SAT-based verification of implementations of incompletely specified Boolean functions

Liudmila Cheremisinova

United Institute of Informatics
Problems of NAS of Belarus
Minsk, Belarus
e-mail: cld@newman.bas-net.by

ABSTRACT

The problem under discussion is to check whether a given system of incompletely specified Boolean functions is implemented by a logical description with functional indeterminacy that is represented by a system of connected blocks each of which is specified by a system of completely or incompletely specified Boolean functions. SAT-based verification methods are considered which formulate the verification problem as checking satisfiability of a conjunctive normal form. The results of investigation of SAT-based verification methods are given.

Keywords

Computer-aided design, formal verification, satisfiabilty of a conjunctive normal form

1. INTRODUCTION

The role of combinational verification becomes more and more important with the rapid increase of the complexity of designs synthesized by modern CAD (computer-aided design) tools. Today, verification is a bottleneck in the overall VLSI design cycle as it consumes up to 70% of design effort [1, 2]. Unfortunately, the capabilities of verification tools has noticeably yielded to the capabilities of design systems, to say nothing of the technological achievements of the semiconductor industry. The problem of verification is to prove the behavioral equivalence of two descriptions of the same device representing different design solutions obtained in a microelectronic device design process.

At present, the approach widely used in industry for the verification of the correctness of integrated circuits is the logical simulation [3] because of its scalability and execution time's predictability. Since in the general case the simulation cannot provide complete verification (because the upper estimate of the number of test sets is equal to 2^n, where n is the number of arguments of the verified descriptions), it is insufficient to use only simulation for the solution of the considered problem [4].

In recent decades, formal verification methods that aim to prove the functional identity of projects in a formal way have been developed rapidly as an alternative to simulation-based verification. The methods are based on reduction of the verification problem to a problem of the satisfiability checking of a conjunctive normal form (CNF) [1–3], making it possible to ensure the completeness of the verification, in contrast to simulation methods. The development of these methods was facilitated by the significant progress in solving CNF satisfiability problem observed in the recent decades: state-of-the-art SAT-solvers [5–7] make it possible to work with CNF that include thousands of disjuncts and variables.

In a typical verification scenario, there are two circuit implementations of the same design, and the problem is to prove their functional equivalence. In contrast to that in the paper, the verification task is examined for the case, when the desired functionality of the system under design is incompletely specified.

We consider the verification problem for the case, when the desired incompletely specified functionality is given in the form of a system of incompletely specified Boolean functions (ISFs) and the compared functional description represents a multi-block structure with blocks specified by systems of completely or incompletely specified functions. A special case of such a multi-block structure is a combinational network or an ISF system. There is one-to-one correspondence between the arguments and functions of both the compared descriptions. ISFs are specified on intervals (cubes) of values of Boolean input variables, and the intervals are large enough.

2. BASIC DEFINITIONS

An ISF system $F(\boldsymbol{x}) = \{f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})\}$ (where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ is a vector) is represented as a mapping of n-dimensional Boolean space B^n of vectors $\boldsymbol{x}_i$ into m-dimensional space $\{0,1,–\}^m$ of vectors $\boldsymbol{f}_j$, where the symbol "–" denotes don't-care condition. A completely specified Boolean function $f(\boldsymbol{x})$ implements (covers) an ISF $g(\boldsymbol{x})$ iff $f(\boldsymbol{x})$ can be derived from $g(\boldsymbol{x})$ by assigning either 0 or 1 to each don't-care point of B^n. An ISF is specified by off-set U_f^0, on-set U_f^1 and dc-set U_f^{dc} as subsets of B^n ($U_f^1 \cup U_f^0 \cup U_f^{dc} = B^n$).

Let us specify a system $F(\boldsymbol{x})$ as a set I_F of multiple-output cubes $(\boldsymbol{u}, \boldsymbol{t})$ each of which is a pair of ternary vectors $\boldsymbol{u}$ and $\boldsymbol{t}$ of sizes n and m. Here the input part of the cube $\boldsymbol{u}$ is a cube in B^n or a set of minterms $\boldsymbol{b}_i$ (elements of B^n), and $\boldsymbol{u}$ can be represented by a conjunction of some literals (variables $x_i \in \boldsymbol{x}$ or its inversions). The output part $\boldsymbol{t}$ is a ternary vector of values of functions for the cube $\boldsymbol{u}$, or $\boldsymbol{t}$ is a conjunction of some literals $f_j \in F$. For each $f_j \in F$ the j-th entry t^j of $\boldsymbol{t}$ is 1 or 0 ($t^j = 1$, 0) if all the minterms of the cube $\boldsymbol{u}$ are in the on-set U_{fj}^1 or in the off-set U_{fj}^0 correspondingly; otherwise t^j is don't-care.

A CNF represents a completely specified Boolean function as a conjunction of one or more clauses, each being in its turn a disjunction of literals. CNF representation is popular among SAT algorithms because each clause must be satisfied (evaluated to 1) for the overall CNF to be satisfied. The SAT problem is concerned with finding a truth assignment of CNF literals, which simultaneously satisfies each of its member clauses. If such an assignment exists the CNF is referred to as satisfiable, and the assignment is known as a satisfying assignment. Matrix representation of CNF formula C containing k clauses and p distinct variables is a ternary matrix $\boldsymbol{C}$ having k rows and p columns.

A system $F(\boldsymbol{x})$ of ISFs given by the set I_F of multiple-output cubes $(\boldsymbol{u}_i, \boldsymbol{t}_i)$ can be represented in matrix form by a pair of ternary matrices $\boldsymbol{U}$ and $\boldsymbol{T}$ of the same cardinalities. For example, ISF system $F(\boldsymbol{x})$ specified by $I_F = \{(x_3 x_4 x_5, f_1), (\bar{x}_2\, \bar{x}_3\, \bar{x}_4, \bar{f}_1 f_2), (\bar{x}_2 x_4 x_5, f_2), (\bar{x}_1 x_2\, \bar{x}_5, \bar{f}_1\, \bar{f}_2), (\bar{x}_2 x_3\, \bar{x}_4, \bar{f}_2), (x_1 x_2, f_1\, \bar{f}_2)\}$ is shown in Fig. 1,a.

$\boldsymbol{U}$					$\boldsymbol{T}$		
x_1	x_2	x_3	x_4	x_5	f_1	f_2	
–	–	1	1	1	1	–	1
–	0	0	0	–	0	1	2
–	0	–	1	1	–	1	3
0	1	–	–	0	0	0	4
–	0	1	0	–	–	0	5
1	1	–	–	–	1	0	6

a)

$\boldsymbol{P}^k$													
x_1	x_2	x_3	x_4	x_5	f_1	f_2	w_1	w_2	w_3	w_4	w_5	w_6	
–	–	1	–	–	–	–	1	–	–	–	–	–	1
–	–	–	1	–	–	–	1	–	–	–	–	–	2
–	–	–	–	1	–	–	1	–	–	–	–	–	3
–	–	–	–	–	0	–	1	–	–	–	–	–	4
–	0	–	–	–	–	–	–	1	–	–	–	–	5
–	–	0	–	–	–	–	–	1	–	–	–	–	6
–	–	–	0	–	–	–	–	1	–	–	–	–	7
–	–	–	–	–	1	0	–	1	–	–	–	–	8
…													
–	–	–	–	–	–	–	0	0	0	0	0	0	25

b)

Figure 1. An example of: a) ISF system in matrix form and b) its encoded prohibitive CNF P^{κ}

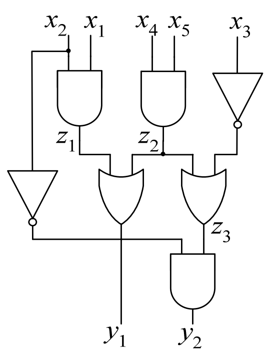
3. SAT-BASED APPROACH TO VERIFICATION

The past ten years have seen efforts in developing commercial formal verification tools (by reducing to SAT) that provide more general results than traditional simulation methods: it is possible to guarantee that a specific property holds for a design under all possible input stimuli. In a modern combinational equivalence checking flow based on formal verification approach, both networks to be verified are transformed into a single comparing circuit. It is derived by combining the pairs of inputs with the same names and feeding the pairs of outputs with the same names into EXOR gates, which are ORed to produce the single output of the comparing circuit. There is constant 0 on the output if and only if the two original circuits are equivalent.

To test whether the circuit output is 1 or 0, the conventional CNF is produced for it. Once the overall problem is formulated in CNF, a SAT solver can be used to solve it [5–7]. A circuit-to-CNF conversion uses as many variables as there are primary inputs and gates in the circuit: for output of each gate its own internal Boolean variable is introduced. And a local CNF is associated with each gate. Then local CNFs are joined in the overall network CNF $C(S)$ by the conjunction operation (Fig. 2). The derivation of the local CNF for a gate representing a local function $y = f(z_1, z_2, \ldots, z_k)$ is based on defining a new Boolean function $\varphi(y,f) = y \sim f$ [5], that is true in the only case when both functions y and f assume the same value. Here are the conventional CNF representations of NOT, n-input AND and OR functions:

$$(z \vee y)(\bar{z} \vee \bar{y});$$
$$(z_1 \vee \bar{y})(z_2 \vee \bar{y}) \ldots (z_n \vee \bar{y})(\bar{z}_1 \vee \bar{z}_2 \vee \ldots \vee \bar{z}_n \vee y);$$
$$(\bar{z}_1 \vee y)(\bar{z}_2 \vee y) \ldots (\bar{z}_n \vee y)(z_1 \vee z_2 \vee \ldots \vee z_n \vee \bar{y}).$$

Further we are focusing on the case when the first of descriptions to be compared is an ISF system and the second one is represented by some sort of multi-block structure. We consider two cases: 1) the structure has no indeterminacy and each its block is represented by CNF system; 2) the structure has indeterminacy and each its block is represented by ISF system.



x_1	x_2	x_3	x_4	x_5	z_1	z_2	z_3	y_1	y_2	
1	–	–	–	–	0	–	–	–	–	1
–	1	–	–	–	0	–	–	–	–	2
0	0	–	–	–	1	–	–	–	–	3
–	–	–	1	–	–	0	–	–	–	4
–	–	–	–	1	–	0	–	–	–	5
–	–	–	0	0	–	1	–	–	–	6
–	–	0	–	–	–	1	0	–	–	7
–	–	1	–	–	–	–	1	–	–	8
–	–	–	–	–	–	0	1	–	–	9
–	–	–	–	–	1	1	–	0	–	10
–	–	–	–	–	0	–	–	1	–	11
–	–	–	–	–	–	0	–	1	–	12
–	0	–	–	–	–	–	–	–	0	13
–	–	–	–	–	–	–	1	–	0	14
–	1	–	–	–	–	–	0	–	1	15

a) b)

Figure 2. An example of: a) combinational circuit and b) its conventional CNF

The above traditional approach which constructs a comparing circuit cannot be applied, just as it is, for the declared cases because at least one of the given forms of functional representation may be specified incompletely.

To reduce the verification problem to SAT we construct two CNFs $P(F)$ and $C(S)$. CNF $P(F)$ describes all assignments contradictory to the first form (ISF system) and is called *prohibitive* CNF of the ISF system. CNF $C(S)$ describes all possible assignments for the second form (logical network or multi-block structure), and it is called traditionally as *conventional* CNF in the case of the structure without indeterminacy (combinational circuit) or, otherwise, it is called *permissible* CNF that is some sort of the conventional CNF for a structure with indeterminacy.

Assertion. A multi-block structure S implements an ISF system $F(\boldsymbol{x})$ if and only if CNF $P(F) \wedge C(S)$ is unsatisfiable [8, 9].

4. SAT BASED VERIFICATION CASES

4.1. Verification case 1

In the case when a multi-block structure has no indeterminacy, it can be easily transformed into a multi-level combinational network which consists of NOT, AND and OR gates. And a problem under discussion is to verify if a given network implements the ISF system. It is true if it takes place for each multiple-output cube. In terms of network CNF, this condition could be reformulated as follows [8]: for every multiple-output cube $(\boldsymbol{u}_i, \boldsymbol{t}_i) \in I_F$ a partial value assignment satisfying the conjunction $\boldsymbol{u}_i \boldsymbol{t}_i$ should satisfy the network CNF.

In other words, a network implements ISF system $F(\boldsymbol{x})$, iff for every multiple-output cube $(\boldsymbol{u}_i, \boldsymbol{t}_i) \in I_F$ a partial value assignment satisfying the conjunction $\boldsymbol{u}_i\, \bar{\boldsymbol{t}}_i$ (i.e. contradicting to $\boldsymbol{u}_i, \boldsymbol{t}_i$) is unsatisfying assignment for the network CNF. If $\boldsymbol{u}_i = x_1^i x_2^i \ldots x_{ni}^i$ and $\boldsymbol{t}_i = f_1^i f_2^i \ldots f_{mi}^i$ then the CNF P_i specifying the contradiction of the multiple-output cube $(\boldsymbol{u}_i, \boldsymbol{t}_i)$, called as the cube-prohibitive CNF, consists of the following $n_i + 1$ clauses:

$$P_i(\boldsymbol{x}, \boldsymbol{f}) = x_1^i x_2^i \ldots x_{ni}^i (\bar{f}_1^i \vee \bar{f}_2^i \vee \ldots \vee \bar{f}_{mi}^i).$$

For example, the prohibitive CNF (Fig. 1) for the multiple-output cube $s_6 = (x_1 x_2, f_1 \bar{f}_2)$ from $F(\mathbf{x}) = (\mathbf{U}, \mathbf{T})$ has three clauses: $P_6(\mathbf{x}, \mathbf{f}) = (x_1)(x_2)(\bar{f}_1 \vee f_2)$.

Appending clauses of P_i to the network CNF $C(S)$ results in CNF $C(P_i) = C(S) \wedge P_i$. It is not difficult to prove that CNF $C(P_i)$ is satisfiable iff the network does not implement the i-th multiple-output cube. As to the whole ISF system it is not implemented by the network iff at least one of its multiple-output cubes is not implemented by the network, i.e. if the following formula is satisfiable [10]:

$$C = C(S) \wedge P(F) = C \wedge (P_1 \vee P_2 \vee \ldots \vee P_l), \qquad (1)$$

where $P(F)$ is the ISF system prohibitive CNF.

To apply any SAT-solver to check whether for the CNF C a satisfying assignment exists it is necessary to convert the formula $P(F)$ to a CNF form. Theoretically this could be done always, but it is NP-hard problem. We are interested in a method of construction of ISF system prohibitive CNF $P(F)$ having linear complexity. Next the method is proposed that is based on encoding multiple-output cubes and their prohibitive CNFs using coding variables $w_i \in \mathbf{w}$. After encoding, prohibitive CNFs $P_i(\mathbf{x}, \mathbf{f})$ are transformed into encoded prohibitive CNFs $P_i^{\kappa}(\mathbf{x}, \mathbf{f}, \mathbf{w})$ and the formula (1) becomes

$$C^{\kappa} = C \wedge (P_1^k \wedge P_2^k \wedge \ldots \wedge P_l^k) \wedge Q(\mathbf{w}), \qquad (2)$$

where $Q(\mathbf{w})$ provides that the CNF C^{κ} will be satisfiable iff at least one CNF $P_i \in P(F)$ is satisfiable. From now on $Q(\mathbf{w})$ is called as *alternative* CNF.

When transforming the formula (1) into the CNF form (2) each cube-prohibitive CNF P_i is encoded by a code in the form of a disjunction $d_i = w_{i1}^{\sigma i1} \vee w_{i2}^{\sigma i2} \vee \ldots \vee w_{ir}^{\sigma ir}$ ($\sigma_{ir} \in \{0,1\}$, $w_{ir}^1 = w_{ir}$ and $w_{ir}^0 = \bar{w}_{ir}$, and $w_{ij} \in \mathbf{w}$):

$$P_i^k(\mathbf{x}, \mathbf{f}, \mathbf{w}) = (x_1^i \vee d_i) \ldots (x^i_{ni} \vee d_i)(\bar{f}_1^i \vee \ldots \vee \bar{f}_{mi}^i \vee d_i) \quad (3)$$

To formulate the conditions the alternative CNF $Q(\mathbf{w})$ in (2) must satisfy for the chosen cube-prohibitive CNF encoding, let us denote by f_Q and f_{di} the functions represented by $Q(\mathbf{w})$ and $d_i(\mathbf{w})$ and by U^1_Q and U^1_{di} – their on-sets.

Assertion **[**10]. Any alternative CNF $Q(\mathbf{w})$ for a given encoding of cube-prohibitive CNFs must satisfy the following conditions:

1) $(\bigwedge_i f_{di}) \wedge f_Q = 0$ or $(\bigcap_i M_{di}^1) \cap M_Q^1 = \varnothing$;

2) $(\bigwedge_{i \neq j} f_{di}) \wedge f_Q \neq 0$ or $(\bigcap_{i \neq j} M_{di}^1) \cap M_Q^1 \neq \varnothing$ for all j.

The first condition ensures the CNF $P(\mathbf{x}, \mathbf{f}, \mathbf{w}) = (P_1^k \wedge P_2^k \wedge \ldots \wedge P_l^k) \wedge Q$ be unsatisfiable when the circuit implements the analyzed ISF system, i.e. when all cube-prohibitive CNFs $P_i(\mathbf{x}, \mathbf{f})$ are unsatisfiable. The second condition ensures the CNF $P(\mathbf{x}, \mathbf{f}, \mathbf{w})$ be satisfiable when the circuit does not implement the analyzed ISF system, i.e. there exists at least one multiple-output cube, for example j-th one, that is not realized by it. Thus, a variable assignment can be found satisfying the cube-prohibitive CNF $P_j(\mathbf{x}, \mathbf{f})$ (and $P_j^k(\mathbf{x}, \mathbf{f}, \mathbf{w})$, too). Fulfillment of the second condition guarantees that there exists at least one assignment of coding variables that ensures satisfiability of $Q(\mathbf{w})$ and all cube prohibitive CNFs P_i^k except the j-th one (that is satisfiable by the assumption).

Two basic methods of encoding multiple-output cubes (satisfying the above Assertion) have been investigated: encoding by codes of unit [8] and logarithmic length [9]. The first method supposes to introduce as many coding variables w_i as there exist multiple-output cubes in the ISF system specification I_F. The second method introduces the minimal number of coding variables that is $r = \lceil log_2|I_F| \rceil$. The method of encoding by codes of logarithmic length allows to reduce substantially the number of coding variables as compared with the method of encoding by unary codes, but codes (and CNF clauses) are dense enough.

Further we focus upon increasing efficiency of verification process using unary encoding of multiple-output cubes. Using it $P_i^k(\mathbf{x}, \mathbf{f}, \mathbf{w})$ (3) changes for the following encoded form:

$$P_i^{\kappa} = (x_1^i \vee w_i)(x_2^i \vee w_i) \ldots (x^i_{ni} \vee w_i)(\bar{f}_1^i \vee \ldots \vee \bar{f}_{mi}^i \vee w_i),$$

and the alternative CNF Q satisfying the above Assertion becomes: $Q = \bar{w}_1 \vee \bar{w}_2 \vee \ldots \vee \bar{w}_l$.

For example, the fragment of the prohibitive CNF for the ISF system in the matrix form is shown in Fig. 1,b. If we combine the circuit conventional CNF (Fig. 2,b) and the prohibitive CNF (identifying y_1, y_2 with f_1, f_2) and then carry out the satisfiability test of the resulting CNF, we may make sure that it is nonsatisfiable. Therefore, the circuit (Fig. 2,a) implements the ISF system (Fig. 1,a).

4.2. Verification case 2

Here we consider the verification problem for the case, when both compared descriptions are incompletely specified. For example, we have a multi-block structure S with indeterminacy such as one in Fig. 3 and an ISF system $F(\mathbf{x})$ such as one in Fig. 1,a.

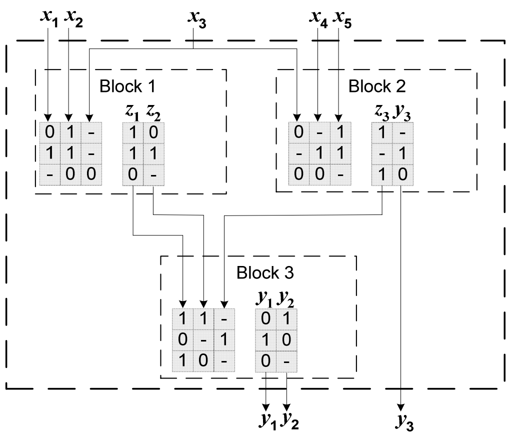


Figure 3. Three-block structure with indeterminacy

Just as in the previous section we formulate the verification problem as verifying whether CNF $C = C(S) \wedge P(F)$ is satisfiable. Here $P(F)$ is the ISF system $F(\mathbf{x})$ prohibitive CNF, $C(S)$ is some sort of the conventional CNF but only for a multi-block structure with indeterminacy or for an ISF system (that can be considered instead of one-block structure). To distinguish it from the conventional CNF let's call it further as the *permissible* CNF. The permissible CNF describes the set of admissible combinations of signals on all the nodes of structure blocks; i.e., each set of values satisfying the CNF is admissible for this structure. The permissible CNF $C(S)$ is the conjunction of permissible CNFs $C(B_i)$ of its blocks or permissible CNFs $C(F_i)$ of their ISF systems.

Three methods of construction of a permissible CNF for an ISF system are proposed. The first one is based on the paraphrased representation of ISFs [11]. And two methods are based on the application of implicative conditions: implication [12] and implication with coding of conditions [13] methods. Here we dwell on the implication method.

Assertion [12]. The permissible CNF $C(G)$ of an ISF system $G(\boldsymbol{x})$ defined by a set of its multiple-output cubes $s_i = (\boldsymbol{u}_i, \boldsymbol{t}_i)$ ($i = 1, 2, \dots, r$) is generated by the formula:

$$(\boldsymbol{u}_1 \rightarrow \boldsymbol{t}_1) \wedge (\boldsymbol{u}_2 \rightarrow \boldsymbol{t}_2) \wedge \dots \wedge (\boldsymbol{u}_r \rightarrow \boldsymbol{t}_r).$$

The permissible CNF for a multiple-output cube $s_i^G = (\boldsymbol{u}_i, \boldsymbol{t}_i)$ with $\boldsymbol{u}_i = x_1^i x_2^i \dots x_{ni}^i$ and $\boldsymbol{t}_i^G = y_1^i y_2^i \dots y_{mi}^i$ consists of as many clauses as the size of the term $\boldsymbol{t}_i$ is:

$$C_i = (\boldsymbol{u}_i \rightarrow \boldsymbol{t}_i) = \bar{\boldsymbol{u}}_i \vee \boldsymbol{t}_i = \bar{x}_1^i \vee \bar{x}_2^i \vee \dots \vee \bar{x}_{ni}^i \vee (y_1^i y_2^i \dots y_{mi}^i) =$$
$$=(\bar{x}_1^i \vee \bar{x}_2^i \vee \dots \vee \bar{x}_{ni}^i \vee y_1^i) \wedge \dots \wedge (\bar{x}_1^i \vee \bar{x}_2^i \vee \dots \vee \bar{x}_{ni}^i \vee y_{mi}^i).$$

For example, the permissible CNF for the first block of the structure (Fig. 3) consists of five clauses: $(x_1 \vee \bar{x}_2 \vee z_1) \wedge (x_1 \vee \bar{x}_2 \vee \bar{z}_2)(x_1 \vee x_2 \vee z_1)(x_1 \vee x_2 \vee z_2)(\bar{x}_2 \vee \bar{x}_3 \vee \bar{z}_1)$.

The key idea of the proposed method of checking whether an ISF system $F(\boldsymbol{x})$ is implemented by a multi-block structure S with indeterminacy is as follows. We obtain the prohibitive CNF $P(F)$ and the permissible CNF $C(S)$.

Assertion. A multi-block structure S with indeterminacy implements an ISF system $F(\boldsymbol{x})$ iff the CNF $P(F) \wedge C(S)$ is unsatisfiable.

One can make sure after constructing $P(F) \wedge C(S)$ for the ISF system in Fig. 1,a and three-block structure in Fig. 3 that there is no satisfying assignment for the CNF. Thus, the structure implements the ISF system.

4.3. Organization of SAT problem solving for verification

Three verification methods are proposed [10] that are based on successive, simultaneous and group testing multiple-output cubes from I_F. The first method formulates as many SAT problems as the number of cubes are there in an ISF system tested. The second method formulates verification task as the only SAT problem (using coding the cubes as shown above).The third method divides the overall set I_F of multiple-output cubes into groups and formulates as many SAT problems as the number of groups are there.

The investigations of the methods [10] have shown that the group method is more effective because it allows 1) to achieve trade-offs between expenses on forming data for SAT-solver and SAT-solver performance; and thereby 2) to reduce the overall verification time.

5. EXPERIMENTAL RESULTS

All the mentioned verification methods have been implemented on C++ programming language. Then the programs were investigated on the sets of pseudo-random pairs of descriptions: ISF system and multi-block structure implementing it (with or without indeterminacy). MiniSat solver [7] has been used in the experiments. The goal of experiments was 1) to compare the competitive methods solving the same task on the same set of examples; 2) to investigate experimentally domains of preferable usage of the proposed methods; 3) to find out the most effective value of group size for group testing verification methods. The experiments have shown that:

1) the group size about 200 gives good enough results: group methods gain stably in efficiency compared with the methods of successive and simultaneous testing of multiple-output cubes, the win gain is about 35% over the method of simultaneous testing;
2) substantial reduction of variables when using logarithmic encoding of multiple-output cubes did not bring about substantial speedup of the solution of verification problem;
3) despite the fact that the implication method is simpler than that of implication with condition coding and gives shorter CNFs, it has smaller speed.
4) simulation based verification methods have 60 times greater speed on average than SAT based methods solving the same task.

REFERENCES

[1] A. Wiemann, *Standardized functional Verification*, Springer, San Carlos, CA USA, 2008.
[2] A. Kuehlmann, A.J. Cornelis van Eijk, "Combinational and Sequential Equivalence Checking", in: *Logic synthesis and Verification* (Ed. S. Hassoun, T. Sasao, R.K. Brayton), Kluwer, pp. 343–372, 2002.
[3] W.K. Lam, *Hardware Design Verification: Simulation and Formal Method-Based Approaches*, New York: Prentice Hall, 2005.
[4] L. Li, M.A. Thornton, and S.A. Szygenda, "Integrated Design Validation: Combining Simulation and Formal Verification for Digital Integrated Circuits", *J. Systemics, Cybernetics and Informatics*, pp. 22–30, vol. 4, no. 2, 2006.
[5] W. Kunz, J. Marques-Silva, and S. Malik, "SAT and ATPG: Algorithms for Boolean Decision Problems", in: *Logic Synthesis and Verification* (Ed. S. Hassoun, T. Sasao, R.K. Brayton), Kluwer, pp. 309–341, 2002.
[6] Goldberg E., Novikov Y.: "BerkMin: A Fast and Robust SAT-Solver", in: *Design, Automation, and Test in Europe*, March 2002, pp. 142–149.
[7] The MiniSat Page / http://minisat.se/MiniSat.html.
[8] L. Cheremisinova, D. Novikov, "SAT-Based Approach to Verification of Logical Descriptions with Functional Indeterminacy", *Proc. 8th Intern. Workchop on Boolean problems*, Freiberg (Sachsen, Germany), Sept. 18–19, pp. 59–66, 2008.
[9] L. Cheremisinova, D. Novikov, "SAT-based Method of Verification Using Logarithmic Encoding", Intern. book series "*Information science and computing*", FOI ITHEA, Bulgaria, pp. 107–114, No 15, 2009.
[10] L.D. Cheremisinova, D. Ya. Novikov, "Formal Verification with Functional Indeterminacy on the Basis of Satisfiability Testing of the Conjunctive Normal Form", *Automatic Control and Computer Sciences*, Allerton Press, Inc., pp. 1–10, Vol. 44, No. 1, 2010.
[11] D.Ya. Novikov, L.D. Cheremisinova, "Verification of Functional Descriptions with Indeterminacy on the Base of Paraphase Presentation of Boolean Functions", *Informatika*, pp. 54–62, no. 3, 2010 (in Russian).
[12] L. Cheremisinova, D. Novikov, "SAT-Based Implicative Method of Implementation Checking for Incompletely Specified Boolean Functions", *Proc. 9th Intern. Workchop on Boolean problems*, Freiberg (Sachsen, Germany), Sept. 16–17, pp. 97–102, 2010.
[13] L.D. Cheremisinova, D.Ya. Novikov, "Analysis of the implementability of descriptions with functional indeterminacy based on the verification of conjunctive normal form satisfiability", *Automatic Control and Computer Sciences*, Allerton Press, Inc., pp. 206–217, vol. 45, no. 4, 2011.