Determining Equivalent Signal Lines by Weight Value Assignment for Logic Verification of Digital Circuits

Zhongliang Pan[1,*], Ling Chen[1] and Yihui Chen[2]

[1]*Zhongliang Pan, and Ling Chen are with the Dept. of Electronics, South China Normal University, Guangzhou, China, 510006*

[2]*Yihui Chen is with the Department of Secondary Vocational Education, Chengdu University, Chengdu, China, 610106.*

Abstract: The VLSI technology has led to the increased complexity in hardward design, therefore the verification for the correctness of circuit operations has become an exrtremely important task. The verification procedure can be reduced by means of the equivalent signal lines in the circuits. In this paper, a new method is presented for determining the equivalent signal lines, the method utilizes the weight value assignment of signal lines in circuits. First of all, the method makes use of the topological information of circuits to perform forward weight value assignments, assign weight values to the signal lines from the primary inputs to primary outputs. Afterwards, carry out the backward weight value assignment, assign weight values to the signal lines from the primary outputs to primary inputs. Secondly, carry out the random pattern simulation to further check the equivalence of signal lines. A lot of experimental results show that the verification of digital circuits can be carried out effectively by using the method proposed in this paper, the time being needed for the verification procedure can be cut down by utilizing the equivalent signal lines.

Keywords: Digital circuits, equivalent signal lines, logic verification, pattern simulation, weight values.

1. INTRODUCTION

With the rapid increase in the complexity of circuits, it is important to ensure the correctness of circuits being designed [1-3]. In the procedure of circuit design, a hierarchical design technique is usually adopted, which include the following design steps: system behavioral specification, behavioral synthesis, register transfer level (RTL) description, logic synthesis, logic description, layout synthesis, layout, etc. It is needed to verify the circuit design after an intermediate design step for the consistency and correctness with respects to the previous level of specification [4-6].

For the patents of the circuit verification, some methods had been proposed. For example, a method that can determine the functional equivalence of designs was presented [7], it can be used in the recognition of polynomial datapath equivalence in the register-transfer level (RTL) designs. The method of minimizing constraints in the circuit formal verification was investigated [8], the unisolated list of constraints comprising all known constraints, and the isolated list of constraints comprising none of the known constraints, were tackled in the method. The use of local variables in the circuit verification was investigated [9], the static formal verification was utilized to translates the property containing local variables into a data structure that can be used in the formal verification. An approach that can prove the correctness of multiplier and multiply-accumulate circuits was presented [10] which utilized the basic description of the circuit implementations and the structural similarity between the specifications and the circuit design under verifications. The verification of deadlock property was discussed [11], a verification method was proposed, which can implement the proof for the absence of the deadlock conditions in the system on chip (SoC).

Up till now, a lot of aspects for the circuit verification have been investigated, such as the verifications of digital circuits with general structure, the verifications of analog circuits and mixed-signal circuits, the circuit verifications based on assertion, the verifications of system-on-chip, the circuit verifications based on Petri nets. In the following, each aspect above is discussed in detail.

By verification of digital circuits, the main objectives are to investigate the techniques which are suitable for the features of digital circuits. The verification method in high-level synthesis of digital circuits was discussed [12]. The formal verification for the data path and controller generation phase was proposed, and the finite state machine description and register transfer-operations were used. A system development life cycle (SDLC) method was presented [13] for the asynchronous circuit verification, the method can be used to describe the desired asynchronous circuit behaviors and refine those descriptions. The verification approach of digital circuits by using a lot of internal and external destabilizing factors was investigated [14], the destabilizing factors impact on the circuit operations was considered when the verification was carried out. The verification of the nano CMOS circuit was discussed [15], and a gate model was given by

*Address correspondence to this author at the Department of Electronics, South China Normal University, Guangzhou, China, 510006;
Tel: 020-39310066; Fax: 020-39310882;
E-mails: panz@scnu.edu.cn, chenl067@126.com

using a simplified transistor model. The statistical simulation considering process variations was performed by means of the gate model. The effects of functional verification in circuit design were investigated [16]. The software system was used in the functional verification, where the verification and simulation environment contains generator and driver etc., the generator produces the inputs stimulus, and the driver translates the input stimuli into the input for the circuit under verification.

The verification process is more complex than digital circuits for the analog circuits and mixed-signal circuits. One of the main objective is to represent and model the analog signals in the circuits. Many methods had been proposed for this. For instance, the model checking of analog circuits for the specification of time constraints was discussed [17], an approach of defining time-based specifications was introduced. For the verification of mixed-signal circuits, the behavioral description of the mixed-signal circuit was transformed into a discrete model [18], and the verification was performed by formal digital verification technique. The static verification method was proposed [19], the method can verify the proper integration for the analog and mixed-signal macro-blocks into an integrated circuit, and the validity of the interconnections between the blocks was verified. The modeling and verification approach of analog/RF circuits in the presence of noise and process variations were presented [20], the approach made use of stochastic differential equations. A method for the design modification and verification of analog circuit was discussed [21], the method utilizes the previous circuit simulation results to the successive changed circuit analysis, and the method can be used for the verification of linear and nonlinear analog circuit. The verification techniques when there are shot noise, thermal noise and process variations in the analog circuits were investigated [22], the stochastic differential equations were used to model these noises.

For circuit verifications based on assertion, the assertions are specified using a variety of Boolean expressions as primitives, along with regular expressions and numerous temporal operators. The assertions can be used at various stages of the design process, such as the circuit verifications.

So far, a lot of method had been presented in the verifications based on assertion, for example, the hardware/ software co-verification technique was discussed [23], an integrated method of both the assertion-based verification and the object-oriented system-level synthesis was given. The assertion-based verification method for system-level design was investigated [24], where the transaction-level concepts were integrated with an assertion language, the method can specify system-level assertions for validating performance. The approach that combines a mixed-signal assertion language and the automatic verification algorithm was discussed [25], the approach was used to verify the heterogeneous systems of the digital, analog and software domain. A transaction level assertion verification method that was built on the top of the SystemC was given [26], the method can support the integration of assertion-based verification in a model driven design. The application of the transaction level assertion in a transaction driven verification environment was discussed [27], and a method to carry out the transaction level assertions by using the system Verilog scope rules was presented. The functional verification and debug of the complex circuits and systems was investigated [28], a method of modeling debugging changes was proposed by means of assertion statements and evaluation their effects, the set of assertions were transformed into a set of constraints that can be used by emulator.

In the aspect of the verifications for system-on-chip (SoC), a typical SoC has following components: a processor, a processor bus, a peripheral bus, a bridge between the two buses, memory devices, data processing and transformation devices, and data ports etc. The special subjects of SoC verifications are following: (1) check the integration between the various components. (2) The complexity of the multiple subsystems in the SoC may be huge, and there are many seemingly independent activities to be verified. (3) The software and hardware are verified simultaneously. Therefore, it is necessary to design special technique for SoC verifications.

For instance, the interrupt mechanism that joins the SoCs hardware and software behaviors was investigated [29], it modeled the interrupts as logical events rather than physical events, and the guidelines for composing the software components of interrupt service routines were given. The verification of multiprocessor system-on-chips (MSoC) was discussed [30], a model of MSoC was proposed, in the model, the multiprocessor was modelled by a number of processing elements, an application on the MSoC was modelled by task graph, the verification of MSoC was performed by translated the model to timed automata. The verification of the mixed-signal in a complex SoC was discussed [31], a high-level radio frequency model was built by using SystemVerilog language, the model can be executed on the digital simulators. An approach that can formally prove protocol compliance for the communication blocks in SoC was investigated [32], the approach used both the property checking on a bounded circuit model with the approximate reach-ability analysis. The SoC level verification for the inter connect of AMBA and AHB was discussed [33], the verification procedure was performed by using the SystemVerilog hardware description and verification language. The verification of interface protocols in the component-based SoC was investigated [34], an automatic stimulus generation method was presented by modeling the interface protocol with the help of the non-deterministic finite-state machine.

In the aspect of the circuit verifications based on Petri nets, the special type of Petri nets can be used to represent the circuit and its composition with the environment, hence, the circuit verification can be performed by means of the Petri nets. For example, the system representations and verification approaches for the asynchronous circuits was presented [35], it was shown that the combination of Petri nets and data flow graph was very effective for the verifications. The method that makes use of the special type of Petri nets to represent asynchronous circuits was discussed [36], the automatic conversion of a circuit netlist into an equivalent Petri net was described, the circuit verification was performed by means of finite prefixes of Petri net unfoldings. The verification model of digital integrated circuits by using of workflow Petri nets was proposed [37], the verification process can be formalized and quantified such that the cir-

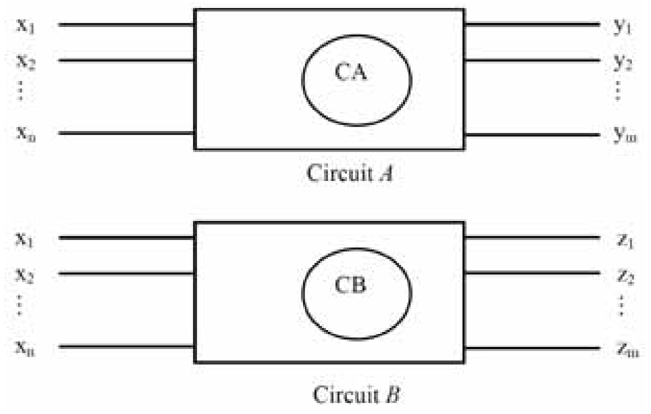

Fig. (1). The structure similarities.

cuits can be subsequently and structurally analyzed. The verification method for the analog and mixed-signal circuits by using Petri nets [38], the labeled hybrid Petri nets were used to model this heterogeneous set of components in the circuits, and a zone-based state space exploration algorithm was employed in the verification method.

In this paper, a new method is presented for determining the equivalent signal lines, the method utilizes the weight value assignment of signal lines in circuits, and can be used in the logic verification of digital circuits.

2. LOGIC VERIFICATION OF CIRCUITS

The verification is a process that ensures a circuit having been designed to exhibit the required behavior. The circuit design cycle consists of many synthesis stages. In general, the most of the synthesis process is implemented automatically, but a lot of errors may occur since the bugs in CAD tools or the human interference. Therefore, in order to ensure the correctness of the final circuit, it is necessary to perform the verification after the circuit has been carried out several synthesis steps.

One type of the circuit verification method is to utilize the information of circuit structure, where the circuit is described hierarchically, which consists of many components. A component is defined at one level and it is an interconnection of these components being defined at lower levels. The system specification of the circuit is made up of the behavioral descriptions of these components at all levels. The verification is performed to show that whether every component is able to fulfill its part of the specification.

Another type of the circuit verification method is to utilize the behavioral information of circuit, where the circuit is modeled by its inputs and outputs or state transitions. In order to detect the design errors, the logic simulation is carried out to compute the circuit responses for a lot of circuit input vectors. About the aspect of circuit hierarchy, the different input vectors are needed to generate for different instances of a component.

The above two types of circuit verification methods can be combined into a hybrid technique, which can employ the advantages of two types of verification methods. In this paper, the verification of two digital circuits is investigated, the main task is to check the functional equivalence of two digital circuits, one of which is the circuit before a specify synthesis step in the circuit design, and the other circuit is the post-synthesis version, where the gate level descriptions of two digital circuits are used.

3. DETERMINING EQUIVALENT LINES BY WEIGHT VALUE ASSIGNMENT

In general, there are some structure similarities between the two digital circuits to be verified. For example, there are a lot of same signal lines, same gates, same circuit blocks, etc., in the two circuits, and the logic functions of these signal lines are equivalent. This is shown in the Fig. (**1**).

Suppose the two circuits to be verified be circuit *A* and circuit *B*. The circuit *B* is the circuit after a synthesis step of the circuit *A*. Practically, in many cases only a small part (named as *CB*) in the circuit *B* is different from the part (named as *CA*) in the circuit *A*, which is shown in the Fig. (**1**). If the whole circuits of circuits *A* and *B* are functional equivalent, then the other parts in the circuits *A* and *B* should be equivalent. Therefore, there are a large number of equivalent signal lines in the circuits *A* and *B*. If the equivalent signal lines can be found and can be utilized in the verification procedure, then the computation complexity of the verification procedure can be reduced greatly. In this paper, a new method is presented for determining the equivalent signal lines by using weight value assignment. The method is made up of ten steps, which is shown in the following Algorithm 1.

ALGORITHM 1

Step 1. Let the two circuits to be verified be circuit *A* and circuit *B*. Let the primary inputs of the circuit *A* and circuit *B* be $x_1, x_2, \cdots, x_n$. Let the primary outputs of circuit *A* and circuit *B* be $y_1, y_2, \cdots, y_m$, and the $z_1, z_2, \cdots, z_m$, respectively.

Step 2. Compute the structure level of every signal lines in the circuit *A* and circuit *B*. The structure levels of all primary input lines are defined as 0. For all non-primary input lines, for instance, for signal line *u*, the structure level of *u* is defined as $S(u)= \max(S(v)) +1$, the *v* belongs to the fanin of *u*. Suppose the maximal values of structure level in the circuit *A* and circuit *B* be S_A and S_B, respectively.

Step 3. For the circuit *A*, use the topological information of circuits to perform forward weight value assignment, i.e., assign weight values to the signal lines from the primary inputs to primary outputs. The procedure is as follows:

Step 3.1. Let the parameter *k* be 0, i.e., $k=0$.

Step 3.2. Assign the values *k* to $k+n$ for the primary inputs $x_1, x_2, \cdots, x_n$, respectively.

Step 3.3. Compute the weight value of each signal line in the circuit by using the following mode. The weight of a fan-out line is the summation of the weights of all fan-out branch lines, here the weight of each branch line is equivalent each other. The weight of the output of a gate is the summation of the weights of its all input lines, the weight of each input line is equivalent each other.

Step 3.4. Make k:=(k+1). If the value of k is greater than n, then go to Step 4, else *store* the weight values of every signal line into the table GA[k], and go to the Step 3.2.

Step 4. For the circuit B, carry out the Step 3.1 to Step 3.4, the weight values of every signal line are stored in the table GB[k]. Let the parameter d be 0.

Step 5. For each signal line L_A (in the circuit A) with the value of structure level d, compare the corresponding n weight values in the table GA[k] and in the table GB[k]. If the n weight values of a signal line L_B (in the circuit B) are equal to the n weight values of signal line L_A respectively, then the signal lines L_A and L_B maybe equivalent, therefore the a node pair (L_A, L_B) is stored into a set ψ.

Repeatedly perform this step (the Step 5) for the structure level d from 1 to S_A. If for a structure level there are not node pairs that maybe equivalent, then the procedure of the Step 5 is stopped, the following Stop 6 continues.

Step 6. For the circuit A, carry out the backward weight value assignment, i.e., assign weight values to the signal lines from the primary outputs to primary inputs. The procedure is as follows:

Step 6.1. Let the parameter w be 0, i.e., w=0.

Step 6.2. Assign the values w to $w+m$ for the primary outputs $y_1, y_2, \cdots, y_m$, respectively.

Step 6.3. Compute the weight value of each signal line in the circuit by using the operations in Step 3.3.

Step 6.4. Make w:=(w+1). If the value of w is greater than m, then go to Step 7, else store the weight value of every signal line into the table HA[m], and go to the Step 6.2.

Step 7. For the circuit B, similarly carry out the Step 6.1 to Step 6.4, the weight values of every signal line are stored in the table HB[m]. Let the parameter h be (S_A–1).

Step 8. For each signal line Q_A (in the circuit A) with the value of structure level h, compare the corresponding m weight values in the table HA[m] and in the table HB[m]. If the m weight values of a line Q_B (in circuit B) are equal to the m weight values of line Q_A respectively, then the signal lines Q_A and Q_B maybe equivalent, therefore the node pair (Q_A, Q_B) is stored into the set ψ.

Repeatedly perform this step (the Step 8) for the structure level h from (S_A–2) to 0. If for a structure level there are not node pairs that maybe equivalent, then the procedure of the Step 8 is stopped.

Step 9. Carry out random pattern simulation for the circuit A and circuit B to further check the equivalence of each node pair in the set ψ.

Step 10. Make use of node pairs in the set ψ, and use Boolean satisfiability(SAT) approach to carry out the equivalence checking of the circuit A and circuit B.

The whole algorithm is terminated.

The implementation of random pattern simulation in the Step 9 of the Algorithm 1 is as follows. (1) Randomly generate a number of circuit input vectors, for example, the number of vectors being generated is N, where the N is a given positive integer. The value of each component in such an input vector is 0 or 1 randomly. (2) Apply all the circuit input vectors being generated to the primary inputs of the circuit. (3) Compute the values of all internal signal lines in the circuit, such that the values of all primary outputs are computed and obtained. In the Step 9 of the Algorithm 1, the value of the N can be chosen as a large integer in order to accurately check the equivalence of each node pair in the set ψ.

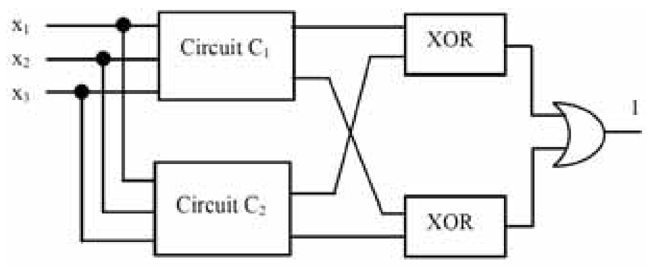

Fig. (2). The structure of constrain circuit.

In this paper, the equivalence checking of two given circuits C_1 and C_2 is formulated as a Boolean satisfiability problem by constructing a constrain circuit as shown in the Fig. (**2**). In the Fig. (**2**), there are three primary inputs and two primary outputs for the circuits C_1 and C_2.

In the constrain circuit, the corresponding primary inputs of the circuits C_1 and C_2 are tied together, and all corresponding primary outputs of the C_1 and C_2 drive several XOR gates which feed an OR gate. The output of the OR gate is set to the value 1.

For the SAT approach in the Step 10 of the Algorithm 1, the following two aspects are needed to carry out the equivalence checking of two circuits: (1) The computation of the CNF formula that is corresponding to the constrain circuit of the two circuits to be verified. (2) The CNF formulas are then given to a CNF-based SAT solver, and find the assignments that satisfy the CNF formulas.

The detail implementation of the SAT approach in the Step 10 of the Algorithm 1 is given in the following Section 4. The Algorithm 1 has been implemented in C++, the experimental results will be given in the Section 4.

4. EXPERIMENTAL RESULTS

The method in this paper for determining equivalent signal lines by using weight value assignment has been implemented in C++, and the method has been applied to carried out the logic verifications of digital circuits in ISCAS'85 benchmark circuits. A lot of experiments have been carried out on a personal computer with 3.0GHz and 1GB main memory. The structures of the ISCAS'85 benchmark circuits are given in the Table **1**.

In the Table **1**, the column "Circuits" denotes the name of a benchmark circuit. The column "PIs" shows the number of primary inputs in a circuit, the column "POs" demonstrates the number of primary outputs in a circuit. The column "Gates" gives the total number of the gates in a circuit. The column "Levels" denotes the number of structure level of the signal lines in a circuit.

Table 1. The ISCAS'85 benchmark circuits.

Circuits	PIs	POs	Gates	Levels
C880	60	26	383	25
C1355	41	32	546	25
C1908	33	25	880	41
C2670	233	140	1193	33
C3540	50	22	1669	48
C5315	178	123	2307	50
C6288	32	32	2416	125
C7552	207	108	3512	44

The implementation of Boolean satisfiability(SAT) approach in the Step 10 of the Algorithm 1 is as follows. The SAT approach constructs the conjunctive normal form (CNF) formula of the constrain circuit, and finds a satisfying assignment for the CNF formula by explicitly enumerating every satisfying pattern for the formula and checking its consistency.

Let the S be a finite set of variables that are over the set of Boolean values {0, 1}. A literal is an instance of a variable. A clause is a disjunction of several literals. A conjunctive normal form(CNF) formula *w* is a conjunction of several clauses.

An assignment μ satisfies a formula *w* if *w*(μ)=1, i.e., the formula is satisfied, and the assignment is a satisfying truth assignment. If there is no satisfying truth assignment for a formula, then the formula is unsatisfiable. For a given formula *w*, the Boolean satisfiability(SAT) approach is to find an assignment μ to satisfy the *w* or to prove that there are not such assignments.

For constructing the formula of the constrain circuit, the CNF formulas of the basic gates are given as follows. Suppose the inputs be x_1, x_2, ···, x_t, and the output be z for the AND, NAND, OR and NOR gates.

For the AND gates, the CNF formula is

$$\left[\prod_{j=1}^{t}(x_j+\bar{z})\right]\cdot\left(\sum_{j=1}^{t}\bar{x}_j+z\right)$$

For the NAND gates, the CNF formula is

$$\left[\prod_{j=1}^{t}(x_j+z)\right]\cdot\left(\sum_{j=1}^{t}\bar{x}_j+\bar{z}\right)$$

For the OR gates, the CNF formula is

$$\left[\prod_{j=1}^{t}(\bar{x}_j+z)\right]\cdot\left(\sum_{j=1}^{t}x_j+\bar{z}\right)$$

For the NOR gates, the CNF formula is

$$\left[\prod_{j=1}^{t}(\bar{x}_j+\bar{z})\right]\cdot\left(\sum_{j=1}^{t}x_j+z\right)$$

For the NOT gate with an input x_1 and the output z, the CNF formula is

$$(x_1+z)\cdot(\bar{x}_1+\bar{z})$$

For the XOR gate with the output z and two inputs x_1 and x_2, the CNF formula is

$$(\bar{x}_1+x_2+z)\cdot(x_1+\bar{x}_2+z)\cdot(x_1+x_2+\bar{z})\cdot(\bar{x}_1+\bar{x}_2+\bar{z})$$

In the following, an example is given for the computation of the CNF formula of a circuit. The CNF formula of the circuit shown in the Fig. (**3**) is given by

$$(\bar{x}_1+e_1)\cdot(\bar{x}_2+e_1)\cdot(x_1+x_2+\bar{e}_1)\cdot(x_3+e_2)\cdot(\bar{x}_3+\bar{e}_2)\cdot$$
$$(e_1+\bar{z})\cdot(e_2+\bar{z})\cdot(\bar{e}_1+\bar{e}_2+z)$$

Here, for a gate with the output z and two inputs x_1 and x_2, the CNF formula of the gate is given as follows: The CNF formula is $(x_1+\bar{z})\cdot(x_2+\bar{z})\cdot(\bar{x}_1+\bar{x}_2+z)$ for the AND gate. The CNF formula is $(\bar{x}_1+z)\cdot(\bar{x}_2+z)\cdot(x_1+x_2+\bar{z})$ for the OR gate.

When the CNF formula of the constrain circuit has been constructed, then the CNF formulas are given to a CNF-based SAT solver, and the SAT solver seeks the satisfying assignments to the variables that are consistent with all the constraints, or outputs the unsatisfiable if no such assignment exists. In this paper, the SAT solver is implemented by

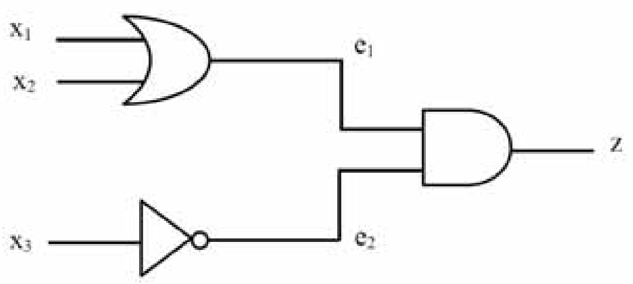

Fig. (3). The circuit about the CNF formula.

using the following steps.

(1) Create the CNF formula of the constrain circuit.

(2) Compute the number (named as λ) of each variable (or its complement) that appearing in the CNF formula. If a variable has bigger number λ, then the variable is in more front position of the ordering. There, a variable ordering can be obtained for the Boolean variables in the CNF formula that is corresponding to the circuit.

(3) Select a variable in terms of the variable ordering, and assign the variable to the value 0. For example, suppose the variable ordering be x_2, x_4, x_1, x_3, where the number λ of the x_2 is maximal, the number λ of the x_3 is minimal. The variable x_2 is selected firstly.

For the CNF formula of the constrain circuit, evaluate the clauses that contain the variable being selected in the above step (3), and propagate any implications from these clauses. If the contradiction is found, then retry the same variable with the value 1. If this also produces a contradiction, then backtrack to previous variable and try its value (0 or 1). If there are not contradictions, then try progressive variables until all variables are assigned. Therefore, the formula is satisfied. If there are contradictions, and continue to make the algorithm to backtrack before the first variable, then there are not satisfying assignments.

In the above step (3), the following implication mode is used: when one variable in a clause evaluates to 0, the remaining variables must be 1 for the clause (and therefore the formula is to be 1). This implied value is the transitive implication. For example, consider the following formula:

$$(A+B)\cdot(A+\overline{B}+C)\cdot(A+\overline{C}+D)\cdot(A+\overline{D}+E)\cdot(A+\overline{E}+F)$$

Let the variable A be 0, this implies B=1, which implies C=1, and produce several implications: D=1, E=1 and F=1.

In this paper, in the experiments for the ISCAS'85 benchmark circuits, the Algorithm 1 is used to carried out the equivalence checking of two types of circuits: The ISCAS'85 benchmark circuits and their modifying versions.

Firstly, a lot of node pairs with functional equivalence are obtained by using the Step 1 to Step 9 in the Algorithm 1.

Secondly, two equivalent nodes in each node pair are replaced by two new variables that are used in the verification procedure of circuits. Therefore, all equivalent node pairs are replaced by the new variables.

Thirdly, the SAT solver is used to perform the equivalence checking of two circuits.

In these experiments, for the whole procedure of equivalence checking of the ISCAS'85 benchmark circuits and their modifying versions, the time being needed by using the method proposed in this paper is: 0.14, 0.20, 0.12, 0.31, 0.24, 0.36, 0.43 and 0.72 for the circuits C880, C1355, C1908, C2670, C3540, C5315, C6288 and C7552, respectively, here all times are given in CPU minutes.

For the equivalence checking of circuits, in order to compare the method in this paper, we have also performed another experiments: where the SAT approach is used only, i.e., the equivalent node pairs are not used. The time being needed by using the approach is: 0.37, 0.58, 0.34, 0.86, 0.61, 1.09, 1.26 and 2.31 for the circuits C880, C1355, C1908, C2670, C3540, C5315, C6288 and C7552, respectively, here all times are given in CPU minutes.

Summarize these experimental results, it is shown that the equivalence checking of circuits can be carried out effectively by using the method proposed in this paper, the time being needed for the verification procedure can be cut down by using the equivalent node pairs. The experimental results also show that: if there are more structure similarities between the two circuits to be verified, then the method in this paper can obtain more equivalent node pairs and can perform the verification in shorter time.

5. CURRENT & FUTURE DEVELOPMENTS

For the complex circuit design it is required to verify the correctness of the circuit implementation with respect to its intended functionality. In this paper, a method is presented for determining the equivalent signal lines, the method utilizes the weight value assignment of signal lines in the circuits, and can be used in the logic verification of digital circuits to reduce the complexity of verification procedure. In the future, some work needs to be done for obtaining more equivalent node pairs by using the technique with low computation complexity.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This work were supported by National Natural Science Foundation of China (No.61072028, 61108039), the Project of Department of the Education of Guangdong Province (No.2012KJCX0040).

REFERENCES

[1] G. D. Guglielmo, L. D. Guglielmo, F. Fummi and G. Pravadelli, "Efficient generation of stimuli for functional verification by back jumping across extended FSMs", *J. Electron. Test. Theor. Appl.*, vol. 27, pp. 137-162, April 2011.

[2] L. Jinpeng, P. Kalla and F. Enescu, "Efficient grobner basis reductions for formal verification of Galois field arithmetic circuits", *IEEE Trans. Computer-Aided. Design.*, vol. 32, pp. 1409-1420, Sep. 2013.

[3] R. J. Wai, Y. W. Lin and H. C. Yang, "Experimental verification of total sliding-mode control for Chua's chaotic circuit", *IET Circ. Device Syst.*, vol. 5, pp. 451-461, Nov. 2011.

[4] M. K. Hsuan, and E. Rosenbaum, "Verification of snapback model by transient I-V measurement for circuit simulation of ESD response", *IEEE Trans. Device Mat. Rel.*, Vol. 13, pp. 371-378, June 2013.

[5] W. Denman, M. H. Zaki and S. Tahar, "Formal verification of bond graph modeled analogue circuits", *IET Circ. Device Syst.*, vol. 5, pp. 243-255, May 2011.

[6] M. Dehbashi, A. Sulflow and G. Fey, "Automated design debugging in a test bench-based verification environment", *Microprocess. Microsyst.*, Vol. 37, pp. 206-217, Mar. 2013.

[7] T. A. Drane and F. R. Exall, *"Method and apparatus for performing formal verification of polynomial data path"*, U. S. Patent 8, 527,924 B2, April 2012.

[8] P. Goyal and A. Jain. "*Constraint minimization method for formal verification*". U. S. Patent 8,316,332 B1, July 2010.

[9] J. Martensson, "*Static formal verification of a circuit design using properties defined with local variables*", U. S. Patent 8,225,249 B1, June 2008.

[10] K. Weber, M. Pflanz,C. Jacobi and U. Krautz, "*Method and system for formal verification of an electronic circuit design*", U. S. Patent 7,890,903 B2, May 2008.

[11] L. Loh and X. Sun, "*Formal verification of deadlock property*", U. S. Patent 8, 381, 148 B1, Feb. 2012.

[12] K. Chandan, D. Sarkar and C. Mandal, "Verification of datapath and controller generation phase in high-level synthesis of digital circuits", *IEEE Trans. Computer-Aided Design*, vol. 29, pp. 479-492, 2010.

[13] K. Lehuu, T. Nguyen, H. B. Thang and A. V. Dinhduc, "Asynchronous circuit verification: from specification to circuit", In: *International Conference on Computing, Management and Telecommunications,* Ho Chi Minh, Vietnam, 2013, pp. 420-425.

[14] R. Goldman, V. Melikyan and E. Babayan, "Digital circuits verification with consideration of destabilizing factors", In: *IEEE 6th International Design and Test Workshop,* Beirut, Lebanon, 2011, pp. 93-98.

[15] T. Qin, A. Zjajo and M. Berkelaar, "Transistor-level gate modeling for nano CMOS circuit verification considering statistical process variations", In: *20th International Workshop on Power and Timing Modeling, Optimization, and Simulation (PATMOS),* Grenoble, France, 2010, pp. 190-199.

[16] I. Rancea and V. Sgarciu, "Functional verification of digital circuits using a software system". In: *IEEE International Conference on Automation, Quality and Testing, Robotics,* ClujNapoca, Romania, 2008, pp. 152-157.

[17] D. Grabowski, D. Platte, L. Hedrich and E. Barke, "Time constrained verification of analog circuits using model-checking algorithms", *Electron. Notes Theor. Comput. Sci.*, vol. 153, pp. 37-52, June 2006.

[18] J. Schonherr, M. Freibothe, B. Straube and J. Bormann, "Semi-formal verification of the steady state behavior of mixed-signal circuits by SAT-based property checking", *Theoret. Comput. Sci.*, vol. 404, pp. 293-307, Sept. 2008.

[19] R. Mukhopadhyay, A. Komuravelli, P. Dasgupta and S. K. Panda, "A static verification approach for architectural integration of mixed-signal integrated circuits", *Integ. VLSI. J.*, vol. 43, pp. 58-71, Jan. 2010.

[20] R. Narayanan, M. H. Zaki and S. Tahar, "Using stochastic differential equation for verification of noise in analog/RF circuits", *J. Elec. Test.*, vol. 26, pp. 97-109, Feb. 2010.

[21] T. Y. Zhou, L. Hang, Z. Dian and T. Tarim, "A fast analog circuit analysis algorithm for design modification and verification", *IEEE. Trans. Computer-Aided Desig.*, vol. 30, pp. 308-313, Feb. 2011.

[22] R. Narayanan, I. Seghaier, M. H. Zaki andS. Tahar, "Statistical runtime verification of analog circuits in presence of noise and process variation", *IEEE. Trans. {VLSI} Syst.*, vol. 21, pp. 1811-1822, Oct 2013.

[23] A. M. Gharehbaghi, B. H. Yaran, S. Hessabi and M. Goudarzi, "An assertion-based verification methodology for system-level design", *Comp. Electro. Eng.*, vol. 33, pp. 269-284, July 2007.

[24] H. Ardakani, A. M. Gharehbaghi and S. Hessabi, "A performance and functional assertion-based verification methodology at transaction-level", In: *International Conference on Microelectronics*, Cairo, Egypt, 2007, pp. 133-136.

[25] S. Lammermann, J. Ruf, T. Kropf, W. Rosenstiel and A. Viehl, "Towards assertion-based verificationof heterogeneous system designs", In: *Design, Automation & Test in Europe Conference (DATE),* Dresden, Germany, 2010, pp. 1171-1176.

[26] K. Tomasena, J. F. Sevillano, J. Perez, A. Cortes, and I. Velez, "A transaction level assertion verification framework in SystemC: an application study", In: *International Conference on Advances in Circuits, Electronics and Micro-electronics, Sliema, Malta*, 2009, pp. 75-80.

[27] N. Sudhish, B.R. Raghavendra, and H. Yagain, "An efficient method for using transaction level assertions in a class based verification environment", In: *International Symposium on Electronic System Design,* Kerala, India, 2011, pp. 72-76.

[28] S. Banerjee, T. Gupta, and S. Gupta, "Logic emulation with forced assertions: a methodology for rapid functional verification and debug", In: *5th Asia Symposium on Quality Electronic Design, Penang,* Malaysia, 2013, pp. 312-320.

[29] X. Xiaoxi, and C. C. Lim, "Modeling interrupts for software-based system-on-chip verification", *IEEE Trans. Computer-Aided Design.*, vol. 29, pp. 993-997, June 2010.

[30] A. Brekling, M.R. Hansen and J. Madsen, "Models and formal verification of multiprocessor system-on-chips", *J. Logic Algebr. Progr.*, vol. 77, pp. 1-19, Sep. 2008.

[31] Y. Xiaokun, N. Xinwei, F. Jeffrey and C. Chiu, "Mixed-signal system-on-a-chip(SoC) verification based on System Verilog model", In: *45th Southeastern Symposium on System Theory,* Waco, USA, 2013, pp. 17-21.

[32] M.D. Nguyen, M. Thalmaier, M. Wedler, J. Bormann, D. Stoffel, and W. Kunz, "Unbounded protocol compliance verification using interval property checking with invariants", *IEEE Trans. Comp. Aided. Design. Integr. Circuit. Syst,* vol. 27, pp. 2068-2082, Nov. 2008.

[33] P.D. Mulani, "SoC level verification using system Verilog", In: *International Conference on Emerging Trends in Engineering and Technology,* Nagpur, India, 2009, pp. 378-380.

[34] S. Chehua, H. Juinndar, and J. Jingyang, "Automatic verification stimulus generation for interface protocols modeled with non-deterministic extended FSM", *IEEE. Trans. {VLSI} Syst.*, vol. 17, pp. 723-727, May 2009.

[35] T.H. Bui, T. Nguyen, and A.V. Dinhduc. "Experiences with representations and verification for asynchronous circuits", In: *4th International Conference on Communications and Electronics,* Hue, Vietnam, 2012, pp. 459-464.

[36] I. Poliakov, A. Mokhov, A. Rafiev, D. Sokolov, and A. Yakovlev, "Automated verification of asynchronous circuits using circuit Petri nets", In: *14th IEEE International Symposium on Asynchronous Circuits and Systems,* Newcastle upon Tyne, United Kingdom, 2008, pp. 161-170.

[37] K. Weinberger, S. Bulach, and R. Bosch, "Application of workflow Petri nets to modeling of formal verification processes in design flow of digital integrated circuits", In: *Design, Automation and Test in Europe, Munich,* Germany, 2008. pp. 937-938.

[38] S. Little, D. Walter, C. Myers, R. Thacker, S. Batchu, and T. Yoneda., "Verification of analog/mixed-signal circuits using labeled hybrid Petri nets", *IEEE Trans. Comp. Aided Design.*, vol. 30, pp. 617-630, April, 2011.

Received: June 09, 2014 Revised: June 22, 2014 Accepted: July 24, 2014