International Journal of Engineering & Technology, 7 (2) (2018) 417-420

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i2.9760
Research paper

Reconfigurable on-chip communication link for efficient communication

S. Beulah Hemalatha [1] *, Dr. T. Vigneswaran [2]

[1] Research Scholar, ECE Department, Bharath University, Chennai 600073, India
[2] Professor, School of Electronics, VIT, Chennai campus, India
**Corresponding author E-mail: beulah.ecevlsi@gmail.com*

Abstract

Application specific reconfiguration of On-chip communication link is a fast growing research area in system on chip (SoC) based system design. Optimization of the communication link is important to achieve a trade-off between efficient communication and low power consumption. So achieving both efficient communication and low power consumption requires a special optimization mechanism. Such Optimization problems can be solved using a genetic algorithm. Here, in this paper genetic algorithm based On-chip communication link reconfiguration is presented. The algorithm will optimize efficiency of communication link with constrain of low power consumption. The parameters involved in power consumption and efficient communication link are coded in the chromosomes. By evolutionary iteration the optimal parameters of the communication link are derived that is used for the communication link successfully in the simulated system. The performance of the simulated system is analyzed which shows the out performance of the proposed system.

Keywords: *Genetic Algorithm; On-Chip Communication Link; Optimization; Reconfiguration; System on Chip (SoC).*

1. Introduction

Application mapping is one of the key complications of Network-on-Chip (NoC) design. It maps the application cores to the processing elements of the NoC topology. This work presents a novel approach for NoC application mapping, [1] which uses adaptive genetic algorithm (AGA) in the mapping. The projected approach adaptively varies the probabilities of crossover and mutation operators in genetic algorithm, aiming to diminish the overall communication cost of NoC. Experimental results show that the projected approach reductions the communication cost by 3% to 7% on average, compared to the existing approach using Standard Genetic Algorithm (SGA).

In this paper, a new design technique of Genetic Algorithm that can synthesize an application specific NoC topology and route the communication through that interconnection network is implemented. This algorithm will work on the system-level of the system on chip (SoC) and it also reduces power consumption in the physical channels and the routers. [2]Genetic algorithms have been very effective for benchmark circuit test generation. In this paper new method for network routing in NoC architecture using GA optimization is implemented. This network routing architecture handles accurate localizations of the faulty parts in NoC. The proposed NoC is based on new error detection mechanisms which is best suited for dynamic NoCs, where the number and position of processing elements or faulty blocks vary during runtime. The design technique solves a multi-objective problem of reduced large search space, low power consumption. Genetic algorithm provides optimum mapping in NoC architecture and measures faulty blocks during run time more effectively.

As the number of transistors on a chip continues to rise, the traditional infrastructure cannot handle concentrated communications on chip efficiently. Thus Network on chip (NoC) was suggested as a promising alternative due to its performance scalability and communication parallelism [3], [4]. Typically, a NoC is designed by a global network which is composed of a set of routers and computational or memory resources which are connected to routers. Since the global network plays a main role in the NoC design, lots of researchers concentrate on this aspect. However, according to the NoC evolution, which forecasts that will be possible to integrate hundreds and even thousands of processing cores in one NoC in a few years, the conventional two Dimensional (2D) interconnections are considered as an unproductive architecture due to several insurmountable problems like the global wire length and packet transfer latency. Temporarily, relying on its potential to improve performance of the chip, its functionality, and device packaging density [5], three Dimensional (3D) NoC is emergent as a novel approach. Considering the advantages of aforementioned technologies, 3D NoC is proposed as a combination of both [6] to enhance architectural advantages of NoC.

Many challenges have to be challenged during the design space exploration of 3D NoC, and among all these issues, mapping is one of key steps which may have great influence on the total system performance at the commencement of NoC design. Usually, at initial stages of a new design, the NoC communication backbone (e.g., the topology and the size) has to be decided first and then a set of IP cores are chosen depending on various kinds of tasks. After those tasks are assigned to different IP cores and the Application characterization graph Then the mapping problem can be defined as how to topologically place the certain set of IP cores onto the resource nodes of the network in such a way that the desired parameters can be optimized [7]. In this work, we assume that steps before mapping are already completed and we mainly focus on the mapping problem.

As silicon [8] technology possesses scalability, it is technically possible that the entire set of complex systems can be integrated on the same silicon die. Because of this idea, Systems on Chips (SOC) have become integrally complex and diverse .They are integrated with multiple processors of various types (RISC, DSP; ASIC, for examples) with devoted hardware or reconfiguration and peripheral.

Old-style concept of computer networking component interconnect-based routers, switches is accepted to extend the integration and perform the design. Network on chip (NOC) or plus generally MPSoCs are relatively aim new approach to integrated circuits on a platform SoC.[9] Then MPSoCs and NoC are broadly employed in embedded systems (for example automotive control engines, mobile phones, etc..) where, once they are set up in the field, they continue to run the same group of applications. Here, we concentrate on mesh-based NoC architectures, where the resources contact with one another through a mesh of switches that are used for routing and buffering the data. A resource can generally be any of the following core: a general processor GP, a memory, DSP, an FPGA. From a layout perception, a two dimensional mesh network topology is the easiest and the links connecting the processing core and switching elements does not depend on the size of the system.

However, routing in such a mesh network is simple. It results in potentially small switches, wide bandwidth, total scalability and short clock cycles[10]. The most difficult task in this NoC topology is the mapping of the core on the mesh so as to adjust certain performances indexes like speed, power, performance and area. Mapping is a quadratic assignment problem that is identified as NP-hard.

This mapping problem rises exponentially with the network complexity based on certain resources, interconnection, communication, and tasks. It is important to state the different methods to identify a mapping that will suit the required performance indexes. Also, the techniques have to handle all possible architectural mapping replacements with a multiple criteria exploration .In fact, the optimization factors are many and mostly they will always be in contrast with one another. So this gives rise to multiple solutionsfor a single problem ,i.e single mapping ,but it includes a set of equal (i.e not dominative) architectural alternative solutions under all possibilities, each of the solution showing a different trade-off among the objective values or the parameters which is to be optimized commonly called the Pareto Set [11].

Then the crucial task for MPSoCs is to minimize the total energy consumption. Initially, a well-structured task graph is formulated, which a fixed acyclic graph is representing a functional abstraction of the application which will work on the MPSoC. Every task is described by the count of clock cycles needed for the implementation of the task. Obviously the time period taken for each task and the energy dissipated for executing it relies on the clock frequency applied throughout the execution of the task.

This is a very difficult problem. Because we have to solve two problems together i.e, the scheduling and assigning of tasks to the processors and at the same time estimate the best path allocation previously mentioned as Network Assignment [12-14].

2. System model

The On-chip communication link consist of reconfigurable serial and parallel link .Optimal packet size, selection of serial and parallel link, clock rate and parallel link size are the parameters which are optimized under the genetic algorithm. Fig 1 shows the communication link of the proposed system design.

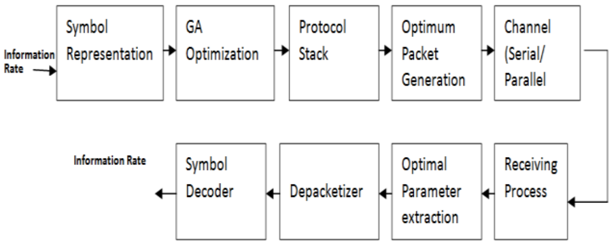


Fig. 1: Communication Link for the on Chip Communication.

The information bit are converted to the symbol by symbol generation unit which is typically low power encoding of the bits .The symbol generated is given to the Genetic algorithm optimizer to analyse the symbol and generate the optimal parameters .The optimized parameter is given to the protocol stack which generates the symbol for protocol stack with generation of header and footer. The output is given to the optimum packet generation which puts the symbols, header and footer information in to the optimal packet .The optimum generated packet is given to the channel which is the reconfigured serial or parallel bus based on the output of the genetic algorithm . The packets are received by the receiving process. The received packets are extracted by optimal packet extractor .The extracted packets will be de-packetized by de-packetizer then it will be decoded by symbol decoder to get the information bits.

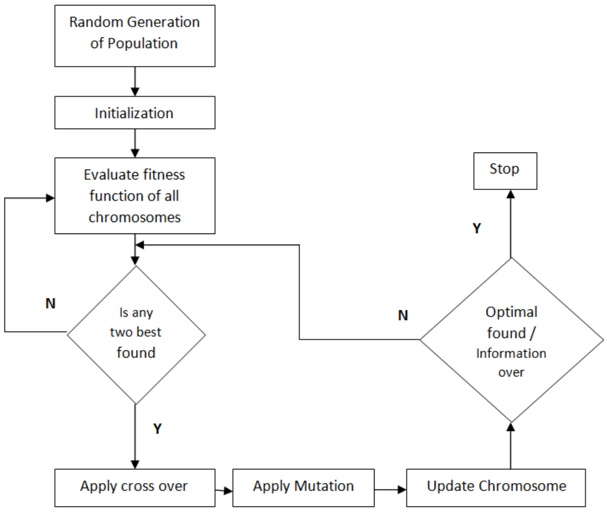

Random Generation of Population
Initialization
Evaluate fitness function of all chromosomes
Is any two best found
N
Y
Apply cross over
Apply Mutation
Update Chromosome
Optimal found / Information over
N
Y
Stop

Fig. 2: Flow Chart of Genetic Algorithm.

Fig 2 shows the genetic algorithm flow chart. Random generation of 50 chromosomes is done by random bit generator .The validation of the random bit are done at initialization process and the values are initialized by the random combination of the communication link. The design of the fitness function is done from the problem of the optimization which is given below.

Optimization problem:

min p_t

Stated that, $r_d \geq r_{min;}$

$$ber \leq BER_{MAX} \quad (1)$$

Above optimization problem minimizes power consumption by keeping the data rate of the On- chip communication link more than or equal to the minimum data rate and the bit error rate of the link is lesser than that of maximum amount the application can tolerate.

Fitness function:

$$f(n)=\alpha\ f_{lp}\ (n)+(1-\alpha)f_{cf}\ (n) \quad (2)$$

Equation 1 gives the fitness function that will be used to evaluate the score of the each chromosome. α is the scale value to give importance to low power requirement and communication efficiency. f_{lp} (n) is the fitness of the power consumption evaluation and f_{cf}(n) is the communication efficiency fitness function.

All the random chromosomes are applied to the fitness function of the above and the best chromosome which provide maximum or minimum value is selected as current parent chromosome which is applied to cross over and mutation process to generate next generation chromosome .If the next generation chromosome satisfies the optimal requirement then the algorithm is terminated else it is repeated by including the new chromosome.

Chromosome design:

An 8 bit chromosome is designed to code the communication link parameters. Table 1 provides the bit allocation on the chromosome and range of value considered for each parameter. Figure 3 shows the chromosome structure which is used to code the parameters.

Table 1: Bit Allocation of the Chromosome for the Parameters

Parameter	Range values	Number of bits
Serial or parallel link selection	0 –serial or 1-parallel	1
Number of links in parallel link		2
Packet size		2
Clock rate		2
-	0-Low;1-High	1

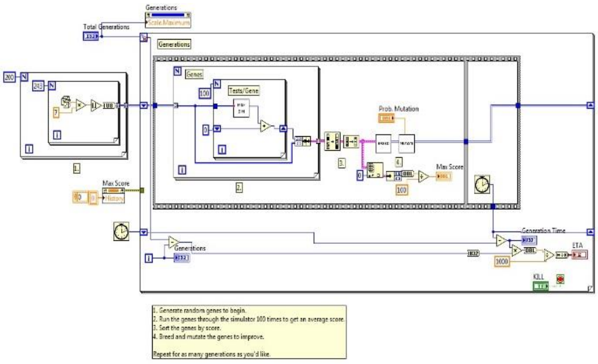
Serial /parallel	Number of links	Packet size	Clock rate	BER

Fig. 3: Chromosome Structure.

Fig 3 gives the structure on which the chromosome is coded using 8 bits from Least Significant Bit (LSB) to Most Significant Bit (MSB).MSB has the bit for the selection of serial or parallel link. The LSB has the BER level demand of the application.

3. Results and discussion

The entire system is designed and tested on LABVIEW FPGA on compact RIO platform. Fig 4 gives the LABVIEW code of the core genetic algorithm.

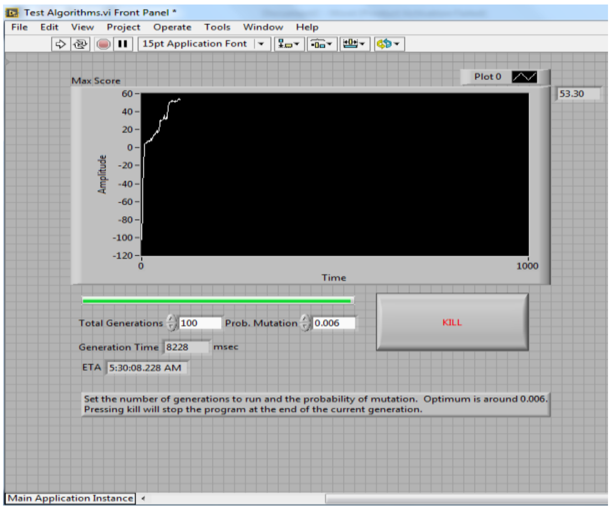


1. Generate random genes to begin.
2. Run the genes through the simulator 100 times to get an average score.
3. Sort the genes by score.
4. Breed and mutate the genes to improve.

Repeat for as many generations as you'd like.

Fig. 4: Genetic Algorithm in Labview.

The code is implemented in four steps .Step1 generates random genes to be included in the evaluation process. In step 2 the chromosomes are evaluated in 100 iterations. After 100th iteration the best chromosome is selected based on the maximum score of the evaluation by mutation and cross over.



53.30

Total Generations 100 Prob. Mutation 0.006

Generation Time 8228 msec

ETA 5:30:08.228 AM

Set the number of generations to run and the probability of mutation. Optimum is around 0.006. Pressing kill will stop the program at the end of the current generation.

Fig 5: Genetic Algorithm Screen Shot.

Fig 5 gives the screenshot of the execution of genetic algorithm on simulation mode in i3 Pentium PC. The figure shows the maximum score of chromosome after 100thiteration is 53.30.Each generation taking 8228 millisecond to compute the score.

Table 2: Resource Utilization for the Genetic Algorithm

Resource	Occupied	Available	%
Quadrature encoding application			
Slice register	746	30064	2.5
Total Logic slice	262	3758	7
LUT	546	15032	3.6
BLOCK RAM	0	52	0
DSP48S	0	38	0
With genetic algorithm			
Slice register	1455	30064	4.8
Total Logic slice	544	3758	14.5
LUT	1662	15032	11.1
BLOCK RAM	0	52	0
DSP48S	0	38	0

Table 2 shows the resource utilization of implementation on quadratic encoder application. It shows the resource utilization of the implementation after and before introducing the genetic algorithm based selection scheme. From the table we can observe that the addition of genetic algorithm increases only a minimum amount of resources.

4. Conclusion

Efficient communication link reconfiguration system using genetic algorithm based optimization is designed and simulated on the FPGA target. The computational complexity of the genetic algorithm and resource utilization summary of the proposed system is analysed. From the above simulation results, the functioning of the system is tested by applying some random inputs. Both the efficient communication and low power consumption was achieved by the proposed algorithm.

References

[1] Frank Wang, ZhibiaoZhang, (2014), “An adaptive genetic algorithm for mesh-based NoC application mapping”, Indonesian journal of Electrical Engineering and Computer science, 12(11), 7869-7875.

[2] N.Ramalakshmi and N.Hemalatha (2015) “Designing genetic algorithm for network-on-chip architecture” International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE), 12(4), 91-96.

[3] L. Benini and G. De Micheli, (2002), “Networks on chips: a new SoC paradigm”, IEEE Computer, 35(1), 70–78. https://doi.org/10.1109/2.976921.

[4] GE Fen and WU Ning, (2010) “Genetic algorithm based mapping and routing approach for network on chip architectures”, Chinese Journal of Electronics, 19(1), 91–96.

[5] A.W. Topol, et al., (2006), “Three-dimensional integrated circuits”, IBM J. Research and Development, 50(4/5), 491-506. https://doi.org/10.1147/rd.504.0491.

[6] B. S. Feero and P. P. Pande, (2009), “Networks-on-chip in a three dimensional environment: a performance evaluation”, IEEE Transactions on Computers, 58(1), 32–45. https://doi.org/10.1109/TC.2008.142.

[7] RaduMarculescu, UmitY.Ogras, Li-ShiuanPeh, et al., (2009), “Outstanding reasearch problems in NoC design: system, microarchitecture, and circuit perspectives”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 28(1), 3–21. https://doi.org/10.1109/TCAD.2008.2010691.

[8] A.Benymina and P.Boulet, (2007), Multi-objective mapping for NoC architectures, Journal of Digital Information Management (JDIM), 5(6), 378-384.

[9] T.Bjerregaard and S.Mahadevan, (2006), “Survey of research and practices of network-on-chip, ACM ComputSurv. 38(1), 1-51. https://doi.org/10.1145/1132952.1132953.

[10] L.Benini, D.Bertozzi, A.Guerri and M.Milano, (2006), “Allocation, scheduling and voltage scaling on energy aware MPSoCs”, CPAIOR, 44-58.

[11] G.Ascia, V.Catania and M.Palesi, (2006), “A Multi-objective Genetic Approach to Mapping Problem on Networkon-Chip”, Journal of Universal Computer Science, 2, 370-394.

[12] D.Shin, J.Kim, (2004), “Power-Aware Communication Optimisation for Networks-on-Chips with Voltage Scalable Links”, in Proc. CODES, 04, 170-175.

[13] Suleyman Tosun, Ozcan Ozturk, Erencan Ozkan, Meltem Ozen, (2015), “Application mapping algorithms for mesh-based network-on-chip architectures”, The Journal of Supercomputing, 71(3), 995-1017. https://doi.org/10.1007/s11227-014-1348-x.

[14] Yonghee Yun; Young Hwan Kim, (2016), “Design-time energy optimization for asymmetric multiprocessor system-on-chip”, International SoC Design Conference (ISOCC).