System Level Approach to NoC Design Space Exploration

R. K. Jena, *Member, IACSIT*

***Abstract*—Network-on-Chip (NoC) has recently emerged as a communication solution for most of the System-on-Chip(SoC) design. Design space exploration and performance evaluation are the most essential task in NoC design. In this paper, we proposed a PSO based integrated design space exploration framework for the NoC design at system level. The results show that our framework optimizes the design matrices like system throughput and average packet latency for the target application.**

***Index Terms*—NoC, PSO, analytical model, design space exploration.**

I. Introduction

As technology scales up and chip integrity grows, on chip communication is playing an increasing dominant role in System-on-Chip (SoC) design. The NoC approach was proposed as a promising solution to complex intra-SoC communication problems [1]. Compared to traditional bus interconnection architecture, NoC is much more extensible and parallelizable. But, the NoC based system requires a reliable methodology for better design space exploration. The success of these methodologies depends on the availability of adequate performance analysis tools that can efficiently guide the design space exploration. The Fig. 1 shows the proposed frame work for the integrated design space exploration. The inputs to the frame work are Noc architecture and the application in the form of task graph. In this paper, we present an integrated approach at system level, where in the first step we maps the IP,s to the mesh based NoC architecture using PSO with an objective to minimize the energy consumption and link bandwidth. In the second step we present an analytical performance analysis methodology for NoCs, based on a novel switch/router model. The router/switch model allows us to compute the average number of packets at each buffer in the network as a function of the traffic arrival process. This model is then used to analyze each router in the network for the given topology, routing algorithm, driver application and mapping to the network. The proposed approach, which is developed for wormhole flow control, provides performance metrics, namely average packet latency per flow, and network throughput. These metrics can be conveniently used for design and optimization purposes, as well as obtaining quick performance estimates.

The remaining of this paper is organized as follows. Section II reviews related work and highlights our contributions. Section III presents the analytical modeling of switch/router, while Section IV discusses our proposed design space exploration framework. Experimental results appear in Section V. Finally, the paper is concluded in Section VI by summarizing our contribution.

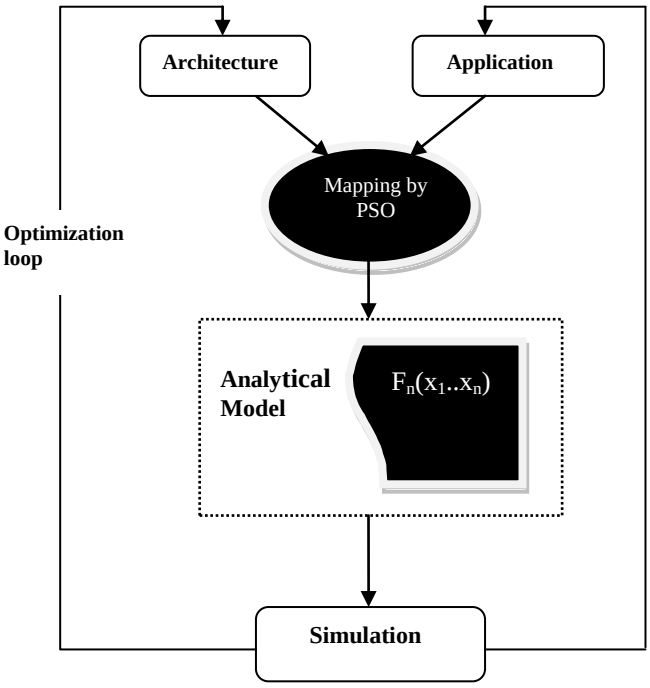

Fig. 1. Proposed framework for design space exploration.

II. Related Work

The design space exploration of NoCs is commonly formulated as a constrained optimization problem [2]. Therefore, performance analysis techniques that can be used in optimization loops are extremely important. There are different issues for design space exploration problems like: structure of switch (router), types of mappings between cores (IPs) and switches(routers), buffer space allocation etc. But the types of mappings between cores and switches are playing an important role in performance of NoC design. The design of regular NoC architecture has been proposed in [3][4]. Various approaches also have been reported for solving the topology selection and NoC mapping problem. In [5], Lei et al. proposed a two-step genetic algorithm that finds a mapping of cores onto NoC architecture to minimize the overall execution time. Authors in [6] introduce a linear-programming based NoC synthesis method. In [7] a BnB algorithm is used to map cores onto a mesh-based NoC architecture and find a flexible routing path with the objective of minimizing energy and satisfying the bandwidth constraints. Chan-Eun Rhee et al.[8] proposed a MILP algorithm for the many-to-many core-switch mapping for the NoC architecture with optimizing the power consumption.

Manuscript received January 11, 2012; revised February 29, 2012.

R. K. Jena is with the Institute of Management Technology, Nagpur, India (e-mail: rk_jena2@yahoo.com).

However, no methodologies can involve the design of link-load balance in the course of mapping and routing selection. In this paper, we propose a PSO based design methodology to minimize the energy consumption of NoC while guaranteeing the balance of link-load. In the other hand, the input buffer size also plays an important role in NoC Optimization. The work on buffer space allocation based performance evaluation uses either simulation or analytical models. Previously many researchers have addressed the problem in different context. The authors in [2] consider the buffer sizing problem and present a performance model based on queuing theory. However, the approach is applicable to only packet switched networks. In [9,10], the authors present two dynamic buffer allocation methods. In this work we proposed buffer allocation for wormhole routing for design space exploration of NoC systems.

III. Analytical Modeling

We have considered mesh based NoC architecture for the design space exploration problem. Many-many mapping between switches and cores has been considered in this work. Before deriving the model for the switch, the following are the assumption for the proposed model similar to the previous works [2].

1) Nodes are independently generate traffic according to a poisson distribution
2) Deterministic XY routing is used
3) Packet length is fixed (L) Flits. Each flit takes one cycle to advance from one to the next.
4) Buffer width is equals to the bit width of a flit
5) The local queue at the injection channel has infinite length
6) The packets are transferred to the local core as soon as they arrive at their destination.

The basic parameters used in this work are summarized in the Table I. According to the theory of finite queuing networks, every input channel buffer can be model as a M/M/1/K finite queue. So, the probability of input buffer being full in the direction D can be calculated as:

$$b_D = \frac{1-\rho_D}{1-\rho_D^{l_D+1}} \times \rho_D^{l_D} \quad (1)$$

$$\rho_D = \frac{\lambda_D}{\mu_D} \quad (2)$$

TABLE I: Parameters and their description

Var	Description
L	The size of packet(L flits)
T_H	The time needed by a router to process header flits
D	Direction, i.e North(N), East(E), South(S), West(W), NE
b_D	The probability of buffer at D input channel being full
ρ_D	Utilization factor of D input channel
μ_D	Packet service at of D input channel
l_D	Buffer size of D input channel
α	Packet injection rate
$d_{x,y}$	The probability of a packet generated by node 'x' to be delivered to node 'y'
T_D	The packet service time at D input channel
B_D	The blocking delay at D input channel
δ_D	The probability of a packet get blocked at the head of D input channel
ω_D	The mean waiting time of a packet at D input channel
λ_D	The packet arrival rate at D input channel
$X_{s1,d1,}$	The packet transmission rate from source(s1) to destination(d1)
P	Number of input channel associated with a switch (D≤ P= 5)
$\lambda_{D\to D'}$	The packet arrival rate which is arrive at D input channel and leaves at D' output channel

The packet arrival rate at 'D' input channel of a switch can be calculated as:

$$\lambda_D = \sum_{\forall s1} \sum_{d1} \alpha_{s1} \times d_{s1,d1} \Re(s1, d1, D) \quad (3)$$

Where $\Re(s1, d1, D)$ is the routing function, it equal '1' if the packet from source (s1) to destination (d1) uses the D channel; it equals '0' otherwise.

The packet service time at D channel can be calculated as:

$$T_D = T_H + M + \delta_D \times \omega_D \quad (4)$$

The contention matrix (F) is required to calculate (δ_D) is given below

$$F = \begin{bmatrix} 0 & f_{12} & f_{13} & \cdots & f_{1p} \\ f_{21} & 0 & f_{23} & \cdots & f_{2p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ f_{p1} & f_{p2} & f_{p3} & \cdots & 0 \end{bmatrix}$$

where f_{ij} the probability that a packet arrives at input of channel 'i' and leaves the switch through channel 'j'.

f_{ij} can be computed as :

$$f_{ij} = \lambda_{i\to j} / \lambda_i \quad (5)$$

λ_i can be calculated using equation "(3)"

$$\delta_D = \sum_{\forall D \neq D} f_{DD} \times \delta_{DD'} \quad (6)$$

$f_{DD'}$ is the forwarding probability of D→D', while $\delta_{DD'}$ is the probability that a packet forwarded from D to D' may got blocked.

The $\delta_{DD'}$ can be calculated as

$$\delta_{DD'} = f_{DD} \times \sum_{\forall i \neq D} f_{iD'} \quad (7)$$

The mean waiting time of each switch can be computed using M\M\1 queuing system as:

$$\omega_D = \lambda_D \times T_D^2 / (1 - \lambda_D \times T_D) \quad (8)$$

The network throughput is defined as the rate of which packet are delivered to the destination. The throughput is calculated at the traffic generation rate at which throughput saturated is found as:

$$\Gamma = \alpha_{min} \sum_{\forall s1,d1} X_{s1d1} \quad (9)$$

where α_{min} is the traffic injection rate at which whole network saturated.

IV. Design Space Exploration Methodology

The proposed framework is divided into two main tasks and is carried out in two phases. The first phase (Phase I) is deals with the mapping of the target application onto the NoC architecture. The second phase (Phase II) is concentrated on the analytical modeling and performance analysis of NoC. The size of the NoC architecture in the proposed framework is decided accordingly to the size of the task graph. It is very important to choose a proper size of the architecture, since it may lead to significant area overhead if the size of the architecture is not chosen properly. So, in order to map all the functioning blocks into the NoC architecture, the following condition should be satisfied. i.e *Total number of vertices* ($|V|$) *in the task graph* ≤ *Total number of vertices* ($|T|$) *in the NoC topology graph.* After application and architecture specification, the next step of our framework flow is the mapping of the task graph onto the topology graph. The details of the mapping step are described next.

A. Proposed Algorithm

The PSO scheme was designed to mimic the cooperation within a biological population, such as a group of birds or a swarm of insects [11]. Within the population, multi-dimensional particles, each a possible solution, are flown through the problem space, in search of optima. Each particle has its own velocity, which is determined by (1) the local best: the memory of the best solution it has obtained thus far and (2) the global best: the best solution found by the entire population. It has been shown that PSO is able to converge to global optima fast without being trapped in local optima, especially when the problem space is complex and irregular.

In the proposed PSO algorithm, the position of the *k*th particle is *m*-dimensional (*m* equals the number of Tiles in the NoC architecture graph) and expressed in a vector form:

$$Y^k = [y_1^k, y_2^k, \dots\ y_d^k, \dots\ y_m^k\ ]$$

where all $y_d^k \in [0,1]$. The component in dimension *d* of the position determines which IP (core) is assigned to Tile 'd' by a mapping process. The process is virtually mapping a continuous variable y_d^k to a binary variable x_{ij}, where *m* equals 3.

The proposed algorithm iteratively changes the particle positions as follows :

$$y_d^k(t+1) = y_d^k(t) + v_d^k(t+1) \qquad (10)$$

where $y_d^k(t)$ is the component in dimension d of particle k at iteration t; $v_d^k(t+1)$ is the velocity at iteration t +1, which is determine by

$$v_d^k(t+1) = wv_d^k(t) + c_1 r_1 \left(Lbest_d - y_d^k(t)\right) + c_2 r_2 \left(Gbest_d - y_d^k(t)\right) \qquad (11)$$

where w is the initial weight, $v_d^k(t)$ is the velocity in the previous iteration, c_1 and c_2 are learning constants, r_1 and r_2 are random numbers in the [0,1] interval. $Lbest_d$ is the component in dimension *d* of the local best. $Gbest_d$ is the component in dimension *d* of the global best. While the swarm size, *w*, c_1 and c_2 are specifiable algorithm parameters. The velocity is constrained within a specified bound to avoid vicious oscillation

$$v_d^k(t+1) = \frac{v_d^k(t+1)}{\left|v_d^k(t+1)\right|} v^{max}$$
$$if \quad \left|v_d^k(t+1)\right| > v^{max} \qquad (12)$$

where the velocity bound v^{max} is often smaller than the domain of the search space. The proposed PSO algorithm is not sensible to the scales of the objectives because it by no means calculates the distances between solutions. Thus its performance is independent from the choice of scales.

B. Analytical Modeling

After mapping of the application onto the NoC architecture, an analytical model is proposed to further explore the design space. The metrics of interest in this phase are queuing delay, buffer space allocation for each input channel, service time, latency etc. The main task of this phase is the buffer space allocation. Beside the mapping output, the other inputs to the buffer space allocation algorithm are system parameters and traffic parameters. The system parameter includes T_H and L, total buffer budget (B), size of mesh(N), and the traffic parameter includes d_{xy} and α . The pseudo-code of the algorithm is given below.

***Algorithm* Buffer allocation (T_H , L, N, B, d_{xy} ,α)**

Begin

1) Calculate λ_D, F, δ_D using eq(3),eq(5) and eq(6)
2) Build and solve the non-linear equation using eq(4)
3) Generate initial buffer configuration
4) Calculate b_D for each input channel using eq(1)
5) Find the channel having largest b_D
6) The buffer size of the bottleneck channel is incremented by one flit, while system free buffer is decremented by one flit
7) Repeat step 5 and 6 until system free buffer is 0

End.

The output of the buffer allocation algorithm is a buffer optimized NoC design. Finally the network throughput is computed for optimized design using "(9)".

V. Experimental Results

This section provides a detailed analysis of experimental results of the proposed framework. We analyzed the result in two parts i.e (1) result of PSO mapping and , (2) the result obtained by final analytical modeling. For PSO mapping algorithm a real bench mark VOPD is used. For the experimental purpose, the required bandwidth of an edge is uniformly distributed over the range [0,500MB/s] and the traffic volume of an edge is uniformly distributed over the range [0,1GB]. The result of our PSO formulation is denoted as MPO. The results of MPO are compared with NMAP algorithm [12] and MGAP algorithm [13].

The result obtained by our approach is compared with NMAP and MGAP. Fig. 2 shows that MPO saves around 12% of maximum link bandwidth as compare to NMAP. MPO saves around 50% of energy consumption in compare to NMAP and around 10% in compare to MGAP.

Fig. 2. Maximum link bandwidth and energy comparisons for VOPD

Whereas the results obtained by MGAP are nearly comparable to MPO as both algorithms are working on similar framework. But, the result of MPO is better than MGAP and MPO and takes less time in compare to MGAP.

In the second part, The analytical result obtained by our approach are compare against those obtained with the simulation platform OMNet++[14]. All the algorithms of the proposed framework is implemented in C++. Some parameter used in simulation are : L=16, a data packet is divided into 16 flits; T_H =2, the router needs 2 clock cycle to make routing decision for the header flit. Initially the size of the mesh is 4. System total buffers are 240 flits, i.e each used channel is assigned 5 flit larger buffers. For simplicity, initially each node has same packet injection rate. In our experiment, we applied our algorithm to two types of traffic models, i.e uniform random traffic model and non-uniform traffic model. In uniform traffic model, a core sends packet to any other core with equal probability (0.0666). Under non-uniform traffic model, a hot core (core having maximum communication) is selected. The hot core receives extra proportion traffic along with normal traffic. In this paper the probability is 0.2 for the hot core. Fig. 3 shows comparison of the average packet latency result of our analytical method to the simulation result obtained from OMNet++ for uniform traffic model. We observed from the result that the latency value estimated by our approach is follow closely to the simulation result. More precisely the error rate up to packet injection rate (0.012) is below (4-6)%. In non-uniform traffic condition the result variation is more in compare to the uniform traffic condition as shown in the Fig. 4.

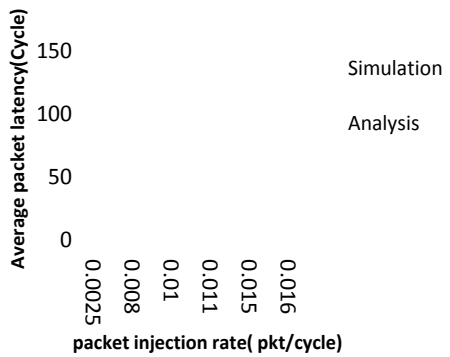

Fig. 3. Average latency (Simulation Vs Analysis) using uniform traffic

From the Fig. 5, it is clear that the variation in network throughput obtained using our proposed analytic framework to the simulation result in case of uniform traffic is very negligible. But, in case of non-uniform traffic the variation is below (3-5) %.

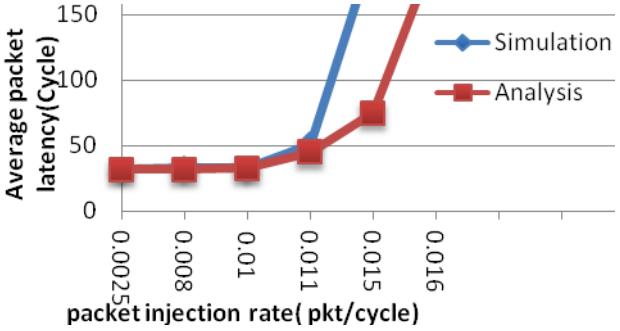

Fig. 4. Average latency (Simulation Vs Analysis) using non-uniform traffic

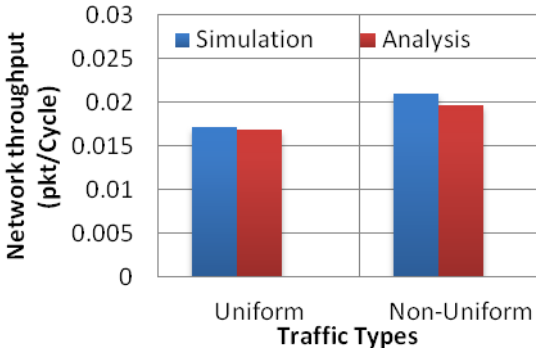

Fig. 5. Network Throughput (Simulation Vs Analysis)

VI. Conclusion

In this paper, we first proposed a multi-objective novel PSO based heuristic algorithm for the design of link-load balance and low energy mesh based NoC architecture. We use particle swarm optimization algorithm to explore the large search space of NoC design effectively. The proposed algorithm optimizes the NoC energy consumption and maximum link bandwidth. The performance of the algorithm is evaluated in comparison with other genetic algorithm based mapping algorithm. The experimental results indicate that the proposed PSO algorithm is a viable alternative for solving the mapping problems for NoC.. Secondly, we presented a NoC design space exploration framework for allocation of switch input buffer space using analytical model. Our approach provides aggregate performance metrics such as average latency and throughput.

References

[1] L. Benini and G. D. Micheli, "Networks on chips: a new soc paradigm," *Computer*, vol.35, pp.70-78, 2002.

[2] J. Hu and R. Marculescu, "Energy- and performance-aware mapping for regular NoC architectures," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 24(4), 2005.

[3] W. J. Dally and B. Towles, "Route Packets, Not Wires:On-Chip Interconnection Networks," in *Proc of DAC'01*, New York, USA, 2001, pp. 684–689.

[4] S. Kumar, A. Jantsch, "A Network on Chip Architecture and Design Methodology," in *Proc. of VLSI'02*, Pittsburgh, Germany, 4, 2002, pp. 105–112.
[5] L. Tang and S. Kumar, "A Two Genetic Algorithm for Mapping Task Graphs to a Network on Chip Architecture," in *Proc. of DSD'03*, Antalya, Turkey, 2003, pp. 180–187.
[6] K. Srinivasan and K. S. Chatha, and G. Konjevod, "Linear programming based techniques for synthesis of network-on-chip architectures," *IEEE Transactions on VLSI Systems*, vol. 14, no. 4, pp. 407–420, 2006.
[7] J. Hu and R. Marculescu, "Exploiting the Routing Flexibility for Energy/Performance Aware Mapping of Regular NoC architecture," in *Proc. of DAT'03*, Munich, Germany, 2003, pp. 1068–1093.
[8] C.-E. Rhee, H.-Y. Jeong, and S. Ha, "Many-to-Many Core-Switch Mapping in 2-D Mesh NoC Architectures," in *Proc of ICCD'04*, San Jose, CA, USA, 2004, pp. 438–443.
[9] R. Dobkin, R. Ginosar and I. Cidon, " QNoC asynchronous router with dynamic virtual channel allocation," in *Proc. First International Symposium on Networks-on-Chip*, New Jersey, USA, 2007, pp.218-218.
[10] Y. Tamir and G. L. Frazier, "Dynamically-allocated multi-queue buffers for VLSI communication switches," *IEEE Tran on Computers*,vol. 41, pp.725-737, 1992.
[11] J. Kennedy, R. C. Eberhart and Y. Shi, *Swarm intelligence*, Morgan Kaufmann Publishers, San Francisco, CA, 2001.
[12] S. Murali and G. D. Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," in *Proc. of Design, Automation, and Test in Europe, IEEE Computer Society*, Feb. 16–20 2004, pp 896–901.
[13] R. K. Jena and G. K. Sharma, "A Multi-Objective Evolutionary Algorithm Based Optimization Model for Network-on-Chip Synthesis," in *Proc. of 4th International conference on IT: New Generation*, April, 2-4, Las Vegas, Nevada, USA,2007 pp. 977-983.
[14] OMNet++ discrete event simulation system user manual [Online]. Available: http://www.omnetpp.org/, 2005.

R. K. Jena is currently Associate professor at the department of Information Technology, Institute of Management Technology, Nagpur, India. He received his M.Tech degree in Computer Science and Enginering from G. J. University, Haryana, India in 1997 and PhD from ABV-IIITM, Gwalior, India in 2010. His current research interest includes CAD for VLSI, Computer Algorithm, Quantum Computing and Business Intelligence.