VISION: A Framework for Voltage Island Aware Synthesis of Interconnection Networks-on-Chip

Nishit Kapadia, Sudeep Pasricha
Department of Electrical and Computer Engineering
Colorado State University, Fort Collins, CO, U.S.A.
{nkapadia, sudeep}@colostate.edu

Abstract

High power dissipation has today become one of the major challenges in chip multiprocessor (CMP) design. Designers in recent years have proposed several techniques to alleviate the power challenge, one of which is the use of voltage islands (VIs) that can help reduce both switching and standby components of power. The use of VIs allows groups of cores to be powered by the same supply source and permits operating different portions of the chip at different voltage levels in order to optimize the overall chip power consumption. However, the problems of *VI* creation, core to *VI* mapping, and VI-aware network on chip (NoC) design to satisfy application performance constraints are non-trivial and will only get harder as the number of cores in CMPs increase into the hundreds. In this paper, we propose a novel framework (VISION) for automating the synthesis of regular networks on chip (NoC) with VIs, to satisfy application performance while minimizing chip power dissipation. Our proposed framework uses a set of novel algorithms and heuristics to generate solutions that reduce network traffic by up to 60% and power dissipation by up to 11%, compared to the best known prior work that also solves the same problem.

Categories and Subject Descriptors

B.7.1 [Integrated Circuits]: Types and Design Styles—VLSI

General Terms

Algorithms, Design, Performance

Keywords

Networks-on-chip, Voltage Islands, Synthesis Algorithms, Chip Multiprocessor, Core-to-Tile Mapping

1. Introduction

Emerging chip multiprocessors (CMPs) today are severely constrained because of their high power dissipation due to high levels of core integration and nanoscale CMOS process technology limitations. High power dissipation on a chip not only reduces overall performance, but also negatively impacts reliability and cooling costs. As the number of cores integrated in CMPs continues to increase into the hundreds [1][2], the complexity of networks on chip (NoC) [3] required to interconnect communicating cores on a chip has also increased. NoCs have been shown to dissipate significant power (e.g., ~30% of chip power in Intel's 80-core teraflop [1] and ~40% of chip power in MIT RAW [2] chips). As a result, reducing both computation and communication power has become a high priority for chip designers.

Among the several circuit and (micro)-architectural techniques proposed by designers in recent years to reduce power dissipation, dynamic voltage and frequency scaling (DVFS) is widely used to reduce dynamic power [4], while clock/power gating techniques [5] are commonly used to reduce leakage power in cores. However, implementing these techniques at a per-core granularity requires separate V_{DD} and ground lines, as well as a prohibitively large number of VLCs (voltage level converters) and MCFIFO (multiple clock first-in-first-out) queue based frequency level converters [6][7], especially as the number of cores on a die increase. The use of voltage islands (*VIs*) limits the overhead of implementing these power management schemes by combining cores into islands that use the same V_{DD} and ground lines, and thus also minimizing the number of VLCs and MCFIFOs required [8]. Different islands can then operate at different voltage levels and perform runtime optimizations independent of each other to more aggressively reduce chip power.

In the presence of VIs, not only can cores run on optimal voltages with minimal overhead to reduce chip power dissipation, but the on-chip interconnection network can also exploit *VIs* to reduce communication power. However, *VIs* complicate the problem of NoC synthesis, requiring designers to revisit the problems of *VI* partitioning, core to die mapping, and routing path allocation so that both computation and communication power can be minimized.

In this paper we address the problem of synthesizing NoCs for regular CMPs with *VIs* to reduce overall power dissipation. We focus on CMPs with homogenous and regular sized cores because of the prevalence of regular topologies in almost all recently released commercial CMPs [1][2]. Our proposed framework for VI-aware Synthesis of IntercOnnection Networks on chip (VISION) combines novel algorithms and heuristics to perform *VI* partitioning, core to die mapping, and routing path allocation to minimize power dissipation while satisfying application bandwidth requirements. Experimental studies presented in Section 5 indicate that our proposed framework outperforms the best known prior work [9] that focuses on the same problem of *VI*-aware regular NoC synthesis. In particular, our framework generates solutions that have lower network traffic by up to 60%, and lower power dissipation by up to 11% compared to solutions generated by using the approach from [9].

2. Related Work

Many researchers [10]-[14] have proposed custom topologies for NoC architectures that improve overall performance at the cost of sacrificing the regularity of mesh-based structures. Although these custom architectures are expected to achieve better latency and area utilization, their design process is more complex and faces several challenges, such as greater crosstalk and uncertainty in link delays due to irregular interconnect structures. Thus, a conservative enough custom design may actually offset the advantages of better performance [15]; especially for medium to large sized (in terms of total number of cores) NoC architectures.

Lackey [8] gives an overview of the preliminary considerations for power and performance for NoC designs with *VIs*. The problem of NoC synthesis on regular structures with multiple supply VIs has since been addressed in several works [9][16]-[21]. By performing voltage-partitioning before core mapping, [9] demonstrated improvements over prior work (e.g., [18]) to achieve better energy consumption and less power overhead by reducing the total number of MCFIFOs and VLCs needed for inter-island communication.

The problem of mapping cores on to regular NoCs has been handled differently by the works above. Heuristics implementing incremental mapping on NoCs with *VIs* that have been proposed to date [9][20] map the cores in order of their communication bandwidths. As the order of mapping is fixed, every new mapping decision is based on optimizing some communication metric with just the previously placed cores, thereby restricting the search space to a possible local minimum and failing to capture a holistic view for optimizing the communication over the entire network.

In this paper, in addition to proposing novel techniques for voltage partitioning and routing path allocation, we present a mapping technique that uses incremental swaps with a distributed decision making process, as opposed to a central one used in prior work.

3. Problem Formulation

We are given the following inputs to our problem:

(i) A core graph *G (V, E)*; with the set of *N* vertices $\{V_1, V_2, V_3, \ldots, V_N\}$ representing the homogeneous cores on which tasks have already been mapped and the set of *M* edges $\{e_1, e_2, e_3, \ldots, e_M\}$ that represent communication dependencies between the cores,

(ii) A regular mesh-based NoC with *T* tiles such that $T \geq N$, and $T = (d^2)$, where *d* is the dimension of the mesh, and each tile consists of a compute core and a NoC router,

(iii) A set of minimum operating voltage levels required for meeting task performance requirements on each of the *N* cores $\{min\_v(V_1), min\_v(V_2), \ldots., min\_v(V_N)\}$,

(iv) A set of *n* candidate voltage levels $\{v_1, v_2, \ldots, v_n\}$ and the corresponding candidate frequency values $\{f_1 f_2, \ldots, f_n\}$ that the cores can take,

(v) The maximum allowable number of voltage islands on the die, Ω; where $\Omega \leq n$.

Objective: Given the above inputs, our goal is to obtain a core to die mapping and synthesize a regular 2D mesh NoC architecture for a specific application, such that all application performance requirements are satisfied, while minimizing the total power dissipation in the compute cores and the communication resources (routers, links, VLCs, MCFIFOs).

Definition 1: Voltage Island Integrity of any island is said to be respected when every core within it has at least one neighbor (out of the maximum of four neighbors possible in a mesh) of the same voltage level as itself. Mathematically:

$$\forall v(t_{xy}) = v_i: \exists\{v(t_{x-1,y}) = v_i \mid v(t_{x,y-1}) = v_i \mid v(t_{x+1,y}) = v_i \mid v(t_{x,y+1}) = v_i\}$$

where t_{xy} represents the tile at the respective co-ordinates of the mesh, with integral values in the range *1* to *d*.

Definition 2: Pre-Routing Traffic is an early estimation of the total traffic assuming that all routing paths are minimal. It is equal to the sum of products of Manhattan distances and bandwidths of all individual communication flows. Mathematically, this can be expressed as: $\sum_j MD_j * BW_j$; where *j* is a uni-directional communication flow between any two cores.

Definition 3: Link-tension is defined as the product of the communication bandwidth and post-mapping Manhattan distance for any edge on the core graph G (V, E). Thus, for two cores adjacent to each other (i.e., Manhattan distance = 1), the tension on the link connecting them is equal to the bandwidth of communication between them.

Definition 4: Total Traffic is the overall bandwidth a routed-network is required to suport. It is the sum total of bandwidths of all communication flows over their respective actual (minimal or non-minimal) path-lengths, expressed as: $\sum_j Path\text{-}length_j * BW_j$; where *j* is a uni-directional communication flow between any two cores.

Not unlike previous works [9][18][19], our proposed NoC synthesis framework is subdivided into three major steps: voltage partitioning, core-to-tile mapping, and routing path allocation. Where our framework differs from previous works is in the algorithms used and more rigorous implementation assumptions that we consider. The problems addressed in the three major steps by our framework are summarized below:

A. Communication-power aware voltage partitioning

The objective in this first step is to assign voltages to cores in the core graph to minimize the computation power and the communication power at the same time, such that the constraints on the minimum operating voltage levels of all *N* cores $\{min\_v(V_1), min\_v(V_2), \ldots., min\_v(V_N)\}$ and the maximum number of *VI*s allowed (*I*), are satisfied.

B. Core to tile mapping

The objective in this second step is to perform a one to one mapping of the voltage assigned cores onto NoC tiles such that the integrity of each *VI* is respected (*Definition 1*), hikes in compute power of cores are avoided, and the total estimated traffic (pre-routing traffic in *Definition 2*) is minimized, to optimize overall communication power dissipation.

C. Routing Path Allocation:

The objective in this third and final step is to allocate routing paths for all communication flows in the NoC, such that communication power overhead associated with MCFIFOs and VLCs needed for inter-island communication is minimized.

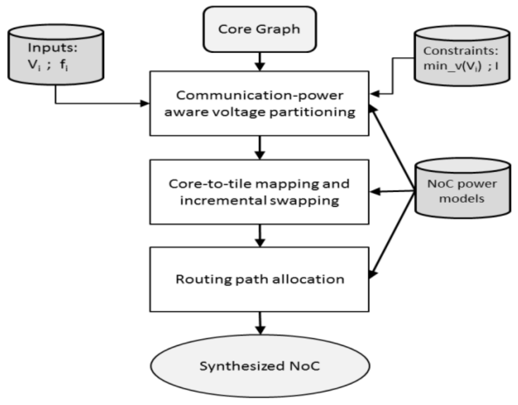

Inputs: V_i ; f_i
Core Graph
Constraints: min_v(V_i) ; I
Communication-power aware voltage partitioning
Core-to-tile mapping and incremental swapping
NoC power models
Routing path allocation
Synthesized NoC

Figure 1: NoC Synthesis design flow

4. NoC Synthesis Flow

In this section, we present our framework for the synthesis of NoCs with *VI*s. Figure 1 shows the high level flow of our synthesis framework that improves upon the state of the art *VI*-aware NoC synthesis frameworks, such as that proposed in [9]. The partitioning stage supports trade-offs between computation power and the estimated communication power, while assigning voltages to cores and partitioning them into islands. The mapping stage attempts to meet physical proximity constraints for cores in islands and performs incremental swaps to reduce pre-routing traffic. Finally, the routing path allocation stage integrates link insertion and routing path selection for communication flows, to minimize traffic and reduce communication latency.

In the following subsections (Sections 4.1-4.3), we present a detailed explanation of the algorithms used in the three major stages of our synthesis framework.

4.1 Communication-power aware voltage partitioning

Voltage partitioning attempts to assign core voltages, in order to meet constraints imposed by application performance and the Ω levels of allowable voltages. The voltage partitioning approach in [9] performs an exhaustive search on all *n* voltage level candidates using a maximum of *Ω* levels of allowable voltages (for *Ω VIs*). Thus, ${}^{n}C_{\Omega}$ combinations of different voltages are generated. Then, each core is assigned the least voltage level out of the available voltages for each of the ${}^{n}C_{\Omega}$ combinations, to ensure that application performance constraints are satisfied. Out of the ${}^{n}C_{\Omega}$ combinations of voltage assignments, the least expensive assignment in terms of power is then finally chosen. The shortcoming of this approach is that the voltage partitioning only optimizes the computation power, ignoring the effects of the communication power on the resultant total power. We propose a new *repartitioning* stage that can be appended to the previous approach to overcome the above mentioned drawback.

After the voltage partitioning of [9] is completed based on optimal computation power, in our repartitioning stage we allow certain cores to be moved to a neighboring voltage island with the next higher allowable voltage level, in order to reduce the communication power associated with the core. Such a move is meant to reduce communication traffic overhead by merging the core with an island it communicates heavily with; and is referred to as an 'allowable move' as long as the increase in resultant power is within a certain threshold ψ. Note that the allowable move is restricted to only the voltage island with an immediately higher voltage level because in low to medium sized NoCs (where the maximum distance between any two cores is restricted by the dimension of the mesh), the computation power dominates the communication power. Therefore, the reduction in communication power by performing a move that would increase the voltage of any core to a much higher level would seldom justify the penalty incurred on the computation power.

With repartitioning, cores that heavily communicate with each other end up residing in the same island and thus are likelier to be mapped in each other's vicinity. The probable reduction in the communication power as well as the total traffic is primarily due to lesser inter-island communication that leads to: *(i)* shorter average communication path lengths, and *(ii)* a reduction in the number of MCFIFOs and VLCs, with lesser inter-island communication. The cost function for any candidate core move from island *i* to island *j* is expressed as:

$$C(i,j) = (P_{Rj} - P_{Ri}) + (O_j - O_i) + \mathcal{H}$$

where P_{Rj} and P_{Ri} are the router powers if the core were in island *j* or island *i* respectively, O_j and O_i are the power overhead of MCFIFOs and VLCs associated with the respective islands, and $\mathcal{H}$ is the compute power hike for the core because of the move. The mapping threshold ψ can be expressed as:

$$\psi = (comm_j/(comm_i))\times(1/(v_j - v_i))*\omega$$

where $comm_j$ and $comm_i$ represent the total communication bandwidths of the core when it is mapped to island *j* and island *i*, respectively, v_j and v_i are the corresponding island voltages, and ω is designer specified parameter that regulates the aggressiveness of the moves. We use a conservative value of 0.0001 to balance computation and communication power, and avoid dramatically increasing total power during a move. Ultimately, the mapping threshold ψ quantifies the projected reduction in post-mapping traffic (and thus the communication power) resulting from the move under consideration. Algorithm 1 below summarizes the repartitioning approach.

Algorithm 1. Repartitioning

```
input: voltage-partitioned core graph G(V, E)
1: Compute the cost C and threshold ψ for all allowable moves
2: Make the move with the lowest cost (below the corresponding move
   threshold ψ) and lock this core in the new island.
3: Re-compute a new set of allowable moves for all the unlocked cores
   and the associated costs and thresholds.
4: Repeat steps (2) and (3) until all costs of the remaining unlocked cores
   are above their respective thresholds
output : voltage-repartitioned core graph G'(V, E)
```

4.2 Core to Tile Mapping

In the next stage, we map cores to tiles on the die. Our proposed approach consists of two major steps: initial mapping generation, and incremental swapping.

4.2.1 Initial Mapping Generation

In this step, we generate an initial core to tile mapping by traversing an Inter-Island Communication Graph *IICG*(V_{isl}, E_{isl}), where the vertices constitute the entire islands and the edges constitute the aggregate communication bandwidth between the respective islands. A breadth-first search (BFS) starting with each of the *Ω* islands as the root node, would produce *Ω* distinct sequences of islands, each of length *Ω*. The order of islands in each sequence is based on decreasing communication bandwidths with the island selected as the root node. Subsequently, the cores are mapped onto the tiles of the NoC in order of the island $sequence_j$ to generate a $mapping_j$.

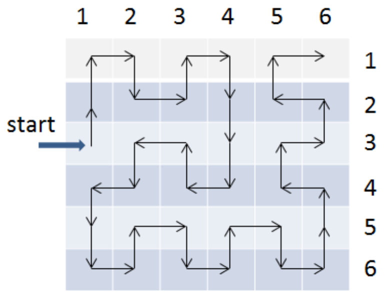


Figure 2: Sequence of tile co-ordinates for the initial mapping on a 36 core (6x6) regular mesh NoC

We follow a pre-defined sequence of tile co-ordinates for the mapping process as shown in Figure 2. Such an ordering of the core to tile mapping grows in both the x and y directions of the mesh in a symmetrical way, thereby keeping the Manhattan distances between the currently placed core and recently placed cores shorter; as compared to, say, growing the mapping in just the x or y directions (i.e., row-wise or column-wise mapping). Furthermore, our ordering ensures that the placement of the current core is adjacent to the previously placed core in order to guarantee *VI* integrity.

For all *Ω* mapping configurations generated, we perform Incremental Swapping, Routing Path Allocation (section 4.3) and compute the resultant total power of the network. Algorithm 2 below summarizes the key steps in the initial mapping generation procedure. It generates *Ω* initial mappings to constitute a *mapping-set*.

Algorithm 2. Initial Mapping Generation

```
input: core graph G'(V, E)
1: Create an inter-island communication graph IICG($V_{isl}$, $E_{isl}$)
2: for j = 1 to Ω do
3:     With $V_j$ as the root node; perform BFS on IICG to get a $sequence_j$ of
       islands
4:     Map the cores within each island in the order of the $sequence_j$ to
       obtain $mapping_j$
5: end for
output : mapping-set
```

4.2.2 Incremental Swapping

The incremental swapping step is intended to reduce link tension (Definition 3) in the NoC. Cores connected by communication links with greater tension have a higher force of attraction between them. The tensions in the mesh network force cores with high communication bandwidths to come closer during the incremental swapping step. After the initial mapping, all communicating cores have some tensions associated with them, in one or more directions. The incremental swap algorithm attempts to decrease the Manhattan distance of the core with the highest tension in the entire mesh by swapping it with another core to reduce the tension. The swaps are allowed to occur between neighbors either in the x-direction, y-direction or diagonally on the mesh structure. A swap is considered valid if it can pass three checks:

1. *Total tension decrease check:* the swap is valid only if it will result in a decreased total tension (total pre-routing traffic).
2. *VI integrity check:* the swap is valid only if it will not disintegrate any islands.
3. *Tabu-direction check:* the swap is valid only if the direction of the move is not in the Tabu list.

If a swap does not pass the checks, the swap is aborted, and the core is marked-off to restrict it from swapping for the next *d* (dimension of the mesh) number of swap attempts (iterations). If a swap is valid, then the x, y tensions and x, y co-ordinates of the cores are updated after the swap, and the complementary move direction for the core is added to a Tabu list (e.g., -y if the core moved in the +y direction) so that the core will never move back in the direction it came from. If a diagonal swap takes place, both the x and y directions are added to the Tabu list. Also, the total pre-routing traffic is updated after every swap. The incremental swapping continues until *d* consecutive aborts have been encountered. Algorithm 3 below describes the key steps in the incremental swap algorithm, which eventually decreases the Manhattan distances of high bandwidth communication flows in the mesh NoC. The input is the mapping set for which each of the *Ω* constituent mappings is processed, to create a final mapping-set. Note that cores are never swapped back in the direction of their previous locations; this gives us a theoretical upper bound on the total number of swaps that the N cores in the mesh can undergo: O(N*2*($N^{1/2}$-1)).

Algorithm 3. Incremental Swapping

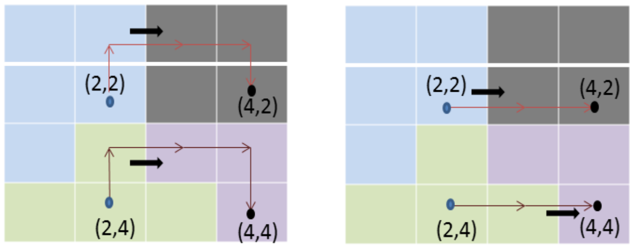
```
input: core graph G'(V, E) and mapping-set
1: for each mapping-solution   mapping-set do
2:   do until d consecutive aborts are encountered
3:     Choose the core with the maximum total tension
4:     Choose the direction with the most tension
5:     Perform the three pre-swap checks
6:     If the swap is invalid, consider other directions
7:     If no directions valid, abort;  mark-off the core for next d iterations
8:     If some direction is found to be valid, perform the swap
9: end for
output : final-mapping-set
```

4.3 Routing Path Allocation

The mapped NoC obtained after the previous stage consists of multiple *VIs* that not only run on different voltage levels, but also different frequencies. Therefore, whenever an inter-island link is inserted, voltage level converters (VLCs) and frequency level converter resources (MCFIFOs) are required in the corresponding routers. Whenever a low voltage core transmits to a higher voltage core, a VLC is needed on the outgoing port of the source router. Also, for any inter-island link, an MCFIFO is needed for the higher frequency/voltage core as the connecting link works at the lower frequency. These frequency and voltage conversion components incur an overhead in terms of power dissipation and delay. Thus, the main objective of routing is to find a path for each communication flow such that communication path lengths and the number of inter-island links are reduced.

The routing algorithm in [9] minimizes the number of inter-island links by calculating the inter-island bandwidths between different neighbor-islands and inserting proportional number of links between them. Then, the inter-island routing is carried out by using just the inserted links. One of the drawbacks of this approach is increased traffic because of less flexible link insertion that is performed once and never changed again during the disjoint routing step, leading to long communication path-lengths. Because the link-insertion and routing steps are disjoint, communication between non-neighbor islands is also not considered during the link-insertion step, giving rise to a possibility of creating isolated islands that do not communicate with any of their neighbor-islands and therefore are unable to communicate with non-neighboring islands.

Our proposed routing algorithm integrates the link-insertion and the routing steps thereby alleviating the above mentioned drawbacks from [9]. We employ minimal routing in order to minimize total traffic. The inter-island links are inserted only when minimal paths cannot be found within the residual bandwidth capacities of the existing inter-island links. Figure 3 shows how the disjoint link-insertion and routing approach from [9] has longer path lengths because the routing is constrained to the available inter-island links (shown with the thick black arrows); whereas, with an integrated approach, path lengths would be optimal. As the inter-island link does not lie in the direct path between the source (2, 2) and the destination (4, 2) in (a); path lengths are longer when compared to (b).



(a) **(b)**

Figure 3: Cores of the same color belong to the same VI. Shown are two communication flows (a) with the algorithm in [9], (b) by integrating link insertion and minimal-path routing

For each communication flow, we consider all candidate minimal paths. Out of these candidate minimal paths, we choose a routing path based on the following optimization objectives (in that order):

1) Minimize total number of inter-island link insertions needed on the path
2) Minimize total number of intra-island link insertions needed on the path

The communication flows with longer minimal paths have more choices for routing and thus have a larger scope for optimization. Also, flows with smaller bandwidths, require lesser residual capacities to be accommodated within existing links. Therefore, communication flows are sorted in the increasing order of their path lengths, in decreasing order of their communication bandwidths for the same path length; and routed in that order. While routing any communication flow over a given path, links insertions are performed whenever the existing link(s) cannot support the bandwidth of the current flow or if no links are available. In summary, this routing scheme optimizes both the number of inter-island links and total traffic in the NoC, thereby resulting in significant savings in communication power.

Finally, a voltage-assigned, mapped and routed NoC is obtained. Algorithm 4 below summarizes the routing path allocation approach.

Algorithm 4. Routing Path Allocation

```
input: core graph G'(V, E), final-mapping-set
1: Sort all communication flows  in the increasing order of path lengths
   and in decreasing order of bandwidths for the same path length
2: for each communication flow in sorted list do
3:   Out of all the candidate minimal paths, choose the set of paths that
     would require the least number of inter-island link insertions
4:   Out of the chosen paths, choose the one path that would result in the
     minimum number of intra-island link insertions
5:   Insert the necessary links and allocate the current flow bandwidth
     over the chosen routing path.
6: end for
output :Synthesized NoC with all communications routed
```

5. Experiments

5.1 Experimental Setup

To determine the quality of solutions generated by our synthesis framework, we performed experimental studies and generated NoCs for different applications. We used the ARM11 MPCore multi-core processors [22] as the base compute cores in our experiments, which support six operating voltage levels as shown in Table 1. Correspondingly, we constrained the maximum number of *VIs* to be synthesized by our framework to six. Our experiments were conducted on three applications based on pseudo-random core graphs derived using TGFF [23], and consisting of 16 (4x4), 36 (6x6), and 64 (8x8) cores, with edge weights annotated with bandwidths representing the inter-core communication requirements. The diverse core counts allow us to ascertain the scalability and applicability of our approach to low and high complexity systems. We conservatively assume that the square of the voltage scales linearly with the frequency, as in previous works [24]. The compute core power values account for both dynamic and leakage power, and vary on average as shown in Table 1.

Table 1: Core voltages and the corresponding frequencies and average power values used in the experiments

V (volts)	1.26	1.2	1.15	1.1	1.0	0.9
Freq. (MHz)	483	437	401	368	304	246
Power (mW)	126	101	85	72	49	32

The router and link powers for different voltages, frequencies and router complexities; and with varying communication loads are obtained from a modified version of ORION 2.0 [25]. The voltage/frequency assignment for a router is the same as the voltage/frequency of the core it is connected to in its corresponding tile. The voltage/frequency pair value for a link connecting two routers is the lower voltage/frequency pair value of the two routers connected by the link. As the VLCs and MCFIFOs required to interact between *VI*s incur a power overhead that is proportional to their voltage supply, we assume that every MCFIFO or VLC introduced in the router consumes 10% of the base router power, based on reported overheads from existing literature [26]. For deadlock avoidance, we use two virtual escape channels per router [27], with appropriate power overhead in our experiments.

5.2 Results

We compare the results of our synthesis framework with the results obtained by using *VI*-aware NoC synthesis approaches presented in two prior works: a heuristic based recent work on *VI*-aware regular NoC synthesis [9] that claims to improve upon other previous works such as [18]; and a synthesis approach based on a genetic algorithm (GA) in [16]. We implemented the frameworks presented in [9] and [16] to the best of our understanding and used the same performance and power models for all implemented approaches to ensure a fair comparison of the algorithms used.

Figure 4 (a)-(c) shows the total traffic that exists in the results generated (for the three applications considered) by the three approaches: GA-based [16], heuristic based [9], and our proposed VISION framework. Results are generated for a range of input Ω values that vary the number of VIs to be generated in the solution. The x-axis shows the solutions for the different input values of Ω. It can be seen that our VISION framework has significantly lower traffic compared to [9] (by up to 60%), as well as [16] (by up to 77%). The repartitioning approach used in VISION clusters the heavily communicating cores into the same *VI*, with a constraint on the hike in the communication power. This not only results in a reduction in the number of inter-island links, but also reduces the average Manhattan distance over all communication paths; as any two cores in separate islands are more likely to be farther apart than if they are within the same island. Our incremental swap mapping always attempts to reduce the communication Manhattan distances for the heaviest communication flows; leading to much lower traffic. Lastly, our routing path allocation algorithm also produces significant reduction in total traffic by prioritizing reduced path lengths. In comparison, the mapping approach in [9] and [16] does not consider reducing the Manhattan distances over the entire network in a holistic manner. Also, unlike VISION, [9] does not employ minimal routing to reduce the total number of inter-island links.

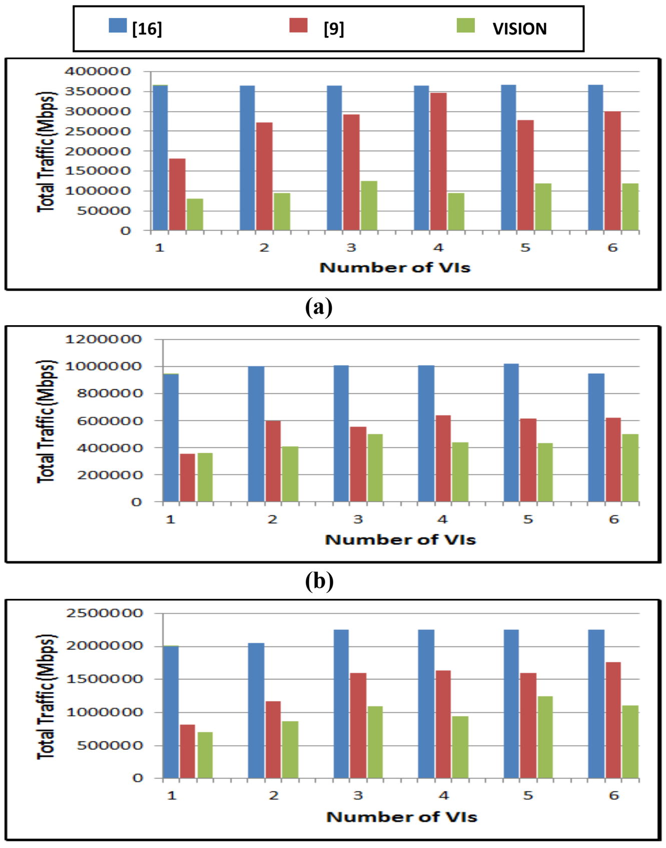


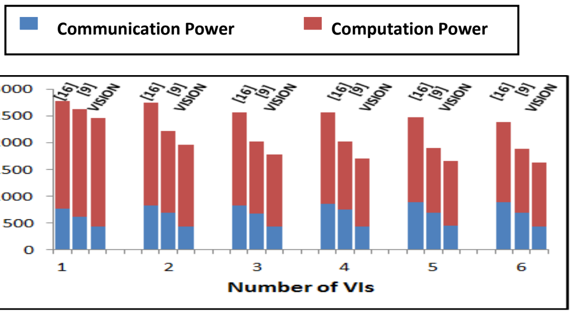
Figure 4: Total traffic in (a) 16 core mesh based NoC, (b) 36 core mesh based NoC, (c) 64 core mesh based NoC

Table 2 shows the total number of inter-island links (accompanied by corresponding VLCs and MCFIFOs) generated by the three approaches. For most cases, the GA based approach [16] has the lowest number of inter-island links compared to [9] and VISION. However, [16] accomplishes this by having many more cores being assigned higher voltages thereby reducing the variability of the voltage distribution of the cores. Thus the number of inter-island links is much lower but with highest penalties on computation power (Figure 5). On average, our VISION framework has fewer inter-island links compared to [9], particularly because our routing path allocation considers all candidate minimal paths and prioritizes the paths with the least number of inter-island links.

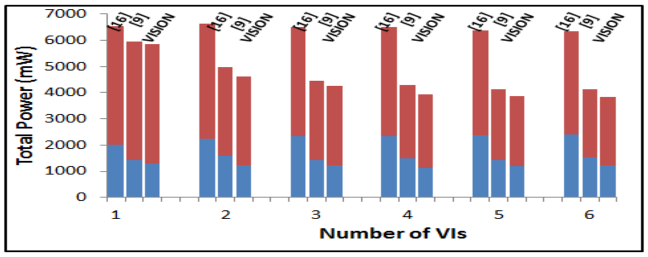
Table 2: Total number of Inter-Island Links

n/w sizes	Core_16			Core_36			Core_64		
# of VIs	*[16]*	*[9]*	*VISION*	*[16]*	*[9]*	*VISION*	*[16]*	*[9]*	*VISION*
1	0	0	0	0	0	0	0	0	0
2	16	12	9	22	18	19	26	49	45
3	20	20	12	37	33	25	40	64	63
4	23	35	16	40	33	24	60	64	62
5	32	33	18	42	50	41	58	85	63
6	25	38	22	76	64	33	58	97	97

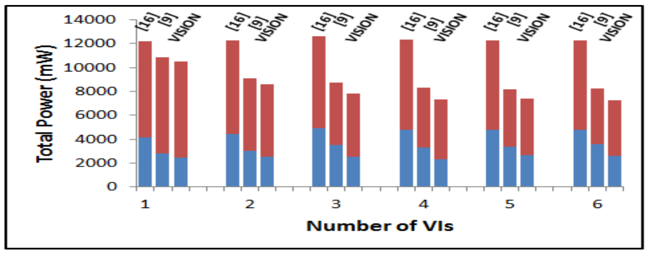
Figure 5 summarizes the total power dissipation of the solutions generated by the three approaches, for the three applications considered. A decomposition of the total power into computation and communication power is also presented. As the number of islands in the NoC increases, it can be seen that the total power decreases. This decrease in total power is due to the decreasing computation power, because with more voltage levels available, the cores in general can run at lower voltages. With reductions in total traffic as well inter-island communication, the communication power for our proposed VISION framework shows quite significant improvements compared to [9] (up to 26%) and [16] (up to 51%). Even though most of these gains are obtained due to our more communication-centric partitioning and mapping techniques, for larger applications (where the total communication power is a more significant portion of the total power, due to longer communication paths), the routing technique used provides for significant improvements as well. In terms of total power dissipation, the VISION framework outperforms both [9] (by up to 11%) and [16] (by up to 40%). These results clearly indicate that our VISION synthesis framework can provide superior solutions when compared to the best known prior works, and is thus a promising tool for automated exploration and synthesis of emerging *VI* enabled homogenous multi-core NoC systems.



(a)



(b)



(c)

Figure 5: Total power (compute + communication) (a) 16 core mesh based NoC, (b) 36 core mesh based NoC, (c) 64 core mesh based NoC

6. Conclusion

We have proposed a novel synthesis framework (VISION) for *VI*-based mesh-based NoCs. Our partitioning techniques consider not only the optimization of computation power, but communication power as well. Our mapping approach uses a distributed decision making over the whole mesh; instead of a centralized approach used in previous works. Our routing path allocation algorithm integrates link-insertion and routing in contrast to previous work. VISION produces up to 26% savings in communication power and up to 11% savings in total power; compared to the best known prior work on VI-aware NoC synthesis [9].

References

[1] S. Vangal et al., “An 80-Tile 1.28 TFLOPS Network-on-Chip in 65 nm CMOS,” Proc. IEEE ISSCC; pp. 98 – 589, Feb. 2007.

[2] Tilera Corporation. TILE64™ Processor. Product Brief. 2007.

[3] L. Benini, G. De-Micheli, “Networks on Chip: A New SoC Paradigm,” Proc. Computer, 49(1):70-71, Jan 2002.

[4] J. M. Rabaey, Digital Integrated Circuits. Prentice Hall, 1996.

[5] F. Fallah, M. Pedram, "Standby and active leakage current control and minimization in CMOS VLSI circuits," IEICE Trans. on Electronics, 88(4): 509–519, 2005.

[6] T. Chelcea and S. Nowick, "A low latency FIFO for mixed-clock system," Proc. IEEE Workshop on VLSI, pp: 119–126, April 2000.

[7] P. Choudhary, et al., "Hardware based frequency/ voltage control of voltage frequency island systems," Proc. CODES+ISSS, 2006.

[8] D. Lackey et al., “Managing Power and Performance for System-on-Chip Designs using Voltage Islands,” Proc. ICCAD., Nov. 2002.

[9] W. Jang, D. Ding, D. Pan, “A Voltage-Frequency Island Aware Energy Optimization Framework for Networks-on-Chip,” Proc. ICCAD, pp: 264 – 269, Nov. 2008.

[10] S. Murali et al., “Designing Application-Specific Networks on Chips with Floorplan Information,” Proc. ICCAD. 2006.

[11] C. Seiculescu et al., “NoC Topology Synthesis for Supporting Shutdown of Voltage Islands in SoCs,” Proc. DAC, July, 2009.

[12] P.Zhou et al., “Application-Specific 3D Network-on-Chip Design Using Simulated Allocation,” Proc. ASPDAC. Jan., 2010.

[13] K. Srinivasan, K. Chatha, “A low complexity heuristic for design of custom network-on-chip architectures,” Proc. DATE, 2006.

[14] K. Srinivasan et al., “Linear-Programming-Based Techniques for Synthesis of Network-on_Chip Architectures,” IEEE Trans. on VLSI systems (TVLSI), 14(4), April 2006.

[15] U. Ogras, R. Marculescu; “It’s a Small World After All”: NoC Performance Optimization Via Long-Range Link Insertion,” IEEE Trans. on VLSI systems (TVLSI), 14(7), July 2006.

[16] L.Leung, C. Tsui; “Energy-Aware Synthesis of Networks-on-Chip Implemented with Voltage Islands,” Proc. DAC, 2007.

[17] P. Ghosh, A. Sen; “Efficient Mapping and Voltage Islanding Technique for Energy Minimization in NoC under Design Constraints,” Proc. SAC, pp: 535–541, 2010.

[18] U. Ogras et al, “Voltage-Frequency Island Partitioning for GALS-based Networks-on-Chip”, Proc. DAC, pp: 110–115, 2007.

[19] J. Hu, R. Marculescu; “Communication and task scheduling of application-specific networks-on-chip,” IEE CDT 152(5), Sep. 2005.

[20] C Chou et al., “Energy and Performance-Aware Incremental Mapping for Networks on Chip with Multiple Voltage Levels,” IEEE Transactions on CAD (TCAD), 27(10): 1866–1879, Oct. 2008.

[21] J. Hu, R. Marculescu; “Energy and Performance-Aware Mapping for Regular NoC Architectures,” IEEE TCAD 24(4), Apr. 2005.

[22] http://www.arm.com/products/processors/selector.php

[23] http://ziyang.eecs.umich.edu/~dickrp/tgff/

[24] S. Murali, et al. “Mapping and configuration methods for multi-use-case networks on chips,” Proc. ASP-DAC, pp. 146-151, 2006

[25] A. Kahng, B. Li, L.-S. Peh and K. Samadi, “ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration,” Proc. DATE, pp: 423–428, Apr. 2009.

[26] T. Chelcea, S. Nowick; “A Low-Latency for Mixed-Clock Systems,” Proc. CSW, pp: 119–126, Apr. 2002.

[27] W. Dally, B. Towles; Principles and Practices of Interconnection Networks. Morgan Kauffman, 2005.