An Automated Technique for Topology and Route Generation of Application Specific On-Chip Interconnection Networks

Krishnan Srinivasan, Karam S. Chatha, and Goran Konjevod
Department of CSE, PO BOX 875406, Arizona State University,
Tempe, AZ 85287-5406
Email: {ksrinivasan,kchatha,goran}@asu.edu

Abstract— **Network-on-chip (NoC)) has been proposed as a solution to the communication challenges of System-on-chip (SoC) design in nanoscale technologies. Application specific SoC design offers the opportunity for incorporating custom NoC architectures that are more suitable for a particular application, and do not necessarily conform to regular topologies. Custom NoC design in nanoscale technologies must address performance requirements, power consumption and physical layout considerations. This paper presents a novel three phase technique that i) generates a performance aware layout of the SoC, ii) maps the cores of the SoC to routers, and iii) generates a unique route for every trace that satisfies the performance and architectural constraints. We present an analysis of the quality of the results of the proposed technique by experimentation with realistic benchmarks.**

I. Introduction

On-chip interconnection networks or Networks-on-Chip (NoC) have been proposed as a solution for addressing the global communication challenges in System-on-Chip architectures that are implemented in nanoscale technologies [1] [2]. An example of the NoC architecture is shown in the right hand side of Figure 1. The figure depicts a physical layout of an example SoC architecture. The various square blocks with labels (P1, P2, and so on) denote processing or storage cores. The black filled boxes denote routers that are connected by physical links.

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. Regular topologies are suitable for general purpose architectures such as the MIT RAW [3] that include homogeneous cores. Application specific SoC architectures consist of heterogeneous cores and memory elements which have vastly different sizes. For such architectures, the custom NoC architecture has been demonstrated to be superior to regular architecture in terms of power, area and performance by Jalabert et. al. [4]. This paper concentrates on the design of custom NoC topologies that are optimized for the target application.

The design of custom NoC architectures poses several challenges to the designer. The interconnection architecture must provide sufficient bandwidth for low latency congestion free communication between the various units in the architecture. Power reduction has emerged as a first order design goal in nanoscale technologies. Hence, the designer must aim to minimize the power consumption for on-chip communication in the interconnection architecture. Due to technology scaling, the physical links consume upwards of 30 % of the total communication power. The power consumption in the physical links is linearly dependent on their length. Therefore, custom NoC design must include physical layout information.

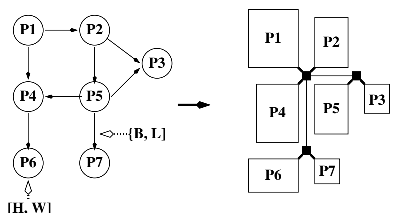

Fig. 1. Application Specific NoC Design Problem

The custom NoC design problem is depicted in Figure 1. The input to the problem is the system-level specification described by a directed communication trace graph in which the nodes represent various computation and storage elements, and edges denote communication between two units. The nodes include physical dimension information (height and width, H and W in the figure) about the processing and storage elements, and edges are annotated with bandwidth and latency requirements (B and L in the figure). The output of the custom NoC design stage (shown in the right side of the figure) is a physical layout aware topology and a unique route for every edge in the system-level specification that satisfies the performance requirements, and minimizes the power (primary goal) and area (secondary goal) of the interconnection architecture. This paper presents an automated technique for solving the custom NoC design problem.

Automated design of custom NoC architectures requires a characterized library of interconnection network building blocks. We characterized the power consumption of the unit router in 100 nm technology with the help of a cycle accurate power and performance evaluator [5]. In the interest of space, we have omitted the complete details of the experiments. The variation of power consumption with injection rate for the input and output ports of a router are shown in Figures 2

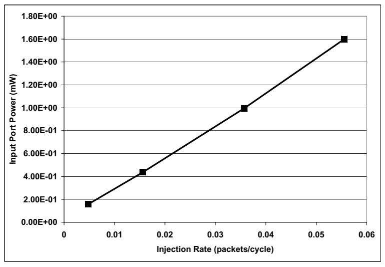

Fig. 2. Input port power consumption

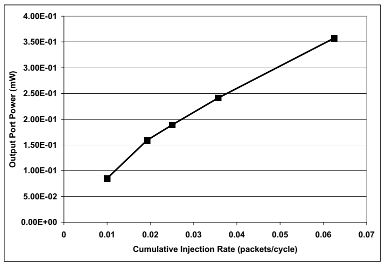

Fig. 3. Output port power consumption

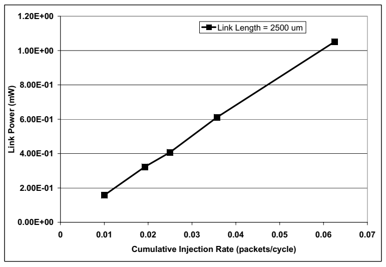

Fig. 4. Link power versus injection rate

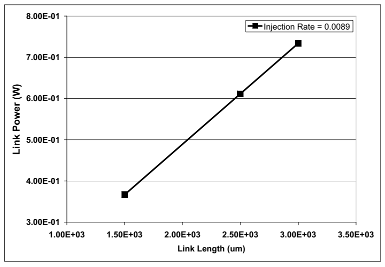

Fig. 5. Link power versus length

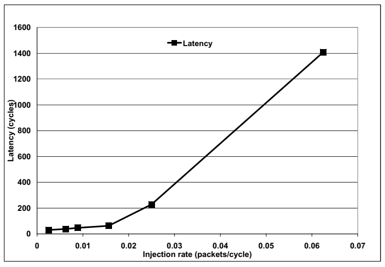

Fig. 6. Latency for 2 routers

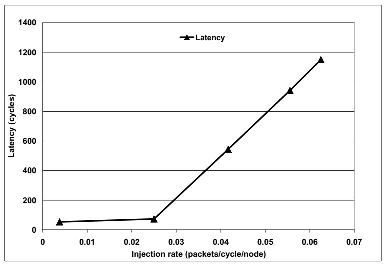

Fig. 7. Latency for 4x4 mesh

and 3, respectively. The power consumption of the input and output ports vary linearly with the injection rates, respectively. Quantitatively, we estimated the power consumption of the input port as $328nW/Mbps$, and $65.5nW/Mbps$ for the output port.

We studied the variation of link power with respect to the bandwidth flowing on the link, and its length. Figure 4 plots the variation of link power with the supported bandwidth. Figure 5 plots the variation of link power with its length. The power consumption of the physical links varies linearly with both the supported bandwidth, and the length of the link. We estimated the power consumption of the links to be equal to $79.9nW/Mbps/mm$.

The variation of average latency for data packets that travel over two router hops with respect to injection rate is shown in Figure 6. The average latency remains almost constant in the un-congested mode, and onset of congestion is marked by a sharp increase in latency. As shown in Figure 7, a similar trend is observed for average latencies of packets in a 4x4 mesh. Our technique prevents network congestion by static routing of the communication traces subject to the peak bandwidth constraint on the router ports. Since the network is always operated in the un-congested mode, we can represent the network latency constraint in terms of router hops (such as 1 or 2) instead of an absolute number (such as 100 cycles).

In the following section we define the NoC design problem.

A. Problem Definition

Given:

- A directed communication trace graph $G(V, E)$, 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.
- 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.
- A router architecture, where 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 $Mbps$ of traffic bandwidth flowing in the input and output direction, respectively for any port of the router.
- A physical link power model denoted by Ψ_l per $Mbps$ per mm.

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 system-level floorplan, and 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, and
- the total system-level power consumption for inter-core communication is minimized (primary goal), and number of router resources is minimized (secondary goal).

In the above problem formulation we assume that the maximum physical link length that permits single clock cycle data transfer between neighboring routers is denoted by the maximum dimension ($H(v)$ or $W(v)$) of a node in the system-level specification. This assumption is based on the fact that it is possible to perform intra-core single clock cycle data transfer from one corner of node to the any of the neighboring corners. We also consider that the dimensions of the routers are much lower than the sizes of the cores. This assumption is supported by the observation of Dally et al. [1] that the entire NoC places an area overhead of 6.6% on the SoC architecture. Therefore, we assume that the routers that are possibly utilized in the layout are located at corners of the cores. If $(X(v), Y(v))$ denotes the lower left hand side corner of the node, the core is mapped to one of the routers located at $(X(v), Y(v))$, $(X(v) + \mathcal{W}_v, Y(v))$, $(X(v), Y(v) + \mathcal{H}_v)$ or $(X(v) + \mathcal{W}_v, Y(v) + \mathcal{H}_v)$.

The power consumption of the NoC can be minimized by minimizing the cumulative traffic flowing through the ports of all routers. Traffic flowing in a network can be reduced by placing communicating cores close to each other. However, the close location of the communicating cores must be traded-off with the latency constraints.

The latency constraints imposed by the traffic traces specify the maximum number of hops allowed for the trace. Bandwidth requirements and latency constraints of communication traces can be viewed as mutually independent. A trace such as a signalling event or a cache miss is not expected to have high bandwidth requirement, but is bound by tight latency constraints. On the other hand, many non-critical multimedia streams have high bandwidth requirement, and their latency is bound only by the period constraint of the application [6]. A NoC design framework has to perform a trade-off between placing communicating cores with high bandwidth, and those with tight latency close to each other to minimize power, and to satisfy the performance constraints, respectively.

The NoC synthesis problem as described above is a variation of the generalized steiner forest problem that is known to be NP hard [7]. We present a three phase technique that i) generates a performance aware layout of the SoC, ii) maps the cores of the SoC to routers, and iii) generates a unique route for every trace that satisfies the performance and architectural constraints.

The paper is organized as follows: in Section II we discuss previous work, in Section III we present our technique, in Section IV we discuss our experimental results, and finally in Section V we conclude the paper.

II. Previous work

Many researchers [8] [9] [10] [11] have presented core mapping and routing techniques for mesh based NoC architectures. Recently, researchers have begun to address the problem of automated design of application specific NoC architectures. Pinto et al. [12] presented a technique for constraint driven communication architecture synthesis of point to point links by utilizing deterministic heuristic based k-way merging. Their technique results in network topologies that have only two routers between each source and sink. Hence, their problem formulation does not address routing. In [6], Murali et al.

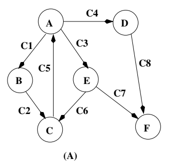

Fig. 8. CTG

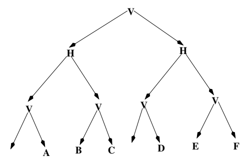

Fig. 9. Slicing Tree

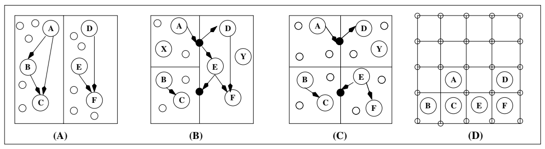

Fig. 10. Initial Floorplanner

presented a technique that integrates physical planning with quality of service. However, they do not address synthesis of custom NoC topologies. We on the other hand, synthesize an application specific custom topology optimized for the target application. Further, they propose a computationally complex solution for the problem, where they iteratively invoke an MILP based placement technique in a tabu search framework. In contrast, we present polynomial time algorithms for integrated floorplanning, and topology synthesis of application specific custom NoC architectures.

III. Synthesis of Custom NoC Architectures

In this section, we present our application specific on-chip interconnection architecture synthesis technique. Our technique operates in three phases. In the first phase, it invokes a performance aware slicing tree based floorplanner to obtain an initial physical layout of the nodes constituting the SoC. In the second phase, the technique invokes a linear programming (LP) based algorithm that maps the processing cores to different routers such that the power utilized for communication is minimized. Finally, in the third phase, it executes a LP based routing algorithm that generates routes for the traces such that the total number of routers utilized in the topology are minimized. In the following sections, we will discuss each phase in detail.

A. Initial Floorplanner

We utilize a slicing tree based initial floorplanner (IF) described in [13]. In this paper, we present an overview of the technique, and refer the reader to [13] for further details. Unlike [13], we do not add extra nodes. Figures 8, 9, and 10 give examples of an input CTG, slicing tree, and various stages of the algorithm execution, respectively. The slicing tree is formed by recursively dividing the layout area into vertical and horizontal sections. In Figure 9, the letter V denotes that the plane is divided into a left and right sub-plane by a vertical cut, and the letter H denotes that the plane is divided into top and bottom sub-planes by a horizontal cut.

IF invokes a graph equicut algorithm proposed by Fiduccia and Mattheyses (FM) [14] to generate the partitions. The partitioning technique assigns nodes to one of the sub-planes such that the total weight of the edges across the cut is minimized. IF assigns a weight to each edge as follows. Bandwidth constraints on the ports of routers can be satisfied by finding alternative (sometimes longer) route for the trace. Latency constraints on the other hand cannot be adhered to by finding alternative paths. Therefore, IF gives higher priority to latency compared to bandwidth. Let e_i be a trace with the highest bandwidth requirement among all traces in the graph. Let e_j be the trace with tightest (lowest) latency constraint among all traces in the graph. IF determines an integer k such that it is the minimum value required to ensure that $\frac{\omega(e_i)}{\sigma(e_i)^k} \leq \frac{\omega(e_j)}{\sigma(e_j)^k}$. Once k is determined, IF assigns an edge weight to each edge given by $\forall e \in E,\ \rho(e) = \frac{\omega(e)}{\sigma(e)^k}$. For two edges with the same edge weight, the one with tighter latency has higher priority. This heuristic ensures that traces with low bandwidth requirements, but with tight latency constraints are given priority over those with high bandwidth requirement and relaxed latency constraints.

After the floorplan is generated, our technique invokes a compaction algorithm that takes the actual sizes of the processing cores into account and generates a final layout. The compaction stage is required as we utilize a slicing tree based floorplanning algorithm. The slicing tree based heuristic assigns the nodes to bounding boxes at rectangular grid locations. As the size of the bounding box is typically larger than the size of the node, we require a compaction stage. The compaction algorithm first moves all nodes toward the center of the layout in the X direction, and then moves all nodes in the Y direction, again toward the center of the layout. The movement in X and Y directions is repeated until no further compaction is possible.

B. Core to router mapping technique

In this section, we present a linear programming based technique called CMT that maps each processing or storage core to one router that is located at the corners of the core. We present a lower bound on the optimal solution, and utilize a randomized rounding algorithm to arrive at the optimal (or near optimal) solution.

For node i, let R_i denote the set of routers to which i can be mapped. Let $X_{i,j}$ denote a (0,1) integer variable that is set to 1 if node i is mapped to router $j \in R_i$, else 0. Each node is mapped to one of the routers located at its four corners. Therefore, there are $4 * |V|$ variables of this type. For each edge (i,k) let $a_{i,j,k,l}$ denote the communication cost when node i is assigned to router j and node k is assigned to router l. The cost denotes the power consumed to perform this communication. Therefore, $a_{i,j,k,l} = \omega_{i,k} * dist_{j,l}$ where $\omega_{i,k}$ is the bandwidth requirement of the edge (i,k), and $dist_{j,l}$ is the Manhattan distance between routers j and l. Let vector $X[4 \cdot |V| : 1]$ denote the possible assignments of nodes to different routers, and matrix $A[4 \cdot |V| \times 4 \cdot |V|]$ (with elements $a_{i,j,k,l} \in A$) denote the costs of the various mappings. The problem is to obtain an assignment of the nodes such that the total communication cost is minimized. The problem is a special case of the quadratic assignment problem (QP) and can be expressed mathematically as follows.

$$minimize\ X^T AX$$

$$Subject\ to\ \forall i \in V, \sum_{j \in R_i} X_{i,j} = 1$$

where R_i denotes the set of routers that node i can be mapped to. The constraints enforce the requirement that each node is mapped to exactly one router. QP is among the hardest problems to solve in combinatorics. It is well known that the formulation presented above cannot be solved in polynomial time unless the matrix A is positive semi-definite. In other words, in order for the QP to be polynomial time solvable, all determinants of the principal submatrices of A should be non-negative [15]. In matrix A, the diagonal elements indicate the communication cost from a node to itself, and therefore, are zero. Hence, A is not positive semi-definite. One way to make the QP polynomial time solvable is to increase the cost of the diagonal elements of A by some value C to make the matrix positive semi-definite, solving the QP, and finally subtracting C obtain the final solution. But this can result in sub-optimal solutions for the QP [15]. The objective function of QP with constant C added to the diagonal elements is of the form

$$\sum_{(i,j) \in E} X_{i,j} \cdot X_{k,l} \cdot a_{i,j,k,l} + C \cdot \sum_{i \in V} \sum_{j} X_{i,j}^2$$

If C is large, the square terms dominate the objective function, and therefore, the minimizer would tend to assign values close to $\frac{1}{|R_i|}(= \frac{1}{4}$ for the stated problem) to the variables $X_{i,j}$. Hence, a randomized rounding scheme that assigns a variable $X_{i,j}$ to 1 with probability $X_{i,j}$, will assign node i to any of the four routers with almost the same probability, and may not perform well.

We are interested in the integer solution of the QP. Since even the continuous version (where variables can take fractional values) is hard to solve, we formulate the integer version as an integer linear program (ILP). An ILP in general is not polynomial time solvable unless $P = NP$. Therefore, we relax the integer constraints on the ILP, and solve the problem as an LP. An optimal solution to any linear program can be generated in time polynomial in the number of variables and constraints [16]. Noting that the integer versions of the QP and LP solve the same problem, we apply a randomized rounding technique on the QP by rounding a variable to 1 with a probability given by the value assigned to the variable by the LP.

1) ILP Formulation: In this section, we present our ILP formulation for the quadratic assignment problem. We give a unique number to each node. For each node i, let $R_i = \{r_{i0}, r_{i1}, r_{i2}, r_{i3}\}$ denote the set of routers to which i can be mapped.

<u>*Variables*</u> We define the following variables.

- Let $X_{i,j}$ denote a (0,1) integer variable that is set to 1 if node i is mapped to router $j \in R_i$, else 0. There are $4 * |V|$ variables of this type in the formulation.
- Let $X_{i,j,k,l}$ denote a (0,1) integer variable that is set to 1 if node i is mapped to router j, and node k is mapped to router l, else 0. We define these variables only when

$(i,k) \in E$ or $(k,i) \in E$. Hence, there are $16 * |E|$ variables of this type.

Objective Function The objective is to minimize the total communication cost. It can be expressed as follows.

$$Minimize\ Z = \sum_{(i,k)\in E} \sum_{j \in R_i} \sum_{l \in R_k} a_{i,j,k,l} * X_{i,j,k,l}$$

where $a_{i,j,k,l}$ is the product of the bandwidth of the traffic for edge (i,k), and the Manhattan distance between routers j and l. We can represent $a_{i,j,k,l}$ as

$$a_{i,j,k,l} = \omega_{i,k} * dist_{j,l}$$

where $\omega_{i,k}$ is the bandwidth requirement of the edge (i,k), and $dist_{j,l}$ is the Manhattan distance between routers j and l.

Note that the objective function is defined only for node pairs that have an edge between them.

Constraints

- Each node should be mapped to exactly one router. This constraint can be modeled as follows.

$$\forall i \in V, \sum_{j \in R_i} X_{i,j} = 1$$

- The variable $X_{i,j,k,l}$ represents node i mapped to router j, and node k mapped to router l. Therefore, if node i is mapped to router j, all communication should take place through that router. This condition is represented by the following two equations.

$$\forall (i,k) \in E, \forall j \in R_i, \sum_{l \in R_k} X_{i,j,k,l} = X_{i,j}$$

$$\forall (i,k) \in E, \forall l \in R_k, \sum_{j \in R_i} X_{i,j,k,l} = X_{k,l}$$

C. Discussion

We show with the help of lemmas that a randomized rounding technique can be successfully applied to obtain near optimal solutions.

Lemma 1: The optimal solutions of the integer versions LP and QP have the same cost.

Proof: Let us denote the integer versions of the problems as ILP and IQP, respectively. We note that the ILP and IQP solve the same problem, and therefore, the cost of their optimal solutions will be the same.

Lemma 2: The expectation of the integer solution generated by randomized rounding of the variables of the QP with probability equal to the value of the variable in the solution is equal to the cost of solution generated by QP.

Proof: Let the cost of the solution of QP be denoted as Z_{QP}, and the cost of the integer version of QP be denoted as Z_{IQP}. Consider an experiment of picking one router j among the four routers placed in the corners of node v_i, with a probability $X_{i,j}$. For an edge $(i,j) \in E$, the cost of mapping nodes i and j is given by

$$E[Z_{IQP}^{i,j}] = \sum_{j} \sum_{l} (P(X_{i,j} = 1) \bigcap P(X_{kl} = 1) \cdot a_{i,j,k,l}$$

The overall expected cost of the solutions is given by

$$E[Z_{IQP}] = \sum_{i,k \in E} \sum_{j} \sum_{l} (P(X_{ij} = 1) \bigcap P(X_{kl} = 1) \cdot a_{i,j,k,l}$$

Noting that the probabilities are independent and $X_{i,j}$ is set to 1 with probability $X_{i,j}$, we get

$$E[Z_{IQP}] = \sum_{i,k \in E} \sum_{j} \sum_{l} X_{i,j} \cdot X_{k,l} \cdot a_{i,j,k,l} = Z_{QP}$$

The lemma proves that there is some feasible assignment of the nodes to routers such that the cost of the solution and that of the infeasible optimal solution obtained by relaxing the integer constraints is the same. Let this optimal feasible solution be represented by Z_{IQP}^*. Let the cost of the LP be denoted as Z_{LP}, and the cost of the ILP be denoted as Z_{ILP}. From the above argument, it follows that

$$Z_{LP} \leq Z_{QP} = Z_{IQP}^* = Z_{ILP}^*$$

Lemma 3: Let Z_{CQP} represent the cost of the solution obtained by adding a constant C to the diagonals of matrix A to make it positive semi-definite. Then, $Z_{QP} \geq Z_{CQP} - |V| \cdot C$.

Proof: From Lemma 2,

$$Z_{IQP}^* = Z_{QP}$$

We also know that

$$Z_{ICQP}^* \geq Z_{CQP}$$

In the optimal integer solution, one and only one $X_{i,j}$ per node i is set to one. The remaining variables are set to zero. Therefore, Z_{ICQP} is minimized only when the first part of the sum in the objective function is minimized. But the first part represents Z_{IQP}. Therefore,

$$Z_{ICQP}^* = Z_{IQP}^* + C \cdot |V|$$

Now, since $Z_{ICQP}^* \geq Z_{CQP}$, and $Z_{IQP}^* = Z_{QP}$, $Z_{QP} \geq Z_{CQP} - |V| \cdot C$.

Therefore, we can solve for Z_{CQP} in polynomial time, and obtain a lower bound on Z_{QP}. From Lemma 2, this gives us a lower bound on the expected value of the integer solution. To obtain an integer solution with a cost given by the lower bound, we utilize the randomized rounding technique.

1) Randomized rounding: Based on the LP formulation, we present a randomized rounding algorithm using Z_{QP} as a lower bound. The algorithm iteratively assigns routers to nodes with probability given by the value of the corresponding variable $(X_{i,j})$, until the number of iterations is maximum, or an optimal solution is found. Once the exit criterion is satisfied, the algorithm returns the best solution found thus far.

2) Merging routers: At the end of the floorplanning and mapping phases, the architecture may have routers that are placed very close to each other. In order to eliminate redundant routers and also to reduce the complexity of the routing stage, we merge pairs of routers that are less than a certain distance apart. The distance is specified by the designer, and can be set to the maximum permitted link length for single clock cycle data transfer. Our merging algorithm checks all pairs of available routers and merges two routers if

- the distance between them is less than the maximum allowable distance under single clock cycle data transfer, and
- merging the two routers does not cause a violation of the single clock cycle data transfer for any other router.

D. Routing Technique

In this section, we present RT, a linear programming based algorithm for routing communication traces such that the total number of routers utilized in the topology is minimized. The problem is a variation of the rectilinear steiner arborescence problem which is known to be NP-Complete [17]. Our heuristic models the problem as an LP formulation, and employs a randomized rounding technique to arrive at the final solution. In the following paragraphs, we discuss our technique in detail.

Variables: We define the following variables.

- Let $X_{i,j}$ denote a (0 1) variable that is set to 1 if the router at location (i,j) is selected for routing.
- Let $X_{i,j,k}$ denote a (0 1) variable that is set to 1 if edge k utilizes router at location (i,j) for routing. The number of variables of this type is equal to the product of the number of edges and the number of $X_{i,j}$ variables.

Objective: The objective is to minimize the number of routers. The objective can be expressed as

$$Minimize\ Z = \sum X_{i,j}$$

Constraints: For each edge in the CTG, consider a bounding box on the layout defined by the location of the source and sink nodes. The bounding box for the communication trace specifies the routers that can be utilized for routing with the shortest Manhattan path length. For edge e, let the bounding box be denoted as B_e.

- An edge in CTG always passes through the source and sink routers. For edge e, let n denote the source node, and m denote the sink node. Let the location of the router that connects to n be (i,j), and the location of the router that connects with m be (k,l). The following two equalities must hold.

$$X_{i,j,e} = 1, \ \ X_{k,l,e} = 1$$

- If a router at location (i,j) is utilized to route an edge e, at least one of its adjacent routers that is closer to the sink node, should also be utilized in the route. Assuming that router $X_{i,j}$ is considered, and locations that take the router close to sink are $X_{i,j+1,e}$, and $X_{i+1,j,e}$, we need the following inequality.

$$\forall i,j \in B_e, X_{i,j+1,e} + X_{i+1,j,e} - X_{i,j,e} \geq 0$$

```
**SPR** (tbd_trace_list)
    for $t \in$ tbd_trace_list
        for $e \in L$ /* For all physical links in $I$ */
            if $(\omega(e) + \omega(t) > \Omega)$ /* BW violation */
                edge_weight(e) = $\infty$ else edge_weight(e) = 1
            end if
        end for
        shortest_path($t, I, \mathcal{R}$)
    end for
end
```

Fig. 11. Shortest path router

Benchmark	Nodes	Edges	Power (μW)	Routers
dsp	6	5	1686	2
263 encoder	7	7	172.6	2
mp3 encoder	8	8	6.48	3
mpeg4	12	13	7392.18	5
mwd	12	13	993.6	3
vopd	12	13	2611.1	5
mp3 enc mp3 dec	13	12	9.15	4
263 dec mp3 dec	14	12	11.98	5
263 enc mp3 enc	15	17	181.6	4
263 enc 263 dec	16	17	159.5	5

TABLE I
RESULTS

- If a router is utilized to route an edge, it should be present in the final solution. Therefore,

$$\forall k, X_{i,j} \geq X_{i,j,k}$$

The objective function makes sure that if a router is not utilized in routing any edge, the corresponding $X_{i,j}$ will be set to zero.

1) Randomized rounding technique: The randomized rounding technique operates as follows. Initially, we solve the LP, and fix all $X_{i,j}$ variables that are assigned a value 1. Among the variables that have fractional values, we randomly pick a variable, and assign it to 1 with a probability given by the fractional value of the variable. The LP is solved again and the randomized rounding step is repeated until all variables are either set to 0 or 1.

At the end of the routing phase, there might be links that violate bandwidth constraints. We un-map the traces with minimum bandwidth requirement from these links, and re-route them by invoking Dijkstra's shortest path algorithm. The re-routing technique is described in the following paragraph.

2) Shortest path router: The shortest path router (SPR) is called for each traffic trace that is unmapped at the end of RT phase. SPR attempts to find alternate routes for these traces. For each trace in tbd_list, SPR sweeps all possible links L of the physical layout grid. It assigns an edge weight of ∞ to all links that would see a bandwidth violation on the ports constituting the links, if the trace was routed through that link. These links are not utilized to generate the route for the trace. This step is followed by calling Dijkstra's shortest path algorithm to find a route for the trace on the mesh.

IV. RESULTS

We present and analyze the experimental results obtained by execution of our technique on representative multimedia applications. We first discuss the benchmark applications, the

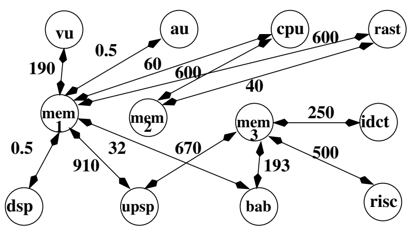

Fig. 12. MPEG 4 decoder CTG

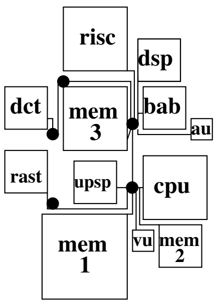

Fig. 13. Floorplan and NoC architecture

experimental setup, and finally, we present and discuss the results.

A. Benchmark applications

We generated custom NoC architectures for six combinations of four multimedia benchmarks namely, mp3 audio encoder, mp3 audio decoder, H.263 video encoder, and H.263 video decoder. The applications were obtained from the work presented by Hu et al. [11]. In addition, we obtained results for four other benchmarks namely, mpeg4 decoder, video object plane decoder (vopd), multi-window display (mwd), and DSP filter application (dsp). The mpeg4 decoder, vopd, and mwd applications were obtained from [4], and the dsp application was obtained from [9].

B. Experimental setup

We estimated the power consumption for the input and output traffic of a port in 100 nm technology to be $328nW/Mbps$ and $65.5nW/Mbps$, respectively. We estimated the link power consumption to be equal to $79.9nW/Mbps/mm$. All results were obtained on a 950 MHz dual sparc processor. We utilized the XPRESS-MP solver [18] to generate our LP solutions.

C. Results and discussion

The LP formulations of CMT technique generated integer solutions for all benchmarks. Since the solution generated by LP is less than or equal to the that generated by the integer version, we conclude that the integer solutions generated by our CMT formulations are optimal. The solutions of the RT formulation required at most 3 iterations of rounding to generate the final design.

The results are presented in Table I. In the table, the first column describes the benchmark application, the second and third columns present the size of the benchmark in terms of nodes and edges respectively, the fourth column presents the power consumption of the NoC, and the fifth column presents the number of routers in the solution. An LP formulation can be solved in polynomial time, and in all our test cases, the LP generated results in fraction of a second.

The communication trace graph for MPEG4 decoder is shown in Figure 12. In the graph, the nodes denote processing cores, and the edges are annotated by bandwidth requirement in Mbps. Figure 13 shows the physical layout and NoC architecture for the MPEG4 decoder application.

V. Conclusion

In this paper, we proposed a novel three phase automated floorplanning and synthesis technique for generation of application specific custom on-chip interconnection architectures. Our technique utilizes a low complexity slicing tree based floorplanner, and linear programming based techniques for core to router mapping and routing of communication traces. Our linear programming based techniques are able to generate optimal results for node to router mapping stage. We demonstrated that the complexity of our techniques is low, and as stated in the results section, the techniques are able to generate solutions in less than a second.

Acknowledgement

The research presented in this paper was supported in part by a grant from the National Science Foundation (IIS-0308268) and Consortium of Embedded Systems

References

[1] W. J. Dally and B. Towles. ”Route Packet, Not Wires: On-Chip Interconnection Networks”. In *Proceedings of DAC*, June 2002.
[2] L. Benini and G. De Micheli. ”Networks on Chips: A New SoC Paradigm”. *IEEE Computer*, pages 70–78, January 2002.
[3] M.B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, J-W Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen M. Frank, S. Amarasinghe, and A. Agrawal. ”The RAW Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs”. *IEEE Micro*, pages 25–35, March-April 2002.
[4] A. Jalabert, S. Murali, L. Benini, and G. De Micheli. ”xpipesCompiler: A tool for instantiating application specific Networks on Chip”. In *DATE*, 2004.
[5] N. Banerjee, P. Vellanki, and K. S. Chatha . “A Power and Performance Model for Network-on-Chip Architectures ”. In *Proceedings of DATE*, Paris, France, February 2004.
[6] S. Murali, L. Benini, and G. De Micheli. ”Mapping and Physical Planning of Networks-on-Chip Architectures with Quality-of-Service Guarantees ”. In *Proceedings of ASPDAC*, 2005.
[7] R. Ravi, M. V. Marathe, S.S. Ravi, D. J. Rosenkrantz, H. B. Hunt III . “Approximation Algorithms for Degree-Constrained Minimum-Cost Network-Design Problems” . *Algorithmica*, 31(1):58–78, 2001.
[8] G. Ascia, V. Catania, and M. Palesi. ”Multi-objective Mapping for Mesh-based NoC Architectures”. In *Proceedings of ISSS-CODES*, 2004.
[9] S. Murali, and G. De Micheli. ”Bandwidth-Constrained Mapping of Cores onto NoC Architectures”. In *DATE*, 2004.
[10] J. Hu, and Radu Marculescu. ”Exploiting the Routing Flexibility for Energy/Performance Aware Mapping of Regular NoC Architectures”. In *DATE*, 2003.
[11] J. Hu and R. Marculescu. ”Energy-Aware Mapping for Tile-based NoC Architectures Under Performance Constraints”. In *ASP-DAC*, 2003.
[12] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli. ”Efficient Synthesis of Networks On Chip”. In *ICCD*, 2003.
[13] Krishnan Srinivasan, and Karam S. Chatha. ”A Technique for Low Energy Mapping and Routing in Network-on-Chip Architectures”. In *ISLPED*, 2005.
[14] C.M Fiduccia and R.M Mattheyses. ”A Linear-Time Heuristic for Improving Network Partitions ”. In *Proceedings of DAC*, 1982.
[15] M. Skutella. “Convex Programming and Semidefinite Programming Relaxations in Scheduling”. *Journal of the ACM*, 48:206–242, 2001.
[16] Martin Grötschel, Laszlo Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer Verlag, 1988.
[17] W. Shi, and C. Su. “The Rectilinear Steiner Arborescence Problem is NP-Complete”. In *Proceedings of SODA*, 2000.
[18] . “www.dashoptimization.com” . 2004.