Application Mapping onto Binary Tree Structured Network-on-Chip using Particle Swarm Optimization

Sunil Raju Gollapalli

Department of Electronics & Communications Engg, AM Reddy Memorial College of Engineering & Technology.

Abstract—This paper addresses the problem of application mapping for Binary Tree based Network-on-Chip. It proposes a new mapping technique based on Particle Swarm Optimization (PSO). The cores of the core graph are mapped to the routers using the above heuristic. Communication cost is a common metric in evaluation of different mapping techniques. Mapping results have direct impact on the performance of the mapped Network-on-Chip (NoC). Our BT-mapping results have been compared with the techniques reported in the literature for a number of benchmark applications. Apart from static communication cost, the dynamic costs, in terms of throughput and latency of the mapped solutions have also been compared.

Keywords–Application mapping, Network-on-Chip, System-on-Chip, Binary Tree topology, Particle Swarm Optimization.

I. INTRODUCTION

Network-on-Chip (NoC) has evolved as a viable strategy to implement Intellectual Property (IP) core based System-on-Chip (SoC) designs. It solves the traditional problems of bandwidth limitations of bus-based SoC design by providing an on-chip network fabric consisting of routers connected in a certain topology. Each core is connected to one of the routers. Conventional data signal exchanges are replaced by message passing between the cores through the router fabric [1]-[4]. A major challenge in NoC based system design is to associate the IP cores implementing tasks of an application with the routers. This is commonly known as the process of application mapping. This has got a very significant role to play in the performance of the overall system as it directly influences the communication time, the required link bandwidth and the admissible delay of the router.

Application-Specific Network-on-Chip (ASNoC) is an emerging paradigm proposed to handle the communication problem between the set of computational resources in modern applications [5]. ASNoC optimizes the design of on-chip network to comply with the requirements of the application [6]. Low communication cost, less area overhead, low energy consumption, and high throughput are the main desirable characteristics of ASNoC based systems [7]. Furthermore, these requirements vary from one application domain to another. For example, while multimedia applications require high bandwidth, real time systems require guaranteed delay, and portable devices require low power consumption. The choice of a network topology for an ASNoC system significantly impacts its performance. It is required to choose carefully the topological structure of ASNoCs to meet the design requirements, while minimizing communication cost, power consumption, area etc. In [6]-[9] many NoC topologies have been proposed. But it has been found that Binary Tree enjoys several advantages over others . The BT topology can be easily implemented inside chips. It has regularity in architecture. The link lengths are small and equal. The routers used within the topology are of same type, but none at the others.A BT based network with N(power of 2) number of IP core has 5(leaf),3(stem),2(root). It can be fabricated on a single metal layer. IP's are connected at the Leaf level node. , Hence, in our proposed work, we have chosen two dimensional BT for the mapping of applications onto NoC. Several application mapping techniques [10]-[17] have been proposed for NoCs, based on BT topology, and detail of which has been presented in Section VI. This paper explores another meta-heuristic, Particle Swarm Optimization (PSO), to attain BT-based NoCs for various applications. The paper contributes the following:

1. A Particle Swarm Optimization (PSO) based approach is presented for application mapping to minimize the overall communication cost.
2. A comparison of static performances between our BT-mapping solutions and the BT based implementations of various benchmark applications using existing BT-mapping algorithms reported in the literature.
3. A comparison of the BT-based NoCs resulting from mappings using existing approaches and the proposed PSO, in terms of dynamic throughput and latency values corresponding to the traffic generated from the applications.
4. Comparison with the works [10]-[14] [17] proves our proposed mapping technique to be a strong competitor as far as performance of the system is concerned.

The rest of the paper is organized as follows. Section II gives a brief survey of previous works on application mapping for BT based NoCs. Section III presents the problem formulation. A brief discussion on PSO is presented in Section IV. Section V presents PSO formulation for application mapping. Section VI embodies both static and dynamic performance analysis by taking some real SoC benchmarks.

II. RELATED WORKS

Various algorithms have been proposed to map an application onto different standard topologies. It may be noted that the mapping of cores onto NoC architecture presents new challenges when compared to the mapping in the domain of parallel processing. A major difference is that the traffic requirements on the links of a NoC are known for a particular application, thus the bandwidth constraints in the NoC architecture need to be satisfied by the mapping. In [10], PMAP, a two-phase mapping algorithm for placing clusters onto processors is presented. In [11], GMAP, and PBB a branch and bound algorithm, has been proposed that map cores onto a tile-based NoC architecture satisfying the bandwidth constraint and minimizing the total energy consumption. In [12], NMAP, a mapping technique has been proposed with minimum path routing in the BT architecture. It also proposes traffic splitting that considers the mapping problem together with the possibility of splitting traffic among various paths with satisfaction of bandwidth constraints. A tool, SUNMAP, has been presented in [13] to automatically select the best standard topology for a given application and producing a mapping of cores onto that topology. An efficient binomial IP mapping and optimization algorithm (BMAP) has been presented in [14] to reduce hardware cost of on-chip network compared to the previously proposed algorithms. In [15] merit of the application specific NoC mapping scheme is not clear, as no comparison has been made with the existing approaches. The mapping technique presented in [16] considers improvement over GA and reports relative improvements only. Thus, it is not clear how good the approach performs compared to other existing techniques. In [10]-[16] authors have not reported dynamic (average network latency and throughput) performance of their mapping for real benchmark applications. In [17], LMAP, a mapping algorithm has been proposed to reduce both static and dynamic cost using Kernighan-Lin (K-L) based partitioning scheme. A study reported in [17] shows that the algorithms NMAP and LMAP perform consistently well for all the benchmarks. While for some of them NMAP shows better performance, for others LMAP does better. This motivates us to look into the meta-heuristic strategies like PSO for application mapping.

In our proposed work we have used Particle Swarm Optimization (PSO) [18]-[20] based mapping technique to minimize the overall communication cost of the system. The work presented here is an attempt to utilize PSO as a vehicle to obtain mapping of applications onto BT based NoCs. In the process, the communication cost is minimized. We name our PSO based mapping technique as PSMAP. PSMAP satisfies the bandwidth constraints of the application and minimizes the average communication delay.

III. PROBLEM FORMULATION

In the application mapping problem for NoC, the communication requirements between the tasks of application are represented via a core graph while the network topology is represented through a topology graph [12] [17].

Definition 1 The core graph is a directed graph, G(C, E) with each vertex $c_i \in C$ representing a core and the directed edge $e_{i,j} \in E$, representing the communication between the cores c_i and c_j. The weight of edge $e_{i,j}$, denoted by $comm_{i,j}$, represents the bandwidth requirement of the communication from c_i to c_j.

Definition 2 The NoC topology graph is a directed graph P(U, F) with each vertex $u_i \in U$ representing a node in the topology and the directed edge $f_{i,j} \in F$ representing a direct communication between the vertices u_i and u_j. The weight of the edge $f_{i,j}$, denoted by $bw_{i,j}$, represents the bandwidth available across the edge $f_{i,j}$.

The mapping of the core graph G(C, E) onto the topology graph P (U, F) is defined by the mapping function map.

map: $C \rightarrow U$, such that, $map(c_i) = u_j$, $\forall c_i \in C$, $\exists u_j \in U$,

while each core is mapped to exactly one node in the NoC topology.

The mapping is defined when $|C| \leq |U|$.

The core graph of benchmark application VOPD is shown in Fig. 1. The NoC graph for 16-node BT is shown in Fig. 2(a). And an example mapping of the VOPD core graph is shown in Fig. 2(b). As noted in [12] [17], the communication between each pair of cores is treated as a flow of single commodity, represented as d^k, k = 1, 2, ⋯ |E|. The value of d^k represents communication bandwidth across the edge and is denoted by vl (d^k). The set of all commodities is represented by D and is defined as:

$$D = \begin{cases} d^k: vl(d^k) = comm_{i,j}, k = 1, 2, \cdots |E|, \forall e_{i,j} \in E \\ with\ source(d^k) = map(v_i), dest(d^k) = map(v \end{cases}$$

The bandwidth constraints are represented by the inequality:

$$\sum_{k=1}^{|E|} x_{i,j}^k \leq bw_{i,j}, \forall_{i,j} \in 1, 2, \cdots, |U|$$

For X-Y deterministic routing, λ are obtained by the following equation:

$$\lambda = \begin{cases} vl(d^k), if\ f_{i,j} \in path(source(d^k), dest(d^k \\ 0, \text{ otherwise} \end{cases}$$

where the path (a, b) represents the X-Y deterministic routing path between the BT nodes a and b.

If the bandwidth constraints are satisfied, the communication cost is given by:

commcost= $\sum_{k=1}^{|E|} vl(d^k) hopcount(source(d^k), dest(d$

where hopcount (a, b) is the minimum number of hops between the nodes a and b.

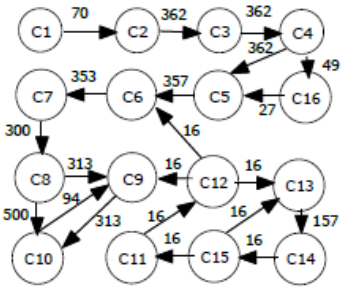
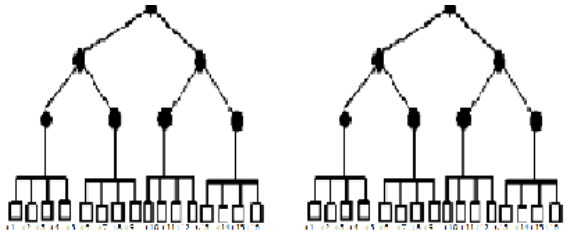


Figure 1. VOPD core graph



(a) Binary Tree Graph (b) Mapping

Figure 2. Mapping of VOPD Core graph onto BT graph

IV. PARTICLE SWARM OPTIMIZATION

There are several mapping algorithms that have been recently proposed to minimize the communication cost. Reducing hopcount between related cores will significantly drop the communication cost. In this paper, Particle Swarm Optimization (PSO) technique is used to minimize communication cost. Particle Swarm Optimization (PSO) [18] [19] is a population based stochastic technique developed by Eberhart and Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling. In PSO system, multiple candidate solutions coexist and collaborate simultaneously. Each solution, called a particle flies in the problem space according to its own experience as well as the experience of neighboring particles. It has been successfully applied in many problem areas. In PSO, each single solution is a particle in the search space, having a fitness value. The quality of a particle is evaluated by its fitness. The velocity information predicts the next moving direction. So, in PSO, each particle utilizes velocity and position information. PSO is initialized with a group of random particles. The particles then update for optimal solutions through generations by following two best values. The first one is the position vector of the best solution (fitness) this particle has achieved so far. This position vector is called personal best (pbest) or local best. Another best position, called the global best (gbest) that is the best position attained so far by any particle in the population. In standard particle swarm optimization, the velocity and position is updated as follows:

$$v_{k+1}^i = w v_k^i + c_1 r_1 \left(pbest^i - x_k^i\right) + c_2 r_2 \left(gbest_k - x_k^i\right) \quad (1)$$

$$x_{k+1}^i = x_k^i + v_{k+1}^i \quad (2)$$

where v_{k+1}^i is the velocity of particle i at $(k+1)^{th}$ generation, x_k^i is the current particle solution (position), r_1 and r_2 are random numbers between 0 and 1, c_1 is the self-confidence (cognitive) factor, c_2 is the swarm confidence (social) factor, and w is the inertia factor. In equation (1) the first term represents effect of inertia of particle, second term represents particle memory influence and third term represents swarm influence. Equation (2) represents the velocity information. In each generation, the equations noted above for velocity and position are evaluated. If a particle goes out of permissible region, it is clamped to V_{max}, which is a user defined parameter.

V. PSO FORMULATION FOR APPLICATION MAPPING

In this section we have formulated the PSO for application mapping to minimize the communication cost.

A. Particle Structure and Fitness

The structure of a particle is enumerated first. A particle corresponds to a possible mapping of cores to the routers. An example of the particle structure is shown in Fig. 3. In that example, the numbers shown within the circles in the boxes are the core numbers present in the core graph. The numbers outside the box are the router numbers of the topology graph. This figure shows that, core 1 is placed to the router 0; core 4 is placed with router 1, and so on. If the number of nodes (routers) present in the topology graph is greater than the number of cores present in the core graph, dummy nodes are added to the core graph to make the two numbers same. Dummy nodes are connected to all core nodes and between themselves. Edges connecting a core node to dummy nodes are assigned cost zero while the edges between dummy nodes are assigned a cost infinity. Let N be the number of cores present in the core graph for mapping of cores onto the topology graph, after connecting dummy nodes, if required. For these N cores, there are N node positions in the topology graph. A particle is a permutation of numbers from 1 to N, which shows the placement of cores to the node positions of the topology graph. The overall communication cost is influenced by the position of cores in a particle. In our formulation, the overall communication cost forms the fitness function.

Fitness of a particle P_i = the overall communication cost after placement of cores of the core graph in different routers specified by the particle.

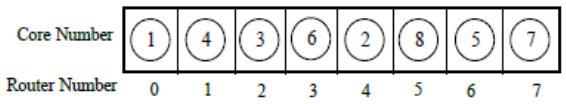


Figure 3. Particle Structure

B. Local and Global bests

In the evolution process, every particle has a local best (pbest), which is the permutation of core positions that gives minimum communication cost, among all permutations that the particle has seen so far. The local best permutation value guides partially the evolution of particle. For a particular generation, the particle resulting in the minimum communication cost is the global best (gbest) for that generation. This parameter also guides the evolution of particles. The local best of each particle and the global best of a generation are modified if the corresponding values in the current iteration are lesser than the values till the previous iteration.

C. Evolution of Generations

The particles are evolved through generations to create new particles which give the result closer to the optimum. In the first generation, the initial population is created randomly and the fitness of individual particles is evaluated. The local best (pbest) of each particle is same as the initial particle. The global best (gbest) of the first generation is the particle giving the least communication cost (smallest fitness function) in the generation. By exchanging the core positions within the particles randomly, second generation is evolved. If they give better fitness values, the local and the global best values are updated. The further generations are evolved through a series of operations called swap operations [20]. The local best of each particle and the global best of a generation are modified if the corresponding values in the current generation are lesser than the values in the previous generation.

C. 1 Swap Operator

The position of a core in a particle P can be represented by its position index. The indexing of the position takes value between 0 to N–1(N being the number of routers). The index corresponds to the router number, as shown in Fig. 3. Let the swap operator be $SO_{j,k}$ (where, j and k = 0, 1, ⋯ N–1) that swaps the j^{th} and k^{th} positions of the particle P to create a new particle P_{new}. For example, let us consider the particle P = {1, 4, 3, 6, 2, 8, 5, 7}, where the numbers represent the core numbers of the core graph and the position represents the router numbers in the topology graph. The swap operator $SO_{4, 6}$ swaps the cores at position 4 and 6, which creates a new particle P_{new}= {1, 4, 3, 6, 5, 8, 2, 7}.

C. 2 Swap Sequence

A swap sequence SS is made up of one or more swap operators. The swap operators of the swap sequence are applied in order upon the particle P to create a new particle P_{new}. For example, let the swap sequence SS = {$SO_{4, 6}$, $SO_{2, 5}$} be applied upon the particle P = {1, 4, 3, 6, 2, 8, 5, 7}. It creates a new particle P_{new}= {1, 4, 8, 6, 5, 3, 2, 7}.

To align a particle P_i with its local best, the swap sequence is identified. Let this be $SS_i^{l\_best}$. Then another swap sequence is identified to align the particle with the global best. Let this be $SS_i^{g\_best}$. Now the swap sequence $SS_i^{l\_best}$ is applied on particle P_i with a probability of α. Let the modified particle be $P_i^{l\_best}$. Then the swap sequence $SS_i^{g\_best}$ is applied on $P_i^{l\_best}$ with a probability of β. This creates a new particle P_i^{new}. The values of probabilities α and β in our calculations are taken as 0.5 each [19]-[21]. Its fitness is evaluated and the local best is updated for particle i, if it is better than the previous local best for the particle. If the best fitness in a generation is better than the global best of the previous generation, the global best is updated.

D. PSO Algorithm

Initialization:

for each particle
Initialize particle
Initialize $SS_i^{l\_best}$ and $SS_i^{g\_best}$
Evaluate fitness value
Set local_best of each particle to itself
Set global_best to the best fit particle

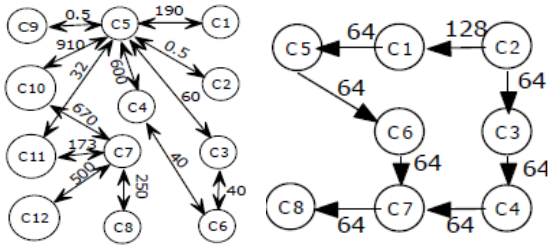
Evolutions:
Do
for each particle P_i
P_i^{new} = Modify P_i by applying $SS_i^{l\_best}$ with probability α and $SS_i^{g\_best}$ with probability β
Evaluate fitness of P_i^{new}
If fitness of P_i^{new} is better than the local best for P_i
then update local_best for P_i
endfor
Find the particle with best fitness and update global_best
While maximum generation (pre-specified) not attained or constant global_best is not attained since many (pre-specified) generations.

VI. SIMULATION RESULTS

A. Static Performance Analysis:

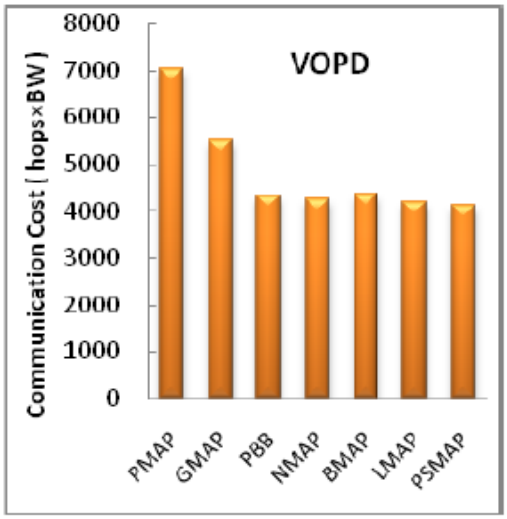
Fig.5 shows the communication costs for our proposed technique PSMAP, comparing it with LMAP, BMAP, NMAP, PBB, GMAP, and PMAP as reported in [17]. It shows the communication costs for the applications with the same bandwidth constraints for all algorithms. In [14] [17], it is reported that NMAP and LMAP have the best performance (communication cost) depending on the application benchmarks, so these two mapping techniques are adopted as reference point to judge our approach. In application VOPD (shown in Fig.1), communication cost improves to 0.96 of that for NMAP. In case of MPEG-4 (shown in Fig. 4(a)), it improves to 0.97. But in case of PIP (shown in Fig. 4(b)) it remains same, as NMAP. If we adopt LMAP as reference, the

communication cost is improved to 0.98 for VOPD, 0.88 for MPEG-4. But in case of PIP it remains same.

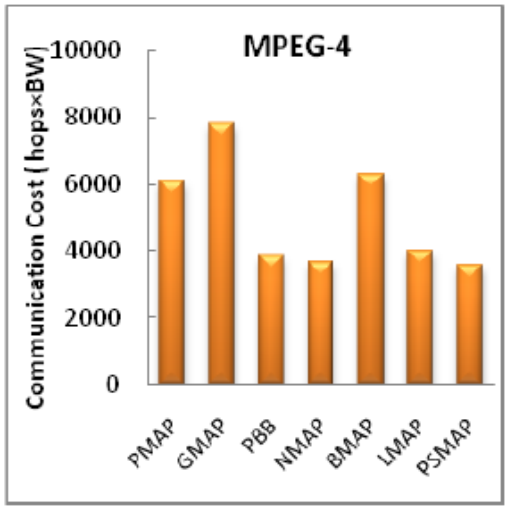


(a)MPEG-4 (b)PIP

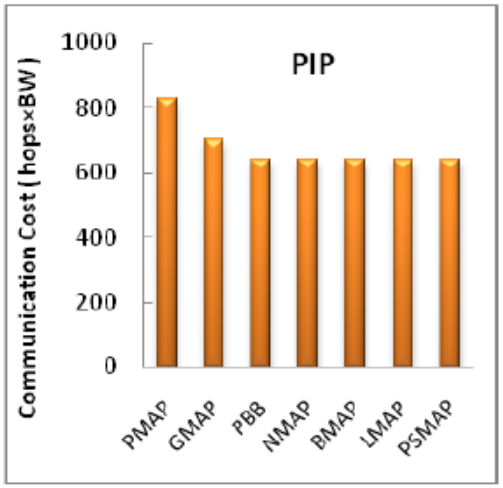
Figure4. Example of Core Graphs, with Communication BW (MB/s)



(a)



(b)

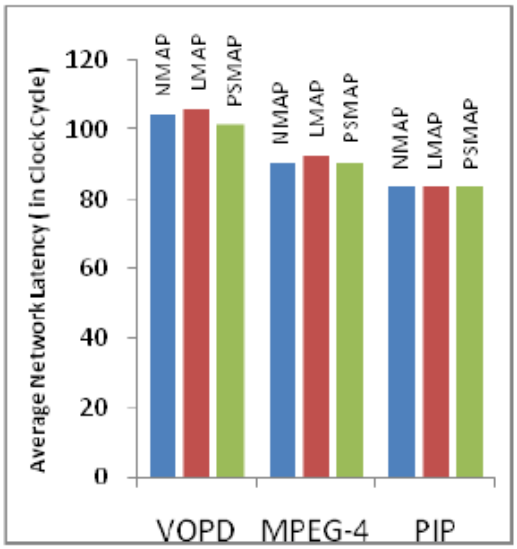


(c)

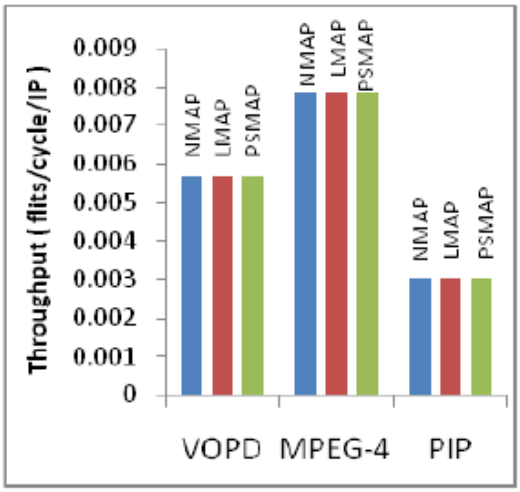
Figure 5. Communication cost of mapping algorithms for applications (a) VOPD, (b) MPEG-4 and (c) PIP

B. Dynamic Performance Evaluation:

Though static analysis gives a rough idea about performance of different topologies and mappings, it is not sufficient, as the real system may face network congestions leading to increased latency and reduction in throughput. To get a better understanding, we next simulate the mapped NoCs (applications VOPD, MPEG-4, PIP) using the cycle-accurate System C based simulators. We have taken NMAP and LMAP as reference to compare with our mapping technique. Taking NMAP values as unity, in VOPD the average network latency is improved to 0.97, in MPEG-4 and PIP it is comparable. If we take LMAP as unity to compare our technique, the average network latency of VOPD is improved to 0.95 and 0.98 for MPEG-4. In case of PIP it is comparable. Fig. 6 shows the simulation results graphically.



(a) Average Network Latency



(b) Average Throughput

Figure 6. Comparison results of PSMAP with NMAP and LMAP.

CONCLUSION

In this paper we have presented a mapping strategy for BT based NoC using PSO technique. It shows reasonable improvement in communication cost while considering static operation of the system. The dynamic performance of this strategy is comparable to the best ones previously available. The static analysis is a congestion free one. On the other hand, in dynamic performance evaluation, the system may face network congestions. Comparison of solutions produced establishes our mapping technique to be a strong competitor of previously available mapping strategies. Here we have shown the results up to 16-core real SoC applications. Future works involve both static and dynamic performance analysis for higher number of core applications and the power requirements of the mappings.

REFERENCES

[1] L. Benini, G. De Micheli, "Networks on Chips: A New SoC Paradigm," IEEE Computer, vol. 35, no. 1, pp. 70-78, Jan. 2002.

[2] W. J. Dally et al., "Route Packets, Not Wires: On-Chip Interconnection Networks", Proceedings of the 38th Design Automation Conference (DAC), pp. 684-689, 2001.

[3] S. Kumar et al., "A Network on Chip Architecture and Design Methodology," Proceedings of ISVLSI, pp. 117-124, 2002.

[4] S. Kundu, S. Chattopadhyay, "Interfacing Cores and Routers in Network-on-Chip Using GALS," IEEE International Symposium on Integrated Circuits (ISIC), Singapore, 2007.

[5] L. Benini, "Application Specific NoC Design," Proceedings of the IEEE Design, Automation and Test in Europe Conference (DATE'06), vol. 1, Munich, Germany, Mar. 6–10, pp. 1–5, 2006.

[6] Y. L. Jeang, W. H. Huang, and W. F. Fang,"A Binary Tree Architecture for Application Specific Network on Chip (ASNOC) Design," IEEE Asia-Pacific Conference on Circuits and Systems, pp.877–880, 2004.

[7] H. Elmiligi, A. A. Morgan, M. W. El-Kharashi, and F. Gebali,"A Topology-based Design Methodologyfor Networks-on-Chip Applications," Proceedings of the second International Design and Test Workshop (IDT'07), pp. 61–65, Cairo, Egypt, Dec. 16–18, 2007.

[8] D. Bertozzi et al., "NoC Synthesis Flow for Customized Domain Specific Multiprocessor Systems-on-Chip," IEEE Transactions on Parallel and Distributed Systems, vol. 16, no. 2, pp. 113–129, Feb. 2005.

[9] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance Evaluation and Design Trade-offs for MP-SOC Interconnect Architectures," IEEE Transactions on Computers, Vol. 54, No. 8, pp.1025–1040, 2005

[10] N. Koziris et al.,"An Efficient Algorithm for the Physical Mapping of Clustered Task Graphs onto Multiprocessor Architectures", Proceedings of 8th EuroPDP, pp. 406-413, Jan, 2000.

[11] J. Hu, R. Marculescu,"Energy-Aware Mapping for Tile-based NOC Architectures Under Performance Constraints," ASP-DAC 2003, Jan 2003.

[12] S. Murali and G. De Micheli, "Bandwidth Constrained Mapping of Cores onto NoC Architectures," Proceedings of Design, Automation and Test in Europe Conference and Exhibition (DATE), vol. 2, pp. 896-901, Feb 2004.

[13] S. Murali, G. De Micheli, "SUNMAP: A Tool for Automatic Topolog Selection and Generation for NoCs," Proceedings of 41st Design Automation Conference (DAC), pp. 914-919, 2004.

[14] T. Shen, C. H. Chao, Y. K. Lien, and A. Y. Wu, "A New Binomial Mapping and Optimization Algorithm for Reduced-Complexity Mesh- based on-Chip Network," Proceedings of NOCS'07, pp. 317-322, May 2007.

[15] A. R. Fekr, A. Khademzadeh, M. Janidarmian, V. S. Bokharaei, "Bandwidth/ Fault/ Contention Aware Application-Specific NoC using PSO as a Mapping Generator," Proceedings of the World Congress on Engineering (WCE), Vol I, June 2010.

[16] W. Lei, L. Xiang, "Energy- and Latency-Aware NoC mapping Based on Discrete Particle Swarm

Optimization," Proceedings of IEEE Internationa Conference on Communications and Mobile Computing, pp. 263-268, 2010.

[17] P. K. Sahu, N. Shah, K. Manna, and S. Chattopadhyay, "A New Application Mapping Algorithm for Mesh based Network-on-Chip Design," IEEE International Conference (INDICON), 2010.

[18] I. Kennedy, and R. C.Eberhart, "Particle Swarm Optimization," Proceedings of IEEE International Conference on Neural Networks, NJ. pp.1942-1948, 1995.

[19] Yuhui Shi and Russell Eberhart, "A Modified Particle Swarm Optimizer," Proceedings of IEEE International Conference on Evolutionary Computation, Anchorage, pp.69-73, May 1998.

[20] K. Wang, L. Huang, C. Zhou, W. Pang, , "Particle Swarm Optimization for Traveling Salesman Problem," Proceedings of the Second International Conference on Machine Learning and Cybermetics. pp. 1583-1585, 2003.

[21] Yuhui Shi and Russell Eberhart, "Parameter Selection in Particle Swarm Optimization," Springer Berlin/ Heidelberg, Vol. 1447/1998, April 2006.