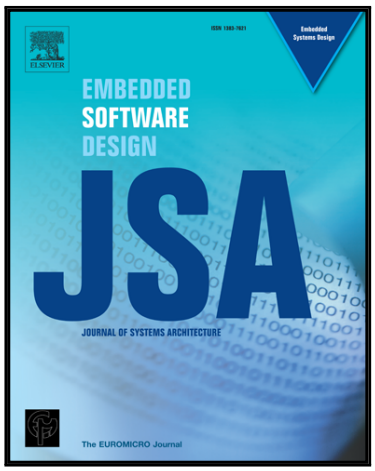


Energy and Performance-Aware Application Mapping for Inhomogeneous 3D Networks-on-Chip

Michael Opoku Agyeman, Ali Ahmadinia, Nader Bagherzadeh

PII: S1383-7621(17)30156-X
DOI: https://doi.org/10.1016/j.sysarc.2018.08.002
Reference: SYSARC 1516

To appear in: *Journal of Systems Architecture*

Received date: 22 March 2017
Revised date: 2 August 2018
Accepted date: 5 August 2018

Please cite this article as: Michael Opoku Agyeman, Ali Ahmadinia, Nader Bagherzadeh, Energy and Performance-Aware Application Mapping for Inhomogeneous 3D Networks-on-Chip, *Journal of Systems Architecture* (2018), doi: https://doi.org/10.1016/j.sysarc.2018.08.002

Energy and Performance-Aware Application Mapping for Inhomogeneous 3D Networks-on-Chip

Michael Opoku Agyeman, Ali Ahmadinia, Nader Bagherzadeh

***Abstract*—Three dimensional Networks-on-Chip (3D NoCs) have evolved as an ideal solution to the communication demands and complexity of future high density many core architectures. However, the design practicality of 3D NoCs faces several challenges such as thermal issues, high power consumption and area overhead of 3D routers as well as high complexity and cost of vertical link implementation. To mitigate the performance and manufacturing cost of 3D NoCs, inhomogeneous architectures have emerged to combine 2D and 3D routers in 3D NoCs producing lower area and energy consumption while maintaining the performance of homogeneous 3D NoCs. Due to the limited number of vertical links, application mapping on inhomogeneous 3D NoCs can be complex. However, application mapping has a great impact on the performance and energy consumption of NoCs. This paper presents an energy and performance aware application mapping algorithm for inhomogeneous 3D NoCs. The algorithm has been evaluated with various realistic traffic patterns and compared with existing mapping algorithms. Experimental results show NoCs mapped with the proposed algorithm have lower energy consumption and significant reduction in packet delays compared to the existing algorithms and comparable average packet latency with Branch-and-Bound.**

***Index Terms*—Multi-core Architectures; Network-on-Chip; 3D Integration**

I. Introduction

Current advances in semiconductor integration and processing capability of System-on-Chip (SoC) enable a large number of high performance cores to be integrated into a single chip. Networks-on-Chip (NoC) have been proposed to adopt the idea of networking in the macro-world for current and future SoCs but with limited resources and tighter constraints. 3D IC fabrication has also emerged to stack several layers of 2D ICs vertically, providing shorter vertical links with enhanced connectivity and lower delays. Combining 3D IC fabrication and Network-on-Chip technique, 3D NoC creates new design and research opportunities and challenges for high density SoCs. High integration and performance opportunity of 3D NoCs encourage enhanced heterogeneity of application design with implementation of multiple applications on a SoC.

Conventionally, homogeneous 3D NoCs have been employed for 3D-Integration where 3D routers are employed for inter-layer and planar communication. However, the 3D routers have a larger area and power consumptions than a 2D router with a similar architecture. Consequently, the homogeneous 3D router distribution may lead to significant area and power overheads if applied to applications whose communication patterns vary significantly among embedded cores. Moreover, Through Silicon Via (TSV) which has been accepted as a viable inter-layer wiring technique has a complex and expensive manufacturing process [1], [2]. To optimize the performance and manufacturing cost of 3D NoCs with minimal distortion to the modularity, inhomogeneous architectures have been proposed to combine 2D and 3D routers in 3D NoCs [3], [4], [5], [2], [6]. Several inhomogeneous 3D architectures focusing on different NoC router architectures, minimal hop-count between 2D and 3D routers in each layer, and uniform distribution of 2D and 3D routers have been proposed [7], [2], [8]. However, due to the limited number of 3D routers and vertical links, mapping of applications to an inhomogeneous 3D NoC can be challenging. Specifically, different applications have variable characteristics at design time and run-time with possible link, node or task failures due to current technology limitations.

In this paper, an energy-aware application mapping technique for inhomogeneous 3D NoCs is proposed. The proposed algorithm efficiently maps a given application with minimized communication energy while maintaining the performance. The application must be known and its communication task graph much be constructed for this approach, similar to the applied benchmarks [34][56-58] as has been used in literature [3-9][13-18]. Consequently, the mapping algorithm involves three main stages:

1) Initial NoC size determination and clustering which assign regions along the 3D NoC vertices to reduce the cost of 3D routers.
2) Architecture matching stage which allocates the application graph to the clusters in the 3D NoC such that optimized numbers of 2D and 3D routers are assigned by enumerating the communication bandwidth and energy constraints in each cluster. Thus, this stage re-evaluates and assigns a suitable NoC size and automatically determines the suitable number of 3D and 2D routers and their assigned tiles.
3) Task to NoC region mapping which assigns the IP cores to the tiles in different clusters with minimized total energy consumption and maximized performance. This stage efficiently exploits area and power efficiencies of 2D routers as well as the higher bandwidth and lower latency characteristics of the vertical links associated with 3D routers.

Corresponding Author: aahmadinia@csusm.edu

M. O. Agyeman is with Department of Computing and Immersive Technologies, University of Northampton, UK

A. Ahmadinia is with Department of Computer Science, California State University San Marcos, San Marcos, CA 92078, USA

N. Bagherzadeh is with Dept. of Electrical Engineering and Computer Science, University of California, Irvine, Irvine, CA 92697, USA

The remainder of the paper is organized as follows: Section II evaluates recent contributions on 3D NoC mapping techniques. Section III formulates the problem of generation efficient 3D NoC architectures. Respectively, Sections IV and V formulate the mapping problem and describe the proposed mapping approach. Experimental results presented in Section VI emphasizes on the performance benefits of employing the proposed mapping algorithm to generate inhomogeneous architectures compared to other techniques. Finally, Section VII presents the concluding remarks of the main findings.

II. Related Work

The evolution of SoC design to the third dimension offers a lot of opportunities such as integration of inhomogeneous cores which results in several challenges including optimal inhomogeneous NoC topologies, router architectures and application mapping techniques [9], [10], [11], [12], [13]. Various 3D NoC topologies are presented and evaluated in [14], [15], [16], [17] where homogeneous 3D routers are employed in each architecture. A 3D router has a larger area and power consumption than a 2D router with similar architectures [18], [13]. Particularly, the 7 port symmetric router has an area and power overhead of 36% and 158%, respectively compared to a conventional 5 port router [8]. Li et al. [19], [13] proposed to replace the large 7 port symmetric 3D routers with 6 port NoC-Bus hybrid 3D routers. However, the hybrid router requires an additional central arbiter per each vertical pillar in the NoC. Moreover, the hybrid router still has a large crossbar and energy consumption. Xiangyu et al [20] have demonstrated that the area overhead of TSV increases with increase in number of 3D layers. Particularly, the area overhead of the TSV for a 4 layer 3D NoC with 5 million gates can reach as high as 10% [6]. Similarly, Bartzas et al. [3] presented a study of the area, power and performance trade-off of combining 2D and 3D routers in 3D mesh and torus topologies. Xu et al. [7] performed an evaluation of the impact of reducing the number of TSVs to half and quarter on the performance of 3D NoCs [21]. Their proposed architectures, quarter/lo and half/lo (quarter/hi and half/hi), aim at generating inhomogeneous 3D NoC with 2D routers placed as close to (far from) 3D routers as possible in each layer [21]. These architectures suffer from uneven distribution of 3D routers and unpredictable delays for different applications. Liu et al. [22] used partition islands of routers to constitute regions for sharing the same TSV pad for inter-layer communication controlled by serialization logic [21]. However, serialization along the TSV bundle causes the average packet delay to increase exponentially as the number of routers per TSV bundle increases [21]. Moreover, the TSV pads have no direct connection to the processing cores, which is a waste of chip area compared to our proposed architectures [21]. Even when bypass links are introduced to enable adaptivity among the TSV sharing regions, these architectures suffer from high packet latencies compared to a homogeneous 3D NoC implemented with a deterministic routing [23], [24]. Also genetic algorithm and simulation annealing employed in [23], [6] for the selection and placement of different TSV patterns (sharing regions) in 3D NoCs have an exponential complexity with a large design exploration space [6]. Similarly, Pasricha [25] proposed a serialization technique for reducing the number of TSVs where link size of TSVs at selected nodes is reduced by a fraction. Thus if the number of TSVs exceeds a threshold, serialization is adopted to reduce the bandwidth of some TSVs. However, due to the reduced bandwidth of the TSVs and serialization logic, such architectures have high average packet latencies. Moreover, due to the higher overhead of serialization receiver and transmitter logic compared to the TSV reduction, such architectures have even higher power consumption compared to homogeneous 3D mesh. Based on the serialization methodology, Pasricha [26] proposed a 3D NoC synthesis framework by augmenting router and placement techniques proposed in [27], these routers have several local ports which have high power consumption due to the increased number of ports and high data rates across the crossbar. On the other hand, existing inhomogeneous architectures [28], [7], [22], [29], [3], [4], [24], [30], [31], [32], [5] assume a fully mapped NoC and do not consider the dynamics of application traffic load in their inhomogeneous architectures [13]. Applications in such 3D NoCs are not optimized as communication bandwidth and performance constraints of the applications were not considered in the architecture generation [13]. To resolve this, a systematic approach for generating inhomogeneous 3D NoC architectures where the TSV and buffer utilization of the given application are exploited is proposed in [6], [13]. However, the systematic approach assumes the adopted mapping algorithm (which could have a high complexity) is optimized. This paper significantly complements the systematic approach by presenting an energy and performance efficient NoC dimension generation technique. In addition, as an effective mapping algorithm is proposed, which generates inhomogeneous 3D NoC architectures. The generated architectures could be ported into the systematic approach for further buffer resizing. The impact of variable buffer sizes is detailed in [6].

Several IP mapping algorithms have been proposed for 2D NoCs that focus on minimizing the overall communication power [33], [34], [35], [36], [37]. However, most of these existing mapping algorithms for 2D NoCs are not suitable for 3D NoCs as communication in the third dimension introduces a new set of NoC constraints, which are not considered in the 2D designs. For example, different area and energy overheads of 3D routers and different transfer delay per dimension are some of these constraints. Moreover, such mapping algorithms will not be suitable for generation of 3D NoCs with inhomogeneous router distributions as they do not take advantage of the power and performance benefits of the router heterogeneity. Branch-and-Bound' algorithm proposed by Hu et al. [34] can be extended to map application to inhomogeneous 3D NoCs [21]. However, this algorithm employs partial exhaustive search trees and has an extremely long run-time. Janidarmian et al. [38] proposed Onyx, a heuristic bandwidth-constrained mapping algorithm for tile-based NoCs. Here, nodes with higher communication data rates are mapped first. Tosun [39] presented CastNet, a heuristic algorithm for mapping tasks onto mesh-based NoCs. Based on the symmetry of the mesh, tiles are grouped into partitions, and then nodes with high

communication volumes are mapped first to the partitions. Murali et al. [40] proposed Nmap, a three-phase algorithm to find near optimal mapping solution which has been extended to 3D NoCs [41].

Several mapping solutions have been proposed for 3D NoCs based on genetic algorithms [42], [43], [44], [45]. However, these algorithms have very high computational complexities. Wang et al. [46] proposed a mapping algorithm for 3D NoCs based on run-time incremental mapping technique [47]. Here, the algorithm tries to map applications to convex regions while utilizing as many vertical links as possible in the mapping process. This approach increases the number of 3D routers (vertical links) which are costly and have high power consumption.

As an effort to reduce the number of TSVs, [41] employ heuristics to map an application to a homogeneous 3D NoC. A number of vertical links with high traffic load are then preserved and the mapping algorithm is reapplied to improve the performance of the application in the generated inhomogeneous 3D NoC. A similar approach is adopted in [48], where Kernighan-Lin partitioning is employed as the mapping algorithm. In both cases, the mapping technique has to be repeated in an iterative manner. Moreover, besides the limitations associated with the Kernighan-Lin algorithm, the resulting mapping solution may not be optimal.

We propose a performance and energy-aware technique for mapping applications to 3D NoCs. The proposed technique automatically generates low latency inhomogeneous 3D NoCs for Embedded platforms by evaluating the number and placement of 2D and 3D routers required as well as the communication characteristics of tasks assigned to the processing elements. Run-time migration and scheduling techniques could be employed to manage end-to-end communication delays of the performance efficient inhomogeneous 3D NoCs generated by our mapping technique [36], [37]. Although mapping multiple tasks per core could improve the performance, we still need to deal with the same problem of mapping the clustered tasks to cores in a 3D-NoC.

It should be also noted that the generated inhomogeneous 3D NoCs rely heavily on the alignment of 3D routers to provide interlayer TSV connections. However, by introducing inhomogeneity in the network, we are able to save more power hungry 3D routers and help reduce the number of expensive TSV. In the following sections, the assumed model is for application, topology, and cost functions are mainly based on the literature [34][56][57][58] for a fair comparison to evaluate the impact of the proposed solution.

III. Problem Formulation

To efficiently exploit the non-uniform bandwidth requirements of various tasks in applications, an efficient algorithm that maps tasks to NoCs can be employed to generate inhomogeneous architectures with improved performance. Consequently, the focus of this paper is to capitalize on the architecture and topology variation of inhomogeneous 3D NoCs through an efficient application mapping technique. An example of an inhomogeneous 3D NoC architecture is shown in Fig. 1.

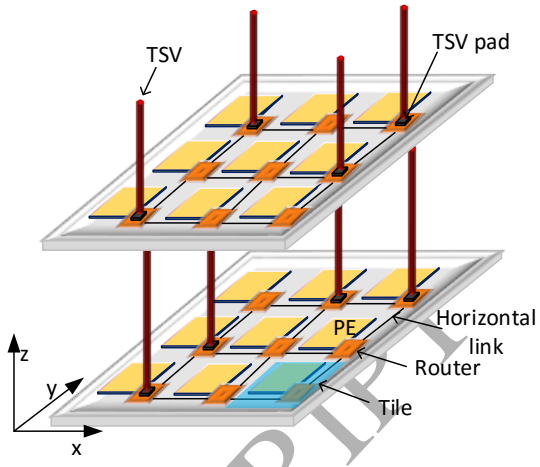


Fig. 1. Inhomogeneous 3D NoC

A. NoC Architecture and Application Formulation

An application $A_x = \{tsk_1, tsk_2, tsk_3 ... tsk_n\}$, where tsk_i represents an unmapped task in A_x, is modeled as a communication graph defined as follows

Definition 1: An application *Communication Graph* $(CG) = (T, W)$ is a weighted directed graph where each node denoted by $tsk_i \in T$ refers to a task and each $w_{ij} \in W$, is the communication between tasks tsk_i and $tsk_j \in T$. The weight of the edge $w_{ij} \in W$ characterizes the communication volume (*bit width*) between tsk_i and tsk_j.

Definition 2: An $X \times Y \times Z$ 3D NoC is said to have a symmetric NoC size if it satisfies:

$$X \approx Y \approx Z \tag{1}$$

Definition 3: A tile cluster is defined as a group of adjacent tiles allocated to a collection of communication dependent tasks. A tile cluster can combine tiles from any dimension.

Definition 4: A tile in a tile cluster is called a vertex if it is located at the boundaries of the tile cluster.

Please note that a tile may be or may not be a vertex as a tile cluster can encompass multiple tiles.

B. NoC Energy Formulation

The energy consumed by a task between a given tile pair t_i and t_j in application A_x is given by

$$E^{ij}_{tsk} = \sum_{\forall_{mn} \in A_x} b_{ij} \times E^{ij}_{bit} \tag{2}$$

where b_{ij} is the communication volume in number of bits transferred between t_i and t_j, mn represents all communication involving the task and E^{ij}_{bit} is the communication energy for sending a single bit of data between the tile pair which is given by (an extended version of 2D energy model [49]):

$$E^{ij}_{bit} = \eta_{3D} \times E_{R_{3D_{bit}}} + \eta_{2D} \times E_{R_{2D_{bit}}} + E^{ij}_{link_{bit}} \tag{3}$$

where η_{3D} and η_{2D} are the number of 3D and 2D routers traversed between the source and destination nodes, respectively.

$E_{R_{3D_{bit}}}$ and $E_{R_{2D_{bit}}}$ are energy consumption for sending a single bit across a 3D and 2D router, respectively. $E^{ij}_{link_{bit}}$ is the link energy consumption defined as:

$$E^{ij}_{link_{bit}} = MDh_{ij} \times E_{Hlink_{bit}} + MDv_{ij} \times E_{Vlink_{bit}} \quad (4)$$

where $E_{Hlink_{bit}}$ is the energy consumption per bit along the planar links and $E_{Vlink_{bit}}$ is the inter-layer link energy consumption per bit. Following the SMIC 90nm technology used in [50], $E_{Hlink_{bit}}$ and $E_{Vlink_{bit}}$ can be calculated as 0.127pJ and 9.56×10^{-3}, respectively. MDh_{ij} and MDv_{ij}, the Manhattan distance of the planar and inter-planar regions between t_i and t_j are given by Equations 5 and 6, respectively.

$$MDh_{ij} = |(t_{i_x} - t_{j_x})| + |(t_{i_y} - t_{j_y})| \quad (5)$$

$$MDv_{ij} = |(t_{i_z} - t_{j_z})| \quad (6)$$

$$MDij = MDv_{ij} + MDh_{ij} \quad (7)$$

C. Mapping Problem Definition

We formulate the mapping problem as follows: Given the communication graph CG of an application A_x find a mapping function $M : A_x \rightarrow NoC$ which maps all the tasks in CG to available tiles in the inhomogeneous architecture with minimized average packet delay such that

$$Min \left\{ EL_{cost} = \sum_{\forall_{ij} \in A_x} \left(E^{ij}_{tsk} \times L^{ij}_{tsk} \right) \right\} \quad (8)$$

where L^{ij}_{tsk} and E^{ij}_{tsk} are respectively the latency and energy consumption of transferring packets of a task between tiles t_i and t_j for a given deadlock free routing algorithm. Similar to [4], [24], [5], [3], [7], EL_{cost} represents the total combined cost of energy and latency of the given mapping, as an equal weight is given to both latency and energy consumption of the NoC.

To illustrate the composition of the problem, Fig.2 shows an example of an application that need to be mapped. The communication task graphs of the applications are given in Step 1. Step 2 estimates dimensions of a 3D NoC (in this case $3 \times 3 \times 3$) based on the total number of tasks available in the application presented to the mapping tool. To reduce the average packet latency and energy consumption, tasks with high communication dependencies must be mapped as close to each other as possible. To initiate the mapping process, tasks in order of their communication volumes are assigned to the NoC regions such that high energy consumer tasks are assigned to as many 2D routers as possible while maintaining a certain number of 3D routers [1] to enhance performance. Tasks that have communication dependencies on allocated tasks are then assigned to the task clusters such that the total hop-count between the tasks is reduced while an equivalent number of tasks is kept within each task cluster. It should be noted tasks assigned to different task clusters can still communicate with each other after the mapping process. Next, all tasks in the application are mapped as shown in Step 4. In this paper, we use single task-to-core mapping in our evaluation. However, similar to [36], [51], [37] the proposed algorithm is applicable to cases where several tasks can be grouped together and mapped to a single tile. The task grouping can be achieved by various cost functions such as the task dependencies and their energy communications [51]. In such cases, we consider these group of tasks as a single vertex in the communication task graph and the total outgoing/incoming bandwidth is taken as input to the mapping function in stage 1. Based on the above discussion, the mapping problem for inhomogeneous 3D NoCs is divided into sub-problems as detailed in the next section.

[1]Minimizing the number of 3D routers may reduce energy and area overheads. On the other hand, adding 3D routers can reduce the number of hop counts and delay. Therefore, the number of 3D routers is determined experimentally. An example of how tasks in the application are allocated is shown in Step 3. First, tasks with highest communication volumes are assigned to different task clusters.

IV. NoC Size and Mapping Constraints

To achieve our aim of optimizing inhomogeneous 3D NoCs, the mapping process is further decomposed into three main sub-problems (NoC size determination, architecture matching and task mapping) which are individually discussed in this section.

A. Initial NoC Size Determination and Partitioning

The choice of NoC size dimensions plays a vital role on the performance and energy consumption of the network. Several issues such as floor planning, temperature and area which affect NoC dimensions have been addressed over the past few years [52], [8], [53], [17], [54].

However, choice of 3D NoC dimensions has not received much attention in terms of delay and energy consumption. For a better understanding of the effect of NoC dimension on packet delay and energy consumption, we simulated 3D NoCs with various NoC dimensions under uniform and hotspot traffic patterns. For the router configuration, an input and output buffer size of 3 and 2 flits are used, respectively. As mentioned before, the impact of buffer size is already discussed in [6]. Figs. 3 and 4 show the variation of average packet latency and total energy consumption of 64 node 3D NoCs with various traffic rates (uniform and hotspot) under XYZ routing. As can be seen, decreasing the regularity of the 3D NoC dimensions significantly increases the packet delays. For instance, Fig. 3(a) shows that regular $4 \times 4 \times 4$ 3D NoC can sustain over 45% more traffic compared to lower average packet latency of $2 \times 16 \times 2$ 3D NoC. It can also be observed in Fig. 4(a) that, the $4 \times 4 \times 4$ 3D NoC has lower energy consumption at low traffic rates compared to other 3D NoCs. Even under high traffic rates, energy consumption of $4 \times 4 \times 4$ is lower than $4 \times 8 \times 2$. Similarly in Fig. 3(b), $4 \times 4 \times 4$ 3D NoCs have lower average packet latency compared to other NoC dimensions under hotspot traffic patterns. Moreover, Fig. 4(b) shows that $4 \times 4\times$ 4 3D NoCs have the lowest energy consumption. In summary, it can be inferred that the symmetry of the NoC dimensions has a significant impact on the average packet latency and total energy consumption. Hence, to optimize the performance of inhomogeneous architectures, an effective approach is required to reduce the irregularity of NoC dimensions at an early phase

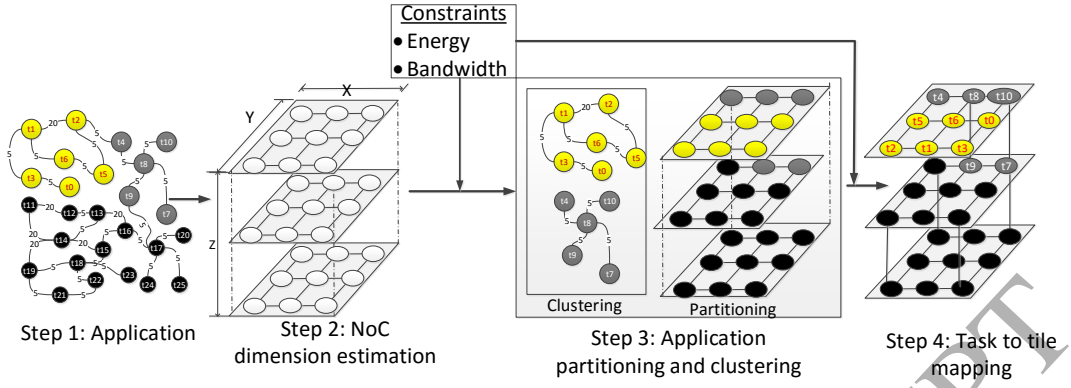
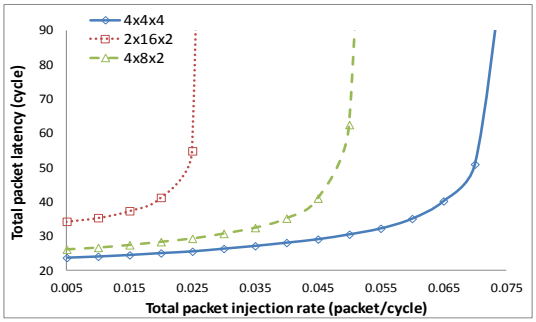
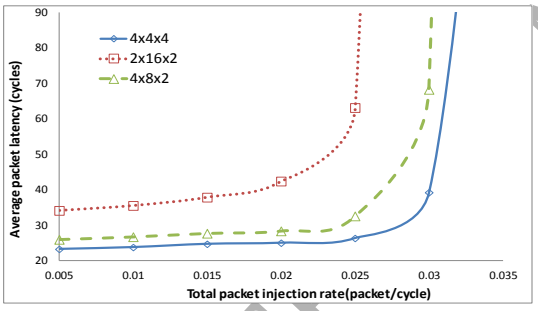


Fig. 2. Example of application to NoC mapping problem

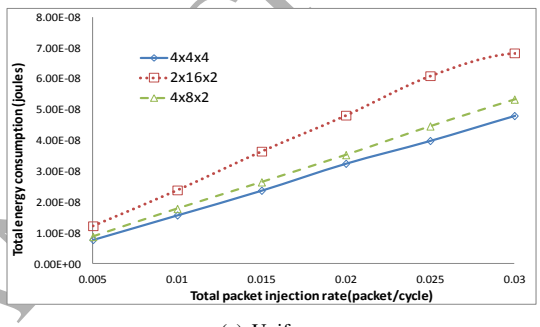


(a) Uniform

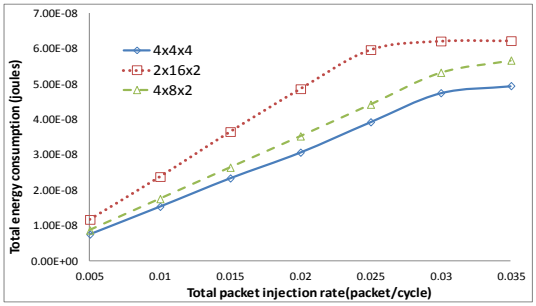


(b) Hotspot

Fig. 3. Comparison of average packet latency of various NoC sizes.



(a) Uniform



(b) Hotspot

Fig. 4. Total energy consumption of various NoC sizes.

of design exploration. In general, as shown in Table I, 3D NoCs with symmetrical dimensions have lower average hop-count. One of the important parameters to consider when

NoC Dimensions (X× Y× Z)	Average hop-count (routers) (s)
4 × 4 × 4	3.81
2 × 16 × 2	6.41
4 × 8 × 2	4.44

TABLE I
AVERAGE HOP-COUNT OF VARIOUS NOC DIMENSIONS

deciding the 3D NoC dimensions is thermal issues which might affect the communication performance of 3D NoCs. This is due to the fact that the resistance of metals is affected by temperature. Hence, we investigate the effect of high temperature on the propagation delay of signals on the inter-layer vertical links. By adopting the RC ladder model from [8], we investigated the effect of high temperature on the propagation delay of signals on the inter-layer vertical links in HSpice using Predictive Technology Model (PTM) [55]. Fig. 5 shows the propagation delay of a linear temperature variation simulated in HSpice. It can be seen that even with 8 layers and a temperature as high as 95^oC, the total propagation delay from the lowest layer to the highest layer is negligible (less than 0.1 ps). Moreover, by generating a symmetric 3D NoC with similar number of tiles in the X, Y and Z dimensions, our technique generates architectures with lower average packet delays and energy consumption. On the other hand, depending how large the design is, the optimal number of layers may vary in terms of IC cost. For example as can be observed in [20], 4 layers were found to be more cost-effective for a 200

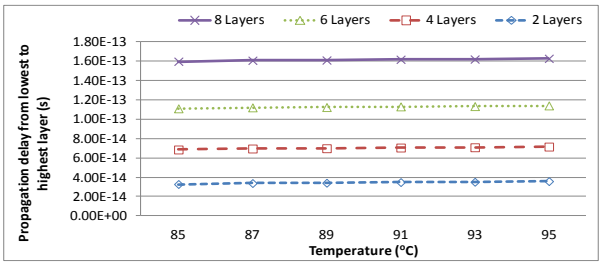


Fig. 5. Variation of inter-layer propagation delay with temperature

million-gate design; but 2 layers are considered optimal for a 100 million-gate design. Hence, the optimized size of each 3D NoC dimension is dictated by the total number of tasks available in the applications to be mapped. Assuming each task is to be mapped to a single PE and each tile has the same size.

Based on the above discussion, an initial estimation of a symmetrical NoC size is made considering the total number of tasks available. To minimize communication delays, task clusters are employed by assigning tasks with high communication dependencies as close to each other as possible. Task clusters are assigned to the NoC regions by grouping adjacent tiles in a regular or irregular shape. Unlike other clustering algorithms, the goal is to minimize the manufacturing cost as well as power consumption of 3D NoCs by utilizing vertical links only when necessary. Also, unless a loop exists, each task cluster is assigned to at least one task with high communication bandwidth requirement to balance the traffic in the network and reduce hotspots as well.

Given A_s as the number of tasks to be mapped. The first stage of this sub-problem is to estimate the size of a *symmetrical* 3D NoC (X, Y, Z) which satisfies Equation 1, such that:

$$Min(X \times Y \times Z \geq |A_s|) \tag{9}$$

In our implementation, the cube root of A_s is calculated and the dimension sizes are rounded up or down to minimize the total dimension size for satisfying equation (9).

B. Architecture matching

The main focus of this stage is to assign high bandwidth tasks in the NoC layers such that a limited number of 3D routers are assigned in the task clusters to generate an optimized inhomogeneous 3D NoC architecture. Though TSVs are expensive and complex to implement with high area overheads, TSVs are shorter than horizontal links with shorter delays. However, the crossbar of routers without TSV (2D routers) has lower power consumption and area overhead compared to that of routers with TSVs (3D routers) [8]. Moreover, 3D routers require more ports which consume a larger amount of memory resources compared to 2D routers [1]. Hence the challenge of this sub-problem is to find and generate an architecture that provides an optimized trade-off between power, energy, area and average packet delay while minimizing the manufacturing cost incurred by the TSVs.

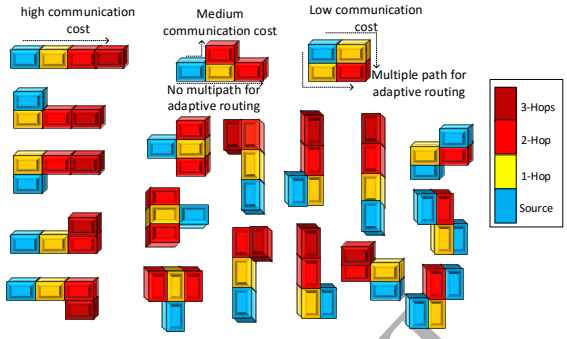


Fig. 6. Communication cost of possible mapping options to 4 tiles.

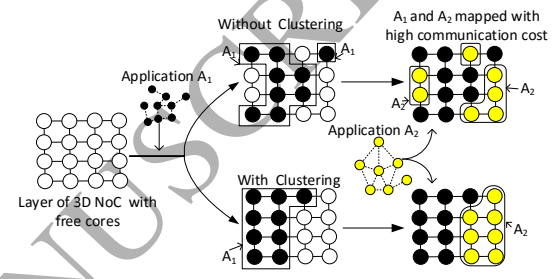


Fig. 7. Effect of task clustering in task mapping. Upper path shows mapping without considering loops which maps A_2 with a higher communication cost. The lower path employs looped communication dependencies among tasks to map the application to a clustered region of tiles. Both application A_1 and A_2 are mapped efficiently

C. Manhattan Distance, Loops and Average Hop-Count in Task Mapping

The final sub-problem of this phase is to map the application to the tiles such that the total communication energy is minimized. Most tasks in applications tend to form loops of communication dependencies. The challenge is how to maximize such loops to reduce packet delays. For example, given a group of 4 tasks, Fig. 6 shows all the possible shapes for the task to be mapped to neighbouring tiles. It can be deduced that, mapping the task to a square region (a.k.a clustered region) produces the lowest communication cost as there are more path diversity even under adaptive routing. For instance both A_1 and A_2 in Fig. 7 have tasks that form communication dependency loops. In the mapping algorithm if loops are not considered, the mapping will follow along the upper part of Fig. 7 which has extra delays with higher communication cost. However, the lower part of Fig. 7, where tasks are mapped to clustered regions of tiles, produces a lower average hop-count with lower communication cost due to the routing diversity provided by the multiple paths between any two nodes [33]. To minimize the complexity of the proposed mapping algorithm, all the tasks in a loop should not be shared with any other loop.

Hence this sub-problem can be summarized as, given the CG of application A_x find a mapping function $M : A_x \rightarrow$

NoC with the objective of

$$Min\left\{3D_{cost} = \frac{N_{3D_R}}{N_{2D_R}}\right\} \quad (10)$$

to optimize objective function 8, where N_{2D_R} and N_{3D_R} are the total number of 2D routers and 3D routers in the inhomogeneous 3D NoC architecture, respectively[2]. This can be achieved by minimizing relative number of 3D routers to 2D routers due to the higher cost of 3D routers. This stages efficiently exploits the lower area and power efficiency of 2D routers as well as the higher bandwidth and lower latency characteristics of the vertical links associated with 3D routers.

V. Robust Mapping and Architecture Generation for Inhomogeneous 3D NoCs

With reference to the sub-problems discussed above, this section presents the implementation details of the proposed technique. Before the mapping process begins, optimized 3D NoC dimensions have to be determined. Experiments conducted in Section IV-A confirm that a 3D NoC with symmetric dimensions has the most efficient average packet delay and energy consumption. Hence given $|A_s|$ as the total number of tasks available in the application to be mapped, an (X, Y, Z) symmetric 3D NoC dimension is determined by:

$$Z = X = Y = \left\lfloor \sqrt[3]{|A_s|} \right\rfloor. \quad (11)$$

Equation 11 generates an equal NoC size in each dimension if the total number of tasks to be mapped is cubic. If the number of tasks does not have a positive integer as the cube root, the total number of tasks generated will be smaller than $|A_s|$. Hence, the values of X, Y and Z in Equation 11 are increased in a stepwise manner until the total number of tiles available is enough to accommodate the total number of tasks. For example, when total number of tasks is 36, 3D NoC dimensions of $4 \times 3 \times 3$ will be generated. To reduce the number of iterations, if the fractional part of $\sqrt[3]{|A_s|}$ is greater than 0.4, the symmetric 3D NoC dimension is determined by:

$$Z = X = Y = \left\lceil \sqrt[3]{|A_s|} \right\rceil. \quad (12)$$

For instance, an application with 26 tasks has 2.5 as $\sqrt[3]{|A_s|}$. In this case a 3D NoC dimension of $(3, 3, 3)$ will be generated. To generate a suitable inhomogeneous architecture and efficiently map an application to the 3D NoC without employing time consuming exhaustive search algorithms, we present an approach which uses task clusters to balance the traffic load. Moreover, the proposed mapping approach adopts bandwidth and loop based mapping explained in Section IV-C to increase the routing efficiency of the NoC while introducing localization along the vertices to reduce the total energy consumption. The proposed mapping technique is explained in Algorithm 1. Before architecture generation and mapping of tasks to tiles, task clusters are formed by grouping tasks which have close communication dependencies. First, tasks in the application are sorted in the descending order and assigned to an initial list (*IniList*) (Line 4). To simplify the task clustering and application mapping, we employ a number of terminologies which are defined as follows:

Definition 5: The $cost$ of assigning a task to a tile t_i in relation to a mapped tile t_j is defined by Equation 13. Thus $cost$ is a direct relation of inter-tile Manhattan distance, energy, latency, and 3D to 2D router ratio:

$$cost = MD_{ij} \times EL_{cost} \times 3D_{cost} \quad (13)$$

Definition 6: Vertex.(0,0,0) is defined as the tile which is closest to the heat sink.

Initially a task cluster is created by the highest bandwidth demanding task. Afterwards, tasks that either form a loop or have a direct communication with this task are assigned to the same task cluster. To map the tasks in a given task cluster to the NoC efficiently, two lists are employed: $TaskCluster\_Masters$ and $TaskCluster\_Slaves$. Every element of the $TaskCluster\_Masters$ represents a task with the highest communication volume as a cluster initiator. However, elements of the $TaskCluster\_Slaves$ are the tasks that either form loops or have a direct communication with elements of the $TaskCluster\_Masters$ list. The creation of the two lists is initiated by assigning the first element of *IniList* to the $TaskCluster\_Masters$. The remaining elements of the initial list (from the second to the last element) are then accessed sequentially. If the element (*current task*) forms a loop with a member of the $TaskCluster\_Masters$, it is assigned to $TaskCluster\_Slaves$ list. The *current task* is then tagged with the task ID of the associated element in the $TaskCluster\_Masters$. On the other hand, if the *current task* has a direct communication with *tsk*, a member of the $TaskCluster\_Masters$, and the number of tasks in the $TaskCluster\_Slaves$ associated with the task ID of *tsk* is less than a set threshold[3], it is assigned to the $TaskCluster\_Slaves$ list. It is then tagged with the task ID of *tsk*. However, the task is automatically assigned to the $TaskCluster\_Masters$ list if it does not meet any of these criteria. First, with energy, communication delay and 3D to 2D router ratio as the cost function (Equation 13), each member of the $TaskCluster\_Masters$ list is mapped to a vertex in each cluster while neighbouring tiles of each vertex are reserved for tasks in the $TaskCluster\_Slaves$ list. Here, the energy parameter and the 3D to 2D router ratio in the cost function ensure that a minimum number of 3D routers are employed. Moreover, inter-tile Manhattan distance and latency as computational arguments in the cost function ensures that high performance 3D routers (short TSV links) are employed within the clusters while satisfying Equation 10. Thus, 3D routers when inserted in the NoC are placed in adjacent tiles between the layers where they provide minimum Manhattan distance and TSVs are inserted as vertical links. The $TaskCluster\_Master$ list is necessary for mapping

[2]We assume an area of $433,628\mu^2$ and $760,416\mu^2$ for the 2D and 3D routers, respectively [17].

[3]After conducting several experiments, a threshold Trs was found suitable if $2 \leq Trs \leq 6$. A threshold outside this range would either lead to several small clusters or very large sized clusters which would increase the complexity of the mapping algorithm. Also, other threshold values will result in a less balanced traffic load distribution which have a mapping solution with inefficient power consumption and performance.

tasks with high communication bandwidth and vertices are used for balancing traffic while maintaining a uniform distribution across the 3D NoC architecture. On the other hand, the $TaskCluster\_Slaves$ list ensures minimum communication hops to enhance localization of traffic. To begin the mapping process, the task with the highest communication volume is mapped to a $Vertex.(0,0,0)$ which is closest to the heat sink (Line 15). Then other tasks of each cluster are mapped to the tiles such that the total energy, communication delay and thermal hotspot are minimized. To ensure minimum inter-task Manhattan distance within the clusters, the tasks are assigned in a unit step to the remaining tiles that have the minimum cost. Thus, a step of $+1$ or -1 is made along the x, y or z dimension depending on the location of the vertex with minimum $cost$ in the cluster. Thus, beginning with the layer closest to the heat-sink and the cluster with the highest communicating task, each task is assigned to a tile. Consequently, tasks with high communication bandwidth are mapped to vertices to balance traffic while maintaining a uniform distribution across the 3D NoC architecture. Moreover, Equations 8 and 10 ensure that the total communication delay, TSVs and total energy constraints are met. Next, vertical links are assigned to tiles with direct interlayer communication in each cluster.

To illustrate the mapping process, Fig. 8 shows an example of an inhomogeneous architecture with a NoC size of $3\times3\times3$ and a total of 3 TSV bundles as an optimized solution generated for the Auto-indust Benchmark [56]. By considering hop-count and TSV cost alone, one might argue that an optimized solution will be to map each distinct group of tasks on each layer. Thus, the largest group in Fig. 8 can be optimally mapped to one layer. The second and third group (4 and 5 tasks in each group, respectively) can also be mapped on another layer. Finally, the group with six tasks can be mapped to the last layer. Hence there will be no need for any inter-layer link. However, intra-layer communication is much slower compared to the inter-layer communication along TSVs. This mapping approach will cause performance degradation which in turn, will also affect the energy consumption negatively. Hence, as an effective solution the mapping tool needs to consider both energy and performance effects simultaneously while minimizing the cost of TSVs. We assume that frequently communicating tasks can be mapped within the layers with minimum delay as long as there is only one hop between the tasks.

A closer look at the *CG* graph shows that though tasks $16, 17$ and 19 have the highest communication volumes (in both ingress and egress traffics directions), they form an acyclic loop with tasks $15, 18$ and 20, while task 21 has a direct communication with task 20. Hence the initial mapping phase assigns task 19 to the first element in the first cluster (19, 17, 16, 15, 18, 20) and maps it to the first vertex (first tile in the lowest layer) while tasks $15-18$ and 20 are mapped to the remaining tasks. Then tasks $21-23$ are mapped to the immediate planar and inter-planar tiles such that the total $cost$ of the mapped tasks is minimum. Task 2, the first task in the next cluster as it has a high communication volume and provides the minimum $cost$ relative to the mapped tiles, is mapped to the vertex which has the minimum $cost$. The

```
1  Initialization Phase:
2  IniList ← Sort A_x by communication volume in descending order
3  while (!IniList.Empty()) do
4      current.task = IniList.Next()
5      if (current.task form a loop with a task in
       TaskCluster_Masters) then
6          TaskCluster_Slaves.Next() ← current.task /* add task to
           TaskCluster_Slaves and tag with the ID of the corresponding
           task in TaskCluster_Masters*/
7          else if (current.task has a direct communication with a task
           tsk in TaskCluster_Masters and tasks associated with
           tsk in TaskCluster_Slaves is less than threshold) then
8              TaskCluster_Slaves.Next() ← current.task
9          end
10         else
11             TaskCluster_Masters.Next() ← current.task
12         end
13     end
14 end
15 Mapping and Architecture Generation: vertex.(0,0,0) ←
   TaskCluster_Masters.Next() . while (!TaskCluster_Masters.Empty()) do
16     current.task = TaskCluster_Masters.Next() ID ←
       TaskCluster_Masters.ID() /*Assign the ID of the current task in
       TaskCluster_Masters*/
17     for each unoccupied vertex do
18         repeat
19             if (vertex.cost is minimum) then
20                 tile ← current.task
21                 for (each unoccupied tile within a threshold
                   Manhattan distance from vertex) do
22                     repeat
23                         /* map tasks in TaskCluster_Slaves which
                           are tagged with the ID of the current task
                           in TaskCluster_Masters*/
24                         if (TaskCluster_Slaves.Next(ID) is
                           minimum) then
25                             tile ← current.task
26                         end
27                     until last tile;
28                 end
29             end
30         until last vertex;
31     end
32 end
```

Algorithm 1: Pseudocode of 3D Mapping Algorithm

immediate tiles are then assigned accordingly to tasks $0-1$ and $3-5$. The next task which gives a high communication bandwidth is task 7 and when mapped to the next vertex, it gives minimum energy consumption. Consequently task 7 is mapped to the next vertex on the last layer and the neighbouring tiles, which provides a minimum hop-count between the communication dependent tasks, assigned to tasks $6, 8$ and 9. Next, the final cluster, $\{10,11,12,13,14\}$ is assigned to a region in the remaining tiles that meets the communication bandwidth and the total number of 3D router constraints. In this case, task 11 is mapped to the vertex instead of task 12 as it generates a more optimized solution with minimum $cost$. Similarly, Fig. 9 shows an example of an inhomogeneous architecture generated for D26-dVOPD Benchmark [57]. In contrast to Auto-indust Benchmark, D26-dVOPD Benchmark has more inter-task connectivities with more cyclic loops. However, by exploiting the communication loops to balance the traffic in the network, the generated architecture has 3 TSV bundles per layer to provide a good performance to power trade-off.

A. Complexity of the Proposed Mapping Algorithm

The aim of this paper is to present a high performance and energy efficient inhomogeneous architecture generation and mapping algorithm which has a low complexity overhead. In Algorithm 1, there are three main loops which must be considered in the worst case analysis of the proposed mapping technique. The first *while* loop (Lines 3 to 14) which generates the $TaskCluster\_Slaves$ and $TaskCluster\_Masters$ has a complexity of $O(n)$, where n is the number of tasks in the given application.

The second *while* loop (Lines 15 to 32) in the mapping and architecture generation phase has two nested *for* loop (Lines 17 to 31 and Lines 21 to 28). The run time of the inner *for* loop depends on the threshold, whereas the outer *for* loop and *while* loop have run times which depend on the number of tasks. Considering the worst possible case of the inner *for* loop, where the maximum threshold value (6) is employed, the loop (Lines 21 to 28) has a constant run time which is independent of the number of task n. Contrarily, the outer *for* loop (Lines 17 to 31) and *while* loop have direct dependency on the unoccupied *vertices* and $TaskCluster\_Masters$, respectively. In the worst-case scenario where the tasks in the given application has no communication dependencies, the number of *vertices* as well as the size of the $TaskCluster\_Masters$ list can

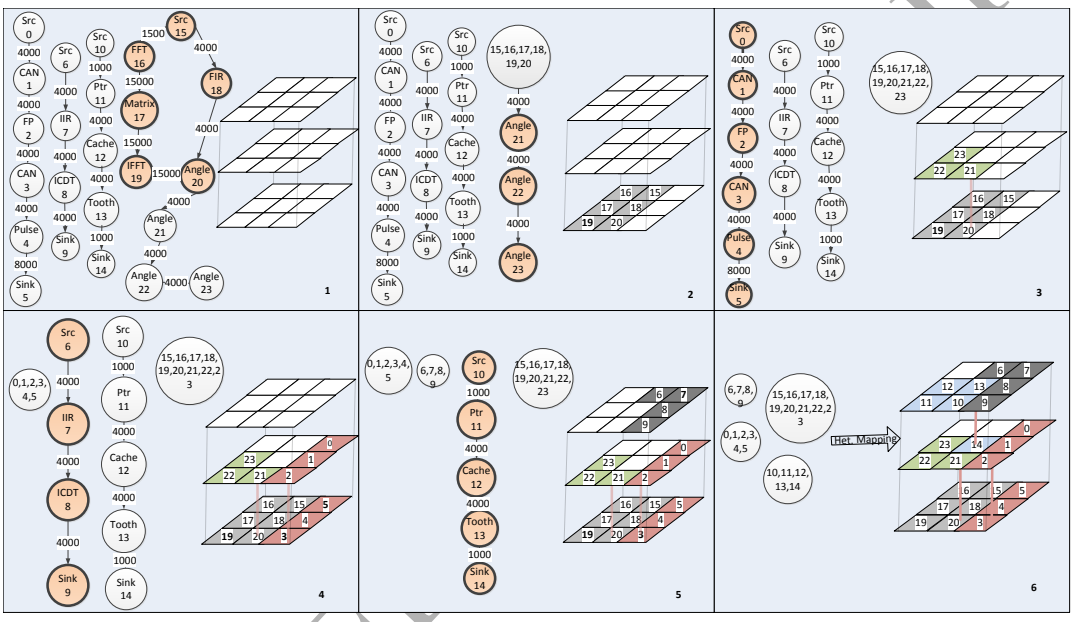


Fig. 8. Inhomogeneous mapping of Auto-indust Benchmark

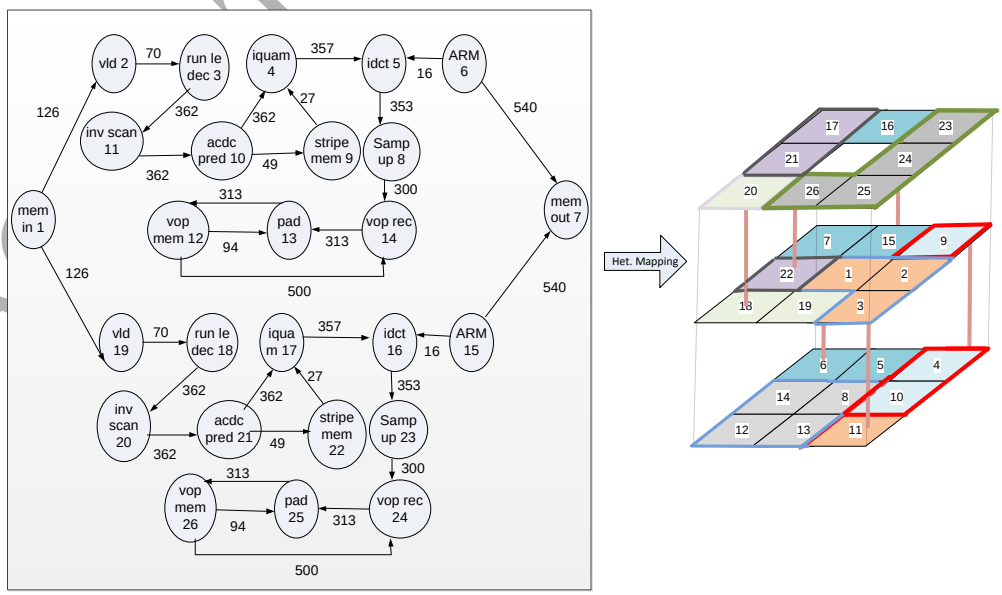


Fig. 9. Inhomogeneous mapping of D26-dVOPD Benchmark

be considered to be equal to the total number of tasks (*n*). Within the outer *for* loop (Lines 17 to 31), the algorithm runs a constant number of instructions which is repeated *n* times. This instruction is repeated for a further *n* number of times within the *while* loop (Lines 15 to 32). Hence the complexity of Lines 15 to 32 (the nested loop) is $O(n^2)$.

Also, Algorithm 1 has a partial dependency on the complexity of the sorting function in Line 2 which is $O(n \log_2 n)$. However, in the worst case $O(n \log_2)n$ (Line 2) and $O(n)$ (Lines 3 to 14) are much lower than $O(n^2)$ (Lines 15 to 32). Therefore the complexity of the proposed algorithm is $O(n^2)$.

VI. Evaluation

In order to evaluate the performance of the proposed technique, a cycle-accurate NoC simulator is used by extending $Worm\_sim$ [34], an existing 2D NoC simulator. Our extended simulator employs wormhole packet switching flow control to accurately simulate 3D NoCs with any configuration of 3D and 2D routers. We implemented the NoC components at the Register Transfer Level (RTL) in VHDL language and implemented on a $40nm$ CMOS process technology. Energy consumption of each component is then imported into the simulation platform. Energy consumption of the NoC was estimated using E_{bit} energy model [49].

As this method can be only used for a known application with a communication task graph we have selected both both synthetic and realistic benchmarks to evaluate the algorithm in different possible patterns of traffic such as typical heavy traffic loads as well as realistic scenarios similar to the applied benchmarks [34] [56][57][58]. In addition, these applications have been used in the literature [3-9][13-18], so it makes feasible to have a fair comparison to evaluate the impact of the proposed solution. In the simulation, a fixed packet size of 5 flits is used in the NoC model, as various flit sizes used which had negligible impact on the result. This is expected as changing the flit size would mainly impact the size of packets rather than the overall communication and energy consumption. In order to evaluate the performance sustainability and energy of the NoC in real-world scenarios: a complex multimedia traffic (MMS) [34], [57], Auto-indust and Telecom (from the E3S benchmark suite) [56] and an AV (Audio-visual) benchmark [58].

The setup is running for a warm-up period of 2000 cycles and performance statistics are collected after a simulation length of $200,000$ simulation cycles. Hence, by introducing different delay and energy models of 2D and 3D routers in the system, we have compared the average packet latency and energy consumption.

A. Performance Evaluation of TSV-Aware Mapping Algorithms

First, we compare the performance of the proposed mapping technique with existing ones (Map_Core_Graph [41], KL_Map_and_TSV_place [48], Branch-and-Bound [33], Onyx [38], CastNet [39], Nmap [40]) and Random mapping. For simplicity, we refer to the proposed mapping technique as *HetMap*. In our mapping approach, 3D routers are treated as an expensive resource during the mapping to automatically generate an inhomogeneous 3D NoC architecture. Similarly, Map_Core_Graph and KL_Map_and_TSV_place exploit mapping algorithms to generate inhomogeneous 3D NoC architectures. Both [41] and [48] estimate the communication cost as a product of the bandwidth requirements of an application and the hop-count between associated cores. Applications are mapped to a homogeneous 3D NoC. Vertical links are then monitored for flow and a percentage of highly utilized vertical links are preserved. The mapping process is then repeated to remap the application into the newly generated inhomogeneous 3D NoC architecture. While [48] adopts Kernighan-Lin partitioning to solve the mapping problem, [41] maps the tasks in order of their bandwidth and their relative communication cost to the mapped task. VOPD, DVOPD and 263ENC-MP3DEC benchmarks from their results are selected to evaluate the communication cost, as we have used the same benchmarks in our evaluation. VOPD, DVOPD and 263ENC-MP3DEC have 16 cores (2× 4× 2), 32 cores (4× 4× 2) and 12 cores (2× 3× 2), respectively. In the inhomogeneous 3D NoC architecture generation process in [41], [48], the number of routers with TSVs are restricted to 25%. We have forced HetMap to use the same number of TSVs and NoC dimensions for this set of experiments. The investigated mapping algorithms have a wide range of computational complexities. We have previously extended the origional 2D NoC based Branch-and-Bound mapping algorithm to 3D NoCs [6]. Similarly NMAP, CastNet and Onyx which were origionally proposed for 2D NoCs have been extended to 3D NoCs for the purpose of our study. Particularly, we have also extended Branch-and-Bound [33], Onyx [38], CastNet [39], Nmap [40] to automatically generate inhomogeneous 3D NoCs by exploiting the initial NoC architecture generation stage of the systematic approach proposed in [6]. Thus the application is mapped with the 3D version of the mapping algorithm under consideration. A full system simulation is then conducted. A constrained number of 3D routers are then placed at nodes with highly utilized vertical links to generate an inhomogeneous 3D NoC.

As Branch-and-Bound employs searching trees in finding solutions, the complexity of the mapping problem increases exponentially with the number of variables [59], [60]. In contrast, the complexity of HetMap is $O(n^2)$, while CastNet, Onyx and Nmap have a complexity of $O(en^2)$, $O(n^2 \log_2 n)$ and $O(n^4 \log_2 n)$, respectively. Here, e is the number of edges representing the minimal path candidates of a mapping solution. So far we have compared the inhomogeneous 3D NoCs generated by our mapping algorithm with inhomogeneous 3D NoC architectures which have been mapped with existing mapping algorithms. Table II shows a comparison of the communication cost in terms of bandwidth (identified by the edge labels in core graph) and hop-count [41], [48] of the above existing approaches and the proposed HetMap application mapping and inhomogeneous 3D NoC architecture generation technique. A the communication cost is how much data and how far the data needs to be transferred, here communication cost of an application is defined to be equal to the sum of product of bandwidth requirement of pairs of cores and number of hops between the corresponding routers

Benchmark	Branch-and-Bound'	CastNet'	Onyx'	Nmap'	[41]	[48]	HetMap
VOPD	4118.10	4205	4264	4270	4265	4189	4119
DVOPD	9540.21	9639	9709	9729	9640	9726	9542
263ENC-MP3DEC	228.3	229.21	230	230.11	230.43	230.47	228.23

TABLE II
COMPARISON OF COMMUNICATION COST' OF DIFFERENT TSV-AWARE MAPPING ALGORITHMS

to which the cores are attached. We adopted this definition from [40], which has been used widely used such as in [41], [48].By extending the performance of the existing mapping algorithms with the systematic approach, the communication costs of these techniques are significantly reduced. As can be seen in Table II, CastNet, Onyx, Nmap', [41] and [48] have similar communication costs. Compared to these algorithms, HetMap generates inhomogeneous 3D NoC architectures with lower communication cost (with the improved Branch-and-Bound as its competitor) due to optimized allocation of TSVs during the application mapping. Moreover, the load balancing approach of application partitioning in HetMap significantly reduces the communication cost among the mapped tasks. In addition to HetMap generating architectures with lower communication cost compared to the approach presented in [48], the complexity of the Kernighan-Lin partitioning algorithm adopted in [48] is much higher ($O(n^3)$) than that of HetMap ($O(n^2)$). Moreover, the complex mapping algorithm in [48] must be repeated to successfully generate the inhomogeneous 3D NoC.

In addition, we have compared the average CPU time required by each of the techniques to successfully map the investigated benchmarks to 3D NoCs[4]. As shown in Table III, on average, the CPU time required by HetMap is similar to that of CastNet', Onyx', Nmap' and even RandomMap, though HetMap produces 3D NoCs with much lower average packet latency. By employing vertices and load balancing, only a few tiles are considered during cost consideration and clustering in HetMap. Hence though several cost functions are considered, HetMap has a short computational time. Despite the fact that HetMap has a lower complexity compared to Onyx', its average CPU time in table III is slightly higher than that of Onyx'. This is because under a considerably small number of nodes the computational time for mapping the tasks in the TaskCluster_Slaves list under HetMap (Algorithm 1 Lines 17 to 31) affects the total CPU times. However, for a relatively large number of nodes, the average CPU time required by HetMap is much lower than that of Onyx'. Due to their iterative nature of the application mapping process, [41], [48] have relatively long CPU time. Although Branch-and-Bound' is comparable with HetMap in terms of average packet latency, its computational time is significantly higher (over 20 minutes [33]). Additionally, Branch-and-Bound' employs exhaustive search trees, and bandwidth and loop based mapping which has a high complexity. Fig. 10 shows that the proposed mapping technique has lower average packet latency compared to existing heuristic mapping algorithms. This is expected as packets in HetMap experience shorter delays due to even distribution of localised traffic in the inhomogeneous architectures. Moreover, the average packet latency of 3D NoCs under HetMap is similar to that of Branch-and-Bound' mapping algorithm. It should be noted that Branch-and-Bound' explores all possible tasks mappings using a search tree and selects the most cost effective (in this case the cost parameter is the latency and energy) mapping solutions. Also by employing clusters and using vertices to balance traffic in the NoC, the average packet energy consumption of NoCs mapped with HetMap is lower than Branch-and-Bound' though they both employ bandwidth and loop mapping. Existing mapping algorithms take the network size and architecture as inputs and generate a suitable task to tile mapping based on the given application and configuration. For a fair comparison, the same NoC size and architecture was used in all cases. However, experimental results of HetMap emphasize on the significance of accounting for the NoC size and vertical links during optimization of task mapping.

Fig. 11 summarizes the average packet energy of the mapping

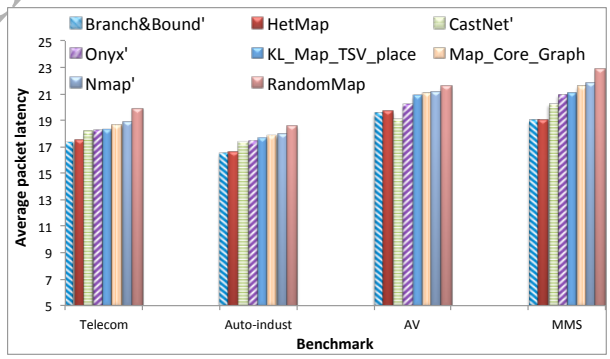


Fig. 10. Comparison of average packet latency of various 3D NoC mapping techniques: Onyx [38], Branch-and-Bound' [6], CastNet [39], Nmap [40] and HetMap (proposed technique).

techniques under various benchmarks. It can be observed that packets in 3D NoC mapped with HetMap have lower energy consumption when compared to other mapping techniques.

Algorithm	Average CPU Time (s)
Branch& Bound'	1500
HetMap	27.7
CastNet'	28.905
Onyx'	26.61
Nmap '	29.67
RandomMap	3.98
Map_Core_Graph	80.68
KL_Map_and_TSV_place	60.73

TABLE III
AVERAGE CPU TIME OF VARIOUS MAPPING TECHNIQUES

[4]As the main focus is on the performance of the mapping algorithm, CPU time required by the application simulation phase of the systematic approach is not considered here.

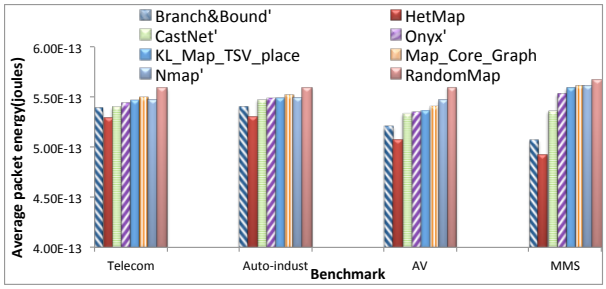


Fig. 11. Average packet energy of 3D NoCs under various mapping techniques.

This is mainly due to the reduced paths and balanced traffic loads both in the planar and inter-planar regions.

Fig. 12 shows that by reducing the number of 3D routers while

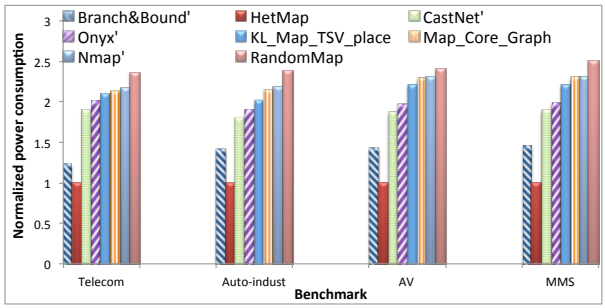


Fig. 12. Normalized total power of 3D NoCs under various mapping techniques.

maintaining the regularity of the 3D NoC in the first place, HetMap has a much lower power consumption compared to existing heuristic mapping techniques. Moreover, Hetmap and Branch-and-Bound' have similar power consumption, though the same number of 3D routers was used in both cases.

To emphasize on the performance benefits of the proposed mapping technique, we have applied a varied set of realistic benchmarks: D_36_4, D_64_4, D_26_media, D_62_pvopd and D_38_tvopd [57]. Similar to Fig. 11, it can be noticed in Fig. 13 that, with the exception of Branch-and-Bound which has similar performance but with higher complexity, HetMap generates 3D NoCs with lower average packet latency compared to existing mapping algorithms. Particularly it can be noted that as the number of nodes and communication dependencies increase, the performance efficiency of HetMap over Onyx, CastNet, Nmap and Random mapping techniques increases, though 3D NoC architectures in HetMap have less 3D routers and TSVs.

Moreover as shown in Fig. 14, the proposed mapping technique has lower average packet energy when compared to Onyx', Map_Core_Graph, Nmap', KL_Map_and_TSV_place, and CastNet' algorithms in all cases. Since RandomMap does not consider the communication dynamics of the NoC, it has the highest average packet energy in all cases. Also, it can be seen that though Branch-and-Bound' tries to find the most energy efficient and reliable mapping solution by using exhaustive search and heuristics which can be time consuming, the average packet energy of HetMap and Branch-and-Bound'

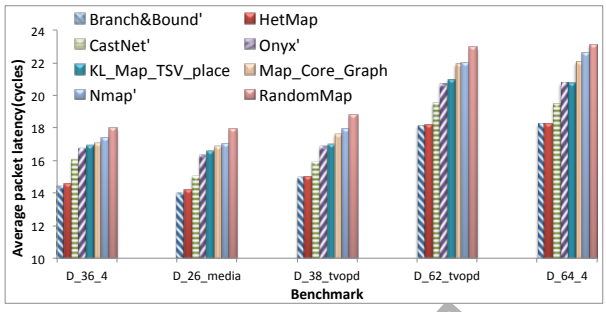


Fig. 13. Average packet latency of 3D NoCs under various mapping techniques under D_36_4, D_64_4, D_26_media, D_62_pvopd and D_38_tvopd benchmarks [57].

are similar in all cases. The reduced packet-energy in HetMap compared to Branch-and-Bound' is mainly due to the impact of the balanced traffic and loop-based mapping in HetMap which in turn reduces hotspots. Similarly, Fig. 15 shows that

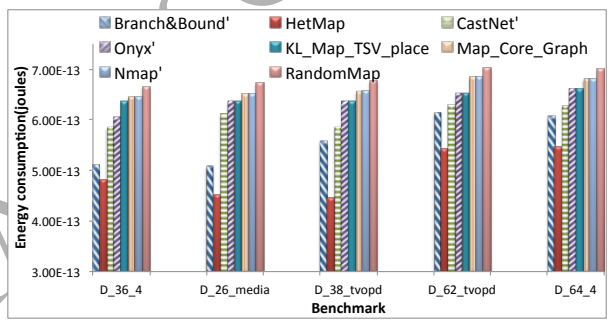


Fig. 14. Average packet energy of 3D NoCs under various mapping techniques under D_36_4, D_64_4, D_26_media, D_62_pvopd and D_38_tvopd benchmarks [57].

Branch-and-Bound' and HetMap have similar power consumption in all cases. Moreover, it can be noted that NoCs generated with HetMap consume less power when compared to NoC mapped with Onyx, CastNet and Random mapping techniques. It can be summarized that, the proposed mapping scheme

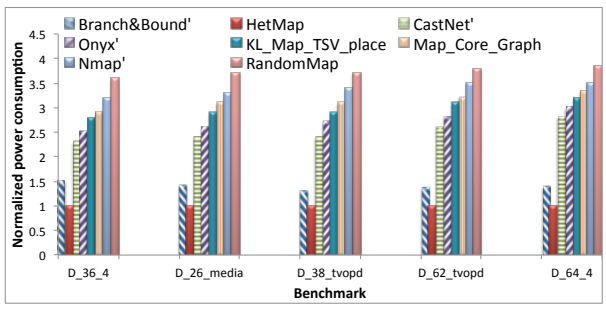


Fig. 15. Normalized total power of 3D NoCs under D_36_4, D_26_media and D_38_tvopd benchmarks [57].

generates 3D NoCs with lower average packet latency and energy consumption compared to existing mapping techniques such as Onyx', Map_Core_Graph, Map_Core_Graph, Nmap' and CastNet'. While the average packet latency and average packet energy of the proposed mapping technique and that

of Branch-and-Bound' are similar, the proposed technique efficiently map applications in a much shorter period of time compared to Branch-and-Bound' algorithm. Moreover in the next section, we demonstrate that the proposed approach generates inhomogeneous architectures with a higher performance and lower power consumption compared to existing inhomogeneous 3D NoCs. In the next section, we compare the average packet latency and energy of inhomogeneous architectures generated by the proposed technique and existing inhomogeneous architectures mapped with the Branch-and-Bound' mapping technique under realistic traffic cases with high core density and high communication dependencies.

B. Performance Evaluation of Inhomogeneous Architectures

To analyze the performance benefits of inhomogeneous architectures generated by the proposed technique, we performed experiments with various mapping algorithms and existing inhomogeneous architectures [7], [4], [5]. For the existing mapping algorithms, we observed consistent characteristics with best performance recorded with Branch-and-Bound' algorithm. Hence, in this section we present our analysis for existing inhomogeneous architectures which have been mapped with Branch-and-Bound' mapping algorithm and compare them with inhomogeneous architectures generated by the proposed technique. Fig. 16 summarizes the average packet latency of various inhomogeneous architectures under different realistic benchmarks. By evenly balancing the intra-layer and inter-layer traffic loads and optimizing the number of 3D routers while minimizing the average hop-count, HetMap generates inhomogeneous architectures which have much lower average packet latencies compared to existing inhomogeneous architectures. This is expected as though existing hop-count based inhomogeneous architectures have evenly distributed 3D routers, they apply fixed 3D placement algorithms which have dynamic packet delays under different application benchmarks. The proposed technique however, considers the traffic dynamics, communication and power constraints, and generates an optimized inhomogeneous architecture with minimum number of TSVs and efficiently map applications to the architecture. Particularly, Fig. 17 shows that the cost in terms of number of TSVs and 3D routers of inhomogeneous architectures generated by the proposed technique is much less than that of the existing inhomogeneous architectures.

Moreover, Figs.18 and 19 show that the architectures generated by the proposed technique have lower average packet energy and are more power efficient compared to the existing inhomogeneous architectures. This is mainly due to the fact that HetMap generates inhomogeneous architectures with less number of 3D routers and more evenly distributed traffic with lower average hop-counts compared to existing inhomogeneous architectures.

C. Performance Evaluation of TSV Reduction Scheme

Inhomogeneous architectures investigated in Section VI-B reduce the area overhead and the manufacturing cost of the NoC by combining 2D and 3D routers in 3D NoC. Thus by introducing 2D routers in the NoC, total number of TSVs is

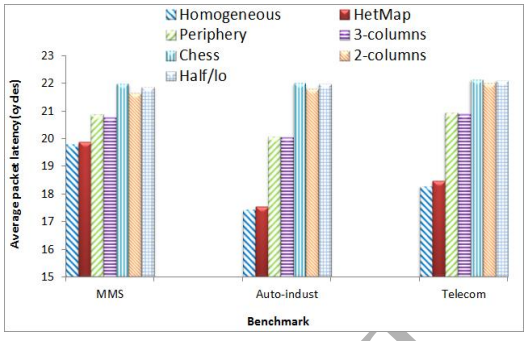


Fig. 16. Average packet latency of various inhomogeneous architectures.

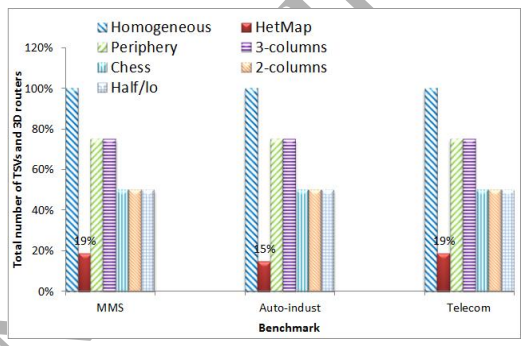


Fig. 17. Total number of TSVs and 3D routers available in various architectures.

reduced. According to [25], if the number of TSVs exceeds a threshold, serialization of a certain degree is adopted to reduce the total number of TSVs. Here, the bandwidth of the TSVs is reduced by the degree of serialization and transceivers are employed at the router interface to the TSVs for repacketization. In this Section, we compare the average packet latency and power consumption of architectures generated by the TSVs reduction technique proposed in HetMap, the serialization technique and that of existing inhomogeneous architectures. For a fair comparison, a serialization degree of 2 (64-bit vertical links reduced to 32-bit links for serialization) is used. Thus, a total of 50% TSVs is present in the architectures compared to a homogeneous 3D NoC of the same NoC dimensions.

Fig. 20 shows the average packet latency of various TSV reduction techniques at various traffic loads. It can be seen when the bandwidth of the TSVs are reduced with serialization logic, the average packet latency of the NoC is much higher than other schemes with equivalent number of TSVs. HetMap on the other hand saturates with a higher traffic load compared to other schemes. This is because HetMap generates architectures with evenly distributed 2D and 3D routers based on the dynamics of the NoC traffic pattern. It can be seen in Fig. 21, that serialization causes an increase in the total power consumption. The power consumption of serialization is even higher than that of homogeneous 3D Mesh. This is because the power consumption overhead of the serialization receiver and transmitter logic is much higher compared to the power savings on the TSVs. However, other TSV reduction schemes

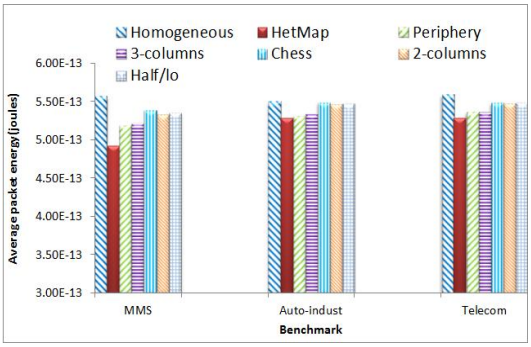


Fig. 18. Average packet energy of various inhomogeneous architectures.

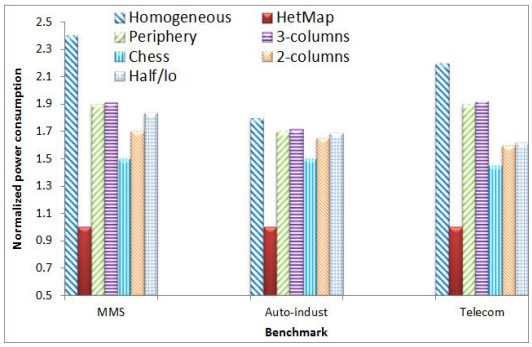


Fig. 19. Normalized total power of various inhomogeneous architectures.

such as HetMap, 2-columns and chess have a much lower average power consumption compared to homogeneous 3D mesh. This is expected as these architectures have a reduced number of 3D routers which are uniformly distributed in the NoC to reduce total average hop-count.

VII. Conclusion

An application mapping algorithm is proposed for inhomogeneous 3D NoCs in this work to improve the efficiency in terms of energy and communication delays for emerging SoC design. The proposed algorithm employs bandwidth-constrained and loop based mapping to minimize power consumption of NoCs. Moreover, the proposed algorithm efficiently generates optimized inhomogeneous architectures with limited number of 3D routers without exhaustively searching

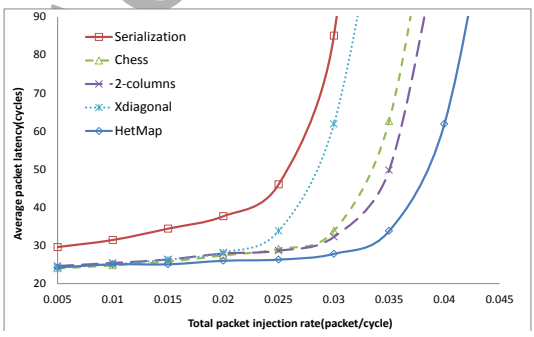


Fig. 20. Average packet latency of various TSV reduction schemes.

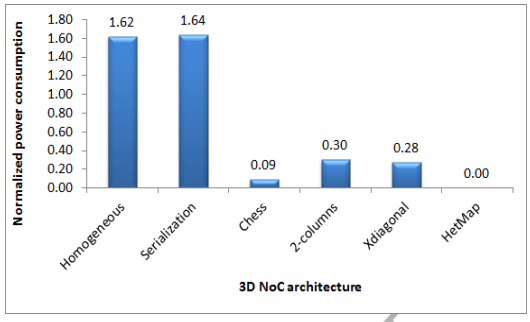


Fig. 21. Normalized power consumption of various TSV reduction schemes.

all possible solutions. Experimental analysis under various realistic case studies shows that 3D NoCs mapped by the proposed approach have much lower average packet delay and energy compared to existing heuristic mapping algorithms (Map_Core_Graph, KL_Map_and_TSV_place, CastNet, Nmap and Onyx). Moreover, the proposed approach is more efficient than Branch-and-Bound mapping in terms of total power consumption though they have similar average packet delays. Also, the application mapping runtime of the proposed technique is comparable to CastNet, Nmap, Onyx and much lower than Map_Core_Graph, KL_Map_and_TSV_place and Branch-and-Bound. Specifically, the proposed algorithm has lower complexity compared to the existing inhomogeneous 3D NoC mapping algorithms (Map_Core_Graph, KL_Map_and_TSV_place). By evenly redistributing the traffic while localizing the communication dependencies, the proposed technique generates inhomogeneous 3D NoCs with minimum number of 3D routers which have less average packet latencies and more energy efficiencies.

References

[1] D. Velenis, M. Stucchi, E. Marinissen, B. Swinnen, and E. Beyne, "Impact of 3d design choices on manufacturing cost," in *IEEE International Conference on 3D System Integration (3DIC)*, 2009, pp. 1 – 5.

[2] M. O. Agyeman and A. Ahmadinia, "Optimised application specific architecture generation and mapping approach for heterogeneous 3d networks-on-chip," in *IEEE International Conference on Computational Science and Engineering*, 2013, pp. 794–801.

[3] K. Siozios, A. Bartzas, and D. Soudris, "Three dimensional network-on-chip architectures," in *Networks-on-Chips: Theory and Practice*, H. E. Fayez Gebali and M. W. El-Kharashi, Eds. CRC Press, 2009, pp. 1–28.

[4] M. O. Agyeman, A. Ahmadinia, and A. Shahrabi, "Low power heterogeneous 3d networks-on-chip architectures," in *International Conference on High Performance Computing and Simulation (HPCS)*, 2011, pp. 533 –538.

[5] ——, "Heterogeneous 3d network-on-chip architectures: area and power aware design techniques," *Journal of Circuits, Systems and Computers*, vol. 22, no. 4, p. 1350016, 2013.

[6] M. O. Agyeman, A. Ahmadinia, and N. Bagherzadeh, "Performance and energy aware inhomogeneous 3d networks-on-chip architecture generation," *IEEE Transactions on Parallel and Distributed Systems*, vol. PP, no. 99, pp. 1–1, 2015.

[7] T. Xu, P. Liljeberg, and H. Tenhunen, "A study of Through Silicon Via impact to 3D Network-on-Chip design," in *International Conference On Electronics and Information Engineering (ICEIE)*, 2010, pp. 333 – 337.

[8] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, V. Narayanan, M. S. Yousif, and C. R. Das, "A novel dimensionally-decomposed router for on-chip communication in 3D architectures," *SIGARCH Comput. Archit. News*, vol. 35, no. 2, pp. 138–149, 2007.

[9] K. C. Chen, S. Y. Lin, H. S. Hung, and A. Y. A. Wu, "Topology-aware adaptive routing for nonstationary irregular mesh in throttled 3d noc systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 10, pp. 2109–2120, 2013.

[10] X.-H. Wang, P. Liu, M. Yang, M. Palesi, Y.-T. Jiang, and M. C. Huang, "Energy efficient run-time incremental mapping for 3-d networks-on-chip," *Journal of Computer Science and Technology*, vol. 28, no. 1, pp. 54–71, 2013.

[11] N. Dahir, R. Al-Dujaily, T. Mak, and A. Yakovlev, "Thermal optimization in network-on-chip-based 3d chip multiprocessors using dynamic programming networks," *ACM Trans. Embed. Comput. Syst.*, vol. 13, no. 4s, pp. 139:1–139:25, Apr. 2014.

[12] P. Vivet, Y. Thonnart, R. Lemaire, E. Beigne, C. Bernard, F. Darve, D. Lattard, I. Miro-Panades, C. Santos, F. Clermidy, S. Cheramy, F. Petrot, E. Flamand, and J. Michailos, "8.1 a 4x4x2 homogeneous scalable 3d network-on-chip circuit with 326mflit/s 0.66pj/b robust and fault-tolerant asynchronous 3d links," in *International Solid-State Circuits Conference (ISSCC)*, 2016, pp. 146–147.

[13] M. O. Agyeman and W. Zong, "An efficient 2d router architecture for extending the performance of inhomogeneous 3d noc-based multi-core architectures," in *2016 International Symposium on Computer Architecture and High Performance Computing Workshops (SBAC-PADW)*, 2016, pp. 79–84.

[14] D. Matos, M. Prass, M. Kreutz, L. Carro, and A. Susin, "Performance evaluation of hierarchical noc topologies for stacked 3d ics," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 1961–1964.

[15] M. H. Jabbar, D. Houzet, and O. Hammami, "Impact of 3d ic on noc topologies: A wire delay consideration," in *Euromicro Conference on Digital System Design (DSD)*, 2013, pp. 68–72.

[16] B. Feero and P. Pande, "Networks-on-Chip in a Three-Dimensional Environment: A Performance Evaluation," *IEEE Transaction on Computers*, vol. 58, pp. 32–45, 2009.

[17] D. Park, S. Eachempati, R. Das, A. K. Mishra, Y. Xie, N. Vijaykrishnan, and C. R. Das, "Mira: A multi-layered on-chip interconnect router architecture," in *International Symposium on Computer Architecture*, 2008, pp. 251–261.

[18] L. P. Carloni, P. Pande, and Y. Xie, "Networks-on-chip in emerging interconnect paradigms: Advantages and challenges," in *ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, 2009, pp. 93–102.

[19] F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir, "Design and Management of 3D Chip Multiprocessors Using Network-in-Memory," in *Int. Symposium on Computer Architecture (ISCA)*, 2006, pp. 130–141.

[20] Y. Xie, J. COng, and S. Sapatneker, "System-level 3d ic cost analysis and design exploration," in *Three Dimensional Integrated Circuit Design*, 2010, pp. 261–280.

[21] M. O. Agyeman and A. Ahmadinia, "A systematic generation of optimized heterogeneous 3d networks-on-chip architecture," in *NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2013, pp. 79–83.

[22] C. Liu, L. Zhang, Y. Han, and X. Li, "Vertical interconnects squeezing in symmetric 3D mesh Network-on-Chip," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2011, pp. 357 –362.

[23] Y. Wang, L. Zhang, Y. Han, and H. L. and Xiaowei Li, "Economizing TSV Resources in 3D Network-on-Chip Design," *IEEE Transactions on VLSI Systems*, 2014.

[24] M. O. Agyeman, A. Ahmadinia, and A. Shahrabi, "Efficient routing techniques in heterogeneous 3d networks-on-chip," *Parallel Computing*, vol. 39, no. 9, pp. 389–407, 2013.

[25] S. Pasricha, "Exploring serial vertical interconnects for 3d ics," in *Design Automation Conference (DAC)*, 2009, pp. 581 –586.

[26] ——, "A framework for tsv serialization-aware synthesis of application specific 3d networks-on-chip," in *International Conference on VLSI Design (VLSID)*, 2012, pp. 268 –273.

[27] C. Seiculescu, S. Murali, L. Benini, and G. De Micheli, "SunFloor 3D: A tool for Networks On Chip topology synthesis for 3D systems on chips," 2009, pp. 9–14.

[28] T. C. Xu, G. Schley, P. Liljeberg, M. Radetzki, J. Plosila, and H. Tenhunen, "Optimal placement of vertical connections in 3d network-on-chip," *Journal of Systems Architecture*, vol. 59, no. 7, pp. 441 – 454, 2013.

[29] F. Miller, T. Wild, and A. Herkersdorf, "Tsv-virtualization for multi-protocol-interconnect in 3d-ics," in *Euromicro Conference on Digital System Design (DSD)*, 2012, pp. 374–381.

[30] A. Bose, P. Ghosal, and S. P. Mohanty, "A low latency scalable 3d noc using bft topology with table based uniform routing," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI),*, 2014, pp. 136–141.

[31] S. R. Rose, A., "Genetic algorithm based optimization of vertical links for efficient 3d noc multicore crypto processor," *International Symposium on Quality Electronic Design (ISQED)*, vol. 8, pp. 1082–1090, 2013.

[32] M. Bahmani, A. Sheibanyrad, F. Ptrot, F. Dubois, and P. Durante, "A 3d-noc router implementation exploiting vertically-partially-connected topologies," in *IEEE Symposium on VLSI (ISVLSI)*, Aug 2012, pp. 9–14.

[33] C. Ababei, H. S. Kia, O. P. Yadav, and J. Hu, "Energy and reliability oriented mapping for regular networks-on-chip," in *NOCS*, 2011, pp. 121–128.

[34] J. Hu and R. Marculescu, "Energy- and performance-aware mapping for regular NoC architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 4, pp. 551–562, 2005.

[35] X. Wang, M. Yang, Y. Jiang, and P. Liu, "A power-aware mapping approach to map ip cores onto nocs under bandwidth and latency constraints," *TACO*, vol. 7, no. 1, 2010.

[36] L. S. Indrusiak, "End-to-end schedulability tests for multiprocessor embedded systems based on networks-on-chip with priority-preemptive arbitration," *Journal of Systems Architecture*, vol. 60, no. 7, pp. 553 – 561, 2014.

[37] A. K. Singh, P. Dziurzanski, H. R. Mendis, and L. S. Indrusiak, "A survey and comparative study of hard and soft real-time dynamic resource allocation strategies for multi-/many-core systems," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 24:1–24:40, 2017.

[38] M. Jamidarmian, A. Khademzadeh, and M. Tavanpour, "Onyx: A new heuristic bandwidth-constrained mapping of cores onto tile-based Network on Chip," *Ieice Electronic Express*, vol. 6, pp. 1–7, 2009.

[39] S. Tosun, "New heuristic algorithms for energy aware application mapping and routing on mesh-based nocs," *Journal of Systems Architecture*, vol. 57, no. 1, pp. 69 – 78, 2011.

[40] S. Murali and G. De Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," in *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2004, pp. 896 – 901.

[41] K. Manna, S. Chattopadhyay, and I. Sengupta, "Through silicon via placement and mapping strategy for 3d mesh based network-on-chip," in *International Conference on VLSI-SoC*, 2014, pp. 1–6.

[42] W. Z. e. a. Wang J, Li L, "Energy-efficient mapping for 3d noc using logistic function based adaptive genetic algorithms," *Chin J Electron*, vol. 23, no. 2, p. 254262, 2014.

[43] J. Wang, L. Li, H. Pan, S. He, and R. Zhang, "Latency-aware mapping for 3d noc using rank-based multi-objective genetic algorithm," in *IEEE International Conference on ASIC (ASICON)*, 2011, pp. 413–416.

[44] H. Elmiligi, F. Gebali, and M. W. El-Kharashi, "Power-aware Mapping for 3D-NoC Designs Using Genetic Algorithms," *Procedia Computer Science*, vol. 34, pp. 538 – 543, 2014.

[45] G. Feng, F. Ge, S. Yu, and N. Wu, "A thermal-aware mapping algorithm for 3d mesh network-on-chip architecture," in *International Conference on ASIC (ASICON)*, 2013, pp. 1–4.

[46] X.-H. Wang, P. Liu, M. Yang, M. Palesi, Y.-T. Jiang, and M. C. Huang, "Energy efficient run-time incremental mapping for 3-d networks-on-chip," *Journal of Computer Science and Technology*, vol. 28, no. 1, pp. 54–71, 2013.

[47] C.-L. Chou and R. Marculescu, "Run-time task allocation considering user behavior in embedded multiprocessor networks-on-chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 1, pp. 78 –91, 2010.

[48] K. Manna, V. S. S. Teja, S. Chattopadhyay, and I. Sengupta, "Tsv placement and core mapping for 3d mesh based network-on-chip design using extended kernighan-lin partitioning," in *IEEE Annual Symposium on VLSI (ISVLSI)*, 2015, pp. 392–397.

[49] T. T. Ye, G. D. Micheli, and L. Benini, "Analysis of power consumption on switch fabrics in network routers," in *Design Automation Conference (DAC)*, 2002, pp. 524 – 529.

[50] Y. H. Yuanqing Cheng, Lei Zhang and X. Li, "Thermal-constrained task allocation for interconnect energy reduction in 3-d homogeneous mpsocs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 2, pp. 239–249, 2013.

[51] M. Mandelli, A. Amory, L. Ost, and F. G. Moraes, "Multi-task dynamic mapping onto noc-based (mpsocs)," in *Proceedings of the 24th Symposium on Integrated Circuits and Systems Design*, 2011, pp. 191–196.

[52] W.-L. Hung, G. M. Link, Y. Xie, N. Vijaykrishnan, and M. J. Irwin, "Interconnect and Thermal-aware Floorplanning for 3D Microprocessors,"

in *International Symposium on Quality Electronic Design (ISQED)*, 2006, pp. 98–104.

[53] J. Hu and R. Marculescu, "Application-specific buffer space allocation for networks-on-chip router design," in *IEEE/ACM International conference on Computer-aided design*, ser. ICCAD '04, Washington, DC, USA, 2004, pp. 354–361.

[54] V. F. Pavlidis and E. G. Friedman, "3-D topologies for networks-on-chip," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 10, pp. 1081–1090, 2007.

[55] Y. Cao, "Predictive Technology Model (PTM)," http://ptm.asu.edu/.

[56] R. Dick, "Embedded system synthesis benchmarks suite(e3s)," ziyang.eecs.umich.edu/dickrp/e3s.

[57] C. Seiculescu, S. Murali, L. Benini, and G. De Micheli, "3d Network on Chip Topology Synthesis: Designing Custom Topologies for Chip Stacks," in *3D Integration for NoC-based SoC Architectures*, A. Sheibanyrad, F. Petrot, and A. Jantsch, Eds. Springer New York, 2011, pp. 193–223.

[58] V. Dumitriu and G. Khan, "Throughput-oriented noc topology generation and analysis for high performance socs," *VLSI*, vol. 17, no. 10, pp. 1433 –1446, 2009.

[59] N. Thakoor and J. Gao, "Branch-and-bound for model selection and its computational complexity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, pp. 655–668, 2011.

[60] M. Sun, M. Telaprolu, H. Lee, and S. Savarese, "Efficient and exact MAP-MRF inference using branch and bound," in *AISTATS*, 2012.

BIOGRAPHY

Michael Opoku Agyeman received the Ph.D. from the Department of Computing at Glasgow Caledonian University (GCU), Glasgow, in 2014. From 2014 - 2015, he was with the Intel Embedded System Research group of The Chinese University of Hong Kong (CUHK) as a Research Associate. Currently he is a lecturer at the Department of Computing and Immersive Technologies at the University of Northampton. His research interests include VLSI SoC design, wired and wireless NoCs.

Ali Ahmadinia received his Ph.D. degree from University of Erlangen-Nuremberg, Germany, in 2006. In 2004-2005, he worked as a research associate in Electronic imaging group, Fraunhofer Institute - Integrated Circuits (IIS), Erlangen, Germany. In 2006-2008, he was a research fellow in the School of Engineering and Electronics, University of Edinburgh, Edinburgh, UK. In 2008, he joined GCU, Glasgow, UK, where he was a senior lecturer in embedded systems. He is currently a faculty member of Department of Computer Science in California State University San Marcos, US. His research has resulted more than 100 international journal and conference publications in the areas of reconfigurable computing and system-on-chip design, wireless and DSP applications.

Nader Bagherzadeh (F'14) is a professor of computer engineering in the department of electrical engineering and computer science at the University of California, Irvine, where he served as a chair from 1998 to 2003. Dr Bagherzadeh has been involved in research and development in the areas of: computer architecture, reconfigurable computing, VLSI chip design, network-on-chip, 3D chips, sensor networks, and computer graphics since he received a Ph.D. degree from the University of Texas at Austin in 1987. He is a Fellow of the IEEE. Professor Bagherzadeh has published more than 250 articles in peer-reviewed journals and conferences. He has trained hundreds of students who have assumed key positions in software and computer systems design companies in the past twenty five years. He has been a PI or Co-PI on more than $8 million worth of research grants for developing next generation computer systems for applications in general purpose computing and digital signal processing.