A Performance Analysis of Chiplet-Based Systems

Neethu Bal Mallya, Panagiotis Strikos, Bhavishya Goel, Ahsen Ejaz, Ioannis Sourdis
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden
{ neethub, strikos, goelb, ahsen, sourdis }@chalmers.se

***Abstract*—As the semiconductor industry struggles to keep Moore's law alive and integrate more functionality on a chip, multi-chiplet chips offer a lower cost alternative to large monolithic chips due to their higher yield. However, chiplet-based chips are naturally Non-Uniform Memory Access (NUMA) systems and therefore suffer from slow remote accesses. NUMA overheads are exacerbated by the limited throughput and higher latency of inter-chiplet communication. This paper offers a comprehensive analysis of chiplet-based systems with different design parameters measuring their performance overheads compared to traditional monolithic multicore designs and their scalability to system and chiplet size. Several design alternatives pertaining to the memory hierarchy, interconnects, and technology aspects are studied. Our analysis shows that although chiplet-based chips can cut (recurring engineering) costs to half, they may give away over a third of the monolithic performance. Part of this performance overhead can be regained with specific design choices.**

I. Introduction

In the multicore era, integrating more resources on a chip is evermore important for the performance scaling of processors. In the past couple of decades, frequency scaling has been limited by power density, and therefore, delivering performance speedup relies primarily on fitting more cores on a chip. However, technology scaling has become more difficult, and large monolithic chips have low yields and, thus, excessive costs. Building chips out of multiple smaller chiplets is a cheaper, higher yield alternative [11], [16], [19], [28].

Die stacking technology has enabled multi-chiplet chips. It was first used for building 3D stacked DRAM chips such as High Bandwidth Memory (HBM) and bringing it closer to processing units, e.g., to a GPU [6] or a vector engine [7]. Later it was employed for disintegrating processors to multiple chiplets, e.g., AMD EPYC and RYZEN architectures, improving yield [19]. Currently, large chips, such as the AMD MI300 [1] and Intel Sapphire Rapids [21], are composed of many CPU and/or GPU chiplets, as well as HBM nodes combining high processing throughput with fast, high-bandwidth memory access.

Despite their improved yield, multi-chiplet chips come with performance overheads. Due to their large size, such systems inevitably use multiple non-uniform access memory nodes. Even early AMD EPYC and RYZEN chips, which provide DRAM access to their CPU chiplets via a single IO die, have a varying access latency by tens of nanoseconds depending on the accessed DRAM controller [19]. AMD MI300 and Intel Sapphire Rapids have even more complex, heterogeneous memory systems composed of multiple HBM nodes and external DRAM. Non-uniform Memory Access (NUMA) machines entail the performance pitfall of long latency remote accesses. Although in the past Cache Only Memory Architectures (COMA) [9] and Cache Coherent NUMA (CC-NUMA) [33] approaches improved data locality and performance of NUMA multi-socket machines, current multi-chiplet chips rely mainly on code optimizations to improve data placement when operating in a flat "HBM + external DDR" mode, or otherwise sacrifice HBM capacity to cache data.

Another performance overhead in multi-chiplet chips, which exacerbates the NUMA effects, is related to the inter-chiplet communication. As opposed to networks on monolithic chips, inter-chiplet connections suffer latency and bandwidth overheads. Exchanging messages with another chiplet requires traversing longer links via microbumps and a silicon interposer, adding extra latency. In addition, the number of available microbumps is limited by the chiplet size and their density constraints, putting a cap on available off-chiplet bandwidth.

Although the yield and cost benefits of multi-chiplet chips have been thoroughly analyzed [11], to the best of our knowledge, the performance with respect to their monolithic counterparts has not been studied. This work fills this gap by evaluating the aforementioned performance overheads of multi-chiplet chips compared to monolithic ones and analyzing the cost-performance tradeoff they offer. We explore the following design points in this study: (i) system size, (ii) chiplet size, (iii) Network-on-Chip (NoC) bandwidth, (iv) Last-Level-Cache (LLC) organization, and (v) silicon interposer type (passive vs. active). We measure the performance overheads of chiplet-based chips varying the above design alternatives and analyze them based on the insight provided by several interconnect and memory system metrics.

Concisely, the contributions of this paper are the following:

- A microarchitectural simulation setup to model large-scale chiplet-based architectures, including detailed models of their memory system and interconnection network;
- The first thorough analysis of performance overheads and cost-performance tradeoffs of chiplet-based chips in comparison to monolithic chips, showing that despite their large cost benefit, chiplet-based designs incur a significant impact on system performance;
- An analysis of various technological aspects that determine specific system parameters such as the length and delay of inter-chiplet links, microbumps budget, yield and cost of chiplet-based and monolithic chips;
- Some design choices are identified to regain some of the performance overheads of chiplet-based chips.

978-3-9826741-0-0/DATE25/© 2025 EDAA

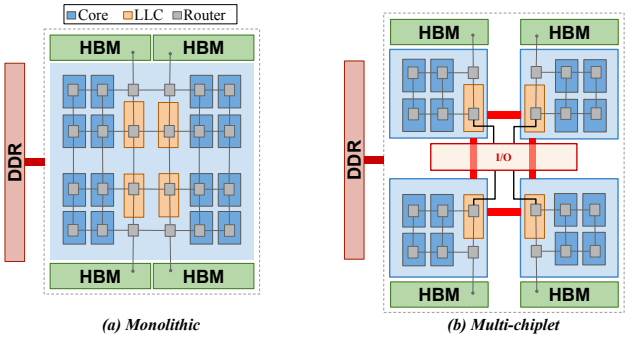


Fig. 1: Monolithic vs. Multi-Chiplet chips.

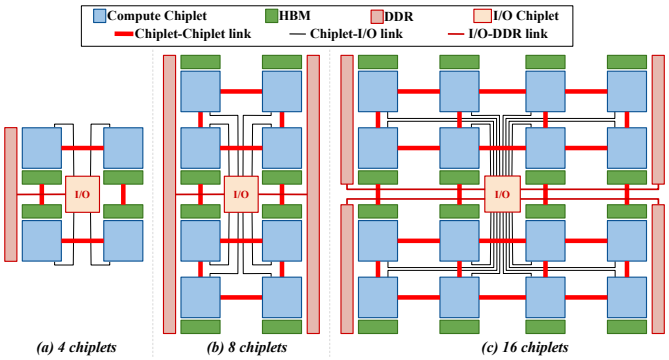


Fig. 2: Different sizes of multi-chiplet chips.

The remainder of this paper is organized as follows: Section II describes the design alternatives analyzed for chiplet-based architectures. Section III explains our experimental methodology. Section IV presents our evaluation results. Finally, Section V summarizes our conclusions.

II. Overview of Chiplet-Based Architectures

The microarchitecture of the chiplet-based chips studied in this paper as well as the monolithic chips used as baselines, are described next. Without loss of generality, the multicore systems are organized in tiles composed of a core with its private L1 and L2 caches, a shared Last-Level Cache (LLC) partitioned in slices, each being closer to a subset of tiles, as well as HBM and external DDR DRAM interfaces. The above are interconnected via a Network-on-chip (NoC), which dedicates a network router for each tile, LLC slice, HBM node, and external DDR controller. The organization and floorplan of the chips illustrated in Figure 1 are inspired by the AMD Zen families [2], [4]. A monolithic chip uses a 2D mesh NoC, has its LLC slices in the middle columns of the 2D mesh, and the HBM and external DDR interfaces at the edges of the chip. The chiplet-based counterpart uses chiplets, which contain a subset of tiles, one or multiple LLC slices and HBM interfaces, while access to external DDR is provided via a separate IO chiplet. The NoC topology within the chiplet remains a 2D mesh, and inter-chiplet links are reduced to a number that can be supported by the microbumps budget of the chiplet, similar to previous work [12], [13].

Figure 2 illustrates the above multi-chiplet organization for different system sizes, i.e., different number of chiplets. It can be observed that the number of HBM nodes scales linearly to the number of chiplets, i.e., one HBM node per chiplet placed next to it. Moreover, the external DDR size and the number of channels also scale linearly to the number of chips.

In the rest of the section, details are provided for system aspects that affect the potential performance overheads of chiplet-based chips. In particular, it describes (i) the design of the interconnects with a focus on inter-chiplet communication and the choice of silicon interposer, (ii) the memory allocation that dictates data placement on the Non-Uniform Memory Access system, and (iii) the choice of the LLC organization. Finally, the yield and cost of the chiplet-based chips are estimated with respect to their monolithic counterparts.

A. Chiplet-based NoCs

One of the first design choices involves deciding whether to use a passive or active silicon interposer for integrating the chiplets. A passive silicon interposer is cheaper as it requires fewer fabrication steps and offers a higher yield, but it offers slower inter-chiplet links because it does not include buffers. An active interposer offers higher throughput because it includes active components to pipeline the links and potentially lower link latency. It may also include network routers offering more advanced topologies. Additionally, an active interposer benefits from lower clock skew/jitter due to repeaters and easier clock synchronization. Minimally active silicon interposers have been shown to have a small cost overhead compared to passive ones [14], [27]. Another design choice studied in previous work is the placement of inter-chiplet links, showing the benefits of concentrating inter-chiplet links to a few edge NoC routers of a chiplet [12], [13].

In our performance analysis, passive silicon interposers as well as minimally active, i.e., with pipelined links, are explored. The total budget of microbumps per chiplet is estimated based on technology parameters, and inter-chiplet links are concentrated to fewer NoC edge routers adjusting their width accordingly. Finally, various intra-chiplet NoC datapath widths are explored, offering different intra-chiplet communication bandwidth.

B. NUMA-aware Memory Allocation

The placement of data in DRAM (HBM and external DDR) is critical for the performance of a NUMA system. Modern operating systems widely support Non-Uniform Memory Access (NUMA) architectures through various mechanisms. For instance, operating systems like Linux [26], Windows [3], and FreeBSD [5] implement NUMA-aware scheduling algorithms that place processes and threads closer to the memory nodes. This approach helps to reduce access latency by optimizing memory locality. Additionally, these operating systems provide APIs that allow user applications to discover the NUMA topology, request memory from specific nodes, and set process affinity, enhancing performance for NUMA-enabled systems.

In this study, we use a Distance-aware Memory Allocation policy. This policy allocates physical memory pages to the

memory node closest to the processor core that first accesses the memory. This approach can significantly improve performance by ensuring that memory is allocated in proximity to the accessing core, reducing the latency of memory access.

C. Last Level Cache Organization

In multi-chiplet systems, the Last-Level Cache (LLC) can be organized through two primary designs:

Sliced LLC: The Sliced LLC architecture, pioneered by Intel starting with the Sandy Bridge micro-architecture, distributes the LLC into multiple "slices" [32]. Each slice acts as an independent cache, but all the slices together form a single logical cache. The physical memory address determines the slice into which data is loaded, effectively distributing memory addresses across slices, and thereby enhancing effective memory bandwidth. In our study, we assign one LLC slice per chiplet and evenly divide the address space among the LLC slices, corresponding to the number of chiplets or High Bandwidth Memory (HBM) nodes associated with a chiplet. The addresses mapped to the external DDR are also divided and assigned into these slices. Accesses from core private caches mapped to local LLC slice exhibit lower latency, whereas accesses mapped to remote LLC slices have to traverse inter-chiplet links, resulting in higher access latency.

Private LLC: Unlike the sliced LLC architecture where the logical LLC cache is distributed across chiplets, in the private LLC architecture each chiplet is assigned its own private LLC cache, and the entire address range is mapped to that private LLC. As a result, all the accesses from the core private caches first go to the local LLC, and only in the case of LLC miss may be required to traverse the inter-chiplet communication link (depending on where the data is mapped in the memory). This design results in reduced inter-chiplet communication overhead compared to the sliced LLC architecture. However, since the same address can now be present in different LLCs across chiplets, LLCs need to be kept coherent, requiring a complex inter-chiplet coherence mechanism. Despite these challenges, the Private LLC approach can offer performance benefits in scenarios where the coherence overhead is manageable or inter-chiplet communication latency is critical.

D. Yield and Cost

The performance analysis of the above multi-chiplet design alternatives needs to be complemented with an evaluation of their cost with respect to their monolithic alternative. This is performed using the Feng-Ma chiplet actuary model [11] with the parameters of the evaluated systems in our study.

More precisely, our work considers monolithic and multi-chiplet chips which are based on AMD EPYC microarchitecture composed of Zen4/Zen4c CPU chiplets manufactured at $5\,\mathrm{nm}$ and an IO chiplet at $14\,\mathrm{nm}$. The chiplet area is estimated, considering 16-core Zen4 chiplets after scaling the L2 and L3 cache sizes to what is used in our performance evaluation, as depicted in Table I. That results in a chiplet size of $66\,\mathrm{mm}^2$, which is similar to the AMD Zen4 chiplets and slightly smaller than the AMD Zen4C chiplet. In addition, each multi-chiplet chip includes an IO chiplet of $400\,\mathrm{mm}^2$, as estimated from AMD EPYC chips of similar technology. Finally, a passive interposer of $65\,\mathrm{nm}$ technology is considered.

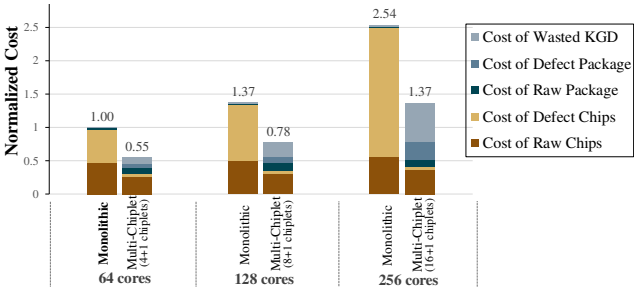


Fig. 3: Cost analysis and comparison of RE cost of monolithic and multi-chiplet chips for different system sizes. Cost is normalized to that of the smallest monolithic.

Based on the above parameters, the above cost model was used to derive the costs of 64-, 128-, and 256-core systems, which are divided in the case of chiplet-based chips to 4+1, 8+1, and 16+1 chiplets, respectively, including the IO chiplet [11]. Figure 3 presents a breakdown of the detailed recursive engineering (RE) cost. The total cost is the sum of the following: (i) cost of raw chips which includes among others the cost of the silicon and the processing during the wafer fabrication, (ii) cost of raw package which covers the materials that are necessary for assembling and packaging the chip as well as testing and verification, (iii) cost of wasted known good dies (KGD) that encapsulates the cost derived from dies that already have been tested to ensure their correct functionality, but still fail, (iv) cost of chip defects, that covers the cost of defects that occur during the wafer fabrication process, and (v) cost of package defects which includes costs related to the packaging process. The total cost is then normalized to that of the smallest monolithic chip.

Overall, the cost of chiplet-based chips is about 55% of that of monolithic chips. The gap between the monolithic and multi-chiplet costs seems to only slightly increase because of the conservative estimation of the model regarding the bonding and packaging multi-chiplet yield. Nevertheless, the model confirms the significant savings of chiplet-based approaches and puts our performance analysis in perspective.

III. Experimental Methodology

A. System Configuration

Our microarchitectural simulation offers detailed modeling of the memory subsystem and interconnection network, as explained in Section III-B, and therefore is computationally intensive for large systems. In order to keep the simulation times of our experiments within affordable bounds (tens of hours per simulation point), the modeled systems are scaled down by a quarter of a real one. A full-scale AMD Zen4C chiplet contains 16 cores, as many considered in our cost analysis of Section II-D[1]. As a consequence, our performance

[1]A yield and cost analysis of a scaled down chiplet would not make sense as the size of the chiplets would be small, and so would be the size of the monolithic chip making it too small to break down into chiplets.

TABLE I: System Configuration[1]

System	
Chiplets	4 chiplets[1]
Cores and Caches	
Cores	4 cores[1] / chiplet, out-of-order, 3.2 GHz
TLB	I-TLB: 512-entry, 4-way, 1 cycle latency D-TLB: 512-entry, 4-way, 1 cycle latency
L1 Cache	L1-I: Private, 32KB, 4-way, 2 cycle access latency L1-D: Private, 32KB, 4-way, 2 cycle access latency
L2 Cache	Private, 256KB, 8-way, 4 cycle access latency
L3 Cache	Shared, 1MB/core, 16-way, 12 cycle access latency[2]
Main Memory	
HBM2	1 GB/chiplet, 2 GHz, 4 channels, 128 bits per channel, tCAS-tRCD-tRP: 14-14-14 ns
DDR4	4 GB, 3.2 GHz, 1 channel, 64 bits per channel, tCAS-tRCD-tRP: 22-22-22 ns
Network	
Intra-chiplet	2 GHz, 3-stage router (VA/SA, ST, LT), 2x3 Mesh, 4 VCs per port, credit-based flow control, 256 bit link for data, 154 bit link for control (coherence) traffic, 5 flit buffers, XY Routing [10]
Inter-chiplet	2 GHz, 3-stage router (VA/SA, ST, LT), 2x2 Mesh, passive interposer, 2 to 3 cycle link latency[3], 7 to 9 flit buffers [3]

[1] This configuration is the default setting. The parameter adjustments are detailed in the respective evaluation sections of the sensitivity studies.
[2] L3 access latency is 8 cycles for 2MB, 12 for 4MB, and 15 for 8MB.
[3] Depending on the maximum inter-chiplet link length [28].

TABLE II: Workload Characteristics

Benchmark	Label	Input	LLC MPKI	Footprint (GB)	Assigned to Mixes mix-id$^{\#of instances}$
LLC MPKI 20-40					
pageRank[2]	PRL2	LDBC (100k)	37.41	0.84	$1^3,2^3,3^2,4^3,5^2,6^3,7^3,8^1$
mcf[1]	MCF	Default	34.01	0.45	$2^2,6^3,8^2$
graphColoring[2]	GCL2	LDBC (100k)	30.70	0.45	$1^1,2^1,4^1,5^2,7^1,8^2$
graphColoring[2]	GCL3	LDBC (10k)	21.26	0.09	$1^2,2^1,3^2,6^2,8^1$
Random Access Workload[3]	RAND	N=30, M=1000, chunk=1024	20.83	0.70	$1^1,2^3,3^1,4^2,6^2,7^1,8^1$
LLC MPKI 10-20					
connectedComp[2]	CCL3	LDBC (10k)	19.33	0.09	$1^3,2^2,3^1,4^3,5^1,6^2,7^2,8^1$
lbm[1]	LBM	Default	18.19	0.40	$3^1,7^2,8^1$
BFS[2]	BFSCR	CA RoadNet	17.25	0.64	$1^1,2^1,3^1,4^2,5^3,7^2,8^1$
fotonik3d[1]	FOTO	Default	17.07	0.59	$1^1,4^1$
pageRank[2]	PRL3	LDBC (10k)	13.96	0.09	$2^1,4^1,5^1$
xalancbmk[1]	XAL	Default	13.62	0.16	$1^1,2^1,3^2,4^1,6^2,7^3,8^1$
blender[1]	BLEN	Default	12.78	0.08	$2^1,4^1,5^1$
shortestPath[2]	SPCR	CA RoadNet	12.30	0.64	$1^1,3^2,5^1,7^1,8^1$
XSBench[4]	XSB	XXL	11.11	0.37	$5^1,8^1$
graphColoring[2]	GCCR	CA RoadNet	10.69	0.63	$1^1,3^1,5^2,6^1,8^1$
LLC MPKI 0-10					
parest[1]	PAR	Default	8.54	0.05	3^1
roms[1]	ROMS	Default	7.58	0.25	8^1
triangleCount[2]	TCL2	LDBC (100k)	6.24	0.55	$6^1,8^1$
graphColoring[2]	GCL1	LDBC (1000k)	5.92	0.29	$1^1,4^1,7^1$
pageRank[2]	PRKR	Knowledge Repo	4.56	0.30	$3^1,5^1$
omnetpp[1]	OMN	Default	4.53	0.16	5^1
BFS[2]	BFSL1	LDBC (1000k)	2.71	0.98	3^1

[1] SPEC CPU 2017 [22], [2] GraphBIG [20], [3] GUPS [24], [4] XSBench [30]

analysis considers chiplets scaled to be a quarter of a chiplet AMD Zen4C or Intel Sapphire Rapids chiplet and as such they contain a quarter of the number of cores and connect to a quarter of HBM channels, as shown in Table I. Moreover, the L2 and L3 caches are undersized in order to put more pressure on the memory system and increase LLC misses per kilo instructions (MPKI), which is otherwise difficult to achieve when simulating systems for only a few billion instructions.

Based on the scaled down chiplet size ($16.5\,\mathrm{mm}^2$[2]), the microbump budget is calculated to be proportional to the number of cores it includes. In addition, the following parameters were taken into account for calculating the microbump budget: (i) a microbump pitch of $45\,\mu\mathrm{m}$, (ii) reserving 40% of the microbumps for power. Then, the number of microbumps available for data were allocated for (i) connecting to the HBM channels, (ii) one bidirectional link to the IO chiplet, (iii) multiple bidirectional links to the other CPU chiplets. Then, the width of the links to IO and CPU chiplets, as well as the total number of links to other CPU chiplets, were adjusted to fit the microbump budget. Finally, the latency of the inter-chiplet links was measured to be 2 or 3 (NoC) clock cycles considering the chiplet's dimensions and the latency of the links on the silicon interposer similar to [8], [28].

B. Simulation Setup

BZSim simulator was used for our experiments [29], extended to model memory system and interconnects of chiplet-based chips. BZSim is based on ZSim simulator [25] integrated with BookSim2 [15] for cycle-accurate intra- and inter-chiplet network modeling, enhanced with a technique to detect and skip simulation of low contention traffic in order to speed up simulation times. BZSim offers microarchitectural simulations with detailed (cycle-accurate) interconnect modeling at an order of magnitude faster simulation speeds compared to GEM5, enabling multi-billion instruction experiments within reasonable times [29]. DRAMSim3 [17] was used for cycle-accurate DRAM modeling and CACTI [31] for estimating cache access times.

The system treats all HBM and external DDR memory as part of a unified flat address space. The virtual memory system was implemented based on HSCC [18]. The cores are configurable with translation lookaside buffers (TLBs) for both instructions and data, as well as with page table walkers (PTWs). Additionally, the memory management modules include a distance-aware allocation policy. This policy allocates pages to the HBM in the chiplet where they are first accessed. If pages are unavailable in the nearest HBM, they are allocated in the next neighboring HBM or in the external DDR.

C. Workloads

We use mixes of multi-programmed workloads from the SPEC CPU2017 benchmark suite [22] (the eight with the highest MPKI), GraphBIG [20], Random access workload from the GUPS suite [24] and XSBench [30] in our experiments. For the SPEC CPU2017 and GraphBIG benchmarks, we use Simpoints [23] to select a representative slice of one billion instructions. We have chosen 22 different workloads, detailed in Table II, and created random multi-programmed mixes of 16 applications designed to run on a system with 16 cores. Each mix of applications has a minimum total memory footprint of 7 GB and a geometric mean LLC MPKI of at least 11. To scale these mixes for systems with 32 or 64 cores, we replicate the 16-application mix twice for the 32-core system and four times for the 64-core system. All experiments run with an average of 125 million instructions per core warm-up period, where memory allocation is enabled, followed by an average of 250 million instructions per core of detailed simulation.

[2]Calculated based on Zen4 after scaling down L2 and L3 sizes proportional to the capacity indicated in Table I.

D. Evaluated Systems

We evaluate three distinct systems as follows:

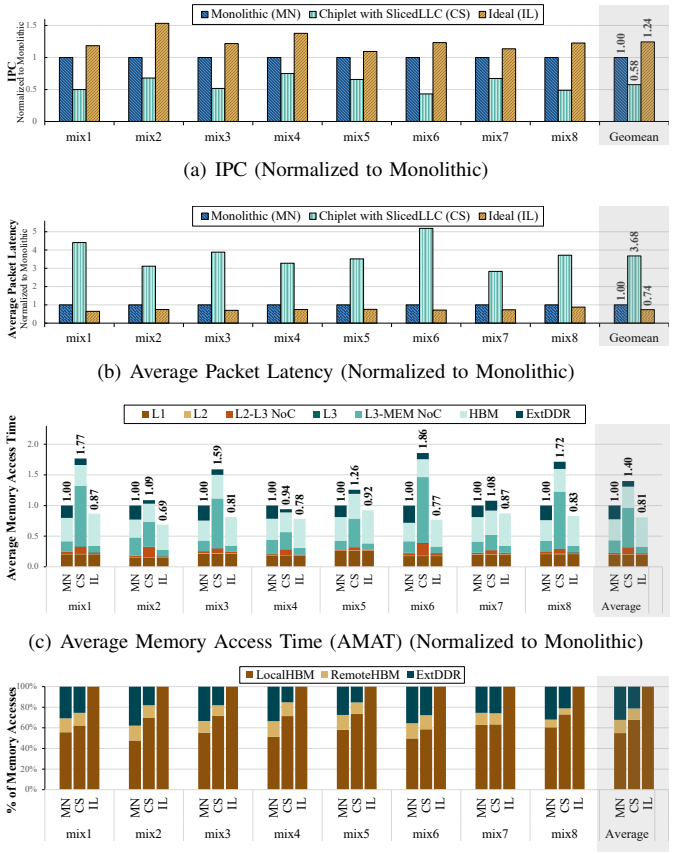
1) **Chiplet-based System (CS)**: A multi-chiplet system with sliced LLC is the focus of our evaluation. The default configuration (depicted in Table I) consists of 4 chiplets, each with 4 cores, integrated on a passive interposer with one LLC slice per chiplet, 256 bit NoC data-links, 4 HBM channels per chiplet, one link to IO chiplet, and one channel to external DDR. The above parameters change in the various sensitivity analyses of the evaluation. One variation of this design is to use chiplets with private, rather than sliced, LLC, denoted as **CP**.
2) **Monolithic (MN)**: A monolithic multicore matching the CS characteristics. Similarly, the default monolithic configuration is a 16-core system with sliced LLC in 4 parts and 256 bit NoC data-links, 16 HBM channels and one external DDR channel.
3) **Ideal (IL)**: An ideal chiplet-based system with the ideal scenario where an LLC miss is always served by the closest HBM channel, assuming the local HBM has infinite capacity, so memory allocation occurs solely within this local HBM. As a result, all memory requests remain local to the chiplet, eliminating the additional latency associated with accessing remote HBM or external DDR.

IV. Performance Evaluation

The performance of chiplet-based systems is evaluated and their overheads with respect to monolithic counterparts are measured. System performance is measured in terms of Instructions Per Cycle (IPC). The Average Memory Access Time (AMAT) is also measured and broken down to: (i) the access time for each cache level, (ii) the Network-on-Chip (NoC) latency between each level (L2-L3 and L3-DRAM), and (iii) the DRAM access time. Furthermore, the Average Packet Latency (APL) and the percentage of accesses to local and remote HBM nodes, as well as to external DDR are reported.

The performance evaluation is structured as follows: first, the default chiplet-based system is compared against the monolithic and ideal systems. Subsequently, a sensitivity analysis of the system size is conducted to examine how performance scales as the number of chiplets increases. Next, the impact of chiplet granularity is explored by analyzing different chiplet sizes (i.e., cores per chiplet) while keeping the system size fixed. Then, the performance of an alternative LLC organization for chiplet-based designs (private LLC per chiplet) is evaluated in comparison with the default sliced LLC. Next, a sensitivity analysis of the intra-chiplet NoC data bandwidth (datapath) is performed. Finally, the impact of passive versus minimally active interposer is measured.

Multi-Chiplet vs. Monolithic vs. Ideal: Figure 4 shows, per mix of programs, the average performance (IPC), average packet latency, average memory access time, and the breakdown of DRAM accesses for the default configuration of a 4-chiplet system as well as for the equivalent 16-core monolithic and ideal systems. The chiplet-based system is able to maintain



(a) IPC (Normalized to Monolithic)

(b) Average Packet Latency (Normalized to Monolithic)

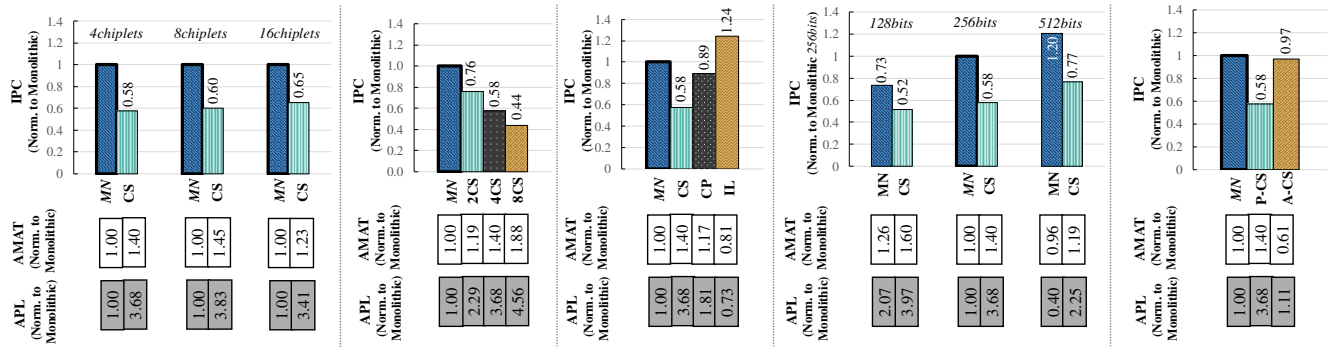
(c) Average Memory Access Time (AMAT) (Normalized to Monolithic)

(d) Memory Accesses

Fig. 4: Multi-Chiplet vs. Monolithic vs. Ideal

only 43%-75% of the monolithic performance and on average 58%, as shown in Figure 4(a). Compared to the ideal system, which is 24% faster than the monolithic, and its LLC misses always go to the closest HBM, the chiplet-based system is 54% slower. The performance overhead of chiplet-based system versus the monolithic is not explained by just observing AMAT, which is on average 40% higher than monolithic. A more detailed look in Figure 4(c) reveals that the longer monolithic AMAT is due to slow external DDR accesses, rather than longer NoC and cache access latency, which is more performance critical and hence puts a heavier toll on chiplet-based performance. In fact, Figure 4(b) confirms the 3.7× longer packet latency of chiplet-based systems compared to monolithic. Finally, it is interesting to analyze the primary source of the performance overheads in the chiplet-based system, which is a fraction of accesses to data placed remotely. Figure 4(d) shows that on average 19% of the data accesses are to the external DDR and 10% of the accesses are to a remote HBM node. For a chiplet-based system, as opposed to a monolithic, all these accesses involve slow (due to limited bandwidth and longer latency) inter-chiplet communication.

Sensitivity Analysis on System Size: An interesting sensitivity analysis is with respect to the system size. Chiplet-based chips are meant to scale better to larger systems in terms of cost. However, it is unclear how their performance overheads



(a) System Size (b) Chiplet Size (c) Sliced vs Private LLC (d) NoC Datapath Width (e) Active vs Passive Interposer

* Geomean values of 8 mixes are shown, normalized to **MN**, as indicated by the thicker outline of the bars; AMAT: Average Memory Access Time; APL: Average Packet Latency

Fig. 5: Sensitivity analyses on system size, chiplet size, LLC organization, NoC datapath width, and interposer type. **MN**: Monolithic, **CS** or **CP**: Chiplet-based with Sliced or Private LLC, **IL**: Ideal, **P** or **A**: passive or active interposer.

change when system size increases while keeping the chiplet size constant. The performance of systems with 4, 8, and 16 CPU chiplets, i.e., 16, 32, and 64 cores, respectively, are evaluated. As shown in Figure 5(a), as the system size increases, the performance gap between chiplet-based systems and their respective monolithic of the same size remains stable or even slightly reduces, ranging from 58% to 65%. This is attributed primarily to the distance-aware data placement, which allows the distance of remote accesses to remain relatively stable.

Sensitivity Analysis on Chiplet Size: Next, a sensitivity analysis on the chiplet size, i.e., the number of cores included in a chiplet, is performed for chiplets of 2 (2CS), 4 (4CS), and 8 cores (8CS) per chiplet on a 16-core system. Figure 5(b) shows the performance of these systems. As expected, performance reduces as the system is disintegrated to a larger number of chiplets. More precisely, 2, 4, and 8 chiplet systems offer 76%, 58%, and 44% of the monolithic IPC, respectively, which is reflected in their AMAT and APL measurements.

LLC Configuration Analysis - Sliced vs. Private: A chiplet-based system with sliced LLC would need to go to a remote chiplet to serve an L2 miss accessing a remote LLC slice if the memory address is mapped to the memory region of another chiplet. An interesting question arises as of the benefit of designing chiplets each with a private LLC. Such private LLC would store cache lines from the entire address space, regardless of whether the memory is mapped to local or remote DRAM (HBM and external DDR). Figure 5(c) depicts the results of this comparison. Chiplets with sliced LLC reduce IPC versus monolithic by 42%, while chiplets with private LLC reduce it by only 11%. A significant reduction in average packet latency of about 50% is achieved by using private LLC because, with this organization, L2 misses do not need to go off-chiplet, as opposed to LLC misses, which are fewer, and may need remote accesses. On the contrary, chiplets with sliced LLC may need remote accesses to serve L2 misses but not for LLC misses. This is also reflected in the AMAT, which is improved by 16% when using chiplets with private LLC compared to chiplets with sliced LLC.

Sensitivity Analysis on NoC Datapath Width: The performance impact of the NoC data-link bandwidth is analyzed for chiplet-based and monolithic systems. The intra-chiplet and monolithic NoC datapath varies from a quarter of a cache line (128 bits), to a full cache line (512 bits). It is worth noting that the width of inter-chiplet links remains constant as defined by the available microbump budget of the chiplets (64 bits). Figure 5(d) shows the IPC, AMAT and average packet latency of the three design points for chiplet-based and monolithic chips normalized to the 256-bit monolithic. It can be observed that wider (intra-chiplet) NoC links improve performance, although the gap with the respective monolithic does not follow a specific trend. Finally, as expected, AMAT improves for wider NoC datapaths, while at the same time, the gap between monolithic and chiplet-based average packet latency increases because the bottleneck of narrower inter-chiplet links is exacerbated.

Sensitivity Analysis on Active vs. Passive Interposer: The last design parameter explored in our study is the use of minimally active (A-CS) versus passive interposer (P-CS). As illustrated in Figure 5(e), a minimally active interposer increases both throughput and latency of inter-chiplet links because it can pipeline them. The performance impact of this is significant as it recovers most of the performance overhead on chiplet-based systems. That comes, however, at a higher system cost due to the lower yield of active interposers.

V. Conclusions

Semiconductor technology has difficulty scaling the integrated resources on a single die because monolithic chips have poor yield, leading to excessive costs. Multi-chiplet chips offer a cheaper alternative as they have a higher yield, but come with certain performance overheads stemming from their NUMA memory system and inter-chiplet interconnection bottlenecks. This paper analyzed these performance overheads. In particular, our study reveals that although chiplet-based chips reduce system costs by almost half compared to monolithic, they give away about a third of the monolithic performance. Our work further showed that part of this performance overhead can be regained with specific design choices. More specifically, designing chiplets with a private LLC improves performance by about 30%. Moreover, chiplet-based systems with active interposers are only 3% shy of the monolithic performance.

ACKNOWLEDGMENT

This work was supported by the Swedish Foundation for Strategic Research (contract number CHI19-0048) under the PRIDE project.

REFERENCES

[1] I. Advanced Micro Devices, "AMD CDNA 3 White Paper," 2023. [Online]. Available: https://www.amd.com/content/dam/amd/en/documents/instinct-tech-docs/white-papers/amd-cdna-3-white-paper.pdf

[2] N. Beck, S. White, M. Paraschou, and S. Naffziger, "Zeppelin: An SoC for Multichip Architectures," in *2018 IEEE International Solid-State Circuits Conference (ISSCC)*, Feb 2018, pp. 40–42.

[3] K. Bridge, "NUMA Support - Win32 Apps," May 2022. [Online]. Available: https://learn.microsoft.com/en-us/windows/win32/procthread/numa-support

[4] T. Burd, N. Beck, S. White, M. Paraschou, N. Kalyanasundharam, G. Donley *et al.*, "Zeppelin: An SoC for Multichip Architectures," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 133–143, 2019.

[5] A. Chadd, "FreeBSD Manual Pages," Oct 2018. [Online]. Available: https://man.freebsd.org/cgi/man.cgi?query=numa&sektion=4&manpath=FreeBSD%2B14.0-RELEASE%2Band%2BPorts

[6] J. Choquette, W. Gandhi, O. Giroux, N. Stam, and R. Krashinsky, "NVIDIA A100 Tensor Core GPU: Performance and Innovation," *IEEE Micro*, vol. 41, no. 2, pp. 29–35, 2021.

[7] N. CORPORATION, "SX-Aurora TSUBASA Architecture Guide Revision 1.1," 2018. [Online]. Available: https://sxauroratsubasa.sakura.ne.jp/documents/guide/pdfs/Aurora_ISA_guide.pdf

[8] A. Coskun, F. Eris, A. Joshi, A. B. Kahng, Y. Ma, and V. Srinivas, "A Cross-Layer Methodology for Design and Optimization of Networks in 2.5D Systems," in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2018, pp. 1–8.

[9] F. Dahlgren and J. Torrellas, "Cache-Only Memory Architectures," *Computer*, vol. 32, no. 6, pp. 72–79, 1999.

[10] W. J. Dally and B. P. Towles, *Principles and Practices of Interconnection Networks*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.

[11] Y. Feng and K. Ma, "Chiplet Actuary: A Quantitative Cost Model and Multi-Chiplet Architecture Exploration," in *Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC)*, Jul 2022, p. 121–126.

[12] Y. Feng, D. Xiang, and K. Ma, "A Scalable Methodology for Designing Efficient Interconnection Network of Chiplets," in *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, Feb 2023, pp. 1059–1071.

[13] Y. Feng, D. Xiang, and K. Ma, "Heterogeneous Die-to-Die Interfaces: Enabling More Flexible Chiplet Interconnection Systems," in *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Oct 2023, p. 930–943.

[14] N. E. Jerger, A. Kannan, Z. Li, and G. H. Loh, "NoC Architectures for Silicon Interposer Systems: Why pay for more wires when you can get them (from your interposer) for free?" in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2014, pp. 458–470.

[15] N. Jiang, D. U. Becker, G. Michelogiannakis, J. Balfour, B. Towles, D. E. Shaw *et al.*, "A Detailed and Flexible Cycle-accurate Network-on-Chip Simulator," in *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Apr 2013, pp. 86–96.

[16] A. Kannan, N. E. Jerger, and G. H. Loh, "Enabling Interposer-based Disintegration of Multi-core Processors," in *2015 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2015, pp. 546–558.

[17] S. Li, Z. Yang, D. Reddy, A. Srivastava, and B. Jacob, "DRAMsim3: A Cycle-Accurate, Thermal-Capable DRAM Simulator," *IEEE Computer Architecture Letters*, vol. 19, no. 2, pp. 106–109, 2020.

[18] H. Liu, Y. Chen, X. Liao, H. Jin, B. He, L. Zheng *et al.*, "Hardware/Software Cooperative Caching for Hybrid DRAM/NVM Memory Architectures," in *Proceedings of the International Conference on Supercomputing (ICS)*, Jun 2017, pp. 26:1–26:10.

[19] S. Naffziger, N. Beck, T. Burd, K. Lepak, G. H. Loh, M. Subramony *et al.*, "Pioneering Chiplet Technology and Design for the AMD EPYC™ and Ryzen™ Processor Families : Industrial Product," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, Jun 2021, pp. 57–70.

[20] L. Nai, Y. Xia, I. G. Tanase, H. Kim, and C.-Y. Lin, "GraphBIG: Understanding Graph Computing in the Context of Industrial Solutions," in *SC '15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Nov 2015, pp. 1–12.

[21] N. Nassif, A. O. Munch, C. L. Molnar, G. Pasdast, S. V. Lyer, Z. Yang *et al.*, "Sapphire Rapids: The Next-Generation Intel Xeon Scalable Processor," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, Feb 2022, pp. 44–46.

[22] R. Panda, S. Song, J. Dean, and L. K. John, "Wait of a Decade: Did SPEC CPU 2017 Broaden the Performance Horizon?" in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2018, pp. 271–282.

[23] E. Perelman, G. Hamerly, and B. Calder, "Picking Statistically Valid and Early Simulation Points," in *2003 12th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Sep 2003, pp. 244–255.

[24] S. J. Plimpton, R. Brightwell, C. Vaughan, K. Underwood, and M. Davis, "A Simple Synchronous Distributed-Memory Algorithm for the HPCC RandomAccess Benchmark," in *2006 IEEE International Conference on Cluster Computing (CLUSTER)*, Sep 2006, pp. 1–7.

[25] D. Sanchez and C. Kozyrakis, "ZSim: Fast and Accurate Microarchitectural Simulation of Thousand-core Systems," in *Proceedings of the 40th Annual International Symposium on Computer Architecture (ISCA)*, Jun 2013, p. 475–486.

[26] K. Sarcar, "What is NUMA?" Nov 1999. [Online]. Available: https://www.kernel.org/doc/html/v5.4/vm/numa.html

[27] D. Stow, I. Akgun, and Y. Xie, "Investigation of Cost-Optimal Network-on-Chip for Passive and Active Interposer Systems," in *2019 ACM/IEEE International Workshop on System Level Interconnect Prediction (SLIP)*, Jun 2019, pp. 1–8.

[28] D. Stow, Y. Xie, T. Siddiqua, and G. H. Loh, "Cost-Effective Design of Scalable High-performance Systems Using Active and Passive Interposers," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2017, pp. 728–735.

[29] P. Strikos, A. Ejaz, and I. Sourdis, "BZSim: Fast, Large-scale Microarchitectural Simulation with Detailed Interconnect Modeling," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, May 2024.

[30] J. Tramm, A. Siegel, T. Islam, and M. Schulz, "XSBench - The development and verification of a performance abstraction for Monte Carlo reactor analysis," in *Proceedings of the International Conference on Physics of Reactors (PHYSOR)*, Sep 2014.

[31] S. J. Wilton and N. P. Jouppi, "CACTI: An Enhanced Cache Access and Cycle Time Model," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 5, pp. 677–688, 1996.

[32] Y. Yarom, Q. Ge, F. Liu, R. B. Lee, and G. Heiser, "Mapping the Intel Last-Level Cache," *Cryptology ePrint Archive*, 2015.

[33] Z. Zhang and J. Torrellas, "Reducing Remote Conflict Misses: NUMA with Remote Cache versus COMA," in *Proceedings Third International Symposium on High-Performance Computer Architecture (HPCA)*, Feb 1997, pp. 272–281.