# Survey of Network-on-Chip (NoC) for Heterogeneous Multicore Systems

Siamak Biglari*, Farahnaz Hosseini*, Aadesh Upadhyay, Hui Zhao
Department of Computer Science and Engineering, University of North Texas
{siamakbiglariardebili, farahnazhosseini, aadeshupadhyay}@my.unt.edu, {hui.zhao}@unt.edu,

*Abstract*—**In recent years, Network-on-Chip (NoC) has emerged as a promising solution for addressing a critical performance bottleneck encountered in designing large-scale multicore systems, i.e., data communication. With advancements in chip manufacturing technologies and the increasing complexity of system designs, the task of designing the communication subsystems has become increasingly challenging. The emergence of hardware accelerators, such as GPUs, FPGAs and ASICs, together with heterogeneous system integration of the CPUs and the accelerators creates new challenges in NoC design. Conventional NoC architectures developed for CPU-based multicore systems are not able to satisfy the traffic demands of heterogeneous systems. In recent years, numerous research efforts have been dedicated to exploring the various aspects of NoC design in hardware accelerators and heterogeneous systems. However, there is a need for a comprehensive understanding of the current state-of-the-art research in this emerging research area. This paper aims to provide a summary of research work conducted in heterogeneous NoC design. Through this survey, we aim to present a comprehensive overview of the current related research, highlighting key findings, challenges, and future directions in this field.**

*Index Terms*—**Network-on-Chip, NoC, Heterogeneous System, Multicore System, Hardware Accelerator, GPU NoC, FPGA NoC, interposer NoC.**

## 1. Introduction

With the slowdown of Moore's law, new computing paradigms are being explored to keep the performance growing. Heterogeneous computing promises a new solution to this need. Heterogeneous systems are computing architectures that integrate different types of processors, such as CPUs, GPUs, FPGAs, and other specialized accelerators, to leverage their unique strengths and optimize performance for various tasks. This approach has become a hot trend following the slowdown of Moore's Law and the end of Dennard scaling. Leading chip vendors like NVIDIA, AMD, Intel, and others have invested in such systems to tackle the nonstop growth in demand for processing diverse workloads, particularly in data-intensive applications like AI and ML, to meet the needs of these state-of-the-art areas [1].

In heterogeneous systems, communications occur when multiple types of processors, compute nodes, and memory nodes share data. Because of this, on-chip communication must support the data movement in a timely and cost-efficient way. However, the current on-chip networks can barely meet the latency and bandwidth requests from applications in today's big data era. On-chip communication becomes a performance bottleneck significantly impacting a processor chip's overall performance. To address this issue, Network-on-Chip (NoC) architectures have been introduced as a promising solution.

NoC provides an efficient and scalable communication framework between different components within a chip . A good NoC design is critical as it can reduce latency, improve bandwidth, and increase power efficiency. By effectively managing data traffic and minimizing congestion [2], NoC architectures are mostly based on the idea of packet-switched networks, where routers use different functionalities such as simple switching and intelligent routing to route data packets in and around the network. Routers have limited area and power, but still require faster data rates [3]. Data buffering is unnecessary in circuit-switching networks since the transmission path is reserved in advance. In contrast, for packet-switching networks, routers need to be saved in buffers to avoid packet dropping.

NoCs were originally designed for CPU-based systems, with many research efforts focusing on CPU NoC design. However, the rise of heterogeneous multicore systems presents new challenges. Conventional NoC architectures struggle to meet the demands of accelerators like GPUs, due to their higher bandwidth requirements, tighter power budgets, and different traffic patterns. As a result, designing an efficient NoC for heterogeneous systems is particularly challenging. In heterogeneous systems that consist of CPUs and accelerators such as GPUs and FPGAs, each type of processing core is designed to target specific types of computing tasks. Therefore, they exhibit different traffic characteristics, resulting in divergent requirements for support from the NoC architectures [12]. Conventional NoC designs may not adequately address the specific needs of different traffic patterns and behaviors of these diverse accelerators, creating a demand for tailored NoC solutions.

This survey aims to review recent work on specific design techniques for heterogeneous NoCs in the context of GPUs, FPGAs, and other accelerated systems. We seek to highlight the importance and potential of heterogeneous system NoC design, help researchers develop a perspective of this research area, and also evaluate future trends. NoC architectures are characterized by the type of accelerators in a heterogeneous system. We examine several recently proposed NoC architec-

*Both authors contributed equally to this work.

TABLE 1: Comparison of GPU NoC Architectures

| Name | Description | Method | Energy | NoC Traffic | Performance | Cost | Reference |
|---|---|---|---|---|---|---|---|
| Checkerboard | reduce NoC area by proposing a "checkerboard" NoC | full-routers and half-routers with limited connectivity | - | Increasing DRAM nodes injection | Increased | Reduced | [4] |
| On-chip network for Efficient Training | Energy-efficient heterogeneous manycore systems for CNN training. | hybrid NoC architecture consists of both wireline and wireless links | 25% EDP saving | 1.8× reduction in network latency | Improves network throughput by a factor of 2.2 for training CNNs | Increased by 1.82% for a die with 20x20 mm | [5] |
| Asymmetric NoC | Asymmetric NoC design to enhances GPU energy efficiency | providing one network for L1-to-L2 communication and a second for L2-to-L1 traffic | 88% energy-delay-product Reduction | reduction in arbitration latency | comparable performance | - | [6] |
| Adapt-NoC | Adaptive NoC architecture with reinforcement learning control | Reinforcement learning-based adaptive NoC topology allocation | 53% NoC energy-efficiency | 34% latency reduction | 10% execution time reduction | 14% less area | [2] |
| HRCnet | Heterogeneous Ring-Chain network for the GPGPU reply network | eliminates conflicts in the network by proposing a ring-similar topology | 60% Reduction | - | 45% Improvement | 42% Reduction | [7] |
| Bandwidth-Efficient On-Chip Interconnect | analyze the communication demands of typical GPGPU applications | propose VC monopolizing and partitioning schemes | - | - | Improved performance | - | [8] |
| Packet Coalescing | Packet Coalescing Exploiting Data Redundancy in GPGPU | Coalesce redundant cache block packets without increasing size | - | reducing average memory access time | 15% Improvement | 0.28% of total cache size of 56 SM | [9] |
| CD-Xbar | Efficient converge-diverge crossbar network for scalable GPUs | converge-diverge crossbar (CD-Xbar) network with round-robin routing and topology-aware CTA scheduling. | 48.5% Reduction | - | 13.9% Improvement | 52.5% Reduction | [10] |
| DUB | Dynamic Underclocking and Bypassing in NoCs | enable bypassing retimer flops and routers while underclocking the NoC frequency | 26% Power Reduction | - | 3% Reduction | - | [11] |

tures, aiming to highlight the transformative potential of NoC development. In the following sections, NoC architectures are reviewed based on the accelerator/processor types. Section 2 reviews research in GPU NoCs; section 3 reviews research in FPGA NoCs; and in sections 4 and 5 we provide reviews on NoC architecture in ASIC accelerators and interposers respectively. Finally, we discuss the prospect of NoC design in section 6.

## 2. GPU NoC Architectures

As a hardware accelerator, GPUs have massive parallel computing capability because they can execute thousands of threads simultaneously. The GPU architecture and memory organization is different from that of CPUs. On-chip GPU traffic happens between the many processing cores and a few memory controllers [4], [13]. Several recent research proposed NoC architectures specifically tailored for this unique traffic pattern and we have explained some of these researches in Table 1. Leveraging the unique GPU traffic pattern, a checkerboard NoC was proposed to reduce the NoC area by employing both full-routers and half-routers [4]. In this design, full-routers allow traffic of all directions while half-routers put limits on the directions allowed. The proposed design can achieve similar performance as in a network containing only full-routers. However, power and area savings can be achieved through the half-routers deployed. Ziabari et al. proposed an asymmetric NoC architecture to achieve energy-efficient communication for GPUs [6]. Less resource is allocated to the request network which has lighter traffic compared to the reply network.

Several works have been done in NoC traffic area [14]–[20]. Zhao et al. [21] proposed a conflict-free mesh NoC for the GPUs. The proposed NoC removes conflicts among different columns by deploying an exclusive subnet. A heterogeneous Ring Chain Network was proposed for GPUs to reduce both latency and contention through unidirectional channels [7]. Jang et al. [8], have proposed efficient Network on-chip designs to meet the communication demands of typical GPGPU applications. Since the GPU traffic is majorly between the processing cores and memory controllers, the location of memory controllers plays a big role in impacting the traffic directions. This work evaluated different memory controller placement schemes and also developed routing algorithms accordingly to reduce traffic conflicts. Packet coalescing is another technique that takes advantage of the GPU traffic patterns [9]. Based on the observation that there are a lot of multicasting packets in the GPU network, packets are coalesced together to reduce the cost and energy consumption.

A major design challenge with GPU accelerators is their high power consumption and NoCs account for a large portion of the total power budget. Several research works have been proposed to reduce the NoC power consumption. Zhao et al. [10] proposed a converge-diverge crossbar (CD-Xbar) network with round-robin routing and topology-aware concurrent thread array (CTA) scheduling. This network includes two types of crossbars, a local crossbar, and a global crossbar. A local crossbar converges input ports from the GPU stream multiprocessors into so-called converged ports. The global crossbar diverges these converged ports to the last-level cache (LLC) slices and memory controllers. Different routing paths are provided by CD-Xbar through the converged ports. Round-robin routing and topology-aware CTA create balanced network traffic among the converged ports with a local crossbar and across crossbars. Compared with mesh, CD-Xbar has the same bisection bandwidth. However, CD-Xbar significantly reduces the active silicon area and power consumption of the NoC.

To save power, Bharadwaj et al. [11], proposed DUB, Dynamic Underclocking, and bypassing techniques for solving the problem of adaptivity of heterogeneity by dynamic voltage frequency scaling (DVFS) techniques. They bypass the timer flops and routers while underclocking the NoC frequency to save power at minimal performance loss. In comparison with baseline, their technique can save more power than oracular DVFS techniques. Leveraging optical NoC design techniques, Bashir et al. [22] proposed an energy-efficient and scalable optical interconnect for GPUs. Their technique clusters the components in a GPU and enables them to communicate with each other. They used separate networks for coherence and non-coherence traffic. Their technique can decrease the static power consumption in optical interconnects. Their method modulates the off-chip light source and increases the performance at the same time.

A low-power GPU NoC called Poet was proposed by Cheng et al. [23]. It is a hybrid optical electric NoC for heterogeneous systems containing CPUs and GPUs. Poet makes use of a Reservation-based Single-write Multiple Reader (R-SWMR) for long-distance connections. This architecture not only can reduce power consumption but can also decrease packet latency compared with a mesh network. Alhubail et al. [24] developed a model based on n Strength Pareto Evolutionary Algorithm2 (SPEA2) and obtained a Pareto optimal set that optimizes communication performance and power consumption of the NoC. They designed a heterogeneous mesh-style NoC to connect CPU and GPU processors and provided solutions for three problems: mapping Processing Elements (PEs) to the routers, assigning the number of virtual channels (VC), and assigning the buffer size for each port of a router.

GPUs play an important role in DNN acceleration and they appear in many ML systems [25], [26]. Developing high-performance NoCs in such systems is essential for improving system-level performance. An NoC architecture was proposed by Choi et al. [5] to tackle the challenge of designing energy-efficient CNN training systems using specialized CPU-GPU-based heterogeneous manycore platforms. They first analyzed on-chip traffic patterns during the training of two deep CNN architectures, LeNet and CDBNet, for image classification. Then they developed a hybrid NoC with both wired and wireless links to enhance the performance of CNN training. This hybrid NoC can significantly increase the network throughput and thus improve the CNN training efficiency.

Chen et al. proposed an NoC-based DNN platform as a new accelerator design paradigm [27]. Their design can reduce the off-chip memory accesses through a flexible interconnect that facilitates the exchanges of data between processing elements on the chip. They analyzed different design parameters to implement the NoC-based DNN accelerator. They combined several techniques, such as neuron clustering, random mapping, and XY-routing. After evaluating their design using LeNet, Mo-FileNet, and VGG-16, the NoC-based DNN accelerator is shown to reduce the accesses to off-chip memory and improve the runtime computational flexibility.

Joardar et al [28], have examined the pros and cons of the various design in 3D-enabled heterogeneous manycore systems. They found that designing an appropriate 3D heterogeneous manycore system is a big challenge, because of the presence of multiple types of PEs. Moreover, there exist big differences in the thread-level parallelism of CPUs and GPUs, and as a result, designing the heterogeneous architecture with both CPUs and GPUs is a complicated task. Their work provides helpful insights for developing 3D-enabled heterogeneous manycore systems.

A network prioritization mechanism was proposed by Cai et al. [29] that can effectively coordinate the on-chip traffic. They developed methods to quantify the performance of CPU and GPU applications based on the observation that on-chip traffic from different PEs has various throughput and performance requirements. Therefore, having a naïve or unoptimized traffic mechanism can result in low performance of CPUs and GPUs. They developed techniques that can improve the overall performance of the system without using virtual networks, complex hardware, or misrouting. Blocked packets are rerouted into the network by creating a bubble that breaks the deadlocks. In addition, blocked packets are rerouted using low-complexity mechanisms which can further improve performance.

## 3. FPGA NoC Architectures

Recently, FPGAs, also known as Field-Programmable Gate Arrays, have taken an important role in accelerating emerging applications such as deep neural networks. FPGAs provide a versatile, high-performance computing solution that can be customized for specific computing applications. In this section, we focus on NoC architectures in FPGAs that work as accelerators.

Several works have been done on optimizing FPGA NoC traffic. Srinivasan et al [30] enhance FPGA design tools to optimize both circuit and NoC performance metrics, improving bandwidth utilization without affecting wirelength and critical path delay. González et al [31] improves the HopliteRT NoC design for real-time FPGA systems by introducing priority-based routing, modifying network topology, correcting timing analysis flaws, and achieving a 2x improvement in packet traversal times with minimal hardware costs. Cache memory bottlenecks in NoC and FPGA systems are a crucial problem causes network slowdown and system hang issues. Shelke et al [32] propose using Discrete Wavelet Transform (DWT) with ring topology to compress cache memory, aiming to improve system speed, increase performance, and reduce complexity.

Attia et al [33] evaluate various NoC router architectures and design parameters to achieve high performance and reduced area and power by using minimum and shareable resources. The proposed method efficiently embeds a hard NoC within the FPGA, offering significant communication throughput with minimal area and power overhead. An FPGA-based NoC simulation acceleration framework is proposed by Prasad et al [34] that optimizes resource utilization by embedding NoC router components, such as FIFO buffers and Crossbar switches, into FPGA hard blocks like Block RAM (BRAMs) and DSP48E1 slices.

A novel approach was introduced by Siast et al. [35] to address the challenges of network-on-chip (NoC) design tailored for FPGAs. Named RingNet, this FPGA-oriented NoC aims to meet specific FPGA requirements and characteristics from major manufacturers. RingNet efficiently utilizes FPGA resources by implementing distributed RAM buffers and employing memory-based connections to prevent network congestion and reduce buffer needs. It leverages virtual cut-through switching to enhance FPGA efficiency. RingNet demonstrates consistent throughput, predictable latency, and equitable network access across various FPGA platforms, including Xilinx, Intel, and Lattice. Compared to the AXI4 architecture, RingNet demands fewer resources and supports higher clock frequencies.

Bhanwala et al [36] developed an architecture based on FPGA on the reconfigurable router for NoC applications. The proposed architecture has four channels including east, west, north, and south, and a crossbar switch. The channels consist of, First in First out (FIFO) buffers and multiplexers. They used FIFO buffers to store the data and the input and output which are controlled using multiplexers. All these presented channels are integrated to form the complete router architecture. Their design has been shown to reduce power consumption.

An optimization technique for cache memory was proposed by Shelke et al. [32] to address issues caused by bottleneck link interconnect throughput, which limits performance, creates numerous files in the cache, slows down the network, and sometimes causes the system to hang. The proposed method involves analyzing the Network on Chip (NoC) and field-programmable gate array (FPGA) using ring topology. This paper introduces the use of Discrete Wavelet Transform (DWT) to compress cache memory. This proposed method can improve system speed. In terms of FPGA design, this technique can also reduce the design complexity.

To address the limitations of conventional buses and crossbar interconnects used in FPGA or ASIC, network-on-chip (NoC) technology has been widely adopted. NoC has been proven to be an efficient and effective solution for interconnecting large System-on-Chip (SoC) designs. Joy [37] aims to develop a tunable NoC for FPGA, maximizing the utilization of FPGA resources to incorporate more components within the available space, making it suitable for multipurpose SoC platforms. Additionally, the tunable NoC can adapt to various FPGA devices and multi-core devices, requiring fewer resources and supporting higher clock frequencies while offering improved average latency.

## 4. ASICs NoC Architectures

ASICs are custom-designed processors for specific applications to achieve optimum speed and power consumption. ASICs are widely used in embedded devices. The design of ASICs takes the longest development cycle compared to GPUs and FPGAs and they have no flexibility after design. With the emergence of deep learning neural networks in AI applications, general-purpose processors are unable to process complex DNNs to meet the requirement of throughput, latency, and power budget. In contrast, ASICs become important accelerators for AI applications due to their superior performance and energy efficiency. As a result, developing NoC architectures in ASICs become an important issue.

To solve the problem of optimizing energy efficiency in deep convoluted neural networks (CNN), Krishna et al. [40] proposed a state-of-the-art accelerator, Eyeriss. Minimizing data movement within the CNN is crucial to increase throughput and efficiency. Eyeriss accomplishes this using a spatial architecture comprising 168 elements and employs a dataflow technique known as row stationery (RS). Additionally, it dynamically reconfigures computation mapping to optimize energy efficiency by locally reusing data, thereby reducing costly data movement.

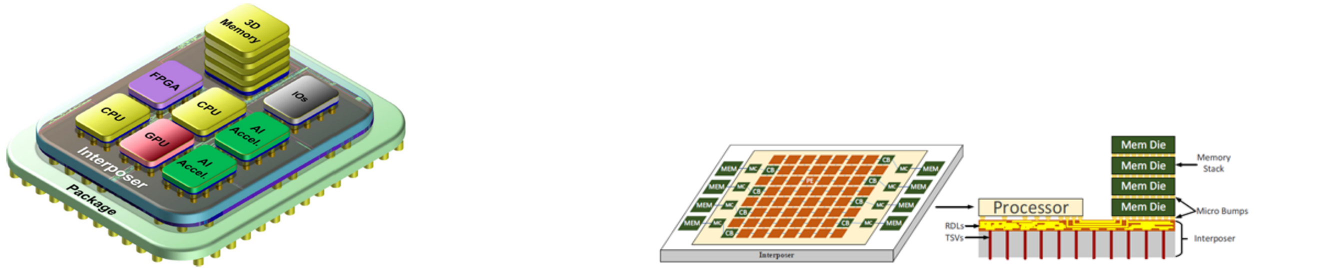
Yang et al [41], introduced a framework, NASAIC that simultaneously identifies multiple DNN architectures. Moreover, it also finds the necessary heterogeneous ASIC accelerator design that satisfies the design specifications while maximizing accuracy. An analytical cost model, MAESTRO, was proposed by Kwon et al. [42] to demystify the complex design space of a DNN accelerator and improve efficiency. The model takes inputs from DNN model descriptions and hardware resource information. In MAESTRO, mapping is depicted using a data-centric representation, enabling the rapid analysis and generation of 20 statistics including latency, energy consumption, and throughput.

To resolve the challenge of improving performance in DNN accelerators within edge devices, Guirado et al. [43] investigated communication flows. Issues such as scalability, dataflow flexibility, and bandwidth directly affect accelerator throughput. Therefore, they developed implementing an interconnect WNoC to address these challenges to enhance DNN accelerator performance. Furthermore, this study qualitatively examines the implications of introducing a novel wireless chip network in this scenario.

Jordar et al [44] have investigated the problems of using multiple normalization layers in convoluted neural networks and the issues about it like reduced accuracy, requiring additional hardware, and producing performance bottlenecks. They proposed a solution named Deep Train, a heterogeneous architecture enabled by Bayesian Optimization methodology. In addition, it also determines the minimum number of normalization operations required for a given CNN. Furthermore, this methodology increases the training speedup by 15 times with no accuracy loss.

To solve the challenge of reducing off-chip memory access in NoC interconnection, Chen et al. [27] introduced DNNoC-sim, a cycle-accurate NoC-based simulator. Firstly, a DNN flattening approach was proposed by the authors to convert various DNN operations into MAC-like operations, supporting convolution and pooling in modern DNN models. Additionally, they introduced a DNN slicing method to evaluate large-scale DNN models on resource-constrained NoC platforms. The evaluation demonstrated a notable decrease in off-chip memory access compared to state-of-the-art DNN models. The paper concludes with a discussion on performance and trade-offs among different design parameters.

To enhance performance in a NoC-based DNN accelerator, Ouyang et al. [45] introduced a multicast mechanism, departing from traditional unicast channels. Their approach



(a) An in-package chiplet system containing different types of PEs and accelerators connected through an interposer. [38]

(b) An abstract cross-section view of an interposer-based heterogeneous multicore system, illustrating the integration of processing elements, cache banks, and memory stacks with the interposer layer. [39]

Fig. 1: Heterogeneous systems connected by an interposer network.

includes a tree-based multicast routing algorithm known for its scalability and reduced packet count in the network. Furthermore, they proposed a router architecture for single flit packets that efficiently transfers flits to multiple destinations in a single process, ensuring high throughput, and low latency, and eliminating head-of-line blocking issues typical of traditional architectures.

Clark [46] et al. proposed an adaptable power management technique that effectively combines power gating and DVFS technique to target both static power and dynamic energy with a SIMO voltage regulator called DozzNoC. Furthermore, the power management system is enhanced by machine learning techniques that predict future traffic load for proactive DVFS mode selection. Also, DozzNoC aids with fast, low-powered, and independently power-gated voltage regulators. Choi [47] et al. have discussed methods to enhance the performance of heterogeneous manycore architectures by designing a hybrid NoC consisting of both wireline and wireless links. Furthermore, they have targeted the resource-intensive backpropagation algorithm which is primarily used as a training method in deep learning. The proposed backpropagation algorithm achieves a substantial improvement of 2x in the network throughput.

To propose a solution to accelerate large-scale neural networks, Firuzan [48] et al developed a reconfigurable NoC architecture. Parallel hardware accelerators are often designed with multi- or many-core systems-on-chip connected by a network-on-chip (NoC). The authors presented reconfigurable network-on-chip architecture for 3D memory-on-logic neural network accelerators. This reconfigurable NoC is able to adapt its topology to the on-chip traffic patterns and can be configured as a tree-like structure to support multicast-based traffic of neural networks. The evaluation of this architecture shows that it can manage multicast-based traffic more effectively than some state-of-the-art topologies, resulting in increased throughput and power efficiency.

## 5. Interposer NoC Architectures

The ongoing increase in data- and compute-intensive applications, such as big data analytics and deep learning, necessitates the development of large-scale chips that offer high computational performance and significant parallelism. SoC complexity and rising silicon costs drive the shift to smaller 'chiplets,' which promise faster SoC construction by integrating various chips (e.g., CPU, GPU, memory) with advanced packaging. In such chiplets systems, the interconnection between different chiplets is implemented through a network in the interposer as shown in Figure 1. Figure 1(a) illustrates the integration of processing elements, cache banks, and memory stacks in an interposer-based processor. A cross-section view of an interposer-based heterogeneous multicore system is shown in Figure 1(b). Several research works have targeted these characteristics and delved deeper into interposer technologies for these chiplet systems [39], [56], [51], [57], [52], [53], [54], [50], [55], some of which we will briefly explain in this section and Table 2.

Li et al. [39] propose a solution to mitigate NoC bottlenecks in silicon interposer-based many-core processors using Equivalent Injection Routers (EIRs). These routers utilize the interposer's wiring to transform few-to-many traffic patterns into many-to-many patterns. Their design example, EquiNox, optimized using N-Queen and Monte Carlo Tree Search methods, shows significant improvements in execution time, energy consumption, and energy-delay product.

2.5D silicon interposer technology which integrates multiple memory stacks with a processor chip has underutilized routing resources and several works have addressed these potential resources [58], [59], [60]. Jerger et al [49] propose an asymmetric NoC architecture that distributes the NoC across both the multi-core chip and the interposer, optimizing the use of available routing capacity. The approach significantly improves system efficiency and performance by separating sub-networks in terms of traffic types, topologies, and other attributes. In Figure 2 we can observe a 2.5D multi-core system with 4 DRAM stacks placed on either side from 2 angles and using different topology.

To solve the challenge of higher manufacturing costs in high-performance systems, Stow et al. [56] examined traditional monolithic 2D SoCs, 2.5D passive interposers, and 2.5D/3D active interposers. They introduced a multi-die core-binning cost model, emphasizing yield improvements achieved through interposer-based partitioning of large multi-core processors. The study demonstrates that both passive and active interposers offer cost-effective integration and fault tolerance, thereby enhancing yield and performance.

A cross-layer co-optimization methodology was presented

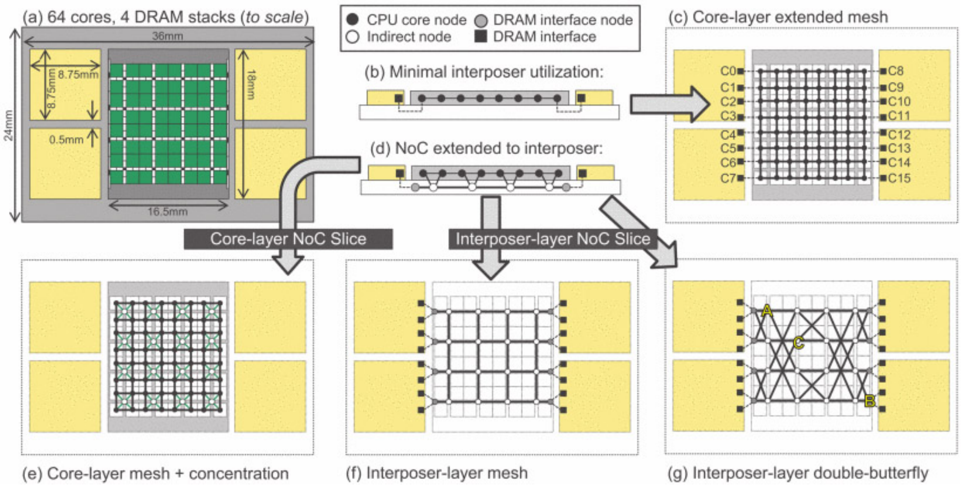

(a) 64 cores, 4 DRAM stacks (to scale)
36mm
24mm
8.75mm
8.75mm
18mm
0.5mm
16.5mm
CPU core node
DRAM interface node
Indirect node
DRAM interface
(b) Minimal interposer utilization:
(c) Core-layer extended mesh
C0
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
C11
C12
C13
C14
C15
(d) NoC extended to interposer:
Core-layer NoC Slice
Interposer-layer NoC Slice
A
C
B
(e) Core-layer mesh + concentration
(f) Interposer-layer mesh
(g) Interposer-layer double-butterfly

Fig. 2: Top view of the 2.5D multi-core system under evaluation, featuring a 64-core CPU chip at the center of the interposer, flanked by four DRAM stacks on either side. (b) Side view illustrating a simple interconnect design that minimizes interposer use. (c) Mesh topology in NoC slice. (d) Side view showing a NoC that is logically divided between the multi-core die and the interposer. (e) Core-layer mesh with concentrated connections to the interposer. (f.g) Concentrated mesh and double butterfly topologies for the interposer NoC [49].

TABLE 2: Comparison of Chiplet NoC Architectures

| Name | Description | Method | Energy | Latency | Performance | Cost | Reference |
|---|---|---|---|---|---|---|---|
| EquiNox | Equivalent Injection Routers optimize NoC performance | transform the few-to-many traffic pattern to many-to-many pattern | 18.9% Reduction | 45.8% Packet Latency Reduction | reduces execution time by 47.7% | die area increased by 4.6% | [39] |
| Reconfigurable NoC | Reconfigurable Interconnection Network of Heterogeneous Chiplets | Reconfigurable network with Kalman Filter for resource prediction | - | Reduced | - | - | [50] |
| Cross-Layer | Cross-layer co-optimization for 2.5D system design | optimize the network topology and chiplet placement and propose gas-station link that enables pipelined interchiplet links in passive interposers | - | - | Increased in comparison with manufacturing cost | Reduced in comparison with same performance | [51] |
| GIA | Reusable General Interposer Architecture for Agile Chiplet Integration | Reusable interposer architecture to reduce costs and accelerate integration | 2.99x power saving | - | 60.92x performance boost | 42% Reduction | [7] |
| Modular Routing | Modular methodology for deadlock-free chiplet integration | turn restrictions for deadlock-free routing to traffic as it flows into or out of the chiplets from the interposer | - | Nearly same as baseline | Nearly same as baseline | overhead is minimal | [52] |
| Neksus | lowers SiP costs and boosts performance with modular chiplet integration | Dedicated interconnect chiplet with mini-chain IP-connection topology | 31% Reduction | - | 28% Improvement | Reduced | [53] |
| Remote Control | Deadlock Avoidance Scheme for Modular Systems-on-Chip | a simple routing oblivious deadlock avoidance scheme based on selective injection-control mechanism | - | 15.49% Improvement | 20% Improvement | 52.5% negligible as compared to the total chiplet area | [54] |
| MemNiSI | MemNiSI optimizes memory network design in 2.5D integration, improving latency and scalability | propose a memory network design to directly connect all the memory modules | - | Up to 15.3% Packet Latency Reduction | - | - | [55] |

by Coskun et al. [51] to address challenges in 2.5-D systems. Their approach integrates homogeneous or heterogeneous chiplets flexibly and cost-effectively. The methodology optimizes network topology and chiplet placement across logical, physical, and circuit layers to improve system performance, lower manufacturing costs, and manage operating temperatures. Additionally, they propose a novel gas-station link for pipelined interchiplet connections within passive interposers. This innovative approach achieves improved performance-cost tradeoffs.

In 2.5D chiplet technology, despite cost savings from chiplet reuse, interposer design, and fabrication introduce high NRE costs, especially for low-volume, application-specific designs. To address this, Li et al. [57] propose a reusable General Interposer Architecture (GIA) to amortize NRE costs and accelerate interposer integration across various chiplet-based systems. The GIA supports both active and passive interposers and is optimized through an end-to-end design automation framework, which configures system assembly, chiplet selection, inter-chiplet network configuration, and placement.

Ensuring correctness, especially avoiding NoC deadlocks, is challenging in chiplet SoC. Yin et al. [52] introduces a modular methodology for deadlock-free routing in multi-chiplet systems on an active silicon interposer. By allowing each chiplet to use its own NoC topology and applying simple turn restrictions to traffic between chiplets and the interposer, the approach ensures high-performance, deadlock-free, and modular SoC construction. Goyal et al. [53] introduce Neksus, a novel architecture designed to reduce System-In-Package (SiP) manufacturing costs, support modular chiplet integration, and leverage interposer properties. Neksus features a dedicated interconnect chiplet that uses a mini-chain IP-connection topology for direct communication, addressing SiP packaging limitations and providing high-bandwidth IP-to-IP communication ideal for bandwidth-intensive mobile applications.

A routing-oblivious deadlock avoidance method, called Remote Control (RC), was proposed by Majumder et al. [54], to address deadlock issues among chiplets in SoCs. Traditional deadlock solutions are not suitable for modular SoC designs that demand customized approaches. The RC scheme employs selective injection control to ensure deadlock freedom and seamless integration of chiplets in modular SoCs. In addressing the integration of CPUs and memories on silicon interposers to improve memory performance, Akgun et al. [55] explored various network topologies. They identified that simple point-to-point connections were inadequate for this purpose. Consequently, they proposed MemNiSI, a memory network design that directly links all memory modules using existing silicon interposer routing resources. Evaluations, involving synthetic and real workloads, demonstrated that MemNiSI effectively reduces average packet latency.

## 6. Discussion and Prospect

This survey paper aims to provide a comprehensive review of state-of-the-art research works conducted in the area of NoC design for heterogeneous systems. NoC has become a promising solution for addressing the critical performance bottleneck encountered in designing large-scale hardware systems. The increasing complexity of system designs has made the task of designing such NoCs increasingly challenging. This survey presented a background of the current research frontier in NoC development, highlighting key findings, challenges, and future directions in this field. The review has demonstrated that the research efforts in this area have significantly contributed to the development of NoC technology. As a result, NoCs are expected to play a crucial role in addressing the challenges of implementing large-scale digital hardware designs in the future.

Although the existing NoC designs have made a significant contribution to removing the performance bottleneck for heterogeneous multicore systems, we believe there is still much room for improvement in the following aspects:

(i) *Taking advantage of special application features.* Emerging applications exhibit new features that demand tailored NoC design. For example, ML and image processing have relaxed accuracy constraints than conventional applications. NoC architectures supporting approximate data delivery can improve the network throughput while still generating acceptable outputs as shown in APPROX-NoC [61]. Another exemplary architecture was proposed by Chen et al. that leverages dataflow reconfigures and maximally reuses data locally to reduce expensive data movement for DNNs [62].

(ii) *Enabling computation/processing in the network.* Contemporary NoCs exhibit temporal and spatial slacks in the form of long periods of idle time or underutilized router resources. With minimal additional logic circuits, the communication layer can perform computation to improve the overall system performance. For example, lightweight processing elements are augmented to NoC routers to compute linear algebra kernels in the SnackNoC [63]. A network architecture was proposed to improve All-Reduce performance through in-network reductions by enabling switches to perform computation [64]. Such active NoC designs can benefit the overall performance and resource utilization but need to carefully balance the tradeoff between performance gain and implementation complexity.

(iii) *Leveraging technology heterogeneity.* Chiplets or interposer-based systems open up more opportunities for developing NoCs using hybrid technologies. Contrary to traditional many-core systems that have limited area and resources, 2.5D or 3D integration provides more resources and relaxed area limitations. Wireless and photonic links can be combined with conventional wirelines to build hybrid NoCs. The challenge lies in how to effectively allocate resources and route the data in order to achieve optimal performance and resource efficiency.

## Acknowledgements

This work was supported in part by NSF grants 2008911, 2046186, and 2051062.

## References

[1] Srikant Bharadwaj and et al. Kite: A family of heterogeneous interposer topologies enabled via accurate interconnect modeling. In *2020 57th DAC*, pages 1–6, 2020.

[2] Hao Zheng and et al. Adapt-noc: A flexible network-on-chip design for heterogeneous manycore architectures. In *2021 HPCA*, pages 723–735, 2021.

[3] Yujie Gao and et al. Traffic-aware energy-efficient hybrid input buffer design for on-chip routers. In *2022 IEEE 15th MCSoC*, pages 395–401, 2022.

[4] Ali Bakhoda and et al. Throughput-effective on-chip networks for manycore accelerators. In *MICRO 2010*, pages 421–432, 2010.

[5] Wonje Choi and et al. On-chip communication network for efficient training of deep convolutional networks on heterogeneous manycore systems. *IEEE Transactions on Computers*, 67(5):672–686, 2018.

[6] Amir Kavyan Ziabari and et al. Asymmetric noc architectures for gpu systems. In *Proceedings of the 9th International Symposium on Networks-on-Chip*, NOCS '15, New York, NY, USA, 2015. ACM.

[7] Xia Zhao and et al. A heterogeneous low-cost and low-latency ring-chain network for gpgpus. In *ICCD 2016*, pages 472–479, 2016.

[8] Hyunjun Jang and et al. Bandwidth-efficient on-chip interconnect designs for gpgpus. In *DAC 2015*, pages 1–6, 2015.

[9] Kyung Hoon Kim and et al. Packet coalescing exploiting data redundancy in gpgpu architectures. In *Proceedings of the International Conference on Supercomputing*, ICS '17, New York, NY, USA, 2017. ACM.

[10] Xia Zhao and et al. Cd-xbar: A converge-diverge crossbar network for high-performance gpus. *IEEE Transactions on Computers*, 68(9):1283–1296, 2019.

[11] Srikant Bharadwaj and et al. Dub: Dynamic underclocking and bypassing in nocs for heterogeneous gpu workloads. In *NOCS 2021*, pages 49–54, 2021.

[12] Yujie Wang. Artificial-intelligence integrated circuits: Comparison of gpu, fpga and asic. *Applied and Computational Engineering*, 4:99–104, 06 2023.

[13] Xianwei Cheng and et al. Amoeba: a coarse grained reconfigurable architecture for dynamic gpu scaling. In *Proceedings of the 34th ACM International Conference on Supercomputing*, ICS '20, New York, NY, USA, 2020. ACM.

[14] Yujie Gao and et al. Traffic-aware energy-efficient hybrid input buffer design for on-chip routers. In *MCSoC 2022*, pages 395–401, 2022.

[15] Qian Gao and et al. Dynamic and traffic-aware medium access control mechanisms for wireless noc architectures. In *ISCAS 2021*, pages 1–5, 2021.

[16] Ebadollah Taheri and et al. Adele: An adaptive congestion-and-energy-aware elevator selection for partially connected 3d nocs. In *DAC 2021*, pages 67–72, 2021.

[17] Siamak Biglari Ardabili and et al. Icla unit: Intra-cluster locality-aware unit to reduce l2 access and noc pressure in gpgpus. *JCSC*, 31(01):2250015, 2022.

[18] Xianwei Cheng and et al. A low-cost and energy-efficient noc architecture for gpgpus. In *ANCS, 2019*, 2019.

[19] Mohammadreza Robaei and et al. Broadcast-based hybrid wired-wireless noc for efficient data transfer in gpu of ce systems. In *IEEE Consumer Electron. Mag. 8(6): 62-67 (2019)*, 2019.

[20] Yuwen Cui and et al. A low-cost conflict-free noc architecture for heterogeneous multicore systems. In *ISVLSI 2020: 300-305*, 2020.

[21] Xia Zhao and et al. A low-cost conflict-free noc for gpgpus. In *DAC 2016*, pages 1–6, 2016.

[22] Janibul Bashir and et al. Gpuopt: Power-efficient photonic network-on-chip for a scalable gpu. *J. Emerg. Technol. Comput. Syst.*, 17(1), sep 2020.

[23] Tao Cheng and et al. Poet: A power efficient hybrid optical noc topology for heterogeneous cpu-gpu systems. In *IECON 2019*, volume 1, pages 3091–3095, 2019.

[24] Lulwah Alhubail and et al. Power and performance optimal noc design for cpu-gpu architecture using formal models. In *DATE 2019*, pages 634–637, 2019.

[25] Xianwei Cheng and et al. Alleviating bottlenecks for dnn execution on gpus via opportunistic computing. In *ISQED 2020*, pages 261–267, 2020.

[26] Khoa Ho and et al. Improving gpu throughput through parallel execution using tensor cores and cuda cores. In *ISVLSI 2022*, pages 223–228, 2022.

[27] Kun-Chih et al. Chen. A noc-based simulator for design and evaluation of deep neural networks. *Microprocessors and Microsystems*, 77:103145, 06 2020.

[28] Biresh Kumar Joardar and et al. Noc-enabled 3d heterogeneous manycore systems for big-data applications. In *ISQED 2022*, pages 1–6, 2022.

[29] Xiangwei Cai and et al. An orchestrated noc prioritization mechanism for heterogeneous cpu-gpu systems. *Integration*, 65, 04 2018.

[30] Srivatsan Srinivasan and et al. Placement optimization for noc-enhanced fpgas. In *FCCM 2023*, pages 41–51, 2023.

[31] Yilian Ribot González and et al. Hoplitert: Real-time noc for fpga. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(11):3650–3661, 2020.

[32] Rohini Shelke and et al. Optimization of memory oriented network-on-chip for fpga. In *ICCCNT 2020*, pages 1–6, 2020.

[33] Sameh Attia and et al. Optimizing fpga-based hard networks-on-chip by minimizing and sharing resources. *Integration*, 63:138–147, 2018.

[34] B. M. Prabhu Prasad and et al. Fpga friendly noc simulation acceleration framework employing the hard blocks. *Computing*, 103(8):1791–1813, aug 2021.

[35] Jakub Siast and et al. Ringnet: A memory-oriented network-on-chip designed for fpga. *IEEE Transactions on VLSI Systems*, 27(6):1284–1297, 2019.

[36] Amit Bhanwala and et al. Fpga based design of low power reconfigurable router for network on chip (noc). In *ICCCA*, pages 1320–1326, 2015.

[37] Varsha Joy. Design and implementation of tunable network on chip for fpga applications. In *2020 Fourth International Conference on I-SMAC*, pages 1087–1092, 2020.

[38] Pascal et al. Vivet. Intact: A 96-core processor with six chiplets 3d-stacked on an active interposer with distributed interconnects and integrated power management. *IEEE Journal of Solid-State Circuits*, 2021.

[39] Yunfan Li and et al. Equinox: Equivalent noc injection routers for silicon interposer-based throughput processors. In *HPCA 2020*, pages 435–446, 2020.

[40] Yu-Hsin Chen and et al. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE Journal of Solid-State Circuits*, 52(1):127–138, 2017.

[41] Lei Yang and et al. Co-exploration of neural architectures and heterogeneous asic accelerator designs targeting multiple tasks. In *DAC 2020*, pages 1–6, 2020.

[42] Hyoukjun Kwon and et al. Maestro: A data-centric approach to understand reuse, performance, and hardware cost of dnn mappings. *IEEE MICRO*, 40(3):20–29, 2020.

[43] Robert Guirado and et al. Understanding the impact of on-chip communication on dnn accelerator performance. In *ICECS 2019*, pages 85–88, 2019.

[44] Biresh Kumar Joardar and et al. High-throughput training of deep cnns on reram-based heterogeneous architectures via optimized normalization layers. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(5):1537–1549, 2022.

[45] Yiming Ouyang and et al. Mmnnn: A tree-based multicast mechanism for noc-based deep neural network accelerators. *Microprocess. Microsyst.*, 85(C), sep 2021.

[46] Mark Clark and et al. Dozznoc: Reducing static and dynamic energy in nocs with low-latency voltage regulators using machine learning. In *IPDPS 2020*, pages 1–11, 2020.

[47] Wonje Choi and et al. Hybrid network-on-chip architectures for accelerating deep learning kernels on heterogeneous manycore platforms. In *CASES 2016*, pages 1–10, 2016.

[48] Arash Firuzan and et al. Reconfigurable network-on-chip for 3d neural network accelerators. In *NOCS 2018*, pages 1–8, 2018.

[49] Natalie Enright Jerger and et al. Noc architectures for silicon interposer systems: Why pay for more wires when you can get them (from your interposer) for free? In *MICRO 2014*, pages 458–470, 2014.

[50] Siamak Biglari and et al. Designing reconfigurable interconnection network of heterogeneous chiplets using kalman filter. In *Proceedings of the Great Lakes Symposium on VLSI 2024*, GLSVLSI '24, page 663–668, New York, NY, USA, 2024. ACM.

[51] Ayse Coskun and et al. Cross-layer co-optimization of network design and chiplet placement in 2.5-d systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(12):5183–5196, 2020.

[52] Jieming Yin and et al. Modular routing design for chiplet-based systems. In *ISCA 2018*, pages 726–738, 2018.

[53] Vidushi Goyal and et al. Neksus: An interconnect for heterogeneous system-in-package architectures. In *IPDPS 2020*, pages 12–21, 2020.

[54] Pritam Majumder and et al. Remote control: A simple deadlock avoidance scheme for modular systems-on-chip. *IEEE Transactions on Computers*, 70(11):1928–1941, 2021.

[55] Itir Akgun and et al. Scalable memory fabric for silicon interposer-based multi-core systems. In *ICCD 2016*, pages 33–40, 2016.

[56] Dylan Stow and et al. Cost-effective design of scalable high-performance systems using active and passive interposers. In *ICCAD 2017*, pages 728–735, 2017.

[57] Fuping Li and et al. Gia: A reusable general interposer architecture for agile chiplet integration. In *ICCAD 2022*, pages 1–9, 2022.

[58] Hyungjun Park and et al. Layer and length-deviation limit aware interposer routing for bend and wirelength minimization. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 14(6):993–1006, 2024.

[59] Jin-Tai Yan. Via-avoidance-oriented interposer routing for layer minimization in 2.5-d ic designs. *IEEE Transactions on VLSI Systems*, 29(11):1889–1902, 2021.

[60] Ehsan Nasiri and et al. Multiple dice working as one: Cad flows and routing architectures for silicon interposer fpgas. *IEEE Transactions on VLSI Systems*, 24(5):1821–1834, 2016.

[61] Rahul Boyapati and et al. Approx-noc: A data approximation framework for network-on-chip architectures. In *2017 ISCA*, 2017.

[62] Yu-Hsin Chen and et al. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. In *IEEE Journal of Solid-State Circuits ( Volume: 52, Issue: 1, January 2017*, 2017.

[63] Karthik Sangaiah and et al. Snacknoc: Processing in the communication layer. In *2020 HPCA*, 2020.

[64] Nan Jiang and et al. An in-network architecture for accelerating shared-memory multiprocessor collectives. In *2020 ISCA*, 2020.