Communication Latency Aware Low Power NoC Synthesis

Yuanfang Hu, Yi Zhu
Dept. of Computer Science and Engineering
Univ. of California, San Diego
{yhu,y2zhu}@cs.ucsd.edu

Hongyu Chen
Synopsys, Inc.
honchen@synopsys.com

Ronald Graham
Chung-Kuan Cheng
Dept. of Computer Science and Engineering
Univ. of California, San Diego
{rgraham,kuan}@cs.ucsd.edu

ABSTRACT

Communication latency and power consumption are two competing objectives in Network-on-Chip (NoC) design. This paper proposes a novel method that unifies these two objectives in a multi-commodity flow (MCF) formulation. With an improved fully polynomial approximation algorithm, power efficient design of an 8×8 NoC can be found for given average latency constraints with certain communication bandwidth requirements. Experimental results suggest that (1) compared with mesh, torus and hypercube topologies, the optimized design can improve power latency product by up to 52.1%, 29.4% and 35.6%, respectively. (2) by sacrificing 2% latency, power consumption of the optimized design can be improved by up to 19.4%, which indicates the importance of power and latency co-optimization in NoC design.

Categories and Subject Descriptors:
B.4.3 [Hardware]: Interconnections (Subsystems)

General Terms: Algorithms, Design

Keywords: Network-on-Chip, Latency, Power, Topology

1. INTRODUCTION

Network-on-Chip (NoC) has been proposed[9][2] as an attractive alternative to traditional dedicated wires to achieve high performance and modularity. With the advance of semiconductor technology, we have observed that more IP blocks, such as processors, memory subsystems and DSPs, are integrated on a single chip and interconnected by NoC [4].

Power efficiency and communication latency are two main concerns in NoC design. On one hand, we hope switch packets be delivered to destinations within the shortest period. On the other hand, power consumption has become one of the first order design considerations of the nano-scale VLSI designs. Our work studies the tradeoffs between NoC power efficiency and its average communication latency.

We focus on two design choices, NoC topology selection and interconnect wire style optimization, to optimize NoC power consumption and latency simultaneously. At the same time, the design satisfies given constraints, such as communication bandwidth requirements and physical on-chip area resource constraints.

In recent years, researchers have studied NoC design issues to improve latency and power consumption. In [4], Hu *et al.* proposed a branch and bound algorithm to map the processing cores onto a tile-based mesh NoC architecture to satisfy bandwidth constraints and minimize total energy consumption. In [11], a variety of NoC topologies are designed and the effect of topology on NoC power consumption is studied. In [7], Ogras et *al.* inserted a few application-specific long-range links to regular mesh based topology to reduce average packet latency. In [5], Hu et *al.* proposed a multi-commodity flow (MCF) based scheme to optimize NoC power consumption.

However, the previous works did not consider power and latency as the design objectives simultaneously and study their relations. In this paper, we propose a design methodology that selects NoC topologies and the corresponding interconnect wire styles, so that power consumption is minimized subject to average communication latency constraints. Our main contributions are as follows:

- We incorporate the latency constraints and power minimization objectives into a unified multi-commodity flow (MCF) model. For any given average point-to-point communication latency requirement, our algorithm finds the optimal NoC topology from a given topology library. Experiments show that (1) compared with mesh, torus and hypercube topologies, the optimized design can improve power latency product by up to 52.1%, 29.4% and 35.6%, respectively. (2) by sacrificing 2% latency, power consumption of optimized design can be improved by up to 19.4%. Carefully balancing between NoC power efficiency and communication latency is important in NoC design.

- We implement the MCF solver using approximation algorithms, which are significantly faster than the commercial LP solver CPLEX. We further optimize the solver by applying the interval estimation technique. Experiments show that this heuristic optimization can improve the convergence time by more than 300 times in NoC of size 7x7.

The paper is structured as follows. Section 2 formulates the latency aware low power NoC synthesis problem. Section 3 and 4 describe the design flow and an improved MCF

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DAC 2006, July 24–28, 2006, San Francisco, California, USA.
Copyright 2006 ACM 1-59593-381-6/06/0007 ...$5.00.

algorithm. In Section 5 we give the experimental results. We conclude in Section 6.

2. PROBLEM STATEMENT

We assume a regular tile based NoC with $n \times n$ tiles. Each tile consists of a *processing core* and a *router*. Each tile is regarded as a grid with certain area and dimension. Each network link can be implemented with multiple wire styles, which have varying properties in terms of their area usage, power efficiency and signaling latency. When routing data packets to adjacent tiles, routing mechanism may dispatch packets to different types of wires according to their wire capacities. Since wires are routed on top of the grids, the total wiring area across a grid cannot exceed its grid area.

We define a NoC topology as a directed graph $G = (V, E)$, where each node $v_i \in V$ represents a tile, and each edge $e_{i,j} \in E$ represents a point-to-point interconnection between tile i and tile j.

An implementation of a NoC topology is a mapping from each edge to a particular wire style, $T(e) : E \rightarrow S$, and a mapping from each edge to the amount of wiring resources assigned to that edge, $C(e) : E \rightarrow R^+$. For a link with certain length, we are provided a library of interconnect wire styles $\{S|S_i = (P_i, A_i, D_i)\}$. A particular wire style S_i has three properties associated with it: the per bit power consumption P_i, the per edge routing area usage A_i, and the delay D_i.

We formulate our problem as the following *communication latency aware low power NoC synthesis problem*:

Latency aware low power NoC synthesis problem: We have an $n \times n$ array of tiles, a library of interconnect wire components of certain lengths:

Input: *The communication demand matrix between each pair of tiles*

Output: *The power-efficient NoC topology $G = (V, E)$, and its physical implementation $T(e), C(e), \forall e \in E$*

Constraints: *(1)The global average latency requirement is satisfied; (2)The horizontal and verticle wiring dimension can not exceed the grid dimension.*

3. DESIGN FLOW

As shown in Fig.1, first we generate a topology library. Then, based on the power and delay libraries, we use a MCF model to evaluate latency constrained NoC power consumption for each topology in the topology library, and select the most power-efficient topology.

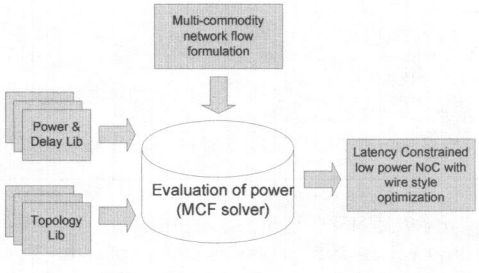

Figure 1: Design Flow

We use the method proposed in [5] to generate NoC topology library. For an $n \times n$ NoC, we generate the topologies (with max node degree of 3) on n nodes and enumerate their linear placements on a row/column. We set an upper bound for total wire length of linear placements, and map a topology to only those placements whose total wire length is no more than the threshold times the minimum wire length of that topology. Then, we duplicate the configuration to all rows/columns to get the candidate $n \times n$ NoC topologies.

In the following subsections, we describe our MCF model, as well as the power and delay models.

3.1 Latency Constrained Minimum Power MCF Formulation

For a given NoC topology graph $G = (V, E)$, we construct a flow graph as shown in Fig. 2. For each link between any two nodes, take link between v_m and v_n as example, right side of Fig. 2 shows the wire style details. There are t edges from node v_m to node v_n, where t is the number of candidate wire styles of the link. Associated with each edge, we have its wire style (P_e, A_e, D_e), as described in Section 2.

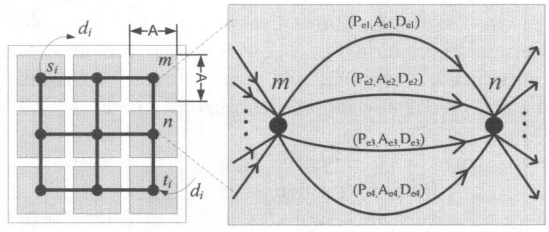

Figure 2: Flow Graph with Wire Style Optimization

Assume there are k commodities, for each commodity i, we are given a demand $d_i > 0$ between source node s_i and destination node t_i. For each vertical or horizontal dimension $Grid(q)$, we are given a grid dimension constraint A so that the sum of all the wire width on this grid is no more than A. Let p_i be the set of paths between s_i and t_i, and let $\mathbf{p} := \cup_i p_i$, variable $f(p)$ denotes the amount of flow sent along path p, for every $p \in \mathbf{p}$. Assume the overall latency is required to be bounded by LT, the MCF formulation is:

$$Min: \quad \sum_{j=1}^{k} \sum_{p \in p_j} \sum_{e \in p} f(p) \cdot P_e \tag{1}$$

$$s.t. \quad \sum_{j=1}^{k} \sum_{p \in p_j} \sum_{e \in p} f(p) \cdot D_e \leq LT \tag{2}$$

$$\forall 1 \leq j \leq k : \sum_{p \in p_j} f(p) \geq d_j \tag{3}$$

$$\forall q : \sum_{e \in Grid(q)} A_e \cdot \sum_{p: e \in p} f(p) \leq A \tag{4}$$

$$\forall p : f(p) \geq 0 \tag{5}$$

The objective is to minimize NoC power consumption, which is the sum of all the communication flows over per-bit power consumption of all the edges (as in Equation (1)). In constraint (2), we ensure that global latency requirement is satisfied. In constraint (3), we ensure that communication demand for each sender/receiver pair is satisfied. Constraint (4) states that the total routing channel dimension is limited by budget A on every grid dimension $Grid(q)$ of the routing channel.

3.2 Power and Delay Models

Interconnects and network routers are two main contributors to NoC power consumption and communication latency. We adopt the concept of *bit energy* proposed in [12] to represent energy when one bit of data is transported through the interconnects or routers. For each network link e, we assume that P_e represents bit energy on link e and the corresponding router, and D_e represents delay on link e and the corresponding router.

$$P_e = P_w + P_r; D_e = D_w + D_r$$

where P_w and P_r are bit energy on interconnects and routers, D_w and D_r are delay of unit flow on interconnects and routers, respectively. When a flow of amount f passes the link and the corresponding router, the total power consumption is: $P = P_e * f$, and total latency is: $D = D_e * f$.

3.2.1 Interconnect Wires

Wire bit energy P_w and delay D_w are closely related to the wire styles. We assume four types of wire styles are available for interconnects, namely, *RC* wires with repeated buffers with wire pitch varying from 1×, 2×, and 4× minimum global wire pitch, and on-chip transmission line with wire pitch equal to $16um$.

For RC wires with repeated buffers, we assume P_w and D_w are proportional to wirelength, i.e. P_w = per grid length big energy * wire length, and D_w = per grid length delay * wire length. For on-chip transmission line, relatively large setup costs should be added to D_w and P_w, respectively. We use transmission line model proposed by Chen et *al.* [1] to estimate transmission line bit energy and delay.

Table 1 lists bit energy and delay per grid length (2mm) of these four types of wire styles in $0.18um$ design technology. The supply voltages, wire geometries and device parameters are from ITRS [13]. For RC wires with repeated buffers, the repeaters are inserted to minimize wire delay. Setup costs of 50ps and 4.4pJ/bit are added to D_w and P_w for transmission line.

Table 1: Delay Model of Wires

wire type	RC-1x	RC-2x	RC-4x	T-line
P_w (pJ/bit)	2.68	2.15	1.99	0.15
D_w (ns)	0.127	0.112	0.100	0.020

Table 2: Power Model of Routers

ports	2	3	4	5	6	7	8
P_r (pJ/bit)	0.22	0.33	0.44	0.55	0.66	0.78	0.90
D_r (ns)	0.599	0.662	0.709	0.756	0.788	0.819	0.835

3.2.2 Network Routers

To estimate router bit energy P_r, we use a power simulator called *Orion* [10]. We assume 1GHz frequency, 4-flit buffer size, 128-bit flit size. We use the router delay model proposed by Peh et *al.* [8] to estimate NoC router delay.

Table 2 shows bit energy and latency of routers in $0.18um$ technology node. When router input/output ports increase, P_r increases almost linearly, and D_r increases in a slower pace.

4. MCF APPROXIMATION ALGORITHM

We use *polynomial time approximation schemes(PTAS)*, which can obtain $(1+\epsilon)$ optimal solutions in polynomial time, where ϵ is an input accuracy threshold. The PTAS for classic maximum concurrent flow problem has been studied in recent works [3] and [6]. We propose an algorithm which is based on the previous framework but adaptive to our own NoC synthesis problem. In addition, we propose the interval estimation technique to speed up the process.

4.1 Baseline Algorithm

First, our baseline algorithm finds the largest λ such that there is a multicommodity flow which routes at least λd_i units of commodity i, where the wiring area for each grid does not exceed the grid area A, and total power and latency are constrained by the budget PW and LT respectively. The problem formulaiton is as follows.

$$\begin{aligned}
&Primal: \\
&Max: && \lambda \\
&\forall j: && \textstyle\sum_{p\in p_j} f(p) \geq \lambda d_j \\
&\forall q: && \textstyle\sum_{e\in Grid(q)} A_e \sum_{p:e\in p} f(p) \leq A \\
& && \textstyle\sum_{i=1}^{k} \sum_{p\in p_i} \sum_{e\in p} f(p) P_e \leq PW \\
& && \textstyle\sum_{i=1}^{k} \sum_{p\in p_i} \sum_{e\in p} f(p) D_e \leq LT \\
&\forall p: && f(p) \geq 0
\end{aligned}$$

The following is the dual problem. Besides a variable X_e for each grid area constraint and a variable Z_j for every commodity demand constraint, the dual problem has another two variables, ϕ_p corresponding to the power budget constraint, and ϕ_d corresponding to the latency budget constraint:

$$\begin{aligned}
&Dual: \\
&Min: && A\textstyle\sum_{q=1}^{n} X_q + PW\phi_p + LT\phi_d \\
&\forall j, \forall P \in \rho: && \textstyle\sum_{e\in P} A_e \sum_{e\in Grid(q)} X_q + \sum_{e\in P} P_e\phi_p \\
& && + \textstyle\sum_{e\in P} D_e\phi_d \geq Z_j \\
& && \textstyle\sum_{j=1}^{k} d_j Z_j \geq 1 \\
&\forall q: && X_q \geq 0 \\
&\forall j: && Z_j \geq 0
\end{aligned}$$

Assume the subroutine $mcf(G, d, LT, PW)$ could return such a λ, the algorithm finds the minimum power that satisfying $\lambda \geq 1$ by binary search, as shown in Algorithm 1, where we use λ_{max} to denote the concurrent value without power budget constraint, i.e. $PW = \infty$.

Algorithm 1 Power Minimization MCF Algorithm

```
1: Input: graph G, demand d, latency constraint LT, threshold ε
2: Output: (1 + ε) optimal power
3: set λ_max ← mcf(G, d, LT, ∞)
4: set lower bound lb ← 0
5: upper bound ub ← total power under λ_max
6: while (ub − lb)/ub > ε do
7:    λ ← mcf(G, d, LT, (lb + ub)/2)
8:    if λ ≥ 1 then
9:       ub ← (lb + ub)/2
10:      else lb ← (lb + ub)/2
11:   end if
12: end while
13: Output ub
```

$mcf(G, d, LT, PW)$ subroutine iteratively updates the primal and dual values till the gap is small enough. The primal value λ is updated by adjusting the flows. To calculate dual values, we define edge length as:

$$l(e) := A_e \sum_{q:e\in Grid(q)} X_q + PW \cdot \phi_p + LT \cdot \phi_d \qquad (6)$$

So dual is equivalent to:

$$Min: \frac{A\sum_{q=1}^{n} X_q + PW\phi_p + LT\phi_d}{\sum_{j=1}^{k} d_j \cdot dist(j)} \quad (7)$$

where $dist(j)$ is the shortest path from the source to the sink of commodity j under the length function $l(e)$. The process is described in Algorithm 2.

Algorithm 2 $(1-\delta)$ Maximum Concurrent Flow Algorithm

Input: graph G, demand d, latency constraint LT, power budget PW, threshold δ
Output: $(1-\delta)$ optimal maximum concurrent value λ
$\forall q$, set $f(e) \leftarrow 0$, $X_q \leftarrow X_0$, $\phi_p \leftarrow X_0$, $\phi_d \leftarrow X_0$
$l(e) \leftarrow A_e \sum_{q:e\in Grid(q)} X_q + PW \cdot \phi_p + LT \cdot \phi_d$
while $A\sum_{q=1}^{n} X_q + PW\phi_p + LT\phi_d \leq 1$ **do**
  for each commodity j **do**
    $rd_j \leftarrow d_j$
    while $rd_j > 0$ **do**
      Route f units of flow from s_i to t_j along the shortest path P
      $f(e) \leftarrow f(e) + f$, $\forall e \in P$
      $X_q \leftarrow X_q(1 + \frac{\delta}{3} \cdot \frac{\sum_{e\in Grid(q)} A_e f(e)}{A})$
      $\phi_p \leftarrow \phi_p(1 + \frac{\delta}{3} \cdot \frac{\sum power}{PW})$, $\phi_d \leftarrow \phi_d(1 + \frac{\delta}{3} \cdot \frac{\sum latency}{PW})$
      $l(e) \leftarrow A_e \sum_{q:e\in Grid(q)} X_q + PW \cdot \phi_p + LT \cdot \phi_d$
      $rd \leftarrow rd - f$
    end while
  end for
  compute primal λ by scaling down all $f(e)$ subject to area, power and latency constraints
  compute dual $D \leftarrow \frac{A\sum_{q=1}^{n} X_q}{\sum_{j=1}^{k} d_j \cdot dist(j)}$
end while
return λ

The baseline algorithm proceeds in phases and each phase is composed of k iterations. In iteration j of the i^{th} phase we route d_j units of commodity j in a sequence of steps. In each step, a shortest path P from source s_j to sink t_j is computed using the current length function. The dual variables X_q are updated as

$$X_q = X_q(1 + \frac{\delta}{3} \cdot \frac{\sum_{e\in Grid(q)} A_e f(e)}{A}) \quad (8)$$

and ϕ_p and ϕ_d are updated in the similar fashion.

Regarding the convergence of Algorithm 2, by carefully choosing the initial values X_0, we have the following theorems:

Theorem 1: When the algorithm terminates, $\frac{\lambda}{D} \geq 1-\delta$.

Theorem 2: The algorithm runs in $O(\delta^{-2}|E|^2)$.

The first theorem guarantees the $(1-\delta)$ optimality and the second one shows the efficiency. The proofs are similar to those in [3] and [6]. In addition, the power and latency constraints can be viewed as two "pseudo edges" with capacities PW and LT, so ϕ_p and ϕ_d have the similar update formula.

4.2 Interval Estimation

While Algorithm 1 needs to obtain MCF solutions with $(1+\epsilon)$ optimal power values, Algorithm 2 returns us $(1-\delta)$ optimal concurrent flow. Therefore the values of ϵ and δ are associated "pseudo polynomially": δ has to be determined by both the value of ϵ and the unit edge cost P_e, which leads to extremely slow convergence in some pathological cases.

We propose a heuristic interval estimation technique to speedup the process. The idea is to estimate the new lower bound lb' and upper bound ub' while performing the approximation algorithms, and break once $ub' - lb' \leq (ub - lb)/2$ in each step of the binary search scheme.

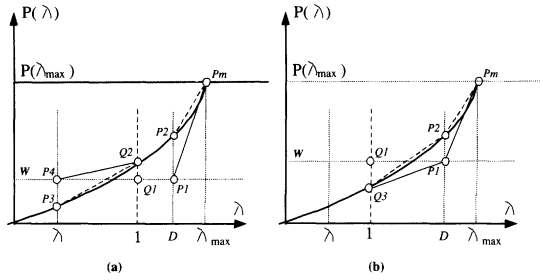

Figure 3: interval estimation

We define a function monotonically increasing $P(\lambda)$, where λ is the concurrent flow and $P(\lambda)$ is the minimum power under this concurrent flow (therefore $P(1)$ is the target optimal value). The curve is shown in Fig 3. Furthermore, we have the following lemma:

Lemma: $P(\lambda)$ is a convex function.

We use the following theorem to estimate the lower bound lb' and upper bound ub':

Theorem 3: Given a feasible primal value λ and a feasible dual value D under the power budget PW, we have

$$PW - s \cdot (D-1) \leq P(1) \leq PW + s \cdot (1-\lambda) \quad (9)$$

where $s = \frac{P(\lambda_{max}) - PW}{\lambda_{max} - D}$. Hence, $lb' \leftarrow \max\{lb', PW - s \cdot (D-1)\}$, $ub' \leftarrow \min\{ub', PW + s \cdot (1-\lambda)\}$

We sketch the proof for $P(1) \leq PW + s \cdot (1-\lambda)$ here. Refer to Fig. 3 (a), let the lines $x = \lambda$, $x = 1$, $x = D$ and $x = \lambda_{max}$ intersect the function curve at P_3, Q_2, P_2 and P_m, and $x = 1$, $x = 1$ and $x = D$ intersect $y = PW$ at P_4, Q_1 and P_1 respectively (we use x and y to denote the two axes). We then have

$$P(1) = PW + S_{P_4Q_2} \cdot (1-\lambda) \quad (10)$$

where $S_{P_4Q_2}$ is the slope of the line P_4Q_2. And, it is easy to identify that $S_{P_4Q_2} \leq S_{P_3Q_2} \leq S_{P_2P_m} \leq S_{P_1P_m}$, by the property of the convex function. And since $s = S_{P_1P_m}$, we have $P(1) \leq PW + s \cdot (1-\lambda)$. Similarly, $PW - s \cdot (D-1) \leq P(1)$ can be proven by the similar approach, as shown in Fig. 3 (b). □

According to Theorem 3, Algorithm 1 and 2 can be improved as Algorithm 3 and Algorithm 4.

Algorithm 3 Modified Power Minimization MCF Algorithm

Input: graph G, demand d, latency constraint LT, threshold ϵ
Output: $(1+\epsilon)$ optimal power
As in Algorithm 1 Steps 3–5
while $(ub - lb)/ub > \epsilon$ **do**
  $(lb', ub') \leftarrow mcf(G, d, LT, (lb + ub)/2)$
  $lb \leftarrow lb'$; $ub \leftarrow ub'$
end while
Output ub

Algorithm 4 Modified Maximum Concurrent Flow Algorithm

1: **Input:** graph G, demand d, latency constraint LT, power budget PW, threshold δ
2: **Output:** new lower bound lb' and upper bound ub'
3: As in Algorithm 2 Steps 3–4
4: $lb' \leftarrow lb$; $ub' \leftarrow ub$
5: **repeat**
6: As in Algorithm 2 Steps 6–18
7: $lb' \leftarrow \max\{lb', PW - s \cdot (D-1)\}$
8: $ub' \leftarrow \min\{ub', PW + s \cdot (1-\lambda)\}$
9: **if** $ub' - lb' \leq (ub - lb)/2$ **then**
10: **return** (lb', ub')
11: **end if**
12: **end repeat**

5. EXPERIMENTAL RESULTS

Our experiments are on NoC of size 8×8 in $0.18um$ design technology. We assume that grid length are $2mm$, and communication demands are evenly distributed, i.e. the bandwidth requirements between every pair of tiles are 1Gb/s. The experiments are based on power/delay parameters described in subsection 3.2. For 8×8 NoC, using the topology generation method proposed by [5], there are 2094 topologies in topology library for design selection. In MCF approximation algorithm, we set error tolerance ϵ to 1%. Since each grid has the same vertical and horizontal dimension, for convenience, we use only the vertical dimension to represent the area budget. This is why the unit of area in our experiments is um.

5.1 Power Consumption and Latency Tradeoffs

To demonstrate tradeoffs between power consumption and average latency in NoC design, we show power savings when a small amount of communication latency is sacrificed. First, we use the MCF model to search the topologies with the minimum latency (no power optimization), then relax this latency constraint by up to 10% and optimize NoC power consumption. Fig. 4 shows the results. The x-axis represents average latency. The y-axis represents power consumption. Each curve represents latency constrained minimum power consumption under certain area budget.

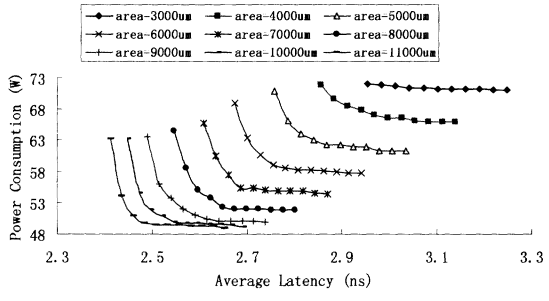

Figure 4: NoC power and latency tradeoffs

As area budgets increase, the curves move toward left-bottom due to wire style optimization, because those aggressively optimized but area-consuming wire styles, such as transmission lines, can be adopted to optimize both power and latency. When area increases from $3000um$ to $11000um$, minimum latency drops 18.3%, from 2.95 ns to 2.41 ns; average power consumption drops 28.3%, from 71.4W to 51.2W.

The slopes of the curves indicate the power consumption reduce rate when communication latency is increased. Take the curve with area $11000um$ as example, when latency constraint is loosen 2%, from 2.41ns to 2.46 ns, the power consumption is reduced from 63.2W to 50.9W, which is a 19.4% improvement. When area is small ($3000um$), the curve is almost flat. This is because area resource becomes bottleneck and flow is congested on chip, so that loose latency constraint will not bring much benefit.

5.2 Topology Selection

Our design methodology selects the optimal topologies from the topology library, which consume minimum power and satisfy latency constraints. In this section, we compare these optimal topologies with traditional topologies, such as mesh, torus and hypercube. Same as in subsection 5.1, we set the latency constraints by relaxing the minimum latency by up to 10%.

Fig. 5 shows comparison among these four types of topologies under three area budgets, $3000um$, $7000um$, and $11000um$, which represent loose, moderate and tight area constraints, respectively. The x-axis represents average latency. The y-axis represents power consumption. Each group includes 3 curves, which represent latency constrained minimum power consumption for a certain topology under various area budgets.

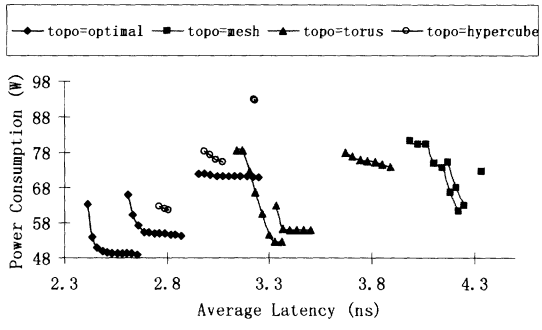

Figure 5: Power latency tradeoffs among various topologies

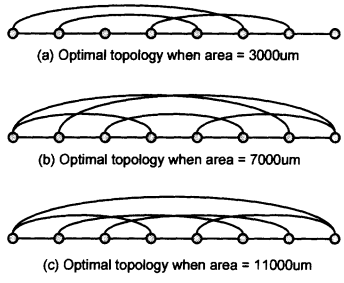

Figure 6: Optimal topologies under various areas

For a certain topology, since looser latency constraints lead to smaller power consumption, and vice versa, we pick the point with minimum power latency product for a quantitative comparison, as shown in Table 3.

The first two columns list area budgets and topologies. Column 3-5 show latency, power consumption, and power latency product for certain topology under given area budgets. The sixth column of the table lists the improvement in terms of power latency product, when comparing our selected optimal topology with mesh, torus and hypercube.

From the table, we observe that mesh is not a desirable topology for NoC of size 8x8. Compared with other topologies, its latency is quite large, because data packets need many hops to arrive the destinations. Also it lacks of long global links and doesn't make full use of wire style optimization, so that when area budgets increase, its power consumption is not as good as torus. Torus and hypercube have their own advantages. In general, torus is better in terms of power consumption, since it has simpler network router architecture; hypercube is better in terms of latency, since it has a lot of shortcut links.

Our selected optimal topologies show big advantages over the other three traditional topologies. They have small power consumption and latency. In terms of power latency production, they achieve an improvement up to 52.1% compared to mesh and 29.4% compared to torus (area = $11000um$), and 28.5% compared to hypercube (area = $7000um$). Fig. 6 shows the connections on one row of our selected optimal topologies under each area budget. Duplicating these connections to every row and column will generate the final topology design.

Table 3: Topology Comparison

area (um)	topo	L (ns)	P (W)	P*L (W*ns)	Imp (%)
3000	mesh	4.34	72.7	315.2	**26.7**
	torus	3.74	76.1	284.7	**18.9**
	cube	3.23	92.8	299.8	**23.0**
	optimal	3.25	71.1	**230.9**	
7000	mesh	4.25	63.0	267.9	**44.5**
	torus	3.37	56.3	189.6	**21.5**
	cube	3.04	76.0	231.2	**35.6**
	optimal	2.69	55.4	**148.8**	
11000	mesh	4.22	61.2	258.3	**52.1**
	torus	3.33	52.7	175.3	**29.4**
	cube	2.76	62.6	173.1	**28.5**
	optimal	2.48	49.8	**123.8**	

5.3 MCF Performance Improvement

To demonstrate the efficiency of the proposed MCF algorithm, we conduct experiments to compare its CPU time with that of CPLEX, a commercial LP solver. We choose torus as the representative topology to make the comparison. We test the performance on 3000, 7000 and 11000 as small, moderate and large grid area, and scale down them by the factor of $k^4/8^4$ for the $k \times k$ case, by approximating the communication demands to be k^4. All the experiments are conducted in a PC with 2.8 GHz CPU and 784MB memory, and CPLEX 9.1 is used. The detail results are shown in Table 4, where columns 3–6 show the result values and CPU time (in seconds) of CPLEX and our approximation algorithm respectively, column 7 shows the gap between the approximate results and the optimal solutions, $(col5 - col3)/col3$, and column 8 shows our speedup, $col4/col6$.

The table shows that our proposed algorithm can obtain correct results within the 1% error bound, which is our input settings. Also, it is much faster than CPLEX, in the 7×7 cases, it has been more than 100 times faster than CPLEX. In 8×8 cases, the approximation method also runs fast, while CPLEX is too slow to produce any results.

6. CONCLUSION

We study the tradeoffs between NoC power efficiency and latency. By adopting a MCF formulation, we are able to reduce NoC power consumption under given latency constraint, through simultaneous optimization of network topologies and wire styles. Experimental results suggest that power and latency co-optimization together with topology selection and wire style assignment are important in NoC design.

Table 4: MCF Performance Improvement

Size	Area	CPLEX		Approx.		Err (%)	Imp
		Obj	CPU	Obj	CPU		
5×5	473	6611	105	6652	11	0.62	9.55
	1069	5389	104	5430	11	0.76	9.45
	1679	5193	10	5234	12	0.78	0.83
6×6	950	16830	1496	16955	65	0.74	23.02
	2215	13195	1910	13298	29	0.78	65.86
	3481	12580	291	12683	29	0.82	10.03
7×7	1759	36860	9963	37156	78	0.80	127.73
	4104	28405	15040	28641	46	0.83	325.96
	6488	27464	8280	27689	56	0.82	147.86
8×8	3000	N/A	N/A	73315	113	N/A	N/A
	7000	N/A	N/A	56207	48	N/A	N/A
	11000	N/A	N/A	52915	62	N/A	N/A

7. ACKNOWLEDGEMENTS

This work was supported in part under grants from California MICRO program. The authors would like to thank Prof. Michael Taylor from UCSD, Prof. David Harris and Prof. Sarah Harris from Harvey Mudd College for their helpful discussion. The first author would like to thank Huaxia Xia for his helpful discussion on the MCF algorithm.

8. REFERENCES

[1] H. Chen, R. Shi, C.K. Cheng, and D. Harris, "Surfliner: A Distortionless Electrical Signaling Scheme for Speed of Light On-Chip Communications," *IEEE Intl. Conf. on Computer Design,* pp.497-502, Oct. 2005.

[2] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," *ACM/IEEE Design Automation Conf.*, pp. 684-689, June 2001.

[3] N. Garg, and J. Konemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," *the 39th Annual IEEE Symp. on Foundations of Computer Science,* pp. 300-309, Nov. 1998.

[4] J. Hu and R. Marculescu, "Energy-Aware Mapping for Tile-based NoC Architectures Under Performance Constraints," *Asia and South Pacific Design Automation Conf.*, pp. 233-239, Jan. 2003.

[5] Y. Hu, H. Chen, Y. Zhu, A. A. Chien, and CK. Cheng, "Physical Synthesis of Energy-Efficient Networks-on-Chip Through Topology Exploration and Wire Style Optimization," *Intl. Conf. on Computer Design,* pp. 111-118, Oct. 2005.

[6] G. Karakostas, "Faster approximation schemes for fractional multicommodity flow problems", *the 13th Annual ACM/SIAM Symp. on Discrete Algorithms*, pp. 166-173, 2002.

[7] U.Y. Ogras and R. Marculescu, "Application-Specific Network-on-Chip Architecture Customization Vis Long-Range Link Insertion", *Intl. Conf. on Computer Aided Design*, pp. 246-253, Nov., 2005.

[8] L.S. Peh, "Flow Control and Micro-Architectural Mechanisms for Extending the Performance of Interconnection Networks", *Ph.D. Thesis, Stanford University*, August, 2001.

[9] C. Seitz, "Let's Route Packets Instead of Wires", *Advanced Research in VLSI: Proceedings of the Sixth MIT Conference,* pp. 133-138, 1990.

[10] H. Wang, L. S. Peh, and S. Malik, "Orion: A Power-Performance Simulator for Interconnection network," *Intl. Symp. on Microarchitecture,* pp. 294-305, Nov. 2002.

[11] H. Wang, L. S. Peh, and S. Malik, "A Technology-aware and Energy-oriented Topology Exploration for On-chip Networks," *Design, Automation and Test in Europe,* pp.1238-1243, Vol.2, Mar. 2005.

[12] T. T. Ye, L. Benini, and G. De Micheli, "Analysis of Power Consumption on Switch Fabrics in Network Routers," *ACM/IEEE Design Automation Conf.*, pp.524-529, June, 2002.

[13] http://public.itrs.net