Broadcast Mechanism Based on Hybrid Wireless/Wired NoC for Efficient Barrier Synchronization in Parallel Computing

Hemanta Kumar Mondal[*,†]
[†] National Institute of Technology Durgapur
hemanta.mondal@ece.nitdgp.ac.in

Navonil Chatterjee, Rodrigo Cataldo, Jean-Philippe Diguet
[*] Lab-STICC, CNRS, Université Bretagne Sud
{chatterjee.navonil, rodrigo-cadore.cataldo, jean-philippe.diguet}@univ-ubs.fr

Abstract— Parallel computing is essential to achieve the manycore architecture performance potential, since it utilizes the parallel nature provided by the hardware for its computing. These applications will inevitably have to synchronize its parallel execution: for instance, broadcast operations for barrier synchronization. Conventional network-on-chip architectures for broadcast operations limit the performance as the synchronization is affected significantly due to the critical path communications that increase the network latency and degrade the performance drastically. A Wireless network-on-chip offers a promising solution to reduce the critical path communication bottlenecks of such conventional architectures by providing hardware broadcast support. We propose efficient barrier synchronization support using hybrid wireless/wired NoC to reduce the cost of broadcast operations. The proposed architecture reduces the barrier synchronization cost up to 42.79% regarding network latency and saves up to 42.65% communication energy consumption for a subset of applications from the PARSEC benchmark.

Index Terms—**Broadcast operation, wireless/wired network-on-chip, parallel computing, barrier synchronization**

I. Introduction

The High-Performance Computing (HPC) platforms, composed of on-chip manycore systems, provide the necessary support for the computational requirements of several types of scientific research, cloud computing, and data center applications. A manycore system requires fast communication infrastructures to fulfill the inter-core communication requirements. The parallel applications, which rely heavily in inter-core communications, also require broadcast communications. Conventional implementations for packed-based Network-on-Chip (NoC) typically lack real hardware support for broadcast, hence it introduces communication bottleneck and degrades system performance. In conventional NoC architectures support broadcast operations in the form of multiple unicast transmissions, which results in a significant increase of the latency and energy consumption of NoC. Recently, NoC architectures have been explored to address the challenges during barrier synchronization [1][2], but these proposed solutions suffer from scalability issues. Emerging interconnect architectures aim to reduce performance by limiting multi-hop communication in conventional wired NoCs. Three emerging interconnects (three-dimensional (3D), photonic, and RF/wireless NoCs [3][4][5]) were introduced to address the long range multi-hop communication bottleneck. Inductive/capacitive-coupled 3D integration technology [6] is another alternative to wireless interconnects but it produces electromagnetic interference through unwanted coupling. The photonic NoC achieves a higher bandwidth than Wireless Network-on-Chip (WiNoC). However, the high bandwidth is not required for parallel applications during broadcast operation for synchronization. Moreover, photonic NoC does not immediately provide any broadcast capability, whereas it is naturally available with RF interconnects. WiNoCs using mm-wave interconnects have emerged as one of the promising solutions for scalable, energy-efficient NoC fabrics with CMOS compatible technology. Beyond broadcast capabilities, WiNoCs provide an end-to-end single-hop communication and so can be an efficient solution for improving the performance of parallel applications significantly [7][8].

WiNoC is a promising solution for the implementation of broadcast communications [9]. However, there are restrictions to its implementation: firstly, a dedicated radio channel is not a feasible solution since it is not realistic to plug a wireless interface to each core. Secondly, the usual token-passing protocol introduce a waiting time that can degrade the single-hop advantage of wireless links. Therefore, a hybrid solution is a possible compromise as long as it is simple enough to provide gains without a prohibitive complexity overhead. We propose a new mechanism to improve the overall system performance by providing efficient barrier synchronization that reduces the cost of broadcast operations significantly. One of the main contributions of this work is that both wired and wireless links are utilized simultaneously for unicast and broadcast operations. The choice is made at runtime according to performance-driven decision. We also propose a lightweight Dynamic Broadcast Mode Controller (DBMC) to control the unicast and broadcast packets at the network level. Hence, during the synchronization operation, the network can use real broadcast messages on a barrier by using a wireless link.

The major contributions of this work are as follows: (i) Proposal of a broadcast-aware WiNoC architecture to improve the performance by simultaneous use of both wired and wireless links; (ii) Design and implementation of lightweight DBMC to control the unicast and broadcast traffic; (iii) Validation under the PARSEC benchmark [10] and comparison with existing architectures. To evaluate the wireless interconnect-based architecture, we have modified the existing Noxim simulator [11] to handle broadcast traffic; (iv) Investigation of the power dissipation at Wireless Interfaces (WIs) To the best of our knowledge, this is the first work that shows a substantial amount of power saving at the WI under a parallel application workload.

The remainder of this paper is organized as follows. Section 2 describes related work. The proposed architecture including barrier synchronization, dynamic broadcast mode controller and communication protocol is discussed in Section 3. Section

4 presents the results of the performance evaluation, and Section 5 concludes this work.

II. Literature Review

Many research works have investigated the barrier synchronization in parallel computing using packed-based NoC architecture. Software-based barrier synchronization implementations increase the latency significantly with system size due to serialization of the barrier operation on such architectures. Hardware-based barrier synchronization is implemented supporting multiple thread groups to overcome this limitation, where each group has its barrier. The proposed method reduces the latency for serialization compared to the traditional planner wired. However, the transmission-line based approach faces several challenges in a manycore system [1]. The transmission-line needs to spread the entire chip area and requires excessive branching points to communicate with every core. Besides, it is not an efficient solution due to cross-talk, inter-channel interference for long transmission lines and large fan-out and power dissipation for large-sized systems. Hardware-based barrier synchronization is implemented using a G-line based network allowing efficient signaling of barrier arrival and release [12]. A hybrid tree-based all-to-all barrier for NoC-based manycore system is explored in [2] to improve the performance by avoiding the off-centered barrier core. However, they are all based on multiple unicast packets, which is not efficient in parallel computing. The latency is also destination-dependent, so the message delivery time is unbalanced. The CMOS-compatible WiNoC architecture can play an essential role in providing an efficient broadcast mechanism for these applications. Recently, OrthoNoC [9] has been introduced using wireless links for the broadcast operations. This proposed scheme significantly improves packet latency overall and demonstrate the interest of NoCs with efficient broadcast mechanisms. However in [9], the total number of wireless hubs is 64 for a 64-core system. Only one wireless pair can communicate at a time through the radio broadcast plan based on a collision detection protocol. This approach is costly in terms of radio hubs, which means large area and static power consumption. It is also oversized when the ratio of broadcast communications is limited as observed in usual parallel computing applications. So we introduce a hybrid solution that combines true wireless and wired broadcast mechanisms. Compared with a fully wireless approach, the problem of availability must be solve by means of hybrid routing and rerouting solutions that we developed in this work.

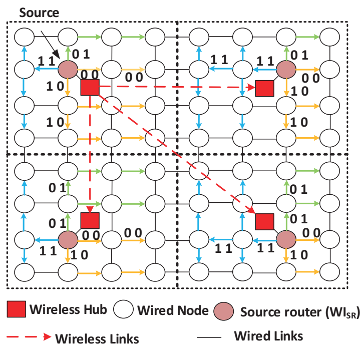


Fig. 1: Broadcast enabled WiNoC architecture.

III. Proposed Architecture

A conventional NoC architecture consists of base routers (BRs) attached to message source/sink components; with all BRs interconnected by wires in a specific topology. A WiNoC is a group of clusters with one WI for each. Our proposed architecture is scalable, in this paper, we consider a single scenario for evaluation of four equally-sized clusters in our 64-core architecture. Our WiNoC architecture places the WI at the center of each cluster, providing inter-cluster wireless links with the minimal median distance to improve the overall system performance. Figure 1 shows the WiNoC topology with BRs and wireless hubs. For example, one WIs is a source hub, and the others are destination hubs during the broadcast operation. Only a single WI can communicate during the broadcast operation. Usually, in a conventional NoC, each core integrates a dedicated synchronization controller to control the barrier messages [8]. The barrier control registers of the controller handle the barrier arrival and release messages among involved cores. They record the number of cores arriving at the barrier and activate the barrier releasing flag to all involved cores only if all cores have reached the barrier. Multiple types of message delivery (e.g., all-to-all, master-slave, butterfly, and tree) can be implemented. However, the network performance is affected dramatically using these conventional approaches due to longest core-to-core critical-path, and the implementation of broadcast messages.

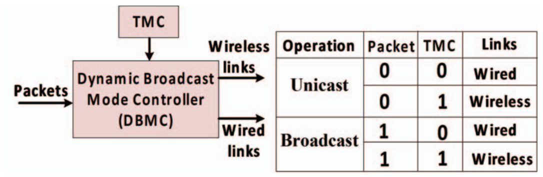
TABLE I: Synchronization events during executions of Bodytrack and Streamcluster on NoC-based multiprocessors.

Application	Type	Events per number of threads		
		16-core	32-core	64-core
Bodytrack	Barrier	2,112	4,288	17,788
	Condition	447	750	4,264
	Mutex	9,000	10,472	37,818
Streamcluster	Barrier	208,064	364,480	728,960
	Condition	381	802	1,274
	Mutex	510	1,054	2,142

A. *Barrier Synchronization*

The POSIX Thread (PThread) standard is one of the most well-known interfaces used for parallel computing. PThread offers multiple procedures for developers to synchronize data among application threads or the threads themselves. In this work, we chose to improve the barrier synchronization procedure, where the architecture explored in this work can achieve significant benefits, as it releases threads by sending the same message to multiple destinations. Barriers are responsible for synchronizing threads to a user-specified location in the application. When all participating threads reach the specified location, they can continue to execute; otherwise, they are blocked. Therefore, we have changed the releasing procedure to generate a single broadcast message instead of multiple unicast ones. The use of synchronization procedures provided by PThread is contingent on the design of the application. The PARSEC benchmark [10] is a collection of applications intended for next-generation shared-memory architectures that employs the PThread standard. From the twelve applications available on PARSEC, we selected two of them as a representative of the PARSEC workload: Bodytrack, for a small number of broadcast messages, and Streamcluster, for a large number of broadcast messages. Bodytrack is a computer-vision application that tracks a 3D pose of a mark-less body.

Streamcluster is a data-mining application that solves the online clustering problem for a stream of input points. Both applications use multiple synchronization procedures; however, Table I shows the barrier procedure has most of the requests. Therefore, we can exploit the broadcast-enabled network for a 64-core system. When considering all the data traffic generated by the application, Streamcluster requires more than 5% of the overall traffic for broadcast messages, as shown in [13]. The performance is affected by these messages dramatically as it increases the congestion and provides poor quality of service. Besides, it increases the power dissipation due to the retransmission of the same packet. CMOS-compatible wireless emerging interconnect offers many advantages to overcome these drawbacks of conventional NoCs. Hence, we consider the WiNoC architecture to reduce number of hops required for communication and to implement broadcast through simultaneous receptions. Once the packet arrives at the WI, the packet is transmitted to neighboring cores by tree-based load-balanced paths using the protocol described in Section 3.3. Experimentally, we found that the proposed architecture reduces the network latency significantly during the broadcasting phase for barrier synchronization.



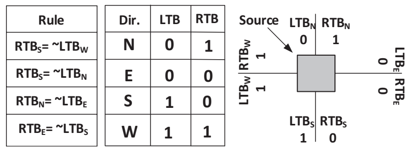
Operation	Packet	TMC	Links
Unicast	0	0	Wired
	0	1	Wireless
Broadcast	1	0	Wired
	1	1	Wireless

Fig. 2: Broadcast mode controller and operations.

B. Dynamic Broadcast Mode Controller

The primary components of on-chip wireless communication infrastructure are the antenna and transceiver. In this work, we employ a zigzag metal antenna adopted from [14], and a non-coherent On-Off Keying (OOK) modulation scheme for the WiNoC transceiver [5]. A unique wireless channel is shared among all WIs, and the token passing mechanism with round robin arbitration is used to provide access to the shared wireless medium. Hence, only a single broadcast operation is possible at the same time using wireless links. In [5], the primary components of the transmitter (TX) circuitry are an up-conversion mixer and a power amplifier (PA). The receiver (RX) consisting of a low noise amplifier (LNA), a down-conversion mixer and a baseband amplifier. For parallel applications, multiple simultaneous broadcast and unicast operations must take place to improve the overall system performance.

Performance can be further improved over the works [2][9] by efficiently combining both wired and wireless infrastructures for broadcast as well as unicast operations. The main reason is that the access to WI is not guaranteed. The wireless channel can be used for on-going communication, or the WI does not have the token in case of a standard token-passing protocol. In this case, the latency gains of WiNoC can be lost. We implement the DBMC, as shown in Figure 2, for efficient broadcast operations. DBMC is associated with each WI source router (WISR) that decides the route of a packet, which is connected with each WI as shown in Figure 1. Consequently, the total number of DBMC is 4 in our experimental 64 core setup and the overhead due to these circuits is shown by results of the RTL implementation of DBMC in Section 4.5. Note that we additionally consider the Hop count to decide the use of WI for unicast packets, this is detailed in the next section (Fig. 4,5). DBMC sends the request to the WI arbiter based on the status of the packets and the Token Management Controller (TMC) for the output port, which is mentioned in the Table of Figure 2. Here, broadcast and unicast operations are represented by the values 1 and 0, respectively. The availability of the token, which is represented by 1, otherwise 0, is collected from TMC [15], which is implemented with a simple 2×2 logic table; thus, the cost is negligible.



Rule
$RTB_S = \sim LTB_W$
$RTB_S = \sim LTB_N$
$RTB_N = \sim LTB_E$
$RTB_E = \sim LTB_S$

Dir.	LTB	RTB
N	0	1
E	0	0
S	1	0
W	1	1

Fig. 3: Cluster-wise Whirl method for wired broadcast

C. Communication Protocol

In this section, we describe various types of broadcast mechanisms such as wired, wireless and hybrid methods for the barrier synchronization.

1) Wired-based Broadcast: The traditional implementation of a barrier scheme selects a master node for collecting barrier arriving messages from all other cores and for broadcasting the barrier release messages to them. Most of packed-based NoCs use unicast communications to implement this protocol, which is an inefficient and unbalanced solution. The tree-based routing scheme called Whirl [16] has been introduced to balance link loads and provide broadcast that ensures non-duplicate packet reception. In this scheme, the source node decides the route for packets. For every broadcast packet, the route is encoded in two bits: the Left Turn Bit (LTB) and the Right Turn Bit (RTB). The current flit motion direction decides its route.

2) Wireless-based Broadcast: For a fast broadcast message, the operations are handled by the wireless NoC architecture, where each core uses an antenna and a transceiver to send and receive the broadcast signals [9]. This architecture is divided into two network planes such as wired plane and wireless plane. The wireless plane is mainly designed for the broadcast operations, and the wired plane is used for unicast traffic only. A hybrid controller is introduced to handle these unicast and broadcast traffic at runtime. In this case, overall system performance in terms of network latency is degraded due to congestion and token holding time at the wireless interfaces as shown in results Section 4.2.

3) Hybrid Broadcast Implementation: To improve the overall system performance and avoid the power and area overheads due to a large number of transceivers and antennas components, we have implemented broadcast mechanism with the help of source routing. Despite the power and area overheads, the advantages of source routing include in-order packet delivery, faster and multiple non-minimal and minimal path routing [17]. An inefficient broadcast operation can affect the overall system performance by up to 40% [13]. Deterministic source routing is one of the most suitable candidates for a broadcast operation. For the intra-cluster and source-routing broadcast communication, all the information related to source and destination path is pre-computed in the form of tree-based load-balanced paths inspired by the Whirl method [16] and stored in a source routing table. We modify the router design to incorporate the source routing scheme. A higher priority is provided to broadcast communication.

For inter-cluster communication, the WISR decides the route of a packet, which is associated with each WI. We consider omnidirectional setup along with token passing protocol to access wireless medium. Any node within the cluster sends a broadcast request packet to the WISR via a unicast packet to the WI. The broadcast packet is transmitted using wireless medium if a token is available for inter-cluster communication; otherwise, the broadcast packet traverses using wired links and ad hoc packet duplications. That is why this broadcast scheme is called hybrid broadcast mechanism. This approach avoids waiting time that would take away the advantage of single-hop wireless links. For intra-cluster, the source router randomly chooses four bits LTBW, LTBN, LTBE and LTBS for each direction W, N, E and S, respectively [16]. At the WISR, the RTBs for each direction is computed from the complement of LTBs as shown in Figure 3. For example, WISR randomly chooses four bits of LTBs: 0, 0, 1, and 1 for N, E, S, and W respectively and computes the corresponding RTBs as shown in the figure. If a given LTB bit is high, then the flit should turn left, and if RTB bit is high, then flit should turn right relative to its current direction. Finally, the (LTB, RTB) pairs create a broadcast tree cluster-wise as shown in the Figure 3.

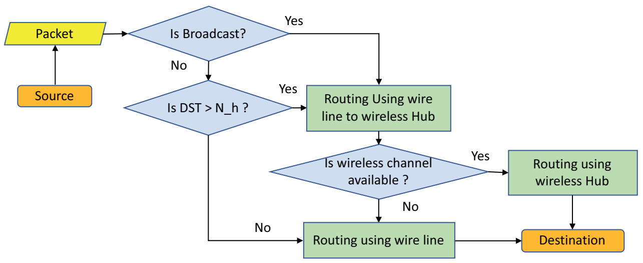

Packet
Source
Is Broadcast?
Yes
No
Is DST > N_h ?
Yes
Routing Using wire line to wireless Hub
Is wireless channel available ?
Yes
Routing using wireless Hub
No
No
Routing using wire line
Destination

Fig. 4: Flowchart showing the packet control mechanism.

Figure 4 describes the runtime hybrid mechanism for packet transmission from the source (SRC) to the destination (DST). For unicast, the number of hop counts (N_h) between SRC and DST is 4 in the simulation setup. We get benefits from wireless links when hop count is greater than 4, which is discussed in [8]. The packet flow route depends on multiple conditions. Initially, broadcast packets are reached to nearest WISR (within a cluster) using XY routing. Intra-cluster packets are transmitted using source routing. Broadcast packets, with a token available, adopt the wireless path for inter-cluster communication. In the latter, instead of waiting for a token for using the wireless links, simultaneous multiple broadcast operations can take place using wired and wireless links. When packets arrive at the DST WI hub, they are transmitted again using source routing from DST WI hub to all cores within the cluster. It is also important to notice that broadcast operations using wireless links are prioritized over long unicast operations to avoid the blockage of paths. This hybrid scheme provides an efficient routing strategy for parallel applications. Figure 5 shows the hardware level implementation of DBMC. The broadcast packets are transmitted over wireless path based on the availability of a token at WI. Deadlock cases are avoided for long distance unicast and for broadcast packets as source routing is used where the paths are pre-computed.

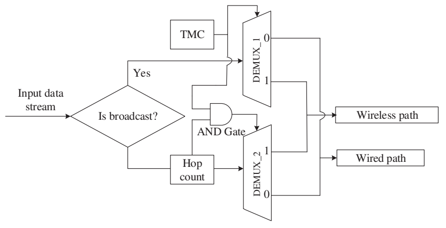


Fig. 5: Dynamic Broadcast Mode Controller circuit.

IV. Performance Evaluation

This section explores the performance benefits of using hybrid communication links for parallel applications that employ barrier synchronization. We compare the proposed architecture to the baseline NoC, Whirl scheme based NoC (Whirl-NoC) and WiNoC architectures. We also address power-aware WiNoC architecture.

TABLE II: Simulation Setup

Arch	Component	Configuration
System	CPU	1GHz $\times$ 86 ISA cores In-order
	L1 cache	private 64KB, 2-way, LRU policy, 64B line, 1 cycle latency
	L2 cache	shared 256KB, 8-way, LRU policy, 64B line, 10 cycle latency
	Cache coherency protocol	MESI with cache directory
Network	Topology	12 $\times$ 12 Mesh, WiNoC
	Routing	XY routing, source routing
	Flit Size and Packet Size	32 bits and 2 flits broadcast and 8 bits unicast

A. Simulation Setup

We consider two applications from the PARSEC benchmark [10]: Bodytrack (small percentage of broadcast messages) and Streamcluster (highest barrier usage) as a representative of standard benchmarks for barrier synchronization. The communication traces are extracted by running the Gem5 [18] full system simulator. The communication traces of the applications are extracted for medium (med) and large (lar) inputs, defining the time the application is being executed. Bodytrack implements the application employing 4 barriers (consequently, 4 sources) and uses 3 types of threads to execute its computation. Streamcluster employs a single barrier for the application implementation; hence, only a single source. Both applications can be released only when 63 cores have reached the barrier. The release procedure is done with a broadcast. Hence, every 64^{th} barrier event occurs a broadcast. The total numbers of broadcast messages are 210, 267, 11390, 1068, and 45560 for Bodytrack_med, Bodytrack_lar, Streamcluster_med, 4x_Bodytarck and 4x_Streamcluster, respectively. The traces are executed on a modified version of the Noxim simulator [11] to validate our proposed architecture concerning latency, throughput and communication energy. The OOK modulation [5] based wireless interconnect along with token passing protocol is implemented within the network simulator. The proposed system, which is a standard system size in current manycore technology trends, consists of 64 cores. The width of all wired links is the same as the flit size (i.e., 32 bits). The NoC switches are driven with a 2.5 GHz clock. The power-gated components of WIs are designed and implemented using Cadence tools. DBMC is synthesized from RTL description by using Synopsys Design Compiler with 28nm CMOS technology. We use Cadence tools to obtain the area usage and power dissipation, and the delay of sleep transistors. Table II presents the summary of the simulation setup. In this work, we consider four NoC topologies. In case of the mesh wired NoC and regular WiNoC, the broadcast messages are transmitted in the form of multiple unicast messages. In case of the Whirl-NoC, broadcast messages are transmitted based on the Whirl algorithm [16]. Finally, for the proposed architecture, both unicast and broadcast messages are transmitted using a hybrid wired/wireless network. Broadcast messages within each cluster are implemented using source routing based on the Whirl algorithm, where pre-computed paths are stored in a table. Inter-cluster communication is implemented using wireless links if a token is available at WI; otherwise, wireline with XY is employed between WISR nodes. In section 4.5 and 4.6, we provide a rough estimation of power [9] where all works have their resulted scaled down to 28nm CMOS for a fair comparison.

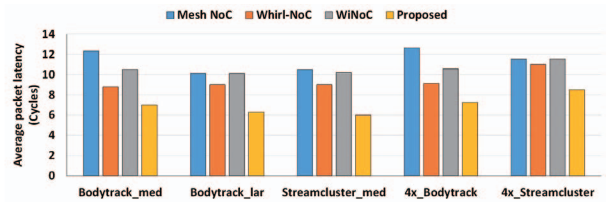


Fig. 6: Average packet latency of different NoC architectures.

B. Network Latency Reduction

Figure 6 shows that the proposed architecture reduces the average packet latency when compared to the mesh wired NoC, Whirl-NoC and WiNoC architectures. It can be observed that regular WiNoC does not outperform the wired mesh NoC in all cases. In the case of Bodytrack_lar the mesh wired NoC performs better than regular WiNoC, but in the case of 4x_Streamcluster they are equivalent. Hence, regular WiNoC cannot be considered as the best solutions for broadcast operation. However, the proposed architecture reduces the network latency up to 43.24% for Bodytrack_med, 37.86% for Bodytrack_lar, 42.79% for Streamcluster_med, 42.79% for 4x_Bodytrack and 26.27% for 4x_Streamcluster over the mesh wired NoC. The proposed architecture reduces the average packet latency up to 50.80% for Bodytrack with medium input, 37.86% for Bodytrack with large input, 41.14% for Streamcluster with medium input, 31.46% for 4x_Bodytrack and 26.37% for 4x_Streamcluster over the WiNoC. The proposed architecture reduces the network latency up to 20.45% for Bodytrack with medium input, 30.23% for Bodytrack with large input, 33.33% for Streamcluster, 20.46% for 4x_Bodytrack and 22.76% for 4x_Streamcluster over the Whirl-NoC. These results demonstrate the proposed architecture achieves significant performance improvements over the existing NoC architectures for broadcast operations.

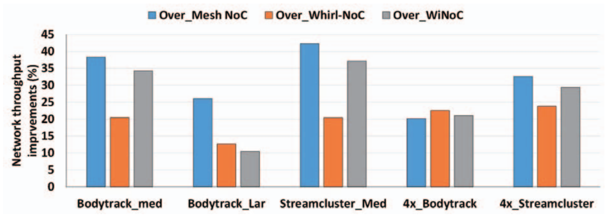


Fig. 7: Throughput improvements over mesh NoC and the regular WiNoC

C. Network Throughput Improvements

Figure 7 shows the peak of the network throughput improvements at saturation over mesh wired NoC, Whirl-NoC and WiNoC architectures. The proposed architecture, as expected, improves performance over mesh topology due to the presence of single hop, long-range wireless, and wired links. From the throughput comparison, it can be observed the hybrid architecture provides better throughput than conventional mesh, Whirl-NoC and WiNoC. It improves the throughput over regular WiNoC by 34.2% for Bodytrack_med, 10.5% for Bodytrack_lar, 37.16% for Streamcluster_med, 21.06% for 4x_Bodytrack and 29.42% for 4x_Streamcluster. Similarly, the proposed architecture improves the throughput up to 20.44% for Bodytrack_med, 12.65% for Bodytrack_lar, 20.40% for Streamcluster_med, 22.55% for 4x_Bodytrack and 23.83% 4x_Streamcluster benchmarks over the Whirl-NoC.

D. Communication Energy Saving

Figure 8 presents the packet energy savings achieved by the proposed architecture according to the application. The packet energy is the energy dissipated in transferring one packet completely from source to destination at network saturation. In order to evaluate our proposed architecture in terms of packet energy, we use energy model for wired link from [9] and wireless link from [19]. The results summarize the average energy consumption of a single packet for mesh NoC, Whirl-NoC, WiNoC, and our proposed architecture. From the figure, it can be observed the proposed architecture saves the communication energy per packet by 17.34% for Bodytrack_med, 41.65% for Bodytrack_lar, 42.65% for Streamcluster_med, 27.52% for 4x_Bodytrack and 26.39% for 4x_Streamcluster over the WiNoC. The proposed architecture also saves energy up to 22.16% for Bodytrack_med, 20.50% for Bodytrack_lar, 28% for Streamcluster_med, 17.54% for 4x_Bodytrack and 23.50% for 4x_Streamcluster over the Whirl-NoC. Therefore, we achieve a significant amount of energy saving using power-gated WI, which will be discussed in the next section.

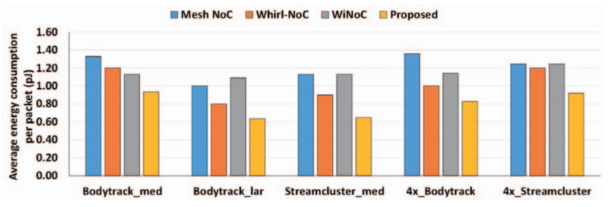


Fig. 8: Average energy consumption of NoC networks.

E. Power Overhead and Saving

Additionally, we investigate the power dissipation at WIs partially. The power dissipation overhead of additional components (i.e., controllers, source routing and power-gating) is 0.30mW. The area overhead for the DBMC is 10.43 μm^2 each. Area occupation is estimated directly from the hardware implementation of DBMC. For parallel applications, the percentage of broadcast messages (5%) is very small but crucial for the system performance. Hence, the significant amount of communications (95%) is based on unicast. Hence, the objective is to reach high-performance when necessary (5%) while minimizing the power consumption, so that the broadcast mechanism do not degrade the energy efficiency. Although our architecture must be efficient for barrier synchronization, these events occur very widely, as they demand milliseconds of application computation. From the hardware perspective, delays of milliseconds can result in millions of inactive cycles, which can be timely exploited by power-gating the WIs. As we consider a token passing protocol based WiNoC, all WIs are not required during unicast. Therefore, there is a huge scope to reduce the DC power dissipation for parallel applications. We adopted the scheme from [19] for WIs to improve the power efficiency. After applying power-gating at WI, the proposed architecture reduced the DC power dissipation in WIs up to 50.46% during 95% of the total simulation time. The major impact regarding wake-up latency is negligible if compared to the delays between each synchronization event. This was achieved not only without any significant performance degradation but also maintained the throughout and latency improvements.

F. Summary of Proposed and Existing Works

Table III presents a summary of proposed and existing works on barrier synchronization for NoC-based systems. In [9], power consumed by router and WI are 6mW and 16mW. Hence, power dissipation by each router is 384mW and total power dissipation by WIs is 1024mW, which is significantly high dissipation. However, in case of our proposed architecture, total power consumption is 384mW for each router, and 64mW for WIs, which is significantly lower as compared to fully radio solution as in [9]. Note also that our approach deals with Hub availability and it is independent of the MAC protocol, so it is compliant with other techniques such as collision detection as used in [9]. Compared to conventional NoC, our proposed hybrid method, significantly reduces the latency and power consumption with small area overhead (less than 1%). The experimental results also show that WiNoC alone is not the ideal solution for broadcast operations, since it can be outperformed by the Whirl-NoC architecture.

TABLE III: Summary of proposed and existing works.

Ref.	Approach	Saving	Penalty
[2]	Transmission-line based broadcast	Worst-case latency: 4ns to 10ns	0.07% of total metal area overhead
[3]	Tree-based broadcast	20% during off-centered	1.3% power overheads
[11]	OrthoNoC/ WiSync	Latency improvement: 30%	64 WIs for 64-core system.
This	Broadcast enabled WiNoC	Latency decreases up to: 42.79% Energy: 42.65% DC power by WI: 50.46%	Less than 1% area overhead only 4 WIs for 64-core system

V. Conclusion

In this paper, we propose an efficient broadcast mechanism based on hybrid wireless/wired NoC to improve the performance of parallel applications by reducing barrier synchronization latency significantly. The experimental results show the proposed method reduces the latency and communication energy consumption significantly. It also improves the network throughput. Besides, we employed the power-gating method with WIs to diminish the DC power dissipation. We observe the proposed WiNoC architecture reduces network latency by up to 42.79% over conventional NoC architectures under applications of PARSEC benchmark. The proposed method also saves up to 50.46% the WIs power dissipation compared to the conventional WiNoC.

Acknowledgment:: This work is part of BBC project, supported by Labex CominLABS.

References

[1] J. Oh *et al.*, "Tlsync: Support for multiple fast barriers using on-chip transmission lines," in *2011 38th Annual International Symposium on Computer Architecture (ISCA)*, June 2011, pp. 105–115.

[2] Z. Wei *et al.*, "Tab barrier: Hybrid barrier synchronization for noc-based processors," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 409–412.

[3] V. F. Pavlidis *et al.*, "3-d topologies for networks-on-chip," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 10, pp. 1081–1090, Oct 2007.

[4] A. Shacham *et al.*, "Photonic networks-on-chip for future generations of chip multiprocessors," *IEEE Transactions on Computers*, vol. 57, no. 9, pp. 1246–1260, Sep. 2008.

[5] K. Chang *et al.*, "Performance evaluation and design trade-offs for wireless network-on-chip architectures," *J. Emerg. Technol. Comput. Syst.*, vol. 8, no. 3, pp. 23:1–23:25, Aug. 2012.

[6] H. Matsutani *et al.*, "A case for wireless 3d nocs for cmps," in *2013 18th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2013, pp. 23–28.

[7] S.-B. Lee *et al.*, "A scalable micro wireless interconnect structure for cmps," in *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '09. New York, NY, USA: ACM, 2009, pp. 217–228.

[8] S. Deb *et al.*, "Design of an energy-efficient cmos-compatible noc architecture with millimeter-wave wireless interconnects," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2382–2396, Dec 2013.

[9] S. Abadal *et al.*, "Orthonoc: A broadcast-oriented dual-plane wireless network-on-chip architecture," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 3, pp. 628–641, March 2018.

[10] C. Bienia *et al.*, "The parsec benchmark suite: Characterization and architectural implications," in *2008 International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Oct 2008, pp. 72–81.

[11] V. Catania *et al.*, "Cycle-accurate network on chip simulation with noxim," *ACM Trans. Model. Comput. Simul.*, vol. 27, no. 1, pp. 4:1–4:25, Aug. 2016.

[12] J. L. Abellán *et al.*, "Efficient hardware barrier synchronization in many-core cmps," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 8, pp. 1453–1466, Aug 2012.

[13] A. Karkar *et al.*, "A survey of emerging interconnects for on-chip efficient multicast and broadcast in many-cores," *IEEE Circuits and Systems Magazine*, vol. 16, no. 1, pp. 58–72, Firstquarter 2016.

[14] B. A. Floyd *et al.*, "Intra-chip wireless interconnect for clock distribution implemented with integrated antennas, receivers, and transmitters," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 5, pp. 543–552, May 2002.

[15] N. Mansoor *et al.*, "An energy-efficient and robust millimeter-wave wireless network-on-chip architecture," in *2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, Oct 2013, pp. 19–24.

[16] T. Krishna *et al.*, "Towards the ideal on-chip fabric for 1-to-many and many-to-1 communication," in *2011 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2011, pp. 71–82.

[17] S. Mubeen *et al.*, "Designing efficient source routing for mesh topology network on chip platforms," in *2010 13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools*, Sep. 2010.

[18] N. Binkert *et al.*, "The gem5 simulator," *SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, Aug. 2011.

[19] H. K. Mondal *et al.*, "P2noc: Power- and performance-aware noc architectures for sustainable computing," *Sustainable Computing: Informatics and Systems*, vol. 16, pp. 25 – 37, 2017.