineering, and complex vision respectively. Financial analysis is the application domain for both blackscholes and swaptions; however, blackscholes and swaptions are employed for small and large working sets, respectively, to evaluate the influence of working set size. Other applications are x264 and vips from the media processing area, however, x264 uses a pipeline model of parallelization whereas vips use data parallelization. This can be considered for the evaluation of the input of various parallelization models. The applications are executed completely and traces have been taken for both the series phases and parallel phases of the benchmarks. The simulation results are observed after continuous injection of 1,000,000 cycles, for each benchmark.

Fig. 19 presents the results for application-level performance. The latency reduction (%) plots have been normalized to TS-Router for all benchmarks. It is observed from the Fig. 19 that URR-Router, EPR-Router, and CUE-Router, outperform TS-Router for all benchmarks. CUE-Router outperforms URR-Router and EPR-Router in terms of performance.

URR-Router shows an average improvement of 6.37% over all the benchmarks, however, the maximum improvement is 12% for the blackscholes benchmark. For EPR-Router, the average improvement achieved is 6.88% while the maximum improvement is 13% for the blackscholes benchmark. For CUE-Router, the average improvement is 8.75% and the substantial increase is 15% using the blackscholes benchmark. The PARSEC benchmark has a relatively low average injection rate, which results in fewer requests reaching SA. As a result, the performance increase for URR-Router is not that significant. This limits the performance of URR-Router and EPR-Router in mitigating the effects of endpoint congestion. However, CUE-Router shows much better improvement in performance than URR-Router and EPR-Router. The vast majority of the packets used in PARSEC benchmarks are small ones, thus injecting little packets into a network makes CUE-Router perform noticeably better.

5.4. Power and area estimation

The implementation of our design is done and we have taken utmost care to reduce the overhead. A new pipeline stage RF is introduced ahead of the SA stage for URR-Router, EPR-Router, and CUE-Router. This aids in the processing of RC information as well as the finalization of uniform requests and EPC requests that must be sent to the SA stage. The main overhead comes from these added registers, the destination of the request needs to be tracked in every VC. A $log_2(N)$ bits of registers are required per VC where N denotes the network size. For 4×4 2D mesh, with 4 VCs for each physical channel, it results in 16 bits of storage for each port. For each port, only 16 bits of storage is required, therefore for a 5-port router, 85 bits of storage is required. The extra

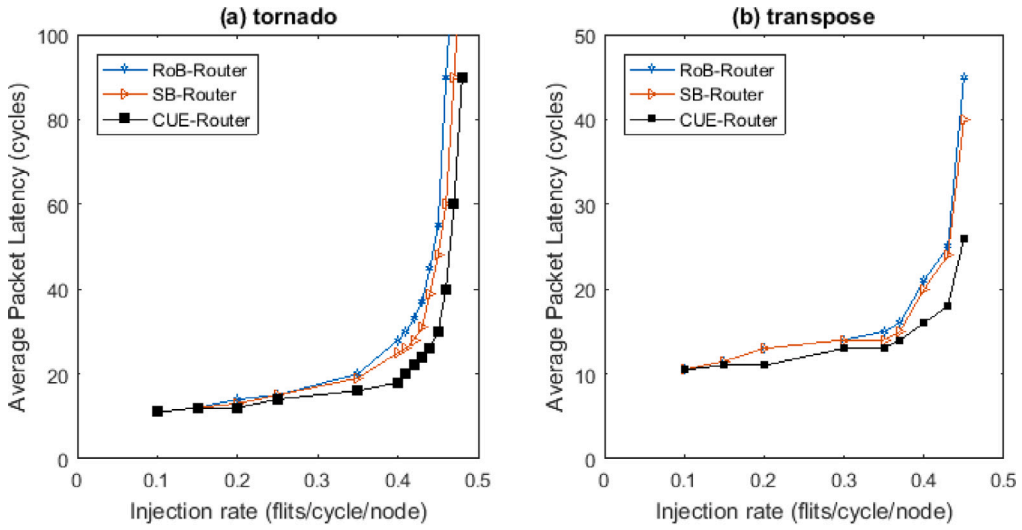

Fig. 18. Plot of average packet latency comparison of the proposed CUE-Router with other routers namely, RoB-Router and SB-Router, under Flattened Butterfly topology for (a) Tornado traffic (b) Transpose traffic.

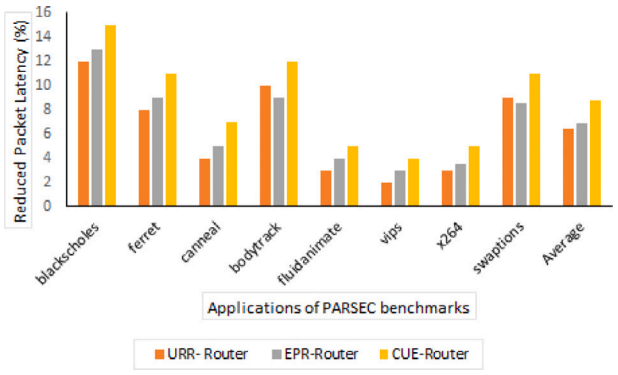

Fig. 19. Plot of reduced packet latency for URR-Router, EPR-Router, and CUE-Router in comparison to TS-Router under the mesh topology using PARSEC benchmarks.

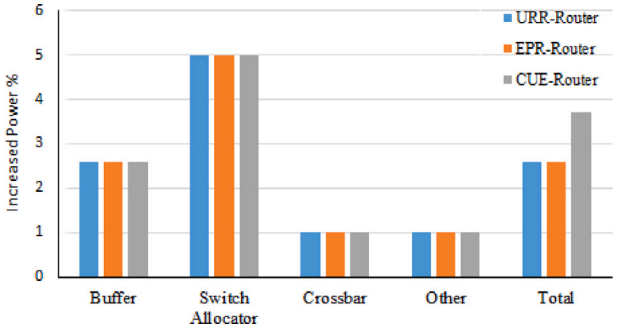

Fig. 20. Rise in power consumption for URR-Router, EPR-Router, and CUE-Router a compared to Baseline Router.

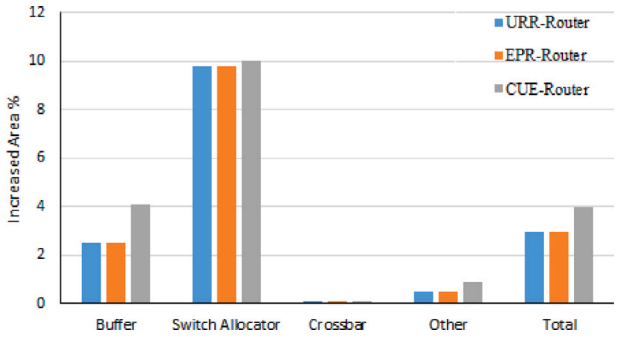

Fig. 21. As compared to Baseline Router, rise in the area of a router for URR-Router, EPR-Router, and CUE-Router.

storage required for a wide flit NoC is roughly equal to around one buffer arrival at the router.

For a p-port router, a $log_2(p)$ bits of the register are required per output port, contributing to p x $log_2(p)$ bits of requirement per router. As a result, CUE-Router has low overhead and minimal impact on the router's critical data path. As previously discussed, each design offers just a few registers, and therefore overhead is minimal.

The Booksim in-detail parameters are evaluated and given to the Cadence Encounter. This is utilized to get an estimate of the area and power of our design. The router model from the simulator served as the foundation for the design. The base router under consideration contains five pipeline levels and 5 input/output ports. It has four VCs per input port, each with an 8-flit buffer size. The setup options and the implementation specifics are modified for the simulation of our design. The injection rate in our evaluations is kept the same as that of the average injection rate for the traces of realistic traffic of PARSEC benchmarks. To match the storage overhead, register buffer space is added. Fig. 20 shows the comparison of the baseline router, URR-Router, EPR-Router, and CUE-Router. The power shown in the results includes both dynamic and leakage power. It should be noted that the register expense is located in the buffer. Thus, the primary cause of the increase in power consumption is due to the buffer and the switch allocator. To summarize, the power consumption of the URR-Router, EPR-Router, and CUE-Router increases by 2.6, 2.6, and 3.7 percent, respectively. The cost of the RF process is added to the switch allocator, and the latter is responsible for most of the consumption of power. Fig. 21 depicts the overhead area of our design. Our design area's costs are generally modest and in line with its energy consumption. The URR-Router, EPR-Router, and CUE-Router have all grown by 3%, 3%, and 4%, respectively.

6. Conclusion and future scope

In on-chip systems, communication latency has emerged as a performance bottleneck. To have an efficient design with optimum SA strategy, router architecture is required to maintain low latency. Recent works related to SA strategies concentrate more on the SA process and

minimize matching efficiency, making it difficult to improve SA performance further. In this paper, we propose a novel router architecture called AS-Router that aims to maximize the AS. First, we introduce URR-Router, which employs the strategy of giving uniform requests the highest priority and allocating EPC requests only if there are no uniform requests in the SA stage. We start designing the SA approach in URR-Router to make sure the impartiality of the SA process. The second instance is EPR-Router, this router reduces the endpoint congestion by giving priority to EPC requests over uniform requests. Finally, we propose a CUE-Router architecture, based on URR-Router and EPR-Router that maintains low latency by optimizing the SA process by switching the priority between uniform requests and EPC requests one after another. CUE-Router aims to send the switch allocator the fewest conflicts of requests by incorporating an additional pipeline stage (RF) ahead of the SA stage for accessing RC information to finalize uniform and EPC requests to be sent for SA. We implement lightweight properties in our design to attain highly desired performance while keeping overheads. When compared to TS-Router, CUE-Router enhances NoCs' overall performance by 8.10% with synthetic traffic, and it enhances performance by an average of 8.75% with application-level traffic. The CUE-Router reduces overhead by 4.1 percent and leakage power consumption by 3.7 percent as compared to the baseline router.

Future work will concentrate on implementing the proposed design with the appropriate modifications needed for 3D NoCs. Additionally, the design can be further optimized in the future to balance out the router frequency overhead. To further reduce NoC latency, the design can be changed in the future to have fewer pipeline stages. The pipeline can be reduced to two stages using some of the NoC routers that have a bypass that is currently in use.

CRediT authorship contribution statement

Monika Katta: Conceptualization, Methodology, Software, Writing – original draft. **T.K. Ramesh:** Data curation. **Juha Plosila:** Data curation.

Declaration of competing interest

• All authors have participated in (a) conception and design, or analysis and interpretation of the data; (b) drafting the article or revising it critically for important intellectual content; and (c) approval of the final version.

• This manuscript has not been submitted to, nor is under review at, another journal or other publishing venue.

• The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript

Acknowledgment

The authors would like to thank the editors and anonymous reviewers for providing insightful suggestions and comments to improve the quality of research paper.

References

[1] T.S. Arulananth, M. Baskar, S.M. Udhaya Sankar, R. Thiagarajan, G. Arul Dalton, Pasupuleti Raja Rajeshwari, Aruru Sai Kumar, A. Suresh, Evaluation of low power consumption network on chip routing architecture, Microprocess. Microsyst. (ISSN: 0141-9331) 82 (2021) 103809, http://dx.doi.org/10.1016/j.micpro.2020.103809.

[2] K. Jin, C. Li, D. Dong, et al., HARE: History-aware adaptive routing algorithm for endpoint congestion in networks-on-chip, Int. J. Parallel Prog. 47 (2019) 433–450, http://dx.doi.org/10.1007/s10766-018-0614-6.

[3] Yuan-Ying Chang, Yoshi Shih-Chieh Huang, Matthew Poremba, Vijaykrishnan Narayanan, Yuan Xie, Candice King, TS-router: On maximizing the quality-of-allocation in the on-chip network, in: Proceedings of the 19th IEEE International Symposium on High Performance Computer Architecture, HPCA'13, 2013, pp. 390–399.

[4] Minseon Ahn, Eun Jung Kim, Pseudo-circuit: Accelerating communication for on-chip interconnection networks, in: Proceedings of the 43rd Annual IEEE/ACM International Symposium on Microarchitecture, Vol. 39, 2010, pp. 9–408.

[5] William J. Dally, Hiromichi Aoki, Deadlock-free adaptive routing in multicomputer networks using virtual channels, IEEE Trans. Parallel Distrib. Syst. 4 (4) (1993) 466–475.

[6] J.V. Escamilla, J. Flich, P.J. Garcia, Head-of-line blocking avoidance in networks-on-chip, in: 2013 IEEE International Symposium on Parallel and Distributed Processing, 796–805. Workshops and Phd Forum, 2013.

[7] B. Wang, Z. Lu, Advance virtual channel reservation. In: Proceedings of the 2019 design, in: Automation and Test in Europe Conference and Exhibition, 2019.

[8] George. Michelogiannakis, Nan. Jiang, Daniel. Becker, William J. Dally, Packet chaining: Efficient single-cycle allocation for on-chip networks, in: Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture, Vol. 8, 2011, pp. 3–94.

[9] Supriya Rao, Supreet Jeloka, Reetuparna Das, David Blaauw, Ronald Dreslinski, Trevor Mudge, Vix: Virtual input crossbar for efficient switch allocation, in: Proceedings of the 51st ACM Annual Design Automation Conference, 2014, pp. 1–6.

[10] Binzhang Fu, John Kim, Footprint: Regulating routing adaptiveness in networks-on-chip, in: Proceedings of the 44th ACM/IEEE Annual International Symposium on Computer Architecture, Vol. 69, ISCA'17, 2017, pp. 1–702.

[11] Sheng Ma, Natalie Enright Jerger, Zhiying Wang, An efficient routing algorithm to support multiple concurrent applications in networks-on-chip, in: In Proceedings of the 38th ACM/IEEE Annual International Symposium on Computer Architecture, Vol. 41, ISCA'11, 2011, pp. 3–424.

[12] Misbah Manzoor, Roohie Naaz Mir, Najeeb-ud-din Hakim, PAAD (partially adaptive and deterministic routing): A deadlock free congestion aware hybrid routing for 2D mesh network-on-chips, Microprocess. Microsyst. (ISSN: 0141-9331) 92 (2022) 104551, http://dx.doi.org/10.1016/j.micpro.2022.104551.

[13] Farah Zulkefli, Phaklen Ehkan, Mohd Warip, Mohd Nazri Bin, ng phing yen, Faiz Fazrul Zakaria, A comparative review of adaptive routing approach for network-on-chip router architecture. 247-254, 2018, http://dx.doi.org/10.1007/978-3-319-59427-9_27.

[14] A. Tajary, E. Tahanian, An adaptive routing algorithm for wireless network on chips, J. Electr. Comput. Eng. Innovations 10 (2) (2022) 487–496.

[15] M. Katta, T.K. Ramesh, J. Plosila, SB-router: A swapped buffer activated low latency network-on-chip router, IEEE Access 9 (2021) 126564–126578, http://dx.doi.org/10.1109/ACCESS.2021.3111294.

[16] Pham Phi-Hung, Jongsun Park, Phuong Mau, Chulwoo Kim, Design and implementation of backtracking wave-pipeline switch to support guaranteed throughput in network-on-chip, IEEE Trans. VLSI Syst. 20 (2012) 270–283, http://dx.doi.org/10.1109/TVLSI.2010.2096520.

[17] Syed Ali Raza Jafri, Hamza Bin Sohail, Mithuna Thottethodi, T.N. Vijaykumar, Apslip: A high-performance adaptive-effort pipelined switch allocator, ECE Tech. Rep. Pap. 451 (2013) 1–14.

[18] N. Jiang, D. Becker, G. Michelogiannakis, W. Dally, Network congestion avoidance through speculative reser vation, in: Proceedings of the 18th IEEE International Symposium on High Performance Computer Architecture (HPCA'12), 2012, pp. 1–12.

[19] S. Ponnan, T.A. Kumar, Hemakumar VS, et al., Congestion aware low power on chip protocols with network on chip with cloud security, J Cloud Comp 11 (2022) 41, http://dx.doi.org/10.1186/s13677-022-00307-4.

[20] Ke Wu, Dezun Dong, Cunlu Li, Shan Huang, Yi Dai, Network congestion avoidance through packet-chaining reservation, in: Proceedings of the 48th ACM International Conference on Parallel Processing, ICPP'19, 2019, pp. 58:1–58:10.

[21] Nychis George, Chris Fallin, Thomas Moscibroda, Onur Mutlu, Srinivasan Seshan, On-chip networks from a networking perspective: Congestion and scalability in many-core interconnects, in: ACM SIGCOMM Computer Communication Review, Vol. 42, 2012, http://dx.doi.org/10.1145/2342356.2342436.

[22] Akbar Reza, Farshad Safaei, A novel heterogeneous congestion criterion for mesh-based networks-on-chip, Microprocess. Microsyst. 84 (2021) 104056, http://dx.doi.org/10.1016/j.micpro.2021.104056.

[23] Nan Jiang, Larry Dennison, William J. Dally, Network endpoint congestion control for fine-grained communication, in: SC '15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2015, pp. 1–12, http://dx.doi.org/10.1145/2807591.2807600, Article No. 35.

[24] Zeng Pengcheng, Yao Shen, Zijun Qiu, Zhun Qiu, Minyi Guo, SRP: A routing protocol for data center networks. 1-6, 2014, http://dx.doi.org/10.1109/APNOMS.2014.6996564.

[25] Jiang Nan, Daniel Becker, George Michelogiannakis, William Dally, Network congestion avoidance through speculative reservation, in: Proceedings - International Symposium on High-Performance Computer Architecture, 2012, pp. 1–12, http://dx.doi.org/10.1109/HPCA.2012.6169047.

[26] Gwangsun Kim, Changhyun Kim, Jiyun Jeong, Mike Parker, John Kim, Contention-based congestion man agement in large-scale networks, in: Proceedings of the 49th Annual IEEE/ACM International Symposium on Microarchi Tecture, 2016, pp. 1–13.

[27] B. Fu, J. Kim, Footprint: Regulating routing adaptiveness in networks-on-chip, in: 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture, ISCA, 2017, pp. 691–702, http://dx.doi.org/10.1145/3079856.3080249.

[28] Escamilla Jose, José Flich, Pedro Garcia, Head-of-line blocking avoidance in networks-on-chip, in: Proceedings - IEEE 27th International Parallel and Distributed Processing Symposium Workshops and PhD Forum, IPDPSW 2013, 2013, pp. 796–805, http://dx.doi.org/10.1109/IPDPSW.2013.214.

[29] Li-Shiuan Peh, William J. Dally, A delay model and speculative architecture for pipelined routers, in: Proceedings of the 7th IEEE International Symposium on High Performance Computer Architecture, HPCA'01, 2001, pp. 255–266.

[30] Farhad Rad, Midia Reshadi, Ahmad Khademzadeh, Flow control and scheduling mechanism to improve network performance in wireless NoC, IET Commun. (2020) http://dx.doi.org/10.1049/iet-com.2019.1033.

[31] William James Dally, Brian Patrick Towles, Principles and Practices of Interconnection Networks, Elsevier, 2004.

[32] J. Kim, J. Balfour, W. Dally, Flattened butterfly topology for on-chip networks, in: Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture, 2007.

[33] I. Pérez, E. Vallejo, M. Moretó, R. Beivide, BST: A BookSim-based toolset to simulate NoCs with single- and multi-hop bypass, in: 2020 IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS, 2020, pp. 47–57, http://dx.doi.org/10.1109/ISPASS48437.2020.00015.

[34] J. Duato, A new theory of deadlock-free adaptive routing in wormhole networks, IEEE Trans. Parallel Distrib. Syst. 12 (1993) 1320–1331.

[35] Christian Bienia, Sanjeev Kumar, Jaswinder Pal Singh, Kai Li, The PARSEC benchmark suite: Characterization and architectural implications, in: Proceedings of the 17th ACM International Conference on Parallel Architectures and Compilation Techniques, 2008, pp. 72–81.

[36] Y. Miura, T.-P. Nakao, N. Fukase, Fault-tolerant adaptive routing algorithm for 2D torus network, Trans. Netw. Commun. 7 (1) (2019) 63, http://dx.doi.org/10.14738/tnc.71.6032.

[37] Mandal Sumit, Anish Krishnakumar, Umit Ogras, Energy-efficient networks-on-chip architectures: Design and run-time optimization, 2021, http://dx.doi.org/10.1007/978-3-030-69131-8_3.

[38] M. Katta, T.K. Ramesh, Maximizing switch allocation matching to reduce latency in network-on-chip, in: IEEE PhD Colloquium on Ethically Driven Innovation & Technology for Society, PhD EDITS, 2022, pp. 1–2.

[39] Katta M. Ramesh T. K., Virtual channel and switch traversal in parallel to improve the latency in network on chip, in: 2nd PhD Colloquium on Ethically Driven Innovation and Technology for Society, PhD EDITS, 2020, pp. 1–2.

[40] Sandeep Darshanala, Vinodhini Manickaraj, N. Murty, Route-on-fly network-on-chip router design with soft-error tolerance, J. Comput. Theor. Nanosci. 17 (2020) 329–333, http://dx.doi.org/10.1166/jctn.2020.8670.

[41] S. Kanagasabapathi, C. Calicut, A routing algorithm and a router architecture for 3D NOC, Comput. Sci. 20 (3) (2019).