System-Level Exploration of In-Package Wireless Communication for Multi-Chiplet Platforms

Rafael Medina[1], Joshua Kein[1], Giovanni Ansaloni[1], Marina Zapater[2]
Sergi Abadal[3], Eduard Alarcón[3] and David Atienza[1]
[1]Embedded Systems Laboratory (ESL), EPFL, Switzerland, [2]REDS Institute, HES-SO, Switzerland,
[3]NaNoNetworking Center in Catalonia (N3Cat), UPC, Spain

ABSTRACT

Multi-Chiplet architectures are being increasingly adopted to support the design of very large systems in a single package, facilitating the integration of heterogeneous components and improving manufacturing yield. However, chiplet-based solutions have to cope with limited inter-chiplet routing resources, which complicate the design of the data interconnect and the power delivery network. Emerging in-package wireless technology is a promising strategy to address these challenges, as it allows to implement flexible chiplet interconnects while freeing package resources for power supply connections. To assess the capabilities of such an approach and its impact from a full-system perspective, herein we present an exploration of the performance of in-package wireless communication, based on dedicated extensions to the gem5-X simulator. We consider different Medium Access Control (MAC) protocols, as well as applications with different runtime profiles, showcasing that current in-package wireless solutions are competitive with wired chiplet interconnects. Our results show how in-package wireless solutions can outperform wired alternatives when running artificial intelligence workloads, achieving up to a 2.64× speed-up when running deep neural networks (DNNs) on a chiplet-based system with 16 cores distributed in four clusters.

CCS CONCEPTS

• **Hardware → Radio frequency and wireless interconnect**; **3D integrated circuits**; **Simulation and emulation**.

KEYWORDS

Multi-Chiplet Systems, On-Package Wireless Communication, Full System-level Simulation, DNNs.

ACM Reference Format:

Rafael Medina[1], Joshua Kein[1], Giovanni Ansaloni[1], Marina Zapater[2], Sergi Abadal[3], Eduard Alarcón[3] and David Atienza[1]. 2023. System-Level Exploration of In-Package Wireless Communication for Multi-Chiplet Platforms. In *28th Asia and South Pacific Design Automation Conference (ASPDAC '23), January 16–19, 2023, Tokyo, Japan.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3566097.3567952

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ASPDAC '23, January 16–19, 2023, Tokyo, Japan
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-9783-4/23/01…$15.00
https://doi.org/10.1145/3566097.3567952

1 INTRODUCTION

Many-Core systems are nowadays widely adopted to address the ever-increasing computation requirements of applications in many domains, such as those of artificial intelligence, by supporting a high degree of parallelism. However, the integration of these systems into a single chip causes design and manufacturing drawbacks. From a manufacturing perspective, increases in die sizes negatively impact yield, and are limited by the maximum size of photomasks. Design issues stem instead from the challenge of integrating and interfacing diverse computing and storage elements on the same die. Chiplet-based solutions overcome these obstacles by hosting, on the same package, several small dies side-by-side. These are then connected by microbumps to the package substrate [13]. The physical proximity of chiplets allows for performance close to that of large monolithic dies, but without the consequent impact on yield. Moreover, the modularity of this approach can be leveraged to pack together chiplets implementing different components (Intellectual Properties, IPs) and process nodes. Chiplets can even be reused across different systems [13, 17], lowering nonrecurring engineering costs.

However, chiplet integration is challenging from a connectivity and power delivery perspective. A first issue arises from the limited number of chiplet-to-substrate microbumps, which support both supply and signal connections. This constraint is caused by the large size of microbumps and the distance between them (both in the order of tens of micrometers in today's technology [21]). A second key challenge is that communication interfaces may be incompatible between chiplets, with various chiplets in the same package implementing different communication protocols, in addition to having different bandwidth and power requirements [13]. Although various chiplet-based interconnect protocols have been proposed [6, 14, 17, 18], standardization efforts are still in their infancy and are not widely adopted [21].

Recently, short-distance wireless communication enabled by on-chip transceivers and nanoantennas has been proposed to address challenges derived from the high integration effort and the paucity of interconnect resources [11] in chiplet-based platforms. Current wireless technology can support very high bandwidths (up to 120 Gbps have been recently demonstrated in [23], and higher bandwidths are promised by emerging graphene technology [1]) and therefore can potentially enable flexible and high-performance in-package connectivity. As shown in Figure 1, by eliminating the need for wires to carry signals, the use of nanoantennas addresses both challenges of chiplet integration outlined above. First, more microbumps are available for power delivery networks to the chiplets, facilitating their design. Second, connectivity protocols are handled

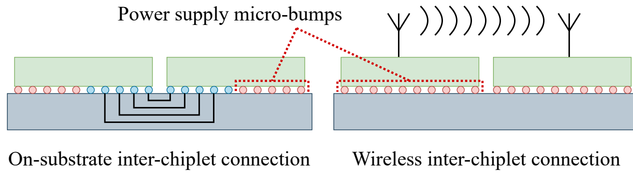

Figure 1: State-of-the-art chiplet-based system where both power and signals are distributed via the substrate (left) and system where signals are transmitted with a wireless interconnect, freeing all microbumps for power supply (right).

by wireless transceivers, transparently from the IPs in the chiplets, hence greatly easing the integration of heterogeneous components.

In this context, the aim of this paper is to explore, from a system-wide perspective, the benefits and pitfalls of in-package wireless interconnects. For the first time, we present a full-system exploration of multi-chiplet wireless systems, executing complex workloads having varying execution patterns. We consider several in-package wireless approaches employing different medium access control (MAC) protocols and bandwidths. In summary, the contributions of this paper are the following:

- We explore the system-level performance of inter-chiplet wireless connectivity across multiple dimensions, ranging from the system architecture to the application characteristics, to the employed medium access control protocol.
- We introduce abstract, yet realistic, modules for modeling in-package wireless communication and collisions. These modules can be flexibly instantiated in virtual platforms to perform system simulations transparently from applications and other system components.
- We show that in-package wireless interconnects can compete with wired chiplet interconnects in a wide range of applications and can even outperform them. In the case of DNN workloads, speed-ups of up to 2.64× are obtained in a 4-cluster system.
- We discuss the impact of the choice of MAC protocol on run-time performance, showcasing that token passing schemes are the best suited for currently attainable bandwidths.

2 EXPLORING WIRELESS INTERCONNECTS

The system-level design space of in-package wireless interconnect solutions encompasses system integration, i.e. the choice of elements to be interfaced via on-chip wireless communication, and the configuration of the transceivers, i.e., how the physical layer and MAC mechanisms are implemented.

Multiple interconnect alternatives can be explored for system integration. A key design decision in this regard is the granularity with which wireless links should be established. In particular, as shown in Figure 2, each processing core and its L1 caches can be interfaced with a dedicated antenna, or processors can be grouped into clusters that share the same antenna at the L2 level. In the latter case, processors in a cluster have to compete for accessing the wireless interconnect, but smaller wireless networks are implemented, which incur in lower collision rates.

A complementary dimension of design-space exploration focuses on the design of the wireless connectivity strategy, as implemented

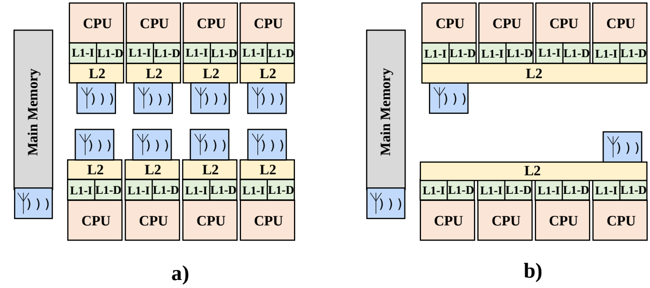

Figure 2: Wireless interconnect configured at different granularities: a) per-core, b) per-cluster.

by on-chip transceivers. Transceivers act as interfaces between components accessing the communication network (e.g., processors, memories, etc.) and the nanoantennas. At the lowest level of abstraction, they implement a physical layer, while at a higher level, they provide an MAC strategy. The physical layer enables serialization/deserialization and modulation/demodulation of data. This layer can also detect collisions when more than one antenna tries to transmit simultaneously. Instead, the MAC layer enforces a protocol that regulates when data should be sent and handles collisions. MAC protocols determine the mechanism for transmission arbitration and collision handling, thus playing a key role in determining run-time performance.

In this work, we analyze how the design choices described above interact in conjunction with technology constraints (i.e., available bandwidth) and application characteristics. Therefore, a multi-parametric exploration is performed in the following manner:

(1) Workloads with varying inter-chiplet communication requirements are targeted.
(2) The number of connected nodes is varied.
(3) A range of wireless bandwidth values is employed.
(4) Different MAC protocols are considered.

Our approach differs from that of recent papers advocating for in-package wireless in many-core systems in two key aspects. First, related works [5, 7, 8, 10, 11] assume the implementation of hybrid interconnects where data may be sent/received via wires or nanoantennas. The routing of packets on these systems generally follows one of two approaches: it can be dynamic, choosing between wired or wireless link depending on load and shortest path [5, 7], or static, reserving the wireless channel only for a particular set of communications [8, 10, 11]. However, hybrid architectures do not address the chiplet integration challenges described in Section 1. As opposed to these works, we instead explore whether in-package wireless can be considered as a replacement for inter-chiplet wired connections, and the circumstances that have an impact on the performance of such a system. Second, as opposed to [8, 10, 11], we do not focus data communication challenges in isolation, but we consider it an element of an entire hardware/software stack, employing full-system simulations to gauge the interactions among those elements in complex architectures and workloads.

3 MODELING WIRELESS COMMUNICATION

In this paper, we introduce a wireless component capable of modeling wireless on-chip communication in full-system simulation,

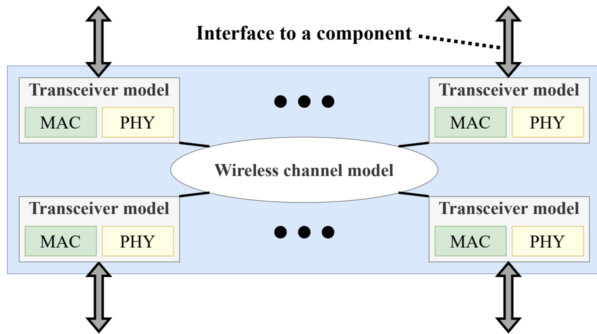

Figure 3: Wireless component modeling the elements of the interconnect: the wireless channel and one transceiver per interfaced component.

enabling explorations introduced in the previous section. The component is implemented as an extension to gem5-X [19], itself based on the industry-standard gem5 simulator [3]. gem5-X emulates the processor ISA, system architecture, I/O hardware, and OS, providing accurate estimations of the energy efficiency and performance of a system when executing an applications. Wireless connections are emulated via a novel module, which can be seamlessly instantiated when defining virtual platforms, transparently from other hardware components as well as software. In the following, we detail the implementation of the physical and MAC layers of the module, as well as the methodology to integrate it into virtual systems.

3.1 Physical layer

The wireless module comprises a physical layer model (termed "PHY" in Figure 3) for each of the devices that are interfaced to the wireless link. The physical layer manages the transmission of data on the channel according to the employed MAC protocol (see Section 3.2) and detects collisions among transmissions. The model is parametric in the available bandwidth and in the time required for accessing the channel via the antenna, which accounts for the delay incurred for the serialization, modulation, and transmission of data. At the physical layer, data transfers are performed according to the following steps:

(1) Upon receiving a data transmission request, the wireless module checks whether the corresponding transceiver is busy, in which case the request is stalled. This occurrence is common when multiple processors in a cluster share the same transceiver.
(2) When the transceiver is free, the duration of a transmission is computed taking into account the channel bandwidth and the size of the data to be sent.
(3) Transmissions start at the allocated time according to the employed MAC protocol, checking for possible collisions.
(4) Data transfers end upon a successful completion (after the previously computed transmission time) or upon the detection of a conflict. In all cases, additional delays are added to take into account serialization and modulation at the transmitter and demodulation and deserialization at the receiver.

The wireless component performs these steps whenever a transmission request is received from one of the interfaced elements, while hiding their implementation from them. In turn, the component is oblivious of the transfer format and does not discriminate among them. Hence, it can handle any type of data transfer, including cache coherency traffic through snoop messages.

3.2 MAC layer

MAC protocols implement the mechanisms required for synchronization, fair access, and collision handling in a shared transmission medium, as is the case of wireless transmission links, which are the focus of this work. Here, we compare and contrast two protocols, emulated in the MAC layer of the wireless transmission module (as shown in Figure 3), which have complementary characteristics. (1) exponential backoff [16] is a random access protocol that manages collisions by retransmitting lost data, while (2) token passing [8] is a controlled access protocol, where conflicts are avoided at the cost of a latency overhead even in the absence of congestion. The choice of random access and controlled access protocols illustrates the two ends of the latency-throughput trade-off. While a wide variety of MAC protocols has been proposed, these variants or combinations of exponential backoff and token passing [9].

Figure 4 illustrates the behavior of the exponential backoff protocol [16]. When employing it, a transceiver receiving a transmission request waits a random time within a dynamically-sized window. The window size is dynamically adjusted according to the network load, in order to balance the transmission rate and the probability of causing a collision. When a collision is detected by the physical layer, the window size increases, and the transceivers involved reschedule the corresponding transmissions. Similarly, when a successful transmission is observed, the length of the window is reduced. The implemented exponential backoff protocol allows to modify the rates of growth and reduction for the window size, as well as tune the maximum window size.

In token passing [8], a token is shared by the transceivers in the interconnect. Only one node owns the token at a time, allowing it to transmit, as shown in Figure 5. The ownership of the token is passed to the next transceiver either at the end of a transmission or after a silent cycle in which the owner node did not initiate a transmission. Since all transceivers can detect the presence or lack of a data transfer across the wireless medium, they can independently update the token owner without a synchronization mechanism.

Different protocols are best suited to different run-time conditions: as explored in Section 5, exponential backoff is better suited to high bandwidth, low traffic scenarios where low latency is achieved, while token passing better adapts to heavy network loads, where it can optimize throughput.

3.3 System Integration

The wireless interconnect module outlined above can be instantiated in a simulated system by connecting computing and storage components at its interfaces. This strategy allows its integration at multiple points of a system hierarchy, including but not limited to the configurations explored in Section 5. Processors, main memories, caches, and accelerators can all be interfaced, as long as they abide by the conventions defined for gem5 components [3].

In more detail, two types of interface are provided to connect elements, facing upwards and downwards in the system hierarchy. Elements generating data requests, such as processors or lower-level caches, are connected to the side facing upward, whereas

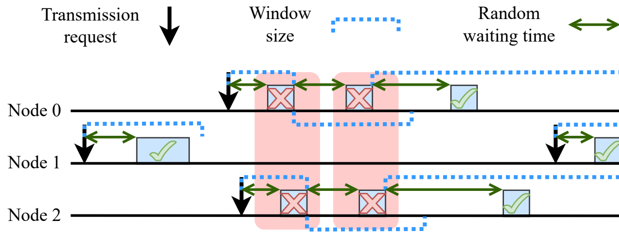

Figure 4: Timing diagram showing the behavior of the exponential backoff protocol under successful transmissions (green ticks) and collisions (red crosses).

elements satisfying those requests, such as memories or higher-level caches, are connected to the downward side. This approach allows to support cache protocols by managing snooping messages. These memory consistency mechanisms are further leveraged to handle synchronization between the interconnected elements.

This structure is similar to the standard interface mechanism provided by gem5, as wireless-specific features for managing the physical and MAC layers are black-boxed at the module interface. Therefore, the OS and applications do not need to be modified to be correctly executed on a wirelessly capable emulated system.

4 EXPERIMENTAL SETUP

4.1 Architectures

As test vehicles for assessing the performance of in-package wireless technology, we considered two realistic systems. The first one comprises four clusters of four cores and a shared L2 cache [17]. Each cluster is implemented in a separate chiplet, and a fifth one is added to accommodate the memory controller. One wireless transceiver per chiplet is assumed. The second system is made up of 16 clusters, where each cluster includes a single core and a private L2 cache, whose size is scaled down to keep the aggregated L2 size constant. In this configuration, 17 chiplets are employed to incorporate the 16 clusters and the memory controller. Again, we considered one transceiver per chiplet. A detailed view of these two configurations is shown in Table 1.

To compare the performance of the described system employing various chiplet-to-chiplet communication mechanisms, three different interconnects are evaluated. First, a single-channel wireless interconnect is simulated where a nanoantenna and transceiver are interfaced to each of the system chiplets. To analyze different wireless configurations, we employ the token passing and exponential backoff protocols described in Section 3.2, while varying the bandwidth of the channel. We also assume a delay overhead of 3 clock cycles due to serialization and deserialization performed by the physical layer, and to package routing [7, 8]. Next, as a first baseline, we modeled a serial wired interconnect with an overall crossbar-like topology. The parameters used for each chiplet-to-chiplet link are a 112 Gbps bandwidth and a latency of 100 nanoseconds, which reflect the state of the art in interchiplet communication implementations [17, 22]. A further baseline considers an —unrealistic— ideal interconnect through which any data transfer only takes one clock cycle. This scheme provides an upper performance bound from an interconnect perspective.

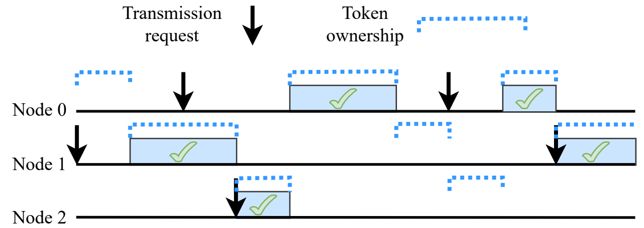

Figure 5: Timing diagram of the token passing protocol.

Table 1: Configuration of the two simulated systems: (a) a 4-cluster system with four cores per cluster and (b) a 16-cluster system with one core per cluster.

Processors		16x cores @2.0 GHz ARMv8, out-of-order
L1 Instruction Caches		16x private, 32 kB, 2-way, 2 cycle access
L1 Data Caches		16x private, 32 kB, 2-way, 2 cycle access
L2 Cache	4 clusters	4x shared, 1 MB, 2-way, 20 cycle access
	16 clusters	16x private, 256 kB, 2-way, 20 cycle access
Memory		DDR4 2400 MHz, 4 GB
System Clock		1600 MHz
Operating System		Ubuntu LTS 16.04

Using McPAT [25], we estimate a very low energy overhead due to wireless communication. For the 4-cluster system, assuming that each core is implemented as an ARM Cortex-A75 processor [2] built in 22 nm technology, McPAT reports a total consumed power of 45.9 W. Current in-package wireless technology accounts for a maximum of 160 mW per transceiver when enabling a 120 Gbps bandwidth, using 65 nm process [23]. In such configuration, we assume the total power consumption amounts to 0.8 W in the four-cluster (5-transceiver) system. Therefore, power consumption of the wireless elements represents less than 2% of the total system power. Since further power reductions due to scaling are not considered, this estimation delineates a higher bound for the relative power consumed by the transceivers.

4.2 Workloads

We considered two sets of benchmarks: communication-intensive programs from the STREAM and SPLASH-2 suites [15, 24], and a collection of Convolutional Neural Networks (CNNs) [4, 12, 20].

The communication-intensive set of applications allowed us to evaluate the performance of interconnects under very high traffic loads. Conversely, CNNs showcased a different, more bursty, pattern for data transfers, as activations only need to be transmitted from one chiplet to another at specific points of the execution, e.g. when the computation of a layer has finished.

The communication pattern in CNNs depends on the strategy used to distribute their execution across the cores of the system. We considered a mapping based on intra-layer parallelism for the two versions of MobileNet [12, 20]. In these cases, each CNN layer is distributed among all cores in a fork-join pattern. Instead, for VGG8M [4], we employed inter-layer mapping, where the network is pipelined among the cores, and each core executes the entirety of a single layer.

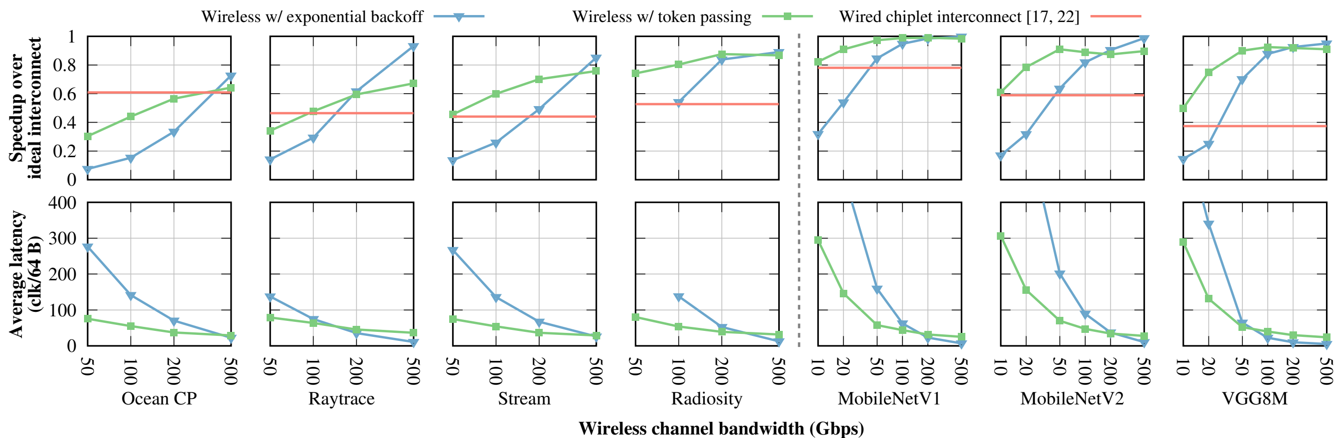

Figure 6: Runtime speed-up over ideal interconnect (top) and average inter-chiplet link latency (bottom) of different chiplet interconnects for representative workloads executed on a 4-cluster system. The dashed line divides the applications of the communication-intensive set (left) and the CNN workloads (right).

5 RESULTS AND ANALYSIS

5.1 4-clusters System

We show a comparison of the performance of the different chiplet interconnects executing the considered benchmarks on the 4-cluster system in Figure 6-top. For wireless links, we consider bandwidths attainable by current technologies (up to 100 Gbps [23]) as well as higher bandwidths, in order to extrapolate our performance results. The results show that wireless interconnection using a token passing protocol achieves more than 40% of ideal interconnect performance, with bandwidths currently attainable of 100 Gbps, when running communication-intensive programs (left part of Figure 6). Furthermore, we observe up to 80% of ideal performance at 100 Gbps for the least demanding set, Radiosity. Overall, token passing over in-package wireless at this link bandwidth is able to outperform the wired interconnect for all the communication-intensive workloads, except for the case of OceanCP. Conversely, wireless interconnects employing the exponential backoff protocol are not able to surpass token passing performance until the bandwidth exceeds 200 or 500 Gbps. The lower performance of exponential backoff is caused by its large number of retransmissions, which are not compensated by a lower best-case latency.

This effect can be further observed in the average transaction latency results reported in Figure 6-bottom: the high number of retransmissions needed at low bandwidths widely increases the time between an initial transmission request and its successful reception. On the other hand, token passing maintains low latencies even at high congestion levels (low link bandwidth), reducing the impact of slow wireless transmissions. For very high bandwidths collisions are less likely, hence exponential backoff achieves lower average latencies and run-times with respect to token passing.

When exploring CNN workloads, the speed-up results in the right part of Figure 6-top illustrate how the wireless interconnect that implements the token passing protocol is able to outperform wired links with bandwidths as low as 10 Gbps. Specifically, it achieves more than 80% of the ideal interconnect performance at 20 Gbps for workloads with intra-layer parallelisms (MobileNet versions 1 & 2) and at 50 Gbps for those that employ inter-layer parallelism (VGG8M). Token passing is more suited for intra-layer parallelism considering that in the fork-join pattern all the nodes try to transmit simultaneously; thus, they are allowed to send data as soon as they hold the token, reducing the amount of silent cycles (i.e., when the token is held by a transceiver but no transmission is required). Conversely, since in the inter-layer approach usually only one node tries to communicate at a time, several silent cycles are experienced between consecutive transmissions from the node. The wireless interconnect with an exponential backoff protocol is able to best the token passing protocols from the 100–200 Gbps range, thanks to the lower likelihood of collisions.

Results showcasing the average data transfer latency when running CNN workloads (on the right part of Figure 6-bottom) highlight that token passing displays a similar behavior as when running the communication-intensive programs, since low-latency data transfers are achieved for all benchmarks. The results regarding exponential backoff depend instead on the applied mapping strategy employed. The fork-join communication patterns present in the intra-layer approaches cause many collisions and subsequent retransmissions, which increases latency. Conversely, latency is lower for inter-layer mappings, thanks to the fewer conflicts incurring when transferring activations in-between layers.

5.2 16-cluster System

The performance of representative CNN workloads executed on the 16-cluster architecture, when using different interconnects, is shown in Figure 7-top. As shown, the token passing protocol over the wireless interconnect exhibits a lower performance when a higher number of transceivers are present, because each one has to wait longer, on average, for the token. Consequently, token passing cannot exceed the performance of the wired interconnect when running MobileNetV1 (the CNN with the highest bandwidth requirements), even for very large-capacity wireless channels. The wired interconnect works better under such high load, since its latency is less affected by congestion than token passing, as it can serve transmissions in the order they are requested. However, token passing is able to outperform it when executing MobileNetV2

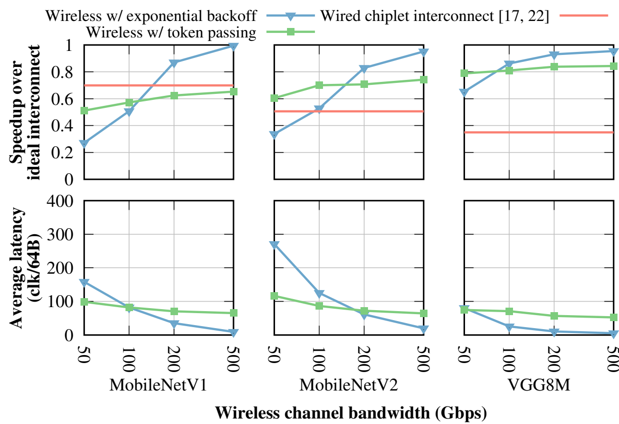

Figure 7: Runtime speed-up over ideal interconnect (top) and average inter-chiplet link latency (bottom) of different chiplet interconnects for representative CNN workloads executed on a 16-cluster system.

and VGG8-M, as bandwidth requirements are lower. Regarding the wireless interconnect employing an exponential backoff protocol, Figure 7 shows lower speed-up results for benchmarks with intra-layer parallelism (MobileNetV1 and V2) with respect to the 4-cluster case, only exceeding the performance of the wired interconnect and that of token passing over wireless with bandwidths higher than 200 Gbps. This performance reduction is due to the greater number of collisions resulting from the increase in the number of transceivers and the rate of data transfers. When executing programs showing inter-layer parallelism (VGG8M), however, collisions are still infrequent even with 16 clusters, and the behavior is similar to the one observed in the 4-cluster system, outperforming token passing with bandwidths at and above 100 Gbps.

The average latency results, depicted in Figure 7-bottom, show that the token passing latency is higher than in the 4-cluster case as a result of the greater amount of nodes in the interconnect. Additionally, it is further increased when executing MobileNetV1 and V2, due to the higher traffic generated by intra-layer parallelism. Exponential backoff, in turn, shows slightly higher latency values than the 4-cluster version.

The 16-cluster results discussed above demonstrate the constraints of scaling a wireless in-package interconnect with current wireless technology. However, as they illustrate, modern many-node networks can match the performance of lower-sized systems under applications with bursty sequential communication patterns. Also, to support a wider range of applications, wireless chiplet interconnects would need to achieve higher bandwidths; we showcase how the 16-cluster system employing a 200 Gbps wireless interconnect can perform better than the wired alternative.

6 CONCLUSIONS

In-package wireless communication shows great promise as an enabler for multi-chiplet designs. While supporting the system interconnect, it does not require chiplet-to-substrate physical connections for data communication, thus decreasing design and integration efforts. To assess the performance of this novel technology from a system perspective, in this paper we have presented a multi-parameter exploration of the capabilities of the emerging wireless in-package technology, enabled by our novel extension to the gem5-X full-system simulator. We have shown that in-package wireless networks can compete with wired and mainstream alternatives. Furthermore, they can outperform them for key workloads, such as CNNs, achieving speed-ups from 1.27× to 2.64× in a 4-cluster system. Additionally, we demonstrated that the choice of MAC protocol greatly impacts performance, with token passing strategies outperforming exponential backoff protocols in the considered systems and state-of-the-art channel bandwidths. Application mappings also impact performance: in the case of CNN, the reduced collision rates of inter-layer parallelization result in lower run-time than intra-layer approaches.

ACKNOWLEDGMENTS

This work has been partially supported by the EC H2020 WiPLASH project (GA No. 863337) and the EC H2020 FVLLMONTI project (GA No. 101016776)

REFERENCES

[1] Sergi Abadal et al. 2022. Graphene-based Wireless Agile Interconnects for Massive Heterogeneous Multi-chip Processors. *IEEE Wireless Communications* (2022).
[2] ARM. 2022. *Arm Cortex-A75 Processor.* https://developer.arm.com/Processors/Cortex-A75
[3] Nathan Binkert et al. 2011. The Gem5 Simulator. *SIGARCH Comput. Archit. News* (2011).
[4] Ken Chatfield et al. 2014. Return of the Devil in the Details: Delving Deep into Convolutional Nets. (2014). http://arxiv.org/abs/1405.3531
[5] Wonje Choi et al. 2018. On-Chip Communication Network for Efficient Training of Deep Convolutional Networks on Heterogeneous Manycore Systems. *IEEE TC* (2018).
[6] Yi Lin Chuang et al. 2013. Unified Methodology for Heterogeneous Integration with CoWoS Technology. In *IEEE ECTC.*
[7] Karthi Duraisamy et al. 2017. Multicast-Aware High-Performance Wireless Network-on-Chip Architectures. *IEEE TVLSIS* (2017).
[8] Vimuth Fernando et al. 2019. Replica: A Wireless Manycore for Communication-Intensive and Approximate Data. In *ACM ASPLOS.*
[9] Antonio Franques et al. 2021. Fuzzy-Token: An Adaptive MAC Protocol for Wireless-Enabled Manycores. In *DATE.*
[10] Sri Harsha Gade and Sujay Deb. 2017. HyWin: Hybrid Wireless NoC with Sandboxed Sub-Networks for CPU/GPU Architectures. *IEEE TC* (2017).
[11] Robert Guirado et al. 2021. Dataflow-Architecture Co-Design for 2.5D DNN Accelerators using Wireless Network-on-Package. In *ASP-DAC.*
[12] Andrew G. Howard et al. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. (2017). http://arxiv.org/abs/1704.04861
[13] Gabriel H. Loh et al. 2021. Understanding Chiplets Today to Anticipate Future Integration Opportunities and Limits. In *IEEE DATE.*
[14] Ravi Mahajan et al. 2016. Embedded Multi-die Interconnect Bridge (EMIB): A High Density, High Bandwidth Packaging Interconnect. In *IEEE ECTC.*
[15] John D. McCalpin. 1995. Memory Bandwidth and Machine Balance in Current High Performance Computers. *IEEE TCCA Newsletter* (1995).
[16] Robert M Metcalfe and David R Boggs. 1976. Ethernet: Distributed Packet Switching for Local Computer Networks. *Commun. ACM* (1976).
[17] Samuel Naffziger et al. 2021. Pioneering chiplet technology and design for the AMD EPYC and Ryzen processor families: Industrial product. In *ISCA.*
[18] Saptadeep Pal et al. 2018. A Case for Packageless Processors. In *IEEE HPCA.*
[19] Yasir M. Qureshi et al. 2021. Gem5-X: A Many-Core Heterogeneous Simulation Platform for Architectural Exploration and Optimization. *ACM TACO* (2021).
[20] Mark Sandler et al. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. (2018). http://arxiv.org/abs/1801.04381
[21] Debendra D. Sharma. 2022. *Universal Chiplet Interconnect Express (UCIe): Building an open chiplet ecosystem.* Technical Report.
[22] Synopsis. 2022. *DesignWare Die-to-Die IP Solutions.* https://www.synopsys.com/designware-ip/interface-ip/die-to-die.html
[23] Korkut K. Tokgoz et al. 2018. A 120Gb/s 16QAM CMOS Millimeter-Wave Wireless Transceiver. In *IEEE ISSCC.*
[24] S.C. Woo et al. 1995. The SPLASH-2 Programs: Characterization and Methodological Considerations. In *ISCA.*
[25] Sam Likun Xi et al. 2015. Quantifying Sources of Error in McPAT and Potential Impacts on Architectural Studies. *IEEE HPCA.*