Indian Journal of Science and Technology

Digital Circuit Design using Chaotic Particle Swarm Optimization Assisted by Genetic Algorithm

Hamid Reza Nikoui* **and Mohadeseh Semsari**

Department of Electronic Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran; nikoui90@gmail.com

Abstract

Optimal digital circuit design could be very beneficial for reducing the expenses of electronic devices. But since it is a complicated problem in general, utilization of sophisticated algorithm for this task seems necessary. In this paper a new method based on combining particle swarm optimization, genetic algorithm and chaotic search is proposed for digital circuit design. The inclusion of wiring minimization in objective function is also proposed and investigated.

Keywords: Digital Circuits, Particle Swarm Optimization, Genetic Algorithm, Chaotic Local Search, Wiring Minimization.

1. Introduction

Digital electronics is one of the main technologies which serve as foundations of our modern world. Digital computers may be the most important devices in today's life, and they are almost completely based on digital logic. It is remarkable that complicated operations done by high speed computers are almost based on some simple binary logic operations of simple gates and transistors [1]. Other types of programmable logic devices like FPGAs, DSPs and etc, also serve as building blocks of modern technologies and industries [2]. All these technologies require optimal design of electronic circuits on chips in a way which pose lower power consumption, smaller area occupation, and lower costs of materials and manufacturing. To obtain such objectives, reducing the number of gates and transistors to represent some determined logical functionality is a main task to be manipulated by designers. Reduction in wirings in the circuits is also of importance because it can affect power and area consumption of the device.

Various kinds of methods have been proposed and developed for design and design optimization of logic circuits. The most basic and famous method for designing circuits of logic gates to represent some function between inputs and outputs is known as Karnaugh map [3]. For smaller circuits one can perform trial and error, or some basic methods such as Quine-McClusky method [4] to obtain optimal design of desired circuit from Karnaugh maps. But for designing circuits with large number of gates and switches more advanced and rather automatic methods are needed.

Computer science provides some general methods for designing complicated structures based on concept of evolution of solutions to a problem. Evolutionary methods such as Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) are used in almost every application and every discipline of engineering in recent years [5, 6]. Basic notions and aspects of evolutionary digital circuit design are presented in [7, 8]. In [9], utilization of Imperialist Competitive Algorithm (ICA) for digital circuit evolutionary design is proposed.

GA which is inspired from evolutions of living beings in successive generations is the most utilized method in the field of evolutionary electronics design [7]. Power consumption optimization of digital circuits using GA has been studied in [10], in which four main strategies for reducing power including V_{dd} assignment, V_{th} assignment, sizing and stack forcing are considered and optimized by means of GA.

In using GA, one represents possible design specifications of a structure (in our case a digital circuit) in form of

* *Corresponding author:*
Hamid Reza Nikoui (nikoui90@gmail.com)

one dimensional arrays of numbers named *chromosomes*. Then by evolving these chromosomes, by maintaining best fitting and generating new better fitting ones, optimal solutions of the problem could be achieved. There are also different variations of genetic algorithm [11, 12].

PSO is another type of iterative optimization methods which is based on evolution of solutions as represented by positions of particles in an N dimensional solutions space [*]. In using PSO, one needs easier operations done on particles than the operations done on chromosomes in GA. This makes PSO rather faster in computational procedure. On the other hand, PSO manipulates continuous optimization problems in a straight forward way which is not so in the case of using GA. In recent years, PSO and its different variations have been used for many different types of applications [13]. To overcome some difficulties about sticking of particles in local extremum traps in search space, modified variations of PSO algorithm are proposed. One of the strategies for avoiding local extremums is to combine Chaotic Local Search (CLS) algorithm with PSO [14]. Chaotic local search also provides better search of solutions space in detail.

In this paper a new hybrid algorithm which combines advantages of GA, PSO and CLS is proposed for designing digital logic circuits. The proposed method can be used for both small and large scale circuits. It could be also utilized for any levels of design from transistor level to modular level design. In contrast to previous related researches to this work, optimization of wiring is also included in proposed method.

In section 2 of this paper the problem of digital circuit design addressed and explained for being solved by evolutionary design algorithms. In section 3 the procedure of the proposed method of this paper combining GA, PSO and CLS is presented. Utilization of proposed method in section 3 on the problem stated in section 2, and the results of circuit design are discussed in section 4.

2. Digital Circuit Evolutionary Design

The problem of designing digital circuits by means of evolutionary methods is stated in this section. However the proposed method of this paper could be used in many kinds of levels in design, the procedure is addressed here only for gate level design without lack of generality.

Consider some logical function from inputs to outputs of a device is to be designed by means of available gates and suitable wiring. General framework of such problem is depicted in Figure 1. The function between inputs and outputs of the circuit could be generally represented by a truth table.

It is assumed that the circuit is to be implemented on a rectangular chip by putting appropriate gates on cellular sections on it and wiring them together in the right way. The typical example of such cellular designation is shown in Figure 2. For every cell in the layout, some information is needed to provide the design specification. This information includes the type of gate in the cell (or generally the type of building block such as transistors or modules), and indices of cells which provide input signals to the gate in the cell. By gate type information, the area occupation of the gate and number of transistors for that gate are also specified. Also, this could be generalized to include other information such as power specifications, feeding voltages, delays and etc. Input signals to a gate are provided by means of wires in the circuit. In [9] wires are considered as gates with zero area consumption. But in this paper it is assumed that wires are set in the shortest rectangular path from connected cells. In this way, the total length of wiring in the circuit L_{wiring} is an objective which is to be minimized to provide lower power dissipation and smaller costs. This objective is provided by optimal placement of gates in the cellular layout.

Principal objective of design should be assumed as exact operation of the circuit with desired functionality as represented by the truth table. It means by setting any input array to the circuit the desired output of that input array should be achieved in the output terminals of circuit. This objective could be provided by defining a term f_{logic} which is the total number of different outputs of a circuit in response to all possible input arrays, to the corresponding desired outputs. So this term should be minimized to (to zero) provide the desired logical function of the circuit.

Two other objectives are obtaining minimum number of gates (N_{gates}) and minimum number of transistors ($N_{transistors}$) used in the circuit.

By these considerations the complete objective function for the problem of this paper could be written as

$$f = f_{logic} + N_{gates} + N_{transistors} + L_{wiring} \quad (1)$$

which should be minimized according to definitions of its terms. Weights of different terms in the overall objective

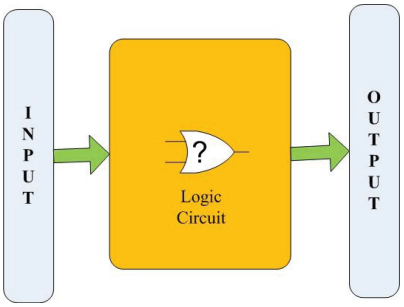

Figure 1. General framework of logic circuit design problem.

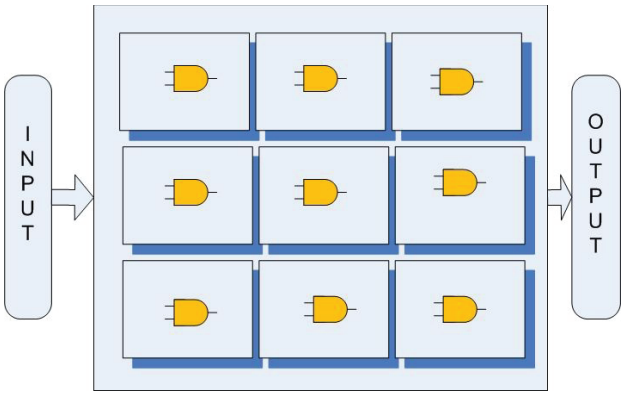

Figure 2. Typical cellular designation.

function could be adjusted by user. The only important consideration about these weights is setting the weight for logical objective term so high that fulfilling the logical functionality of the circuit be assured.

Definition of solution representation is another important issue in evolutionary design of logical circuits. In the problem in hand, design specifications are included in cells information. If information of each cell is represented in an array of numbers, then by catenation of these arrays into one larger 1-dimensional array of numbers the solution representation is achieved. Then we call these representations as *chroparticles*, as both operations of GA and PSO are considered to be performed on these objects. The structure of a typical *chroparticles* is shown in Figure 3.

3. Chaotic Genetic Assisted Particle Swarm Optimization (CGAPSO)

In this section the proposed method for design optimization of digital circuits are introduced. The proposed method is a

x1,1	y1,1	x2,1	y2,1	g1	x1,2	y1,2	x2,2	y2,2	g2	x1,3	y1,3	x2,3	y2,3	g3	x1,4	y1,4	x2,4	y2,4	g4
Cell 1 information					Cell 2 information					Cell 3 information					Cell 4 information				

Figure 3. A typical chroparticles.

hybrid algorithm which combines advantages of GA, PSO and CLS together. Due to cellular layout of circuit, utilization of crossover operation in GA is advantageous for this problem. On the other hand, fast computational procedure of PSO has its own benefits. CLS is also included in the method to avoid local minimums and providing search in details.

Procedures of GA and PSO are presented in literature very frequently [5, 6]. So here, only the CLS procedure presented. Then the overall proposed method of CGAPSO is introduced.

3.1 Chaotic Local Search

Chaotic systems are a type of nonlinear systems with even deterministic behavior but very sensitive to changes in initial conditions. In continuous time systems, chaotic behavior could only happen in systems with at least three state variables (e.g. Lorenz system), but in discrete time systems one state may be sufficient (as in the case of logistic map).

By utilization of dynamics of a chaotic system, one can perform a search in the space of solutions for a problem [14]. In this study, logistic map as described by

$$x_k + 1 = \mu x_k (1 - x_k) \tag{2}$$

is used as the chaotic dynamics for searching the solutions space. Logistic map dynamics is chaotic when $\mu = 4$ and $x_0 \notin \{0, 0.25, 0.5, 0.75, 1\}$. In Figure 4 the behavior of this system is depicted for $x_0 = 0.1$.

The procedure for chaotic search is as follows:

1. Set the time step variable as $k=0$. With initial guess for optimization variables x_i^k, map them in the interval $(x_{min,i}, x_{max,i})$ and obtain chaotic variables cx_i^k as below

$$cx_i^k = \frac{x_i^k - x_{\min,i}}{x_{\max,i} - x_{\min,i}} \tag{3}$$

2. Determine next values for chaotic variables cx_i^{k+1} by means of logistic map dynamics:

$$cx_i^{k+1} = 4cx_i^k \left(1 - cx_i^k\right) \tag{4}$$

3. Calculate new values for optimization variables by means of inverse transformation:

$$x_i^{k+1} = x_{min,i} + cx_i^{k+1}\left(x_{max,i} - x_{min,i}\right) \quad (5)$$

4. Evaluate new optimization variables (new solution) according to objective function.
5. If new solution is better than previous one, update the solution. Iterate the procedure again until some stopping criterion (e.g. maximum number of iterations) is met.

3.2 Procedure of CGAPSO

In the method of CGAPSO, similar to conventional PSO and GA methods an initial population of solutions as individuals (here named as *chroparticles*) is generated randomly. Then by performing some iterative operations on the population, the individuals are evolved. The procedure of proposed CGAPSO method is described step by step as below.

CGAPSO algorithm:

1. Generate initial population of individuals (chroparticles) randomly.
2. Set velocity vectors for each chroparticle randomly.
3. Evaluate objective function for chroparticles in the population and sort them according their objective value.
4. Set *gbest* as best chroparticle in the whole population. Set $pbest_i$ as best position experienced by *i-th* chroparticle.
5. Update velocities of chroparticles by PSO equation below

$$v_i(t+1) = Iv_i(t) + r_1C_1(gbest - x_i(t)) + r_2C_2(pbest_i - x_i(t)), \quad (6)$$

where I is inertia factor, r_1 and r_2 are random numbers, and C_1 and C_2 are constant coefficients.

6. Update chroparticles by equation below in which the δ determines the time step.

$$x_i(t+1) = x_i(t) + \delta v_i(t+1), \quad (7)$$

7. Perform crossover operation on a portion of chroparticles.
8. Perform mutation operation on a portion of chroparticles.

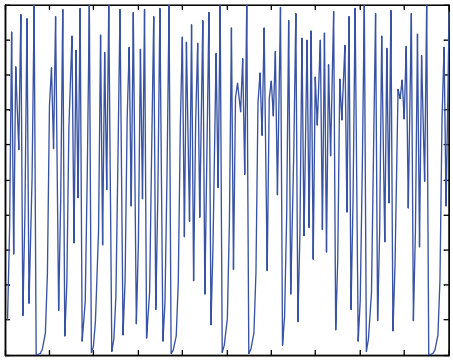

Figure 4. Behavior of logistic map for $\mu = 4$ and $x_0 = 0.1$.

9. Perform CLS (described in section 3–1) on a portion of chroparticles.
10. If stopping criterion is met finish the algorithm, otherwise repeat the procedure from step 3 again.

The flowchart of CGAPSO is shown in Figure 5.

4. Results of Digital Circuit Design by CGAPSO

In this section results of utilization of CGAPSO method for designing some logical circuits are presented. In each case the truth table of logical function is presented at first. Then the circuit as designed by CGAPSO is depicted and its specifications are presented in a table. The parameters of CGAPSO are set as shown in Table 1.

Case 1:

Truth table of this case is as shown in Table 2. The algorithm of CGAPSO solved this problem and the curve of best value of objective function in successive iterations is shown in Figure 6. The resulted circuit is also depicted in Figure 7.

Case 2:

In second case the number of inputs is 4 and the truth table is presented in Table 3. Here in two different simulations, the effect of wiring term in objective function is investigated. In first step, the term for length of wiring in the circuit is not considered and the resulted circuit became as shown in Figure 8.

In next step, the wiring term has been revived. The convergence curve of the algorithm became as shown in Figure 9 and the optimized circuit is depicted in Figure 10. It is clear that inclusion of wiring term had considerable positive effect on the final circuit.

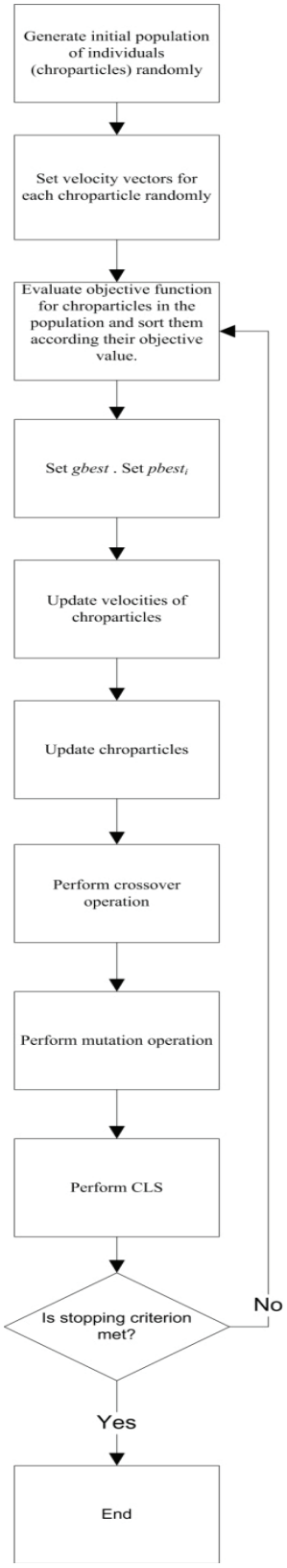

Figure 5. Flowchart of CGAPSO.

Table 1. CGAPSO main parameters

Parameter	Value
C_1	2.5
C_2	1.5
Crossover percent	40
Mutation percent	20
CLS percent	40
I	0.2
δ	1

Table 2. Truth table for case 1

Input			Output		
I_1	I_2	I_3	O_1	O_2	O_3
0	0	0	1	1	1
1	0	0	1	0	1
0	1	0	1	0	1
1	1	0	1	1	0
0	0	1	1	1	1
1	0	1	1	0	1
0	1	1	0	0	1
1	1	1	0	1	0

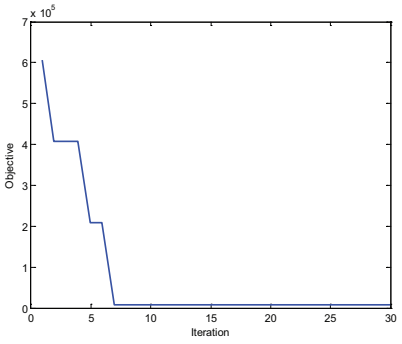

Figure 6. Curve of best value of objective function in successive iterations for case 1.

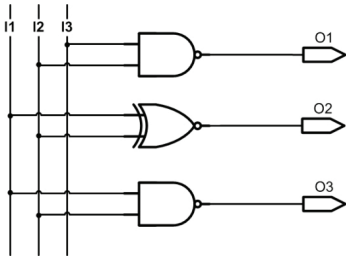

Figure 7. The resulted circuit for case 1.

Table 3. Truth table of case 2

Input				Output			
I_1	I_2	I_3	I_4	O_1	O_2	O_3	O_4
0	0	0	0	1	1	0	0
1	0	0	0	1	1	0	0
0	1	0	0	1	1	0	1
1	1	0	0	1	1	0	1
0	0	1	0	0	1	0	0
1	0	1	0	0	1	0	0
0	1	1	0	0	1	0	1
1	1	1	0	0	1	0	1
0	0	0	1	1	1	0	0
1	0	0	1	1	1	0	0
0	1	0	1	1	1	0	0
1	1	0	1	1	1	0	0
0	0	1	1	0	1	0	0
1	0	1	1	0	1	0	0
0	1	1	1	0	1	0	0
1	1	1	1	0	1	0	0

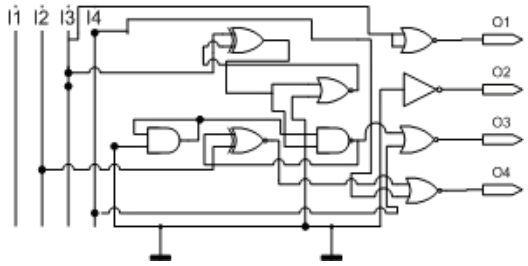

Figure 8. Resulted circuit of case 2 without wiring term.

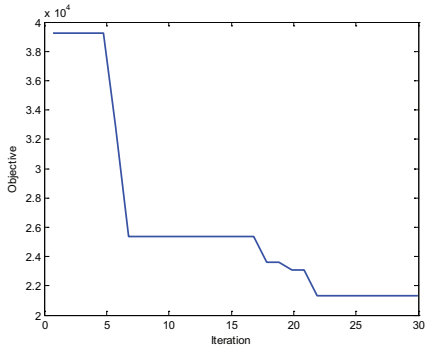

Figure 9. Curve of best value of objective function in successive iterations for case 2.

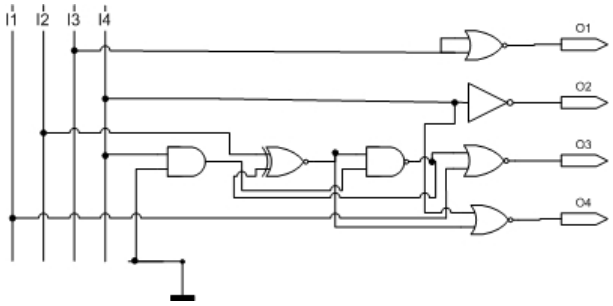

Figure 10. Resulted circuit of case 2 with wiring term.

5. Conclusion

In this paper a new method named CGAPSO is proposed for designing digital circuits. The proposed method is a combination of particle swarm optimization, genetic algorithm and chaotic local search methods. In addressing the design problem, it is proposed in this paper that wiring minimization terms should be included in the objective function to obtain better results. The effectiveness of proposed method and objective is shown by results of simulation for 2 cases of digital circuit design. Especially by exclusion and then reviving the wiring term in objective function its necessity is investigated.

6. References

1. Brown S D, and Vranesic Z G (2000). Fundamentals of digital logic with VHDL design, 2nd Edn., vol 6, McGraw-Hill New York.
2. Brown S, and Rose J (1996). FPGA and CPLD architectures: A tutorial, Design & Test of Computers, IEEE, vol 13, No. 2, 42–57.
3. Karnaugh M (1953). The map method for synthesis of combinational logic circuits, Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics, vol 72, No. 5, 593–599.
4. McCluskey E J (1956). Detection of group invariance or total symmetry of a Boolean function, Bell System Technical Journal, vol 35(6), 1445–1453.
5. Goldberg D E (1989). Genetic algorithms in search, optimization, and machine learning, Reading MA: Addison-Wesley.
6. Kennedy J, and Eberhart R (1995). Particle swarm optimization, IEEE International Conference on Neural Networks, 1995 Proceedings, vol 4, 1942–1948.

7. Miller J F, Job D Et al. (2000). Principles in the evolutionary design of digital circuits—Part I, Genetic programming and evolvable machines, vol 1, No. 1–2, 7–35.
8. Miller J F, Job D et al. (2000). Principles in the evolutionary design of digital circuits—Part II, Genetic programming and evolvable machines, vol 1, No. 3, 259–288.
9. Anjomshoa M, Mahani A et al. (2011). Evolutionary design and optimization of digital circuits using imperialist competitive algorithm, International Journal of Computer Applications, vol 32, No. 1, 14–19.
10. Hung W, Xie Y et al. (2010). Total power optimization for combinational logic using genetic algorithms, Journal of Signal Processing Systems, vol 58, No. 2, 145–160.
11. Mühlenbein H, Schomisch M et al. (1991). The parallel genetic algorithm as function optimizer, Parallel Computing, vol 17, No. 6, 619–632.
12. Buckley J J, and Hayashi Y (1994). Fuzzy genetic algorithm and applications, Fuzzy Sets and Systems, vol 61, No. 2, 129–136.
13. Parsopoulos K E, and Vrahatis M N (2010). Particle swarm optimization and intelligence: advances and applications, Information Science Reference Hershey, 1–4.
14. Liu B, Wang L et al. (2005). Improved particle swarm optimization combined with chaos, Chaos, Solitons & Fractals, vol 25, No. 5, 1261–1271.