International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
NATIONAL CONFERENCE on Developments, Advances & Trends in Engineering Sciences
(NCDATES- 09th & 10th January 2015)

RESEARCH ARTICLE OPEN ACCESS

FPGA Based Design and Implementation of Genetic Algorithm Using Verilog

Vaseem Ahmed Qureshi[1], S. Hanmandlu[2], B. Papachary[3]
[1]Associate Professor, Department of ECE, CMR Engineering College, Hyderabad.
[2]Assistant Professor, Department of ECE, Hyderabad Institute of Technology & Management, Hyderabad.
[3]Associate Professor, Department of ECE, CMR Engineering College, Hyderabad.
E-mail: qureshi.vaseem@gmail.com Hnu.mtech@gmail.com biroju.chary@gmail.com

Abstract-
In this paper an efficient approach is presented to design and implementation of Genetic Algorithm for function optimization. Genetic algorithms (GA) are known as one of robust heuristic algorithms for optimization of problems in various fields of engineering. It generates solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. GA is composed of population of strings or chromosomes and three evolutionary operators are selection, crossover and mutation. The different modules of Genetic algorithm are designed and simulated with ModelSim 5.7d simulator and implemented on Spartan3 FPGA target device and synthesized to obtain gate level net list using Leonardo Spectrum. This paper shows the simulation result and RTL of genetic operators and hardware synthesis results. Results show enhanced performance in terms of speed and area utilization.
Keywords- Genetic Algorithm; FPGA; Simulation; Verilog

I. INTRODUCTION

The Genetic Algorithm (GA) was invented by Prof. John Holland at the University of Michigan in 1975 and it has been made widely popular by Prof. David Goldberg at the University of Illinois [1]. GA is a class of evolutionary algorithms that typically use fixed-length character strings to represent their genetic information, together with a population of individuals that undergo crossover and mutation in order to find interesting regions of the search space. GA work on the principle of "survival of the fittest", where the less fit members of a particular generation are replaced by new members formed by combining parts of highly fit members. The objective of the GA is to find an optimal solution to a problem. It does not depend on the specific areas of the problem belongs to, and has been widely used in function optimization, automatic control, image processing, machine learning and other technology areas [2].

II. THE GA CYCLE

Genetic Algorithms work by evolving a population of members over a number of generations. First, the initial population of members is generated. After this, the fitness of each member is evaluated, where the fitness is a function of the application. Two members or individuals are then selected simultaneously. The selection is usually proportional to the fitness value, i.e., the members with higher fitness have a greater probability of being selected. These members are then crossed to form new individuals and these new members are mutated to avoid convergence to local optima. The resulting members replace the less fit members of the old population. Thus, the fitness value of the population is improved with every new generation.

A Genetic Algorithm operates through a simple cycle of stages. Fig.1 shows the cycle of GA [11].

i. Creation of a population of strings.
ii. Evaluation of each string.
iii. Selection of best strings and
iv. Genetic manipulation to create new population of strings.

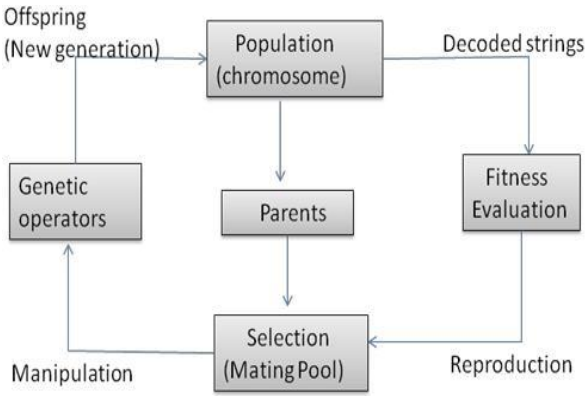

Fig.1 The Cycle of GA

1) Population:
All genetic algorithms work on a population or a

collection of several alternative solutions to the given problem. Each individual in the population is called a string or chromosome. These individuals are decoded as binary strings, and the individual characters or symbols in the strings are referred to as genes.

2) Fitness Evaluation:
The fitness function is an important component of the GA and has an important role in directing the search toward high-quality solutions. An individual's fitness reflects its quality as a solution to the problem, relative to the other individuals.

3) Selection:
Based on the fitness function the mating pool selects chromosomes which undergo crossover and mutation.

4) Crossover:
The Genetic operators are Crossover and Mutation. Crossover is the main operator used for reproduction. It combines portions of two parents to create two new individuals, called offspring, which inherit a combination of the features of the parents. The different process of crossover is:

i. One point Crossover:

Chromosome 1	**11010** | **1010**
Chromosome 2	**11010** | **1001**
Offspring 1	**11010** | **1001**
Offspring 2	**11010** | **1010**

Table 1

ii. Two point Crossover:

Chromosome 1	**110** | **101** | **010**
Chromosome 2	**100** |**100** | **001**
Offspring 1	**110** |**100** | **010**
Offspring 2	**100** |**101** | **001**

Table 2

iii. Uniform Crossover:

Chromosome 1	**11101001000**
Chromosome 2	**00001010101**
Offspring 1	**10101010001**
Offspring 2	**01001001100**

Table 3

5) Mutation:
Mutation is an incremental change made to each member of the population. Mutation enables new features to be introduced into a population. Mutation can be done by flipping a bit called point mutation. The GA operation invokes the mutation operation on the new bit strings very rarely, that is with a low probability, generating a random number for each bit and flipping this bit only if the random number is less than or equal to the mutation probability.

Original Offspring	**110110010**
Mutated Offspring	**110010010**

Table 4

After Crossover and Mutation new Offspring are produced and replaces the old generation. The new generation gives the better fitness function than the previous generation and optimizes the function.

III. BASIC STRUCTURE OF GA

Generate initial
population;
Evaluate
(Population);

Best individual = Best of
(Population); For i =1 to Num of
Generations Do

Select Parent1 (P1) and Parent2 (P2) from
population; Crossover (P1, P2, C1, C2);
Mutate Child1 (C1);

Mutate
Child2 (C2);
New
Population;

Evaluate (New
population); Population
= New Population;
Solution = Best
individual;

IV. APPLICATION OF GA IN FUNCTION OPTIMIZATION

Optimization is the process of adjusting the inputs to or characteristics of a device, mathematical process, or experiment to find the minimum or maximum output or result.

The genetic algorithm is used to optimize the function $f(x) = x^2$ where the range of x is taken from 0 to 30. Initial population of four strings is taken where each string is of 5-bit [5]. For the fitness function $f(x) = x^2$ where $0 \leq x \leq 30$, the maximum fitness value that can be attained is 900 [4].

To design fitness function $f(x) = x^2$, a bit multiplier is used. The benefit of using bit multiplier is that is that it is very fast and accurate. Fig.1 to 4 shows the simulation result and Fig.5 to 7 gives the RTL of

crossover, mutation and random number generator module.

V. SIMULATION RESULT AND RTL

A. Crossover Module

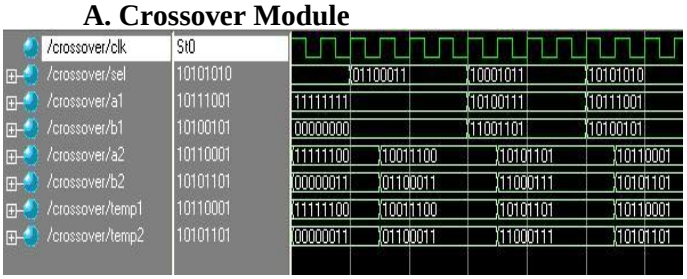

Fig.1 Model Sim Based Simulation of Crossover Module

B. Mutation Module

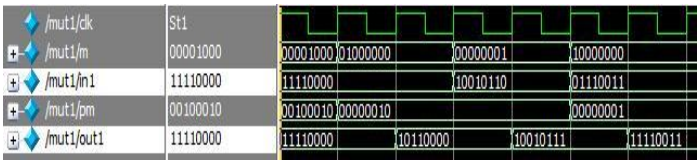

Fig.2 Model Sim Based Simulation of Mutation

Module C. Random number Generator

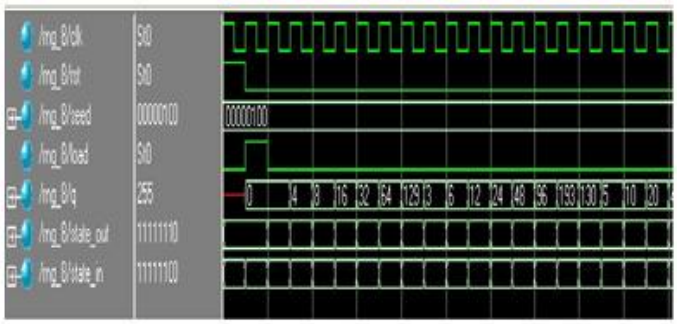

Fig.3 Model Sim Based Simulation of Random Number Generator module

D. Genetic Function Module

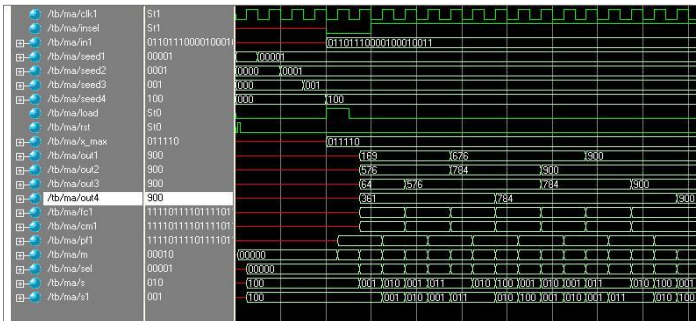

Fig.4 Model Sim Based Simulation of GA for f(x) = x^2

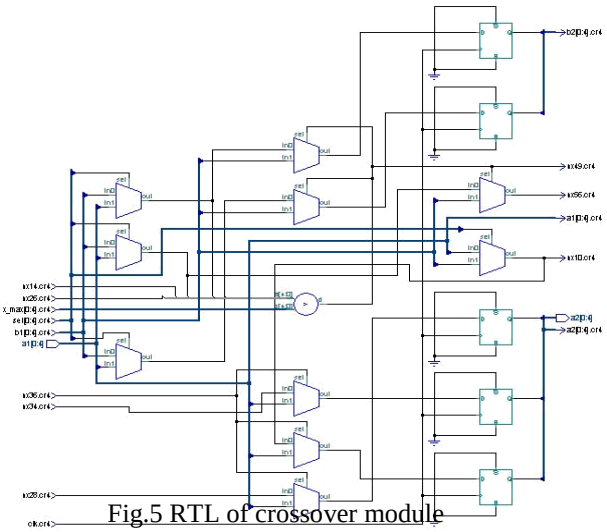

Fig.5 RTL of crossover module

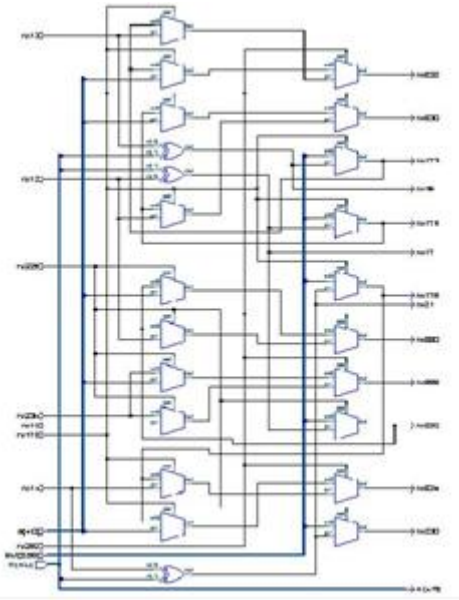

Fig.6 RTL of mutation module

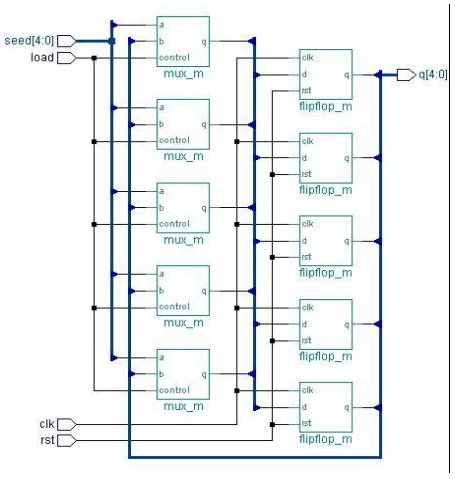

Fig.7 RTL of Random Number Generator module

Output verification of GA results with the help of Matlab. Fig.8 shows the function $f(x) = x^2 + y^2$ before applying GA. The function f(x) is maximum at x=20 and y=20. Fig.9 shows the function after applying GA and optimizes the function at generation 500.

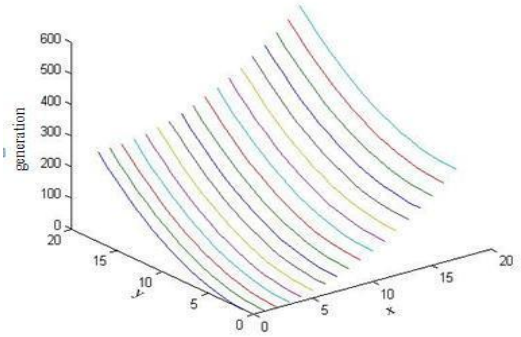

Fig8. $F(x) = x^2 + y^2$ without GA

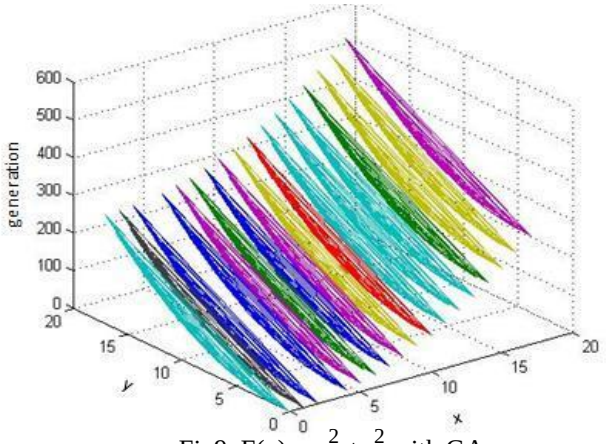

Fig9. $F(x) = x^2 + y^2$ with GA

VI. HARDWARE SYNTHESIS RESULTS

Table5 shows the area utilization of proposed design implemented on xc3s200pq208 target device in terms of IOs, Global buffers, Function Generators, CLB Slices, Flip flops.

Resource	Used	Avail	Utilization
IOs	108	141	76.60%
Global buffers	1	16	6.25%
Function Generators	601	3840	15.65%
CLB Slices	301	1920	15.68%
Dffs or Latches	105	4263	2.46%

Table5. Area Utilization

VII. CONCLUSION

The results in this paper illustrate to apply Genetic algorithm in function optimization. The different modules of GA are Random number generator, crossover, mutation and fitness module which are designed using MODELSIM Verilog and synthesized using Leonardo Spectrum and gives the RTL of the design. The simulation and post synthesize result are same. The result gives the satisfactory performance in terms of speed and area utilization.

ACKNOWLEDGMENT

This work is being carried out as M.Tech final year project under the supervision of Shri N.B Singh in the Central Electronics Engineering Research Institute, Pilani, India. Thanks to Director, CEERI to allow us to work at CEERI.

REFERENCES

[1] Pinaki Mazumder, Elizabeth M.Rudnick, "*Genetic Algorithms for VLSI Design, Layout and Test Automatio",* Addision Wesley,1999

[2] Guangya Liu and Jingli Chen, "The Application of Genetic Algorithm Based on Matlab in Function Optimization", International Conference on Electrical and Control Engineering (ICECE), 2011, Page(s): 5034 – 5037

[3] Pratibha Bajpai and Dr.Manoj Kumar," Genetic Algorithm – an Approach to Solve Global Optimization Problems", Indian Journal of Computer Science and Engineering, Vol 1, Issue 3, Oct-Nov 2010, Page(s): 199-206.

[4] N. Shruthi and P. Carla, "Hardware Implementation of the Genetic Algorithm Modules for Intelligent System", 48th Midwest Symposium on Circuits and Systems, 2005, Vol.2, Page(s):1733 - 1736.

[5] Scott, S.D.; Samal, A.; Seth, S.; " HGA: A Hardware Based Genetic Algorithm", Proceedings of the Third International ACM Symposium on Field-Programmable Gate Arrays, 1995. FPGA'95, 1995, Page(s): 53 - 59

[6] Yajuan Chen, Qinghai Wu," Design and Implementation of PID Controller Based on FPGA and Genetic Algorithm", 2011 International Conference on Electronics and Optoelectronics, Page(s):308 – 311.

[7] Jumrern Pimery, Pinit Kumhom," Development of a Flexible Hardware Core for Genetic Algorithm", IEEE International Conference on Intelligent Computing and Intelligent System, 2009, Page(s): 867 – 870.

[8] Rattapasakorn T.," Genetic algorithms on application specific integrated circuit", IEEE International Conference on Industrial Technology, 2002. IEEE ICIT '02. 2002, vol.2,Page(s): 1342 - 1344

[9] Arvind S.Babu, R.Chockalingam, S.Kavitha, "A Hybrid Genetic Algorithm Approach to a Departmental Class Timetabling Problem Using Efficient Data Structures", International Journal of Computer Applications,Vol 1, No:17,Page(s). 99-103.

[10] M. Srinivas, L. M. Patnaik," Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms", IEEE Transactions on Systems, Man and Cybernetics, Vol. 24, No. **4,** April 1994, Page(s):656-667.
http://www.learnartificialneuralnetworks.com /geneticalg.html