July 2014

VHDL IMPLEMENTATION OF GENETIC ALGORITHM FOR 2-BIT ADDER

VEDAVATHI. A

The Oxford College of Engineering , Visvesvaraya Technological University, Bangalore, India.,
starvedhachitty@gmail.com

MEENA. K.V

The Oxford College of Engineering , Visvesvaraya Technological University, Bangalore, India.,
kvmeena@gmail.com

GAYATRI MALHOTRA

CSG, ISRO satellite Centre Indian Space Research Organisation Bangalore, India, gayatri@isac.gov.in

Follow this and additional works at: <https://www.interscience.in/ijeee>

Part of the [Power and Energy Commons](#)

Recommended Citation

A, VEDAVATHI.; K.V, MEENA.; and MALHOTRA, GAYATRI (2014) "VHDL IMPLEMENTATION OF GENETIC ALGORITHM FOR 2-BIT ADDER," *International Journal of Electronics and Electrical Engineering*: Vol. 3 : Iss. 1 , Article 3.

DOI: 10.47893/IJEEE.2014.1120

Available at: <https://www.interscience.in/ijeee/vol3/iss1/3>

This Article is brought to you for free and open access by the Interscience Journals at Interscience Research Network. It has been accepted for inclusion in International Journal of Electronics and Electrical Engineering by an authorized editor of Interscience Research Network. For more information, please contact sritampatnaik@gmail.com.

VHDL IMPLEMENTATION OF GENETIC ALGORITHM FOR 2-BIT ADDER

VEDAVATHLA¹, MEENA. K.V² & GAYATRI MALHOTRA³

^{1&2}The Oxford College of Engineering, Visvesvaraya Technological University, Bangalore, India.

³CSG, ISRO satellite Centre Indian Space Research Organisation Bangalore, India

Email: starvedhachitty@gmail.com, kvmeena@gmail.com & gayatri@isac.gov.in

Abstract - Future planetary and deep space exploration demands that the space vehicles should have robust system architectures and be reconfigurable in unpredictable environment. The Evolutionary design of electronic circuits, or Evolvable hardware (EHW), is a discipline that allows the user to automatically obtain the desired circuit design. The circuit configuration is under control of Evolutionary algorithms. The most commonly used evolutionary algorithm is Genetic Algorithm. The paper discusses on Cartesian Genetic Programming for evolving gate level designs and proposes Evolvable unit for 2-bit adder based on Genetic Algorithm.

Keywords— Cartesian Genetic Programming, Evolvable hardware, Genetic Algorithm, 2-bit Adder, Reconfigurable FPGA, Virtual Reconfigurable Circuit.

I. INTRODUCTION

Digital reconfigurable circuits implemented using FPGA suits many different area of applications. Generally reconfiguration sequence is determined at design time. To adapt the system to a new environment needs totally new configuration of hardware that was not considered at design time. Evolvable Hardware is the field that is associated with dynamic adaption of hardware using bio-inspired techniques like Genetic Algorithm (GA). GA is a stochastic search method that operates on a population of potential solutions and applies the principle of survival of the fittest to produce better approximation to solution.

Evolvable Hardware (EHW) is useful broadly in two areas: (1) for automatic generation of new solutions and (2) implementation of autonomous adaptive devices. For case (1), the evolutionary algorithm (EA) will be used in design phase and this approach is applied in this paper. For case (2), the EA is responsible for continual adaptation of a device in changing environment or for automatic functional recovery [3] of a device after damage.

The application of GA in combination logic design is existing from long time. Different methods proposed are able to solve the functional output for combination logic and some methods emphasize on the optimization of gate usage too [1], [2]. Reconfigurable computing is performed to effectively utilize hardware resources and to maximize the system efficiency while combining it to evolvable hardware concept we can accomplish, what is called the evolvable computing.

The objective of this paper is to propose a technique to perform evolvable computing at the level of HDL.

As a case study 2-bit adder circuit for FPGA at the level of VHDL is designed. The structure of the digital circuit is encoded into one-dimensional genotype like in Cartesian Genetic Programming (CGP) and represented by a finite string of bits. The types of gates used are Wire, AND, OR, XOR, NOT and its combinations.

The paper is organized as follows. Section 2 is about FPGA for evolvable hardware and GA. Section 3 is about Virtual Reconfigurable Circuit and its implementation using CGP. Section 4 describes the proposed evolvable unit for 2-bit adder. Section 5 Experimental results. Section 6 is about discussion and conclusion.

II. FPGA FOR EVOLVABLE HARDWARE

Implementation of Evolvable system using FPGA is cost-effective and flexible. Currently Xilinx is the most popular platform for implementation of Evolvable systems. There are various approaches for Xilinx FPGA to use it for re-configuration [4], [5]. The FPGA based EHW can be done using one of two options:

- (1) The FPGA serves in the fitness calculation only. The EA executed on a personal computer, sends configuration bits to FPGA in order to obtain fitness values.
- (2) The entire system is built on FPGA.

In a case of EHW, the chromosomes are transmitted to the configuration bit streams and the configuration bit streams are uploaded into the FPGA. Each configuration bit in the chromosome defines some architecture feature of the reconfigurable hardware. A bit might give the state of switch that connected to circuit components and every candidate configuration

must be checked for fitness -which measures how closely it matches a target response[6],[1]. There are two different ways of checking fitness, but which is method used depends on whether the evolution is done extrinsically or intrinsically:

- (1) *Extrinsic evolution* in which a software circuit simulator is used to evaluate circuit configuration.
- (2) *Intrinsic evolution* in which circuits are sought using a hardware accelerator. Every chromosome is downloaded and physical testing measures fitness using hardware.

In most of commercial system such features as flexibility, scalability and implementation easiness play vital role selection of appropriate design. The flexibility defines whether the system accepts to evolve different kinds of problem. The scalability defines the capability of a designed system to able to scale from evolution scale from evolution of small tasks to larger problem without significant modification in the structure. And finally implementation easiness that define how easy to implement the desired solution. These three parameters will give a high priority in development of an EHW implementation [9], [4].

Fig. 1 Illustrated types of intrinsic EHW implementation.

Fig. 1 Types of implementation of intrinsic EHW.

The proposed FPGA designs featured not only with significant change in chip structure, but also in limitation of accessible information about chip performance. Hence FPGA designers suggest solution to implement the run time configuration intrinsic EHW system: FPGA-Based Run-Time Configuration System [4]

FPGA-Based Run-Time Configuration System

The proposed FPGA-based Run-Time Configuration System is divided into two parts: Evolutional strategy and Reconfigurable Hardware [4].

Fig. 2 FPGA-Based Runtime Configuration System Architecture

The reconfigurable hardware is used as the target to evolve. It executes the desired functionalities. The fitness function calculation is also partially implemented in this module [4]. Es block is composed of the implementation of evolutionary Strategy itself, the fitness value evolution, the chromosomes back-up and the communication protocol management in order to configure the evolved target (Reconfigurable hardware). This system will allow using the smallest number of gates possible and to have a very high speed during the reconfiguration as well as for performance of evolutionary process. The complexity of the communication between the ES and Reconfigurable Hardware is expressed by the response of the protocols in order to able to reconfigure the target from the ES in otherwise it is necessary to understand the bit stream.

GENETIC ALGORITHM

A **genetic algorithm (GA)** is a search heuristic that mimics the process of natural evolution. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which deals with three operators namely reproduction mutation and crossover. Fig.2. shows components of evolutionary Algorithm, Evolutionary algorithm (EA) is a computer algorithm that is based on principles of natural evolution and self adaption. The major components of evolutionary algorithm are representation variation, evolution based on fitness, selection, population, termination.

Fig3.Components of Evolutionary algorithm (EA)

The major components of evolutionary algorithm are representation, variation, evolution (based on fitness), selection, population, and termination [15]. Representation: it refers the data structure that encodes all the problem parameters needed to describe a solution. Variation: it is a random process that creates a new solution from existing solution by

changing some or all parameters. Population is a number of chromosomes that are available to the best. The fitness of chromosome is defined as the percentage of the correct output bits for every input combination of the complete specification.

In a genetic algorithm, a population of strings (called chromosomes or the genotype of the genome), which encode candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem, evolves toward better solutions. Solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. The fitness function is defined over the genetic representation and measures the *quality* of the represented solution. The fitness function is always problem dependent.

Initialization: Initially many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. **Selection:** During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a *fitness-based* process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. **Reproduction:** The next step is to generate a second generation population of solutions from those selected through genetic operators: crossover (also called recombination), and/or mutation. **Termination:** This generational process is repeated until a termination condition has been reached.

Fig4. Flow chart of working principle of GA.

III. VIRTUAL RECONFIGURABLE CIRCUIT AND ITS IMPLEMENTATION USING CGP

VRC is a reconfiguration layer developed on the top of FPGA in order to obtain fast reconfiguration and application-specific programmable elements. The use of VRC has allowed us to introduce a novel approach to the design of complete evolvable system in single FPGA [6], [12]. When the VRC is uploaded into the FPGA then the configuration bits stream has to cause that there will be created the following units in the FPGA: an array of programmable elements (PE), programmable inter connection network, configuration port. In most cases the VRC takes a regular Two-Dimensional array of programmable elements. A very efficient and successful approach – called *Cartesian Genetic Programming* (CGP) has been developed for the evolutionary design. The main advantage of CGP is that it uses the representation similar to real reconfigurable hardware.

For dynamic adaption EHW using bio-inspired techniques like Genetic Algorithm (GA). GA is a stochastic search method that operates on a population of potential solutions and applies the principle of survival of the fittest to produce better approximation to solution. HScone GA, Roulette GA, and Compact GA this are the different genetic algorithms are used for implementation.

Fig. 5 Genetic Processing Unit (GPU) Evolutionary Execution Phase

Genetic Programming

In CGP, a reconfigurable circuit is modeled as an array of (columns) x (rows) programmable nodes [12]. The number of circuit inputs and outputs are fixed. A node inputs can be connected to the outputs of some elements in the proceeding column or some of the circuit inputs. A node has up to inputs and a single output. Each and every node has functions (i.e. programmed to implement) defined in F set [2], [4]. The nodes in the same column are not allowed to be connected each other, and any node may connect or not connected.

The circuit output can be taken from any node output. Feedback is not allowed and thus the only

combinational circuits can be designed. Fig. 6 shows a model of a reconfigurable circuit and its corresponding configuration bit stream uploaded to establish the circuit connection.

The numbers 0, 1, 2 represents input and the numbers 12,10,6 represents output.

Fig. 6 Example of a circuit and its configuration in a Cartesian genetic programming with parameters : $F=\{FO,F1\}$, $n_c = 3$, $nr = 2$, $ni = 3$, $no = 2$, $nf=2$. Shows a model of a reconfiguration information is (input1,input 2,function F) 1,2,1, 2,3,0, 3,5,7, 0,0,4, 3,4,6, 0,6,1, 6,8.the last two integers indicates outputs of the circuit
The length of the chromosome [1] measured in the gene is

$$\Lambda = n_r \cdot n_c (n_n + 1) + n_o \quad (1)$$

Where n_r is number of rows, n_c is number of columns, n_o is number of outputs, n_n is a node of n_i inputs

3.1 Size of the design circuit

Consider that CGP, as defined in above section, is issued to evolve combinational circuits. Let a set H contain all logical functions of the form given by the mapping $\{0,1\}^{n_i} \rightarrow \{0,1\}^{n_o}$ all most all Boolean functions have the number of gates required for implementation of a particular circuit at least $2^{n_i n_i - 1}$ this is known as Shannon's effect [13],[1].

$$|H| = 2^{n_o 2^{n_i}} \quad (2)$$

Assume that the outputs fixed to the last column of the PEs and not modified by evolution. In this case 1 back is 1, and then the number of chromosomes that form different configurations of the reconfigurable circuit is given by

$$|C| = n_f^{nc,nr} (ni + nr)^{nn,nr(nc-1)} ni^{nn,nr} \quad (3)$$

Where $|C|$ is in fact the size of the search space [14]. Consider in above fig 6, where each PE can be connected either to some of primary inputs(5 options) i.e. 3 inputs and 2 functions(Fo,F1),hence 3 bits are needed to select the configuration of a single input of a PE. We will need 9 bits to configure a single PE. Let us analyze the length of the chromosome is $\Lambda=20$ or $(2 \times 3 \times 3 + 2 = 20)$ and size of the search space .the number of physically different circuits is up to $|C| =$.while the number of different logical behavior is $|H| =$.The choice of PE roughly corresponds to the Shannon's effect which indicates that about 6PE are needed to implement any circuits of 3inputs if no gates were shared. Implementation of VRC is based on

multiplexers. The configuration memory is connected to multiplexers that control the routing and selection of function in PE.

IV. THE PROPOSED EVOLVABLE UNIT FOR 2-BIT ADDER

Evolvable Hardware technique have the potential to significantly increase the functionality of deployed hardware system for space mission as it enables self-configurability [15].EHW uses simulated evolution to search for new hardware configurations. Evolutionary algorithm is the most commonly used algorithm. Evolutionary algorithm (EA) is a computer algorithm that is based on principles of natural evolution and self adaptation [6]. The most commonly used Evolutionary Algorithm is Genetic Algorithm (GA). GA have been applied to EHW because of it's perfectly with the configuration bits used in FPGAs.

Fig. 7 shows a proposed 2-bit model of a reconfigurable circuit and its corresponding configuration bit stream uploaded to establish the circuit connection.

The configuration of every PE is represented in the chromosomes as (input1, input2, and function) which define the connection of two inputs and the function realized in the PE. Every node i.e. PE is programmed to implement one of functions defined in the F set. a0, a1, b0, b1 are the input and the numbers 12,10,6 represents output. The numbers 12, 6 represents output sum and number 10 represents output carry. Each of the input numbers is represented in binary. Crossover and mutation operations are performed with thus 4 bit binary numbers.

The length of the chromosome measured in genes, $\Lambda = 51$. Let a set H contain all logical functions of the form given by the mapping $\{0,1\}^{n_i} \rightarrow \{0,1\}^{n_o}$ [13], were $|H| = 2^{n_o 2^{n_i}}$ The number of gates required for implementation of a particular circuit is at least $2^{n_i n_i - 1}$ is 4. 2-bit adder is implemented by using 13 gates shown in fig. 7. PEs (nodes) are connected either to some of the primary inputs or in the previous column. They are 16 options, i.e. 4-bits are needed to select a single input of a PE. Because PE has two inputs and other 3-bits are needed to select function i.e. 5 functional options, hence we will need 11 bits (4input1 bits +4 input2 bits+3 function bits) to configure a single PE.

Fig. 7 2 bit adder circuit and its configuration in (CSG) Cartesian genetic Programming with parameters

:F={F1,F2,F3,F4,f5}, =4,nr=4, =4 ,no=3, =5.the configuration informations:a0,b0,1, a1,b1,4,a0,b1,3, a1,b1,6, 1,8,2, 1,b1,1, a1,b1,2, a0,b0,8,3,3,5, 4,a1,2, 7,5,2, 3,11,3, 9,8,3,12,10,6.The last integers determined the connection of outputs. Gates 13, 14, 15 are not utilized

Table I Logic function expressed in terms of Functions

Logic Function	Functions (F)
XOR GATE	F1
AND GATE	F2
OR GATE	F3
NOT GATE	F4
WIRE	F5

4.1 Algorithm for Random number generation

The initial population is obtained using Pseudo random number generation technique

- Initially 44 bits binary data is given as input.
- Then MSB and LSB is XORed for each and every rising clock edge.
- Store the result obtained in some location and repeat step 2 for all possibilities.

4.2 Algorithm for fitness calculation

- Store the user outputs for different combinations off 2-bit adder inputs.
- For each input combinations, store the evolved outputs.
- Compare the reference outputs with user outputs for each of the combination of inputs.
- If equal, increment the counter, and assign fitness value.
- After comparison get next input combinations. Check all 15 different input combinations.
- If yes, then stop, else go to step 3.

4.3 Simple generational genetic algorithm procedure:

- Choose the initial population of individuals
- Evaluate the fitness of each individual in that population
- Repeat on this generation until termination (sufficient fitness achieved):
- Select the best-fit individuals from reproduction
- Breed new individuals through crossover and mutation operations to give birth to offspring

- Evaluate the individual fitness of new individuals
- Replace least-fit population with new individuals

V. EXPERIMENTAL RESULTS

Fig 8.Random number generation

Fig 9 Fitness calculation

VI. DISCUSSION AND CONCLUSION

This paper has contained a design of 2-bit adder, FPGA based EHW development and proposed Evolution unit set-up can be applied for future space application. Proposed 2-bit adder is designed by using 16 number of PEs. The configuration of every PE is represented in the chromosomes as input1, input2, and function. We will need 11 bits to configure a single PE. The length of the chromosome is $\Lambda=51$ and size of the search space is up to $|C| =$. An Evolvable Hardware technique has potentially and significantly increased the functionality of deployed hardware system for space mission as its enable Self-Reconfigurability. Future planetary and deep space exploration demands that the space vehicles should have robust system architecture and be dynamically reconfigurable to explore unpredictable environment. For dynamic adaption EHW using bio-inspired techniques like Genetic Algorithm (GA). GA is a stochastic search method that operates on a population of potential solutions and applies the principle of survival of the fittest to produce better approximation to solution. Hsclone GA, Roulette GA, and Compact GA this are the different genetic algorithms are used for implementation. Implementation of Evolvable system using FPGA is cost-effective and flexible. Currently Xilinx is the most popular platform for implementation of Evolvable systems. There are various approaches for Xilinx FPGA to use it for re-configuration. The use of proposed intrinsic run-time configurations will allows

one to overcome at the certain level the scalability problems and reduce the time required to re-design the system every time.

REFERENCES

- [1] K.H.Chong,I.B .Aris,M.A.Sinan "Digital Circuit Structure Design via Evolutionary Algorithm Method"Journal of Applied science 7(3):380-385,2007.ISSN 1812-5654.
- [2] Lucas Sekania,"*Extrinsic and Intrinsi Evolution of multifunctional combinational modules*" IEEE Congress on Evolutionary Computation 2006Adrian stoica,"*Towards Evolvable Hardware Chips: Experiments with a Programmable Transister Array*".
- [3] Adian stoica, Ricardo Zebulum,and Dider Keymeum "*Progress and Challenge in Building Evolvable Devices*"California Institute of Technology.
- [4] Cyrille Lambert, Tatiana Kalgovnova,and Emanuele stomea" *FPGA –Based System for Evolvable Hardware*"processing of world acadomy of science,Engineering and technology volume12 march 2006 ISSN 1307-688.
- [5] Zdenec Vasicaek and Lukas Sekania "An Evolvable Hardware System in Xilinx Virtex-II pro FPGA" Innovative Computation and Application, Vol. 1, No. 1 ,2007.
- [6] L Sekania , Stepan Friedl "An Evolvable Combinationl unit for FPGAs –Draft".Computing Informatics,Vol. 23,2004,461-486,V 2005-july-7.
- [7] Yang Zhange,Stephen L,Smith,Andy M.Tyrrell "Digital Circuit Design using Intrinsic Evolvable Hardware"proceeding of the 2004 NASA/DOD Conference on Evolution Hardware (EH'04)0-7695-2145-2/04 2004 IEEE
- [8] L.Sekanina,"*Towards Evolvable IP cores for FPGAs*"In:proc. Of The 2003 NASA/DoD Conference on Evolvable Hardware,Los Alamitos,US, ICSP, pp. 145-154, ISBN 0-7695-1977-6. 2003
- [9] Carlos A. Coello Coello,Alan D. Christian and Arturo Hernoandez Aguirre "*Automated design of Combinational Logic Circuits using Genetic Algorithms*".
- [10] L.Sekanina,s Feriedl "*On Routine Implementation of vvirtual Evolvable devices using COMBO6*" In:proc. Of The 2004 NASA/DoD Conference on Evolvable Hardware,Los Alamitos,US, ICSP, pp.63-70, ISBN 0-7695-2145-2. 2004.
- [11] JBits: *Java based interface for reconfigurable computing* Steve Guccione, Delon Levi and Prasanna Sundararajan Xilinx Inc., 2100 Logic Drive San Jose, CA 95124 (USA)
- [12] Miller, J-Thomson, P:"*Cartesian Genetic Programming*" "In proc of the 3rd European Conference on Genetic Programming, LNCS 1802, Springer verlag ,Berlin ,2000,pp. 121-132.
- [13] Gruska J "*foundation of computing combinational circuits using evolvable hardware*"Thomson publishing computer press 1997.
- [14] Sekania L-Rujica R:"*Design of the Specific Fast Reconfigurable Chip using Common FPGA*.In: proc. Of the 3rd IEEE Design and diagnostic of Electronic Circuits and System DDEC,00,Polygrafia SAF Bratislava,Slovakia 2000,pp. 161-168.
- [15] G.Malhotr,"Evolvable Hardware and its relevant for Future Space Mission",DOS-ISRO 10-12 Dec 2008.
- [16] D B Verneker. G Malhotra, V colaco "*Reconfigurable FPGA using Genetic Algorithm* "International Conference and Workshop on emerging Technology (ICWET 2010).
- [17] Xin Yao and Tetsuya Higuchi ,"*Promises and Challenge of Evolvable Hardware* "IEEE Transaction On System, Man and Cybernetic-part C:Application and reviews ,Vol.29 No.1Feb 1999.