IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 11 October 2023, accepted 28 October 2023, date of publication 1 November 2023, date of current version 7 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3329233

SURVEY

A Survey on Dynamic Application Mapping Approaches for Real-Time Network-on-Chip-Based Platforms

SHAROON SALEEM[1], FAWAD HUSSAIN[1], (Senior Member, IEEE), WAQAR AMIN[1], REHAN AHMED[2], YOUSAF BIN ZIKRIA[3], (Senior Member, IEEE), FARRUH ISHMANOV[4], AND HEEJUNG YU[5], (Senior Member, IEEE)

[1]Department of Computer Engineering, University of Engineering and Technology, Taxila (UET Taxila), Taxila 47050, Pakistan
[2]School of Electrical Engineering and Computer Science, National University of Science and Technology, Islamabad 44000, Pakistan
[3]Victorian Institute of Technology (VIT), The Rocks, NSW 2000, Australia
[4]Department of Electronics and Communication Engineering, Kwangwoon University, Seoul 01897, South Korea
[5]Department of Electronics and Information Engineering, Korea University, Sejong 30019, South Korea

Corresponding authors: Heejung Yu (heejungyu@korea.ac.kr) and Farruh Ishmanov (farruh@kw.ac.kr)

This work was supported in part by the "Regional Innovation Strategy (RIS)" through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (MOE) under Grant 2021RIS-004; in part by the Ministry of Science and ICT (MSIT), South Korea, through the Information Technology Research Center (ITRC) Support Program, Supervised by the Institute for Information and Communications Technology Planning and Evaluation (IITP), under Grant IITP-2023-RS-2022-00164800; in part by the Korea University Grant; and in part by the Research Grant of Kwangwoon University, Seoul, South Korea, in 2023.

ABSTRACT Network-on-Chip (NoC) has been unfolded as a superior alternative for integrating a considerably greater extent of cores on a single chip. Recently, multi-core systems have become prevalent because of the increased processing demands for high-performance embedded applications. Application mapping techniques play a significant role in enhancing the extensive performance of such complex multi-core platforms. Developing and implementing efficient application mapping techniques are required for system design to meet the demand of such complicated multi-core systems. The paper primarily focuses on dynamic application mapping techniques, classifying them into a number of subcategories. It highlights such approaches and techniques that aim to enhance the performance of the NoC-based systems by optimizing them in terms of communication cost, latency, energy consumption, power, execution, and computational time. Future challenges, trends, and simulation tools have also been spotlighted.

INDEX TERMS Network-on-chip, application mapping, system-on-chip, VOPD.

I. INTRODUCTION

Due to the rise in complexity of embedded This is some system devices, system-on-chip (SoC) incorporating the numerous processing cores on a single chip to perform various functions is the primary paradigm of today's digital world. The SoC uses a shared medium bus to communicate intellectual property (IP) cores and is widely utilized in domain-specific devices [1], [2], such as aerospace, medical sciences, microprocessors-based technology, and wireless communications. Due to the higher density of components in SoC, the implementation of a shared-bus architecture is getting complicated. As the number of cores increases on the chip, the performance is not enhanced and scaled by the increased processing cores due to the limitations of the shared-bus architecture. Network-on-chip (NoC) has emerged as a feasible substitute to cater to the new inter-core communication demands of the growing number of components on a chip and faster communication between the cores [3]. NoC architecture is considered a part of or a subset of SoC-based technology [4]. NoC uses IP cores connected with the routers and inter-switch links [5], and the communication between the cores is done by transferring packets with these routers and links. Messages are divided

The associate editor coordinating the review of this manuscript and approving it for publication was Gian Domenico Licciardo.

into smaller packets to be transmitted between various cores that allow for the efficient use of network resources. NoC employs a routing algorithm for the determination of the path that each packet follows from the source to the destination. Various switching techniques are developed to transfer packets such as circuit and packet switching. A physical path from source to destination is reserved before the data transmission in circuit switching. While in packet switching, each message is partitioned into fixed-length packets which are transmitted without reserving the entire path. Buffers are placed at the NoC routers to temporarily store packets that need transmission. NoC provides considerable refinement in SoC communication infrastructure, mainly by replacing the bus-based architecture. NoC design and development are undoubtedly one of the significant research areas of digital system design in the current era. The research problems in a NoC domain are broadly distributed into different categories. One of the categories belongs to communication infrastructure, such as the architecture of the router, optimization of buffer utilization, topology in NoC, clocking, floor planning, and layout. Another category deals with a communication archetype, such as packet routing policies, network congestion, switching techniques, ensuring reliability in NoC systems, and thermal and power management. The evaluation framework is another category that deals with the system's bandwidth, throughput, and latency. Once these design and communication infrastructure-related requirements are fulfilled and finalized, a key challenge in the final design and architecture of a NoC platform is the association of IP cores to routers. Thus, application mapping is a significant step in the overall design process [6] that directly affects the communication time, cost, bandwidth, and overall NoC performance. Many surveys [4], [7], [8] and textbooks [9], [10] related to NoC have been published. However, there have been various advanced research problems in the field of NoC that need to be addressed. Besides other research problems reported in the literature, the mapping takes on a significant and distinguished place amongst other research problems. It presents a multitude of complexities and numerous challenges, particularly when faced with dynamic scenarios. These include fault occurrences, network congestion, load balancing, and the management of high energy consumption caused by task placements. The main purpose of this survey is to furnish an extensive overview of the major traits of NoC mapping, and the main emphasis is on Application Mapping in such dynamic scenarios. The distinct techniques on the basis of dynamic scenarios are first segregated into different categories, and their various dimensions, along with their contributions to enhancing the performance metrics are presented, which is the main contribution in a real-time mapping domain. Also, research directions are provided extensively in this survey paper. The rest of the survey is organized as follows: Section I presents the broader overview and some essential components related to application mapping problems in a NoC-based multi-core platform. Detail discussion on application mapping techniques is presented in Section II. Section III presents details related to the simulators and performance metrics. Section IV presents the latest trends and future related to application mapping. Finally, Section V concludes this article.

A. SCOPE

In this study, we focus on the implementation of application mapping techniques and strategies in NoC systems in dynamic scenarios. To accurately approximate area and overall power usage for NoC platforms, the simulation work is carried out to evaluate and derive latency, power consumption, and communication bandwidth. The initial steps are to develop communication infrastructure, methodology, and evaluation setup. After the initial steps, one of the highly significant steps is to obtain the optimal placement and mapping of application tasks to processing cores in a NoC platform. This helps to improve the communication cost, the overall latency, and system performance.

B. APPLICATION MAPPING OVERVIEW

An application mapping design problem has become a crucial aspect of NoC architecture and design. It is required to efficiently map and place the application tasks onto the appropriate processing elements to fully exploit the resources incorporating NoC. The application task mapping is allocating tasks of an application onto cores while considering the various optimization factors and criteria, such as total execution time, latency, and energy consumption [6]. The parameters mentioned above are directly affected by optimal task mapping decisions. Numerous algorithms and techniques have been designed and developed to achieve optimal task mappings. The application mapping problem is known as a non-deterministic polynomial-time (NP)-hard problem [7].

In [11] and [12], research problems related to NoC and a general discussion on NoC architectures, platforms, and applications are provided. A comprehensive survey of application mapping techniques and their classifications into static and dynamic categories is presented in [13]. The authors of [14] and [15] have presented various mapping techniques on multi-core systems. A survey on fault tolerant-based application mapping techniques for NoC platforms has been presented in [16], which discusses techniques used to recover the system from the faults. Several approaches are discussed comprehensively in the survey, such as mapping techniques integrated with routing, redundancy techniques, and remapping configurations. In addition, a performance comparison is also spotlighted. A survey on optimization of mapping and scheduling techniques for NoC platforms has been provided in [17]. The survey also classifies mapping techniques into static and dynamic domains. A comprehensive survey on application mapping techniques and their comparative analysis have been presented in [18]. The survey discusses the classification of application mapping techniques focusing

on static mapping techniques. Moreover, a comparison in terms of communication cost, power, latency, and energy consumption is also presented for implementation of real-time applications, such as video object plane decoder (VOPD) and moving picture expert group-4 (MPEG-4).

Many surveys and reviews are available related to the other three research dimensions of NoC architecture and design. However, the fourth dimension, i.e., application mapping, is not reviewed to that extent. Although many comprehensive surveys are available that discuss the application mapping problem and approaches in detail, there still exists room to further elaborate and discuss the application mapping techniques in dynamic scenarios. The primary aim and emphasis of this study is to present a classification of application mapping techniques, especially in real-time dynamic scenarios. This study mainly targets different domains concerning several measures, such as the execution of new enabling applications in the system, fault avoidance, critical thermal conditions, and readiness. This paper also presents application mapping in congestion-based scenarios and cases where the NoC status in terms of loaded links and cores is taken into account. These are the real-time challenges confronted by the system, and dynamic mapping approaches that are efficient and effective can address them. Finally, futuristic trends in the NoC mapping domains are also highlighted. Figure 1 depicts the NoC application mapping process that consists of an application core graph to be mapped onto 2×3 mesh topology. The cores are mapped to various available locations as shown in the mesh topology.

C. GENERATION OF TASKS GRAPHS

Any application that runs on the NoC platform is characterized by a task graph consisting of an instruction set that sequentially executes on the processor [19]. The communication task graph (CTG) shows the application divided into various specific tasks and the application's communication sample. It represents the data volume exchanged between the applications' tasks. A task graph N can be represented as $N = N(T, C)$; the vertexes of a task graph stand for a set of tasks ($T = \{T1, T2, T3, \cdots, Tn\}$) and tasks have the associated information regarding their communication. The directed arcs ($C = \{c_{i,j} | i, j = 1, 2, 3, \cdots, n\}$) which occur amongst the tasks characterize interdependencies and volume of data between two application tasks or nodes.

Application task graphs are obtained in the following way:

1) By the use of the embedded system synthesis benchmarks suite (E3S)
2) By generating the task graphs using the task graphs for free (TGFF) tool [20]
3) By real-world applications having multi-threaded features, attained by using the communication extraction from threaded applications (CETA) tool [21].
4) The use of a scaling algorithm such as the benchmark scaling algorithm in [22] for the generation of extensive and large task graphs.

In a NoC design, task graphs are often linked with deadlines to ensure the timely execution of tasks in a multi-core platform. These deadlines are significant for real-time and time-sensitive applications. There are tasks in real-time systems with hard deadlines that must be completed within a specified time frame; otherwise, the system may fail to produce the correct results. Scheduling techniques can be employed to ensure the tasks meet their deadlines efficiently, adjusting computational and communication resources with real-time requirements. Such methods optimize the task execution to ensure reliable task completion. To meet the task deadline, considerations of task dependencies, delays, and resource allocation are required, and scheduling analysis is incorporated to ensure the efficient execution of applications.

Generating and simulating task graphs have varying complexities to deal with, such as type and size of application, communication bandwidth measurement between tasks, and their deadline requirements. Also, it would require certain NoC setups, such as selecting appropriate topology, routing protocol, and injection rate with various other run-time parameters. One important consideration is task modeling complexity of representing real-world applications such as task graphs with accurate timing and dependencies constraints for complex dynamic systems. The other challenge could be realistic workload generation that mimics the behavior of actual real-time systems. In order to accurately simulate it on real-time systems, the simulation of real-time task graphs requires precise modeling of underlying resource-intensive hardware/ software modules. Even scaling up the simulation to cater to large-scale systems with numerous tasks and modules while maintaining real-time responsiveness is one of the significant challenges. Addressing these challenges while looking into the constraints in task graph generation and simulation requires effective design and analysis of real-time systems.

This survey mainly emphasizes run-time-based application techniques and their various dimensions. For detailed literature regarding task generation tools, the readers may explore graph generation research presented in [23], [24], and [25] in which information related to generations of synthetic graphs is presented.

II. CLASSIFYING THE APPLICATION MAPPING STRATEGIES

Mapping application tasks to cores in the NoC platform can either be performed at the design time (i.e., static or offline mapping) or at the run-time (i.e., dynamic or online mapping). Design time techniques use static platforms and predefined tasks with specific communication characteristics. Thus, the static techniques are unsuitable for dynamic scenarios with variable workloads and other variable factors in which new applications may emerge at the system's run-time that need execution on the NoC platform. In the case of static mapping, placement of application tasks onto the cores in NoC is defined at the design time, i.e., the decision regarding mapping tasks is made before the execution.

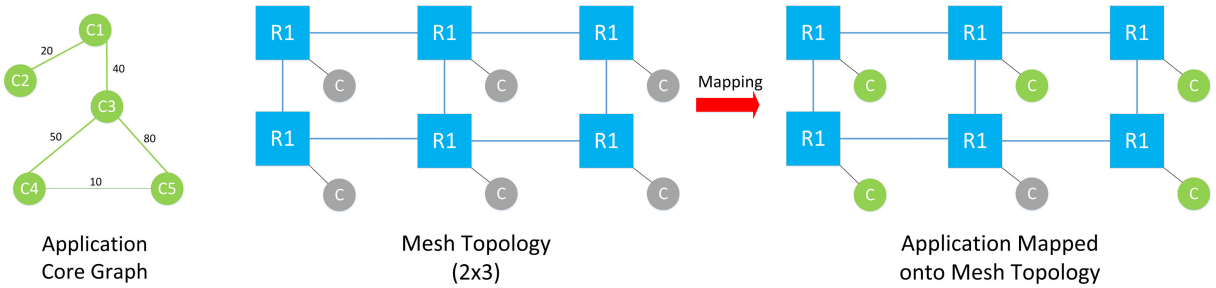


FIGURE 1. Application mapping onto NoC.

It means that the mapping is performed offline before the application execution. The static mapping policy always attempts to define the finest possible positioning of tasks in the offline design time mode for a particular application and the target inter-communication infrastructure. Once the mapping is finalized, the mapped tasks are executed at run-time, and the mapping cannot be changed. Due to the algorithm complexity and the lengthy execution time of the mapping solutions, the static mapping techniques may not be appropriate for dynamic workload scenarios [26].

A. CHALLENGES AND CONSTRAINTS OF REAL-TIME/RUN-TIME MAPPING

A dynamic mapping technique can allocate each application task onto a core at run-time in contrast to static mapping. A system may encounter various challenges and face constraints at run-time that arise from dynamic variations of system conditions in its operation. Real-time NoC mapping is a complex optimization problem as it requires specialized strategies and techniques to meet real-time constraints while adapting to varying conditions. At a high level, some of the key constraints and challenges while generating an optimal mapping for real-time systems are resource and timing constraints, dynamic workload-related challenges and constraints, fault tolerance, resource sharing, and reconfiguration-related challenges.

Availability of resources and satisfying task deadlines are significant challenges that need to be addressed using state-of-the-art mapping methods. One significant challenge is the potential performance degradation brought on by an overwhelming workload on some cores while others remain underutilized. Also, faults may occur at any core, further impacting the system's performance. Another major issue is congestion, which may occur when links between the cores become overloaded, leading to reduced system efficiency. One other challenge of thermal variations that may threaten the system's performance and reliability. Addressing these challenges during run-time is a crucial issue. The system must be able to adapt to these changing conditions. The adaption is necessary in order to optimize performance, balance workload, mitigate faults and critical thermal conditions, and alleviate congestion on heavily utilized links. Optimal dynamic application mapping algorithms offer a solution to these run-time variations and critical challenges. These algorithms involve intelligent algorithms and techniques that allocate tasks to the cores more effectively. Dynamic reconfiguration of the system's resources is performed in order to mitigate all the issues and challenges discussed. However, the algorithm may increase the computational complexity at the run-time, thus increasing the energy usage and the run-time execution delay. These challenges can be solved by designing and developing efficient NoC Application Mapping techniques. Many such techniques that address the complex challenges are presented in this research work and organized into sub-categories. Design time mapping, i.e., static mapping, is generally recommended and widely used for NoC multi-core systems because dynamic mapping may significantly cause excess communication overhead, eventually increasing the system's overall delay, thus affecting system performance. However, the static mapping techniques do not consider the overall network load, average channel load between the cores, and load balancing amongst the cores. Therefore, the dynamic mapping techniques have significant advantages over the static mapping algorithms. In summary, Figure 2 presents the categorization of various mapping techniques.

B. DYNAMIC MAPPING TECHNIQUES IN NOC

Dynamic (run-time) mapping approaches are needed when application tasks are required to be inserted into the system at their run-time. After the tasks have been mapped, task migration strategies can revise the placement of tasks that are already mapped if there are changes in the user requirement or application execution is needed to enter the system. Dynamic mapping can be either distributed or centralized, depending upon the type of control management unit that performs mapping-related activities at run-time. In centralized dynamic mapping, a centralized core, referred to as a manager core, makes the mapping decisions, whereas, in distributed mapping, multiple agents or nodes share network information or status to make the optimal mapping decision. Applications, such as multimedia and networking that run on heterogeneous multi-processor SoCs (MPSoCs), usually have a diverse workload. This means that a variable number of tasks are simultaneously being executed at the MPSoC. For dynamical workloads, this can necessitate the execution of task mapping at run-time to satisfy real-time constraints. A multi-core processor also uses task

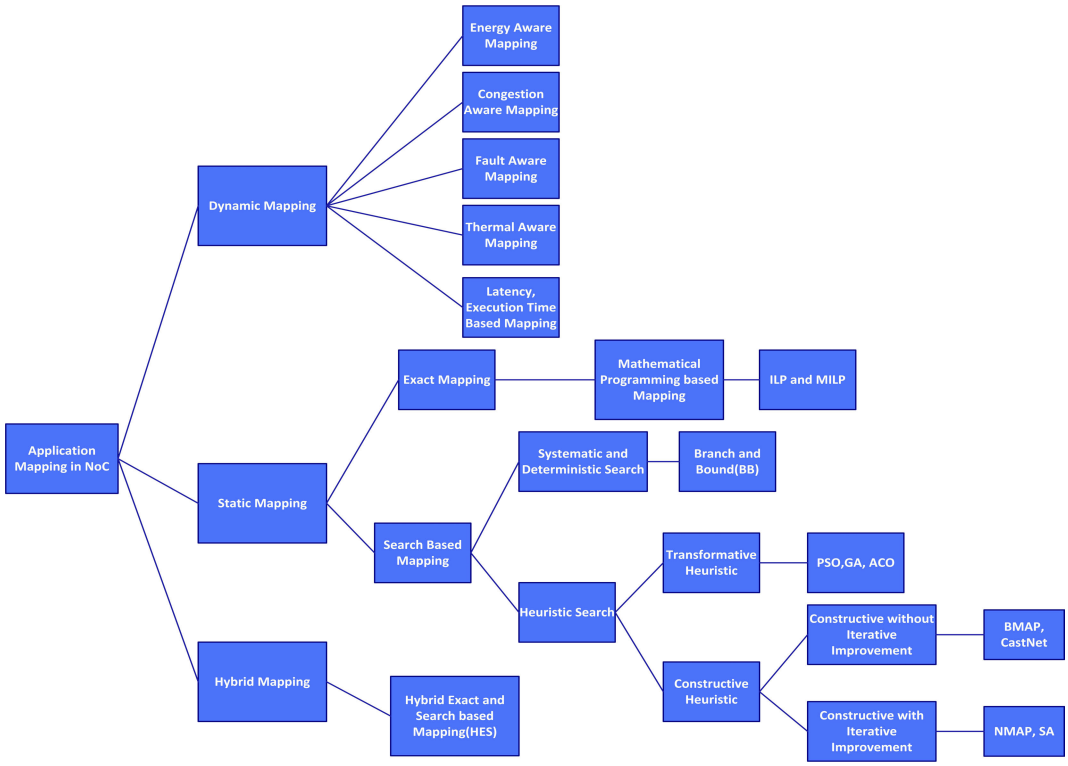


FIGURE 2. Classifications of application mapping techniques.

migration strategies to enhance run-time efficiency. It entails either relocating tasks when the performance obstruction is found or distributing the workload more evenly among the multiple processors. In contrast to task migration, dynamic task mapping allows the insertion of new tasks into the system on the fly. Various heuristic mapping algorithms are prominently used in run-time mappings. These heuristics are rule-based mechanisms that consider certain factors such as task communication patterns, load balancing, and proximity. They help to optimize tasks to PE allocation in real-time systems while ensuring faster communication, and then the system performance is improved.

The primary cost function in mapping techniques is optimizing the occupation of links in NoC systems. Performance improvement can be achieved if a mapping algorithm can minimize NoC congestion. MPSoC architectures can be considered as processing nodes that communicate with one another through a network. The processing nodes or cores can run either hardware or software tasks. The hardware operations are carried out in reconfigurable units or by dedicated IPs. The tasks are allocated to the MPSoC resources, though the number of tasks may surpass the number of available processing elements. One processor may be nominated as a manager processor, which manages and controls the system's resources. When the MPSoC begins its implementation, only the immediately required activities are assigned to the processing elements. When the execution is started, only initially needed tasks are mapped to the system. When a given task attempts to interact with another task that is not yet present, a new task is assigned. The main responsibilities of a multi-processor (MP) are resource control/management, task binding and mapping, task relocation or migration, and managing the reconfiguration process. When tasks begin to execute, the communication requests made by the application tasks are first carried to the MP, and they can dynamically change the configurations. If the destination is not present, the MP executes a dynamic mapping algorithm to map the tasks to the cores.

First free (FF) mapping in [27] finds the first available core and assigns it a task. It is one of the basic and simplest mapping algorithms. The task mapping follows the sequential order from the origin of the NoC, i.e., $(0, 0)$ location, to the end of the NoC platform. The mapping technique does not consider the congestion or cost evaluation while the mapping is performed. In the nearest neighbor (NN) mapping technique, the free node is searched, and the task is allocated to that free node around the node requesting the task execution. The algorithm aims to map the tasks as close as possible to each other within a small area. The FF and NN mapping techniques are discussed in [27] and [28]. Although the tasks in the NN algorithm are mapped closer to one another, this approach does not consider the inter-communication cost among the tasks. Thus, the tasks may be placed at a distance from each other, and then it results in increased overhead. In [29], the authors proposed a smart NN-based task mapping approach. The proposed algorithm divides an architecture into virtual clusters with virtual boundaries such that the tasks of the multiple

applications can share the resources in different clusters. Each cluster provisions one initial task allocated to its central PE. Once the initial mapped task executes and requests for its corresponding communicating task, the algorithm maps the next incoming requested task onto the same PE if the PE has the required capacity. This results in the decreased overhead of communication. By mapping multiple tasks on the same PEs, the approach can be beneficial as it mainly allows mappings of tasks closer enough to each other for the sake of communication overhead reduction.

Communication-aware technique [30] having packing-based NN technique, which investigates the available resources before suggesting the near to communicating tasks on the same PE, is presented. Moreover, the proposed approach prioritizes the application tasks in close proximity to further minimize the communication overhead. The experimentations and analysis show that the proposed technique is more effective in alleviating the congestion in NoC than the other existing alternative techniques. The communication energy mapping technique discussed in [31] maps tasks with higher communication close to each other. The algorithm maps the task having the highest communication volume to the first available PE. The neighboring tasks of the already mapped task get sorted in order of the communication volume. The task with the highest communication is selected and mapped to the core having the smallest distance from the already mapped Processing element or core. Similarly, all tasks are mapped to the nearest available PE.

Consider the application task graph [31] as shown in Figure 3. It contains the task graph showing tasks (circled nodes) from 0 to 12. The communication volume of all the independent tasks, which is computed by all the incoming and outgoing volumes of data at the specific node, is represented in the form of a list (green-colored array/list shown at the bottom of the task graph). Because task 4 has the highest communication volume, i.e., 55, it is first mapped to the mesh-based NoC platform in Figure 3 (shown on the right-hand side by grey interconnected blocks). Its neighbors are nodes 1, 7, and 8, and the node 1 has a higher communication volume, so node 1 is mapped next in the nearest possible location to the already mapped task 4 in the mesh-based NoC. The other two neighbors are 7 and 8, which need to be mapped next. Now, node 8 has higher communication than node 7, so task 8 will be mapped with task 4 next on mesh-based NoC. Then, task 7 will be mapped at the closest possible location. Similarly, all the tasks are mapped based on the communication volume with the already mapped tasks.

Mapping of different Applications-a motivational Example: An example of a NoC-based heterogeneous system consisting of two types of cores is presented in [32]. Multiple performance-constrained applications are concurrently executed on the cores, as shown in Figure 4. Consider such a multitasking multi-core-based scenario in which a computer system user runs an integrated development environment, e.g., a music player, an email

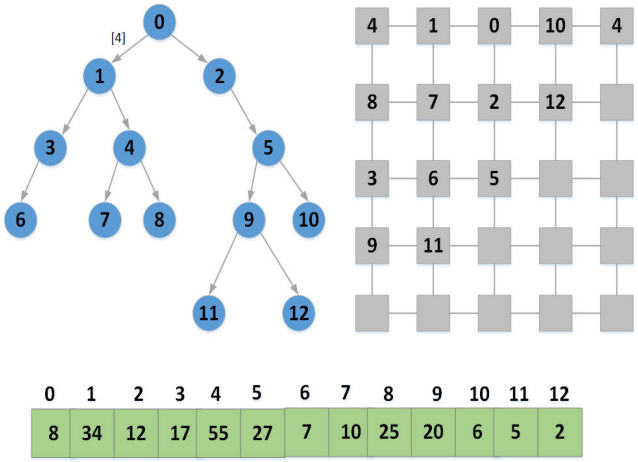


FIGURE 3. Task placement in a communication energy algorithm.

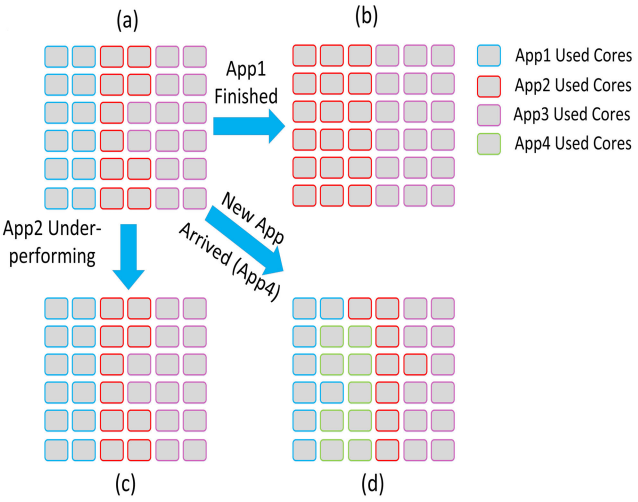


FIGURE 4. Various dynamic execution scenarios, (a) System with 3 running Apps, (b) App1 finishes execution, (c) App2 under-performing needing more resources, (d) New App enters the system.

service such as round cube, or starts downloading a file. The initial mapping for three applications, i.e., App1, App2, and App3, may be decided according to their performance constraints at the design time. While the application tasks execute on cores, three run-time execution scenarios might be possible that need dynamic or run-time mapping: 1) one of the applications finishes its execution; 2) an application(s) may face degradation in performance due to Congestion in the network or variation in the workload, and 3) a new application(s) arrives in the platform to be executed.

In the case where App1 completes its execution, the resources which it occupies can now be dynamically assigned to App2 and App3, which might be helpful for them to finish their execution in a short time, as shown in Figure 4 (b). Dynamic remapping is performed to allocate free cores to the running application tasks to perform their tasks faster. This may enhance the performance and be beneficial in reducing energy consumption because the two applications are able to execute and finish earlier. Dynamic mapping or remapping is also required in the case where there might be

performance degradation due to congestion or variation in workload in the network as presented in [32]. In this case, applications might face different workloads in the execution phase. The contention or workload is monitored, and action is taken so the tasks are remapped to the cores to avoid performance degradation. Figure. 4 (c) shows a case where App2 gets more resources to deal with the performance degradation problem faced during execution. The cores can also be acquired from the over-performing application, which is App3, if there are no free cores available in the system. In the last case, the user launches a new application while the other application tasks are also running and being executed on the cores. Suppose the running applications have acquired all the cores. In that case, the dynamic algorithm might check for available possibilities to remap the current tasks and assign PEs to the newly inserted application tasks to satisfy the performance constraints. Figure 4 (d) shows such a case where new App4 gets added to the system while the other applications, i.e., App1, App2, and App3, are already executing. The dynamic algorithm makes such decisions to allocate PEs of over-performing applications, i.e., App1 and App2, to App4. At the same time, App3 continues to execute with the same number of cores. Dynamic mapping techniques comprise a variety of approaches designed to address the various challenges encountered by run-time systems. Consequently, these techniques can be further categorized based on the specific conditions or circumstances that necessitate the use of dynamic application mapping. One such category is congestion-aware dynamic application mapping. In such dynamic mapping, the application tasks are assigned to the PEs considering the congestion in the network links. Also, once the cores are mapped and executed, any link faces more congestion, and then the tasks can be re-mapped to the other cores having links with lesser congestion. Therefore, dynamic mapping facilitates the development of congestion-aware mapping solutions in run-time or real-time execution of the NoC. Another category is thermal-aware dynamic mapping, which considers the thermal characteristics of the components used in the NoC when mapping application tasks. Dynamic mapping also has significant importance for mapping techniques that adapt based on the occurrence of faults in the cores or routers. So, once the cores on which a task is mapped are prone to any defect or fault, the task is re-mapped to the other available cores so that the system is fault-aware and avoids any permanent fault by using fault-aware dynamic mapping techniques. Dynamic application mapping is also applied to scenarios where the mapping can be adapted at a run-time based on the load balancing performed amongst the network cores. Application mapping techniques are further discussed as follows:

1) ENERGY-AWARE DYNAMIC APPLICATION MAPPING TECHNIQUES

Reduction of energy consumption and management of network load are key challenges in designing multi-core NoC systems. Application mapping plays a vital part in the design of such systems. Efficient and optimal task mapping helps significantly minimize the total energy consumption and communication overhead between interdependent tasks; hence it plays a major role in the performance enhancement of the system.

In [30], the proposed technique investigates the available cores ahead of recommending the adjacent tasks of the application on the same PEs. The technique tries to place tasks in close proximity to deal with and reduce the overhead involved in communication. The proposed algorithm works to decrease energy consumption as well as the total application execution time. However, as the degree of nodes in the application graph increases, more tasks are mapped onto the same processing cores and potentially leads to deadline-based misses. Compiler-based task mapping technique [33] is presented in one of the works, which reduces energy consumption and improves the efficiency of the system. The experimental outcomes and analysis show that energy consumption is calculated based on the scenarios with and without packet routing and task-to-core mapping. The authors in [31] have systematically studied selected run-time application mapping techniques using the same collection of assumptions, platform, and device models. An extension of one of the algorithms, i.e., communication-aware packing based nearest neighbor (CPNN) technique, has also been proposed that helps to minimize communication overhead between the interlinked tasks. In addition, the authors incorporated high-level Petri nets to perform systematic verification and abstraction of the proposed methodology. The proposed mapping algorithm achieves a reduction in communication cost as well as end-to-end packet latency. It also reduces energy consumption for small mesh NoCs.

The authors in [34] proposed an efficient knapsack-based bin-packing technique for workload management that positions tasks so that available processing cores are exploited to the greatest extent possible. An algorithm with task swapping capability has also been presented that attempts to further refine the task placement as decided by bin packing algorithms. The experimentation is performed to improve the energy and network load. An energy-aware based scheme [35] is presented for NoC that aims to obtain the reduction in the communication energy. In the first phase of the technique, the evaluation of the connectivity state of applications at run-time is performed, and the task having the largest amount of communication is picked and mapped. In the next phase, the tasks adjacent to the already selected tasks are sorted by communication volume and allocated to the nearest PEs. In [36], the authors proposed a task assignment scheme while incorporating user behavior. The capability of having a user behavior effect empowers the system to react to real-time changes more quickly, and then the system adapts dynamically according to the user requirements. By using the sequence of events, the algorithm determines the user behavior once it interacts with the system.

Several techniques are proposed and presented for the solution of the task allocation problem while considering communication energy optimization. The incorporation of user behavior in the algorithm further reduces communication energy. However, additional computational overhead is introduced due to the machine learning schemes and techniques for processing user behavior. The Mesh network can be divided into multiple clusters so the applications are able to run in their specific clusters. In some cases, the application's tasks exceed the number of the selected cluster's processors. In such cases, if the tasks from the same application are assigned to different clusters, they may cause performance degradation. This issue is addressed in [37] in which an efficient mapping approach incorporating multiple clusters is presented in which the placement of the tasks belonging to the same application is performed in the same cluster's region. In the first phase of the algorithm, static clustering is performed to collect the data regarding the count of tasks per application. In the next phase, dynamic clustering is implemented to accommodate the desired applications in the clusters. The technique achieves energy reduction while enhancing the performance of the system.

A dynamic cluster size-based technique [38] is presented for a multi-core NoC platform that incorporates distributed resource control and management. The architecture of the NoC is distributed into several clusters of the same size at the initialization phase. The clusters can then borrow cores from neighboring clusters to efficiently assign the application tasks at run-time. The technique effectively balances the workload of NoC and the Energy consumption reduction.

For the application task mapping of tera-scale-based NoC platforms, a reclustering-based approach having decentralized-agent phenomena is presented in [39]. The authors suggested a cluster-based task mapping strategy for task mapping in which the NoC cluster's size changes at run-time accordingly based on the load of the system and the communication-based pattern, thus minimizing overall energy consumption. In [40], a run-time mapping heuristic is proposed based on a tradeoff between processor load and communication volume. The presented architectures are divided into clusters with fixed sizes at design time. Then, a reclustering process can change the cluster size at run-time based on the application mapping. Energy-aware scheduling is investigated in [41], and the scheme is applied to real-time streaming applications executed on edge devices. A revolutionary re-timing-enabled strategy is designed to convert the dependent-based workload to an independent task-based model to free up energy and unused slack in processors. In addition, ARSH-FATI, which is a novel population-based algorithm, is presented that can dynamically switch between exploitative and explorative search approaches in real-time to maximize efficiency. In addition. a contention-conscious earliest edge consistent deadline first (EECDF) scheduling technique is also proposed that jointly performs application task mapping and ordering. An efficient mapping technique [42], which is known as a hierarchical and dependency-aware (HAD) technique, is presented. In this technique, one core is dedicated to being a global manager that runs the HAD scheme. The number of transitions of the tasks is contrasted with the available cores in the initial step, while in the second step, the mean occupied position (MP) is calculated to decide the placement of the cores. In some cases, application tasks are not always executed on allocated PEs simultaneously. Thus, there can be PEs having mapped tasks that are still idle and waiting for the data to be received for processing. These situations may arise in NoC and may lead to low utilization issues of PEs while application mapping schemes are applied. A hierarchical and dependency-aware application mapping algorithm has been reported in [43] that covers the above-mentioned issues and aspects of spatial mapping and task interdependency. The proposed algorithm works such that when an already mapped task completes the execution, the allocated PE is released, and another unmapped task is allowed to be mapped. Thus, fewer PEs are required in the region in the region selection phase. The algorithm helps to increase the elasticity of future tasks and creates an impact on the latency and higher utilization of the resources.

2) RELIABILITY-AWARE OR FAULT-TOLERANT APPLICATION MAPPING TECHNIQUES

The emergence of deep sub-micron-based technology has aggravated various reliability-related concerns in multi-core platforms, including NoC. Due to the quick rise in the elevation of permanent faults, aggressive technology scaling has spotlighted the issues and concerns of Reliability. Many factors, such as accelerated aging effects, including electro-migration, manufacturing, and testing-based challenges, are the major causes of these faults. The overall reliability is also concerned with the existence of soft upsets produced due to the cross talks, coupling noise, and transient faults. To overcome the major issues that can degrade the system in terms of reliability, the above-mentioned factors are quickly becoming an issue to be reckoned with [44]. Considering these faults, the system's reliability has become one of the key challenges in NoC architecture and design [45], due to which extensive research in this area is carried out in today's era.

Various research works presented fault models and categorization related to NoC design. The major problems and factors causing reliability issues have been discussed in the research work [46], [47], [48] fault-tolerant models and various techniques are also presented that relate to the diagnosis and avoidance of faults to intensify Reliability. To deal with faults at the application level, dynamic application mapping techniques have emerged to be prominent and efficient techniques that have contributed to enhancing the overall Reliability of the system. If any core in the NoC becomes faulty, its mapped tasks can be remapped to any other available non-faulty core. So, unlike static mapping, the decision of migrating tasks of the faulty cores to the

available cores can be made at the run-time considering the status of the NoC platform, multiple events triggered, which includes different fault scenarios, availability of the resources or new enabling applications, etc. Failures in single and multiple PEs are addressed by incorporating a general manager [49] that performs the remapping. A highly efficient mapping technique is implemented to identify the initial mapping region in the offline mode. The second step maps applications dynamically by using the optimized algorithm Kuhn-Munkres that remaps applications to PE in a newer region nearer to the initial region. The scheme makes an effort to minimize the communication overhead and the energy while dealing with the faults at the same time.

A dynamic decentralized resource based [50] and application-driven task mapping technique is presented for three-structured-based streaming applications. The technique's major aim is to reduce contention and communication costs while considering faults. To mitigate faults at the systems level, a system-level run-time fault-tolerant application mapping technique [51] is presented. The main objective is to optimize the NoC performance and consumption while considering specific permanent, intermittent, and transient faults in NoC. As a dynamic technique, it identifies spare core placement and performs migration of faulty cores to spare ones, thus reducing the failure contamination area (FCA). The paper emphasizes various scenarios of spare core placement for faulty cores. The proposed technique is highly efficient, and significant throughput improvements are achieved as compared to the techniques not incorporating faults. A methodology for heterogeneous multi-core processors has been proposed in [52] that estimates near-optimal task mappings intending to enhance the factor, mean workload to failure (MWTF). An artificial neural network (ANN) can estimate the vulnerability factor in RISC-V processing cores at run-time to achieve this. An optimized run-time application mapping has been achieved using the proposed technique, mainly targeting better MWTF and MWTF/energy tradeoffs.

In [53], a dynamic fault tolerant mapping technique has been presented that considers the application tasks' temporal attribute and the faults' timing record. Regarding the state of the cores, a fault-aware technique has been presented that reduces the overall communication energy and improves the quality of service for the running applications. A fault-tolerant-enabled mapping scheme is introduced in [54], which proposes the placement of spare cores flexibly in the mesh of tree-based NoC. The algorithm incorporates integer linear programming (ILP) and particle swarm optimization (PSO)-based application mapping under fault scenarios so that the tasks on the faulty cores can be executed at the spare cores in case of failure. The experiments are carried out in static as well as dynamic scenarios to optimize latency and throughput. In [40], a run-time mapping heuristic is proposed based on a trade-off between processor load and communication volume. The proposed architectures are divided into clusters having fixed sizes at design time. Then, a re-clustering process can change the cluster size at run-time based on the application mapping. The presented work aims to increase system reliability while making a trade-off between processors' load and communication volume. Several fault-tolerant application mapping approaches are presented that aim to mitigate permanent faults. They are also implanted in field programmable gate arrays (FPGAs). However, architectural-based aspects are limited in certain approaches. The fault-tolerant-based mapping approach for NoC has been reported in [55] that offers to place the spare cores flexibly onto NoC having torus topology and takes into account the faults as well. ILP and PSO have been presented in order to place spare cores while mapping. In the next phase, the routing algorithm and spare ore placement are performed on FPGA, incorporating real-time fault injection and fault-tolerant mechanism. The algorithm is analyzed and compared with other techniques in dynamic scenarios and is found to be efficient with regard to communication cost and run-time. The experimental flow of the proposed fault-based application mapping is shown in Figure 5.

Fault tolerance core mapping (FTCM) is a system-level fault-based task mapping scheme introduced in [56]. In order to manage faults in a multi-core-based NoC platform, the scheme offers core mapping based on variable-sized applications while taking into account the spare cores taking responsibility for the failed cores' tasks. The mapping is first performed using weighted communication energy and the average distance between the nodes. Secondly, it incorporates the spare core placement strategy. The tasks of the cores having higher failure probability are shifted to the spare cores in the network, thus enhancing the overall reliability.

To achieve reliability in NoC, a hybrid reliability-based mapping approach has been presented in [57] that employs multi-objective based optimization along with reinforcement learning (RL) to provide optimal mapping solution in fault scenarios. The algorithm works in two phases. In the initial step, a set of best remapping solutions are generated offline for various fault scenarios that use a biogeography-based optimization algorithm to reduce energy and migration costs. In the next step, an artificial neural network agent has been trained to select the optimal remapping solution from the already generated ones. Using the proposed algorithm, this algorithm features to recover from the failures at the run-time. This technique proves to be better in terms of reliability and ensures a reduction in overhead that is caused by re-mapping the solution's storage. The RL-based run-time application mapping platform is shown in Figure 6. Many research efforts have been made in order to solve application mapping problems, but they do not always consider other dimensions, such as topology or routing algorithm, in their approach. However, one of the research works [58] delves into fault-tolerant techniques for application-specific NoCs (ASNoCs) while also considering routing algorithms. The work addresses persistent defects in ASNoC interconnection links by offering a meta-heuristic

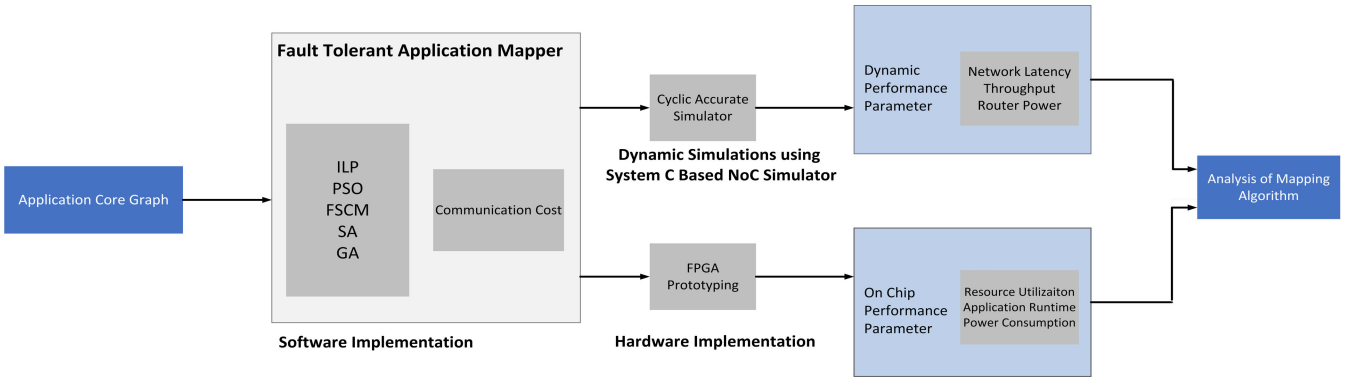


FIGURE 5. Experimental flow of fault-tolerant application mapping onto torus topology based NoC.

approach based on particle swarm optimization (PSO) to deploy spare links and introducing fault-tolerant routing algorithms. The approach is implemented using an FPGA board. The experimental outcomes show considerable gains in performance measures, validating the efficacy of their strategy in comparison to earlier efforts. Overall, the research provides insightful recommendations for improving the efficiency and reliability of ASNoCs in the presence of link problems. Some techniques are developed for the already-known task graphs. However, researchers strive to develop techniques in order to tackle unseen graphs as well. One such technique, i.e., fault-tolerant multiapplication onto regular NoC (FANC) [59], has been proposed, which introduces a fault-tolerant application mapping approach based on Machine Learning (ML). The technique carries adaptability to unseen graphs and topologies while offering robustness as well as an improved convergence rate. It incorporates an ML-based model to extract pertinent information from the search data, integrating it into the search process efficiently. The experimental results prove the technique to be effective in terms of better communication cost, latency, throughput, and power usage. The fault-aware mapping techniques in NoC improve system reliability by seamlessly transferring tasks of faulty cores to the available free cores. However, it entails a trade-off between performance and workload distribution. Energy and power consumption. The techniques are efficiently developed such that they can mitigate faults but, at the same time, require careful consideration of these tradeoffs.

3) CONGESTION-AWARE DYNAMIC MAPPING TECHNIQUES

NoC platforms allow multiple PEs to exchange and process information simultaneously communicated over multiple links. Hence, it is highly desirable to entail such a mechanism or functionality that parallel communication between different numbers of PEs is free of congestion or has the least possible congestion. The occurrence of congestion and network contention within NoC can remarkably affect the entire system's performance in terms of degradation in efficiency and throughput [60], [61]. Network contention can be one of the causes of the start-up time being delayed or the completion time of tasks being held up for longer than expected. Thus, a reduction of network contention within NoC is one of the main concerns of researchers today. Network congestion is one of the major challenges [62] and occurs when some other data transaction occupies a channel or a link, and the remaining data packets have to wait and are queued up until the required link is available.

Dynamic or run-time application mapping can be carried out while considering the congestion and contention among the various cores or links. Different dynamic application mapping approaches are presented in the literature, which attempts to map the task of the applications on the processing cores considering the congestion in the NoC Platform. The work in [27] aims to reduce execution time and congestion by implementing a congestion-aware application mapping approach in NoC-based systems with dynamic variable workloads. Moreover, comparisons of various application mapping algorithms, such as path load (PL), minimum maximum channel load (MMC), and minimum average channel load (MAC), are conducted based on average channel occupation. Furthermore, a clustering approach for mapping multiple applications on multiple clusters is also presented. To reduce congestion and the communication energy in NoC, the author in [63] proposed a task mapping heuristic and a definite switch-based platform that can reduce the congestion occurrence due to simultaneous access to the shared memory in NoC Platform. The proposed technique is limited to the single task execution mapped to each processing element. Only single shared memory is incorporated while conducting experiments for the smaller Mesh sizes such as 4×4. In [29] the author has proposed a dynamic task mapping technique that maps multiple higher communicating tasks onto that same PE or at PEs closer to it. The approach reduces overall communication overhead, congestion in the network, and energy. However, load balancing between the available cores is not considered in the proposed technique. Communication energy-based global optimization is not achieved for all the processing cores, as only the cores nearby are considered for placement on the same PEs. The author in [64] proposed a run-time application mapping technique that employs a pre-processing step, which helps minimize the communication

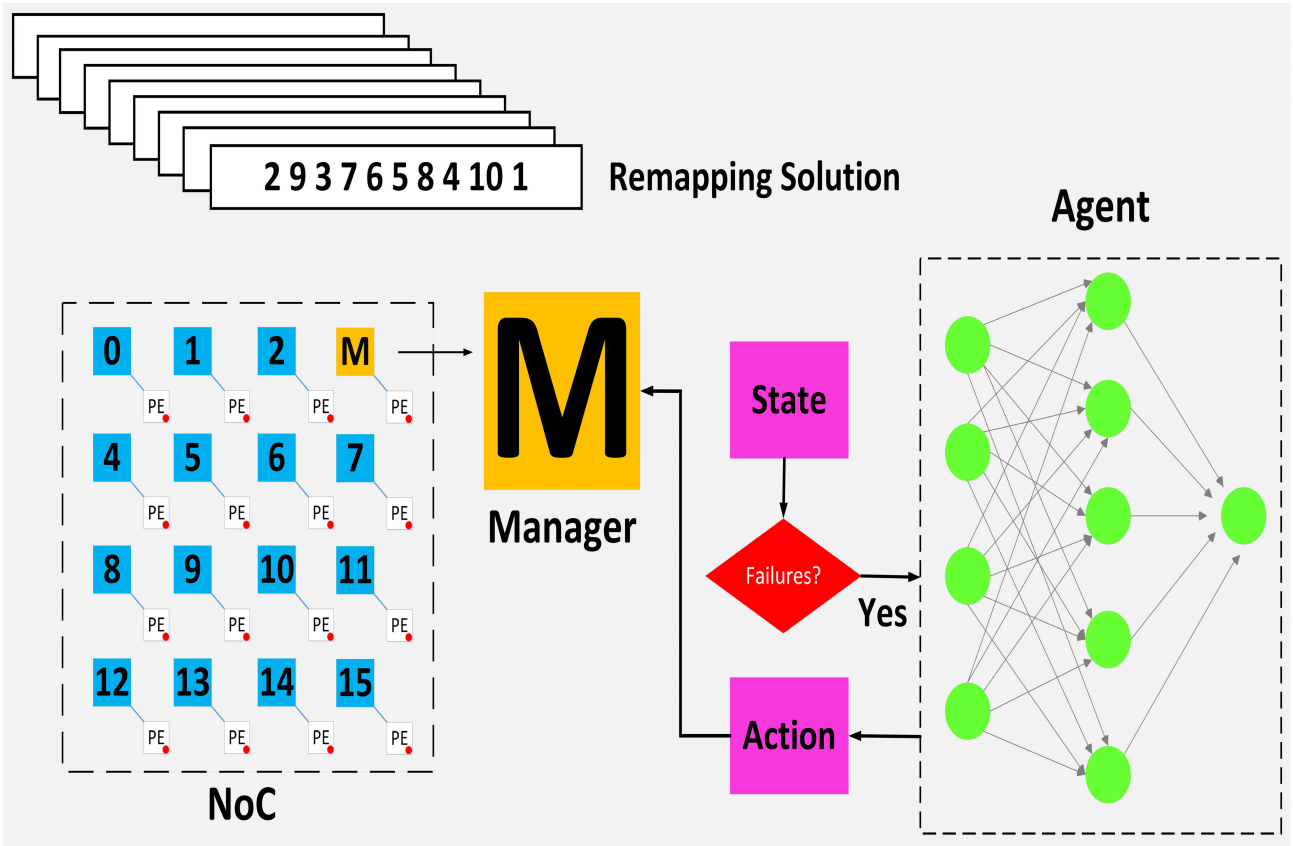


FIGURE 6. RL Based run-time application mapping [57].

overhead and evenly balance the load between the PEs. This dynamic technique helps to enhance the system performance by optimizing application execution time, resource usage, and energy consumption. However, computation time and algorithm execution time are increased due to pre-processing steps in the algorithms. Experiments have been conducted on the smaller NoC sizes with a limited number of applications. To deal with communication overhead and to optimize NoC in terms of congestion, run-time mapping heuristics have been proposed in [65]. In this technique, the application's communicating tasks are assigned to the same processing cores or close to each other within a particular region. The proposed algorithm creates virtual clusters to map the initial tasks. The analysis has been performed with a 8*times*8 NoC, and the work only considers the single task to be mapped on PEs.

Similarly, in [66], a dynamic application algorithm is proposed that considers NoC contention. The presented algorithm is divided into two stages. The first step involves identifying a rectangular area appropriate for mapping the incoming application. Then, in the following steps, application tasks are assigned to processing cores within the specified area of the algorithm based on the communication between the tasks. To reduce internal and external NoC congestion, another run-time application mapping approach is proposed in [67] that attempts to reduce the internal and external NoC congestion. The main drawback of the technique is that for mapping the first task, its highest number of edges is considered. However, the communication of the tasks is not considered for selecting the first task.

In [68], an efficient dynamic task mapping approach is presented, which incorporates congestion speculation (DTMCS). The algorithm not only maps application tasks to the PEs efficiently but also takes into account future traffic patterns based on link utilization. Instead of only focusing on improving the current packet blocking situation, the presented algorithm attempts to improve the overall congestion by the link usage-based speculation window in NoC platforms executing applications with real-time traffic. The analysis is done using a large communication volume with complex traffic patterns, and the presented algorithm shows minimal communication latency. In [69], an optimized platform CAP-W for wireless NoC is presented with congestion-aware features that aim to reduce internal and external congestion. The algorithm consists of three phases. An adaptive routing algorithm capable of balancing the usage of wired and wireless network links is proposed as the first step of the algorithm. In the second phase, a run-time application mapping technique is implemented that minimizes congestion probability. In the third phase, a task migration scheme is implemented that considers the dynamic variation regarding application behaviors. The analysis shows

significant improvement in congestion in the platform over processing cores of wireless NoC.

A congestion-aware mapping scheme has been proposed in [70], which optimizes the core mapping using link betweenness centrality to relieve congestion from heavily loaded NoC links. The data volume traversing through the NoC links is monitored. The mapping is performed based on that information to optimize the NoC in latency and congestion. Mapping and scheduling of the applications in dynamic schemes either do not consider communication workload or approximate network congestion while not considering its effect on the performance of tasks regarding their timings. An enhanced technique for dynamic task allocation as well as the scheduling [71] is developed, having the capability of link contention awareness, which together assigns the tasks on the PEs while scheduling the communications on the links. The proposed technique can select a route to mitigate network contention while the application tasks are being executed. The analysis shows the proposed technique's efficacy regarding deadline satisfaction and improving the latency of the executed applications. A task mapping technique [67], for a proximate mapping region, is presented that attempts to minimize internal as well as external congestion. For initiating the allocation process, the algorithm chooses the first processing element with the largest number of free neighbors close to it. However, the technique can cause disintegrated free processing cores as the algorithm selects the initial PE with sufficient neighboring PEs for the allocation of all application tasks. In [72], a contiguity adjustable based square allocation (CASqA) technique has been proposed that aims to allocate the application's task onto the multi-core architecture within a square region. The algorithm allows enough flexibility to customize mapped processors while focusing on the application mapping solution's internal congestion and energy dissipation reduction. Congestion-aware mapping techniques address various contention-related issues in NoC at loaded links. These techniques effectively mitigate congestion occurrences, minimize computational complexity, reduce remapping overhead, and help to achieve balance amongst system objectives such as energy efficiency and response time.

4) THERMAL-BASED OR TEMPERATURE-AWARE DYNAMIC MAPPING TECHNIQUES

Due to technological advancements, it is now achievable to integrate a significant range of IP cores, processing elements, and memory modules on a single chip. However, the close placement of the cores has produced thermal hotspots within the multi-core platform. Ensuring and establishing thermal protection in real-time systems is a difficult challenge, particularly in scenarios where multiple applications are in execution at different times. Due to technological advancements and technology scaling, integrated circuits are becoming vulnerable to many thermal-based hazards, which may occur due to increased power density and many other factors, along with the close spacing among the cores. Critical temperature rises have a number of major consequences, including a decrease in transmission speed, a decline in reliability, and an increase in energy usage. The research work in [73] demonstrated that with a 10 percent rise in temperature, the latency of on-chip interconnects rises by 5 percent. The rise in temperature of the on-chip elements can be the major cause of the permanent failures that deteriorate the system performance and are irreversible. In certain scenarios, the failed core may cause the system to totally collapse. Therefore, the temperate rise has become a significant concern in NoC. In multi-core real-time-based platforms, the applications are dynamically entered into the system for the execution [74].

Therefore, application mapping is one of the major steps in the NoC multi-core platform design that influences the overall system performance. While the applications arrive and enter the system to be executed, they exhibit variable thermal attributes that are unknown ahead of the execution. Static approaches are not fruitful as they consider the thermal profiles in the offline mode instead of the run-time scenarios. Therefore, in real-time platforms, it is of major concern to use dynamic thermal-based task mapping strategies for managing the temperature of on-chip elements while dynamically executing the application on the NoC multi-core-based platform. Thermal aware mapping ensures that the on-chip interconnect meets the thermal specifications. However, the time involved in the migration of the tasks might negatively impact the efficiency of the application in terms of timing. Hence, thermal aware task mapping is a challenging task in terms of real-time dynamic framework.

To deal with the thermal instability issues of the multi-core platforms, various system-level thermal mapping techniques, scheduling techniques having thermal-aware characteristics, and dynamic voltage scaling techniques may be employed. In [74], an improved thermal-based dynamic approach incorporating task mapping and scheduling has been presented that uses a non-profiling scheme for NoC platforms. The presented thermal-based technique maintains the integrated circuit's thermal safety specifications by jointly considering the threshold-based thermal control scheme along with a run-time task (re)allocation technique. The research work reduces the chip's peak temperature compared to the other dynamic approaches and reduces the application's average packet delay while satisfying the task deadlines. Figure 7 shows the overview of the system model.

Communication latency and chip peak temperature optimization may be difficult in certain dynamic scenarios. In certain cases, mapping tasks close to each other can reduce the latency, but it may lead to poor heat dissipation. To resolve this issue, an efficient dynamic application mapping scheme is proposed in [75] that reduces the application's communication latency and running time while managing the thermal attributes. This technique first selects the 3D cuboid area within the NoC network for each incoming

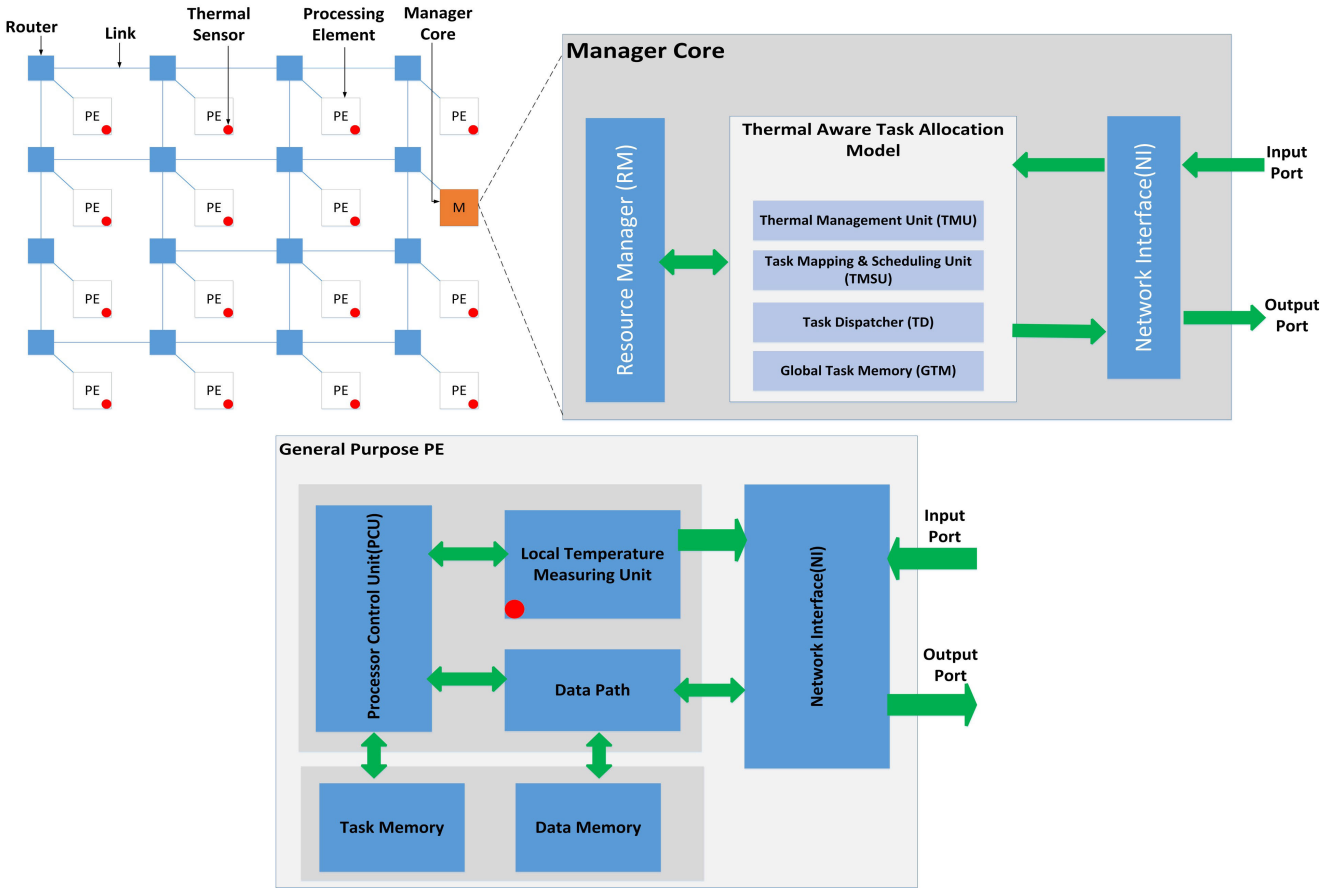


FIGURE 7. Overview of system model [74].

application by having a certain number of occupied vertical layers along with its distance to the heat sink. This is done to allow optimization regarding both the performance and the peak temperature. In the later phase, exact core regions are determined, and task to core mapping algorithm executes for the final mapping. This algorithm gives optimized results in terms of total running time and communication time while also taking into account the thermal constraints.

In [76], a power-thermal task allocation technique has been proposed with a reconfigurable NoC framework. Balanced mapping is provided for multiple applications in multi-core CPU-GPU-based NoC systems in the dark silicon era. Task mapping and resource configuration are formulated by incorporating LP to look for the optimum solution. Distributed resource management with an efficient, balanced mapping technique is proposed to minimize the thermal hotspots and improve the run-time performance considering the thermal budget and communication constraints of multi-core NoC. For the purpose of comparisons, implementation of minimum path contiguous task mapping has also been implemented. A fault self-testing/ recovery mechanism [77], [78], is proposed, and a remapping technique is implemented that ensures the migration of tasks in dynamic scenarios to cope with permanent Faults of the processing elements. The proposed work presents two modules, and one is the task migration module incorporated in each PE that is responsible for extracting critical data from faulty PEs. The other module is run-time-based task remapping, which searches for the optimum remapping for that given fault scenario. The work reduces power consumption, maintains load balancing, and avoids thermal hot spots.

To obtain a balanced thermal profile while the applications are in the execution phase, a distributed thermal management framework is presented in [79]. A neural network is implemented for the thermal throttling, and the algorithm assesses the task migration based on benefits between neighboring cores. Although the algorithm performs well in terms of thermal management of the cores, but additional functionality is needed for the neural networks of each core to be trained. For real-time chip temperature estimation, a thermal management-based technique is presented in [80], which uses a support vector machine (SVM). Dynamic thermal control on overheated cores entails voltage/frequency minimization to keep the chip's temperature within limits. The real-time mapping and scheduling problem incorporating a temperature-aware mechanism is methodized in [81]. It proposes an efficient phased steady-state mixed ILP approach that performs effective scheduling and allocation decisions based on the thermal profiles of the cores to affect the chip peak temperature directly. Application task

assignment and scheduling technique are presented that allows the designers to solve the large instances-based mapping problem but at the cost of accuracy for the running time improvements.

In [82], a highly efficient technique to enhance the lifetime and reliability has been presented that dynamically migrates the workload in a predictable fashion such that assignments and executions of the tasks are completed on time. While this strategy considers the availability of the destination core for meeting the task's deadline, its scalability regarding the platform size is constrained. In [83], a temperature-aware task-mapping algorithm is presented that can prevent hotspots by using adaptive multi-threshold values to achieve a highly uniform thermal distribution. The algorithm measures the temperature of cores and switches tasks as soon as the core's temperature exceeds the normal temperature of the chip. If a core reaches an absolute maximum temperature, it is turned off. Thus, by achieving an extremely evenly thermal distribution and stopping hot cores from reaching a certain threshold, the algorithm enhances the overall reliability of the NoC.

In [84], the proposed work uses an adaptive thermal-based scheduling technique for multi-core-based systems (A-TMS) that incorporates a number of run-time controllers to modify the service levels of the applications based on core temperature and device usage. To consider the criticality regarding the timing constraints and the tasks, the study in [85] employs a process based on workload reassignment. It employs a thermal-based application mapping approach which reduces temperate overheads and minimizes energy, enhancing the system's reliability. However, to avert and deal with the thermal emergencies that occur on the running core, the algorithm has the provision to drop the tasks to maintain reliability. In the run-time, reconfiguration of the mapping solution is done on the basis of dynamic variations in the system in terms of resources and latency of the system. In [86], the mapping reconfiguration approach is used such that a set of mapping solutions is produced at the design stage for a number of applications considering resource usage and performance characteristics. The tasks are migrated to the other PEs based on predictable reconfigurations. At the design stage, low latency-based migration routes having minimum allocated overhead between the source and target mappings are calculated. These are used at the run-time to apply mapping reconfiguration for a certain application if needed. The proposed algorithm is efficient in terms of low allocation overhead and acceptable latency. To deal with the temperature-related issues in 3D NoC platforms, an earlier constraint programming-based approach from 2D NoC is extended and applied for efficient core mapping in 3D NoC Architecture presented in [87]. In this approach, the processing cores are categorized and set to low power, medium power, and high power. They are assigned to certain places at mesh-based NoC to reduce overall communication cost and computational time while considering the thermal constraints. Moreover, tasks are scheduled on the cores based on the matching performance level to optimize the overall computation time. Thermal-aware application mapping approaches offer various advantages such as improved system reliability, optimal thermal management, and enhanced performance. However, the implementation of these techniques increases the complexity of the design, requires additional resources, and adds greater overhead in terms of power and latency. In addition, certain thermally-aware mapping techniques may encounter scale difficulties with larger and more complex systems, limiting their applicability in certain circumstances.

5) SCHEDULING TECHNIQUES INCORPORATED IN NOC SYSTEMS

The performance of the system can be depicted using certain metrics, such as time consumed by computation and communication in NoC to run and execute a certain application. It is one of the major factors involved in the design. The computation time is normally dependent upon the design of the IP cores. However, the communication time is not only determined by the routing protocol, but task scheduling methods also have a major impact on it. The tasks must be executed in a certain well-established order to achieve optimal performance. This comes under the umbrella of a scheduling problem. The optimized solution to such a problem is using an efficient scheduling algorithm. A novel adaptive approach has been proposed [97], that has the ability to handle dynamism for multiple applications on the NoC platform under throughput or energy consumption constraints. First, at the design time, a set of non-dominated schedules for the tasks and their mappings are computed. During the run-time, a lightweight, adaptive scheduler is implemented to reconfigure the allocation of run-time applications on the basis of the status of a network for available cores. The run-time adaptive scheduler chooses the optimum topology for the specific application and performs task mapping using the Tetris algorithm. The algorithm works in an adaptive manner that can alter and remap the mapping of the applications at the run-time, thus delivering enough power to achieve the application's constraints.

contention and energy-aware task mapping and edge scheduling (CA-TMES) approaches called CA-TMES-quick search and CA-TMES search are presented in [88] that are used to allocate tasks to PEs in the NoC platform. The inter-voltage frequency island (inter-VFI) communication and overall makespan reduction have been achieved by mapping the tasks on the processors of VFI. In [89], an efficient scheduling technique has been proposed that schedules the periodic tasks onto the NoC architecture while employing an earlier deadline first task ordering policy. The computation of the useful temporal attributes is also performed in the presented work, allowing the tasks to satisfy the relevant constraints while monitoring the communication cost. A heuristic technique for improving energy and contention-based tasks with precedence constraints has been

developed in [90]. The analysis shows that the algorithm optimizes energy usage using efficient task mapping and scheduling approaches. However, re-timings based on task level to reduce overall energy have not been considered. An energy-aware based scheduling approach is proposed for real-time applications in [41]. A re-timing-based approach is designed to convert the workload having dependencies into an independent model to get the resources. A population-based scheme known as ARSH-FATI has also been proposed that can dynamically switch between the explorative and exploitative capabilities of search schemes for performance tradeoffs. In addition, an algorithm that is communication and contention aware based on EECDF is also developed for efficient scheduling between the tasks. In [91], an aging-aware scheduling method that considers the negative bias temperature instability (NBTI) aging effect and its influence on the task scheduling framework in NoC is presented. The task scheduling model has been developed with the help of the NBTI aging model, which helps to analyze the core degradation. Considering the performance degradation and the communication overhead between the cores, a meta-heuristic scheduling technique is developed based on PSO. The technique proves to be better regarding the completion time and enhanced throughput of the system compared to non-aging-aware scenarios. To deal with real-time scheduling in multi-core systems, an algorithm for the homogeneous platform is proposed in [89], with reconfigurable dependent and periodic tasks to be assigned to multiple processing cores in a NoC-based system. Reconfigurations are automated operations that allow the addition and/or removal of application tasks and their shared messages in the NoC platform. The proposed platforms incorporate a periodic task model and a novel scheduling strategy for the computation of useful temporal attributes that allow the tasks to satisfy the constraints, and the technique worked while managing the communication cost in NoC. Scheduling-based NoC application approaches can enhance resource efficiency, task distribution to cores, and communication overhead. However, challenges include scheduling complexity, overhead, and dynamic workload variations, or system uncertainty.

6) DYNAMIC MAPPING TECHNIQUES FOR PERFORMANCE ENHANCEMENT (TECHNIQUES IMPROVING LATENCY AND TIME, LOAD BALANCING, INCREASING THROUGHPUT, ETC.)

In [92], the best neighbor (BN) task mapping technique has been introduced that attempts to place the tasks on the best possible neighbors. The technique helps to reduce the execution time. Performance evaluation with different benchmark applications has been presented, including VOPD, MPEG-4, etc. A homogeneous NoC platform-based run-time dynamic task mapping scheme for heterogeneous NoCs is presented in [93]. The algorithm incorporates the User behavior during the application task assignment process, allowing the system to adapt more effectively according to the changes and adjust suitably at run-time. In [94], another homogeneous NoC platform-based two-step run-time approach has been proposed that is able to assign various applications tasks on the NoC in an incremental fashion while incorporating multiple voltage levels. The first step is to find a suitable area of NoC for mapping followed by a greedy dynamic-based heuristic that performs the final mapping tasks. The proposed algorithm helps reduce the overall execution time, and significant communication energy has also been estimated to be reduced. A distributed agent-based mapping technique to address the dynamic application mapping problems in NoC is proposed in [95].

In the agent-based mapping technique, agents are actually constituted of a small task that manages and keeps a record of the information about resources and their state. These agents can be placed on any node in the platform. The agents interact with one another to explore computing elements suitable for the mapping purpose. Global agents (GAs) are incorporated to keep all the clusters' global information. In contrast, the mapping requests for the local clusters are maintained by cluster agents that share these requests and information with the GAs. In an attempt to minimize communication, run-time mapping time, and task relocation time, a spiral mapping algorithm known as dynamic spiral application mapping (DSM) is proposed in [96]. The technique searches for suitable placement of tasks so as to assign the tasks in a spiral fashion, taking the start from the center towards the boundaries of the NoC-based network. The work in [65] describes the blend of hardware and software processing elements formulated on MPSoC Platform. The proposed architecture nominates one of the processing nodes out of all the nodes to act as a managing node whose responsibility is to keep the status of resource management, task binding, task resettling, and reconfiguration control. On the basis of the above-mentioned information, the manager node assigns new mapping to the application tasks. A mapping algorithm based on two phases is also proposed. In the initial phase of the algorithm, preliminary mapping on the basis of the first available place in the network is accomplished. New tasks that request allocation are mapped in the second step by employing the run-time technique. Similar to this, more comprehensive work with new mapping methods has been proposed in [30].

In [97], a distributed run-time mapping technique is introduced, which incorporates a strategy involving local managers and controllers. The technique is further divided into three sub-classes: the initial core, controller-based core, and manager core. The initial cores are the selection of cores for mapping purposes. While the resources are managed by the manager core, the controller core observes the status and activity of distinct subregions of the NoC. Another proactive region selection scheme (MapPro) presented in [98] is the first node selection-based technique, and the technique aims to reduce execution time and congestion. This scheme exploits the idle time between the two consecutive mapping requests to look for suitable candidate mapping regions. A dynamic multi-agent and distributed resource

management-based technique, i.e., DistRM, is proposed in [99]. In this approach, instead of a centralized placement of the manager core for managing the tasks, the technique incorporates the agents that are distributed over the NoC platform to manage the task mapping service to available cores. A mapping approach that incorporates online and offline applications to be mapped on the same network has been proposed in [100]. The major objective of the technique is to minimize the average latency by mapping critical run-time applications on nearby nodes. On the other hand, non-critical applications are distributed across the available NoC platform nodes. Non-contiguous application mapping may be able to improve the throughput of NoC. However, increased communication distance causing network delays may occur due to mapping tasks done in the non-contiguous region. The mapping in the work [30], [31], [65] is performed as early as possible basis in which the new task is assigned to the PE as soon as the task enters the system for the execution. In [95], a decentralized run-time agent-based mapping technique method for heterogeneous NoCs is presented in which cluster-based agents are incorporated for mapping. A cluster agent has the charge to perform mapping within a specific cluster while the global agent monitors and keeps the information that relates to all the clusters and works with one of the specific cluster agents for the task mapping activity in that specific cluster. Despite the fact that this scheme decreases total network traffic for deciding the network's current state, the computational and communication overhead has risen due to the interactions performed between the multiple agents. A communication-centric predictive task scheduling algorithm is presented in [101], which has characteristics of prediction and is communication-aware at the same time. For the purpose of task prioritization and processor selection, a prediction matrix in PPTS is used. In the processor selection phase, a method incorporating communication-aware features is adopted so to choose the processor that minimizes the overall communication. The analysis of experimental results shows better results when applied to randomly generated graphs and real-world application graphs. A summary of the different dynamic application mapping schemes is presented in Table 1. Although many research efforts are being made to enhance performance once the applications enter or leave NoC Platform. However, most fragmentation of applications is not considered in approaches. The research work [102] explores the concept of dynamic run-time mapping in NoC-based many-core architectures. The contribution of the work involves defragmenting fragmented applications to reduce migrations and improve mapping quality without requiring a threshold for the process. It achieves fine-grain defragmentation by evaluating freed resources after each task, enhancing overall performance. The study shows promising improvement in system speed and resource efficiency by implementing run-time mapping defragmentation. Figure 8 shows mapping solutions before and after defragmentation is applied. A hybrid approach incorporating both design time and run-time requirements is proposed in the research work HyDra [103], which aims to reduce communication cost and energy overhead. The approach generates multiple mappings during design time, prioritizing communication optimization. The dynamic mapping phase utilizes design time mapping and reconfigures them on the basis of run-time availability of the resources and application requirements. The complete Framework of the Hydra dynamic mapping algorithm is presented in Figure 9. In [104], a machine learning framework has been presented to provide mapping solutions that are near to optimal with improved latency and throughput. The algorithm is used to intelligently learn from past experience and recommend the NoC system to get to its performance, power, and reliability objective. In [105], mathematical modeling of application mapping problem is done using linear programming that incorporates computation and communication capacity along with power-related constraints as well. Later, simulated annealing (SA) and GA algorithms have also been implemented, having power budget constraints related properties to solve the large-scale application mapping problems. In [106], run-time task allocation with reconfigurable NoC is proposed in which distributed resource managers are incorporated that dynamically reconfigure voltage and frequency levels of NoC. A mapping algorithm MinEnergy is presented to minimize overall power and energy. A linear optimization model for best solution is presented and a cutting-edge optimal mapping approach is presented for comparison. In [107], problem of large-scaled applications are mapped onto heterogeneous NoCs with an effort to minimize the hotspots. First, a linear programming model is implemented to search for an optimal solution, followed by a heuristic algorithm for faster search space exploration at design time and run-time in extensive NoCs. Latency and throughput-related improvement is carried out while reducing the hotspots as well.

III. EXERIMENTAL SETUP

To evaluate the application mapping algorithms for multi-core NoC systems, different benchmarks are used. Random applications are also used in the mapping techniques that are generated by the TGFF tool, and the random applications serve to evaluate and compare algorithms. Similar to the TGFF Tool, other tools like it (TGG) or TGF benchmarks (E3S) are also broadly utilized to generate task graphs of different sizes for comparison purposes. Experiments are also conducted on real-time benchmarks to evaluate the performance of the distinct applied techniques. Other than VOPD, MPEG-4, picture in picture (PIP), H263 Encoder and many other real-time application benchmarks, PARSEC and SPLASH are also widely used benchmarks for the evaluation of run-time mapping techniques. PARSEC benchmarks are used in NoC research to assess the performance of on chip communication networks in terms of real-time and parallel workloads and also support multithreaded applications. SPLASH is also used for the comprehensive evaluation of network bandwidth, latency,

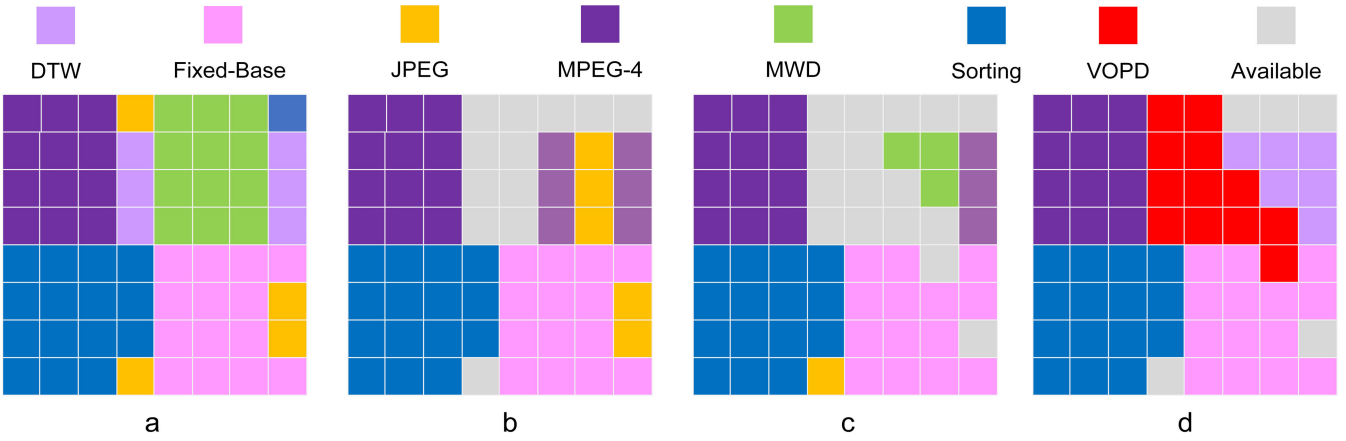


FIGURE 8. Mapping states before defragmentation (a), after defragmentation (b,c), and with a new application mappped after defragmentation (d).

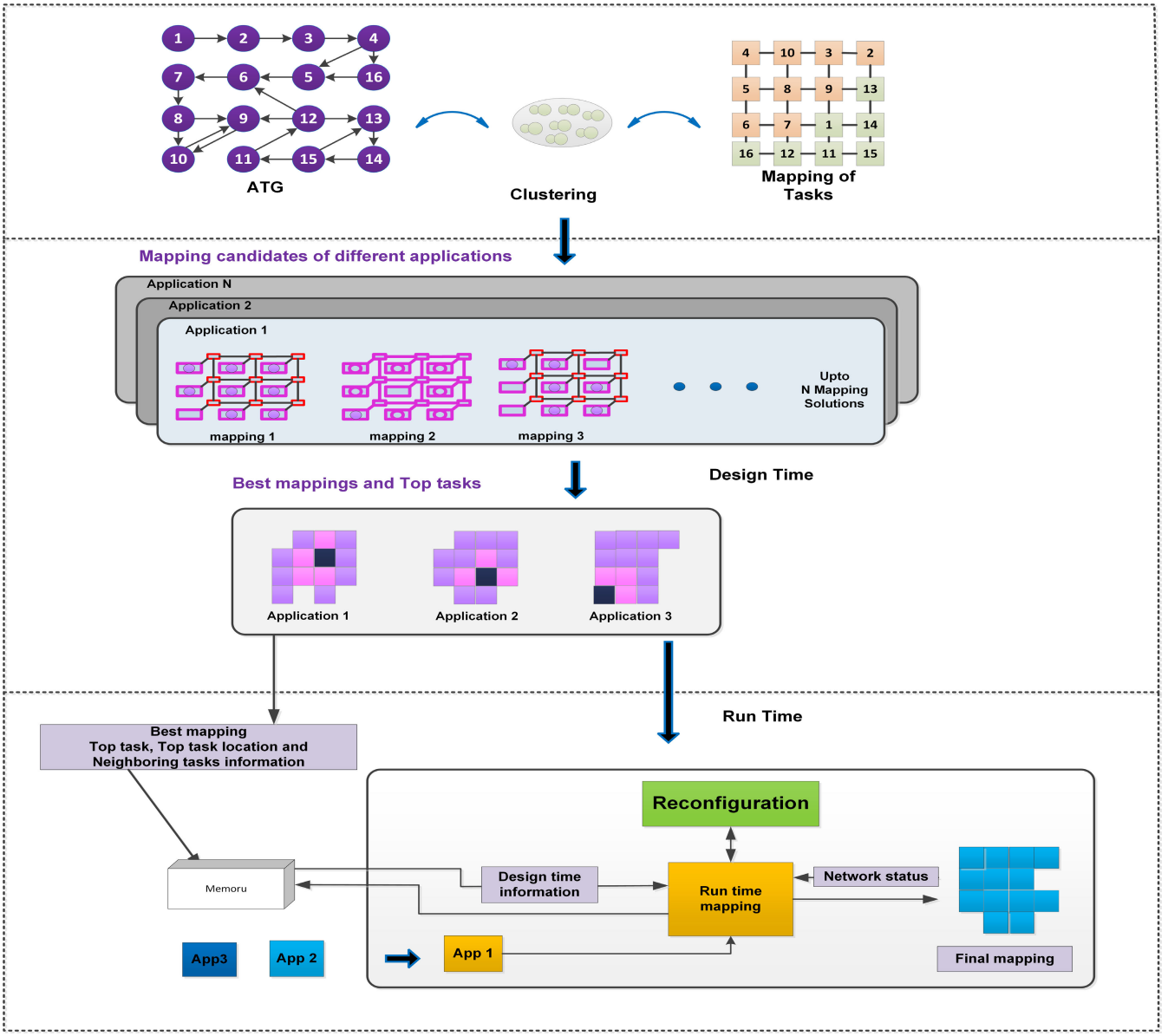


FIGURE 9. Hydra-a hybrid mapping framework.

and workload related metrics. NoC Simulators employ a variety of real world and synthetic benchmarks to evaluate the performance. NoCTweak [108] is the highly parameterizable NoC based open source simulator that is used for early investigation of energy and performance efficiency of multi-core based NoC platform. NoCTweak can be used to evaluate based on multiple performance metrics, including latency, power, energy consumption, throughput, and link bandwidth. Its integration with an ORION Tool [109], [110] allows the designers to find power at different CMOS nodes according to certain power models. The tool also allows the use of Synopsys Design Compiler to synthesize RTL designs of the router components. With certain traffic patterns, post layout power data of the elements can be given to it for evaluation of power and energy parameters. However, NoCTweak has certain limitations regarding certain design capabilities, such as topology, mapping optimization, and precise energy and performance models. The shortcomings of certain NoC tools are addressed in one of the modified NoCTweak simulators, ENoCTweak, developed to evaluate various task mapping techniques. The mainframe is based on the NoCTweak simulator while integrating other NoC components and simulation platforms. In the design of dynamic multi-core NoC platforms, Graphite simulator [111] is used to simulate NoC-based systems. Graphite is a parallel and distributed based simulation platform that performs high-level architectural assessment and evaluation, and it is built on Pin dynamic binary based instrumentation tool [112] and supports various protocols and types of networks. To deal with and evaluate the performance of multiple simultaneous running applications, an Application Scheduler as well as Multi-application synchronization modules can be incorporated in Graphite. Garnet is also one of the widely used NoC Simulator integrated with gem5 simulation framework. It replicates the behavior of real interconnected networks, which allows researchers to closely emulate and evaluate on-chip network's performance. It mainly provides a versatile platform for the study of NoC behavior, traffic patterns, and topology, that aids in advancement in chip design and performance.

IV. MATHEMATICAL MODELS

Mathematical models are developed to evaluate and assess multiple performance parameters, including communication cost, energy, network latency, throughput of the system, and power usage. NoC-based communication cost is evaluated by:

$$Cost = \sum_{i,j} [B_{t_i,t_j} \times N_{i,j}], \tag{1}$$

where B_{t_i,t_j} is bandwidth between the tiles t_i and t_j and $N_{i,j}$ denotes the Manhattan distance. In detail, the Manhattan distance between any two nodes (x_i, y_i) and (x_j, y_j) in NoC platform can be given by

$$N_{i,j} = |x_i - x_j| + |y_i - y_j|. \tag{2}$$

Average latency of the NoC architecture is given by

$$Lt_{av} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N_i} \sum_{j=1}^{N_i} Lt_{i,j} \tag{3}$$

where $Lt_{i,j}$ is latency of packet j from one tile to other, N is the total number of processors in the NoC and N_i is the number of packets that a processor receives after i warm-up time. The average throughput of the network, Th_{av}, is given by

$$Th_{av} = \frac{1}{N(T_{sim} - T_{wrm})} \sum_{i=1}^{N} N_i, \tag{4}$$

where T_{sim} is the simulation time and t_{wrm} is the warn-up time.

The average power, Pw_{av}, is expressed by

$$Pw_{av} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N_i} [\alpha_{i,j} \cdot Pw_{act,j} + (1 - \alpha_{i,j}) \cdot Pw_{inact,j}] \tag{5}$$

where $Pw_{act,j}$ is the active power, $Pw_{inact,j}$ is the inactive power of component j, and $\alpha_{i,j}$ is the active measurement of the component j in the router i.

The energy that each packet uses in terms of average energy, En_p, is given by

$$En_p = \frac{(T_{sim} - T_{wrm})}{(N \cdot N_p)} \sum_{i=1}^{N} \sum_{j=1}^{N_i} [\alpha_{i,j} Pw_{act,j} + (1 - \alpha_{i,j}) \cdot Pw_{inact,j}], \tag{6}$$

where $(N \cdot N_p)$ stands for the total count of packets that traveled in the NoC network.

In NoC based multiprocessor based platform, the energy consumption is divided into two major parts 1) computational energy 2) communication energy. Computational energy is consumed in the PE and determined by the task's attributes and the PE's efficiency [113]. The energy used by the network elements for the transfer of the data amongst the processors is termed communication energy. According to the bit energy based model presented in [113], the communication energy is represented in terms of bit energy consumption. Bit energy is the energy used to transmit one bit of information from one tile to another tile. Bit energy between tile i and j is given by

$$EBit_{i,j} = n_{i,j} ERBit + (n_{i,j} - 1) ELBit + 2ECBit, \tag{7}$$

where $EBit_{i,j}$ is the energy required to transmit one bit of information from tile i to tile j. $ERBit$ is the dynamic energy consumed to transmit a bit through the router components. $ELBit$ signifies the dynamic energy consumed to transmit the bit on the links between the two tiles. $ECBit$ represents the bit energy consumed on the link that occurs between the router and PE. Finally, $n_{i,j}$ characterizes the number of hops that the bit passes while travelling between the tiles i to j, and it is expressed by (2).

Thus, the total communication energy consumption by the NoC platform can be represented by

$$E_{NoC} = K[n_{avg}ERBit + (n_{avg} - 1)ELBit + 2ECBit] \quad (8)$$

where K represents the total number of bits transferred between source and destination tiles and n_{avg} represents the average number of hops.

Computational energy consumed by each PE, E_j is computed by

$$E_j = n_j^{run} E_{run} + n_j^{idle} E_{idle} \quad (9)$$

where n_j^{run} represents the number of cycles for which the processor p_j is in the running state while n_j^{idle} represents the number of cycles in which p_j is in idle state [37]. E_{run} and E_{idle} give a process's running and idle states energy consumption, respectively. Total computational energy consumption E_{comp} consists of energy consumed by all the PEs and is given by

$$E_{Comp} = \sum_{j=1}^{n} E_j \quad (10)$$

Hence, the total system energy consumption is calculated by accumulating the computational energy consumption and communication energy consumption represented by

$$E_{total} = E_{Comp} + E_{NoC} \quad (11)$$

A. MIGRATION COST

If a task *ti* from a source core that is placed at tile *i* is migrated to a destination core placed at tile *j*, the migration cost for migrating a task *ti* is given by

$$MC_i = size_i \times n_{i,j} \quad (12)$$

where $size_i$ refers to the size of task in bytes, $n_{i,j}$ is the number of hops a packet travels between tile i and tile j and it is calculated by (2).

Lets consider that set $MSet$ consists of all the migrations to be done such as $MSet = \{m1, m2, ..., mn\}$. Every item $mk \in MSet$ carries certain characteristics tk, src, dst. Let tk task that needs migration, src being the source tile, and the destination tile is given by dst where tk is migrated.

The accumulated migration cost for all the migrations in the set given above is calculated by

$$MCost_{total} = \sum_{mk \in MSet} MC_k \quad (13)$$

To formulate the migration energy usage, we can use (7) in a modified form as represented in (13). In order to compute the migration energy, $MCost_{total}$ is converted into a number of bits obtained. Then, the total migration cost illustrated in number of bytes is given by

$$E_{mig} = \left(\frac{MCost_{total}}{n_{avg}} \times 8\right) \times (n_{avg}ERBit + (n_{avg} - 1)ELBit + 2ECBit) \quad (14)$$

B. RECONFIGURATION TIME

Reconfiguration time is one of the most significant parameters to be considered, as slow reconfiguration may affect the performance of the run-time system due to delays in the reconfiguration process. Performing an in-depth analysis of worst case reconfiguration timings is important to ensure that the real-time demands of an application remain unaffected by the reconfiguration process. Reconfiguration time is represented in (15):

$$R_t = S_t + C_{st} + M_{gt} \quad (15)$$

where we represent reconfiguration time with Rt, St represents the Suspension time of the task, Cst is the search time of core for migration and Mg is the migration time.

Suspension time St is the measurement of duration for halting all task executions and transferring their messages to the designated Processing Elements(PEs) while ensuring no ongoing application messages are being transferred. Core Search time is defined as the duration for searching the appropriate core for the reconfiguration, and migration time represents the time it takes to move the tasks from the current halted core to the newly identified core.

C. EXECUTION TIME

Execution time comprises the mapping time (the time to find the placement), communication time, computation time at the PE, and waiting time when no resource is free in the platform.

D. RUN-TIME OVERHEAD

We define run-time overhead as the time the manager core takes to execute the algorithm on the arrived applications until it completes the task allocation process.

V. FUTURE WORK AND CHALLENGES

Future trends and key challenges are discussed in this section that will take the application mapping perspective and techniques into the next generation. Some trends are discussed, which are as follows:

A. APPLICATION MAPPING INCORPORATING MACHINE LEARNING

Machine learning schemes and approaches have been largely employed in a range of massive and dynamic data-intensive domains over the last few years. ML techniques have shown exceptional results when applied to advanced problems in network traffic-based predictions, resource control and management, classification, routing decisions, congestion control, and fault management. Because of these ML-based approaches' learning capability and features, they are a great option for resource allocation for High-precision-based prediction, and accurate design parameters can be obtained using sophisticated approaches incorporating deep neural network (DNN) and ML techniques. High-precision-based prediction and accurate design parameters can be obtained using sophisticated approaches incorporating deep neural

network (DNN) and ML techniques. ML approaches are classified based on their ability to adapt to modifications or changes that occur. The training process of these schemes is intensive in terms of data and consumes much time, and it often necessitates extensive research during the design phase. These methods are learned using predefined platforms having certain workloads and are improved during the execution by learning new improvements in workload behavior to increase prediction accuracy. Extensively used ML based models, including convolutional neural networks (CNNs) and DNNs are the ANNs that take various inputs in series and learn through the automatic procedures to approximate a goal function that is best suitable based on them.

Application mapping approaches in multi-core-based platforms significantly affect the system's efficiency. The presented research work in [114] addresses energy efficiency by estimating energy utilization and the system performance resulting from data and task allocation on computational elements. In this case, the prediction models are created by employing carefully chosen supervised ML approaches applied to raw data developed offline from the systems executions. The approaches comprise SVD, ANNs, and adaptive boosting (AdaBoost). Similar work has been proposed in [115] that discusses a learning methodology that can be applied to many core systems to forecast mapping-related performances. The prediction models that emerge will help to enhance dynamic resource allocation decisions. In [116], the authors explored different NoC architecture and mapping approaches for DNN. An effective mesh NoC platform and a load balancing-based efficient mapping solution has also been presented. The analysis and experimentation demonstrate the efficacy of the presented techniques for accelerating DNNs while improving power and energy in NoC. In [117], an automated mapping technique that starts at the single core level with minimum run-time and memory accesses has been presented as the main optimization objectives. The technique is then applied to an appropriate multi-core mapping scheme and tested using a flexible system-level simulation with an NoC system. Design space exploration is performed such that the widely used CNNs AlexNet and VGG-16 are mapped to the multi-core platforms with varying core counts and computing capacity per core to explore the trade-off. In [118], an optimization approach in multi-core neural network chip helps map logical cores onto physical cores while preventing deadlock. An algorithm incorporating Simulated Annealing along with two deadlock-free constraints is developed for the mapping-related problem while the weighted communication between the tasks is considered to be the objective function. Two multi-layer perceptrons (MLPs), as well as two CNN-based applications, are used for the sake of evaluation of multi-core neural network platforms. Experimentation and analysis revealed that the presented algorithm saves power consumption and routing time for the purpose of inter-core communication, hence improving the routing diversity as well. The algorithm proved to be indispensable for multi-core neural network-based chip design as it significantly improves the time and power consumption of hardware implementing neural network algorithms. In [119] a neural network framework incorporating RL is implemented to solve the application mapping problem. It is based on the algorithm RL-MAP, which has a policy network for providing the mapping sequence. There is a critic network that provides an estimation of the communication cost of the mapping sequences. The policy distribution is updated by the actor network in some specific direction as per the critic's suggestions. Unsupervised data is used to train the proposed RL-MAP algorithm to estimate the different combinations of the cores to reduce the overall communication cost of the NoC Platform. Finally, a 2-opt local search-based algorithm is used to improve the solutions further. The RL-based technique is better than the heuristic algorithms in terms of communication cost and run-time.

B. APPLICATION-MAPPING IN 3D NOC

To cater to the inter-communication and complexity-related demands of upcoming high density multi-core platforms and architectures, 3D NoCs have been revolutionized as an optimal solution. The 3D architecture has allowed the integration of various layers of processing elements on a single device, which has helped enhance the system performance and reduced the device area, power usage, and signal transmission delay. However, the architecture feasibility of 3D NoC designs faces numerous key challenges, including thermal-related problems, area overhead of 3D routers, high-power usage, and increased complexity and expense of vertical connection deployment. The efficient and better thermal control and management of 3D structures is a key challenging task and necessitates research into effective methodologies. Application mapping has a remarkable effect on NoC efficiency and energy usage. In [75] an efficient run-time mapping scheme has been presented that reduces both the application running time and the communication latency while considering the thermal-constrained design for 3D NoCs.

In [87] an earlier constraint programming (CP) technique for heterogeneous 2D NoC architecture is applied to a 3D model in this article. The approach selects core types from a range of low-power to high- power core types and assigns them to suitable locations on the mesh in a single step, minimizing total computation and communication expense while meeting temperature constraints. To obtain the optimized results, in addition to the core location problem, task assignments should be scheduled in accordance with the cores with matching performance levels to reduce total completion time. In [120], the authors have presented in depth a number of notable research practices in 3D NoC system architecture and testing.

TABLE 1. Summary of literature review on dynamic mapping approaches.

Ref.	Category	Mapping Technique	Experimental Setup	Benchmarks	Comparison	Optimization Goal
[30]	Energy Based	Dynamic Mapping	ModelSim Co-Simulation Platform	TGFF	NN CNN	Energy Consumption and Improvement related to total execution time
[33]	Energy Based	Compiler Based Application Mapping	Simics and Orion	SpecFP2000	-	Improvement- Energy Consumption
[31]	Energy Based	CPNN	Heterogeneous NoCr (HNOCS) simulator	-	FF NN CPNN SNN	Reduced communication cost, average hop count, and end-to-end Latency
[34]	Energy BASED	Knapsack based bin packing algorithm	Customized Simulation Framework	-	FF NN SC	Energy consumption and network load
[35]	Energy Based	Energy aware online mapping algorithm	Noxim	Barnes FFM-3 MPGdec MPGenc Ocean-contiguous-partitions Water-spatial	FF NN	Reduction in Communication Energy
[36]	Energy Based	User Behavior Based Task Allocation Algorithm	C++ Based Simulation Model	E3S	Arbitrary contiguous allocation Method	Reduced Communication Energy
[37]	Energy Based	Clustering Based Application Mapping	Noxim	-	NN SNN CPNN	Latency, Energy and Power Consumption
[38]	Energy Based	Distributed Resource Management with Dynamic Clusters	-	MPEG Multispec Synthetic Application	Centralize Mapping Approach	Reduced Execution Time and Energy
[41]	Energy Based	ARSH-FATI	Cortex Processor based Simulation Environment incorporating Samsung Exynos 5422	Automatic Target Recognition (ATR) JPEG-encoder Office Auto Industry-1 MP3-decoder Auto Industry-2 Consumer-1 Consumer-2	CA-TMES-Search and CA-TMES-Quick	Reduced Energy Consumption
[49]	Fault Aware/ Reliability Based	Dynamic Remapping based Algrithm	-	VOPD MPEG-4	SA	System Reliabilityy
[50]	Fault Aware/ Reliability Based	Self Embedding Algorithm	Java NoC simulator	E3s TGFF	-	Improved Fault Tolerant System
[51]	Fault Aware/ Reliability Based	FARM	C++ Based Simulation Model	VOPD, MPEG4 PIP Multi-Window Display Application	NN	Throughput Improvements in Fault Scenarios
[52]	Fault Aware/ Reliability Based	Fault Tolerant Based Mapping	C++ Based Simulation Model	TGFF	FTS FTM FR FARM FASA FADRA	Reduced Energy Consumption Quality of Service Improvement
[55]	Fault Aware	ILP-PSO Based Algorithm	cycle accurate NoC simulator	PIP MPEG-4 MWD h263 Encoder VOPD TGFF	FSCM SA GA	Communication Cost Network Latency Throughput Router Power Consumption
[56]	Fault Aware	FTCM	Noxim	TGFF	FARM FASA	Improved Communication energy
[63]	Congestion Aware	PL Algorithm	-	-	FF NN	Execution Time and Congestion Reduction

TABLE 1. *(Continued.)* **Summary of literature review on dynamic mapping approaches.**

[64]	Congestion Aware	Communiication Aware Algorithm	-	H.264 decoder MP3 decoder Motion-JPEG decoder 2D Wavelet transform codes	FC BN SA	Reduced Congestion and Power Consumption
[29]	Congestion Aware	CA	HNOCS	-	FF NN	Reduced Channel Load and End to End Latency
[65]	Congestion Aware	Computation and Communication Aware Algorithm	Model-Sim	-	SNN	Reduced Execution Time Resource Usage Energy Consumption
[66]	Congestion AWare	Heterogeneous based Maping	ModelSim	-	BN NN	Reduced Execution Time and Average Channel Load
[68]	Congestion Aware	CoNA	System Based Simulation Environment	-	FF NN INC	Reduction in Average Network Latency
[69]	Congestion Aware	DTMCS	Noxim	TGFF E3S	PL	Overall Congestion and Average communication latency
[70]	Congestion Aware	CAP-W	XMulator	SPLASH-2	NN BN INC CAP-W	Reduced Congestion
[72]	Congestion Aware	Contention-awareness Algorithm	C++ based simulator	TGFF	CA-NN PL DTMCS	Task Deadline Satisfaction Improvement and Communication Latency
[73]	Congestion Aware	CASqA	SystemC Based Simulator	-	NN CoNA	Improved Network Performance And Reduced Energy Consumption
[75]	Thermal Aware	Dynamic Mapping and Scheduling Based Technique	C ++ based simulator	TGFF	ATM DEAMS TAPP	Reduction in chip peak temperature Reducing the Average Packet Delay
[76]	Thermal Aware	Thermal Based and Defragmentation Based Technique	C++ NoC Based Simulator	SPLASH-2 PARSEC	B2T FL	Reduction in Communication Cost Reduction in total running time
[77]	Thermal Aware	LP Based task-resource co-allocation with reconfigurable NoC Framework	Gem5 and Garnet	E3S	Minimum-Path (MP) BalancedMap (BM)	Reduction in Latency and Throughput
[78]	Thermal Aware	System level Fault Tolerant Approach	-	M-JPEG encoder and a H.264 decoder	-	Reduced Execution Time and Overhead
[80]	Thermal Aware	Thermal Aware Task Migration Technique	Hotspot	SPEC 2000	ARMA	Reduced hotspots Reduced migration overhead
[84]	Thermal Aware	Temperature-Aware Adaptive Task-Mapping	Heterogeneous platform Based Simulation	-	Conventional mapping and temperature-aware scheduling algorithms	Enhanced Reliability
[85]	Thermal Aware	A-TMS	Hotspot based Simulation	-	LP	Stable Performance and Maximized /System Utilization
[86]	Thermal Aware	Adaptive reliability Enhancement Strategy	Sniper 6.0 multicore simulator	SPLASH-2 and PARSEC	-	Error mitigation with Optimized power and thermal overheads.
[87]	Thermal Aware	Technique incorporating mapping reconfiguration	-	E3S	Off-line Allocation Approach.	Low allocation overhead and affordable latency
[88]	Thermal Aware	Constraint Programming (CP) Methodology	-	MCSL	-	Reduction in computation time and communication cost under thermal constraints
[129]	Scheduling Based	Adaptive Mapping and scheduling Technique	-	-	Hybrid Approach.	Energy Saving

TABLE 1. *(Continued.)* **Summary of literature review on dynamic mapping approaches.**

[90]	Scheduling Based	Reconfiguration Based Technique	OptimalMappingTasks Simulation Tool	Task-Generator based benchmark	EDF-based algorithms ILP	Reduced Energy and Power Consumption
[41]	Scheduling Based	ARSH-FATI	Matlab	E3S	CA-TMES-Search CA-TMES-Quick	Energy-Efficiency
[93]	Misc Techniques	Best Near Mapping Technique	VHDL Base Simulation Model	VOPD MPEG4 MWD RBERG	FF	Reduced Execution Time
[94]	Misc Techniques	User Aware Dynamic mapping	C++ Based Simulator	E3S	-	Improvement in Communication Cost
[95]	Misc Techniques	Energy and Power Aware Incremental Mapping	C++ Based Simulator	E3S TGFF	NN	Improvement in Communication Cost
[96]	Misc Techniques	ADAM-Runtime Agent Based Dynamic Mapping	Xilinx Vertex 2 FPGA Based Simulation Model	VOPD MPEG4 MWD	Nearest Neighbour(NN)	Reduced Execution Time and Computational Effort
[97]	Misc Techniques	Dynamic Spiral Mapping DSM	NA	-	FDSM PDSM	Reduced configuration time
[103]	Misc Techniques	-	System-C based simulation model	-	NN BN	Communication overhead Reduction
[99]	Misc Techniques	Run-time mapping technique	Intel Single-Chip Cloud	-	DistRM DRM	Reduced Communication overhead
[100]	Misc Techniques	Proactive region selection strategy MapPro	NOXIM	VOPD MPEG-4	INC SHiC NN	Reduced Average Latency
[101]	Misc Techniques	Multi-agent-based approach DistRM	C based simulation model	-	Centralized	Improvement in Communication overhead
[102]	Misc Techniques	Mixed-criticality online task mapping	NOXIM	UAV MPEG-4 VOPD	-	Improvement- Average Latency

C. APPLICATION-MAPPING IN WIRELESS NOCS

Due to the higher performance, low energy dissipation, higher bandwidth, low latency, and flexible topology configuration, Wireless NoC (WiNoC) has appeared as an optimistic future on-chip platform. It is an emerging technology that is progressing at a fast pace to become an alternative communication infrastructure for the standard wired NoC. While the chip cores are scaled for large NoC systems, wired-based links may restrict network performance and efficiency. Multi-hop information transfer between far-apart nodes can also result in very high latency and high energy and power usage in NoCs. To deal with these growing needs to further extend the NoC size, new interconnect paradigms for NoCs based on radio frequency (RF) linkage [121] are introduced. WiNoC designs have been proposed as a prospectively scalable interconnect platform that uses wireless interconnects between the PEs for communication to reduce communication latency and energy consumption. Since all the PEs in WiNoCs share the wireless channels, optimized mapping and task migration/ scheduling techniques significantly reduce the congestion and hence enhance the performance in NoC. In [122] a dynamic task mapping algorithm (DAMA) is proposed for WiNoC that aims to minimize the congestion. DAMA works in three major phases: node selection, task mapping, and neighborhood allocation. During the initial phase, the first most appropriate node is selected to map the task, which is mapped on the PE in the second step. In the next phase, the remaining tasks are assigned to the available Processing cores so the overall mapping is optimized and aims to reduce congestion. Though experimental analysis presents that DAMA is proven to be effective for minimizing internal as well as external congestion probability, however, task migration is not done. In [69] and [123], the authors have proposed a congestion-aware WiNoC-based platform, CAP-W, which attempts to improve both internal and external congestion. The work discusses the shortcomings of congestion-based routing algorithms and explores that the wireless channel usage will be increased considerably in a case where an optimized mapping strategy is not employed. To enhance the system's performance to a certain extent, the accomplishment of an efficient application mapping strategy is an essential part, along with adaptive routing for optimized results. The proposed scheme, i.e., a CAP-W platform, is a triple-layer platform mainly composed of three parts: mapping, migration, and routing layers. The mapping layer works to

minimize the congestion probability. The main responsibility of the mapping layer is the selection of a suitable core as the initial candidate core and finding the suitable task to be mapped onto that selected PE. It then allocates the remaining tasks on the basis of contiguity. While taking into account the dynamic variation of the applications, the mapping layer attempts to improve the congestion situation by modifying the primary task mappings. Moreover, the routing layer separates the short and long-distance communications between cores to balance the utilization of both the wired and the wireless networks. A novel technique for the design of a congestion-aware application-centric WiNoC platform has been proposed in [124]. MoT-based topology has been used as a communication framework to use the advantages of both mesh and tree topologies. Optimization related to performance enhancement is considered at the same time while long distant wire/wireless links are used in the MoT topology for a given application. Moreover, a congestion-aware approach with an adaptive routing algorithm is presented to reduce the congestion probability. The analysis presented in the research work shows that the presented work improves network throughput, energy consumption, and latency.

Various Articles and research studies present an RL-based and ML-based framework, for application mapping problems for systems at a run-time. Despite this advancement, challenges and gaps remain, prompting future research into broader NoC topologies, real-time energy, congestion, thermal-awareness, and fault-tolerant mapping approaches while incorporating ML-based approaches to optimize NoC. Many fault-aware mapping solutions are developed to minimize communication overhead during core failures. However, an inherent trade-off persists between communication cost and system performance, prompting further exploration. Subsequent investigations could focus on devising precise methodologies that incorporate scheduling considerations into fault-tolerant application mapping for diverse NoC topologies, thereby addressing existing gaps and advancing overall system reliability. Future directions also include exploring fault-tolerant frameworks with reconfigurable techniques for better performance and also investigating applicability in 3D NoC environments. Advancements in WiNoC-based application mapping will focus on adaptive algorithms, real-time reconfiguration, and energy-aware strategies, enabling dynamic optimization in the face of evolving workloads and hardware capabilities for next-generation high-performance computing systems. Hybrid mapping and reconfiguration strategies exist for NoC, involving design time allocation, exploration, run-time mapping, and reconfiguration based on optimal design time mappings. However, future directions include potential enhancements for more sophisticated NoC-based MPSoC platforms, such as heterogeneous core systems and hierarchical wireless nodes, expanding the applicability to further dynamic workload distribution, other NoC Topologies, and real-time scenarios.

VI. CONCLUSION

The paper reviews the application mapping techniques for the multi-core NoC platform. Application task mapping is one of the significant research areas to optimize NoC architecture, and various algorithms have been implemented to optimize the performance metrics. In this survey, we have tried to spotlight dynamic application mapping approaches. The mapping techniques' classification has been presented with a primary emphasis on the dynamic scenarios and their run-time implementations. The dynamic mapping algorithms are performed at the system's run-time, and the tasks can dynamically be adjusted according to specific criteria or conditions of the network, such as network load, resource utilization, and congestion in the links or routers. In dynamic approaches, one of the challenges is to reduce the time spent on mapping decisions at run-time, which adds to the overall time overhead on the system. There is a need to develop efficient mapping techniques to make mapping decisions at run-time, so greedy algorithms are mostly used to search for the solution space locally and efficiently. Moreover, thermal-aware and fault-aware dynamic mapping approaches are also discussed in which the mapping decisions are made according to specific fault- and thermal-based scenarios. Experimental setup and a few simulators are discussed, along with performance metrics used for comparing and analyzing the algorithms.

REFERENCES

[1] S. K. Roy, R. Devaraj, and A. Sarkar, "SAFLA: Scheduling multiple real-time periodic task graphs on heterogeneous systems," *IEEE Trans. Comput.*, vol. 72, no. 4, pp. 1067–1080, Apr. 2023.

[2] S. K. Roy, R. Devaraj, and A. Sarkar, "Contention cognizant scheduling of task graphs on shared bus-based heterogeneous platforms," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 2, pp. 281–293, Feb. 2022.

[3] G. Michelogiannakis and W. J. Dally, "Elastic buffer flow control for on-chip networks," *IEEE Trans. Comput.*, vol. 62, no. 2, pp. 295–309, Feb. 2013.

[4] A. Agarwal, C. Iskander, and R. Shankar, "Survey of network on chip (NoC) architectures & contributions," *J. Eng., Comput. Archit.*, vol. 3, no. 1, pp. 21–27, 2009.

[5] G. Dimitrakopoulos, E. Kalligeros, and K. Galanopoulos, "Merged switch allocation and traversal in network-on-chip switches," *IEEE Trans. Comput.*, vol. 62, no. 10, pp. 2001–2012, Oct. 2013.

[6] C. Celik and C. F. Bazlamacci, "Effect of application mapping on network-on-chip performance," in *Proc. 20th Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process.*, Feb. 2012, pp. 465–472.

[7] R. Pop and S. Kumar, "A survey of techniques for mapping and scheduling applications to network on chip systems," School Eng., Jönköping Univ., Jönköping, Sweden, Res. Rep. 4, 2004, no. 4.

[8] K. Paramasivam, "Network on-chip and its research challenges," *ICTACT J. Microelectron.*, vol. 1, no. 2, pp. 83–87, Jul. 2015.

[9] R. Marculescu and P. Bogdan, "The chip is the network: Toward a science of network-on-chip design," *Found. Trends Electron. Design Autom.*, vol. 2, no. 4, pp. 371–461, 2007.

[10] S. Kundu and S. Chattopadhyay, *Network-on-Chip: The Next Generation of System-on-Chip Integration*. New York, NY, USA: Taylor & Francis, 2014.

[11] R. Marculescu, J. Hu, and U. Y. Ogras, "Key research problems in NoC design: A holistic perspective," in *Proc. 3rd IEEE/ACM/IFIP Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES ISSS)*, Sep. 2005, pp. 69–74.

[12] R. Marculescu, U. Y. Ogras, L.-S. Peh, N. E. Jerger, and Y. Hoskote, "Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 1, pp. 3–21, Jan. 2009.

[13] P. K. Sahu and S. Chattopadhyay, "A survey on application mapping strategies for network-on-chip design," *J. Syst. Archit.*, vol. 59, no. 1, pp. 60–76, Jan. 2013.

[14] A. K. Singh, M. Shafique, A. Kumar, and J. Henkel, "Mapping on multi/many-core systems: Survey of current and emerging trends," in *Proc. 50th ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, May 2013, pp. 1–10.

[15] T. Bjerregaard and S. Mahadevan, "A survey of research and practices of network-on-chip," *ACM Comput. Surv.*, vol. 38, no. 1, p. 1, Jun. 2006.

[16] N. Kadri and M. Koudil, "A survey on fault-tolerant application mapping techniques for network-on-chip," *J. Syst. Archit.*, vol. 92, pp. 39–52, Jan. 2019.

[17] D. Belkebir and A. Zga, "Mapping and scheduling techniques in NoC: A survey of the state of the art," in *Proc. Int. Conf. Netw. Adv. Syst. (ICNAS)*, Jun. 2019, pp. 1–6.

[18] W. Amin, F. Hussain, S. Anjum, S. Khan, N. K. Baloch, Z. Nain, and S. W. Kim, "Performance evaluation of application mapping approaches for network-on-chip designs," *IEEE Access*, vol. 8, pp. 63607–63631, 2020.

[19] Y.-K. Kwok and I. Ahmad, "Static scheduling algorithms for allocating directed task graphs to multiprocessors," *ACM Comput. Surv.*, vol. 31, no. 4, pp. 406–471, Dec. 1999.

[20] R. P. Dick, D. L. Rhodes, and W. Wolf, "TGFF: Task graphs for free," in *Proc. 6th Int. Workshop Hardw./Softw. Codesign. (CODES/CASHE)*, Mar. 1998, pp. 97–101.

[21] R. P. Dick and A.-H. Liu, "Automatic run-time extraction of communication graphs from multithreaded applications," in *Proc. 4th Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES ISSS)*, Oct. 2006, pp. 46–51.

[22] Y. Xue and P. Bogdan, "Scalable and realistic benchmark synthesis for efficient NoC performance evaluation: A complex network analysis approach," in *Proc. Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES ISSS)*, Oct. 2016, pp. 1–10.

[23] S.-H. Lim, S. M. Lee, S. Powers, M. Shankar, and N. Imam, "Survey of approaches to generate realistic synthetic graphs," Oak Ridge Nat. Lab., Oak Ridge, TN, USA, Tech. Rep., 2015.

[24] Y. Zhu, Y. Du, Y. Wang, Y. Xu, J. Zhang, Q. Liu, and S. Wu, "A survey on deep graph generation: Methods and applications," 2022, *arXiv:2203.06714*.

[25] M. Bayati, J. H. Kim, and A. Saberi, "A sequential algorithm for generating random graphs," *Algorithmica*, vol. 58, no. 4, pp. 860–910, Dec. 2010.

[26] E. Carvalho, C. Marcon, N. Calazans, and F. Moraes, "Evaluation of static and dynamic task mapping algorithms in NoC-based MPSoCs," in *Proc. Int. Symp. Syst.-on-Chip*, Oct. 2009, pp. 87–90.

[27] E. Carvalho, N. Calazans, and F. Moraes, "Heuristics for dynamic task mapping in NoC-based heterogeneous MPSoCs," in *Proc. 18th IEEE/IFIP Int. Workshop Rapid Syst. Prototyping (RSP)*, May 2007, pp. 34–40.

[28] E. Carvalho, N. Calazans, and F. Moraes, "Congestion-aware task mapping in NoC-based MPSoCs with dynamic workload," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Mar. 2007, pp. 459–460.

[29] A. K. Singh, W. Jigang, A. Kumar, and T. Srikanthan, "Run-time mapping of multiple communicating tasks on MPSoC platforms," *Proc. Comput. Sci.*, vol. 1, no. 1, pp. 1019–1026, May 2010.

[30] A. K. Singh, T. Srikanthan, A. Kumar, and W. Jigang, "Communication-aware heuristics for run-time task mapping on NoC-based MPSoC platforms," *J. Syst. Archit.*, vol. 56, no. 7, pp. 242–255, Jul. 2010.

[31] T. Maqsood, S. Ali, S. U. R. Malik, and S. A. Madani, "Dynamic task mapping for network-on-chip based systems," *J. Syst. Archit.*, vol. 61, no. 7, pp. 293–306, Aug. 2015.

[32] K. R. Basireddy, A. K. Singh, B. M. Al-Hashimi, and G. V. Merrett, "AdaMD: Adaptive mapping and DVFS for energy-efficient heterogeneous multicores," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 2206–2217, Oct. 2020.

[33] G. Chen, F. Li, and M. Kandemir, "Compiler-directed application mapping for NoC based chip multiprocessors," *ACM SIGPLAN Notices*, vol. 42, no. 7, pp. 155–157, Jul. 2007.

[34] T. Maqsood, N. Tziritas, T. Loukopoulos, S. A. Madani, S. U. Khan, C.-Z. Xu, and A. Y. Zomaya, "Energy and communication aware task mapping for MPSoCs," *J. Parallel Distrib. Comput.*, vol. 121, pp. 71–89, Nov. 2018.

[35] B. Xie, T. Chen, W. Hu, X. Tang, and D. Wang, "An energy-aware online task mapping algorithm in NoC-based system," *J. Supercomput.*, vol. 64, no. 3, pp. 1021–1037, Jun. 2013.

[36] C.-L. Chou and R. Marculescu, "Run-time task allocation considering user behavior in embedded multiprocessor networks-on-chip," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 1, pp. 78–91, Jan. 2010.

[37] K. Gaffour, M. K. Benhaoua, S. Dey, A. K. Singh, and A. E. H. Benyamina, "Dynamic clustering approach for run-time applications mapping on NoC-based multi/many-core systems," in *Proc. 2nd Int. Conf. Embedded Distrib. Syst. (EDiS)*, Nov. 2020, pp. 15–20.

[38] G. Castilhos, M. Mandelli, G. Madalozzo, and F. Moraes, "Distributed resource management in NoC-based MPSoCs with dynamic cluster sizes," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Aug. 2013, pp. 153–158.

[39] Y. Cui, W. Zhang, and H. Yu, "Decentralized agent based re-clustering for task mapping of tera-scale network-on-chip system," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2012, pp. 2437–2440.

[40] M. Mandelli, L. Ost, G. Sassatelli, and F. Moraes, "Trading-off system load and communication in mapping heuristics for improving NoC-based MPSoCs reliability," in *Proc. 16th Int. Symp. Quality Electron. Design*, Mar. 2015, pp. 392–396.

[41] U. U. Tariq, H. Ali, L. Liu, J. Panneerselvam, and J. Hardy, "Energy-efficient scheduling of streaming applications in VFI-NoC-HMPSoC based edge devices," *J. Ambient Intell. Hum. Comput.*, vol. 12, no. 11, pp. 9991–10007, Nov. 2021.

[42] C.-H. Huang, C.-Y. Chen, and H.-Y. Huang, "Hierarchical and dependency-aware task mapping for NoC-based systems," in *Proc. 11th Int. Workshop Netw. Chip Architectures (NoCArc)*, Oct. 2018, pp. 1–6.

[43] C.-H. Huang, "HDA: Hierarchical and dependency-aware task mapping for network-on-chip based embedded systems," *J. Syst. Archit.*, vol. 108, Sep. 2020, Art. no. 101740.

[44] D. DiTomaso, A. Kodi, and A. Louri, "QORE: A fault tolerant network-on-chip architecture with power-efficient quad-function channel (QFC) buffers," in *Proc. IEEE 20th Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2014, pp. 320–331.

[45] C. Constantinescu, "Trends and challenges in VLSI circuit reliability," *IEEE Micro*, vol. 23, no. 4, pp. 14–19, Jul. 2003.

[46] P. P. Pande, C. Grecu, A. Ivanov, R. Saleh, and G. De Micheli, "Design, synthesis, and test of networks on chips," *IEEE Design Test Comput.*, vol. 22, no. 5, pp. 404–413, May 2005.

[47] Y. He and G. Chen, "An inclusive fault model for network-on-chip," in *Proc. IEEE 11th Int. Conf. ASIC (ASICON)*, Nov. 2015, pp. 1–4.

[48] B. Aghaei, A. Khademzadeh, M. Reshadi, and K. Badie, "Link testing: A survey of current trends in network on chip," *J. Electron. Test.*, vol. 33, no. 2, pp. 209–225, Apr. 2017.

[49] C. Ababei and R. Katti, "Achieving network on chip fault tolerance by adaptive remapping," in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, May 2009, pp. 1–4.

[50] A. Weichslgartner, S. Wildermann, and J. Teich, "Dynamic decentralized mapping of tree-structured applications on NoC architectures," in *Proc. 5th ACM/IEEE Int. Symp.*, May 2011, pp. 201–208.

[51] C.-L. Chou and R. Marculescu, "FARM: Fault-aware resource management in NoC-based multiprocessor platforms," in *Proc. Design, Autom. Test Eur.*, Mar. 2011, pp. 1–6.

[52] R. B. Tonetto, H. M. G. de A. Rocha, B. Zatt, A. C. S. Beck, and G. L. Nazar, "A reliability-oriented machine learning strategy for heterogeneous multicore application mapping," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Oct. 2020, pp. 1–5.

[53] S. Paul, N. Chatterjee, and P. Ghosal, "A permanent fault tolerant dynamic task allocation approach for network-on-chip based multicore systems," *J. Syst. Archit.*, vol. 97, pp. 287–303, Aug. 2019.

[54] P. V. Bhanu and J. Soumya, "Fault-tolerant application mapping on mesh-of-tree based network-on-chip," *J. Syst. Archit.*, vol. 116, Jun. 2021, Art. no. 102026.

[55] P. V. Bhanu, R. Govindan, P. Kattamuri, J. Soumya, and L. R. Cenkeramaddi, "Flexible spare core placement in torus topology based NoCs and its validation on an FPGA," *IEEE Access*, vol. 9, pp. 45935–45954, 2021.

[56] N. K. R. Beechu, V. M. Harishchandra, and N. K. Y. Balachandra, "High-performance and energy-efficient fault-tolerance core mapping in NoC," *Sustain. Computing: Informat. Syst.*, vol. 16, pp. 1–10, Dec. 2017.

[57] N. Kadri and M. Koudil, "Multi-objective biogeography-based optimization and reinforcement learning hybridization for network-on chip reliability improvement," *J. Parallel Distrib. Comput.*, vol. 161, pp. 20–36, Mar. 2022.

[58] P. V. Bhanu, R. Govindan, R. Kumar, V. Singh, J. Soumya, and L. R. Cenkeramaddi, "Fault-tolerant application-specific topology-based NoC and its prototype on an FPGA," *IEEE Access*, vol. 9, pp. 76759–76779, 2021.

[59] J. Choudhary, C. S. Sudarsan, and J. Soumya, "A performance-centric ML-based multi-application mapping technique for regular network-on-chip," *Memories, Mater., Devices, Circuits Syst.*, vol. 4, Jul. 2023, Art. no. 100059.

[60] G. Kim, D. Kim, S. Park, Y. Kim, K. Lee, I. Hong, K. Bong, and H.-J. Yoo, "An augmented reality processor with a congestion-aware network-on-chip scheduler," *IEEE Micro*, vol. 34, no. 6, pp. 31–41, Nov. 2014.

[61] H.-L. Chao, Y.-R. Chen, S.-Y. Tung, P.-A. Hsiung, and S.-J. Chen, "Congestion-aware scheduling for NoC-based reconfigurable systems," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2012, pp. 1561–1566.

[62] F. Rad, M. Reshadi, and A. Khademzadeh, "A survey and taxonomy of congestion control mechanisms in wireless network on chip," *J. Syst. Archit.*, vol. 108, Sep. 2020, Art. no. 101807.

[63] S.-H. Lee, Y.-C. Yoon, and S.-Y. Hwang, "Communication-aware task assignment algorithm for MPSoC using shared memory," *J. Syst. Archit.*, vol. 56, no. 7, pp. 233–241, Jul. 2010.

[64] S. Kaushik, A. K. Singh, and T. Srikanthan, "Computation and communication aware run-time mapping for NoC-based MPSoC platforms," in *Proc. IEEE Int. SOC Conf.*, Sep. 2011, pp. 185–190.

[65] A. K. Singh, W. Jigang, A. Prakash, and T. Srikanthan, "Mapping algorithms for NoC-based heterogeneous MPSoC platforms," in *Proc. 12th Euromicro Conf. Digit. Syst. Design, Architectures, Methods Tools*, Aug. 2009, pp. 133–140.

[66] C. Wang, Y. Zhu, J. Jiang, X. Liu, and X. Han, "A dynamic contention-aware application allocation algorithm for many-core processor," in *Proc. IEEE IEEE 17th Int. Conf. High Perform. Comput. Commun. 7th Int. Symp. Cyberspace Saf. Secur., IEEE 12th Int. Conf. Embedded Softw. Syst.*, Aug. 2015, pp. 308–315.

[67] M. Fattah, M. Ramirez, M. Daneshtalab, P. Liljeberg, and J. Plosila, "CoNA: Dynamic application mapping for congestion reduction in many-core systems," in *Proc. IEEE 30th Int. Conf. Comput. Design (ICCD)*, Sep. 2012, pp. 364–370.

[68] H.-L. Chao, S.-Y. Tung, and P.-A. Hsiung, "Dynamic task mapping with congestion speculation for reconfigurable network-on-chip," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 10, no. 1, pp. 1–25, Mar. 2017.

[69] A. Rezaei, M. Daneshtalab, M. Palesi, and D. Zhao, "Efficient congestion-aware scheme for wireless on-chip networks," in *Proc. 24th Euromicro Int. Conf. Parallel, Distrib., Network-Based Process. (PDP)*, Feb. 2016, pp. 742–749.

[70] T. Maqsood, K. Bilal, and S. A. Madani, "Congestion-aware core mapping for network-on-chip based systems using betweenness centrality," *Future Gener. Comput. Syst.*, vol. 82, pp. 459–471, May 2018.

[71] S. Paul, N. Chatterjee, and P. Ghosal, "Dynamic task allocation and scheduling with contention-awareness for network-on-chip based multicore systems," *J. Syst. Archit.*, vol. 115, May 2021, Art. no. 102020.

[72] M. Fattah, P. Liljeberg, J. Plosila, and H. Tenhunen, "Adjustable contiguity of run-time task allocation in networked many-core systems," in *Proc. 19th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2014, pp. 349–354.

[73] C. Addo-Quaye, "Thermal-aware mapping and placement for 3-D NoC designs," in *Proc. IEEE Int. SOC Conf.*, Sep. 2005, pp. 25–28.

[74] S. Paul, N. Chatterjee, and P. Ghosal, "Dynamic task mapping and scheduling with temperature-awareness on network-on-chip based multicore systems," *J. Syst. Archit.*, vol. 98, pp. 271–288, Sep. 2019.

[75] B. Li, X. Wang, A. K. Singh, and T. Mak, "On runtime communication and thermal-aware application mapping and defragmentation in 3D NoC systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 12, pp. 2775–2789, Dec. 2019.

[76] M. F. Reza, D. Zhao, and M. Bayoumi, "Power-thermal aware balanced task-resource co-allocation in heterogeneous many CPU-GPU cores NoC in dark silicon era," in *Proc. 31st IEEE Int. Syst.-on-Chip Conf. (SOCC)*, Sep. 2018, pp. 260–265.

[77] O. Derin, E. Cannella, G. Tuveri, P. Meloni, T. Stefanov, L. Fiorin, L. Raffo, and M. Sami, "A system-level approach to adaptivity and fault-tolerance in NoC-based MPSoCs: The MADNESS project," *Microprocessors Microsyst.*, vol. 37, nos. 6–7, pp. 515–529, Aug. 2013.

[78] P. Meloni, G. Tuveri, L. Raffo, E. Cannella, T. Stefanov, O. Derin, L. Fiorin, and M. Sami, "System adaptivity and fault-tolerance in NoC-based MPSoCs: The MADNESS project approach," in *Proc. 15th Euromicro Conf. Digit. Syst. Design*, Sep. 2012, pp. 517–524.

[79] Y. Ge, Q. Qiu, and Q. Wu, "A multi-agent framework for thermal aware task migration in many-core systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 10, pp. 1758–1771, Oct. 2012.

[80] B. Yun, K. G. Shin, and S. Wang, "Predicting thermal behavior for temperature management in time-critical multicore systems," in *Proc. IEEE 19th Real-Time Embedded Technol. Appl. Symp. (RTAS)*, Apr. 2013, pp. 185–194.

[81] T. Chantem, R. P. Dick, and X. Sharon Hu, "Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs," in *Proc. Design, Autom. Test Eur.*, Mar. 2008, pp. 288–293.

[82] M. Becker, K. Sandström, M. Behnam, and T. Nolte, "Limiting temperature gradients on many-cores by adaptive reallocation of real-time workloads," in *Proc. IEEE Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2014, pp. 1–8.

[83] H. Sarhan, O. K. Eddash, M. Raymond, A. Wassal, and Y. Ismail, "Temperature-aware adaptive task-mapping targeting uniform thermal distribution in MPSoC platforms," in *Proc. Int. Conf. Energy Aware Comput.*, Dec. 2010, pp. 1–3.

[84] H.-H. Chu, Y.-C. Kao, and Y.-S. Chen, "Adaptive thermal-aware task scheduling for multi-core systems," *J. Syst. Softw.*, vol. 99, pp. 155–174, Jan. 2015.

[85] I. Alouani, T. Wild, A. Herkersdorf, and S. Niar, "Adaptive reliability for fault tolerant multicore systems," in *Proc. Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2017, pp. 538–542.

[86] B. Pourmohseni, S. Wildermann, M. Glaß, and J. Teich, "Hard real-time application mapping reconfiguration for NoC-based many-core systems," *Real-Time Syst.*, vol. 55, no. 2, pp. 433–469, Apr. 2019.

[87] A. Demiriz, H. Ahangari, and O. Ozturk, "Temperature-aware core mapping for heterogeneous 3D NoC design through constraint programming," in *Proc. 28th Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, Mar. 2020, pp. 312–318.

[88] J.-J. Han, M. Lin, D. Zhu, and L. T. Yang, "Contention-aware energy management scheme for NoC-based multicore real-time systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 3, pp. 691–701, Mar. 2015.

[89] A. Gammoudi, A. BenZina, M. Khalgui, and D. Chillet, "Energy-efficient scheduling of real-time tasks in reconfigurable homogeneous multicore platforms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 12, pp. 5092–5105, Dec. 2020.

[90] U. U. Tariq, H. Ali, L. Liu, J. Panneerselvam, and X. Zhai, "Energy-efficient static task scheduling on VFI-based NoC-HMPSoCs for intelligent edge devices in cyber-physical systems," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 6, pp. 1–22, Nov. 2019.

[91] J. Tu, T. Yang, L. Yin, S. Xie, R. Xu, and J. Sun, "Aging-aware task scheduling for mesh-based network-on-chips under aging effect," *J. Circuits, Syst. Comput.*, vol. 28, no. 9, Aug. 2019, Art. no. 1950146.

[92] E. L. D. S. Carvalho, N. L. V. Calazans, and F. G. Moraes, "Dynamic task mapping for MPSoCs," *IEEE Design Test Comput.*, vol. 27, no. 5, pp. 26–35, Sep. 2010.

[93] C.-L. Chou and R. Marculescu, "User-aware dynamic task allocation in networks-on-chip," in *Proc. Design, Autom. Test Eur.*, Mar. 2008, pp. 1232–1237.

[94] C.-L. Chou, U. Y. Ogras, and R. Marculescu, "Energy- and performance-aware incremental mapping for networks on chip with multiple voltage levels," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 10, pp. 1866–1879, Oct. 2008.

[95] M. A. A. Faruque, R. Krist, and J. Henkel, "ADAM: Run-time agent-based distributed application mapping for on-chip communication," in *Proc. 45th ACM/IEEE Design Autom. Conf.*, Jun. 2008, pp. 760–765.

[96] A. Mehran, A. Khademzadeh, and S. Saeidi, "DSM: A heuristic dynamic spiral mapping algorithm for network on chip," *IEICE Electron. Exp.*, vol. 5, no. 13, pp. 464–471, 2008.

[97] V. Tsoutsouras, I. Anagnostopoulos, D. Masouros, and D. Soudris, "A hierarchical distributed runtime resource management scheme for NoC-based many-cores," *ACM Trans. Embedded Comput. Syst.*, vol. 17, no. 3, pp. 1–26, May 2018.

[98] M.-H. Haghbayan, A. Kanduri, A.-M. Rahmani, P. Liljeberg, A. Jantsch, and H. Tenhunen, "MapPro: Proactive runtime mapping for dynamic workloads by quantifying ripple effect of applications on networks-on-chip," in *Proc. 9th Int. Symp. Netw.-on-Chip*, Sep. 2015, pp. 1–8.

[99] S. Kobbe, L. Bauer, D. Lohmann, W. Schröder-Preikschat, and J. Henkel, "DistRM: Distributed resource management for on-chip many-core systems," in *Proc. 9th IEEE/ACM/IFIP Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES ISSS)*, Oct. 2011, pp. 119–128.

[100] M. Fattah, A.-M. Rahmani, T. C. Xu, A. Kanduri, P. Liljeberg, J. Plosila, and H. Tenhunen, "Mixed-criticality run-time task mapping for NoC-based many-core systems," in *Proc. 22nd Euromicro Int. Conf. Parallel, Distrib., Netw.-Based Process.*, Feb. 2014, pp. 458–465.

[101] Y. Yao, Y. Song, H. Ge, Y. Huang, and D. Zhang, "A communication-aware and predictive list scheduling algorithm for network-on-chip based heterogeneous muti-processor system-on-chip," *Microelectron. J.*, vol. 121, Mar. 2022, Art. no. 105367.

[102] A. E. Dalzotto, C. D. S. Borges, M. Ruaro, and F. G. Moraes, "Leveraging NoC-based many-core performance through runtime mapping defragmentation," in *Proc. 29th IEEE Int. Conf. Electron., Circuits Syst. (ICECS)*, Oct. 2022, pp. 1–4.

[103] W. Amin, F. Hussain, S. Anjum, S. Saleem, W. Ahmad, and M. Hussain, "HyDra: Hybrid task mapping application framework for NoC-based MPSoCs," *IEEE Access*, vol. 11, pp. 52309–52326, 2023.

[104] M. F. Reza, "Machine learning enabled solutions for design and optimization challenges in networks-on-chip based multi/many-core architectures," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 19, no. 3, pp. 1–26, Jul. 2023.

[105] M. F. Reza and Z. McCloud, "Heuristics-enabled high-performance application mapping in network-on-chip based multicore systems," in *Proc. IEEE Int. Conf. Omni-Layer Intell. Syst. (COINS)*, Jul. 2023, pp. 1–6.

[106] M. F. Reza, D. Zhao, and M. Bayoumi, "Energy-efficient task-resource co-allocation and heterogeneous multi-core NoC design in dark silicon era," *Microprocessors Microsyst.*, vol. 86, Oct. 2021, Art. no. 104055.

[107] M. F. Reza, D. Zhao, H. Wu, and M. Bayoumi, "Hotspot-aware task-resource co-allocation for heterogeneous many-core networks-on-chip," *Comput. Electr. Eng.*, vol. 68, pp. 581–602, May 2018.

[108] A. T. Tran and B. Baas, "NoCTweak: A highly parameterizable simulator for early exploration of performance and energy of networks on-chip," VLSI Comput. Lab, Dept. ECE, Univ. California, Davis, Davis, CA, USA, Tech. Rep. ECE-VCL-2012-2, 2012.

[109] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A power-performance simulator for interconnection networks," in *Proc. 35th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Nov. 2002, pp. 294–305.

[110] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration," in *Proc. Design, Autom. Test Eur. Conf. Exhib.*, Apr. 2009, pp. 423–428.

[111] J. E. Miller, H. Kasture, G. Kurian, C. Gruenwald, N. Beckmann, C. Celio, J. Eastep, and A. Agarwal, "Graphite: A distributed parallel simulator for multicores," in *Proc. 16th Int. Symp. High-Perform. Comput. Archit. (HPCA)*, Jan. 2010, pp. 1–12.

[112] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and H. Kim, "Pin: Building customized program analysis tools with dynamic instrumentation," *ACM SIGPLAN Notices*, vol. 40, no. 6, pp. 190–200, 2005.

[113] W. Amin, F. Hussain, and S. Anjum, "IHPSA: An improved bio-inspired hybrid optimization algorithm for task mapping in network on chip," *Microprocessors Microsyst.*, vol. 90, Apr. 2022, Art. no. 104493.

[114] A. Gamatié, X. An, Y. Zhang, A. Kang, and G. Sassatelli, "Empirical model-based performance prediction for application mapping on multi-core architectures," *J. Syst. Archit.*, vol. 98, pp. 1–16, Sep. 2019.

[115] A. Gamatié, R. Ursu, M. Selva, and G. Sassatelli, "Performance prediction of application mapping in manycore systems with artificial neural networks," in *Proc. IEEE 10th Int. Symp. Embedded Multicore/Many-Core Syst.-on-Chip (MCSOC)*, Sep. 2016, pp. 185–192.

[116] M. F. Reza and P. Ampadu, "Energy-efficient and high-performance NoC architecture and mapping solution for deep neural networks," in *Proc. 13th IEEE/ACM Int. Symp. Netw.-on-Chip*, Oct. 2019, pp. 1–8.

[117] A. Bytyn, R. Ahlsdorf, R. Leupers, and G. Ascheid, "Dataflow aware mapping of convolutional neural networks onto many-core platforms with network-on-chip interconnect," 2020, *arXiv:2006.12274*.

[118] C. Ma, Q. Zhao, G. Li, L. Deng, and G. Wang, "A deadlock-free physical mapping method on the many-core neural network chip," *Neurocomputing*, vol. 401, pp. 327–337, Aug. 2020.

[119] S. Jagadheesh, P. V. Bhanu, and J. Soumya, "NoC application mapping optimization using reinforcement learning," *ACM Trans. Design Autom. Electron. Syst.*, vol. 27, no. 6, pp. 1–16, Nov. 2022.

[120] K. Manna and J. Mathew, *Design and Test Strategies for 2D/3D Integration for NoC-Based Multicore Architectures*. Springer, 2019.

[121] A. B. Achballah, S. B. Othman, and S. B. Saoud, "Problems and challenges of emerging technology networks-on-chip: A review," *Microprocessors Microsyst.*, vol. 53, pp. 1–20, Aug. 2017.

[122] A. Rezaei, M. Daneshtalab, D. Zhao, F. Safaei, X. Wang, and M. Ebrahimi, "Dynamic application mapping algorithm for wireless network-on-chip," in *Proc. 23rd Euromicro Int. Conf. Parallel, Distrib., Netw.-Based Process.*, Mar. 2015, pp. 421–424.

[123] A. Rezaei, M. Daneshtalab, and D. Zhao, "CAP-W: Congestion-aware platform for wireless-based network-on-chip in many-core era," *Microprocessors Microsyst.*, vol. 52, pp. 23–33, Jul. 2017.

[124] A. Dehghani, "A design flow for an optimized congestion-aware application-specific wireless network-on-chip architecture," *Future Gener. Comput. Syst.*, vol. 106, pp. 234–249, May 2020.

[125] I. Assayad and A. Girault, "Adaptive mapping for multiple applications on parallel architectures," in *Proc. Int. Symp. Ubiquitous Netw.* Springer, 2017, pp. 584–595.

SHAROON SALEEM received the B.Sc. degree in computer engineering and the M.S. degree from the University of Engineering and Technology, Taxila (UET Taxila), Pakistan, in 2007 and 2015, respectively, where he is currently pursuing the Ph.D. degree. He is a Lecturer with the Computer Engineering Department, UET Taxila. He had been involved in various projects in the domain of embedded systems development in his career. His research interests include network-on-chip (NoC), fault-tolerant systems, and reconfigurable digital system designs. He is also working on the low-complexity and low-cost application mapping techniques in network-on-chip.

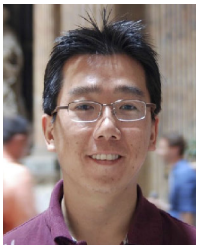
FAWAD HUSSAIN (Senior Member, IEEE) received the B.Sc. degree in computer engineering, the M.Sc. degree in electrical engineering, and the Ph.D. degree in computer engineering from the University of Engineering and Technology, Taxila (UET Taxila), Pakistan, in 2005, 2009, and 2015, respectively. He is currently an Assistant Professor with the Computer Engineering Department, UET Taxila. His research interests include speech and audio processing, computer vision, natural language processing, and network-on-chip (NoC). He is currently leading research for the M.S. and Ph.D. students in the mentioned areas of interest. He is a Reviewer of a few of the well-known peer-reviewed journals, such as *Advanced Engineering Informatics*, IEEE Access, *Journal of Ambient Intelligence and Humanized Computing*, and *Artificial Intelligence Review*.

WAQAR AMIN received the B.Sc. degree in computer engineering and the M.S. degree from the University of Engineering and Technology, Taxila (UET Taxila), Pakistan, in 2007 and 2014, respectively, where he is currently pursuing the Ph.D. degree. He has vast experience in the research and development of various embedded systems companies. He was involved in the development of many GSM and 3G systems. He is currently working on the low-cost application mapping on NoC. His research interests include fault-tolerant systems, network-on-chip (NoC), self-healing, and reconfigurable systems designs.

REHAN AHMED received the B.Sc. degree in computer engineering from the University of Engineering and Technology, Taxila, Pakistan, in 2004, the M.Sc. degree in electrical engineering with a focus on embedded systems from the Technical University of Munich, Munich, Germany, in 2009, and the Ph.D. degree in electrical engineering with a focus on FPGA architectures and CAD from The University of British Columbia, Vancouver, BC, Canada, in 2015. From 2015 to 2017, he was a Senior Hardware Engineer at IXIA, Vancouver, (now a Keysight business unit), where he worked on CloudStorm 100GE Application and Security Test Load Module. Since 2017, he has been an Assistant Professor with the School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, Pakistan. His industrial experience includes work at SIEMENS, Munich, Microsoft Research, Redmond, USA, and IXIA. He is currently working on developing RISC-V-based processor cores and efficient hardware for AI on edge. He has authored two books chapters and 15 articles and holds one U.S. patent.

YOUSAF BIN ZIKRIA (Senior Member, IEEE) received the Ph.D. degree from the Department of Information and Communication Engineering, Yeungnam University, South Korea, in 2016. From 2018 to 2022, he was an Assistant Professor with the Department of Information and Communication Engineering, College of Engineering, Yeungnam University, Gyeongsan-Si, South Korea. He is currently a Senior Lecturer with the Victorian Institute of Technology, Sydney, Australia. He has more than ten years of experience in research, academia, and industry in the fields of information and communication engineering and computer science. He has authored more than 100 scientific peer-reviewed journals, magazines, conference papers, patents, and book chapters. His journal article's cumulative impact factor (IF) is more than 450. He published papers at top venues, such as IEEE COMMUNICATIONS, SURVEYS, AND TUTORIALS, *IEEE Wireless Communications Magazine*, *IEEE Network*, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, *Future Generation Computer Systems* (Elsevier), *Sustainable Cities and Society* (Elsevier), *Journal of Network and Computer Applications* (Elsevier), and *Cities* (Elsevier). He has been listed in the world's top 2% of researchers, from 2021 to 2023, published by Elsevier and Stanford University.

FARRUH ISHMANOV received the B.S. degree in information systems from the Tashkent State University of Economics, Uzbekistan, in 2007, and the M.S. and Ph.D. degrees from the Department of Information and Communication Engineering, Yeungnam University, South Korea, in 2009 and 2014, respectively. At the Tashkent State University of Economics, he studied and worked with the Multimedia Laboratory during the undergraduate years. In March 2015, he joined the Department of Electronics and Communication Engineering, Kwangwoon University, South Korea, where he is currently an Assistant Professor. His research interests include resource management and security in wireless sensor networks and the IoT. He was awarded the Korean Government IITA Scholarship for the M.S. degree.



HEEJUNG YU (Senior Member, IEEE) received the B.S. degree in radio science and engineering from Korea University, Seoul, South Korea, in 1999, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2001 and 2011, respectively. From 2001 to 2012, he was with the Electronics and Telecommunications Research Institute, Daejeon, and from 2012 to 2019, he was with Yeungman University, Gyeongsan, South Korea. He is currently a Professor with the Department of Electronics and Information Engineering, Korea University, Sejong, South Korea. His research interests include statistical signal processing and communication theory.