International Journal of Electronics Communication and Computer Engineering
Volume 5, Issue 5, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Presentation of an Algorithm Configuration for Network-on-Chip Architecture with Reconfiguration Ability

Ali Asghari
Shafagh University, Tonekabon, Iran
Email: aliasghari@shafagh.ac.ir

Ali Abbass Zoraghchian
Allameh Mohaddes University
Noor, Iran
Email: zaliabbass@yahoo.com

Mohammad Trik
Sardasht Islamic Azad University,
Sardasht, Iran
Email: trik.mohammad@gmail.com

***Abstract* – Due to the challenge that the number of cores are combined on a single chip, the network on chip (NoC) has gradually become a popular solution. And recently, researchers have focused on improving the performance of NoC chips to achieve good performance. In this paper, we present an Algorithm Configuration for Network-on-Chip Architecture with reconfiguration ability for designing NoC with specific use. NoC architecture with reconfigurable capability reduces the complexity of the design and makes the layout of the NoC relatively more flexible compared to the layout of creating the topology map and the mapping design. Moreover, our Algorithm Configuration is for the optimized networks with better performance. NoC architecture with reconfigurable capability can be determined with a proper size for a specific purpose, and it can be configured according to the communication relation in order to ensure that the final system is optimized. A simulator with a proper cycle for imitating the three networks was deployed, and it was designed with our plan and two others methods for the same performance in the same environment. The results show that our system works efficiently.**

***Keywords* – Reconfiguration Ability, Design of NoC, Building Topology, Mapping.**

I. Introduction and Previous Works

With the rapid development of the semiconductor industry, the number of transistors integrated on a single chip will reach the billions. For designing the chip, the basic problem is how to achieve effective connectivity between cores. With the increasing integration of semiconducting IP cores on a single chip, indicated its traditional structure based on the bass limitations in scalability, reusability, and time synchronization [1]. To solve this problem, NoC which presents concepts and technologies of computer networks for designing of the chips is proposed as an alternative binding structures for designing of the chip [2]. NoC has better scalability and reusability, and can solve problems of time synchronization. These advantages have turned NoC into a solution to the challenges of communication in SoC. Furthermore, NoC can achieve the improved performance of network throughput, low latency and lower power consumption.

There are two traditional ways for designing specific applications of NoC. One of these ways is to create topology at first, and determining the position of all the parts of network on the chip after creating topology. Creation of the topology is for obtaining connectivity between nodes. Another way of is mapping which is based on a fixed grid structures such as two-dimensional mesh, torus, and so forth. In this research, we name the first way, design of creating topology map and refer to the second way as layout of mapping. If we use the design of creating topology map for designing NoC, we should optimize the network structure in two steps. Creating topology and planning. Although we can gain an optimized structural, its complexity is usually high. If we use the mapping design, the structure is limited owing to the determined connectivity of a fixed network structures.

To achieve NoC architectures with high performance, a lot of researches have been done, A [3] network- creating algorithm called Pareto Simulated Annealing was proposed by J. Palermo and others. This efficient approach chose the enacted network structure among transferring structures which is obtained by adding some links to the circular structures for the direction of Spidergon structure. In table 5, an approach called NoCOUT can attain structures which supports switching and point to point connections which are caused by adding network interfaces to the structures which are emerged through a process proposed by J Chan and others. Moreover, the obtained structures are also optimized. In [6] topology formation algorithm was proposed by N. Choudhary and others based on genetic algorithm. This proposed algorithm introduces initial modification and concentration operations in order to present its new productions, and finally achieves its best output. In table [7] the notion of dependency graph message by *S. Dnizyak* and others was presented, and one algorithm was proposed in order to generate network structures without the involvement according to dependency graph message.

To get rid of the deficiencies described above, in this study, we propose an Algorithm Configuration for Network-on-Chip Architecture with Reconfiguration Ability algorithm which is to be re-configured according to different applications.

The aims of Algorithm Configuration are increasing the network throughput,reducing latenncy and consuming NOC power with specific applications truly improve the performance.

II. The Designing of NoC based on Reconfigurable NoC-based Architecture

In this section, we introduce a reconfigurable NoC architecture. Then, we briefly introduce the design process

International Journal of Electronics Communication and Computer Engineering
Volume 5, Issue 5, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

based on a reconfigurable network architecture. And finally, we discuss the final stages of the Algorithm Configuration to complete NoC design for specific applications.

A. NoC Reconfigurable Architecture:

NoC Reconfigurable Architecture is presented in Figure 1. Our example is a 4 × 4 grid size. This structure is based on two-dimensional mesh [8] that its IP cores are connected to fixed operators. The difference is that each operator holds 8 ports which makes it possible operator's connectivity to more than 4 cores of IP. Accordingly, as shown in the Figure, one IP core gets.

Connected to all its adjoining operators. Besides, the multiplexer is introduced to the links connected to IP cores and operators. Therefore, As the operator of one core IP is connected in order to change according to a different connectivity relation which is presented as the result of Algorithm Configuration. After obtaining the connectivity relation between the IP cores and operators, we can set all multiplexers so they can select the operator, and each IP core is connected and the final structure of the network is obtained.

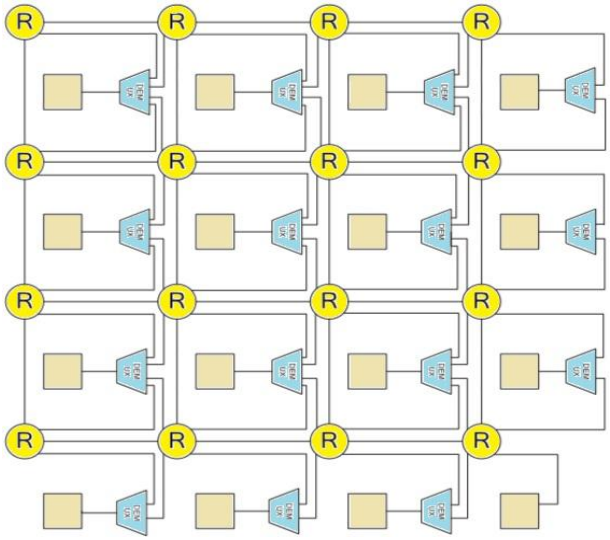

Figure 1 :NoC Reconfigurable Architecture

B. Design Process:

Our process for designing NoC is divided into three main steps:

B-1) specify the size of NoC reconfigurable architecture

B-2) Specify the location of each IP core and operator and the IP core connected to it.

B-3) Reconfigure NoC Reconfigurable Architecture according to the obtained data in the second step.

Input of the design process is the adjoining matrix which is obtained by one application and provides the data which IP cores would interact with other IP cores, and the size of the relation between two IP cores is distinct. And according to the number of IP cores in a specific application, we can determine NoC reconfigurable architecture which is appropriate for this purpose.

Then, According to the provided information by the adjoining matrix, we run the proposed Algorithm for changing the shape. We therefore attain the connectivity relation between the IP cores and the operators and the position of each IP core in a reconfigurable NoC architectures. Finally, we reconfigure NoC reconfigurable architecture in such a way that the multiplexer can be configured according to the information obtained in previous stages we set.

C. The Algorithm Configuration for NoC Reconfigurable Architecture:

1) Description of the problem:

A functional relation graph (ACG) $= G(V, E, W)$ is a weight- oriented graph consisting of a set of vertex V and a set of E –directed edges. Each edge $v_i \in V$ denotes a core. Each directed edge $e_{i,j} \in E$ implies that there v_i is a relationship v_j between edge and edge v_j And the weight of the edge $e_{i,j}$ implies the size of the relation between edges $e_{i,j}$ and v_i and v_j .ACG is applied for an activity [9], called video Objective Plane Decoder (VOPD) which is shown in Figure 2.

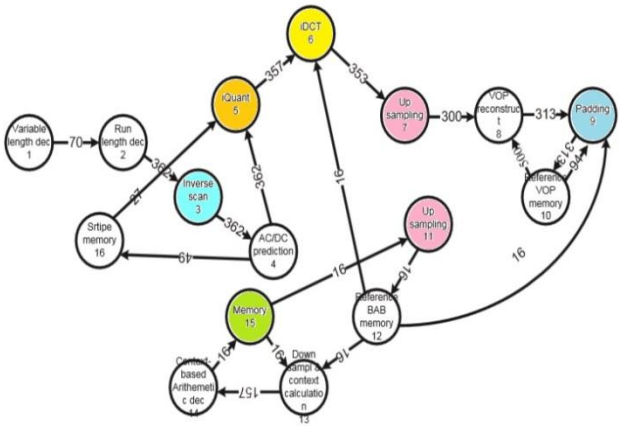

Fig.2. ACG for VOPD

A topology graph (TG) $TG = G(R, P, L)$is an undirected graph consisting of a set of R operators, a set of IP cores and a set of L links. Each IP core $p_i \in P$ is connected to a carrier through a link, and every carrier is connected to several operators through other links. One example of TG for VOPD is shown in Figure 3.

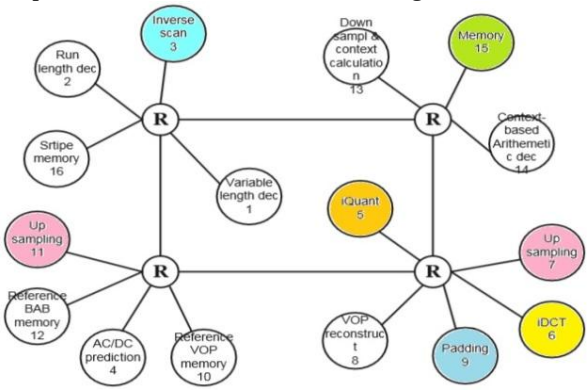

Fig.3. TG for VOPD

Regarding the above definition, the design of the network is described as: According to the connectivity relation and connectivity sizes in ACG, is obtained to TG satisfaction: 1) the performance of TG should be maximum 2) the average delay of data transmission should be minimal 3) Power consumption should be minimal.

International Journal of Electronics Communication and Computer Engineering
Volume 5, Issue 5, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

In this study, we aimed to lower power through Algorithm Configuration for NoC application. And the proposed energy consumption model is applied for calculating the power consumption of the network. In Figure 10, the consumed E power for data transmission can be calculated as.follows:

1) $E = E_s + E_l$

Which E_s, E_l implies the power consumption by carriers and links in a r elative way.

The $E_{bit}^{i,j}$ consumed power due to transmitting one bit of operator r_i to operator r_j is as follows:

2) $E_{bit}^{i,j} = (h_{i,j} + 1) \times E_{sbit} + h_{i,j} \times E_{lbit})$

Which $h_{i,j}$ implies on hop between the operators and r_i and r_j , E_{sbit}, E_{lbit} implies on the consumed power by each operator on each link. The total network power consumption is as follows:

3) $E_t = \sum_{i-1,j-1}^{i-N_r,j-N_r} C_{i,j} \times E_{bit}^{i,j}$

which $C_{i,j}$ implies the relation of the carrier r_i to carrier N_r and r_j implies on the number of the operators.

2) Algorithm Configuration

As it is mentioned above, two main stages before configuration of Reconfigurable NoC architecture are:

1) Determining the size of NoC Reconfigurable Architecture

2) Determining the connectivity relation between operators and IP cores and placing each IP core in its right place.

In the first phase, select an NoC architecture with reconfigurable ability for a specific purpose. We use the principle which the number of candidate's opportunities for IP cores in NoC reconfigurable architecture should be as near as possible to the number of IP cores at work. In addition, the sizes of the two dimensions should be very different. This thought is carried out like a dummy code shown below in Figure 4:

```
Initialize N← the number of IP cores, x ← 1, y ← 1, g_min ← 1;
While (g_min == 1) {
    For x = 1 to N{
      For y = 1 to N{
       If N == x*y && x >= y && (x-y)/x <= 1/3 && (x-y)/x < g_min then
          g_min ← (x-y)/x;}}
    N←N+1;}
Return(x, y);
```

Fig.4. Pseudo code to specify the size of the reconfigurable ability for NoC architecture

In the pseudo code, inequality is applied to reduce the distance between the sizes of two dimensions. However, we can also determine the distance according to different needs. Since the on-chip resources are limited, we can achieve the aim to increase the efficiency of resource use of the chip through this principle.

In the second step, we investigated the connectivity relation between IP cores and operators, and are looking for a right place for each IP core. Since NoC reconfigurable architecture is based on two dimensional mesh in which specific areas are for IP cores and operators which there is no need for mapping for a specific place. And at the same time, complexity of the design reduces. For improving the performance of the network, we do our best in order to connect IP cores which always had connection with each other through one operator. We introduce the category concept for expressing the IP cores which have connection. And we place the group in the same category. After dividing all the IP cores into several categories according to the proposed algorithm, we determine the method of placing the categories based on connectivity relation among categories in order to reduce the total power consumption of the network. The dummy code of Figure 5 indicates the division process of IP cores.

```
Initialize N← the number of IP cores;
For i = 1 to N{
    For j = 1 to N{
    If n_i communicates most frequently with n_j then
    Create IP pair(n_i,n_j);}}
N_IP_pair = N;
For i = 1 to ⌈N_IP_pair /2⌉{
    If IP pair(n_i,n_j) and IP pair(n_j,n_i)exist at the same time then{
      existing cluster A ← IP pair(n_i,n_j) ;
      delete IP pair(n_i,n_j) and IP pair(n_j,n_i);}}
Put the existing clusters in order as the intra-cluster communication volume
decreases;
Do{
    For(all left IP pairs(n_i,n_j)){
      If  n_i has been the element in one of the existing cluster A then{
          M←the number of IP cores in cluster A;
          M←M+1;
          If  M <= 4 then
              delete IP pair(n_i,n_j),A←A ∪ {n_i};
          If  M > 4 then{
              delete IP pair(n_i,n_j);
              founding one new existing cluster B ← {n_i};}}
      If  n_j has been the element in one of the existing cluster A then{
          M←the number of IP cores in cluster A;
          M←M+1;
          If  M <= 4 then
              delete IP pair(n_i,n_j),A←A ∪ {n_j};
          If  M > 4 then{
              delete IP pair(n_i,n_j)
              founding one new existing cluster B ← {n_j};}}}}
While(there are IP pairs left)
Return(several clusters);
```

Fig.5. Pseudo code to divide IP cores into clusters

After IP cores are divided into several categories, we place each core of our IP in a suitable location in the NoC reconfigurable architecture and connect each IP core to one appropriate operator in a way that the final network becomes optimized. With the above steps, the communication within the cluster is the main connectivity. To reduce power consumption, we should connect IP cores in the clusters which have the highest correlation within clusters in the first place. The selected operator to connect the IP cores is the central operator. The definition of central operator for networks in different size is shown in Table 1. Then, we focus on the relationships within the group. We look for a category which is always in connection with proper categories, and determine the position of each IP core in category and operator. Each IP core is connected in order to reduce power consumption. Finally, we repeat the above steps till all the categories are placed.

Table 1- definition of Central Operator

Properties x and y (x and y denote the sizes of two dimensions)	The central operator position (A implies B implies row-column grid)
is even X ، y is even	$A = \mathrm{x}/2, \mathrm{B} = y/2$
X is even ، y is odd	$A = \mathrm{x}/2, \mathrm{B} = (y+1)/2$
X is odd ،y is even	$A = (\mathrm{x}+1)/2, \mathrm{B} = y/2$
X is odd ،Y is odd	$A = (\mathrm{x}+1)/2, \mathrm{B} = (y+1)/2$

III. Results

As we have previously shown, we initially create the topology, and use the planner to specify the location of IP cores, operators and links on the chip. But network optimization in these two steps increases the complexity of the scheme. Another assumption is the mapping which is related to constant connectivity relation. In our proposed plan, we combine the topology creation with NoC reconfigurable architecture which their locations are confirmed for IP cores and operators, and the connectivity relation is changing slowly. With this plan, we can be released from the same mapping and fixed structures and create optimized structures.

We applied the proposed design for VOPD and use a simulator with accurate cycle for evaluating the performance of the final network structure. In order to prove that we can achieve better performance by our access. We compare the results with the network structures which have been developed in two other methods. (Plan B and C) for VOPD and simulate these structures in similar simulation environment. The results are shown in Fig. 6.

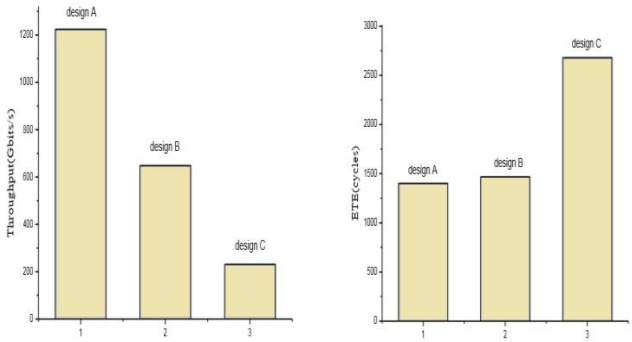

Fig.6. Throughput and ETE of different schemes on VOPD

Figure 6 (a) and (b) shows the performance of the different networks and ETE in a relative manner. The first column of the two first figures presents the performance and ETE network which we attain. The other two columns indicate the performance of the other two methods. We can conclude from these two figures that the obtained structure of our assumption is obviously optimized

IV. Conclusion and Future Work

In this study, we propose Algorithm Configuration for Network-on-Chip Architecture with Reconfiguration Ability to complete a specific task. This lowers the complexity of the design and remove the limitations of constant connectivity relation at the same time. Compared with two other design methods, Our plan can achieve better performance. To continue the research, we focus on NoC architectures with reconfigurable capabilities to improve the performance in the future.

References

[1] W. Dally and B. Towles, "Route packets, not wires: on- chip interconnection networks," Proceedings of the Design Automation Conference, pp.684-689,2001.

[2] Chen J, Sheu S, Yang C, "A new multichannel access protocol for IEEE 802. 11 ad hoc wireless LANs," 14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications (PIMRC), vol. 3, pp.2291- 2296, 7-10 Sept. 2003.

[3] G. Palermo, C. Silvano, G. Mariani, R. Locatelli and M. Coppola, "Application-Specific Topology Design Customization for STNoC," Proceedings of 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools, pp.547-550,29-31 Aug. 2007.

[4] M. Coppola, R. Locatelli, G. Maruccia, L. Pieralisi and A. Scandurra, "Spidergon: a novel on-chip communication network," International Symposium on System-on-Chip, pp.15,16-18 Nov. 2004.

[5] J. Chan and S. Parameswaran, "NoCOUT: NoC topology generation with mixed packet-switched and point-to-point networks," Proceedings of Asia and South Pacific in Design Automation Conference (ASPDAC), pp.265-270,21-24 March. 2008.

[6] N. Choudhary, M.S. Gaur, V. Laxmi and V. Singh, "Genetic algorithm based topology generation for application specific Network-on-Chip," Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), pp.3156-3159, 30 May-2 June. 2010.

[7] S. Deniziak and R. Tomaszewski, "Contention-avoiding custom topology generation for network-on-chip," Proceedings of 12th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS'09), pp.234-237,15-17 April. 2009.

[8] S. Kumar, J. Axel, S. Juha-Pekka, F. Martti, M. Mikael, O. Johny, et al. "A network on chip architecture and design methodology," Proceedings of IEEE Computer Society Annual Symposium on VLSI, pp. 105 –112, 2002.

[9] V. Dumitriu and G. N.Khan, "Throughput-Oriented NoC Topology Generation and Analysis for High Performance SoCs," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 17, pp. 1433-1446,2009.

[10] T.T. Ye, L. Benini and G. De Micheli, "Packetized on-chip interconnect communication analysis for MPSoC," Proceedings of Design, Automation and Test in Europe Conference and Exhibition, pp. 344-349, 2003