Design of Network-on-Chip Architectures With a Genetic Algorithm-Based Technique

Glenn Leary, Krishnan Srinivasan, Krishna Mehta, and Karam S. Chatha, *Member, IEEE*

***Abstract*—The network-on-chip (NoC) design problem requires the generation of a power and resource efficient interconnection architecture that can support the communication requirements for the SoC with the desired performance. This paper presents a genetic algorithm-based automated design technique that synthesizes an application specific NoC topology and routes the communication traces on the interconnection network. The technique operates on the system-level floorplan of the system on chip (SoC) and accounts for the power consumption in the physical links and the routers. The design technique solves a multi-objective problem of minimizing the power consumption and the router resources. It generates a Pareto curve of the solution set, such that each point in the curve represents a tradeoff between power consumption and associated number of NoC routers. The performance and quality of solutions produced by the technique are evaluated by experimentation with benchmark applications and comparisons with existing approaches.**

***Index Terms*—Design automation, genetic algorithms, network-on-chip (NoC), routing.**

I. Introduction

FUTURE system-on-chip (SoC) architectures will consist of hundreds of processing and memory elements communicating at several gigabytes per second, and will be implemented in nanoscale technologies. Communication architecture will be a key determinant of the overall performance and power consumption of these architectures. Network-on-chip (NoC) has emerged as the dominant solution for the interconnection architecture design problems of SoC design in nanoscale technologies by both academia [1], [2] , and the industry [3].

This paper addresses the design of an NoC in the context of application specific SoC architecture. Application specific SoC design offers the opportunity for incorporating custom NoC architectures that are optimized for the target problem domain, and do not necessarily conform to regular topologies such as mesh or torus. Specifically, this paper addresses the following problem [4]. Given:

- Directed communication trace graph (CTG) $G(V, E)$, where each $v_i \in V$ denotes either a processing element or a memory unit (henceforth called a node), and the directed edge $e_k = \{v_i, v_j\} \in E$ denotes a communication trace from v_i to v_j. For every $v_i \in V$, the height and width of the core is denoted by $\mathcal{H}_i$ and $\mathcal{W}_i$, respectively. The CTG is a generalized representation of the communication traces that allows cycles, and multiple edges between two processing cores.
- For every $e_k = \{v_i, v_j\} \in E, \omega(e_k)$ denotes the bandwidth requirement in bits per cycle, and $\sigma(e_k)$ denotes the latency constraint in hops.
- Router architecture, where η denotes the maximum number of input/output ports of the router, and Ω denotes the peak input and output bandwidth that the router can support on any one port. Thus, each port of a router can support equal bandwidth in input and output modes. Since a node $v \in V$ is attached to a port of a router, the bandwidth to any node from a router, and from any node to a router is less than Ω. Two quantities Ψ_i and Ψ_o that denote the power consumed per megabits per second of traffic bandwidth flowing in the input and output direction, respectively for any port of the router.
- Value $D_{\max}$ that denotes the maximum allowable distance between two routers to ensure single clock cycle data transfer.
- Physical link power model denoted by Ψ_l with units nW per Mbps per mm.
- System-level floorplan of the cores in the SoC.

Let $\mathcal{R}$ denote the set of routers utilized in the synthesized architecture, E_r represent the set of links between two routers, and E_v represent the set of links between routers and nodes. The objective of the NoC design problem is to generate a network topology $T(\mathcal{R}, V, E_r, E_v)$ such that:

- for every $e_k = (v_i, v_j) \in E$, there exists a route $p = \{(v_i, r_i), (r_i, r_j), \ldots (r_k, v_j)\}$ in T that satisfies $\omega(e_k)$, and $\sigma(e_k)$;
- the bandwidth constraints on the ports of the routers are satisfied;
- the total system-level power consumption for inter-core communication is minimized.

The application specific network topology generation problem is a variation of the generalized steiner forest problem [5], which is known to be NP hard. A survey of existing approaches that address application specific NoC design was presented by Benini [6]. Existing techniques for NoC design with irregular topologies [7]–[12] have largely ignored the length constraints posed by physical links and also their

Manuscript received March 13, 2006; revised October 06, 2006. First published February 03, 2009; current version published April 22, 2009. This work was supported in part by the National Science Foundation CAREER Award (CCF-0546462).

G. Leary and K. S. Chatha are with the Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: kchatha@asu.edu).

K. Srinivasan was with the Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287 USA. He is now with Sonics Inc., Milpitas, CA 95035 USA.

K. Mehta was with the Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287 USA. She is now with Freescale Inc., Tempe, AZ 85284 USA.

Digital Object Identifier 10.1109/TVLSI.2008.2011205

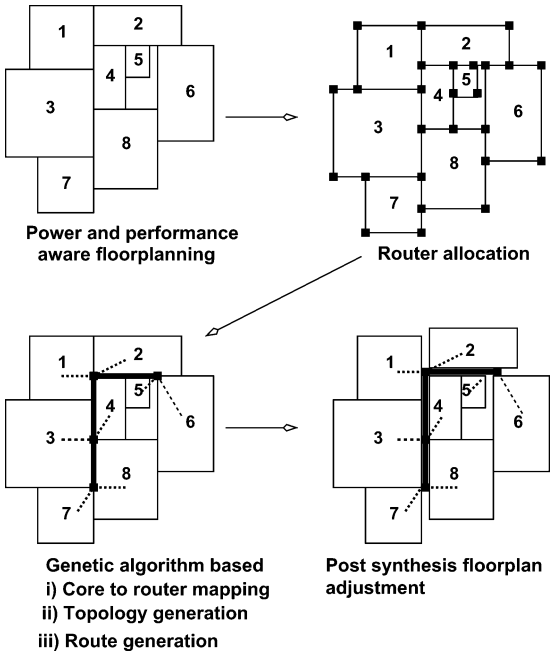

Fig. 1. Design flow for synthesis of application specific NoC.

contribution to the overall power consumption of the interconnection architecture. Consequently, they do not consider the system-level floorplan during NoC design. Srinivasan *et al.* [4] observed that the physical links are expected to consume upward of 30% of the NoC power consumption. In this paper, we also consider the system-level floorplan of the SoC, and account for the length constraints and power consumption due to the physical links. In contrast to [4] and [13] which utilize linear programming and deterministic heuristic techniques to solve the problem defined above, the paper proposes a novel genetic algorithm (GA)-based approach.[1] Our GA-based technique solves a multi-objective problem that addresses NoC power consumption and interconnection resources. The GA-based technique has the ability to escape local minima and generate excellent quality solutions in reasonable time. Further, the GA-based technique can generate a set of Pareto points where each point represents a solution with a certain power and router resource consumption. The Pareto points provide the designer with an option to choose a solution corresponding to the desired tradeoff between power and router resource consumption. We demonstrate the quality of results produced by our technique by experimenting with several benchmark designs and comparisons with existing approaches.

II. Genetic Algorithm for NoC Topology Design and Route Generation

The overall design flow for application specific NoC synthesis is depicted in Fig. 1 with the help of an example. Our technique takes the communication trace graph, a system-level floorplan and interconnection architecture elements as input. As a first step, our technique allocates routers at physical locations in the floorplan. The selection of the router locations should reduce the design space of the GA, and thus its runtime. Similar to [4], the possible router locations are assumed to be at the corners of the computation cores as shown in Fig. 1. As the possible physical locations of the routers are known, we can determine the shortest distance from any node to the routers. By the same argument, we can also determine all inter-router distances. Therefore, we can estimate the link lengths (and resulting link power consumption) in the NoC with a high degree of confidence. The physical link lengths of inter-router and node-router connections are constrained by the clock period of the core that is initiating the write operation. The designer can specify a maximum length of the physical link that permits the single clock cycle data transfer.

As the next step, our GA-based automated design technique selects the routers to be utilized in the NoC, maps the nodes to router ports, constructs the topology of the network, and routes the traffic traces on the interconnection architecture. The final layout in Fig. 1 shows the node to router mapping by the dotted lines, and the NoC topology by the thick lines. During topology generation, the physical dimensions of the routers were neglected. At the end of the topology generation stage, we adjust the SoC floorplan such that the area of the routers is taken into account. However, since the router dimensions are small, we do not expect a significant variation between the floorplans before and after the router dimensions are taken into account.

The NoC architectures generated by our technique can result in deadlocks between various traffic traces. We address deadlock avoidance during NoC generation phase, as well as a post processing step. Our static routing algorithm utilizes the knowledge of the underlying router architecture to generate deadlock avoiding routes. In case deadlocks cannot be avoided, our post processing step adds virtual channels in suitable router ports to break the deadlocks [1], [14].

A. Overview of GA

A GA is based on the biological phenomenon of genetic evolution. It maintains a set of solutions known as the population or a generation. GA operates in an iterative manner and evolves a new generation from the current generation by application of genetic operators. 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. The fitness criteria is a cost function that captures the optimization goal. The selection of solutions based on their fitness criteria models the evolutionary behavior known as the survival of the fittest. A GA-based technique typically applies three operators namely reproduction, crossover, and mutation to produce new members. Reproduction duplicates a solution across generations, crossover combines two solutions to generate two new solutions, and mutation modifies an existing solution to generate one new solution. The algorithm continues to operate in an iterative manner until the termination condition is reached.

B. Data Structures for Representation of Solution

GA-based optimization requires a representation of the population that supports efficient application of genetic operators.

[1]The research presented in this paper is an extended version of our conference paper [10].

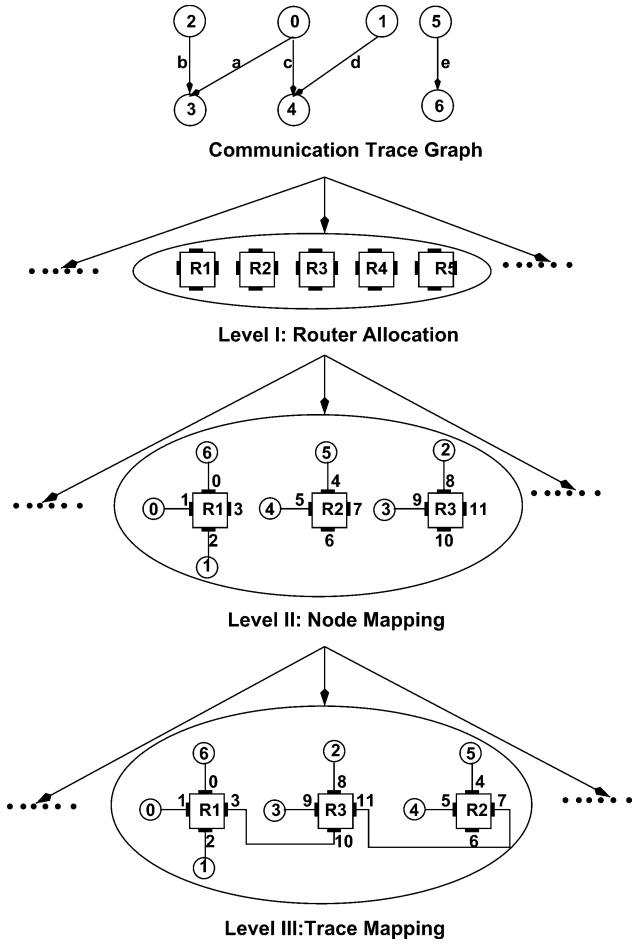

Fig. 2. Hierarchical representation.

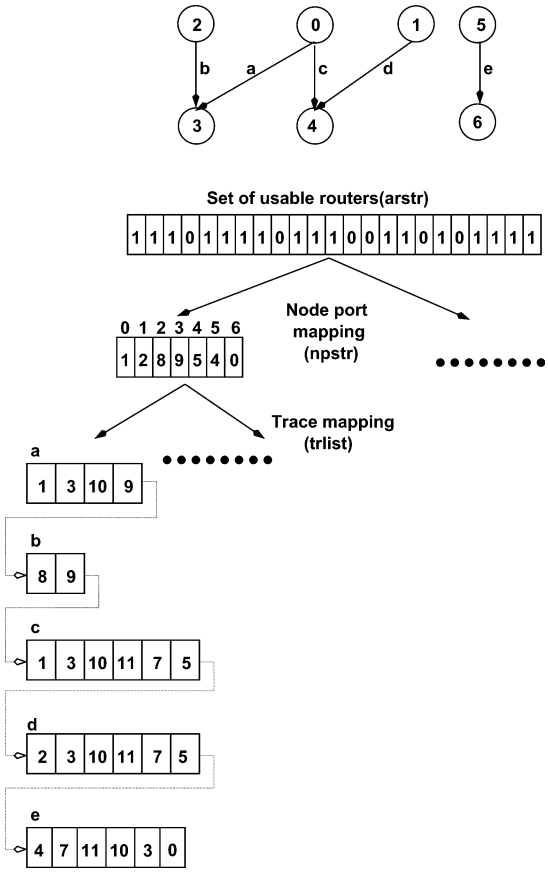

Fig. 3. Array-based data-structure.

Our GA-based technique models the population in a hierarchical manner consisting of three levels. The first level represents the number of routers, the second level denotes the mapping of the CTG nodes to the ports of the routers in the solution, and finally the third level specifies the routing of the communication traces from the source node to the sink node possibly via the ports of intermediate routers. Fig. 2 shows the hierarchical representation of the population in our GA-based technique. At the first level, our technique maintains I different architectures with various number of routers in each architecture. At the second level, for each router specification at level 1, our technique saves J different node to port mappings. Finally, at the third level, our technique maintains K different mappings of the communication traces on the ports of the routers for every node to port mapping at level 2. Note that our technique does not explicitly model the physical links between the routers. Rather, the physical links are derived from the mapping of the communication traces. For the rest of the paper we will refer to the 3 levels as router level or level 1, node level or level 2, and trace level or level 3, respectively.

1) Router Level Data Structure: Application of the genetic operators requires the representation of the population as strings of chromosomes. Strings of chromosomes are represented by sets of arrays. Fig. 3 depicts the chromosome representation of the solution shown in Fig. 2. At the first level, the number of routers in a solution is represented by a binary array $\mathrm{arstr}[\mathrm{MAXROUTER}]$, where $\mathrm{MAXROUTER}$ is the total number of routers in the architecture. Note that the location and number of routers are determined before the GA is invoked. $\mathrm{MAXROUTER}$ is only the upper-limit on the routers that can be utilized. For the example shown in Figs. 2 and 3, $\mathrm{MAXROUTER} = 24$. We denote each router by r_i where i is an integer such that $0 \leq i \leq \mathrm{MAXROUTER} - 1$. Each router that can be possibly utilized in the architecture is assigned a random location in the array given by $\mathrm{loc}(r_i)$. For example, r_4 may be assigned to location $\mathrm{arstr}[0](\mathrm{loc}(r_4) = 0)$, r_2 may be assigned to location $\mathrm{arstr}[1](\mathrm{loc}(r_2) = 1)$, and so on. Finally, all ports of the maximum number of routers that can possibly be utilized in the architecture are assigned a unique number. Hence, the four ports of r_4 at location $\mathrm{arstr}[0]$ are numbered 0, 1, 2, and 3, the four ports of r_2 at location $\mathrm{arstr}[1]$ are numbered 4, 5, 6, and 7, and so on. The GA maintains I instances of array arstr at the first level. arstr is a binary array where a "1" in location "i" denotes that the router assigned to $\mathrm{arstr}[\mathrm{i}]$ is possibly utilized in the architecture, and a "0" denotes that the router is not utilized. We say that the router is possibly utilized because even though the router is in the architecture, a communication trace may not be routed through it.

2) Node Level Data Structure: As mentioned earlier, for each instance of arstr array, the GA maintains J instances of node to port mapping. The node to port mapping at the second level is stored in an integer array given by $\mathrm{npstr}[|V|]$. Location i contains the port number to which node $v_i \in V$ is assigned. In

Fig. 3, the string npstr represents one such node to port mapping where node 0 is mapped to port 1, node 1 is mapped to port 2, node 2 is mapped to port 8, and so on. Note that since each port can map only one node, a port is not repeated in the string.

Our technique can generate NoC topologies with heterogeneous routers that have variable number of input/output ports. The number of ports in a particular router are only known once the final solution has been generated. Therefore, our technique initially assumes a constant maximum number of ports at each router. The number of ports is given by the router with maximum number of ports in the router library. For example, if the maximum number of ports in a router that is available in the IP library are 8, the GA initially assumes that all routers have 8 ports. Once the topology has been generated, the required number of ports in each router is known, and the suitable IP-block can be selected from the component library. Thus, our NoC design topology supports heterogeneous router architectures.

3) Trace Level Data Structure: Our GA maintains K instances of communication trace mappings for every instance of npstr array. The communication trace mapping is represented by a linked list trlist of integer arrays trstr_i, where the ith array refers to the communication mapping of trace i. For example in Fig. 3, the first trstr array refers to trace a, second trstr array refers to trace b and so on. Each trstr array is an ordered set of ports indicating the flow of the traffic. For example, the communication trace a passing through ports $1,\ 3,\ 10,\ 9$ in that order is represented by a array given by $[1, 3, 10, 9]$.

C. Legality Criteria for Solution Representation

In this section, we present the necessary and sufficient legality criteria for solution representation at the router, node and trace level. A violation of the criteria results in an infeasible solution.

We note that a node is mapped to a unique port. Therefore, at the router level, the only legality criterion to be satisfied is that the total number of ports, which is given by the number of "1"s in the router level string multiplied by the number of ports per router, is greater than or equal to the number of nodes in the CTG.

At the node level, the following legality criteria must hold.

1) Since a port can map only one node, a port is assigned to one and only one location in npstr.
2) A port cannot belong to a router that is not included in the corresponding router level string. Therefore, all ports that are in npstr belong to a router r_i that is included in the corresponding router level string. That is, $\mathrm{arstr}[\mathrm{loc}(r_i)] = 1$.
3) The distance between the node to the port must be less than or equal to the maximum designer specified distance to enable single clock cycle data transfer.

The GA has to check for several legality conditions at the trace level. The conditions are enumerated as follows.

1) Since the traffic trace routes are represented by arrays, every traffic trace has an associated array.
2) The array starts at position 0 with the port to which the source of the trace is mapped, followed by a different port of the same router. The array ends with an integer that represents a port mapped to the sink node.
3) A port that is assigned to a node represents the origin or termination of traces that have that particular node as a source or sink, respectively. Therefore, no other port number in the array (except for the ports mapping source and sink nodes) represents a port mapped to a node.
4) If a trace enters a port of a router, it must leave the router through one of its output ports. In order to represent this constraint, we adopt the convention that even numbered positions in the array represent ports into which the trace enters, and odd numbered positions in the array represent ports from which the trace leaves.
5) From the previous constraint, it follows that if a trace enters a port, the next integer is the port number from which the trace leaves the same router.
6) In order to avoid cycles, a port number in the array is not repeated.
7) Since the route does not contain cycles, at most two ports of a router can appear in an array. Violation of the constraint will result in a router being visited more than once, thus resulting in a cycle. This constraint, along with constraints 4 and 5 ensures that the number of ports belonging to the same router appearing in the trace array is either two or zero.
8) If a pair of ports adjacent to each other represent ports of two different routers, they should be consistent over all arrays. This pair represents two ports of different routers connected together by a physical link. This constraint forbids a port from being connected to multiple ports.
9) If a traffic is routed through a router, it must pass through two ports of that router. The two ports combined contribute toward one router hop. Therefore, the length of the array must be less than or equal to twice the latency constraint, where latency constraint is represented in number of hops.
10) The distance between two adjacent routers in the trace array must permit single clock cycle transfer of a word between the routers. As explained earlier the designer specifies the maximum distance between two routers for single clock cycle transfer. Therefore, this constraint can be satisfied if the length of the physical link between two routers is less than the designer specified distance.
11) The bandwidth constraint at the ports included in the array is not violated.

A communication trace mapping is legal if it contains legal arrays corresponding to all traffic traces.

D. Generation of Initial Population and Modified Shortest Path Algorithm (MSP)

In this section, we discuss the algorithms utilized for generation of initial population. An initial population is obtained by generating I router allocation arrays, J node to port mapping arrays for each router allocation, and K communication trace mapping arrays for each node mapping. The initial router allocation is generated by uniformly random assignment of "0"s and "1"s to all locations of every instance of arstr array. Similarly, the node to port mapping is obtained by uniformly random selection of a node of the CTG and mapping it to a port which is also selected by a uniformly random function. The node to port assignment is subject to the legality criterion described in Section II-C.

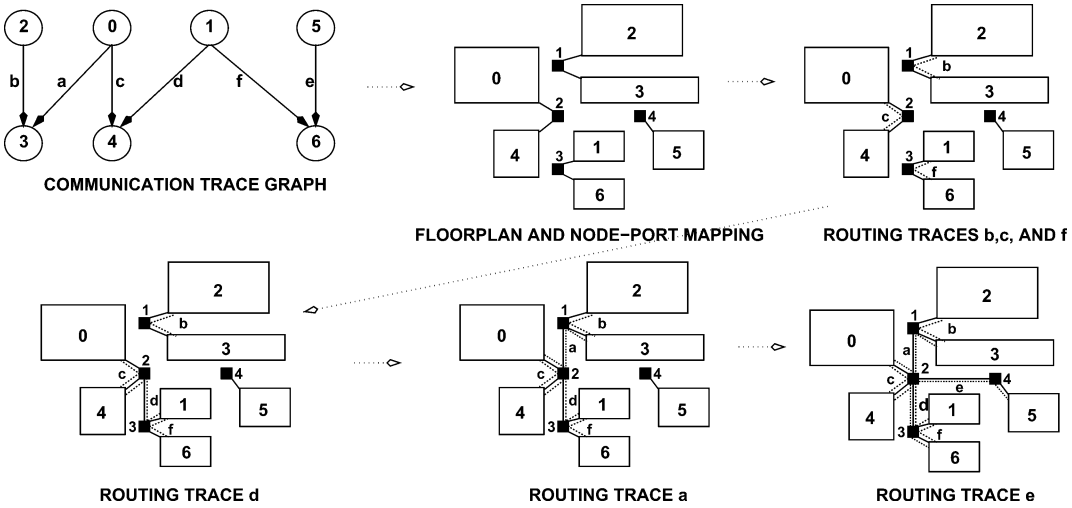

Fig. 4. Communication trace mapping.

The generation of an initial communication trace mapping is more involved. Initially, a trace is selected at random, and a MSP is applied to generate the traffic mapping from the source to the sink node, respectively. As mentioned earlier, a trace mapping is denoted by an array of port numbers. Every trace that is mapped through ports of multiple routers establishes a physical link between two consecutive ports of different routers. MSP differs from classic shortest path since it maps the traffic along the shortest route subject to the links established by already mapped traces, and the bandwidth constraints on the ports of the routers. Moreover, it generates traffic routes such that they minimize the possibility of deadlock. In the following paragraphs, we present the algorithm in detail. Deadlock avoidance will be discussed later in Section II-A.

Since the MSP algorithm finds routes with minimum power consumption, the distance between any two routers is measured in terms of link and router power consumption. Given two routers $r1$ and $r2$, the cost of establishing a path from $r1$ to $r2$ for a trace t is given by

$$c_{(r1,r2)} = \begin{cases} (\Psi_i + \Psi_o + \mathrm{dist}(r1,r2) \times \Psi_l) \times \omega(t), \\ \text{if } \mathrm{dist}(r1,r2) \leq D_{\max} \text{ and no} \\ \text{bandwidth or port violations in } r1 \text{ and } r2 \\ \infty, \text{ otherwise} \end{cases}$$

where $\mathrm{dist}(r1, r2)$ is the Manhattan distance between routers $r1$ and $r2$, and $D_{\max}$ is the maximum allowable distance between two routers to ensure single clock cycle data transfer. Therefore, the cost function establishes the following two properties.

- A shortest path directly corresponds to a path with minimum power consumption.
- If routers $r1$ and $r2$ are further than the maximum allowed inter-router distance apart, or the route results in bandwidth or port violations in $r1$ or $r2$, the cost is set to ∞ and the trace is not routed through that link.

Consider the CTG and initial node to port mapping shown in Fig. 4. The flow of the algorithm is denoted by the dotted arrows. The communication trace mapping is generated by selecting a trace at random, for example "b". We show the trace "b" by a dotted curve since it is routed through just one router. Similarly, traces "c" and "f" are also shown by dotted curves. The next trace that is selected is "d". As the source (node 1) and sink (node 4) of trace "d" are mapped to different routers, we invoke the MSP. The distance between source and sink nodes of a communication trace is given as the path from source to sink with minimum power consumption, subject to the previously mapped traces, and bandwidth constraints on the ports. Since, no inter-router trace has been mapped, traffic "d" is mapped by the shortest path. Mapping of trace "d" effectively establishes a link between the routers to which nodes 1, and 4 are mapped. In other words, routers $r2$ and $r3$ are connected by a physical link. The next trace that is picked is "a", and its routing establishes a physical link between the routers mapping the corresponding nodes, which are routers $r2$ and $r1$. Finally, trace "e" is routed. First, it is routed from router $r4$ to router $r2$ since it is the only router that lies within the maximum allowed distance for single clock cycle data transfer. This establishes a link between routers $r2$ and $r4$. The remaining part of the route is generated by utilizing the already existing link between routers $r2$ and $r3$.

It is possible that MSP is unable to find a path from the source to sink, or the available path violates the latency constraint. In such a case, the traffic trace is left unmapped, and a penalty is accrued as described in the discussion on the fitness function in the following section.

Fig. 5 presents the MSP algorithm. Line 1 is the initialization phase where all traces are marked "free". The algorithm iterates until "free" traces are available. In each iteration of the loop, it obtains a random free trace (line 3), and attempts to obtain a shortest path for the trace. If a valid route is not found, the trace is unmapped (line 6). On the other hand, if a valid route is found, the corresponding physical links are updated to accommodate the route (line 8). At each iteration, the selected trace is tagged, such that it is not explored again.

E. Pareto Points and Fitness Calculation

Our technique generates the Pareto points on the power consumption versus router requirement plot for a given application. For example, Fig. 6 depicts three Pareto points corresponding to the solutions with least power consumption for NoC topologies

```
Modified Shortest Path
begin
1    mark_all_traces_free()
2    while free_traces_available()
3      t = get_free_trace()
4      r(t) = shortest_path(t)
5      if (r(t) ≠ valid)
6        unmap(r(t))
7      else
8        update_physical_links()
9      end-if
10     tag(t)
11   end while
end
```

Fig. 5. Pseudo code for MSP algorithm.

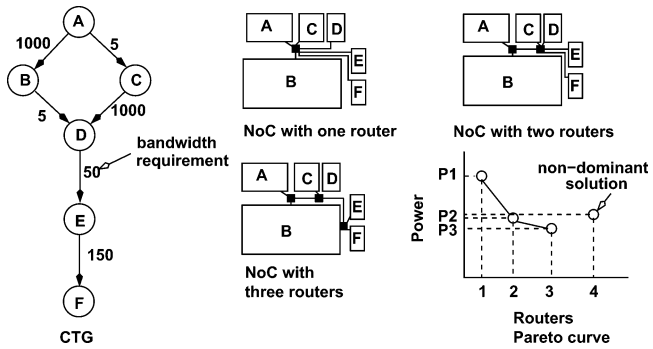

Fig. 6. Pareto points for NoC generation.

with one, two and three routers, respectively. We observe that it is prudent to choose a solution with two routers, over a solution with one router that results in high power consumption, and a solution with three routers that results in a small reduction in power consumption. Given the Pareto curve for a particular application the designer can select the solution that offers the best tradeoff between power consumption and router requirement.

In order to generate the Pareto curve, we define the well known concept of dominant solutions in the context of multi-objective optimization [15]. In the GA, a solution Z^* is said to dominate Z if Z^* is better than Z in all objectives. In Fig. 6, solution 3 dominates solution 4, as solution 3 has lower power consumption, as well as lower number of router resources. A solution is non-dominant if there exists a solution that dominates it. In the example shown in Fig. 6, the solution with four routers consumes more power than the solution with three routers, and hence, is not part of the Pareto curve. At each generation, corresponding to each router, our GA maintains the legal solution (with legal routes for all traces) that consumes least power. On termination, the GA outputs the Pareto curve obtained from the set of solutions belonging to the last generation.

The size of the population is larger than the number of Pareto points. Therefore, in addition to the Pareto points, the GA also maintains non-dominant solutions for each generation. A non-dominant solution is selected on the basis of its fitness. The fitness calculation of each solution includes the area under the solution on the power consumption versus router requirement plot. The area under the solution is given by the product of the projections on the X- and Y-axes, respectively. Higher the area under the solution, lower the fitness of the solution, and vice versa.

In a particular generation not all members of the population represent solutions in which all traces are routed. Such solutions

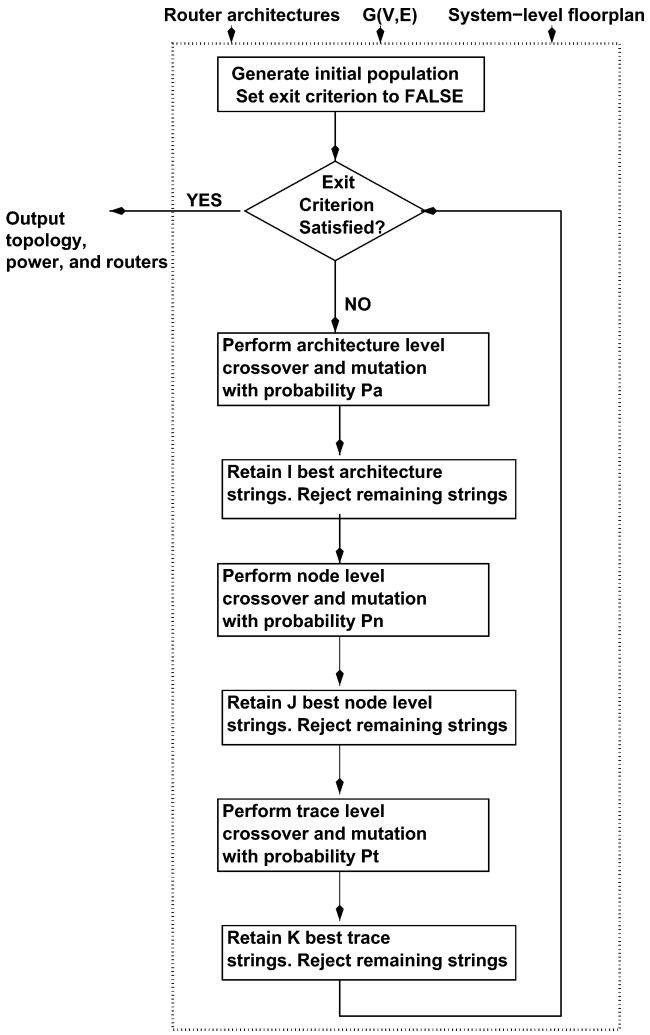

Fig. 7. GA for custom NoC design.

are denoted as illegal solutions. An illegal solution can occur if the MSP is unable to route all traces successfully. As mentioned earlier, a dominant solution is a legal solution. A non-dominant solution can be an illegal solution.

The overall fitness of a non-dominant solution is given by

$$\text{fitness(solution)} = \frac{1}{r \times p + \gamma * u}$$

where r is the number of routers in the topology, p is the power consumption, γ is the weight given to unmapped traces, u is the number of unmapped traces. The value $(r \times p)$ denotes the area of the solution on the power consumption versus router requirement plot. The value of γ is set very high to effectively pull down the fitness of illegal solutions.

During the evolution of the next generation of a population, the GA first selects the dominant solutions and then selects the non-dominant solutions based on their fitness.

F. Overview of the Optimization Technique

Fig. 7 shows the top level flowchart of our GA-based optimization technique. The input to the technique is the set of router architectures, the communication trace graph $G(V, E)$, and the

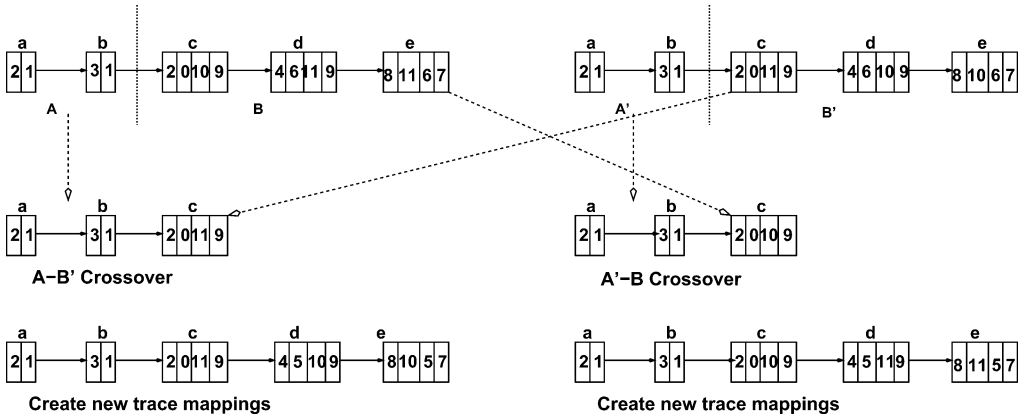

Fig. 8. Trace level crossover.

system-level floorplan. An initial population of solutions is generated using the algorithms described in Section II-B, and the fitness of each trace level string (solution) is calculated. Initially, the exit criterion is set to false. Our technique applies genetic operators at the three levels of solution hierarchy with different probabilities. For each genetic operation at a higher level of hierarchy, the GA explores several operations at the lower levels. This approach aids in a structured design space exploration for the problem. At each level the number of solutions produced by crossover is much larger than those produced by mutation. At each hierarchical level new individual solutions are produced by the application of the genetic operators. A new generation is produced by selection of the M fittest members among the current generation and the new individual solutions, where $M = I$ for level 1, $M = J$ for level 2, and $M = K$ for level 3. Selection of members of current generation for the next generation models the reproduction operation. At all three levels of the hierarchy, the fitness is given by the strongest complete solution at the trace level. Since each router allocation has J instances of node mapping and every node mapping has K instances of trace mapping, the fitness of a particular router allocation (at level 1) is given by the strongest trace among the $J \times K$ possible trace level mappings. Termination condition is reached if the N successive generations do not result in any change in the Pareto curve. We set N to summation of the number of nodes and edges in the CTG, that is $N = |V| + |E|$.

G. Genetic Operators

In this section, we discuss the crossover and mutation operators that are utilized to produce new individuals from an old generation. The reproduction operator is implicitly applied during the creation of the new generation from the set of current generation, and new members. Since the solution is modeled at three levels, we will consider the application of these operators at three levels.

1) Crossover Operation: The crossover operator selects two solutions or parents from the previous generation and produces two new solutions or children. In this section, we discuss the crossover operation of strings at the trace, node, and router levels.

Trace Level Crossover: The trace level crossover operation is applied to every set of K traces to generate K_{cross} new individuals. The trace level crossover operation is illustrated in Fig. 8. Two trace mapping link lists belonging to the same node mapping are chosen randomly from the K linked lists. Each of the two linked lists is partitioned into two at the same randomly selected cut point. The cut point for the two lists is shown by vertical dotted line in the top of Fig. 8. In Fig. 8, the linked lists are divided into $A - B$, and $A' - B'$ sub-lists, respectively. The crossover operator proceeds to create two new individual solutions by appending the lists as $A - B'$, and $A' - B$. As a convention, during the concatenation, the first half of the list (A or A') dominates the second half (B' or B). Therefore, based on the communication trace mapping of A, some trace mappings of B' may violate the legality criteria presented in Section II-C. The violations that can occur are as follows.

- It may not be possible to generate a physical route for the trace due to the previously imposed physical connections by traces belonging to A.
- Mapping the trace on the links may lead to a bandwidth violation in one or more ports of the routers.

If a communication trace mapping in B' is not legal due to a trace mapping in A, that particular communication trace is remapped by invoking the MSP algorithm. For example, in the left part of Fig. 8, trace "d" is remapped as the route for trace "c" is modified by crossover. A communication trace is left unmapped if the MSP is unable to generate a route from source to destination.

Node Level Crossover: The node level crossover operation is applied to every set of J mappings to generate J_{cross} new individuals. The node level crossover operation is illustrated in Fig. 9. Two node level arrays belonging to the same router allocation are chosen at random from the J mappings. The two arrays are divided into two ($A - B$, and $A' - B'$ in the figure) at the same randomly selected position. The two arrays are concatenated as $A - B'$, and $A' - B$ such that a port is mapped to at most one node. The second legality condition for node level crossover (see Section II-C) is automatically satisfied as the children and parent strings always belong to the same router level string. As in the case of trace level crossover, the mapping in A or A' dominates the mapping in B' or B. It is possible that the some of the nodes belonging to B (or B') cannot be mapped as the corresponding port in the parent string A (or A' respectively) maps a different node. In such cases, a new mapping for

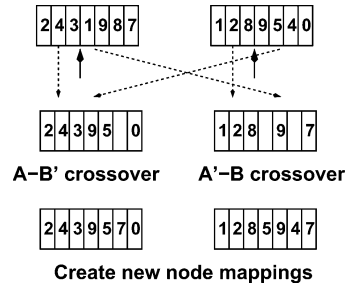

Fig. 9. Node level crossover.

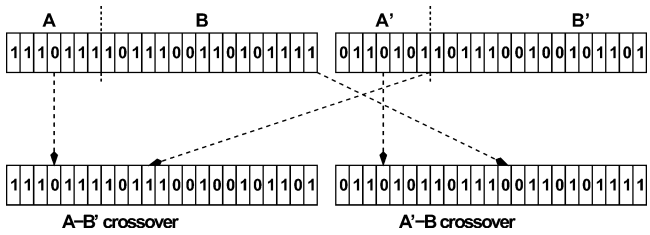

Fig. 10. Router level crossover.

the node is generated randomly. For example, as shown in the figure, the mapping of node 5 in array $A - B'$ is initially left blank since port 4 is occupied by node 1. Hence, the mapping of node 5 is generated randomly. Similarly, the mapping of nodes 4 and 5 are randomly selected in the array $A' - B$.

Once the node level mappings are defined, the communication trace mappings are generated for the two new individual solutions. The technique duplicates as many communication trace mappings as possible from the parents' trace mappings to the children. The technique picks the first trace level strings of A and B, and duplicates as many routes as possible on the first trace level string of the child, such that the trace level legality constraints are not violated. The procedure is repeated for the remaining trace level strings of the parents. Since the node mapping of the children is not identical to that of the parents, some of the duplicated routes (trace mappings) for the traces may be illegal. These traces are not duplicated, and are instead regenerated by invoking the MSP algorithm.

Router Level Crossover : The router level crossover operation is applied to every set of I router allocations to generate I_{cross} new individuals. The router level crossover mechanism is illustrated in Fig. 10. Two router level allocations are chosen randomly from I arrays, and a random point is selected to divide each array into two. In Fig. 10, the two parts have been shown as $A-B$, and $A'-B'$, respectively. Two new solutions are created by concatenating the arrays as $A-B'$ and $A'-B$, respectively. If the number of ports in the new string is less than $|V|$, the crossover is rejected, and the process is repeated with re-selection of router allocation arrays. Once the router level allocations are defined, node level and communication trace mappings are generated for the two individual solutions. As before, as many node level mappings and corresponding trace routes are duplicated in the children as possible. All node level mappings of sub-array A (and A') are duplicated. All node level mappings of sub-array B' (and B) are also duplicated subject to legality criteria for node mapping, presented in Section II-C. Nodes whose mapping violated the legality criteria are randomly assigned to open ports. The communication trace mappings are generated as in the node level crossover.

2) Mutation Operation: In this section, we discuss the mutation operation at the trace, node, and router levels, respectively. At each level, the mutation operator randomly selects a parent solution from the current generation, and randomly causes a small local change to generate a new individual solution.

Trace Level Mutation: The trace level mutation operation is applied to every set of K traces to generate K_{mutate} new individuals. The mutation operator is applied to every set of trace level mappings for every node level mapping in the population. The trace level mutation operator first selects a trace mapping at random from the K traces assigned to a particular node. At the trace level, there are some traces that are mapped to the architecture, and some are left unmapped due to violation of legality criteria. The mutation operator then selects a mapped trace at random and adds it to the set of unmapped traces. Next, it proceeds to randomly select an unmapped trace and map it to the architecture by invocation of the MSP algorithm. The process continues until as many unmapped traces are mapped as possible subject to the legal mapping constraints.

Node Level Mutation: The node level mutation operation is applied to every set of J mappings to generate J_{mutate} new Individuals. The node level mutation is applied to every set of node mappings for every router allocation in the current generation. Two ports (u, v) in a particular router allocation are selected at random. A node mapping for the same router allocation is also selected at random. If the node mapping has either one or two nodes assigned to u or v, their mapping is exchanged. If the node mapping has only one node mapped to a port, say u, its mapping is changed to v. If the node mapping has no nodes assigned to either u or v, the process is repeated by selection of two new ports. Once a new node mapping is generated, all communication traces associated with the moved nodes are added to the set of unmapped traces in all trace mappings. The MSP algorithm is invoked for each communication trace mapping to map unmapped traces.

Router Level Mutation: The router level mutation operation is applied to every set of i router allocations to generate I_{mutate} new individuals. The router level mutation is applied to every router allocation in the current generation. As mentioned before, the router allocation is specified by an array of binary digits. Router level mutation is applied by the selection of a random location in the array, and the inversion of the corresponding bit. If a "0" is inverted to "1" a router is added and no change is applied to the lower levels. On the other hand, if a "1" is inverted to "0" a router is removed. Hence, all node level mappings that contain any ports belonging to the removed router and associated communication traces are regenerated similar to the initial population creation.

H. Post-Synthesis Floorplan Adjustment

During the interconnection architecture stage, we assume that the routers are located at the corners of the cores of the floorplan. After the NoC topology generation stage, we adjust the floorplan such that the actual size of the routers are taken into account. As mentioned before, the area occupied by the routers is very small,

and our router architecture with 5 ports, first-input–first-output (FIFO) depth of 16, width of 32, and 2 virtual channels consumed only 0.19 mm^2 in 65-nm technology. Therefore, this adjustment of the floorplan after NoC synthesis does not cause significant difference between the pre-synthesis and post-synthesis floorplans.

I. Deadlock Avoidance

The MSP router discussed earlier attempts to avoid certain types of deadlocks. The router architecture assumed by our technique has separate input and output ports to route traffic to and from the router.[2] In other words, the traffic traces entering and leaving the router from a port are isolated. The MSP algorithm assigns the same path in the forward as well as the return directions (subject to bandwidth constraints) to route traces between two routers in the NoC topology. Consequently, it avoids deadlocks that occur due to overlap between forward and return paths.

However, shortest path strategy that is aimed at power consumption minimization cannot prevent deadlocks in general. As we utilize static routing scheme we can detect the possibility of deadlock *a priori* by construction of a channel dependency graph [16]. A routing technique is deadlock free if the channel dependency graph does not contain cycles. As MSP is a deterministic custom routing algorithm, it provides us the opportunity to statically inspect the routes to determine which router ports can cause deadlocks. Additional virtual channels can be added to these ports such that the routing is deadlock free [16].

III. Results

In this section, we present the results obtained by execution of our technique on several multimedia benchmark applications. We compare the results generated by the GA with an optimal ILP-based technique [4], and a recursive partitioning based heuristic called ANOC [13] that address the same problem. We also evaluate the overall design flow by comparing the designs obtained by post- and pre-floorplanning NoC synthesis. Finally, we present the NoC designs and Pareto curve for the set-top box application.

A. Experimental Setup

1) Benchmark Applications: We generated custom NoC architectures for five multimedia (rows G1-G5), and five network processing (rows G6-G10) benchmarks (see Table I). For the benchmarks G1 through G7, the size of the ARM core was estimated to be $1.5 \times 1.5 = 2.25$ mm^2. For benchmarks G8 through G10, the size of the cores were provided in [20]. The network interface, whose area is estimated to be 0.2 mm^2[21] was included in the calculation of the core area. The router architecture utilized in the paper consists of 2 virtual channels, FIFO depth of 16, and width of 32 bits. In 65-nm technology, the area of the router was estimated to be 0.19 and 0.37 mm^2 for 5- and 9-port routers, respectively.

2) Power Models and Floorplanner: We characterized our router architecture in a 65-nm TSMC low power library. In this technology, the power consumption for the input and output port was estimated to be 204 nW/Mb/s and 94 nW/Mb/s, respectively. The link power consumption was estimated to be 89 nW/Mb/s/mm. We utilized the Parquet floorplanner [22] to obtain our floorplans.

[2]The detailed explanation of the architecture can be obtained from [4].

TABLE I
Benchmarks

Graph	Graph ID	Nodes	Edges
JPEG Encoder [17]	G1	8	21
MPEG-4 decoder [8]	G2	12	13
MWD [8]	G3	12	13
VOPD [8]	G4	12	13
Set-top Box [18]	G5	25	27
AH Auth-IPv4 [19]	G6	9	8
Diffserv-IPv4 [19]	G7	11	10
NP1 [20]	G8	15	22
NP2 [20]	G9	17	24
NP3 [20]	G10	24	42

TABLE II
Technique Nomenclature

Max. link length	Max. ports	Technique
$2.5mm$	5	$GA_{2,5}$
$2.5mm$	∞	$GA_{2,\infty}$
∞	5	$GA_{\infty,5}$
∞	∞	$GA_{\infty,\infty}$

3) Number of Ports Per Router and Maximum Link Length: Our experiments were guided by two parameters namely, maximum number of ports per router that can be synthesized such that the timing constraints are met, and the maximum link length that ensures a single clock cycle transmission. The IP library may either provide a hard router architecture core with a fixed number of ports, or a soft core with parameterizable number of router ports. If the router architecture is a hard IP, the NoC synthesis tool must take the number of ports of the router as a constraint. On the other hand, if the IP is a soft core, the constraint on the number of ports is not applicable. We present results when the number of ports per router is limited to 5 (hard router IP core), as well as when the number of ports is parameterizable (soft IP core).

For 65-nm technology, we estimated the link delay to be 0.02 ns/mm. At a router operating frequency of 333 MHz, this delay does not contribute significantly to the overall delay, and can be ignored. Benini *et al.* [2] predict that in the future, NoCs will be clocked at 5 GHz or more. At this frequency, the router delay will be of the order of a tenth of a nanosecond. The link delay is expected to stay at almost the same value of 0.02 ns/mm [2]. Thus, the link and router delay are comparable, and this puts a constraint on the maximum allowable link length between two routers. More recently, Vangal *et al.* [23] presented a 5-port router aimed at mesh topologies that can operate at 4 GHz in 65-nm technology. The inter-router distance in their design was 1.5 mm along the X-axis and 2 mm along the Y-axis. We obtained experimental results with and without link length constraints. For the constrained case similar to Vangal *et al.* [23], we set a maximum link length of 2.5 mm.

Table II assigns a unique name for the instances of our technique with and without link length constraints, and with and without port constraints, respectively.

B. GA Tuning Parameters

We utilized experimental data to determine the GA tuning parameters. It was observed that a GA population size of 1000

TABLE III
COMPARISON AGAINST EXISTING APPROACHES

Base Technique	Comparative Techniques		Metrics	Benchmarks										Avg.
				G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	
$ILP_{2,5}$	$GA_{2,5}$	Min. power	Power	1.006	1.046	1.049	1.001	1.11	1.02	1	1	1.016	1.088	1.034
			Router	1	1.25	1.25	1	1.111	1.333	1	1	1.4	0.889	1.123
		Min. router	Power	1.006	1.047	1.073	1.001	1.161	1.029	1	1.001	1.03	1.086	1.043
			Router	1	1	1	1	1	1	1	1	1	0.889	0.989
	$ANOC_{2,5}$		Power	-	-	-	-	-	-	-	-	-	-	-
			Router	-	-	-	-	-	-	-	-	-	-	-
$ILP_{2,\infty}$	$GA_{2,\infty}$	Min. power	Power	1	1.01	1	1.007	1.038	1	1.002	1	1	1.037	1.009
			Router	1	0.5	1	1	1	1	1	0.5	1	1.25	0.925
		Min. router	Power	1	1.01	1	1.09	1.295	1.11	1.002	1.096	1.05	1.14	1.079
			Router	1	0.25	1	0.333	0.333	0.5	0.5	0.333	0.667	0.5	0.542
	$ANOC_{2,\infty}$		Power	1	1.117	1.16	1.048	1.249	1.11	1.002	1.177	1.087	1.409	1.136
			Router	1	0.75	2.33	1	1.33	0.5	0.5	1	2.667	1.5	1.258
$ILP_{\infty,5}$	$GA_{\infty,5}$	Min. power	Power	1.006	1.03	1.04	1.001	1.067	1.024	1	1	1.03	1.069	1.027
			Router	1	1	1.2	1	1.25	1.33	1	1	1	1	1.078
		Min. router	Power	1.006	1.03	1.044	1.001	1.164	1.029	1	1	1.054	1.11	1.044
			Router	1	1	0.8	1	1	1	1	1	0.833	0.889	0.952
	$ANOC_{\infty,5}$		Power	-	-	-	-	-	-	-	-	-	-	-
			Router	-	-	-	-	-	-	-	-	-	-	-
$ILP_{\infty,\infty}$	$GA_{\infty,\infty}$	Min. power	Power	1	1	1.007	1.003	1.024	1	1.002	1	1.028	1.019	1.008
			Router	1	1	0.667	1.5	1.2	1	0.5	1	1.333	0.667	0.987
		Min. router	Power	1	1	1.143	1.095	1.339	1.11	1.002	1.056	1.3	1.14	1.119
			Router	1	1	0.33	0.5	0.2	0.333	0.333	0.5	0.333	0.333	0.487
	$ANOC_{\infty,\infty}$		Power	1	1.086	1.222	1	1.148	1.11	1.002	1.204	1.033	1.156	1.096
			Router	1	1	1	1	0.8	0.333	0.333	0.5	0.667	0.667	0.73

served as a good tradeoff between design space available for exploration, and the runtime of the algorithm. The population size of 1000 was divided into 10 router level strings, 10 node level strings per router level string, and 10 traffic level strings per node level string. We set the termination condition to be $|V| + |E|$ iterations with no change in Pareto curve. The router level crossover probability was set at 0.1, node level probability at 0.5, and trace level probability at 1. The value of γ used in the calculation of fitness of the solution was set to 1 000 000.

C. Results and Discussion

1) Comparison With Existing Approaches: We compared the results produced by our GA with an optimal ILP-based technique [4], and a heuristic technique called ANOC [13] that address the same problem. The ILP has an exponential runtime complexity, and takes several hours to generate optimal solutions for many benchmarks. ANOC is a technique that operates on the given system-level floorplan, and invokes a recursive bi-partitioning-based heuristic to generate the final NoC. Both ILP and ANOC are single objective techniques that minimize power consumption. Further, the ANOC technique does not consider constraints on the number of ports in the router.

Table III presents the results for the four different cases discussed in the experimental setup section. The values in the table are normalized to the corresponding results produced by the optimal ILP-based technique. In the table for each design constraint (link length and number of port) combination we present the power and router requirements for minimum power and minimum router design generated by our GA based approach. Finally, the last column of the table presents the average values for the power and router requirement comparisons.

In comparison to ILP, the GA designs for minimum power on an average consume only slightly higher power (maximum is 3.4% for $GA_{2,5}$). However, the GA designs for minimum router on an average require fewer resources than the designs produced by ILP. This is due to the fact that the ILP-based approach primarily optimizes power consumption and does not directly optimize router resource consumption. The low deviation in average power consumption for the GA designs with minimum power in comparison to ILP reinforces our claim on the quality of the solutions generated by the GA. The slightly lower power consumption of the ILP comes at a much higher runtime.

TABLE IV
RUNTIMES

Runtime (secs)			
Graph size	GA	ILP	ANOC
5	≈ 5	≈ 5	< 1
15	≈ 300	> 42100	< 1
25	≈ 1800	> 42100	< 1

ANOC was able to generate solutions only when the port constraints were not imposed. Thus, the table includes comparative results for $\mathrm{ANOC}_{2,\infty}$ and $\mathrm{ANOC}_{\infty,\infty}$.[3] Our GA based approach out performs ANOC in all instances. ANOC is a low complexity heuristic, and does not explore a large design space before arriving at the final solution. On the other hand, the GA explores several solutions as it evolves through successive generations, and consequently generates superior designs.

Table IV presents the comparison of run times of the three techniques. It took the ILP several hours to converge to the optimal solution. On the other hand, the GA converged to its best solution in a few minutes. Due to its low complexity, ANOC was able to generate solutions within 1 s for all benchmarks. Although the GA takes more time to generate solutions, its solution quality is better than ANOC.

Our router architecture included two virtual channels per input port, and two virtual channels per output port. For this

[3]When no port constraints and no link length constraints are imposed, it is possible to generate a solution trivially by connecting all the cores to a single router. However, due to the contribution of link power consumption, a solution with only one router may not always be the optimal solution. Consequently, the results generated by the various techniques are different.

TABLE V
COMPARISON BETWEEN $\mathrm{GA^{post}}$ AND $\mathrm{GA^{pre}}$

No.	Base Tech.	Comp. Tech.	Metrics	Benchmarks										Avg.
				G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	
1	$GA^{post}_{2,5}$	$GA^{pre}_{2,5}$	Power	-	-	-	-	-	1.515	1.558	-	-	-	-
			Router	-	-	-	-	-	1	1	-	-	-	-
2	$GA^{post}_{2,\infty}$	$GA^{pre}_{2,\infty}$	Power	1.062	-	-	-	-	1.7	1.2	-	-	-	-
			Router	1	-	-	-	-	0.5	0.5	-	-	-	-
3	$GA^{post}_{\infty,5}$	$GA^{pre}_{\infty,5}$	Power	1.551	1.584	1.859	1.719	1.832	1.515	1.558	1.662	1.712	1.719	1.671
			Router	1	0.8	0.667	1	0.9	1	1	0.833	1.167	0.818	0.918
4	$GA^{post}_{\infty,\infty}$	$GA^{pre}_{\infty,\infty}$	Power	1.427	1.9	1.12	1.968	1.519	1.706	1.207	1.369	1.682	1.402	1.53
			Router	1	0.5	0.25	0.5	0.167	0.5	0.5	0.5	0.25	0.5	0.467

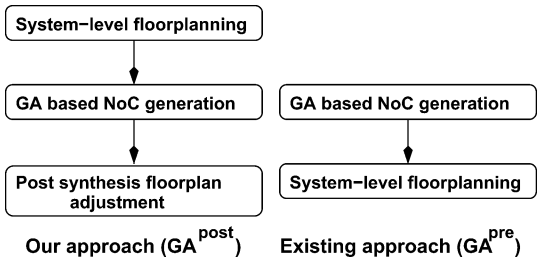

Fig. 11. Our approach versus existing approach.

architecture, we determined that deadlocks do not occur in any of our designs. The two virtual channels are able to successfully break any cycles in the channel dependency graph.

2) Comparison of Pre- and Post-Floorplan NoC Synthesis: In this section, we compare our approach of synthesizing the NoC on the system-level floorplan with an approach that synthesizes the NoC and then invokes the floorplanner to generate the layout. We compared the results for the four cases, $\mathrm{GA}_{2,5}$, $\mathrm{GA}_{2,\infty}$, $\mathrm{GA}_{\infty,5}$, and $\mathrm{GA}_{\infty,\infty}$. In each of the four cases, we utilized the Parquet floorplanner to generate the system-level layout, and our GA based technique for NoC architecture generation.

Table V contrasts our approach with existing approach. Our approach ($\mathrm{GA^{post}}$) invokes a system-level floorplanner, followed by the GA for NoC topology generation, and finally, floorplan adjustment to generate the NoC design (see Fig. 11). The existing approach ($\mathrm{GA^{pre}}$) generates the NoC topology by invoking the GA, followed by a call to the floorplanner to obtain the final NoC with floorplan. In Table V, we present normalized (with respect to $\mathrm{GA^{post}}$) power consumption and router requirement values for the minimum power designs generated by the two design flows.

Comparison for $\mathrm{GA}_{2,5}$*:* $\mathrm{GA^{pre}}$ generates the interconnection architecture without any knowledge of the floorplan, and then floorplans the NoC with link bandwidth as net weights such that the link power consumption is minimized. Since the NoC architecture generation phase has no information about the link lengths, the final solution has long links that violate the link length constraints. In our experiments, the link lengths were violated for all benchmarks except G6 and G7. On the other hand, our approach generates the NoC with the knowledge of the link lengths, and therefore, is able generate valid solutions for all benchmarks. Further, even for benchmarks G6 and G7 our technique results in solutions with lower power consumption. Again, the difference is due to lower link power consumption in our designs (as the number of routers are same).

Comparison for $\mathrm{GA}_{2,\infty}$ *:* Since link lengths and corresponding link power consumption was not taken into consideration during the synthesis stage, $\mathrm{GA^{pre}}$ trivially generated solutions with just one router. Consequently, the final floorplan had link length violations on all designs except G1, G6, and G7. Again, the power consumption of our designs was consistently lower than $\mathrm{GA^{pre}}$ even though the number of routers were some times higher (G6 and G7). As in the previous experiment the difference was due to the lower link power consumption.

Comparison for $\mathrm{GA}_{\infty,5}$*:* In this case, $\mathrm{GA^{pre}}$ was able to generate valid solutions. However, all the solutions had higher power consumption as opposed to our designs. On an average the solutions generated by $\mathrm{GA^{pre}}$ consumed 1.671 times more power than our designs even though the router requirements were comparable (about 0.918 times our designs). Again, as our technique accounts for link power consumption it consistently generates superior designs.

Comparison for $\mathrm{GA}_{\infty,\infty}$*:* In this case, neither port constraints nor link length constraints were applied. As a result, $\mathrm{GA^{pre}}$ generated solutions with just one router, which is trivially the best solution when link power consumption is not considered. But when the floorplanner was invoked, long link lengths resulted in larger power consumption. On the other hand, due to the knowledge of the floorplan, our approach was able to optimize the link power consumption, and introduce additional routers to generate solutions with lower overall power consumption. The $\mathrm{GA^{pre}}$ solutions on an average consume 1.53 and 0.467 times the power consumption and routers, respectively, in comparison to our designs.

Summary of Design Flow Comparisons: The comparisons presented in the above paragraphs demonstrate the need for integrating system-level floorplanning in the NoC design flow. The results for $\mathrm{GA}_{2,5}$ and $\mathrm{GA}_{2,\infty}$ demonstrate that a floorplan agnostic NoC synthesis stage followed by a call to an existing floorplanner results in long link lengths that violate the link length constraint. Further, even in the absence of link length constraints our technique consistently generates superior solutions in comparison to $\mathrm{GA^{pre}}$ as it accounts for the link power consumption. On an average for legal solutions our approach consumes 36.83% lower power in comparison to an approach that does NoC synthesis before system-level floorplanning.

3) NoC Designs for Set-Top Box: We present NoC designs produced by the GA for the set-top box application (benchmark G5). We refer the reader to [18] for the CTG of the set-top box

TABLE VI
CORE DESCRIPTIONS FOR THE SET-TOP BOX

ID	Description	ID	Description	ID	Description
0	ASIC	9	DSP	18	CPU
1	DSP	10	CPU	19	MEM
2	DSP	11	ASIC	20	DSP
3	DSP	12	ASIC	21	DSP
4	ASIC	13	MEM	22	DSP
5	CPU	14	ASIC	23	MEM
6	MEM	15	DSP	24	ASIC
7	DSP	16	DSP		
8	DSP	17	DSP		

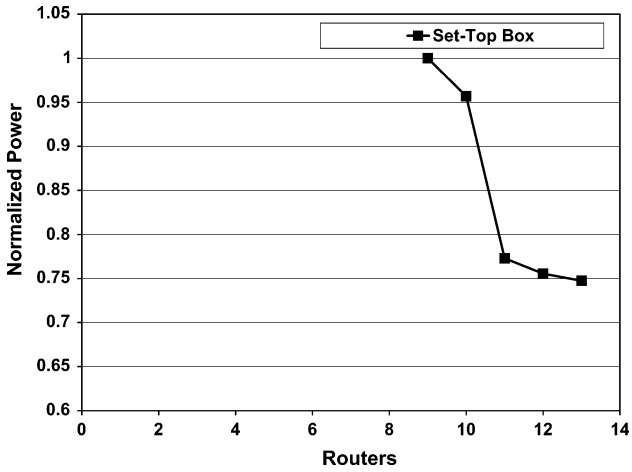

Fig. 12. Pareto curve for the set-top box application.

application. Table VI presents the description of the nodes in the CTG.

Fig. 12 depicts the Pareto curve for the benchmark. The X-axis of the figure denotes the number of router resources in the solution, and the Y-axis denotes the corresponding power consumption. The set-top box application generated five Pareto points corresponding to 9–13 routers. As can be observed from the Pareto curve for the set-top box, the best tradeoff between power consumption and the corresponding router resources required was offered by the solution with 11 routers. This particular design and observation can only be obtained by generating a Pareto curve, thus substantiating our approach.

Figs. 13 and 14 present the custom topologies for the designs with minimum power and least number of routers, respectively for the set-top box application produced by our GA-based technique with 5-port routers and a link length constraint of 2.5 mm. The black squares in the floorplan denote the routers. The solid grey lines on the floorplan represent the connectivity of the cores to the routers.

Fig. 15 plots the variation in power consumption in successive generations for the five Pareto points of the set-top box benchmark. The plot is normalized to the highest power consumption among all generations. The least power solution did not improve for the 9 and 10 router solutions. The 11 and 12 router solutions improved 3 times, and the 13 router solution improved twice. The 9 and 10 router solutions remained dominant over all generations. The 11, 12, and 13 router solutions became dominant in the 17th, 87th, and 84th generation, respectively. Fig. 16 plots the number of illegal solutions at the end of each generation. At

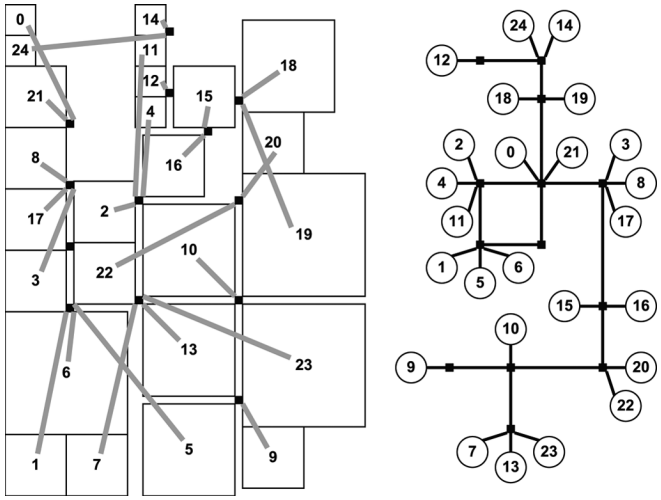

Fig. 13. NoC design for set-top box: least power solution.

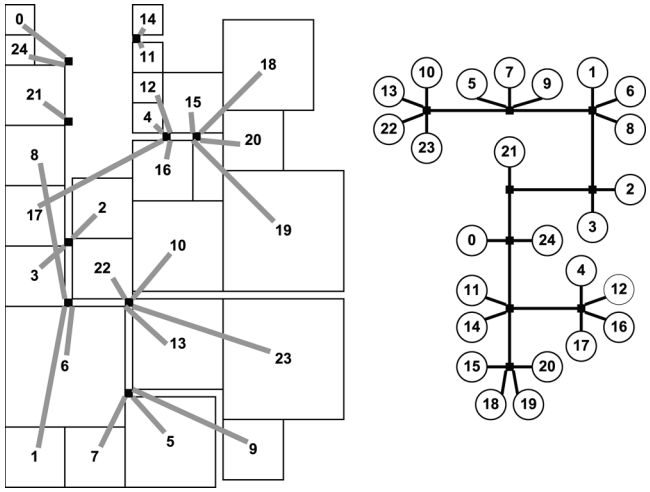

Fig. 14. NoC design for set-top box: least router solution.

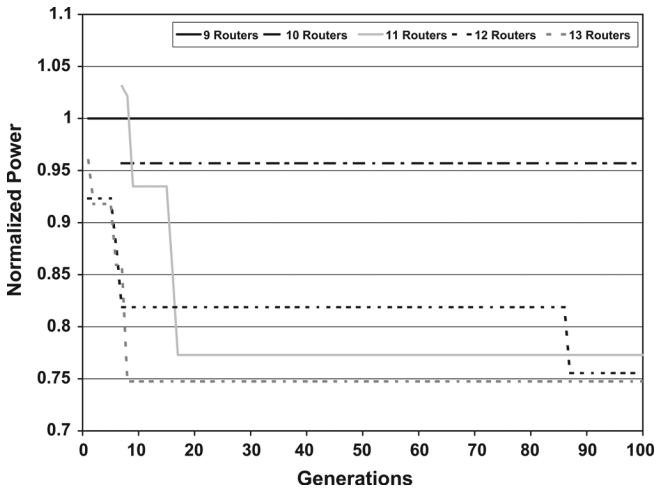

Fig. 15. Power consumption across generations.

the end of the first generation, there were several illegal solutions in the population. However, due to their low fitness value, illegal solutions were filtered away in successive generations.

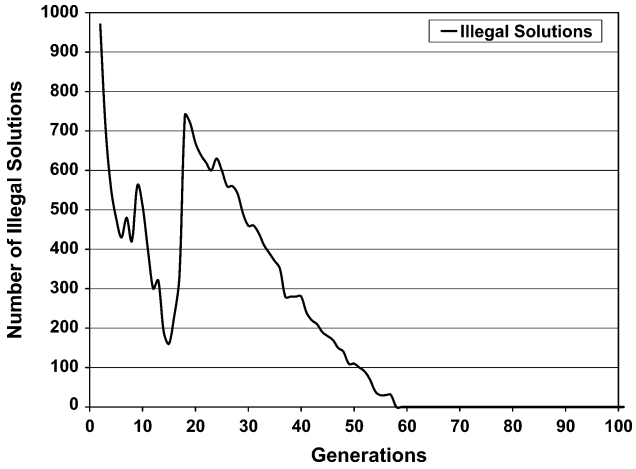

Fig. 16. Number of illegal solutions across generations.

As the number of generations evolved, the number of illegal solutions reduced to only a few in number.

IV. Conclusion

In this paper, we presented a novel GA-based technique for application specific custom NoC design. Our overall design flow consists of two phases namely, system-level floorplanning, and interconnection architecture generation. In the first phase, we invoke an existing flooplanner with an objective of minimizing a power-performance cost function. For the second phase, we presented a novel GA-based technique for application specific on-chip interconnection network generation. We compared our technique with an optimal ILP formulation [4], and an existing heuristic technique called ANOC [13]. While ILP and ANOC suffer from high runtime and low solution quality respectively, experimental results on several multimedia and network processing benchmarks demonstrate that our GA based technique is able to generate close to optimal solutions (within 3% for minimum power consumption solutions) in a reasonable time. Further, in comparison to an approach that synthesizes a NoC architecture with out the knowledge of a system-level floorplan our methodology can accept link length constraints, and our designs on an average consume over 36.83% lower power. Our future work will address integration of NoC design techniques with the computation architecture design stage.

References

[1] W. J. Dally and B. Towles, "Route packet, not wires: On-chip interconnection networks," in *Proc. DAC*, Jun. 2002, pp. 684–689.

[2] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70–78, Jan. 2002.

[3] ST Microelectronics, Geneva, Switzerland, "ST network on chip," 2005 [Online]. Available: http://www.st.com/stonline/press/news/back2005/b9014t.htm

[4] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Linear programming based techniques for synthesis of network-on-chip architectures," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 4, pp. 407–420, Apr. 2006.

[5] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt, III, "Approximation algorithms for degree-constrained minimum-cost network-design problems," *Algorithmica*, vol. 31, no. 1, pp. 58–78, 2001.

[6] L. Benini, "Application specific NoC design," in *Proc. DATE*, Mar. 2006, pp. 491–495.

[7] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli, "Efficient synthesis of networks on chip," in *Proc. ICCD*, 2003, pp. 146–150.

[8] A. Jalabert, S. Murali, L. Benini, and G. De Micheli, "Xpipescompiler: A tool for instantiating application specific networks on chip," in *Proc. DATE*, 2004, pp. 884–889.

[9] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Linear programming based techniques for synthesis of network-on-chip architectures," presented at the ICCD, San Jose, CA, Oct. 2004.

[10] K. Srinivasan and K. S. Chatha, "ISIS: A genetic algorithm based technique for synthesis of on-chip interconnection networks," presented at the VLSI Des., Calcutta, India, Jan. 2005.

[11] U. Ogras and R. Marculescu, "Energy and performance driven NoC communication architecture synthesis using a decomposition approach," in *Proc. DATE*, 2005, pp. 352–357.

[12] U. Ogras and R. Marculescu, "Application specific network-on-chip architecture customization via long range link insertion," in *Proc. ICCAD*, 2005, pp. 246–253.

[13] K. Srinivasan and K. S. Chatha, "A technique for design of application specific network-on-chip architectures," presented at the DATE, Munich, Germany, Mar. 2006.

[14] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks: An Engineering Approach*. Los Alamitos, CA: IEEE Computer Society, 1997.

[15] R. P. Dick and N. K. Jha, "MOGAC: A multiobjective genetic algorithm for hardware-software cosynthesis of distributed embedded systems," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 17, no. 10, pp. 920–935, Oct. 1998.

[16] W. J. Dally and C. L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. Computers*, vol. C-36, no. 5, pp. 547–553, May 1987.

[17] V. L. Lorenzo, "JPEG hardware compressor," OpenCores.org, Feb. 2008. [Online]. Available: http://www.opencores.org/projects.cgi/web/jpeg/overview

[18] J. Hu and R. Marculescu, "Energy-aware mapping for tile-based noc architectures under performance constraints," in *Proc. ASP-DAC*, 2003, pp. 233–239.

[19] V. Ramamurthi, J. McCollum, C. Ostler, and K. S. Chatha, "System-level methodology for programming CMP based multi-threaded network processor architectures," presented at the ISVLSI, Tampa, FL, May 2005.

[20] S. Pasricha, N. Dutt, E. Bozorgzadeh, and M. Ben-Romdhane, "FABSYN: Floorplan-aware bus architecture synthesis," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 3, pp. 241–253, Mar. 2006.

[21] M. Alho and J. Nurmi, "Implementation of interface router IP for proteo network-on-chip," presented at the Int. Workshop Des. Diagnost. Electron. Circuits Syst., Poznan, Poland, Apr. 2003.

[22] S. N. Adya and I. L. Markov, "Fixed outline floorplanning: Enabling hierarchical design," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 6, pp. 1120–1135, Dec. 2003.

[23] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, and N. Borkar, "A 80-tile 1.28tflops network-on-chip in 65 nm CMOS," in *Proc. Int. Solid-State Circuits Conf.*, San Francisco, CA, Feb. 2007, pp. 5–7.

Glenn Leary received the B.S. degree in computer systems engineering from Arizona State University, Tempe, in 2005, where he is currently pursuing the Ph.D. degree in computer science.

His research interests include the areas of network-on-chip (NoC) and system-on-chip (SoC) design. In particular, he has focused on the design of router architectures, and power and performance optimization of NoC and SoC architectures.

Krishnan Srinivasan received the B.S. degree in electrical engineering from the Regional Institute of Technology, Jamshedpur, India, in 1999, the M.S. degree in electrical engineering and the Ph.D. degree in computer science from Arizona State University, Tempe, in 2002 and 2006, respectively.

His research interests include the field of power and performance optimization of system-on-chip (SoC) and network-on-chip (NoC) architectures.

Krishna Mehta received the B.S. degree in computer engineering from the University of Mumbai, Mumbai, India, in 2005, and the M.S. degree in computer science and engineering from Arizona State University, Tempe, in 2007.

She is currently an Field-Programmable Gate Array (FPGA) Engineer with Freescale Inc., Tempe, AZ. Her research interests include system level design on FPGAs and network-on-chip (NoC) Architectures.

Karam S. Chatha (M'01) received the B.E. (with honors) degree in computer technology from Bombay University, Kalina, Mumbai, in 1993, and the M.S. and Ph.D. degrees in computer science and engineering from University of Cincinnati, Cincinnati, OH, in 1997 and 2001, respectively.

He is currently an Associate Professor with the Department of Computer Science and Engineering, Arizona State University, Tempe. His research interests are centered on topics related to application specific digital system design, including architectures, design methodologies, and computer-aided design (CAD) tools. In particular, he has focused on network-on-chip (NoC) design, multiprocessor system-on-chip (MPSoC) programming, hardware-software codesign, and reconfigurable and adaptive computing.

Dr. Chatha was a recipient of the NSF CAREER Award 2006, and Best Paper Awards from the International Conference of Computer-Aided Design (ICCAD) 2007, and the International Workshop on Field Programmable Logic (FPL) 1999. He is a member of the ACM.