Xuesong YAN[1], Qinghua WU[2], Hanmin LIU[3]

China University of Geosciences (1), WuHan Institute of Technology(2) , Wuhan Institute of Ship Building Technology(3)

Orthogonal Evolutionary Algorithm and its Application in Circuit Design

Abstract. *During the space electronic system in carries out the exploratory mission in the deep space, it maybe faced with kinds of violent natural environment, to electric circuit's performance, the volume, the weight and the stability proposed a higher request, the traditional circuit design method already more and more with difficulty satisfied this kind of request. In this paper, proposed a new orthogonal evolutionary algorithm which uses in the electronic system circuit optimization design and through the experiment proved, the algorithm obtains the circuit structure to surpass the tradition circuit design method. For the case studies, this means has proved to be efficient and the experiment results show that the new means have got the better results.*

Streszczenie. *W artykule zaproponowano nowy ortogonalny algorytm ewolucyjny pozwalający na optymalizację projektowania systemów elektronicznych. Eksperymenty dowiodły że alorytm jest znacznie skuteczniejszy od tradycyjnych metod projektowania. (**Ortogonalny algorytm ewolucyjny i jego zastosowanie do projektowania obwodów**).*

Keywords: Orthogonal Evolutionary Algorithm, Circuit Design, Adaptive local search
Słowa kluczowe: algorytm ewolucyjny, projektowanie obwodów..

Introduction

Evolutionary Electronics applies the concepts of genetic algorithms to the evolution of electronic circuits. The main idea behind this research field is that each possible electronic circuit can be represented as an individual or a chromosome of an evolutionary process, which performs standard genetic operations over the circuits. Due to the broad scope of the area, researchers have been focusing on different problems, such as placement, Field Programmable Gate Array (FPGA) mapping, optimization of combinational and sequential digital circuits, synthesis of digital circuits, synthesis of passive and active analog circuits, synthesis of operational amplifiers, and transistor size optimization. Of great relevance are the works focusing on “intrinsic” hardware evolution in which fitness evaluation is performed in silicon, allowing a higher degree of exploration of the physical properties of the medium. This particular area is frequently called Evolvable Hardware[1-3].

In the sequence of this work, Coello, Christiansen and Aguirre presented a computer program that automatically generates high-quality circuit designs[4]. Miller, Thompson and Fogarty applied evolutionary algorithms for the design of arithmetic circuits[5]. Kalganova, Miller and Lipnitskaya proposed another technique for designing multiple-valued circuits[6]. In order to solve complex systems, Torresen proposed the method of increased complexity evolution. The idea is to evolve a system gradually as a kind of divide-and-conquer method[7]. Based on the Miller’s method, Yan applied Gene expression programming (GEP) for the design of electronic circuits and the case study shows this technology was effective[8-9].

A major bottleneck in the evolutionary design of electronic circuits is the problem of scale. This refers to the very fast growth of the number of gates, used in the target circuit, as the number of inputs of the evolved logic function increases. This results in a huge search space that is difficult to explore even with evolutionary techniques. Another related obstacle is the time required to calculate the fitness value of a circuit. A possible method to solve this problem is to use building blocks either than simple gates. Nevertheless, this technique leads to another difficulty, which is how to define building blocks that are suitable for evolution.

Orthogonal evolutionary algorithm for cicuit design

Coding

In our algorithm, the chromosome representation we use Miller’s[10]. This representation is based on the FPGA of Xilinx Virtex-II. The starting point in this technique is to consider, for each potential design, a geometry (of a fixed size array) of uncommitted logic cells that exist between a set of desired inputs and outputs. The uncommitted logic cells are typical of the resource provided on the Xilinx FPGA part under consideration. An uncommitted logic cell refers to a two-input, single-output logic module with no fixed functionality. The functionality may then be chosen, at the implementation time, to be any two input variable logic function. In this technique, a chromosome is defined as a set of interconnections and gate level functionality for these cells from outputs back toward the inputs based upon a numbered rectangular grid of the cells themselves, as in Fig.1. The inputs that are made available are logic ‘0’, logic ‘1’, all primary inputs and primary inputs inverted. To illustrate this consider a 3 x 3 array of logic cells between two required primary inputs and two required outputs.

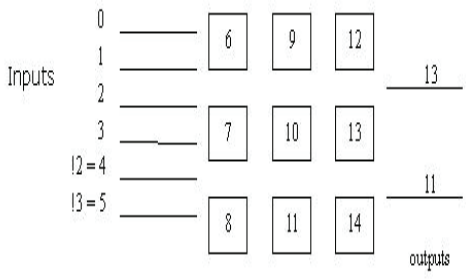

Fig.1. A 3 x 3 geometry of uncommitted logic cells with inputs, outputs and netlist numbering

The inputs 0 and 1 are standard within the chromosome, and represent the fixed values, logic ‘0’ and logic ‘1’ respectively. The inputs (two in this case) are numbered 2 and 3, with 2 being the most significant. The lines 4 and 5 represent the inverted inputs 2 and 3 respectively. The logic cells which form the array are numbered column-wise from 6 to 14. The outputs are numbered 13 and 11, meaning that the most significant output is connected to the output of cell 13 and the least significant output is connected to the output of cell 11. These integer values, whilst denoting the physical location of each input, cell or output within the

structure, now also represent connections or routes between the various points. In other words, this numbering system may be used to define a netlist for any combinational circuit. Thus, a chromosome is merely a list of these integer values, where the position on the list tells us the cell or output which is being referred to, while the value tells us the connection (of cell or input) to which that point is connected, and the cells functionality.

Each of the logic cells is capable of assuming the functionality of any two-input logic gate, or, alternatively a 2-1 multiplexer (MUX) with single control input. In the geo osome is shown belo 02-1 13-5 243 08-10 78-4 6119 64-9 2117 13 11

02-1 13-5 243 08-10 78-4 6119 64-9 2117 13 11

Fig.2 A typical netlist chromosome for the 3 x 3 geometry of Fig.1.

Notice, in this arrangement that the chromosome is split up into groups of three integers. The first two values relate to the connections of the two inputs to the gate or MUX. The third value may either be positive - in which case it is taken to represent the control input of a MUX - or negative - in which case it is taken to represent a two-input gate, where the modulus of the number indicates the function according to Table 4 below. The first input to the cell is called A and the second input called B for convenience. For the logic operations, the C language symbols are used: (i) & for AND, (ii) | for OR, (iii) ^ for exclusive-OR, and (iv) ! for NOT. There are only 12 entries on this table out of a possible 16 as 4 of the combinations: (i) all zeroes, (ii) all ones, (iii) input A passed straight through, and (iv) input B passed straight through are considered trivial - because these are already included among the possible input combinations, and they do not affect subsequent network connections in cascade.

Table.1. Cell gate functionality according to negative gene value in chromosome.

Gene Value	Gate Function
-1	A & B
-2	A & !B
-3	!A & B
-4	A ^ B
-5	A | B
-6	!A & !B
-7	!A ^ B
-8	!A
-9	A | !B
-10	!B
-11	!A | B
-12	!A | !B

This means that the first cell with output number 6 and characterized by the triple {0, 2, -1} has it's a input connected to '0', its B input connected to input 2, and since the third value is -1, the cell is an AND gate (thus in this case always produces a logical output of 0). Picking out the cell who's output is labeled 9, which is characterized by the triple {2, 6, 7}, it can be seen that its A input is connected to input 2 and its B input is connected to the output of cell 6, while since the third number is positive the cell is a MUX with control input connected to the output of cell 7.

Initialize population:

The traditional method of genetic algorithm is randomly initialized population, that is, generate a series of random numbers in the solution space of the question. In this paper, the orthogonal genetic algorithm using the orthogonal initialization [11]. For the general condition, before seeking out the optimal solution the location of the global optimal solution is impossible to know. for some high-dimensional and multi-mode functions to optimize, the function itself has a lot of poles, and the global optimum location of the function is unknown. If the initial population of chromosomes can be evenly distributed in the feasible solution space, the algorithm can evenly search in the solution space for the global optimum. Orthogonal initialization is to use the orthogonal table has the dispersion and uniformity comparable, the individual will be initialized uniformly dispersed into the search space, so the orthogonal design method can be used to generate uniformly distributed initial population.

Adaptive local search oprator:

Local search operator has a strong local search ability, and then can solve the shortcomings of genetic algorithm has the weak ability for the local search. And the population according to the current state of adaptive evolution of the local search space adaptive local search operator will undoubtedly greatly enhance the ability of local search. In the initial stage of the evolution, the current optimal solution from the global optimum region is still relatively far away, this time the adaptive local search operator to require search a large neighborhood space to find more optimal solution, it can maintain the population diversity. When the population has evolved to the region containing the global optimum, the adaptive local search operator to require a relatively small area to search in order to improve the accuracy of the global optimal solution.

In this paper, the adaptive local search operator is the adaptive orthogonal local search operator. Adaptive orthogonal local search operator is aimed at the neighborhood of a point to search, so the key point is to identify a point as the center of the hypercube, the hypercube in the orthogonal test, expect to be better Solution.

Framework of algorithm:

Step 1: Orthogonal initialize population;

Step 2: Using selection method select the excellent individual;

Step 3: Using select opertor and strategy select the two parents individuals;

Step 4: Using the orthogonal crossover operation

Step 5: Executive the adaptive local search strategy;

Step 6: Executive the adaptive orthogonal mutation strategy for the repeat individuals in the population;

Step 7: Repeat Step 3 to Step 7 until end of the algorithm.

Experimental result

Case 1

Evolving the one-bit adder was easier to do on a larger geometry but resulted in a less efficient circuit. That is many genetic algorithm was able to discover 100% functional solutions was intimately related to the size of the geometry, but our algorithm use small geometry to find the fully functional solutions.

The Fig. 3 is the one-bit adder without using optimum algorithm, and our resulting circuits as shown in Fig. 4. From the figure we know it is a gratifying result to obtain as it is clear that this design is an optimum solution.

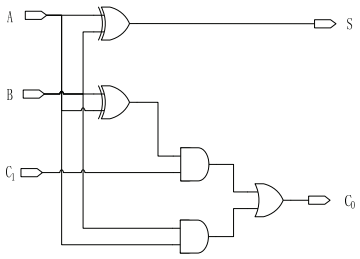

Fig.3. One-bit full adder without optimum

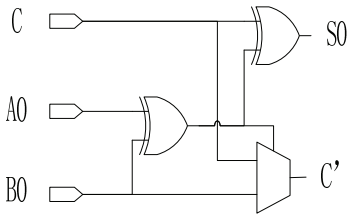

Fig.4. The evolved optimum one-bit full adder circuit

Case 2

A two-bit full adder, with a truth table with 5 inputs and 3 outputs. In this case, Our algorithm use small geometry to find the fully functional solutions, the matrix has a size of 3×3. The original circuit is showed in Fig. 5 and our resulting circuits as shown in Fig.6. From the figures we know it is a gratifying result to obtain as it is clear that this design is an optimum solution.

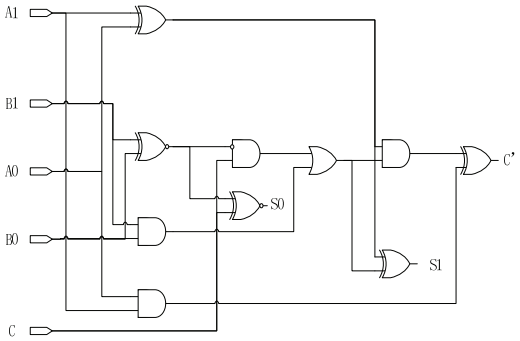

Fig. 5. Two-bit full adder without optimum

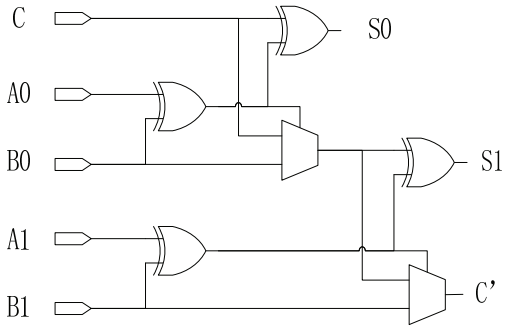

Fig. 6. The evolved optimum two-bit full adder circuit

Case 3

A three-bit full adder, with a truth table with 7 inputs and 4 outputs. In this case, Our algorithm use small geometry to find the fully functional solutions, the matrix has a size of 4×4. The original circuit is showed in Fig. 7 and our resulting circuits as shown in Fig.8. From the figures we know it is a gratifying result to obtain as it is clear that this design is an optimum solution.

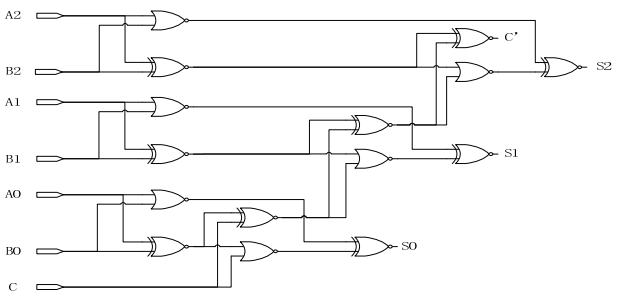

Fig. 7. Two-bit full adder without optimum

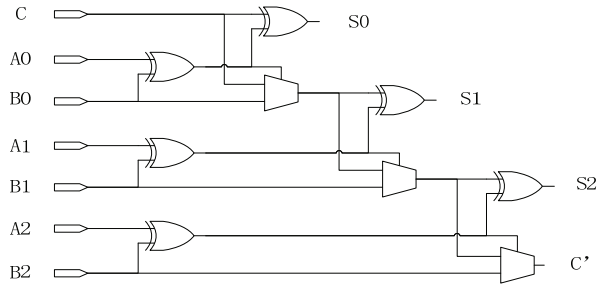

Fig. 8. The evolved optimum three-bit full adder circuit

Case 4

A four-bit full adder, with a truth table with 9 inputs and 5 outputs. In this case, Our algorithm use small geometry to find the fully functional solutions, the matrix has a size of 4×4. The original circuit is showed in Fig. 9 and our resulting circuits as shown in Fig.10. From the figures we know it is a gratifying result to obtain as it is clear that this design is an optimum solution.

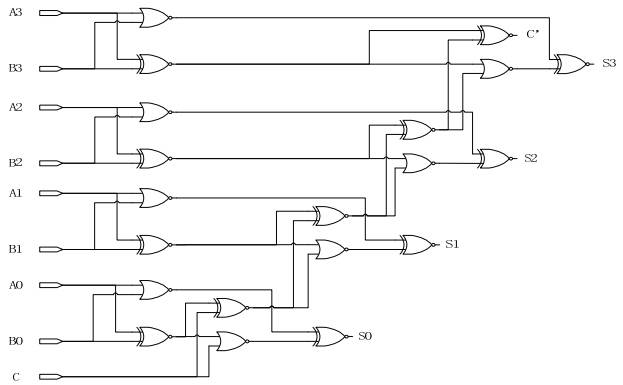

Fig. 9. Two-bit full adder without optimum

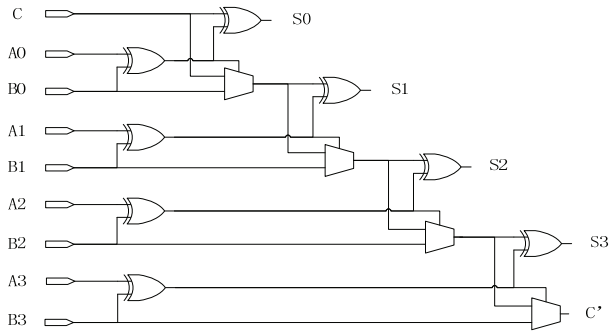

Fig. 10. Two-bit full adder without optimum

Conclusions

This paper proposed a new means for designing electronic circuits given a set of logic gates. The final circuit is optimized in terms of complexity (with the minimum number of gates). For the case studies this means has proved to be efficient, experiments show that we have better results.

There are still many avenues for further work. Other ways of representing rectangular arrays of logic cells may be devised and also, the relationship between cell connectivity and the evolvability of designs has still to be explored. There are many wider issues to be considered also which relate to the problem of evolving much larger circuits. It is a feature of the current technique that one has to specify the functionality of the target circuit using a complete truth table, however this is impractical for circuits with large numbers of inputs.

This paper is supported by the Provincial Natural Science Foundation of Hubei (No. 2011CDB334).

REFERENCES

[1] Zebulum, R. S., Pacheco, M. A. and Vellasco, M. M., Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms, CRC Press, 2001.

[2] Thompson, A. and Layzell, P, Analysis of unconventional evolved electronics, Communications of the ACM(1997), Vol. 42, 71-79.

[3] Louis, S.J. and Rawlins, G. J., Designer Genetic Algorithms: Genetic Algorithms in Structure Design, in Proceedings of the Fourth International Conference on Genetic Algorithms, 1991.

[4] Coello, C. A., Christiansen, A. D. and Aguirre, A. H., Using Genetic Algorithms to Design Combinational Logic Circuits, Intelligent Engineering through Artificial Neural Networks(1996). Vol. 6, 391-396.

[5] Miller, J. F., Thompson, P. and Fogarty, T, Algorithms and Evolution Strategies in Engineering and Computer Science: Recent Advancements and Industrial Applications(1997). Chapter 6, Wiley.

[6] Kalganova, T., Miller, J. F. and Lipnitskaya, N., Multiple_Valued Combinational Circuits Synthesised using Evolvable Hardware, in Proceedings of the 7th Workshop on Post-Binary Ultra Large Scale Integration Systems(1998).

[7] Torresen, J., A Divide-and-Conquer Approach to Evolvable Hardware, in Proceedings of the SecondInternational Conference on Evolvable Hardware(1998). Vol. 1478, 57-65.

[8] X.S.Yan,Wei Wei et.al; Design Electronic Circuits by Means of Gene Expression Programming , Proceedings of the First NASA/ESA Conference on Adaptive Hardware and Systems, IEEE Press(2006), 194-199.

[9] X.S.Yan et.al; Designing Electronic Circuits by Means of Gene Expression Programming Ⅱ , Proceedings of the 7th International Conference on Evolvable Systems: From Biology to Hardware, Lecture Notes in Computer Science, Springer Press(2007), 319-330.

[10] J. F. MILLER, Designing Electronic Circuits Using Evolutionary Algorithms, Dept. of Computer Studies, Napier University(2003).

[11]Leung Yiu-Wing,Wang Yuping. An orthogonal genetic algorithm with quantization for global numerical optimization. IEEE Transactions on Evolutionary Computation(2001), 5(1), 41-53.

***Authors**: Xuesong Yan, associate prof., School of Computer Science, China University of Geosciences, Wuhan 430074, China E-mail: yanxs1999@126.com; Qinghua Wu, Faculty of Computer Science and Engineering, WuHan Institute of Technology, Wuhan 430074, China, E-mail: wuqinghua@sina.com.cn; Hanmin Liu, Faculty of Computer Science and Engineering, Wuhan Institute of Ship Building Technology, Wuhan 430050, China, E-mail: wuqinghua@sina.com.cn.*