2012 25th International Conference on VLSI Design

A Framework for TSV Serialization-aware Synthesis of Application Specific 3D Networks-on-Chip

Sudeep Pasricha
Colorado State University, Fort Collins, CO, USA
sudeep@colostate.edu

Abstract *– With increasing performance-per-watt implementation requirements for emerging applications and barriers in interconnect scaling for ultra-deep submicron (UDSM) technologies, traditional 2D integrated circuits (2D-ICs) are being pushed to their limit. Three dimensional integrated circuits (3D-ICs) have recently emerged as a promising solution that can overcome many of the performance, area, and power concerns in 2D-ICs. In this paper we propose a novel framework (MORPHEUS) for the synthesis of application-specific 3D networks on chip (NoCs). The goal is to generate 3D NoCs that meet application performance constraints while minimizing power dissipation. MORPHEUS incorporates thermal-aware core layout, 3D topology and route generation, and placement of network interfaces (NIs), routers, and serialized vertical through silicon vias (TSVs). Experimental studies on several chip multiprocessor (CMP) applications indicate that our generated solutions notably reduce power dissipation (up to 2.3×) and average latency (up to 1.2×) over 2D NoCs. Comparisons with a previous work on application-specific 3D NoC synthesis also show improvements in power dissipation (up to 1.9×) and average latency (up to 1.6×).*

I. INTRODUCTION

The rise in application complexity in recent years together with the need for power efficiency in computing systems has led to more and more cores being integrated on a single chip. Such chip multiprocessors (CMPs) have demonstrated superior performance-per-watt than their uniprocessor counterparts. If the trend of integrating greater number of cores on a chip is to continue in the coming years, two major challenges need to be overcome. Firstly, the ongoing reduction in lithographic features is becoming increasingly more expensive. As a result, there is a practical limitation on the number of cores that can be integrated viably on the single active layer available to CMPs today. Secondly, with rising core counts, the amount of wires on chip has been steadily rising. Due to effects such as parasitic resistivity, crosstalk, and electromigration interference in UDSM nodes, long global interconnects have become a major delay and power bottleneck [1]. According to the ITRS [2], focus on new technologies and methodologies and not further reduction in feature size is the key to further performance-per-watt enhancements.

Of the several different disruptive technologies that are being investigated today, 3D integrated circuits (3D-ICs) with wafer-to-wafer bonding technology is one of the most promising candidates that can achieve power, performance, cost, and area demands of emerging applications in the coming years [3]-[6]. In wafer-to-wafer bonded 3D-ICs, active devices (processors, memories) are placed on multiple layers and vertical Through Silicon Vias (TSVs) are used to connect components across the stacked layers. Multiple active layers in 3D-ICs can enable increased integration of cores within the same area footprint as traditional single layer 2D-ICs. In addition, long global interconnects between cores can be replaced by much shorter inter-layer TSVs, improving performance and reducing on-chip power dissipation. Recent 3D-IC test chips from IBM [3][4] and Tezzaron [5] have confirmed the benefits of 3D integration technology.

With the advent of many-core CMPs in recent years, the on-chip communication fabric has been evolving to cope with increased bandwidth and reliability requirements. Traditionally used bus based architectures have given way to packet switched network on chips (NoCs) that offer higher bandwidth, reliability, and scalability in UDSM technologies. With the introduction of 3D-ICs, it is expected that NoC fabrics will be extended into the third dimension. Recent research has begun exploring various 3D NoC topologies [7]-[9] and shown significant performance improvements for these topologies over 2D NoCs. However, the design of such 3D NoC fabrics customized for specific applications has received very little attention to date. Even though a significant body of work exists for the synthesis of 2D bus-based [10]-[12] and 2D NoC [13]-[18] communication architectures, the techniques are not directly applicable for 3D NoC synthesis because of the peculiar challenges of 3D IC design. For instance, vertical TSVs have a larger pitch (5μm×5μm or more) that takes up space in the active layers and has at least an order of magnitude greater footprint than regular vias in the metal layers. These TSV interconnects are therefore expected to be limited in number. This heterogeneity and limited density of TSVs needs to be considered while synthesizing and optimizing 3D NoCs. Additionally, cores and routers can be placed on one of the many layers available, which dramatically increases design space complexity in 3D NoC based communication architectures.

In this paper, we propose a novel framework (MORPHEUS) for the application-specific synthesis of 3D NoCs, optimized for low power dissipation. MORPHEUS automates the process of thermal-aware core layout, 3D topology and route generation, and placement of network interfaces (NIs), routers, and serialized vertical TSVs. Experimental studies on several CMP applications indicate that our generated solutions notably reduce power dissipation (up to 2.3×) and average latency (up to 1.2×) over 2D NoCs. Compared to solutions generated by a previous work on application-specific 3D NoC synthesis, MORPHEUS generates solutions with lower power dissipation (up to 1.9×) and lower average latency (up to 1.6×).

II. RELATED WORK

Over the last several years, there has been a growing interest in 3D ICs as a means to alleviate the interconnect bottleneck problem currently facing 2D-ICs. A key challenge with 3D-ICs is its high thermal density due to multiple cores being stacked together, that can adversely impact chip performance and reliability. Therefore several researchers have proposed thermal-aware floorplanning techniques for 3D-ICs [19]-[21]. A few researchers have explored interconnect architectures for 3D-ICs such as 3D mesh and stacked mesh topologies [7] and a hybrid bus-NoC topology [8]. Some recent work has looked at decomposing cores (processors [24][25], NoC routers [26], and on-chip cache [27]) into the third dimension which allows reducing wire latency at the intra-core level, as opposed to the inter-core level. Circuit level models for TSVs were presented in [9].

The problem of custom interconnect architecture synthesis for 2D-ICs has received a lot of attention in the past, for point-to-point and bus based architectures [10]-[12] and NoC topologies [13]-[18]. Only recently have approaches been proposed for the synthesis of custom NoCs for 3D-ICs [28]-[32]. An ILP based synthesis technique for a 3D network with low-radix routers is proposed in [28]. However, the generated solution has many long links and the scalability of the approach is not clear. A methodology for application-specific topology synthesis and route computation for 3D-ICs that performs localized synthesis optimizations for every layer is proposed in [29], based on the author's previous work on 2D NoC synthesis. The 3D NoC synthesis approach in [30] extends [29] by additionally

1063-9667/12 $26.00 © 2012 IEEE
DOI 10.1109/VLSID.2012.82

IEEE computer society

determining switches placement and iteratively adding TSV links during synthesis. A technique for application-specific 3D NoC synthesis based on a low-level greedy rip-up and reroute procedure for determining routes is proposed in [31]. Every core is initially allocated a router that is later merged using a greedy heuristic. However, the method used for router and TSV allocation in layers is not clearly specified. A multi-commodity flow formulation is used in [32] to solve a similar problem, but with analytical (e.g., queuing) models of the NoC and high level abstractions of applications, which may reduce the overall accuracy of the solution.

Unlike existing approaches, our application-specific 3D NoC synthesis framework (MORPHEUS) enables a more comprehensive exploration of the design space by additionally integrating TSV serialization, NI placement, and thermal-aware core placement, together with layout-aware partitioning and allocation of routers and TSVs, to optimize power while meeting performance constraints.

III. 3D Integrated Circuits

Before presenting details of the MORPHEUS synthesis flow for 3D NoCs, we briefly discuss two relevant issues that are important to consider when designing application-specific NoCs for 3D ICs.

A major concern in the adoption of 3D ICs is the increased power densities that can result from placing a core on top of another core. As high peak temperatures due to increasing power densities can cause catastrophic IC failure and this is already a major ckoncern in 2D architectures, the move to 3D will accentuate the thermal problem. The problem of thermal-aware 3D core layout is thus tightly coupled with the problem of 3D NoC synthesis, and cannot be ignored as in some previous 3D NoC synthesis approaches [28]-[31]. Another critical restriction that severely constrains the design space in 3D-ICs and must be considered during synthesis is the limitation on the number of TSVs between layers. This limitation is due to the high area overhead of TSV pads that are required to interface with TSVs in each active layer. It is clear today that TSV fabrication technology lacks maturity and has low yield due to unsuccessful wafer alignment prior to and during the wafer bonding process. This is expected to remain a major challenge in the years to come. A simple and effective way to improve yield is to add hardware redundancy by using larger (e.g., double area) TSV pads. As misalignments are caused by the unavoidable shift of bonding pads with respect to their nominal position, using larger square pads can improve misalignment tolerance by an order of magnitude [6]. However, the large pads can complicate routing in active layers. This motivates the need for some form of serialization of TSVs to reduce TSV pad footprint in active layers. In [33], it was shown that TSV serialization can reduce TSV pad area overhead in active layers by as much as 70% at a negligible performance and power overhead. This motivates utilizing TSV serialization in our application specific 3D NoC synthesis framework.

IV. MORPHEUS Synthesis Framework

A. Inputs and Problem Description

We assume that we are given a set of computational and memory cores onto which application tasks have already been mapped, after a hardware-software partitioning phase. The cores are arranged in a core dependency graph (*CDG*) which is one of the inputs to our framework. The *CDG* is an annotated directed graph $G(V, E)$ where each node $v_i \in V$ corresponds to a core, directed edge $e_{ij} \in E$ is a communication flow from v_i to v_j, and edge weight $w(e_{ij})$ is given by:

$$w(e_{ij}) = \sigma * \mu(e_{ij}) + (1 - \sigma) * \rho(e_{ij})$$

where $\mu(e_{ij})$ and $\rho(e_{ij})$ are the latency and bandwidth constraints respectively for e_{ij}, and σ is a designer-specified parameter based on application characteristics. Each core v_i is a rectangular shaped hard macro that has a fixed width (W_i) and height (H_i) associated with it, and the layer to which it is mapped in the 3D IC is specified by $layer_i$.

The number of layers (ζ) in the 3D IC on which the application is to be implemented and the maximum die dimensions ($W_{die} \times H_{die}$) are also specified as inputs to our framework. As the TSV density is limited due to practical implementation concerns, we assume a maximum TSV density threshold between adjacent layers (δ) as a designer-specified input that depends on the chosen 3D-IC implementation technology. The core to layer assignment $V \rightarrow \zeta$ is assumed as an input from the designer, and is usually based on temperature or power density concerns. For instance, a designer may choose to interleave high power density computational cores with cooler layers comprising of low power density memory cores. NoC architecture parameters (e.g., operating voltage, clock frequency, link width) can either be specified by the designer, or varied in steps in a user-defined range, with our framework being invoked at each step. Finally, the technology library node (e.g., 65nm, 45nm) is an input that enables accurate delay and power estimation in the framework.

Problem Definition: *Given an application CDG with bandwidth and latency constraints, a core to layer ($V \rightarrow \zeta$) mapping, number of 3D IC layers (ζ), TSV density threshold (δ), maximum die dimensions ($W_{die} \times H_{die}$), NoC architecture parameters, and a target technology node, the goal of the MORPHEUS framework is to synthesize an application-specific 3D NoC topology with a layout for all cores, NIs, TSVs, routers, and links that satisfies all performance (bandwidth, latency) constraints in the application while minimizing NoC power.*

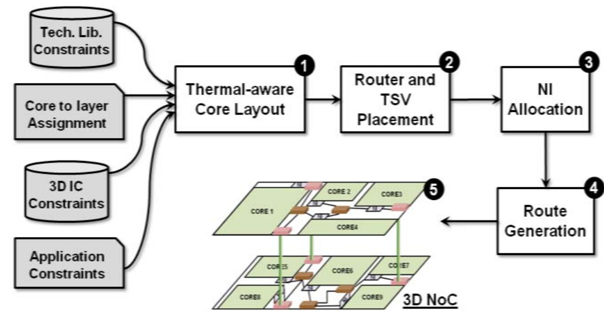


Figure 1. MORPHEUS 3D NoC synthesis framework

B. MORPHEUS Synthesis Flow Overview

Fig. 1 gives a high level overview of the MORPHEUS application-specific 3D NoC synthesis framework. In the first phase, a thermal-aware core layout using a 3D floorplanner is performed to place the cores assigned in each layer in a manner that minimizes peak temperature, wirelength, and chip area. The output of this step is a complete layout of the cores in every layer in the 3D-IC. In the next phase, the placement of NoC routers and TSV pads on the active layers is determined. A partitioning-based heuristic is used in this phase to explore the design space for a spectrum of router and TSV densities, while preserving the TSV density threshold (δ) constraint, in part by utilizing serialization. Next the NI placement for each core is determined to minimize critical path lengths. Finally, deadlock free routes are determined between cores. The output of the framework is a Pareto set of valid solutions (satisfying all performance constraints) that trade-off power with performance slack. The following subsections describe the various phases in the flow in more detail.

C. Thermal-aware Core Layout

Given a core to layer mapping, the MORPHEUS framework performs 3D floorplanning in the first phase to obtain a placement of cores in each layer. This core layout is often influenced by non-network-based interconnections (e.g., off-chip interface pin locations), and is an important step as it can have a significant impact on the quality of the synthesized NoC architecture. As stacked 3D-ICs can have significant reliability and performance issues due to higher

power density and peak temperatures than their 2D-IC counterparts, it becomes essential to perform core placement with thermal-awareness. Communication bandwidth and latency constraints must also be satisfied, by minimizing wirelength. In addition, overall chip area should also be minimized.

To solve this multi-objective core layout problem, we make use of a *genetic algorithm* (*GA*) based approach. A genetic algorithm is an iterative exploration algorithm that is based on a computational analogy with biological adaptive systems. The algorithm starts with an initial random pool of solutions (each represented by a chromosome) that are evaluated at each iteration (generation) by a fitness score obtained from an objective cost function. A new generation is created by first increasing the population by generating new individual solutions, and then selecting a constant number of solutions based on their fitness criteria. Genetic operators such as *crossover* and *mutation* are used to create an evolution of the solutions at every iteration. The best solution is finally selected after a specified number of generations. A detailed discussion of genetic algorithms can be found in [34]. In our GA formulation, we encode the location and orientation of each core in the chromosome using a sequence-pair representation [35]. The GA objective function is aimed at a hybrid optimization of thermal, communication, and area costs. The thermal cost is represented by the peak temperature (T_{peak}) of the design, and the area cost (A_{cost}) is given by the chip area. The communication cost (C_{cost}) is a hybrid formulation that combines bandwidth and latency constraints as follows:

$$C_{cost} = \sigma * \sum_{\forall e_{ij}} \frac{\rho(e_{ij}) * l(e_{ij})}{\max(\rho) * \max(l)} + (1-\sigma) * \sum_{\forall e_{ij}} \frac{\min(\mu) * l(e_{ij})}{\mu(e_{ij}) * \max(l)}$$

where $l(e_{ij})$ is the wirelength between cores v_i and v_j, $max(l)$ is the maximum wirelength between two cores with a communication flow, $max(\rho)$ and $min(\mu)$ are the maximum bandwidth, and minimum latency constraints among all flows, and σ is a weight parameter set by a designer based on application characteristics. Then the objective function for our GA formulation is given as:

$$F(T_{peak}, C_{cost}, A_{cost}) = \frac{a_1}{\log(T_{peak})} + \frac{a_2}{\log(C_{cost})} + \frac{a_3}{\log(A_{cost})}$$

where a_1, a_2, and a_3 are weighting parameters used to guide the optimization. The logarithmic values provide a more descriptive range and characterization for input variations. For any potential layout solution, C_{cost} can be calculated based on HPWL (half perimeter wire length) distances and μ/ρ constraints for each communication flow. The chip area A_{cost} can also be determined after a layout of every core is obtained. To obtain T_{peak} estimates, we make use of a 3D adaptation of Hotspot [36] which is a well-known tool for temperature estimation in 2D-ICs. Based on the average power dissipation of each core (known for every core based on technology library used and application characteristics), physical dimensions, and location of a core in the 3D-IC, Hotspot returns temperature estimates for the chip.

To accommodate placement of smaller components such as routers, NIs, and TSV pads in subsequent phases of the synthesis flow, we increase core dimensions by a small margin φ before core layout. After layout, a compaction step is performed to reduce core dimensions and create whitespace for inserting routers, NIs, and TSV pads later. φ is set to 5% in our framework, although higher values can also be used to ease wiring congestion. After the layout phase, each core v_i has a placement $P(v_i) = (x_i, y_i, z_i)$ with (x_i, y_i) referring to the bottom left coordinates of the core in a layer and z_i referring to the layer to which the core is mapped. Together with the core width and height, a unique placement is obtained for every core in the 3D-IC. Once such a layout is available, MORPHEUS can more accurately determine inter-core wiring delays and power dissipation.

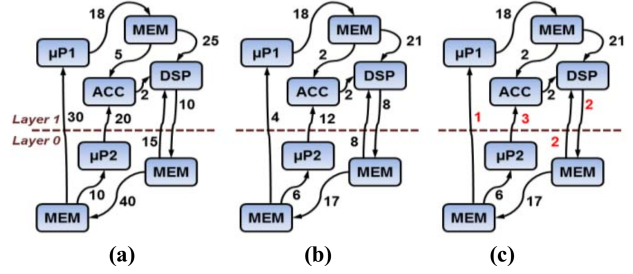


Figure 2. (a) CDG, (b) ECDG, (c) ESCDG

D. Router and TSV Pad Placement

In the next phase, we determine the number and location of routers and TSVs in the synthesized solutions. Our procedure for router synthesis extends the technique proposed in [30] for switch synthesis in 3D NoCs by adding support for serialization and using layout awareness to obtain lower average power and latency implementations. We make use of a min-cut partitioning approach to determine the optimal number of routers in the implementation.

Algorithm 1 describes the steps in this phase for router and TSV allocation and placement. In the initial steps we transform the core dependency graph *CDG(V,E)* into an enhanced core dependency graph *ECDG(V,E')* by updating the edge weights with a relative inter-core distance term to enable a more accurate estimation of flow criticality (Steps 2-3). Then we transform *ECDG(V,E')* into an enhanced scaled core dependency graph *ESCDG(V,E")* by scaling the inter-layer links that need to traverse TSVs by a factor of $\omega*|layer_i - layer_j|$ (Step 4). This scaling is done so that the subsequent min-cut partitioning step does not lead to cores in different layers that do not have any flows between layers being assigned to the same partition, which can lead to an excessive number of inter-layer TSVs. Fig. 2 shows an example of how a *CDG* is transformed into an *ECDG* and then into an *ESCDG* (with $\omega=4$). Note how the criticality of the links changes when going from a *CDG* to an *ECDG* due to the addition of more accurate inter-core wirelength information available after the core layout phase from Section IV.C.

Algorithm 1: Router and TSV Pad Allocation and Placement

```
1:  // Generate ESCDG(V,E”) from CDG(V,E)
2:  for each e_ij ϵ E do
3:     w(e_ij) = w(e_ij)*l(e_ij)/max(l)
4:     if layer_i ≠ layer_j then w(e_ij) = w(e_ij)/(ω*|layer_i - layer_j|)
5:  end for
6:  // Create min-cut partitions
7:  for t = 1 to |V| do
8:     using Kerninghan-Lin min-cut heuristic, create t partitions in ESCDG
       to create solution instance s_t, with δ’ TSVs
9:     while δ’ > δ do
10:        select e_ij with layer_i ≠ layer_j and min w(e_ij)
11:        serialize TSV(e_ij) by degree k
12:        w(e_ij) = w(e_ij)*k
13:        δ’ = δ’- (k-1)/k*link_width
14:    end while
15:    if δ’ ≤ δ then FP_check_store()
16: end for
```

Next, a wide spectrum of router counts from 1 to the number of cores in the application ($|V|$) is swept (Steps 7-16). The routers are assumed to be wormhole switched with a predictive-forwarding enabled four-stage pipeline [22], and a parameterizable number of ports. In each iteration, t min-cut partitions are created using the Kerninghan-Lin algorithm (Step 8). The cores in each partition share the same router. If there exist multiple inter-partition edges between two partitions I and J, these are merged into a single edge e_{IJ} with weight $w(e_{IJ}) = \sum w(e_{ij})$, $\forall$ e_{ij}, where i and j are cores such that $i \in I$ and $j \in J$. The idea behind the partitioning step is to ensure a minimal

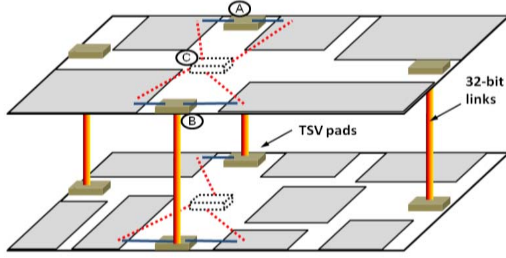


Figure 3. TSV serialization example

number of hops for communication flows that are more critical. Cores within a partition can reach each other within a single router hop.

If the total number of TSVs (δ') in the generated solution instance (s_t) is greater than the TSV density threshold (δ), we explore using serialization of TSVs to reduce the number of TSVs. Fig. 3 shows how TSV serialization can be beneficial during the synthesis process. Suppose the maximum number of TSVs is limited to 3*32=96 per layer (i.e., 3 links that are 32 bits wide each). Then the scenario shown in Fig. 3 is an invalid solution as it has 4*32=128 TSVs between layers. To get a valid solution, the number of TSVs must be reduced to 96. One way to do this is by merging and replacing the closest TSVs (*A* and *B*) with a single TSV (*C*). However, doing so can end up increasing wiring costs as shown in the figure, where solid links to TSV locations *A* and *B* from cores are now replaced by longer (dotted) links to TSV location *C* in both layers. This can dramatically increase the number of repeaters, pipeline buffers, and consequently increase power dissipation and delay. To avoid such a scenario, if serialization of degree 2 is employed at locations *A* and *B*, the TSV links are reduced to half at each location. The inter layer TSV threshold is thus satisfied without requiring TSVs at locations A and B to be replaced by a TSV at a location C. In this manner, serialization can prevent unnecessary routing congestion and reduce power dissipation in the 3D network.

The serialization process in our algorithm starts by selecting the TSV with the least communication cost (Step 10) and serializes it by a degree *k* (Step 11). We set *k*=2 in our approach, but higher degrees can also be considered. Next the communication cost of the edge with the serialized link is increased by the factor *k* (Step 12), and the number of TSV links in s_i is reduced by *(k-1)/k*link_width* which is the number of TSVs reduced due to serialization (Step 13). The serialization process continues (Steps 9-14) till the TSV threshold constraint is no longer violated. Finally, if $\delta' \leq \delta$ either after serialization or as generated in Step 8, we invoke our GA floorplanner with *FP_check_store()* to perform router and TSV pad placement (Step 15). The floorplanner keeps the relative locations of cores fixed and performs a placement of the routers and TSV pads (with area overhead added for any serialization circuitry used) while optimizing the GA objective function as described in the previous section. The output layout is checked to ensure no latency constraint is violated by calculating the number of cycles to traverse pipelined wires and routers for each flow. If latency violations are detected, the floorplanning phase is repeated after increasing weights on violated edges. If a valid solution is obtained after the phase, it is stored, otherwise it is discarded. The final output of this phase is a set of *m* valid solutions $S = \{s_1, s_2, ..., s_m\}$ that meet all latency and bandwidth constraints of the application.

E. NI Allocation

The next phase is to determine the location of the network interface (NI) component for each core. The NI is the bridge between the core and the network, and is generally located at the core boundary. Depending on the core internals, pin layout, and core orientation, there can be some flexibility during NI allocation. For every core v_i, we define a set $P_i = \{p_1, p_2, ..., p_n\}$ that provides valid locations for NI location at the core periphery. For instance, Fig. 4 shows three possible NI locations for *core9*. The location of a core's NI can have a significant impact on inter-core wire length and routing, and consequently communication delay and power dissipation. In Fig.4, suppose *core1* has a communication flow to *core3*. If NI location *A* is chosen for *core1*, the wirelength and cost will be much higher than if location *B* had been selected. The pitfalls of inefficient NI placement are exacerbated for 3D ICs. For instance, for a communication flow between *core4* and *core6*, and TSV location as shown in Fig. 4, if NI locations *D* and *E* are fixed in the two layers because of their proximity (i.e., small Manhattan distance), it may lead to an excessive wirelength than if locations *F* and *G* were chosen that have a higher Manhattan distance of separation. The anomaly exists due to the limited number and location of TSVs in 3D-ICs. This is the motivation for considering NI placement after the TSV and router allocation phase in the MORPHEUS framework.

To determine NI locations for all cores, we make use of a greedy shortest path heuristic for each solution in the valid solution set *S*. For a given valid solution, first a topological sort in non-ascending order is performed for all the cores based on aggregate incident communication costs $\sum w(e_{ij})$ on each core. This allows us to ascertain the relative criticality of the cores. Next, we select the core v_i at the top of the sorted list, select an NI location from P_i, calculate minimum cost paths to each core after floorplanning, and sum up the costs to obtain a single fitness cost for the NI location. The process is repeated for all possible NI locations in P_i. The NI location with the lowest fitness cost is selected and fixed for core v_i. The corresponding NIs at the destination cores that are part of the minimum cost paths from the selected source NI location are also fixed. The process is repeated by selecting the next core in the sorted list with an unselected NI location, or if its NI location has been previously fixed but at least one of its destination NIs remains unselected. The output after this process is a layout with fixed NI locations for every core.

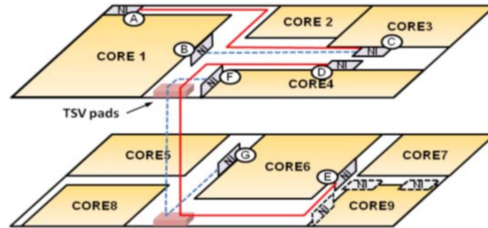


Figure 4. Network interface placement issues

F. Deadlock Free Route Generation

Finally, after the core, router, TSV, and NI locations have been fixed in the 3D-IC for each valid solution, it is important to check for possible deadlock conditions. To analyze deadlocks, we create a flow dependency graph and check for possible cycles. To avoid possible deadlock conditions, we make use of the rich body of literature in the area of deadlock avoidance in interconnection networks [16][18] and make use of escape virtual channels in the routers as a means to break deadlocks, wherever our analysis indicates a need.

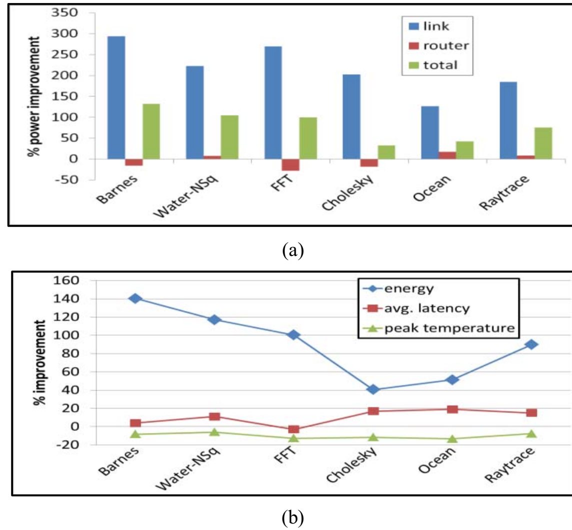
V. EXPERIMENTS

To validate our proposed MORPHEUS synthesis framework, we used it to synthesize different CMP applications on a 3D IC. Six applications from the well-known SPLASH-2 benchmark suite (*Barnes, Water-NSq, FFT, Cholesky, Ocean, Raytrace*) [37] were selected, then parallelized, and mapped onto multiple irregular sized cores. These CMP applications were used as inputs to our synthesis approach. Table I summarizes the details of the CMP applications, such as number of cores (including processors and on-chip memories), and the number of layers on which the cores are to be mapped in the 3D IC implementation.

TABLE I. CMP Applications

CMP applications	Description	Cores	Layers
Barnes	Galaxy evolution	32	2
Water-NSq	Forces/potentials of H_2O molecules	38	2
FFT	FFT kernel	44	2
Cholesky	Cholesky factorization kernel	76	4
Ocean	Ocean movements	88	4
Raytrace	3-D ray tracing	112	4

The maximum TSV threshold (*δ*) between adjacent layers was fixed at 1024 (i.e., 32 32-bit links) and the maximum die area constraint was kept at 16mm×16mm. A pad size of 5μm×5μm was assumed for each TSV. We made use of an in-house SystemC-based 3D NoC simulator to simulate and generate performance and power results. Power estimation modules were integrated into the simulator from a modified version of Orion 2.0 [38] and CACTII [39]. Long links were pipelined to maintain high operating frequency operation. The latency and power impact of pipelining was considered in our final results. We targeted our results for the 45nm technology library, and the NoC was clocked at 1 GHz. For the GA floorplanner, we set the population size to 100, crossover and mutation probabilities to 0.9 and 0.01 respectively, and maximum generation to 100,000 based on our experience and guidelines from extensive simulations in [23].

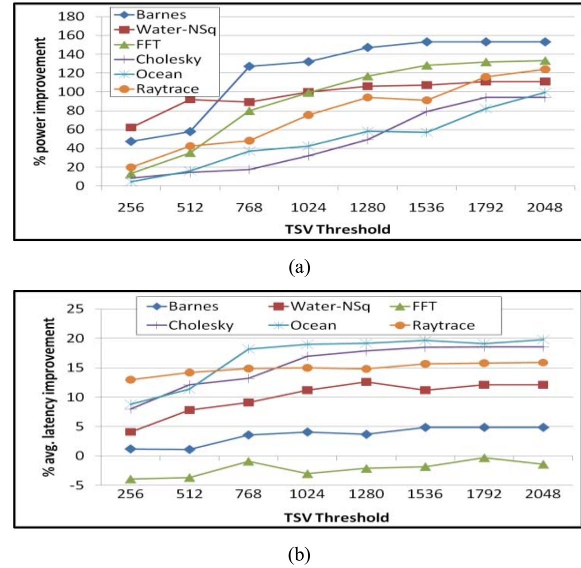


(a)

(b)

Figure 5. Application-specific 2D vs. 3D NoC comparison (a) average power dissipation, (b) energy and average latency

Our first experiment compares the results generated by the MORPHEUS framework for an application-specific 3D NoC, with an application-specific 2D NoC. The 2D NoC was synthesized using a subset of techniques in MORPHEUS that are relevant to 2D NoCs, to ensure a fair comparison. Fig. 5(a) shows the percentage improvement in power dissipation for the most power efficient application-specific 3D NoC solution, over the most power efficient application-specific 2D NoC solution, for each CMP application. In general, 3D-ICs replace long interconnects with much shorter TSVs, resulting in a significant reduction in wiring, and consequently link power dissipation. The routers however become more complicated, due to additional ports for vertical transfers and the overhead of serialization/de-serialization circuitry. Overall there is as much as a 2.3× reduction in average power dissipation for the generated application-specific 3D NoCs compared to their 2D counterparts. While a reduction in average power can improve reliability and reduce cooling costs (particularly relevant for 3D ICs that have high power densities due to active layer stacking), energy consumption is also an important metric relevant especially for battery-driven devices. Fig. 5(b) shows the percentage improvement in total energy consumption and average latency for the synthesized application-specific 3D NoC solutions over the synthesized application-specific 2D NoC solutions. The average latency goes down slightly as high latency long interconnects are replaced by much shorter and low latency TSVs. The overall energy consumption also decreases by as much as 2.4× for the application-specific 3D NoC solutions, compared to the application-specific 2D NoC solutions. Finally, peak temperature estimates were found to be higher by 5.9%-13.3% for the 3D-IC implementations compared to the 2D-IC implementations, underscoring the need for additional thermal-aware design techniques such as throttling, adaptive voltage/frequency scaling in 3D-ICs.



(a)

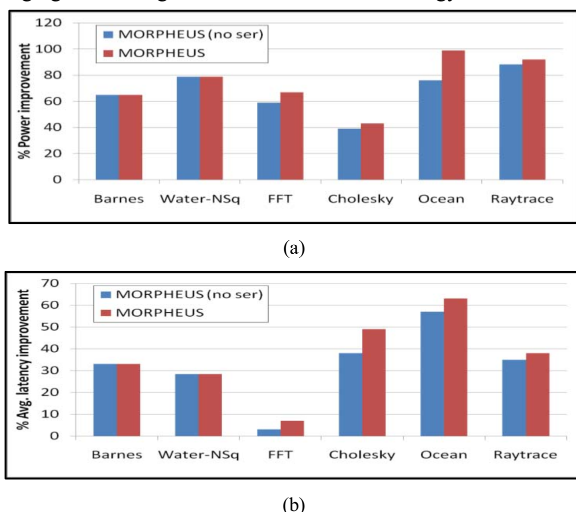
(b)

Figure 6. Impact of varying TSV threshold in MORPHEUS (a) power dissipation, (b) average latency

The next set of experiments show the impact of changing the TSV threshold (*δ*) on the power and average latency of the application-specific 3D NoC solutions generated by MORPHEUS. The solutions considered are the most power efficient ones from the set of synthesized solutions for each TSV threshold value. Fig. 6(a) shows the percentage power improvement for the synthesized application-specific 3D NoC over the synthesized 2D application-specific NoC baseline. It can be seen that for applications with fewer cores, the power dissipation reduces rapidly initially as the allowed number of TSVs is increased, but the improvements begin to saturate because of relatively few inter-layer communication flows that can take advantage of the increased number of allowed TSVs. For applications with larger numbers of cores and additional layers, the inter-layer communication demand is higher in general, leading to greater reduction in link power dissipation as the allowed number of TSVs is increased. However, the increased power dissipation in routers negates some of the benefits of lower link power dissipation. Fig 6(b) shows the percentage average latency improvement for the synthesized application-specific 3D NoC over the synthesized application-specific 2D NoC baseline. For almost all applications, the latency improvement saturates after a point. Except for *FFT*, the average latency is reduced for all the applications by as much as 19%. For FFT, the increase in complexity and greater traffic loading in the 3D-enabled routers translates into lesser opportunities for predictive forwarding in the router, and thus the latency increases slightly.

The final set of experiments compare the solutions generated by MORPHEUS with and without serialization, and with the solutions obtained from a previously proposed framework for synthesizing application-specific 3D NoCs [30]. Even though [30] does not address

core layout, to ensure a fair comparison we use the same core layout for [30] as generated by MORPHEUS for our solutions. In this manner, we specifically compare the algorithmic effectiveness of the two approaches independent of the initial core layout step. For the comparison, we again select the most power efficient solution generated by the approach outlined in [30] and by MORPHEUS for the given applications. Fig 7(a) shows the percentage improvement in power dissipation, while Fig 7(b) shows the improvement in average latency for the solutions generated by MORPHEUS compared to the solutions generated by [30]. While applications with fewer cores (*Barnes*, *Water-NSq*) do not particularly benefit from serialization, other larger applications can be seen to clearly gain from using the TSV serialization technique. The improvements over [30] come from better NI allocation, serialization, and better allocation for routers and TSVs in the 3D IC. Overall our results indicate an up to 98% improvement in power dissipation and up to 62% improvement in average latency for the MORPHEUS framework, compared to [30]. These results highlight the effectiveness of the automated MORPHEUS application-specific 3D NoC synthesis framework for emerging CMP designs that utilize 3D-IC technology.



(a)

(b)

Figure 7. Comparing MORPHEUS vs. MORPHEUS without serialization vs. [30] (a) power dissipation, (b) average latency

VI. CONCLUSION

On-chip communication architectures are significant factors in determining performance and power dissipation for emerging CMP applications. However, designing an on-chip interconnect fabric especially for 3D ICs is a major challenge due its much larger design space compared to 2D ICs. In this paper, we proposed a framework (MORPHEUS) for the automated synthesis of application-specific 3D NoCs. MORPHEUS combines techniques for thermal-aware core layout, TSV serialization, allocation of routers, NIs, and TSVs, and deadlock-free path generation with the goal of generating low power solutions that satisfy all application performance constraints. The effectiveness of this framework can be seen from the notable power and latency improvements obtained over application-specific 2D NoCs, as well as solutions generated by heuristics from previously proposed work in the area of application-specific 3D NoC synthesis. Our future work will explore incorporating the reliability metric during 3D NoC synthesis within MORPHEUS.

REFERENCES

[1] S. Pasricha, and N. Dutt. “On-Chip Communication Architectures”, Morgan Kauffman, ISBN 978-0-12-373892-9, Apr 2008

[2] International Technology Roadmap for Semiconductors (ITRS), 2007.

[3] A. W. Topol et al., “Three-dimensional integrated circuits,” IBM J. Res. & Dev. Vol. 50 No. 4/5 Jul/Sep 2006.

[4] K. Bernstein, et al., “Interconnects in the Third Dimension: Design Challenges for 3D ICs,” Proc. DAC 2007, pp.562-567.

[5] R. S. Patti, “Three-Dimensional Integrated Circuits and the Future of System-on-Chip Designs”, Proc IEEE, Vol 94, No. 6, Jun 2006.

[6] V. F. Pavlidis, E. G. Friedman, “Three-dimensional Integrated Circuit Design”, Morgan Kaufmann, Sep 2008.

[7] B. Feero, P.P. Pande, “Performance Evaluation for Three-Dimensional Networks-On-Chip”, Proc. ISVLSI 2007.

[8] F. Li et al., “Design and Management of 3D Chip Multiprocessors Using Network-in-Memory”, Proc. ISCA 2006, pp. 130-141.

[9] I. Loi et al., “Supporting vertical links for 3D networks on chip: toward an automated design and analysis flow”, Proc. NanoNet 2007.

[10] S. Pasricha, et al., "Floorplan-aware Automated Synthesis of Bus-based Communication Architectures", IEEE/ACM DAC 2005.

[11] J. Hu et al., “System-Level Point-to-Point Communication Synthesis Using Floorplanning Information”, Proc. ASPDAC 2002.

[12] S. Pasricha, N. Dutt, M. Ben-Romdhane, “Constraint-Driven Bus Matrix Synthesis for MPSoC”, Proc. ASPDAC 2006.

[13] S. Murali, et al. “Synthesis of Predictable Networks-on-Chip-Based Interconnect Architectures for Chip Multiprocessors”, IEEE TVLSI 15:8, 2007

[14] A.Pinto et al., “Efficient Synthesis of Networks on Chip”, ICCD 2003, pp. 146-150, Oct 2003.

[15] K. Srinivasan et al., “An Automated Technique for Topology and Route Generation of Application Specific On-Chip Interconnection Networks”, Proc. ICCAD 2005.

[16] S. Kwon, S. Pasricha, "POSEIDON: A Framework for Application-Specific Network-on-Chip Synthesis for Heterogeneous Chip Multiprocessors", IEEE ISQED 2011

[17] J. Xu et al., “A design methodology for application-specific networks-on-chip”, ACM TECS, 2006.

[18] S. Murali et al., “Designing Application-Specific Networks on Chips with Floorplan Information”, pp. 355-362, ICCAD 2006.

[19] Z. Li, et al., “Efficient thermal-oriented 3D floorplanning and thermal via planning for two-stacked-die integration”, ACM TODAES 11:2, Apr 2006.

[20] C. Addo-Quaye, “Thermal-aware mapping and placement for 3-D NoC designs,” Proc. IEEE Int. Syst.-on-Chip Conf., 2005, pp. 25–28.

[21] E. Wong, et al. “3D Floorplanning with Thermal Vias” Proc. DATE 2006.

[22] H. Matsutani, “Prediction Router: Yet Another Low Latency On-Chip Router Architecture”, Proc. HPCA 2009.

[23] J.Schaffer et al., “A study of control parameters affecting online performance of genetic algorithms for function optimization,” Proc. of International Conference on Genetic Algorithms, pp.51-60, 1989.

[24] K. Puttaswamy, G.H.Loh, “Thermal Herding: Microarchitecture Techniques for Controlling Hotspots in High-Performance 3D-Integrated Processors”, Proc. HPCA 2007, pp. 193-204.

[25] Y. Liu, et al., "Fine Grain 3D Integration for Microarchitecture Design Through Cube Packing Exploration", Proc. ICCD, 2007.

[26] D. Park et al. “MIRA: A Multi-layered On-Chip Interconnect Router Architecture”, Proc. ISCA 2008, pp. 251-261.

[27] K. Puttaswamy, G. H. Loh, "Implementing caches in a 3D technology for high performance processors" Proc. ICCD 2005.

[28] Y. Xu, et al., “A Low-Radix and Low-Diameter 3D Interconnection Network Design”, Proc. HPCA 2009.

[29] S. Murali, C. Seiculescu, L. Benini, G. De Micheli, “Synthesis of Networks on Chips for 3D Systems on Chips”, Proc. ASPDAC 2009.

[30] C. Seiculescu, et al., “SunFloor 3D: A Tool for Networks on Chip Topology Synthesis for 3D Systems on Chips”, Proc. DATE 2009.

[31] S. Yan, B. Lin, “Design of Application-Specific 3D Networks-on-Chip Architectures”, Proc. ICCD, 2008.

[32] P. Zhou, P.-H. Yuh, S. Sapatnekar, “Application-Specific 3D Network-on-Chip Design Using Simulated Allocation”, Proc. ASPDAC 2010

[33] S. Pasricha, “Exploring Serial Vertical Interconnects for 3D ICs”, Proc. DAC, 2009.

[34] A. Eiben, et al, “Introduction to Evolutionary Computing”, Springer 2003.

[35] S. Nakaya et al., “An Adaptive Genetic Algorithm For Vlsi Floorplanning Based On Sequence-Pair”, Proc. ISCAS 2000.

[36] W. Huang, et al. "Differentiating the Roles of IR Measurement and Simulation for Power and Temperature-Aware Design." Proc. ISPASS 2009.

[37] S.C. Woo et al.“The SPLASH-2 programs: Characterization and methodological considerations”, Proc. ISCA, 1995.

[38] A. Kahng, et al., “ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration”, Proc. DATE, 2009.

[39] CACTI 6.5, http://www.hpl.hp.com/research/cacti/