# FIT: Fast Irregular Topology generation algorithm for application specific NoCs

**Suleyman Tosun**[a)]
*Computer Engineering Department, Ankara University*
*Besevler, 06500, Ankara, Turkey*
a) *tosun@eng.ankara.edu.tr*

**Abstract:** This work presents a low complexity irregular topology generation algorithm for Netwok-on-Chip (NoC) design flow. The objective of the algorithm is minimizing the energy consumption of the final design. We tested the effectiveness of our algorithm by comparing it with its earlier counterparts on several multimedia applications.

**Keywords:** Network-on-Chip, topology generation, energy
**Classification:** Science and engineering for electronics

## References

[1] 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, 2006.
[2] K.-C. Chang and T.-F. Chen, "Low-power algorithm for automatic topology generation for application-specific networks on chips," *IET Comput. Digit. Tech.*, vol. 2, no. 3, pp. 239–249, 2008.
[3] Y. Ar, S. Tosun, and H. Kaplan, "TopGen: A New Algorithm for Automatic Topology Generation for Network on Chip Architectures to Reduce Power Consumption," *Proc. AICT2009*, Azerbaijan, Baku, 2009.
[4] J. Hu and R. Marculescu, "Exploiting the Routing Flexibility for Energy/Performance Aware Mapping of Regular NoC Architectures," *Proc. Design, Automation and Test in Europe*, 2003.

## 1 Introduction

Network-on-Chip (NoC) has been proposed in the beginning of this millenium as a new communication infrastructure for integrated circuits.The topology selection is the very first step of the NoC design flow. The topologies for NoC designs can be chategorized into two main groups as regular (e.g. mesh, torus, fat tree) and irregular (i.e. custom) topologies. While the former group exhibits the advantages of implementation easiness and reusability, the latter group, on the contrary, can be customized for the given application offering a huge optimization space for the designer. Early work focused on the energy minimization as an optimization criteria for the NoC designs [1, 2, 3]. Sev-

DOI: 10.1587/elex.7.1132
Received June 09, 2010
Accepted July 12, 2010
Published August 10, 2010

eral algorithms have been presented so far for energy aware irregular topology generation, utilizing different optimization techniques such as MILP (Mixed Integer Linear Programming) [1] and heuristic methods [2, 3]. Linear programming based methods take very long run times to obtain the solution. The MILP based method presented in [1] adheres to this timing disadvantage, which makes it inapplicable to design automation flow. In [2], the authors present a heuristic method consisting of three successive phases, which are core clustering, cluster optimizing, and physical router mapping. They utilize Gomory-Hu cut-tree algorithm for core clustering step. However, their method may not obtain optimal clustering as they also claim. The optimization method for this problem is not scalable since it does not follow well defined steps. Their router mapping method is also non optimal and the topologies created by this phase may need additional optimization steps. In [3], we proposed a two phase irregular topology generation algorithm. In this method, we first decompose the given application into clusters. We then construct the topology by mapping the clusters to routers. The bottleneck of this algorithm is the clustering phase that it may not produce the minimal flow among clusters. Additionally, it has high complexity.

In this work, we present a novel application specific irregular topology generation algorithm called FIT. The objective of our algorithm is minimizing the energy consumption of the final NoC design. We tested our algorithm on several multimedia benchmarks and we observed very promising improvements against earlier work. The novel contributions of this algorithm are twofold. First, it proposes new node clustering and topology construction algorithms separately, each of which has low complexity and easy to implement. Second, the algorithm can easily be modified to optimize different NoC parameters such as minimizing number of routers.

### 1.1 Energy model

The goal of this study is minimizing the energy consumed by the network resources of NoC architectures since the communication structures take significant portion of the consumed energy in today's electronic circuits. This energy consumption is directly proportional to the amount of bit transitions over the network. In order to estimate the energy consumption of NoC architecture, we should use an energy model based on the total bit transitions. In this work, we use the energy model presented in [4]. It is based on the average energy consumption of sending one bit data from core $v_i$ to core $v_j$ and can be formulated as

$$E_{T_{bit}}^{v_i,v_j} = \eta_{v_i,v_j} \times E_{S_{bit}} + (\eta_{v_i,v_j} - 1) \times E_{L_{bit}}, \qquad (1)$$

where $E_{S_{bit}}$ and $E_{L_{bit}}$ represent the energy consumed by the switches and the links, respectively, while $\eta_{v_i,v_j}$ represents the number of routers the bit passes through. Clearly, the bit passes through $\eta - 1$ links.

© IEICE 2010
DOI: 10.1587/elex.7.1132
Received June 09, 2010
Accepted July 12, 2010
Published August 10, 2010

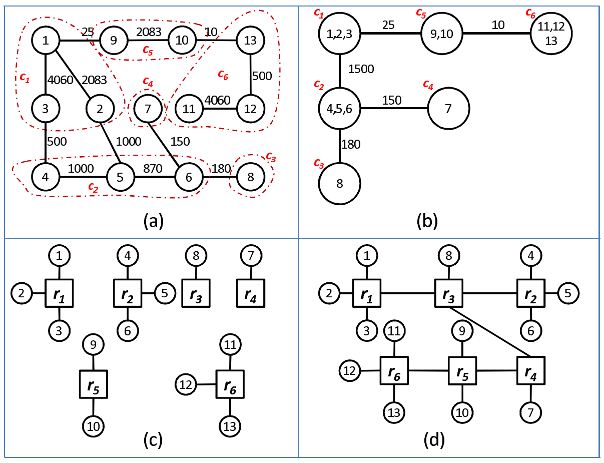

**Fig. 1.** Design steps from CFG (a) to TG (d).

## 1.2 Problem definition

The topology generation problem can be defined as determining a topology such that all communicating cores of the application can transmit data to each other over the network and the energy consumption is minimized. The resultant topology must ensure that each router can be reached from all other routers of the network and all the cores must be connected to at most one router port. To define this problem formally, we give the following definitions.

**Definition 1** *A CFG is a graph $G(V,E)$, where each vertex $v_i \in V$ represents a core (i.e. a node) in the application and each edge $e_{i,j} \in E$ represents a dependency between two tasks $v_i$ and $v_j$. The amount of data transfer between $v_i$ and $v_j$ is represented by $w_{i,j}$ for all $e_{i,j}$ in bits per second.*

**Definition 2** *A topology graph TG is a connected graph T(R, L, F, U), where R represents the set of routers, L represents the set of links between two routers, F represents the set of cores connected to router ports, and U represents the set of links connecting cores to router ports.*

In Fig. 1.(a) and (d), we give a CFG of MP3-Encoder [1] and the TG generated by our topology generation tool. Using the above definitions, the application specific topology generation problem can be formulated as follows.

**Problem 1** *Given a CFG and a set of routers R with p ports, generate a topology T(R, L, F, U) such that, for every $e_{i,j}$, there exists a routing path $p_{i,j} \in P$ and the total energy consumed by the network communication $(E_{NoC})$ is minimized, where*

$$E_{NoC} = \sum_{\forall e_{i,j} \in E} w_{i,j} \times E_{T_{bit}}^{v_i,v_j}. \tag{2}$$

## 2 FIT: Fast Irregular Topology generation algorithm

Our topology generation algorithm has two main steps: Node clustering and the topology construction. We give the pseudo-code of FIT in Fig. 2.

In *the node clustering step*, we aim to decompose the given CFG into clusters such that the communication among the nodes of the same cluster is maximized resulting in minimized traffic among clusters. After the clustering

© IEICE 2010
DOI: 10.1587/elex.7.1132
Received June 09, 2010
Accepted July 12, 2010
Published August 10, 2010

```
Data: $G = (V, E)$ and the port number $p$ of routers.
Result: $T(R, L, F, U)$, routing table $P$, and $E_{NoC}$.
begin
    Calculate the minimum number of routers $r$
    Construct initial cluster list $C = \{c_1, c_2, ...., c_n\}$ where
    $c_1 = \{v_1\}, c_2 = \{v_2\}, ..., c_n = \{v_n\}$
    Order $E$ in descending order to $L[|E|]$ based on the edge weights
    for $k \leftarrow 1$ to $|E|$ do
        Select $e_{i,j}$ from $E[k]$
        if $|c_i| + |c_j| < p$ then
            $c_i = c_i + c_j$
            $C = C - c_j$
    while $|C| > r$ do
        $hop\_dist = \infty$, $weight = -1$
        for $i \leftarrow 1$ to $|C|$ do
            for $j \leftarrow 2$ to $|C|$ do
                if $i \neq j \wedge |c_i| + |c_j| < p$ then
                    Calculate $d_{i,j}$, the shortest path distance (number
                    of hops) between $c_i$ and $c_j$, using Dijkstra
                    Calculate $q_{i,j}$, the total weight of $c_i$ and $c_j$
                        // $q_{i,j} = w_{i,k} + w_{n,j}$, where the path of $c_i$
                      // and $c_j$ is $p_{i,j} = \{(c_i, c_k), (c_k, c_l), ..., (c_n, c_j)\}$
                    if $d_{i,j} \leq hop\_dist \wedge q_{i,j} > weight$ then
                        $hop\_dist = d_{i,j}$ , $weight = q_{i,j}$
                        $Cluster\_1 = c_i$, $Cluster\_2 = c_j$
        $Cluster\_1 = Cluster\_1 + Cluster\_2$
        $C = C - Cluster\_2$
    Construct $R, F, U$ from $C$      // Nodes of clusters to routers
    $C' = c_{max}$,$R' = r_{max}$        // $c_{max}$ is the cluster with the
                                         // highest weighted edge
    $R = R - r_{max}$
    $Empty\_port = p - |c_{max}|$
    while $R \neq \emptyset$ do
        Select $c_i \in C$ with the highest communication with $C'$
        if $R - r_i = \emptyset$ then
            Connect $r_i - r_j$ via $l_{i,j}$ with minimum $Comm\_Cost_{NoC}^{R'+r_i}$
            $L = L + l_{i,j}$, $R = R - r_i$, $R' = R' + r_i$, $C' = C' + c_i$
        else
            if $Empty\_port + (p - |c_i|) - 2 > 0$ then
                Connect $r_i - r_j$ via $l_{i,j}$ with minimum $Comm\_Cost_{NoC}^{R'+r_i}$
                $L = L + l_{i,j}$, $R = R - r_i$, $R' = R' + r_i$, $C' = C' + c_i$
                $Empty\_port = Empty\_port + (p - |c_i|) - 2$
            else
                Select $c_k \in C$ with the highest communication with $C'$
                and $Empty\_port + (p - |c_k|) - 2 > 1$
                Connect $r_k - r_l$ via $l_{k,l}$ with min. $Comm\_Cost_{NoC}^{R'+r_k}$
                Connect $r_i - r_j$ via $l_{i,j}$ with min. $Comm\_Cost_{NoC}^{R'+r_k+r_i}$
                $L = L + l_{k,l} + l_{i,j}$, $R = R - r_k + r_i$, $R' = R' + r_k + r_i$,
                $C' = C' + c_k + c_i$
                $Empty\_port = Empty\_port + (p - |c_k|) + (p - |c_i|) - 4$
Construct routing table $P$ using Dijsktra
Return $T$, $P$, and $E_{NoC}$
```

**Fig. 2.** Pseudo-code of the algorithm.

step, the nodes of a cluster will be mapped onto the same router. In this way, the heavily communicating nodes communicate over the same router and less number of bits travels over the network links.

The maximum number of clusters and their sizes depend on the given router specifications. If we have $p$ ports for each router, we can have at most $p-1$ nodes for each cluster since one port of a router must be used to connect it to the network. In our algorithm, the number of clusters is equal to the number of routers of the application, which can be determined as follows.

**Theorem 1** *Given a set of routers $R$ with $p$ ports ($p > 2$) and a CFG with $n$ nodes, the minimum number of routers $r$ to generate a valid topology*

© IEICE 2010
DOI: 10.1587/elex.7.1132
Received June 09, 2010
Accepted July 12, 2010
Published August 10, 2010

*can be found by Eq. 3.*

$$r = \left\lceil \frac{|n-2|}{p-2} \right\rceil. \tag{3}$$

**Proof of Theorem 1** *For $r$ routers, each of which having $p$ ports, we can have at most $pr$ ports. Each link consumes two ports to connect two routers. For $r$ routers, we can have at least $r-1$ links, which means the total links consume at least $2(r-1)$ ports. Then, the remaining ports can be used for connecting at most $pr-2(r-1)$ nodes. This inequality can be written as:*

$$n \leq pr - 2(r-1). \tag{4}$$

*Solving this inequality for $r$ gives us Eq. (3).* Q.E.D.

In our node clustering algorithm (lines 2-21 in Fig. 2), we initially accept each node as a cluster. We then merge the clusters based on the communication weights between them until the number of clusters is equal to the number of available routers (lines 4-9). To do this, we order the edges in decreasing order based on their edge weights. If there is a tie among the edges, we give priority to the edge that its connected clusters have the highest total communication with their neighbors. We then merge two clusters connected by this edge. If we cannot merge these two clusters as a result of port constraints, we skip this edge and pick the next one.

Having visited all edges of the CFG, we may not reduce the number of clusters to its limits. In such a case, using Dijkstra's shortest path algorithm, we calculate the minimum edge distances of the cluster pairs that can be merged. We then merge two clusters with the smallest distance and the highest communication weights. The rationale behind this selection is to try connecting heavily communicating nodes close to each other to have smaller hop distances. We continue this merging process until we meet the cluster number constraint (lines 10-21). We give the core clustering and router mapping in Fig. 1.(b) and (c), respectively.

The second step of FIT is *the topology construction* (lines 22-43). After determining the clusters, we map each of their nodes to corresponding routers (line 22). We then construct our topology by connecting routers to the network one by one (lines 26-41). We start with the router that has the highest communication traffic with other routers. The traffic between routers is taken from the corresponding cluster communication data. We then select the router to connect to the network based on its communication weight with the network. We connect this router if it does not block the connection of remaining routers. If they do, we pick an intermediate router that has the highest communication with the network and it can help connecting this router without blocking the remaining routers. We selectively connect all routers to the network. We then determine the routing path of communicating nodes using Dijkstra's algorithm and calculate the total energy consumption (lines 23-43). The topology for MP3 Enc. is given in Fig. 1.d.

**Complexity of FIT:** The complexity of edge ordering is $O(E \log E)$. The complexity of node clustering is $O(E)$ on the best case (lines 5-9 in

© IEICE 2010
DOI: 10.1587/elex.7.1132
Received June 09, 2010
Accepted July 12, 2010
Published August 10, 2010

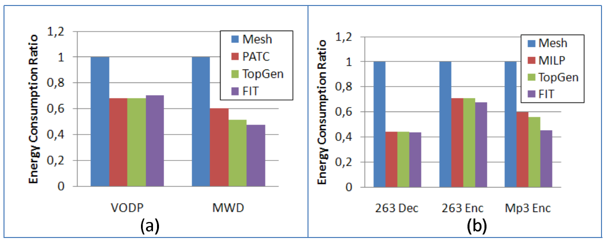

**Fig. 3.** Energy consumption comparisons.

Fig. 2) and $O(V^2(E + V\log V))$ on the worst case (lines 10-21), where the term $O(E + V\log V)$ is the complexity of Dijkstra. The topology generation algorithm takes $O(R^2)$ times (lines 26-41). From this analysis, the worst case complexity of FIT is given by $O(V^2(E + V\log V))$ while the best case complexity is $O(E + R^2)$.

## 3 Experimental results

We evaluated FIT using five multimedia benchmarks; VOPD and MWD from [2] and 263 Enc., 263 Dec. and MP3 Dec. from [1]. We compared the estimated energy consumption values obtained by FIT against the ones obtained by PATC [2], MILP based-based technique [1], and TopGen [3]. We also calculated the energy consumption of the optimum design on $4 \times 4$ mesh topology. In our designs, we used four port routers and 100-nm technology size parameters. As power consumption model, we use the model in [1]. In this model, the power consumptions of the routers are estimated as 328 nW/Mb/s and 65.5 nW/Mb/s for input and output ports, respectively. The link power consumption is estimated as 79.6 nW/Mb/s/mm. We give the energy consumption comparisons of the mentioned algorithms in Fig. 3. The values on the bar-charts are determined by taking the ratio of the energy consumption values over the mesh based design measurements. As the bar-charts show, our algorithm obtains the best solutions in most cases except VOPD. We observed energy improvement values of 8.24% against TopGen, 21.02% against PATC, and 25.38% against MILP. The energy consumption improvements show the accuracy of our FIT algorithm.

The CPU time of an optimization algorithm is an important evaluation criterion since it affects the scalability of the algorithm on varying problem sizes. In our case, MILP has the worst timing results as expected [1]. The solutions for MILP in Fig 3 are obtained after aborting MILP solver in 8 hours. On the other hand, our algorithm determined the results in seconds ranging from three to eight seconds. The other two algorithms [2] and [3] took similar CPU times as our algorithm. However, the obtained results show that our algorithm has better energy values than [2] and [3].

## 4 Conclusion

In this paper, we proposed a new **F**ast **I**rregular **T**opology (**FIT**) generation algorithm for NoC architectures. It outperforms the earlier algorithms in

© IEICE 2010
DOI: 10.1587/elex.7.1132
Received June 09, 2010
Accepted July 12, 2010
Published August 10, 2010

performance and energy consumption values for evaluated benchmarks.

## Acknowledgments

This work is supported by Scientific and Technological Research Council of Turkey (TUBITAK) under the project ID 108E233.

© IEICE 2010
DOI: 10.1587/elex.7.1132
Received June 09, 2010
Accepted July 12, 2010
Published August 10, 2010