WiBS: A Modular and Scalable Wireless Infrastructure in a Cycle-Accurate NoC Simulator

Manjari Saha[1 2], Abhijit Das[3] and John Jose[1]
[1]Indian Institute of Technology Guwahati, Assam, India
[2]Government College of Engineering and Textile Technology, Serampore, West Bengal, India
[3]Univ Rennes, Inria, Lannion, France
{manjari.saha, johnjose}@iitg.ac.in, abhijit.a.das@inria.fr

***Abstract*—Network-on-Chip (NoC) has become a fundamental building block in modern Chip Multi-Processors (CMPs) and plays a significant role in their performance. Nevertheless, ever-increasing core counts coupled with shrinking CMOS technology increases communication latency in Electrical NoC (ENoC). Radio frequency enabled Wireless NoC (WiNoC) is one of the promising solutions proposed to supplement the ENoC and form a wired-wireless hybrid infrastructure. However, except for Noxim, there is no other open-source simulation infrastructure for WiNoC exploration. This work implements a cycle-accurate WiNoC infrastructure in a popular ENoC simulator. The infrastructure is modular and scalable for any network size. Among other important features, it implements a broadcast-enabled one-hop communication and a credit exchange mechanism for radio hubs to avoid packet re-transmission. It is a work in progress and hence supports easy integration of new features for exploration.**

***Index Terms*—Wireless NoC (WiNoC), Simulator, BookSim, Noxim, Chip Multi-Processor (CMP), Packet, Router, Hub.**

I. Introduction

Modern Chip Multi-Processors (CMPs) use Network-on-Chip (NoC) as the communication infrastructure [1]. With the ever-increasing number of cores in the CMPs, on-chip network latency became one of the main factors in performance. The continued shrinking of CMOS technology is exposing the limitations of metallic wire-based Electrical NoC (ENoC) [2]. To maintain performance as well as scalability, supplement or even replacement of ENoC came in the forms of 3D NoC [3], Wireless NoC (WiNoC) [4], Optical NoC (ONoC) [5], etc. WiNoC infrastructure, in particular, received significant attention due to its compatibility with the CMOS technology [6].

Radio frequency enabled WiNoC has natural support for broadcast-based one-hop communication. It is often employed alongside ENoC as a hybrid infrastructure to improve performances of multicast/broadcast [7][8], cache coherence protocols [9][10] and Deep Neural Network (DNN) accelerators [11][12]. Surprisingly, despite such a vast scope of WiNoC exploration, the availability of open-source simulation infrastructure is very limited. Most of the existing works report results either using an in-house simulator or modifying an existing simulator without code availability. While there are a number of detailed and open-sourced simulation infrastructures, Noxim [13] is the only one supporting WiNoC. It has detailed implementation of wireless radio hubs and channels. Nevertheless, the following observations might challenge the implementation and cycle accuracy of Noxim infrastructure:

- **No Broadcast:** Ideally, a sender radio hub broadcasts a packet through a given channel. All the hubs sensing the channel receive the packet and check if it is intended for them. Only the destination hub accepts the packet while others drop it. However, in Noxim, instead of broadcasting, the sender hub pushes the packet into the channel buffer, and the destination hub pops it for use.
- **Packet Re-Transmission:** Ideally, a credit exchange mechanism is in place to know the buffer availability in the destination hub before sending a packet. Absence of this mechanism forces a packet re-transmission when the buffers are full in the destination hub. As hubs in Noxim use the channel buffer to send and receive packets, a re-transmission is initiated when the channel buffer is full.

Interestingly, implementing a true broadcast infrastructure will automatically facilitate the implementation of a credit exchange mechanism. For example, the buffer availability of hubs can be broadcast in a single hop to avoid packet re-transmission. Motivated by these observations, this work presents an open-source wireless infrastructure in a cycle-accurate NoC simulator. More precisely, this work enables WiNoC support in BookSim [14], as it has a modular infrastructure, and has detailed implementation and rich features for key NoC building blocks. The developed infrastructure is named *WiBS*[1], and has the following important features:

1) Alongside the traditional router-to-router communication, WiBS provides a cycle-accurate hub-to-hub communication using broadcast. It also offers multiple channel access schemes to avoid transmission collision.
2) WiBS implements buffers, allocators/arbiters and a credit exchange mechanism for the hubs to manage flow control and avoid expensive packet re-transmission.
3) All the topologies, traffic patterns and routing algorithms of BookSim are also supported in WiBS. The infrastructure is configurable and scalable for any network size. WiBS is written using the standard C++ programming paradigm and can be easily extended for exploration.

[1]https://github.com/manjari-saha/wibs

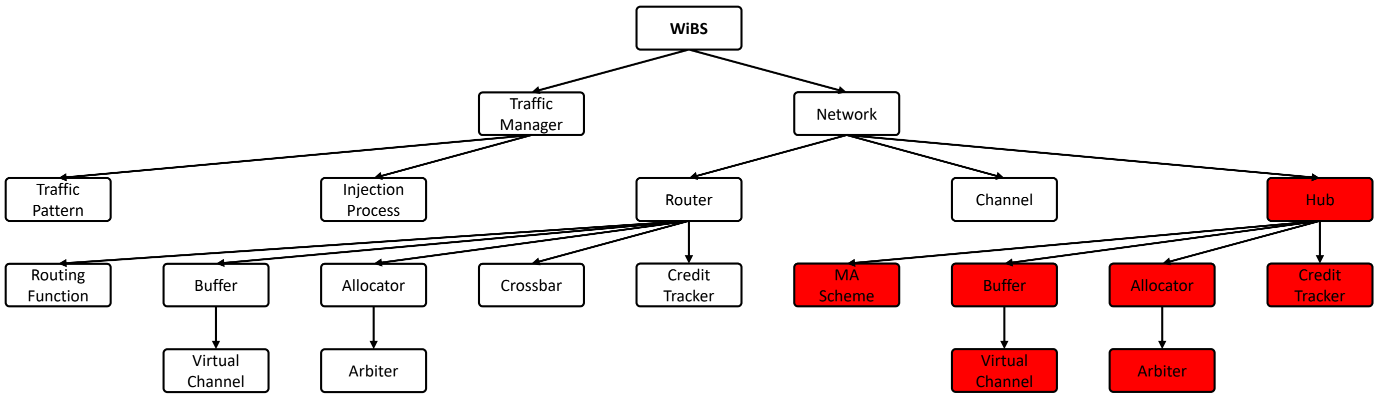

Figure 1: Building blocks of the WiBS infrastructure.

Table 1: Popular NoC simulators

Simulator	Topology Flexibility	Open-Source	WiNoC
NIRGAM [15]	✗	✓	✗
GARNET [16]	✗	✓	✗
DARSIM [17]	✗	✓	✗
ATLAS [18]	✗	✓	✗
TOPAZ [19]	✗	✗	✗
FlexNoC [20]	✓	✗	–
BookSim [14]	✓	✓	✗
Noxim [13]	✓	✓	✓
WiBS	✓	✓	✓

II. Related Works

Since the early 2000s, when NoC was presented as a communication infrastructure, quite a few simulators have been proposed/used for the fast exploration and evaluation of research ideas. Some of them are listed in Table 1 and compared with respect to their topology flexibility, code availability and WiNoC support. For example, GARNET [16] became very popular after it was included in the GEMS [21] (now gem5 [22]) full-system simulator. However, GARNET lacks WiNoC support. Similarly, FlexNoC [20] is very popular in the industry, but understandably the infrastructure is not open-sourced. In terms of flexibility, modular infrastructure, detailed implementation and rich features of key NoC building blocks, BookSim has an edge. Nevertheless, it also lacks WiNoC support. Noxim is the only infrastructure with open-sourced WiNoC support and a lot of flexibility. The proposed WiBS simulation infrastructure attempts the best of both worlds. It has the features of BookSim as well as the ability of Noxim.

III. BookSim Infrastructure

As the proposed WiBS infrastructure is developed on top of the existing BookSim simulator, some familiarity with its modules is necessary to understand the wireless extension. Hence, this section briefly describes the major building blocks of BookSim. In Figure 1, except the ones shown in red colour, all the other blocks are part of the BookSim infrastructure.

A. *Traffic Manager*

It is the wrapper around the on-chip network being simulated. It creates, injects and ejects packets according to the user-specified configuration, like packet size, traffic pattern, injection rate, etc. It also collects the simulation statistics.

1) Traffic Pattern: It specifies a synthetic traffic scenario for the network. It also defines the relationship between the source and destination of a packet. BookSim supports many popular traffic patterns like, uniform, transpose, tornado, etc.

2) Injection Process: Its primary purpose is to specify the rate at which new packets should be injected into the network.

B. *Network*

Based on the specified topology, it connects different nodes of the network through routers. All communication between neighbouring routers takes place through explicit channels.

1) Router: It implements the microarchitecture of an input-buffered router. It has the following essential components:

- Routing Function: It dictates the path of a packet from its source to the destination. It should be deadlock-free.
- Buffer: It is divided into multiple Virtual Channels (VCs) and stores incoming packets while they wait for routing.
- Allocator: It is managed by two separate arbiters for VCs and switches. It chooses a winner for every output port.
- Crossbar: It connects the input and output port of all the winning packets to reach their downstream routers.
- Credit Tracker: It communicates the VC availability to the upstream routers and facilitates flow control decisions.

2) Channel: There are two dedicated channels between neighbouring routers, one for data and the other for credit transfer. The channel width determines the size of a *flit*; the smallest unit of transfer. One packet can have multiple flits.

BookSim has the flexibility to configure almost all of these parameters using command-line arguments, without really modifying the source code. Hence, it was chosen for WiBS.

IV. WiBS Infrastructure

Figure 1 shows all the building blocks of the proposed WiBS, where the ones shown in red colours are responsible for the wireless infrastructure. The key features that define an

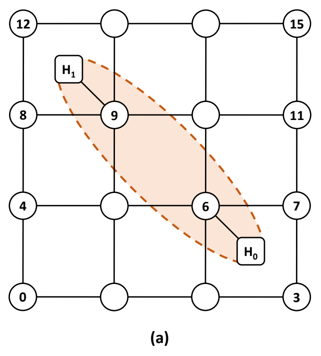
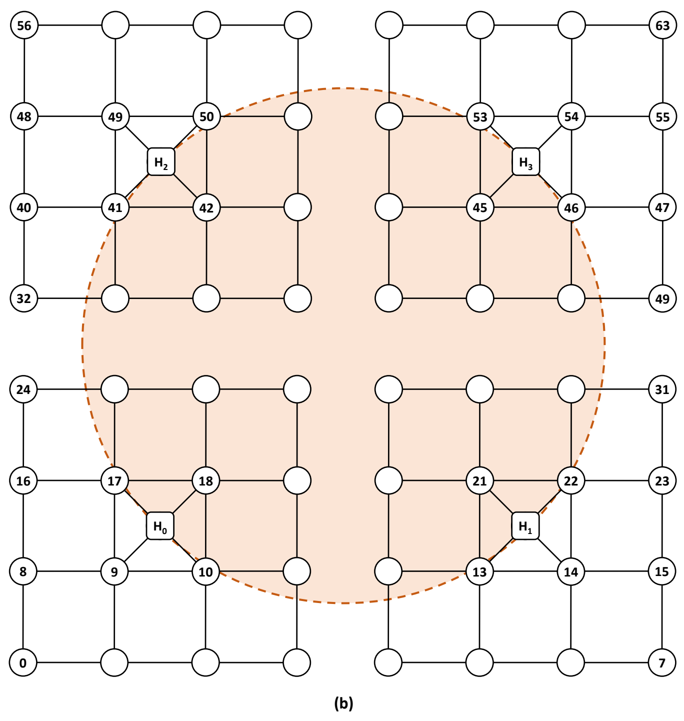
Figure 2: Sample topologies with routers and hubs.

NoC communication infrastructure are: topology, routing and arbitration, flow control, and router and hub microarchitecture [23]. This section describes these features for the WiBS.

A. Topology

It describes how routers and hubs are connected via channels to form the network. *Hop* refers to the direct connection between a pair of routers or hubs. Topology also determines path diversity; alternate routes between a source and destination. Ideally, a path with the minimum number of hops is preferred for network performance. Figure 2 shows two sample topologies available with WiBS, a 4×4 and an 8×8 2D Mesh. These are examples of ENoC-WiNoC hybrid topologies, where a packet can travel in wired and/or wireless channels. All the existing topologies available with BookSim can also be used in WiBS. Additionally, any new or irregular topology can be easily configured in WiBS. As shown in Figure 2(a) and (b), a hub can be connected with any number of routers; two in the 4×4 and four in the 8×8 2D Mesh topologies. Due

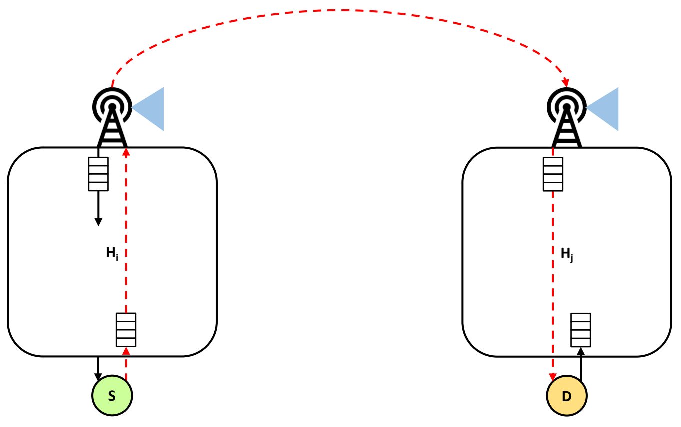

Figure 3: Packet traversal in a hybrid path.

to the broadcast support (to be discussed in Section IV-D) available in the WiBS, it is also possible to set up a one-hop communication between specific source and destination pairs. For example, the hubs from the 8×8 2D Mesh-based chiplet-style topology in Figure 2(b) can be represented as follows:

$$
\begin{aligned}
H_0 &= \{9, 10, 17, 18\}\\
H_1 &= \{13, 14, 21, 22\}\\
H_2 &= \{41, 42, 49, 50\}\\
H_3 &= \{45, 46, 53, 54\}
\end{aligned}
$$

Here, sources from an H_i can reach their destination in any of the H_js in just one hop through the wireless (shaded) channel.

B. Routing and Arbitration

After determining the topology, a routing algorithm dictates the path packets should take from the source to destination. In hybrid topologies, the path of a packet could be fully-wired or wired-wireless. The following interconnections are possible:

- Router-to-Router (RtoR)
- Router-to-Hub (RtoH)
- Hub-to-Router (HtoR), and
- Hub-to-Hub (HtoH)

RtoR, RtoH, and HtoR are wired interconnections, and HtoH is wireless. While RtoR is already available in BookSim, the proposed WiBS implements the other three interconnections. Figure 3 shows the packet traversal in a hybrid path involving RtoH, HtoH and then HtoR interconnections. The typical actions performed by a routing algorithm in the routers and hubs are presented in Figure 4. When a router receives a new packet, the routing algorithm checks if the destination is within the zone, i.e., reachable through wired channels. In that case, the packet uses the wired path involving only the RtoR interconnections. Otherwise, the packet is forwarded towards the nearest hub to use the wireless channel. When a hub receives a new packet, the routing algorithm checks if it is from another hub, i.e., came through the wireless channel. In that case, the packet is either forwarded or dropped based on

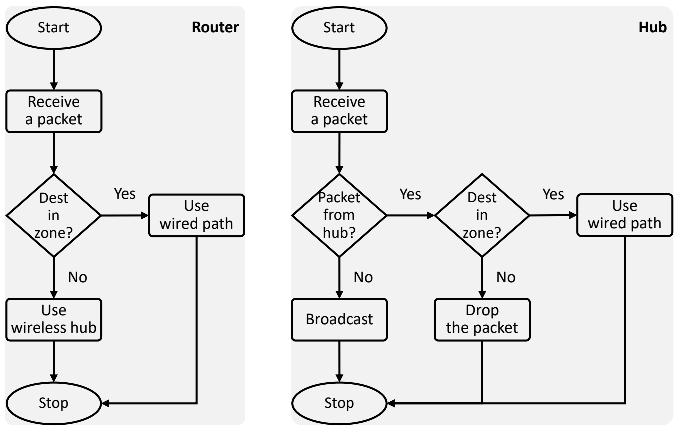

Figure 4: Actions by routing algorithm in routers and hubs.

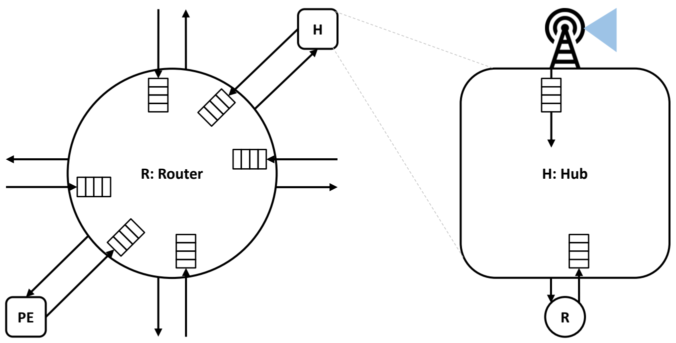

Figure 5: Conceptual view of a router and a hub

the zone of its destination. Otherwise, the packet came from a router and hence is broadcast through the wireless channel.

RtoR, RtoH and HtoR interconnections have dedicated wired channels, hence no transmission collision. However, it is not the case with HtoH, which uses a wireless channel. All the hubs share a single wireless channel, and without an arbitration policy in place, collision is inevitable when multiple hubs attempt to access the channel simultaneously. The solution comes in the form of multiple channel access schemes, like:

- Frequency Division Multiple Access (FDMA): Based on the radio frequency, the channel is divided into multiple sub-channels, one for each hub. FDMA is not scalable as the number of sub-channels has an upper-bound [24].
- Time Division Multiple Access (TDMA): Based on a time slot, hubs access the entire channel one by one. TDMA needs a time counter for channel-hub scheduling.
- Code Division Multiple Access (CDMA): Using different codes for their transmission, all the hubs can access the channel simultaneously. CDMA is complex in nature, requires encoder-decoder circuits resulting in overheads.
- Token Ring (TR): All the hubs are connected in a virtual ring and pass a token. At any point, only the hub holding the token can access the channel. TR is inefficient.

WiBS works with all the routing algorithms available in BookSim. It can also support any other deterministic, oblivious or adaptive routing algorithm. For accessing the wireless channel, WiBS has implementations of TDMA and TR, but its modular design allows easy implementation of others.

C. Flow Control

It manages network transfer and decides when a packet can take the next channel on its path, and when it has to wait at some router (hub). While topology and routing algorithm decides the theoretical latency and throughput, flow control determines how close to that expectation a network can operate. Modern NoC infrastructures use a VC-based flow control mechanism, and the same is inherited in WiBS from BookSim. Flow control requires information about VC availability in the neighbouring routers (hubs), and a credit exchange mechanism takes care of it. Such a mechanism is already in place for the routers in BookSim (refer Section III-B1). For the newly added hubs, WiBS implements a credit exchange mechanism exploiting the broadcast channel. Before sending a packet through the wireless channel, the sender hub requests the VC availability in the destination hub(s) to avoid packet re-transmission. The VC credit is broadcast and reaches the sender in one system clock cycle. There is a provision in WiBS to use a dedicated low data-rate channel for credit exchange.

D. Router and Hub Microarchitecture

Most of the features presented so far are implemented by the microarchitectures of a router and a hub. Figure 5 shows their conceptual views. Booksim already has the implementation of an input-buffered, pipelined router with the following stages:

- Queuing/Routing
- VC Allocation
- Switch Allocation
- Crossbar Traversal

As shown in Figure 5, packets can enter a router from its neighbours (north, east, south, west), the Processing Element (PE) and the connected hub. Packets get queued in the input port VCs and take part in routing and arbitration decisions. The winning packets leave the router through a crossbar switch. BookSim allows router pipeline stages to run in parallel to get a two-stage router, one for the routing and arbitration and the second for switch traversal. The proposed WiBS implements the hub microarchitecture with the following pipeline stages:

- Queuing/Token Receiving
- VC Allocation
- Channel Allocation
- Broadcast

WiBS allows a hub to be connected with any number of routers. As shown in Figure 5, packets can enter a hub from the connected routers or through the wireless transceiver. Packets entering from the connected routers want to use the wireless channel. For the sake of easy explanation, the TR channel access scheme (refer Section IV-B) is considered. To be able to access the channel, the hub tries to get hold of the token. In the meantime, incoming packets get queued in the corresponding VCs. Then hub allocates VCs in the downstream hubs. By the time hub receives the token, packets from multiple connected

Table 2: Simulation configuration

Network Size (Cores)	256 (16×16)
Topology	2D Mesh
Routing	XY DOR, West First
Number of Hubs	0, 4, 8
Wireless Channel Access	Token Ring (TR)
Wireless Data Rate (Gbps)	32
Packet Size (flits)	16
Flit Size (bits)	32
Number of VCs in Router	4
Depth of Router VCs (flits)	4
Number of VCs in Hub	4
Depth of Hub VCs (flits)	16
Simulation Time	1000
Traffic Pattern	Uniform Random

routers could be waiting in the VCs for the channel. Hence, WiBS implements a channel allocation policy to decide on a winner. The current implementation has round-robin and ageing, but can be easily modified for others. Once the winner is decided, the chosen packet is duplicated for one less than the number of hubs. These duplicate packets are sent across the network for every hub to achieve broadcast. Upon receiving a packet through the transceiver, a hub examines its header for the destination. If the packet is destined for a router reachable from the hub, it is ejected through one of the connected routers. Otherwise, the packet is simply dropped. Barring the time to receive the token, WiBS provides a 2-stage hub, one for the VC and channel allocation and the second for the broadcast.

V. Evaluation

This section presents some interesting results obtained using the proposed WiBS infrastructure. Table 2 presents the simulation configuration, where some of the parameters added in BookSim for WiBS are shown in italics. The number of system clock cycles required for a packet to reach from source to its destination is called packet latency. It is one of the most important performance metrics to evaluate any NoC communication infrastructure. Hence, all the results presented in this section are with respect to average packet latency. In BookSim, the average packet latency is calculated as the average duration between the injection and ejection time of packets. However, WiBS implements broadcast using duplicate packets, which get discarded except in the destination hub. Those packets are not accounted for in the calculation of average latency. The nomenclature of a simulation configuration is given by *A-B-C*, where *A* is the simulator (WiBS/Noxim), *B* is the routing algorithm (XY DOR/West First), and *C* is the number of hubs (0/4/8). For example, *WiBS-WF-4* means WIBS is run with West First routing algorithm with 4 hubs, keeping all the other parameters of Table 2 unchanged. 0 hubs mean a wired NoC.

A. Wired vs Hybrid

Figure 6 shows a comparison between the wired NoC previously available with BookSim and the hybrid NoC implemented in WiBS. With *Uniform Random* traffic pattern,

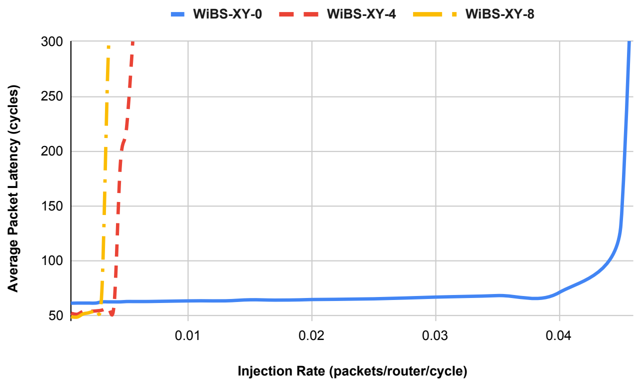

Figure 6: Wired vs wireless

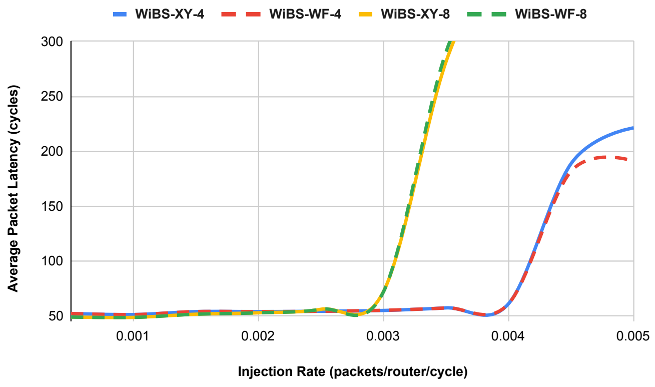

Figure 7: XY DOR vs west first

the packets get evenly distributed over the entire network. It results in roughly 75% of the packets trying to reach their destination through the wireless channel in *WiBS-XY-4* and even more in *WiBS-XY-8*. Hence, while the packets enjoy better performance at low injection rate, the network saturates early due to congestion in the hubs. In general, the number of hubs is inversely proportional to the time to network saturation.

B. XY DOR vs West First

XY Dimension Order Routing (DOR) is a deterministic routing algorithm that dictates the packets to take a path along the *X* dimension followed by *Y* dimension to reach their destination. West First is an adaptive routing algorithm that dictates the packets to take a path along the *West* direction followed by other directions to reach their destination. In general, *West First* performs better due to its path diversity.

WiBS supports both of these algorithms, and Figure 7 shows their performance. Contrary to the expectation, both algorithms exhibit similar behaviour at low injection rate. In hybrid NoC, a packet takes the wired path (*XY DOR*/*West First*) to reach the nearest hub, then takes the wireless path to reach the destination hub and finally takes the wired path (*XY DOR*/*West First*) again to reach its destination router. The distance from a router to the nearest hub is very short, hence

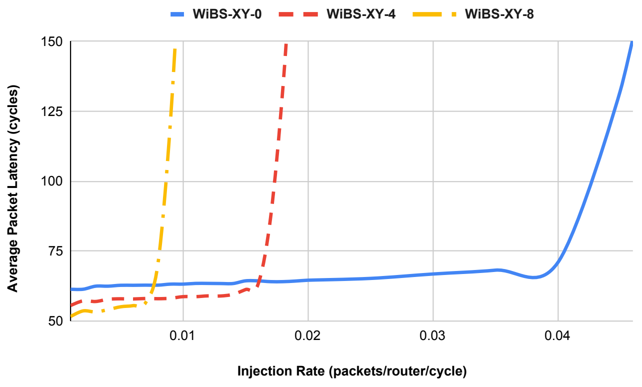

Figure 8: Wireless utilisation tuning

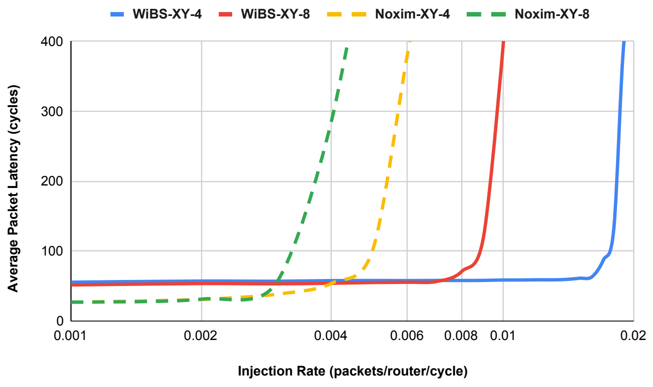

Figure 9: Comparison with Noxim

very little path diversity. As discussed in Section V-A, since most packets try to use the wireless channel, *West First* can not do much. Interestingly, at high injection rate, when the neighbouring routers and hubs are congested, West First shows some promise. Observe the gap between the solid blue (*WiBS-XY-4*) and dotted red (*WiBS-WF-4*) lines towards the end.

C. Wireless Utilisation Tuning

To prevent the network from early saturation (as observed in Figure 6), WiBS implements a wireless utilisation tuner. For example, it is possible to specify that the wireless channel will be used only if the wired path of a packet from source to destination exceeds a certain hop-count threshold. Figure 8 shows a result with the wireless utilisation tuner set to a hop-count threshold of 8. For the same configuration, now the network saturates much later. This can be used for exploration.

D. Comparison with Noxim

And finally, Figure 9 compares a simulation run from the proposed WiBS infrastructure with state-of-the-art Noxim simulator. In all fairness, it is not possible to compare them exactly point by point, but the objective here is to understand their nature. The trend in their results is similar; better performance at low injection rate and saturation at high injection rate. Noxim performs better than WiBS at low injection rate, while WiBS saturates later at high injection rate. This could be due to the way the routing algorithms (e.g., broadcast) are implemented or the difference in wireless channel utilisation.

VI. Discussion and Future Work

WiBS is an attempt to have an open-source WiNoC simulation infrastructure for research exploration. While it has implementations for major aspects of the performance, it does not support overhead (e.g., area, power, etc.) estimation yet. However, it is a work in progress and has support for easy integration of new features. The immediate future work includes multi-channel modelling, where the challenge is to manage token passing for each channel. The routing algorithm also requires modification as two communicating hubs might not have a common channel anymore. After the multi-channel implementation, FDMA and CDMA access schemes are next.

References

[1] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Inteconnection Networks," in *DAC*, 2001.

[2] R. Ho *et al.*, "The Future of Wires," *Proceedings of the IEEE*, 2001.

[3] A. B. Ahmed *et al.*, "Architecture and Design of Efficient 3D Network-on-Chip (3D NoC) for Custom Multicore SoC," in *BWCCA*, 2010.

[4] S. Deb *et al.*, "Wireless NoC as Interconnection Backbone for Multicore Chips: Promises and Challenges," *IEEE JETCAS*, 2012.

[5] Y. H. Kao and H. J. Chao, "BLOCON: A Bufferless Photonic Clos Network-on-Chip Architecture," in *NOCS*, 2011.

[6] A. Karkar *et al.*, "A Survey of Emerging Interconnects for On-Chip Efficient Multicast and Broadcast in Many-Cores," *IEEE MCAS*, 2016.

[7] K. Duraisamy *et al.*, "Multicast-Aware High-Performance Wireless Network-on-Chip Architectures," *IEEE TVLSI*, 2016.

[8] S. Abadal *et al.*, "Scalability of Broadcast Performance in Wireless Network-on-Chip," *IEEE TPDS*, 2016.

[9] A. Franques *et al.*, "WiDir: A Wireless-Enabled Directory Cache Coherence Protocol," in *HPCA*, 2021.

[10] S. H. Gade and S. Deb, "A Novel Hybrid Cache Coherence with Global Snooping for Many-Core Architectures," *ACM TODAES*, 2021.

[11] R. Guirado *et al.*, "Dataflow-Architecture Co-Design for 2.5D DNN Accelerators using Wireless Network-on-Package," in *ASP-DAC*, 2021.

[12] S. Liu *et al.*, "Exploiting Wireless Technology for Energy-Efficient Accelerators with Multiple Dataflows and Precision," *IEEE TCSI*, 2022.

[13] V. Catania *et al.*, "Noxim: An Open, Extensible and Cycle-Accurate Network on Chip Simulator," in *ASAP*, 2015.

[14] N. Jiang *et al.*, "A Detailed and Flexible Cycle-Accurate Network-on-Chip Simulator," in *ISPASS*, 2013, pp. 86–96.

[15] L. Jain *et al.*, "NIRGAM: A Simulator for NoC Interconnect Routing and Application Modeling," in *DATE*, 2007.

[16] N. Agarwal *et al.*, "GARNET: A Detailed On-Chip Network Model inside a Full-System Simulator," in *ISPASS*, 2009, pp. 33–42.

[17] M. Lis *et al.*, "DARSIM: A Parallel Cycle-Level NoC Simulator," in *MoBS*, 2010.

[18] A. Mello *et al.*, "ATLAS - An Environment for NoC Generation and Evaluation," in *DATE*, 2011.

[19] P. Abad *et al.*, "TOPAZ: An Open-Source Interconnection Network Simulator for Chip Multiprocessors and Supercomputers," in *NOCS*, 2012.

[20] (2009) Arteris FlexNoC. [Online]. Available: https://www.arteris.com/flexnoc

[21] M. M. K. Martin *et al.*, "Multifacet's General Execution-Driven Multiprocessor Simulator (GEMS) Toolset," *ACM SIGARCH CAN*, 2005.

[22] N. Binkert *et al.*, "The gem5 Simulator," *ACM SIGARCH CAN*, 2011.

[23] A. Das, "Designing Data-Aware Network-on-Chip for Performance," Ph.D. dissertation, Indian Institute of Technology Guwahati, 2021.

[24] K. Duraisamy *et al.*, "Enhancing Performance of Wireless NoCs with Distributed MAC Protocols," in *ISQED*, 2015.