# A Survey of On-Chip Hybrid Interconnect for Multicore Architectures

Cuong Pham-Quoc[1,2(✉)]

[1] Ho Chi Minh City University of Technology (HCMUT),
268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Vietnam
cuongpham@hcmut.edu.vn
[2] Vietnam National University - Ho Chi Minh City (VNU-HCM),
Ho Chi Minh City, Vietnam

**Abstract.** In this paper, we present a survey of hybrid interconnects for multicore architecture proposed in the literature. Before making a survey, we introduce an overview of on-chip hybrid interconnects and the taxonomies to classify them. We also present different architectures of standard interconnects that are frequently used for multi/many-core system in both the academia and industry. Finally, we conduct a survey of hybrid interconnects where we categorize them into two different groups. The first one includes hybrid interconnects that create the interconnects by using different Network-on-Chip topologies. We named this group as *topology-mixture hybrid interconnect*. The second group, named *architecture-mixture hybrid interconnects*, combines different architectures, such as bus and NoC, to form hybrid interconnects.

**Keywords:** Multicore systems · Hybrid interconnect · FPGA · ASIC

## 1 Introduction

In the past years, it has become clear that the continued scaling in transistor dimensions can no longer significantly increase processor performance. Factors like the power wall, memory wall, and instruction-level parallelism (ILP) wall have shifted the effort to increase performance towards parallel multicore processing. On the other hand, with the rapid development of technology, more and more transistors are integrated into a single chip. Today, it is possible to integrate more than 30 billion transistors [10] into one system. However, the more transistors are integrated into a system; the more challenges need to be addressed, such as power consumption, thermal emission, and memory access bottleneck. Therefore, homogeneous and heterogeneous multicore systems were introduced to efficiently utilize such large numbers of transistors. Compared to homogeneous multicore systems, heterogeneous multicore systems offer more computation power and efficient energy consumption [24] because of the efficiency of specialized cores for specific tasks.

Interconnect in a multicore system plays an important role because data is exchanged between all components, typically between processing elements (PEs)


C. V. Phan and T. D. Nguyen (Eds.): ICCASA 2022, LNICST 475, pp. 59–75, 2023.
https://doi.org/10.1007/978-3-031-28816-6_5

and memory modules, using the interconnect. However, interconnect design is one of the two open issues along with programming model in multicore system design [32]. During the last decades, many on-chip interconnects, especially hybrid interconnects, have been proposed, along with the rising number of PEs in the systems. Figure 1 (adapted from [26]) summarizes the evolution of on-chip interconnects.

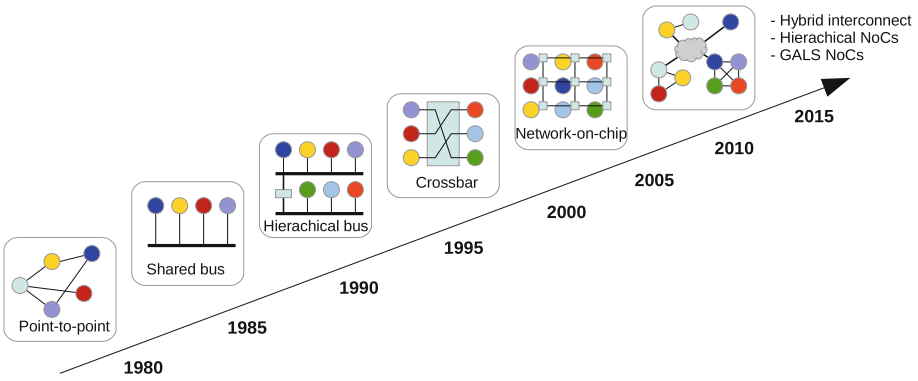


**Fig. 1.** The evolution of the on-chip interconnects (re-draw from [26])

In this paper, we conduct a survey of hybrid interconnects where we categorize them into two different groups. The first one includes hybrid interconnects that create the interconnects by using different Network-on-Chip topologies. We named this group as *topology-mixture hybrid interconnect*. The second group, named *architecture-mixture hybrid interconnects*, combines different architectures, such as bus and NoC, to form hybrid interconnects.

## 2 On-Chip Interconnect: An Overview

In modern computing systems, especially for big data processing multi/many-core systems, PEs cannot function independently. Therefore, these systems require communication networks, so-called the interconnection network, to transfer data from PE to PE or memory. System performance is substantially improved when deploying a suitable interconnection network because data communication overhead may take up to 50% processing time of entire applications. Performance, scalability, and cost are fundamental factors in choosing the right interconnection network [8].

As a sub-category of a more comprehensive - data communication network, an on-chip interconnect creates a connection and delivers required data for systems nodes[1] in a system-on-chip (SoC). Many approaches can be used to categorize interconnects into groups. In this paper, we presented five taxonomies for the classifying purpose.

[1] A node is any part that joins the network like a PE or a buffer.

### 2.1 Technique-Based Classification

We can classified on-chip interconnects two groups: shared memory and message passing [30] according the techniques used for transferring data from node to node.

- *Shared memory*: in this technique, a buffer/cache is responsible for exchanging data of PEs. When communicating, PE stores shared data into the buffer/cache in a joint address space so other PEs can load data for further processing. Bus-based communication, cache sharing, and crossbar are famous examples of this technique.
- *Message passing*: in this technique, explicit messages are responsible for conducting data communication among nodes. The source PE encapsulates data into packets according to the interconnect protocols before forwarding them to the destination via a the on-chip interconnect. Network-on-Chips (NoCs) are delegates of this type.

### 2.2 Topology-Based Classification

Duato et al. categorize interconnects into four groups including shared medium networks, direct networks, indirect networks and hybrid networks. The categories are based on the way PEs connected together [8].

- *Shared medium networks*: all computing nodes are connected to the same physical component for transferring data. Any communication between any pair of nodes will be conducted through the component. This group is similar to the Shared memory classification above.
- *Direct networks*: NoCs illustrate this group where a node is connected to a subset of other nodes in the system through P2P links. Network routers/adapters are attached to every node to encode data according to the NoCs protocol.
- *Indirect networks*: a crossbar is a good instance of this type of network in which one or more switches attach communicating nodes together.
- *Hybrid network*: this type of connection merges more than one type of network to alleviate this type's drawbacks by exploring others' benefits.

### 2.3 Link-Based Classification

Gama et al. classify interconnect into static and dynamic networks according to the characteristic of links that connect nodes [13].

- *Static networks*: links attaching nodes are fixed for data transferring. A link is dedicated to any pair of nodes. NoCs or buffer/cache sharing are delegates for this type.
- *Dynamic networks*: in contrast with the static one, a dynamic network includes switches and links that are reserved for data communication between two nodes for a while before being updated for the others. Buses and crossbars are instances of this type.

### 2.4 Routing Technique-Based Classification

El-Rewini et al. use the routing techniques for transferring messages from a source to a destination to classify interconnect into two classes: circuit switching and packet switching [9].

- *Circuit switching networks*: this type creates a physical path from a source node to a destination before transmitting data through the network. The published route exists for an entire communication interval. During this period, no other nodes contend to use the path. Buses, crossbars, and cache/buffer sharing are examples of this type.
- *Packet switching networks*: The networks encapsulate transferred data into fixed-length network packets. These packets are transmitted independently from a source to a destination through various paths. Some well-known examples of this switching technique are wormhole or virtual cut-through routing.

### 2.5 Architecture-Based Classification

On interconnects can be categorized according to the hardware architectures [11,23]. Below, we discuss shared cache/buffer (or shared memory), buses, crossbars, and network-on-chips because they are primary used in multi/many-core systems.

- *Shared cache/buffer*: in this architecture, as the name mentioned, nodes transfer communication data through a shared cache/buffer as illustrated in Fig. 2(a). Data movement between PEs is conducted with load/store behaviors through memory interfaces.
- *Bus*: The bus is one of the simplest and widely used in both sing core and multi/many-core systems. A bus attaches all system nodes as shown in Fig. 2(b). A bus-protocol [29] with request and granted behaviors is responsible for exchanging data between nodes.
- *Crossbar*: A generic crossbar includes $n$ inputs and $m$ outputs that can create an arbitrary connection between any pair of input and output. Figure 2(c) illustrates a $2 \times 2$ crossbar. Crossbars are frequently used to make connection of $n$ computing and $m$ storage (memory) nodes.
- *NoC*: network-on-chip is a generic architecture that is mainly used for transferring in multi/many-core systems. A NoC contains a set of routers connected by dedicated links. Router connections define different network topologies according to the connection patterns. Ring, 2D-mesh, torus or tree are primary network topologies explored in computing systems. Figure 2(d) shows a $3 \times 3$ 2D-mesh NoC.

Table 1 summarizes and shows relations of categories and hardware architecture of interconnects. Figure 3 presents pros and cons of four primary interconnect architectures mainly used in computing systems. Five parameters including latency, area-efficiency, scalability, system-performance, and power-efficiency are discussed in this comparison. Due to sequentially transferring data [33], buses

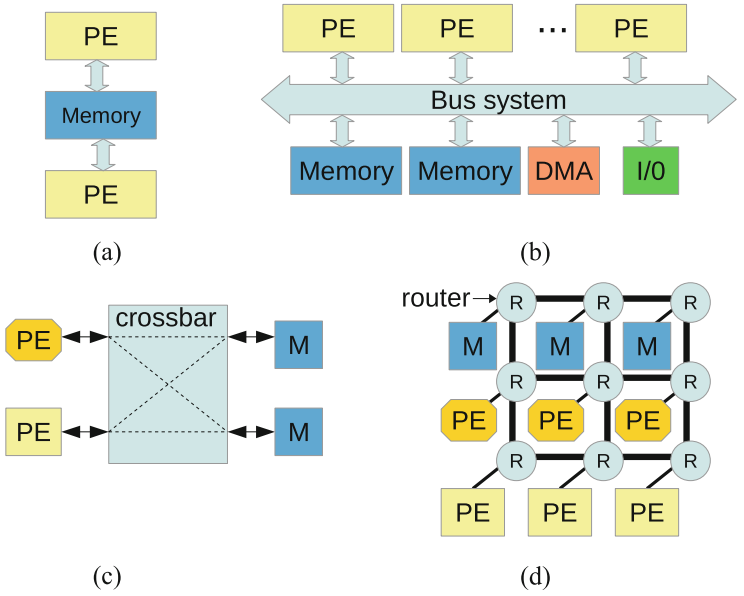


**Fig. 2.** (a) Shared cache/buffer; (b) Bus; (c) Crossbar; (d) Network-on-Chip

**Table 1.** Summary and relation of different interconnects

| Classification | Shared cache/buffer | Bus | Crossbar | NoC |
|---|---|---|---|---|
| Technique | SM | SM | SM | MP |
| Topology | SMed | SMed | Indirect | Direct |
| Link | Dynamic | Dynamic | Dynamic | Static |
| Routing | CS | CS | CS | PS |

- SM: Shared memory
- MP: Message passing
- SMed: Shared medium
- CS: Circuit switching
- PS: Packet switching

cannot offer high performance and suffer from low scalability. However, because of simplicity, buses are very efficient in terms of power consumption and hardware area. Since the connection is established for a source and a destination, the latency for data transferring is relatively low. Although shared cache/buffer provides an area-efficient, low hardware complex, and power-efficient architecture, it suffers from the worst scalability due to port limitations. However, compared to buses, shared cache/buffer does not introduce any data communication delay because there is no communication competition. Hence, systems with shared cache/buffer can achieve good performance. A crossbar can offer better system performance with low communication latency than a bus because multiple con-

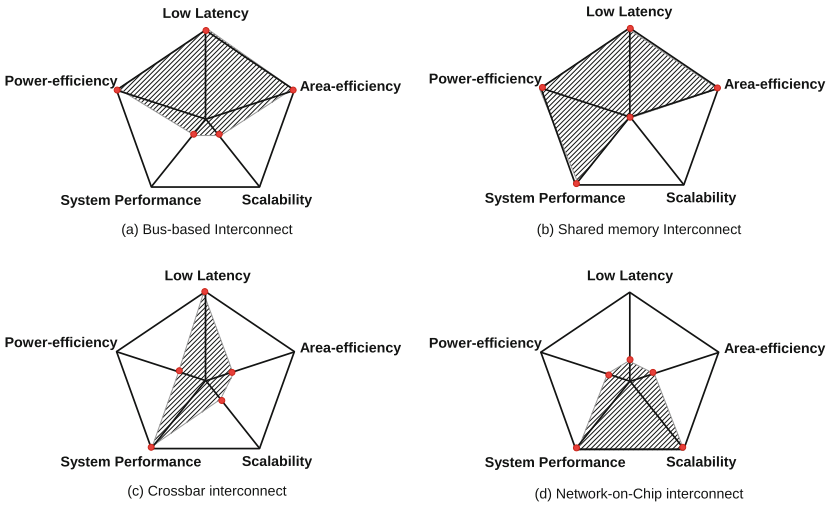


**Fig. 3.** Advantages and disadvantages of different interconnect architectures

nections between inputs and outputs can be established simultaneously [17]. However, when adding more ports, the resource usage for crossbars expands dramatically. Due to the rapid increase in hardware resource usage, a crossbar has high hardware complexity and power consumption. Finally, although offering absolute advantages such as high system performance and scalability, NoCs consume much power, require a vast amount of resources, and introduce high latency [16]. Therefore, hybrid interconnects taking all advantages of various architectures is a promising approach for multi/many-core systems that require high performance for time-consuming applications.

## 3 Survey of Hybrid Interconnects

As mentioned above, hybrid interconnect is a promising approach for improving system performance in multi/many-core platforms. In this section, we survey hybrid interconnect designs in the literature. We presented five approaches to categorize standalone on-chip connection networks into groups. Each group provides various advantages but also suffers from many drawbacks. For instance, direct networks are more straightforward than indirect ones in implementation cost. However, the former provides lower system performance and scalability than the latter. Besides, although circuit switching techniques offer higher bandwidth communication channels, packet switching ones will not block any messages because no routers or links are reserved for any physical paths. However,

the encoding and decoding processes of the packet switching techniques introduce overhead that may reduce system performance unawareness. Therefore, hybrid interconnects are getting accepted and proposed more and more in the last decades for exploiting the successes of various interconnect architectures and techniques.

From our perspective, hybrid interconnects can be designed with two different mixtures, including multiple NoC topologies and architectures. The first approach combines NoCs topologies like the 2D-mesh with the ring to create a hybrid mesh-ring interconnect. We call this approach as *Topology-mixture hybrid interconnect*. Meanwhile, the second approach exploits the advantages of various interconnect architectures. A new hybrid interconnect is designed by combining different architecture like a bus and an NoC. We call this hybrid type as *Architecture-mixture hybrid interconnect*. The following sections summarize the hybrid interconnects of each type.

### 3.1 Topology-Mixture

Network-on-chip topology [18] defines a structure that connects routers through physical links so that data can be transferred from a source to a destination. Some mainly used standard topologies are 2D-mesh, ring, hypercube, tree, and star, as illustrated in Fig. 4. Although each standard topology offers particular advantages, each has drawbacks that the others can improve. For instance, communication latency scalability and traffic concentration at center nodes are the two main obstacles of the 2D-mesh [5]. Meanwhile, ring topology cannot guarantee consistent latency for all nodes [30]. Therefore, mixture-topology or application-specific topology interconnects can help solve the drawbacks to improve advantages further. The following sections present mixture-topology interconnects in the literature.

CMesh (concentrated mesh) proposed by Balfour et al. in [2] connects every four nodes by a star topology. A 2D-mesh network links these 4-node groups at the higher level. The most significant advantage of CMesh compared to the original mesh is less average hop count. Kim et al. [21] extended the CMesh network and presented the Flattened Butterfly one where dedicated physical channels link 4-node groups in a row or a column. Hence, the dedicated P2P channels reduce the average hop counts to two. The simulated results show that CMesh improved area efficiency by 24% and reduced energy consumption by 48%. Meanwhile, the Flattened Butterfly used fewer hardware resources than 2D-mesh and CMesh 4× and 2.5×, respectively.

Murali et al. [28] introduced a design method to automatically synthesize a custom-tailored, application-specific NoC. The automated designed NoCs satisfy the targeted application domain's design objectives and constraints. The design framework considers two major minimizations: network power consumption and hops count. Hence, the framework executes the following step with task graphs as inputs to achieve the goals: (1) considering multiple topologies with variant switches number; (2) conducting the topologies floor-planning automatically; (3)

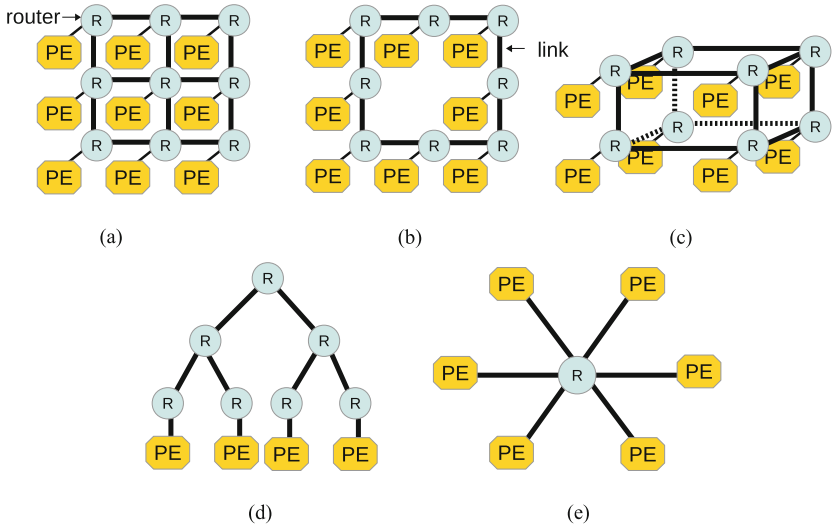


**Fig. 4.** Examples of NoC topologies: (a) 2D-mesh; (b) ring; (c) hypercube; (d) tree; and (e) star.

selecting the most optimized topologies with all the design objectives and constraints satisfied. An ARM-based embedded platform was used for evaluating the framework. The experimental results show that synthesized topologies automatically improved system performance by 1.73× with 2.78× power consumption reduced on average compared to the standard topologies.

Balkan et al. proposed the Mesh-of-Tree (MoT) [3] interconnection network that connects PEs and memory through two tree-based networks. In contrast to other architectures, the communicating nodes link with the root nodes instead of being connected to the leaf nodes. In this approach, a fan-out tree manages PEs while another fan-in tree handles memory modules. The fan-in and fan-out tree leaf nodes attach through 1-to-1 connecting channels. Two primary characteristics of the MoT network are a unique path between a pair of a source and a destination and not interfering in transferring packets from sources to destinations. The proposed architecture network is simulated for validation. The experimental results show that MoT improved network throughput by 76% and 28% compared to the butterfly and hypercube topologies, respectively.

Extending the MoT network, Balkan et al. presented the hybrid MoT-BF network architecture [4]. The extended version of MoT, MoT-BF, merges the MoT topology with the area-efficient butterfly network (BF). The ultimate target of this extension is to achieve low hardware resource usage for the MoT network implementation. Consequently, $2 \times 2$ butterfly networks replace the fan-in and fan-out trees' leaf nodes and some intermediate nodes. Furthermore, parameter $h$, the level of the MoT-$h$-BF network, was introduced as the number of intermediate nodes was replaced. Based on the simulated results, a 64 node MoT-BF

save area overhead by 34% with only 0.5% throughput penalty compared to the original MoT network.

Stensgaard et al. introduced the ReNoC [34] architecture to allow reconfiguring of NoC topologies with different applications' task graphs. ReNoC's nodes comprise traditional NoC routers in the proposed architecture, but topology switches wrap them. These switches can create links between NoC's routers and links or links together to bypass unneeded routers. This ability of switches allows various topologies to be defined according to applications' task graphs. The ultimate results could be a combination of 2D-meshes, P2P links, rings, or stars topologies. The implementation of ReNoC with the 90nm ASIC technology shows that ReNoC requires only 25% hardware resource compared to a static 2D-mesh with a 56% reduction in energy consumption.

G-Star/L-Hybrid proposed by Kim et al. [22] mixes a star topology global network and a star-mesh local network. The proposed network aims to degrade packet-drop rates. After trying different topologies combinations with various application domains, the authors decided that the star and mesh topologies combination is the most optimized approach regarding packet-drop rate. Experimental results show that the proposed architecture saves 45.5% packet-drop rate compared to others. In addition, the architecture also offers better power and area efficiency.

Modarressi presented the VIP hybrid interconnect architecture [27] that exploits the NoCs' scalability and P2P links' superior communication performance. The framework to define VIP's topology follows the below steps with applications' task graphs as a parameter: 1) physically assigning applications' tasks to a 2D-mesh NoC's routers; 2) building numerous P2P links for applications' tasks; 3) routing data packets through P2P links to save the 2D-mesh NoC's energy consumption and latency. The proposed VIP architecture is evaluated by simulation that introduces 20% NoC power consumption on average saved.

Bourduas et al. [5] presented various hierarchical topologies with ring networks. These hierarchies' ultimate goal is to lessen global traffic's hop counts and latencies. Therefore, the approach divides generic 2D meshes into several smallest $2 \times 2$ meshes, called sub-meshes. A ring connects every four sub-meshes to create a local mesh. Accordingly, these local meshes are then linked through another ring. A newly designed ring-mesh bridge element transfers packets between mesh and ring nodes. Besides, this approach also designed two new ring architectures. The first one aims for simplicity and low implementation cost. In contrast, the second one exploits the wormhole and virtual channel techniques to guarantee flexibility and performance. The simulated results prove the proposal's goals when outperforming a 2D-mesh if the number of nodes is less than forty-four.

Wang et al. introduced DMesh [38] consisting of E-subnet and W-subnet networks. Each subnet's router includes diagonal links to neighbor ones. These subnet networks transfer data packets separately eastward (E-subnet) and westward (W-subnet). When a source node starts forwarding data, packets are delivered through either E-router or W-router according to the destination's direction. The

authors also introduce a new routing technique for the proposal. SystemC-based simulation results show that DMesh is better than other $8 \times 8$ networks.

An extension of CMesh was proposed by Camacho et al. called PC-Mesh [6]. In PC-Mesh, extra 2D-meshes link batches of four consecutive nodes that the original CMesh network does not attach. With extra links, PC-Mesh offers higher fault tolerance and decreases latency in hops count. Furthermore, the authors propose a new routing algorithm to utilize the extra 2D-mesh networks because of multiple connections from a node to switches. The simulated results indicate that PC-Mesh can improve performance 2× and save 50% energy consumption compared to CMesh.

Yin et al. proposed a hybrid-switch NoC [39] that integrates P2P links with a standard 2D-mesh network. The architecture uses explicit configuration messages to define these P2P channels for often disseminating nodes. In other words, the architecture uses both packet and circuit switching techniques. Packets in the former technique are buffered, routed, and delivered at routers, while the latter uses dedicated channels for transferring data without communication overhead. Experimental results with simulation reveal that the architecture achieves 12% better system performance and saves 24% energy consumption compared NoCs without dedicated links.

Swaminathan et al. [35] introduced a hybrid NoC topology that merges 2D-mesh, torus, and folded. The mesh links attach two neighboring routers. Meanwhile, the folded-like channels connect odd routers or even routers in a row or a column. Finally, the torus-like links unite two border routers in a row/column. Because of extra channels, the authors designed a new routing algorithm for the proposed topology. Under the support of various topologies, the architecture decreases the average hop count compared to a single topology and enhances communication throughput. The experimental results with simulation indicate that the hybrid NoC improves 24% system performance compared to 2D-mesh networks.

Kang et al. [20] introduced an extension of MoT called 3-D MoT that allows topology to be reconfigurable. As a result, packets from nodes to nodes can travel through traditional or user-demand routes with the support of reconfigurable switches modified from the original ones. Experimental results state that 3-D MoT saves 13.34% execution time compared to conventional networks. Besides, the power consumption is also lower than the baseline networks. However, one of the biggest obstacles to this proposal is the lack of a formal approach for application designers.

Table 2 concludes all the aforementioned mixture topologies interconnects. We also summarize the targets of each approach in the last column of the table. According to the goals, most proposals focus on performance (latency or throughput) because it is the most important factor in multi/many-core systems.

**Table 2.** Topology mixture hybrid interconnects

| Research | Mixture topology | Input data[a] | Goals |
|---|---|---|---|
| [2,21] | Mesh/Star | Static[b] | Area, Power |
| [28] | Various[c] | User constraints | Performance, Power |
| [3] | Mesh/Tree | Static | Throughput |
| [4] | Mesh/Tree/ Butterfly | Static | Throughput |
| [34] | Various | Task graph | Area, Energy |
| [22] | Mesh/Star | Static | Packet drop, Area, Power |
| [27] | Mesh/P2P[d] | Task graph | Power |
| [5] | Mesh/Ring | Static | Latency |
| [38] | Mesh/Mesh | Static | Performance |
| [6] | Several Meshes | Static | Performance, Energy |
| [39] | Mesh/P2P | Communication rate | Performance, Energy |
| [35] | Folded/Mesh/ Torus | Static | Performance |
| [20] | Mesh/Tree | N/A | Performance |

[a]Parameters determine the topology.
[b]*Static*: fixed design without any parameters taken into account.
[c]*Various* multiple topologies used.
[d]Point-to-point.

### 3.2 Architecture-Mixture

Although many computing systems use shared caches/buffers, buses, crossbars, and NoCs as the primary communication infrastructure, these interconnect architectures still have some drawbacks, as presented in Sect. 2.5. Hence, many researchers have proposed hybrid interconnects exploiting multiple interconnect architectures to alleviate drawbacks of this type with the advantages of the others. In this section, we survey architecture-mixture hybrid interconnects published in recent years.

Richardson introduced dTDMA/NoC [31] hybrid interconnect, including buses and an NoC. The system uses a bus to link more frequent communication nodes to create an affinity group. Meanwhile, nodes outside these affinity sections transfer data through the NoC. The basic ideas upon which dTDMA/NoC is proposed are two following heuristics: 1) Buses supply higher transferring performance than NoCs within less than nine node groups; 2) When data rate increases, the performance of NoCs downgrades is extensively faster than buses. Hence, buses link nodes into affinity groups according to the frequency of nodes' communication. Each group of nodes connects with an NoC's router through a particular component called a bridge. Meanwhile, nodes outside all groups are also attached to routers for transferring data. Experimental simulation results show that the proposed hybrid interconnect improves the systems' performance

and energy efficiency. In the worst case, the proposal reduces 15.2% of latency and 8% of power consumption compared to traditional meshes.

Grot et al. introduced MECS [15] (Multidrop Express Channels) that exploits a CMesh NoC, presented in the mixture-topology group and bus-like one-to-many (1-to-m) channels. Although the architecture of the 1-to-m channel is likely a bus, only a primary node can broadcast data to secondary nodes linked through the channel. With the support of 1-to-m channels, the interconnect can handle multicast and broadcast with slightly additional overhead. The authors conducted simulations with many workloads and compared them with CMesh and Flatten Butterfly [21]. Experimental results show that MECS with 64 nodes saves 9% latency compared to the other interconnects.

Manevich et al. presented BENoC (Bus-enhanced NoC) [25] to combine an NoC with a technological bus. The bus with low and predictable latency transfers control signals in the system-wide distribution and issues broadcast and multicast. Hence, buses can help avoid the complexity and overhead of these behaviors for short messages with the NoC. Experimental results with simulations show that the BENoC obtains speedup by 3× compared to traditional NoCs.

Das et al. [7] used buses and a NoC to define hierarchical hybrid interconnects. A bus links every eight nodes to form a local network. In addition, these buses connect to 2D-mesh NoC's routers through adapters to define the entire network. Data transmission in the hybrid network can be performed entirely through the bus or become global transfers through NoC's routers to the destination. Simulation results show that the proposed interconnect is 14% better than traditional meshes in performance.

Avakian et al. presented a reconfigurable hybrid interconnect called RAMS [1] consisting of bus-based subsystems linked to mesh NoC's routers. Based on the heuristic that buses better support a small number of communication nodes (vary from 1 to 8), RAMS exploits scalable bus-based multi/many-core subsystems connected with each NoC's router. Compared to 2D-mesh NoC, the experimental results show that RAMS performs better than the original NoC.

Tsai et al. in [37] introduced a combination of buses and NoC for a hybrid interconnect. Instead of connecting only buses like RAMS, NoC's routers in this proposal attach both bus-based subsystems and computing cores. Using data communication graphs of applications, the framework in this work classifies more frequent communication cores into affinity groups. Meanwhile, ungrouped cores function as independent computing nodes. Due to the high communication rate, a bus links computing nodes in an affinity group to define a subsystem. Finally, these subsystems attach to routers through network interfaces. Experimental simulation results show that the proposed architecture saves latency by 17.6.

Zarkesh-Ha et al. proposed a similar mixture of buses and 2D-mesh called HNoC [40]. Local buses transfer data for nearest-neighbor communication while the global 2D-mesh NoC is responsible for further communication. In other words, along with the 2D-mesh NoC, nodes of two adjacent routers are linked through a local bus to conduct nearest-neighbor communication. Consequently, global traffic through the NoC is reduced at a higher throughput and lower

energy consumption. Simulation results indicate that 4.5× throughput improvement and 58% energy consumption reduction were obtained compared to only 2D-mesh.

Giefers et al. [12] presented a hybrid interconnect with three different architectures, including a reconfigurable mesh transformable to buses, a traditional NoC, and a barrier network. The mesh contains reconfigurable switches linked to computing nodes that can configure the switch dynamically to form buses. Along with switches, nodes also attach to NoC's routers for further communication. Finally, the barrier network is responsible for controlling the synchronization of nodes. Experimental results with an FPGA-based multi/many-cores platform show that the combination of three architectures offers the highest performance compared to any single one.

MORPHEUS [14] is a well-known heterogeneous hardware accelerator with multiple interconnect architectures. Buses transfer control and synchronization signals while a high-throughput NoC handle data communication among computing nodes and external memory like Flash or DDRAM.

Jin et al. introduced the *duo* [19] hybrid interconnect with a conventional 2D-mesh NoC linked with a reconfigurable multidrop channels bus (similar to MECS [15]). Thanks to the reconfiguration, each row or column requires a single channel rather than $2(n-1)$ like MECS. A framework trace communication patterns of applications to classify them. The framework then defines channels for application domains according to the traced patterns. The simulation results indicate that 15% of latency and 27% of energy consumption were obtained compared to 2D-mesh.

Zhao et al. in [41] presented a bus-NoC hybrid interconnect with buses linked to an NoC. Each NoC's router includes an interface for sharing physical channels between the NoC and buses. The interface is programmable, so NoC's routers can be bypassed to form a bus. When buses are defined from links, NoC's routers store data packets inside them to wait to finish bus transactions. The evaluation results prove that 12% of system performance and 37% of energy efficiency were obtained.

Todorov et al. in [36] designed a deterministic synthesis framework to define application-specific interconnects with buses and an NoC. The application use-cases with the bandwidth, latency, and packet size are parameters of the synthesis tool to divide computing nodes into clusters. Low communication clusters link to shared buses while high communication ones attach to NoC's router. The author also proposed a deadlock-free routing algorithm to handle the data flow of the interconnect. Experimental results demonstrate that the interconnect achieves similar latency with traditional NoC with 22.6% of hardware resource reduction compared to a conventional NoC.

Table 3 concludes all the aforementioned architecture-mixture interconnects. We also summarize the goals of each proposal in the last column of the table. According to the goals, most approaches target performance like the mixture topologies interconnects above.

**Table 3.** Architecture mixture hybrid interconnect

| Research | Mixture topology | Input data[a] | Goals |
|---|---|---|---|
| [31] | Bus/NoC | Communication rate | Latency, Power |
| [15] | Bus-like/NoC | Static | Latency |
| [25] | Bus/NoC | Static[b] | Performance |
| [7] | Bus/NoC | Static | Performance |
| [1] | Bus/NoC | Memory access rate | Performance |
| [37] | Bus/NoC | Communication bandwidth | Latency |
| [40] | Bus/NoC | Static | Throughput, Energy |
| [12] | Bus/NoC/ Barrier | Static | Performance |
| [14] | Bus/NoC | Static | Throughput |
| [19] | Bus-like/NoC | Communication rate | Latency, Energy |
| [41] | Bus/NoC | Static | Performance, Energy |
| [36] | Bus/NoC routers | Bandwidth and Latency | Area |

[a]Parameters determine the topology.
[b]*Static*: fixed design without any parameters taken into account.

## 4 Conclusion

In this paper, we summarizes on-chip hybrid interconnect architectures for the literature. We classify these hybrid interconnects into two different categories. The first one includes hybrid interconnects that create the interconnects by using different Network-on-Chip topologies. We named this group as *topology-mixture hybrid interconnect*. The second group, named *architecture-mixture hybrid interconnects*, combines different architectures, such as bus and NoC, to form hybrid interconnects. Each of proposals in each class provide different approach with various goals. Researchers can choose the most appropriate one for their multi/many-core systems.

**Acknowledgment.** We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

## References

1. Avakian, A., et al.: A reconfigurable architecture for multicore systems. In: IPDPSW, pp. 1–8 (2010). https://doi.org/10.1109/IPDPSW.2010.5470753
2. Balfour, J., Dally, W.J.: Design tradeoffs for tiled CMP on-chip networks. In: Proceedings of the 20th Annual International Conference on Supercomputing, ICS 2006, pp. 187–198. ACM, New York (2006). https://doi.org/10.1145/1183401.1183430. http://doi.acm.org/10.1145/1183401.1183430
3. Balkan, A., Qu, G., Vishkin, U.: A mesh-of-trees interconnection network for single-chip parallel processing. In: International Conference on Application-Specific Systems, Architectures and Processors, ASAP 2006, pp. 73–80 (2006). https://doi.org/10.1109/ASAP.2006.6

4. Balkan, A.O., Qu, G., Vishkin, U.: An area-efficient high-throughput hybrid interconnection network for single-chip parallel processing. In: Proceedings of the 45th Annual Design Automation Conference, DAC 2008, pp. 435–440, ACM, New York (2008). https://doi.org/10.1145/1391469.1391583. http://doi.acm.org/10.1145/1391469.1391583
5. Bourduas, S., Zilic, Z.: Modeling and evaluation of ring-based interconnects for network-on-chip. J. Syst. Archit. **57**(1), 39–60 (2011). https://doi.org/10.1016/j.sysarc.2010.07.002. http://www.sciencedirect.com/science/article/pii/S138376211000069X. Special Issue On-Chip Parallel and Network-Based Systems
6. Camacho, J., Flich, J., Roca, A., Duato, J.: PC-mesh: a dynamic parallel concentrated mesh. In: 2011 International Conference on Parallel Processing (ICPP), pp. 642–651 (2011). https://doi.org/10.1109/ICPP.2011.21
7. Das, R., et al.: Design and evaluation of a hierarchical on-chip interconnect for next-generation CMPs. In: HPCA 2009, pp. 175–186 (2009). https://doi.org/10.1109/HPCA.2009.4798252
8. Duato, J., Yalamanchili, S., Lionel, N.: Interconnection Networks: An Engineering Approach. Morgan Kaufmann Publishers Inc., San Francisco (2002)
9. El-Rewini, H., Abd-El-Barr, M.: Advanced Computer Architecture and Parallel Processing (Wiley Series on Parallel and Distributed Computing). Wiley-Interscience (2005)
10. Gazettabyte: Altera's 30 billion transistor FPGA (2016). http://www.gazettabyte.com/home/2015/6/28/alteras-30-billion-transistor-fpga.html
11. Gebali, F.: Interconnection Networks, pp. 83–103. Wiley (2011). https://doi.org/10.1002/9780470932025.ch5
12. Giefers, H., Platzner, M.: A triple hybrid interconnect for many-cores: reconfigurable mesh, NoC and barrier. In: FPL, pp. 223–228 (2010). https://doi.org/10.1109/FPL.2010.52
13. Grama, A., Gupta, A., Karypis, G., Kumar, V.: Introduction to Parallel Computing, 2nd edn. Addison-Wesley Longman Publishing Co., Inc., Boston (2002)
14. Grasset, A., et al.: The Morpheus heterogeneous dynamically reconfigurable platform. Int. J. Parallel Program. **39**(3), 328–356 (2011)
15. Grot, B., et al.: Express cube topologies for on-chip interconnects. In: HPCA, pp. 163–174 (2009). https://doi.org/10.1109/HPCA.2009.4798251
16. Guerrier, P., Greiner, A.: A generic architecture for on-chip packet-switched interconnections. In: DATE, pp. 250–256 (2000). https://doi.org/10.1109/DATE.2000.840047
17. Hur, J.: Customizing and hardwiring on-chip interconnects in FPGAs. Ph.D. thesis, Delft University of Technology, Delft, Netherlands (2011)
18. Jerger, N.E., Peh, L.S.: On-Chip Networks, 1st edn. Morgan and Claypool Publishers (2009)
19. Jin, Y., et al.: Communication-aware globally-coordinated on-chip networks. Parallel Distrib. Syst. **23**(2), 242–254 (2012). https://doi.org/10.1109/TPDS.2011.164
20. Kang, K., Park, S., Lee, J.B., Benini, L., Micheli, G.D.: A power-efficient 3-D on-chip interconnect for multi-core accelerators with stacked L2 cache. In: 2016 Design, Automation Test in Europe Conference Exhibition (DATE), pp. 1465–1468 (2016)
21. Kim, J., Balfour, J., Dally, W.: Flattened butterfly topology for on-chip networks. Comput. Archit. Lett. **6**(2), 37–40 (2007). https://doi.org/10.1109/L-CA.2007.10
22. Kim, W.J., Hwang, S.Y.: Design of an area-efficient and low-power NoC architecture using a hybrid network topology. IEICE Trans. Fundam. Electron. Commun.

Comput. Sci. **E91-A**(11), 3297–3303 (2008). https://doi.org/10.1093/ietfec/e91-a.11.3297
23. Kogel, T., Leupers, R., Meyr, H.: Classification of platform elements. In: Kogel, T., Leupers, R., Meyr, H. (eds.) Integrated System-Level Modeling of Network-on-Chip enabled Multi-Processor Platforms, pp. 15–32. Springer, Dordrecht (2006). https://doi.org/10.1007/1-4020-4826-2_3
24. Kumar, R., et al.: Heterogeneous chip multiprocessors. Computer **38**(11), 32–38 (2005)
25. Manevich, R., et al.: Best of both worlds: a bus enhanced NoC (BENoC). In: Networks-on-Chip, pp. 173–182 (2009). https://doi.org/10.1109/NOCS.2009.5071465
26. Matos, D., Concatto, C., Carro, L.: Reconfigurable intercommunication infrastructure: NoCs. In: Beck, A.C.S., Lang Lisbôa, C.A., Carro, L. (eds.) Adaptable Embedded Systems, pp. 119–161. Springer, New York (2013). https://doi.org/10.1007/978-1-4614-1746-0_5
27. Modarressi, M., Tavakkol, A., Sarbazi-Azad, H.: Virtual point-to-point connections for NoCs. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **29**(6), 855–868 (2010). https://doi.org/10.1109/TCAD.2010.2048402
28. Murali, S., et al.: Designing application-specific networks on chips with floorplan information. In: Proceedings of the 2006 IEEE/ACM International Conference on Computer-Aided Design, ICCAD 2006, pp. 355–362. ACM, New York (2006). https://doi.org/10.1145/1233501.1233573. http://doi.acm.org/10.1145/1233501.1233573
29. Pasricha, S., Dutt, N.: Basic concepts of bus-based communication architectures, chapter 2. In: Pasricha, S., Dutt, N. (eds.) On-Chip Communication Architectures, pp. 17–41. Systems on Silicon, Morgan Kaufmann (2008). https://doi.org/10.1016/B978-0-12-373892-9.00002-5. http://www.sciencedirect.com/science/article/pii/B9780123738929000025
30. Pham, D., Holt, J., Deshpande, S.: Embedded multicore systems: design challenges and opportunities. In: Hübner, M., Becker, J. (eds.) Multiprocessor System-on-Chip, pp. 197–222. Springer, New York (2011). https://doi.org/10.1007/978-1-4419-6460-1_9
31. Richardson, T., et al.: A hybrid SoC interconnect with dynamic TDMA-based transaction-less buses and on-chip networks. In: VLSI Design, p. 8 (2006). https://doi.org/10.1109/VLSID.2006.10
32. Rutzig, M., et al.: Multicore platforms: processors, communication and memories. In: Beck, A., Lang Lisbôa, C., Carro, L. (eds.) Adaptable Embedded Systems, pp. 243–277. Springer, Heidelberg (2013). https://doi.org/10.1007/978-1-4614-1746-0_8
33. Sanchez, D., et al.: An analysis of on-chip interconnection networks for large-scale chip multiprocessors. ACM Trans. Archit. Code Optim. **7**(1), 4:1–4:28 (2010)
34. Stensgaard, M., Sparso, J.: ReNoC: a network-on-chip architecture with reconfigurable topology. In: Second ACM/IEEE International Symposium on Networks-on-Chip, 2008, NoCS 2008, pp. 55–64 (2008). https://doi.org/10.1109/NOCS.2008.4492725
35. Swaminathan, K., Gopi, S., Rajkumar, Lakshminarayanan, G., Ko, S.B.: A novel hybrid topology for network on chip. In: 2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1–6 (2014). https://doi.org/10.1109/CCECE.2014.6901083

36. Todorov, V., Mueller-Gritschneder, D., Reinig, H., Schlichtmann, U., et al.: Deterministic synthesis of hybrid application-specific network-on-chip topologies. Comput.-Aided Des. Integr. Circuits Syst. **33**(10), 1503–1516 (2014). https://doi.org/10.1109/TCAD.2014.2331556
37. Tsai, K.L., et al.: Design of low latency on-chip communication based on hybrid NoC architecture. In: NEWCAS, pp. 257–260 (2010). https://doi.org/10.1109/NEWCAS.2010.5603934
38. Wang, C., Hu, W.H., Lee, S.E., Bagherzadeh, N.: Area and power-efficient innovative congestion-aware network-on-chip architecture. J. Syst. Archit. **57**(1), 24–38 (2011). https://doi.org/10.1016/j.sysarc.2010.10.009. http://www.sciencedirect.com/science/article/pii/S1383762110001359. Special Issue On-Chip Parallel And Network-Based Systems
39. Yin, J., Zhou, P., Sapatnekar, S.S., Zhai, A.: Energy-efficient time-division multiplexed hybrid-switched NoC for heterogeneous multicore systems. In: Proceedings of the 2014 IEEE 28th International Parallel and Distributed Processing Symposium, IPDPS 2014, Washington, DC, USA, pp. 293–303. IEEE Computer Society (2014). https://doi.org/10.1109/IPDPS.2014.40
40. Zarkesh-Ha, P., et al.: Hybrid network on chip (HNoC): local buses with a global mesh architecture. In: System Level Interconnect Prediction, pp. 9–14. ACM, New York (2010)
41. Zhao, H., et al.: A hybrid NoC design for cache coherence optimization for chip multiprocessors. In: DAC, pp. 834–842. ACM, New York (2012)