MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

KHARKOV NATIONAL UNIVERSITY OF RADIOELECTRONICS

ISBN 966-659-088-3

Proceedings of East-West Design & Test Workshop (EWDTW'04)

Yalta, Alushta, Crimea, Ukraine, September 23 – 26, 2004

CONTENTS

VERIFICATION TESTS GENERATION FEATURES FOR MICROPROCESSOR- BASED STRUCTURES

GENNADIY KRIVULYA, ALEXANDR SHKIL, YEVGENIYA SYREVITCH, OLGA ANTIPENKO

Abstract. A model of a microprocessor - based device as a bichromatic multidigraph with vertexes of two types is offered. Test generation features for functional testing using the updated algorithm of path activation in a structural model are described. The range method of data representation of different format data is introduced. Algorithms for execution of direct implication and backtracing of different types of operations and their program realization are represented.

Keywords: design verification and validation, test generation, multibit implication, range method, HDL.

1. Introduction

All set of methods of the determined test generation for digital devices can be divided into two large groups: structural and functional. Originally structural methods were oriented to a gate level of model performance of digital devices. However growth of complexity and rise of a component integration have led to a fact that models of increased integration elements began to be applied as the primitive elements (PE) of devices [1,2]. To the advantages of such approach it is possible to refer simple construction of a model of the device and formalizing of test generation procedures, and to the lacks - large dimension of a device model; and difficulties on creation and maintaining of the library of PE models, which can contain hundreds components.

With the purpose of overcoming these lacks the functional approach to construction of the tests was developed and has received a wide circulation [3, 4]. It can be used for digital devices of any complexity, including microsystems with program and microprogram control, as it allows receiving high level models of such devices. However functional methods are badly formalized because different types of function boxes, such as control block, operational block, address block etc. are present in microsystems. It is not obviously possible to formalize the method, which would have a possibility to handle so heterogeneous types of devices on the basis of the uniform approach.

In the given work the method of tests generation which is further development of the functional approach is offered. On a design stage of the digital device its decomposition on so-called homogeneously tested segments is carried out. The authors consider a method of tests generation for one of types of segments, namely, for the operational device (OD).

2. Test construction for microprocessor structures

The operational devices (OD) are characterized by processing of multibit information words and a possibility of selection in them a controlling part with microprogram handle. Structurally OD consists of the registers and combinational circuits fulfilling a given set of microoperations. The operators of the hardware description languages (HDL), executions, circumscribing the process, of microoperations are selected from the offered model OD in quality PE. Device decomposition up to a level of selected primitives during of HDL description compilation is fulfilled according to the real structure of OD. Therefore, in such model the number of PE is limited to a set of HDL operators, and for test generation it is possible to use procedures of path activation in a model of a device, similar to the activation procedures of structural methods. Let's describe OD model in detail, outgoing from its external performance on HDL. Typical performance of a device model in HDL [5] contains as a minimum two types of language constructions: signals and operators. The signals are divided on entry, output and internal. The operators are subdivided into two main types: executable and controlling.

On the assumption of such performance OD on HDL, a model of a device as a bichromatic multidigraph with vertexes of two types V = {Va, Vb} is offered. Vertexes of the first type Va correspond to a set of signals, vertexes of the second type Vb – to a set of operators, and arcs Å – to informational controlling single and multibit links between operators and signals.

Each arc starts in a vertex of the first type, and is ended in a vertex of the second type or on the contrary. Vertexes of the first type are signals, and of the second - operators. For each signal a label defining its digit capacity, and for each operator of a model - its type defined by a set of HDL operators are put in correspondence.

Vertexes of the first type are separable on two classes: external informational inputs and outputs, and also internal signals and constant. The second class - control signals. Such way of vertexes decomposition of the first type allows dividing the initial graph of a model into subgraphs, which boundaries are the vertexes of the first class. It is possible to divide signals which form boundaries of the subgraph on inputs and outputs for the given subgraph. The propagation of information in each such subgraph is carried out for one clock cycle of synchronization and corresponds to one microcommand.

Vertexes of the second type are separable on two subtypes: executable operators of OD data processing; and control expression. Control signals are always connected to outputs of control expression which can accept only two values (0 or 1), and each of which corresponds to an executable operator.

Depending on vertexes present in subgraphs there are three types of the subgraph: informational I - graph (a part of OD for a data conversion); controlling C - graph (handling controlling of signals), address A - graph (creation of the RAM address).

Features of test generation for a this-like model consist in the following:

1. support of elementary tests propagation for checking some function from external inputs up to the checked up function is fulfilled.

2. propagation of responses on test effect from the function on external inputs is fulfilled.

3. path activation in the graph: propagation starts from an external input of the I-graph, is fulfilled from one memory variable to another through the checked up function f_i up to an external output of the I-graph. All functions on this site are fixed.

4. control signals for all microoperations are transmitted for justification in the C - graph. There they are advanced back up to inputs of the C - graph, deriving a field of the microcommand.

3. Problem statement

The given model is on the nature structurally functional and allows synthesizing tests on the basis of logical path activation in a model structure. General aspects of testing are those:

1. informational stream Đ (similarly to the symbol D in the Roth algorithm) are placed on one of inputs of the function;

2. values from a given alphabet providing propagation of the stream Đ through the function from an active input onto an output are installed on the remaining inputs.

At such approach there is a number of problems, which require solutions:

1. parallel processing of multibit paths of OD.

2. Compact model performance of arithmetic and predicate functions for their processing at support of propagation of elementary tests to the function and responses from it.

3. Usage of the same operands for functions of arithmetic and logical types.

4. Range method of data representation

Multidigit logic and arithmetic functions are different on the nature. The former operate with vectors, the later - with numbers. Thus, a binary N-bit vector $B = (b_1,...,b_N)$ $b_n \in \{0,1\}$ can be represented by a corresponding number, and a ternary one – by a set of numbers. Notice that the same operands can participate both in arithmetic, and in logic operations. Therefore, the idea of using unified representation of operands [6] has arisen.

We propose so-called method range representation. When using range representation it is necessary to store only two values. Operations on such bases allow narrowing areas of solution while searching for them.

Each variable is assigned by a set of its all-possible ranges (APR):

$$A = \{APR_1, APR_2 ..., APR_k\}, \qquad (1)$$

where k is the number of set elements.

Every APR is set by a pair (min, max), where min and max are integer-based values from $-\infty$ to $+\infty$.

Consider the simplest case where variables are not beforehand restricted to any ranges. At that it is necessary to calculate only one element from APR

In a case of unsigned vector-variable setting minimum and maximum goes by following formulas:

$$\min = 0\,,\ \max = 2^n - 1\,, \qquad (2)$$

where n – number of bits necessary for variable.

For further argumentations let's represent every variable by a k-dimension set of pairs. The upper pair element will contain minimum value of i-th APR, the lower one – maximum. So for a variable A:

$$A = \left\{ \begin{bmatrix} a_{\min} \\ a_{\max} \end{bmatrix}_1 \ldots \begin{bmatrix} a_{\min} \\ a_{\max} \end{bmatrix}_i \ldots \begin{bmatrix} a_{\min} \\ a_{\max} \end{bmatrix}_k \right\}. \qquad (3)$$

While executing implication procedures there is a possibility to get an empty set of solutions in one or several APRs of a variable. An empty APR is defined by a condition:

$$\begin{bmatrix} a_{\min} \\ a_{\max} \end{bmatrix}_i = \varnothing\,, \text{ if } a_{\min} > a_{\max}\,, \qquad (4)$$

where $a_{\min}$ – minimum, $a_{\max}$ – maximum values, $i \in (0, k)$.

Note, that because we consider the simplest case when variable value is not restricted, so *k*=1. So:

$$A = \begin{bmatrix} a_{\min} \\ a_{\max} \end{bmatrix}. \qquad (5)$$

Notice, that a constant doesn't have minimum and maximum, but only a fixed value. Hence, for constants min and max values coincide and represent a constant.

For constants, transformation of vector representation into integer-based is made using the following formula:

$$A = \sum_{0}^{n-1} 2^i * a_i\,, \qquad (6)$$

where i – a bit number, a_i – a bit value, n – a number of bits in the vector.

5. Partly defined vectors

Often during executing implication procedures some bits in the vector are set into certain value (either 0 or 1), or some constraints for the vector value are laid on. So, define a procedure of getting a set of ranges for partly defined vectors.

A variable $A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ can take values either 0 or 1.

From one point, a pair of values (0,1) makes up a set X. From the other point, let's assume that we should understand a vector bit, set into either 0 or 1, as defined; correspondingly as undefined we should call a bit which has a value "unknown". Let's set «unknown» as x.

We should outline once again that all operations are done with unsigned vectors.

An n-bit vector is given. Set every bit of the vector as a_i, where $i = \overline{0, n-1}$ is a number of a bit. So, an array of values:

$$A = [a_{n-1}\ a_{n-2} \ldots a_2\ a_1\ a_0]. \qquad (7)$$

At that a_0 – the least significant bit, a_{n-1} – the most one. Set a_d – the least defined bit (not equal to 'x'). So d is the number of the least significant bit:

$$d = \min_{i=\overline{0,n-1}} \left(i \,\middle|\, a_i \neq \text{'x'} \right) \qquad (8)$$

Set u as a number of undefined (equal to 'x') bit to the left of a_d, i.e. for $i > d$. Calculate u by the following formula:

$$u = \sum_{i=d+1}^{n-1} k_i\,, \text{ where } \begin{cases} k_i = 1 \text{ for } a_i = \text{'x'} \\ k_i = 0 \text{ for } a_i \neq \text{'x'} \end{cases} \qquad (9)$$

The number of ranges is:

$$p = 2^u\,. \qquad (10)$$

The length of each range is:

$$L = 2^d\,. \qquad (11)$$

Set an array W of undefined values to the left of a_d with the length of u.

$$W = [w_{u-1}\ w_{u-2} \ldots w_2\ w_1\ w_0], \qquad (12)$$

where $w_j = 2^i \,|\, a_i = \text{'x'}$, $i = \overline{(d+1), (n-1)}$, $j = \overline{0, (u-1)}$.

Set a binary u-bit vector V, that has zero in all bits:

$$V = [v_{u-1}\ v_{u-2} \ldots v_2\ v_1\ v_0] = [00\ldots00]. \qquad (13)$$

The main algorithm is given below:

Step1. To get a range define all undefined values of the origin vector A with zero values. We get a vector:

$$A' = [a'_{n-1}\ a'_{n-2} \ldots a'_2\ a'_1\ a'_0], \qquad (14)$$

where $\begin{cases} a'_i = 0 \text{ for } a_i = \text{'x'} \\ a'_i = a_i \text{ for } a_i \neq \text{'x'} \end{cases}$ $i = \overline{0, n-1}$.

The minimum value of the first range

$$a_{\min_c} = \sum_{i=0}^{u-1} (a'_i * 2_i). \quad (15)$$

The maximum value:

$$a_{\max_1} = a_{\min_1} + L - 1. \quad (16)$$

The first range is:

$$\begin{bmatrix} a_{\min_1} \\ a_{\max_1} \end{bmatrix}_1. \quad (17)$$

Step2. Set loop counter C equal 2.

Step3. Increase vector V value with 1: $V = V + 1$

The c-th range is defined as:

$$a_{\min_c} = \sum_{i=0}^{u-1} v_i * w_i + a_{\min_1},$$

$$a_{\max_c} = a_{\min_c} + L - 1. \quad (18)$$

Step4. Increase loop counter C value with 1: $C = C + 1$. If $C \le p$, then go to Step3. After exiting the algorithm we should get p ranges.

6. Arithmetic operations

For greater visualization of formulas below, we replace words *minimum (min)* and *maximum (max)* with digits 1 and 2 in definition of ranges accordingly.

Let us consider the procedure of direct implication for the operation of addition C=A+B. After performing the operands in the range representation, we get:

$$\begin{bmatrix} c1 \\ c2 \end{bmatrix} = \begin{bmatrix} a1 \\ a2 \end{bmatrix} + \begin{bmatrix} b1 \\ b2 \end{bmatrix}. \quad (19)$$

For backtracing of addition:

$$\begin{bmatrix} a1 \\ a2 \end{bmatrix} = \begin{bmatrix} c1 \\ c2 \end{bmatrix} - \begin{bmatrix} b1 \\ b2 \end{bmatrix},$$

$$\begin{bmatrix} b1 \\ b2 \end{bmatrix} = \begin{bmatrix} c1 \\ c2 \end{bmatrix} - \begin{bmatrix} a1 \\ a2 \end{bmatrix}. \quad (20)$$

For direct implication C=A+B we take into account, that the result, obtained at the given operation, cannot have larger digit capacity, than digit capacity of an output variable. From here, boundaries of a minimum can only be augmented, and of maximum – only be diminished. Transforming, we receive:

$$\begin{bmatrix} c1 \\ c2 \end{bmatrix} = \begin{bmatrix} \max(c1, a1 + b1) \\ \min(c2, a2 + b2) \end{bmatrix}. \quad (21)$$

For backtracing, a basic expression is A=C-B. When it is necessary to receive marginally possible value on a variable A, it is necessary to subtract the maximum value B from the minimum value C, in other words:

$$\min A = \min C - \max B. \quad (22)$$

For finding maximum A, it is necessary:

$$\max A = \max C - \min B. \quad (23)$$

It is known, that the variable can be presented by its minimum and maximum. Thus,

Restricting possible variants of values for the variable A by its digit capacity from above and below, we receive the formulas.

$$\begin{bmatrix} a1 \\ a2 \end{bmatrix} = \begin{bmatrix} \max(a1, c1 - b2) \\ \min(a2, c2 - b1) \end{bmatrix}. \quad (24)$$

For $B = C - A$ it is similar:

$$\begin{bmatrix} b1 \\ b2 \end{bmatrix} = \begin{bmatrix} \max(b1, c1 - a2) \\ \min(b2, c2 - a1) \end{bmatrix}. \quad (25)$$

Example 2. Given $A\begin{bmatrix} 5 \\ 5 \end{bmatrix}$, $B\begin{bmatrix} 3 \\ 3 \end{bmatrix}$, $C\begin{bmatrix} 0 \\ 15 \end{bmatrix}$. Execute direct implication procedure for the operation of addition.

Solution. By Eq. (21) we have $C\begin{bmatrix} 8 \\ 8 \end{bmatrix} = 8$.

For the operation of subtraction $C = A - B$ direct implication looks like as it is given below (similarly to obtaining of Eqs. (21) and (24)–(25)):

$$\begin{bmatrix} c1 \\ c2 \end{bmatrix} = \begin{bmatrix} \max(c1, a1 - b2) \\ \min(c2, a2 - b1) \end{bmatrix} \quad (26)$$

For backtracing of subtraction the basic equation is $A = C + B$, that is similar to direct implication for addition:

$$\begin{bmatrix} a1 \\ a2 \end{bmatrix} = \begin{bmatrix} \max(a1, c1 + b1) \\ \min(a2, c2 + b2) \end{bmatrix}. \quad (27)$$

The basic equations for backtracing of B looks like $B = A - C$, that coincides with direct implication for subtraction:

$$\begin{bmatrix} b1 \\ b2 \end{bmatrix} = \begin{bmatrix} \max(b1, a1 - c2) \\ \min(b2, a2 - c1) \end{bmatrix}. \quad (28)$$

Example 3. Given $A\begin{bmatrix}0\\7\end{bmatrix}$, $B\begin{bmatrix}0\\3\end{bmatrix}$, $C\begin{bmatrix}0\\15\end{bmatrix}$. Execute the direct implication procedure for the operation of subtraction.

Solution. By Eq. (26) we have $C\begin{bmatrix}0\\7\end{bmatrix}$.

The procedures of direct implication and backtracing for variables given by a set of ranges allow increasing accuracy of obtained area of solutions. Let's consider this case on an example of the operation of addition $C = A + B$:

Example 4. Given

$$A\left\{\begin{bmatrix}2\\3\end{bmatrix}_1,\begin{bmatrix}6\\7\end{bmatrix}_2\right\}, B\begin{bmatrix}3\\3\end{bmatrix}, C\begin{bmatrix}0\\15\end{bmatrix}.$$

Solution. For the first range from the set of ranges of the variable A, we receive

$$\begin{bmatrix}c1\\c2\end{bmatrix}_1 = \begin{bmatrix}\max(0,2+3)\\ \min(15,3+3)\end{bmatrix} = \begin{bmatrix}5\\6\end{bmatrix}.$$

For the second: $\begin{bmatrix}c1\\c2\end{bmatrix}_2 = \begin{bmatrix}\max(0,6+3)\\ \min(15,7+3)\end{bmatrix} = \begin{bmatrix}9\\10\end{bmatrix}$.

Thus, the result of direct implication is the set $C\left\{\begin{bmatrix}5\\6\end{bmatrix}_1,\begin{bmatrix}9\\10\end{bmatrix}_2\right\}$.

During execution of the implication procedures, there are situations of obtaining empty solutions, which allows narrowing the area of considered solutions.

Example 6. Given $A\begin{bmatrix}0\\15\end{bmatrix}, B\begin{bmatrix}9\\10\end{bmatrix}, C\left\{\begin{bmatrix}4\\7\end{bmatrix}_1,\begin{bmatrix}12\\15\end{bmatrix}_2\right\}$.

Execute backtracing of the operation of addition.

Solution. As a result of backtracing the range with the index 1 has appeared empty $A\left\{\begin{bmatrix}0\\-2\end{bmatrix}_1,\begin{bmatrix}2\\6\end{bmatrix}_2\right\}$.

Finally $A\begin{bmatrix}2\\6\end{bmatrix}$.

7. Logic operations

As the logic operations are fulfilled digit-by-digit, that, accordingly, they will be handled digit-by-digit. Thus it is necessary to vary the order of performance of operands as ranges. That is before procedures of direct implication and backtracing above input multibit operands the following operations are fulfilled: each multibit vector is represented as a collection of bits, and the bits do not depend from each other. Then in correspondence with a method of ranges for representation a single bit we shall receive a set of ranges with serial numbers. Further for processing the ranges with identical serial numbers are taken. The interesting task is processing logic operations of the Boolean algebra with the help of the mathematical formulas that would allow using unified range method. Let's consider operations with one-bit operands. Let's execute procedures of direct implication and backtracing C=A and B with one-bit operands. From the truth table of the AND gate it is known, that the output, equal to 0, is defined uniquely by logical 0 on any of its inputs, and in only in case of all 1-s on inputs the output possess the value 1. That is the output is defined by minimum value on any input (it fairly only for one-bit operations): $C = \min(A,B)$. (29)

Taking into account influence of output values and adding limitations from above and below, in the terms of range dependences we receive the following formula:

$$\begin{bmatrix}c1\\c2\end{bmatrix} = \begin{bmatrix}\max(c1,\min(a1,b1))\\ \min(c2,\min(a2,b2))\end{bmatrix}. \tag{30}$$

For backtracing

$$\begin{bmatrix}a1\\a2\end{bmatrix} = \begin{bmatrix}\max(a1,c1)\\ \min(a2,\max(c1,1-b2))\end{bmatrix},$$

$$\begin{bmatrix}b1\\b2\end{bmatrix} = \begin{bmatrix}\max(b1,c1)\\ \min(b2,\max(c1,1-a2))\end{bmatrix}. \tag{31}$$

Example 6. Given $A\begin{bmatrix}0\\0\end{bmatrix}$, $B\begin{bmatrix}0\\1\end{bmatrix}$, $C\begin{bmatrix}0\\1\end{bmatrix}$. Execute the procedure of direct implication for the equation C = A and B.

Solution. We receive

$$C = \begin{bmatrix}c1\\c2\end{bmatrix} = \begin{bmatrix}\max(c1,\min(a1,b1))\\ \min(c2,\min(a2,b2))\end{bmatrix} =$$
$$= \begin{bmatrix}\max(0,\min(0,0))\\ \min(1,\min(0,1))\end{bmatrix} = \begin{bmatrix}0\\0\end{bmatrix} = 0$$

Example 7. Given $A\begin{bmatrix}1\\1\end{bmatrix}, B\begin{bmatrix}1\\1\end{bmatrix}, C\begin{bmatrix}0\\0\end{bmatrix}$. Let's execute the procedure of direct implication for the equation C= A and B.

Solution. We receive

$$C = \begin{bmatrix}c1\\c2\end{bmatrix} = \begin{bmatrix}\max(c1,\min(a1,b1))\\ \min(c2,\min(a2,b2)\end{bmatrix} =$$
$$= \begin{bmatrix}\max(0,(\min(1,1))\\ \min(0,\min(1,1))\end{bmatrix} = \begin{bmatrix}1\\0\end{bmatrix} = \varnothing$$

In this case “empty” means, that the output can not be equal to the earlier preset value at that given input values. And in fact, C=A and B=1 and 1=1, instead of 0, as was given earlier.

Before executing backtracing with logic operations let’s set a rule, on which we should start processing an input given x (that is not given) first. Subsequent inputs are computed on the basis of before processed ones.

Example 8. Given A$\begin{bmatrix}0\\0\end{bmatrix}$, B$\begin{bmatrix}0\\1\end{bmatrix}$, C$\begin{bmatrix}1\\1\end{bmatrix}$. Let’s execute the procedure of backtracing for C = A and B.

Solution. The input B is not given, it means, that it is processed first. We receive B$\begin{bmatrix}1\\1\end{bmatrix}$ that is on the input B - logical 1. Now we process the input A. We receive A$\begin{bmatrix}1\\0\end{bmatrix}$, that is equivalently to “empty”. In this case “empty” displays, that, having 0 on one of inputs, it is impossible to receive 1 on the output. Thus 0 on the input A, obtained on earlier steps or set forcedly, is not right, if we have the given output. At the same time if we execute the procedure of direct implication, where the inputs are determinative, we will receive “empty” on the output. It coincides with the truth table, by which the output C is equal to 0, if even one of inputs is equal to 0.

Similarly we are reasoning for C = A or B.

Taking into account the influence of output values, in the terms of range dependences we receive:

$$\begin{bmatrix}c1\\c2\end{bmatrix}=\begin{bmatrix}\max\,(c1,\max\,(a1,b1))\\ \min\,(c2,\max\,(a2,b2)\end{bmatrix}. \qquad (32)$$

For backtracing the formulas look like:

$$\begin{bmatrix}a1\\a2\end{bmatrix}=\begin{bmatrix}\max\,(a1,\min\,(c2,1-b1))\\ \min\,(a2,c2)\end{bmatrix},$$

$$\begin{bmatrix}b1\\b2\end{bmatrix}=\begin{bmatrix}\max\,(b1,\min(c2,1-a1))\\ \min\,(b2,c2)\end{bmatrix}. \qquad (33)$$

Example 9. Given A$\begin{bmatrix}0\\0\end{bmatrix}$, B$\begin{bmatrix}1\\1\end{bmatrix}$, C$\begin{bmatrix}0\\1\end{bmatrix}$. Let’s execute the procedure of direct implication for C=A or B.

Solution. We receive

$$C=\begin{bmatrix}c1\\c2\end{bmatrix}=\begin{bmatrix}\max\,(c1,\max(a1,b1))\\ \min\,(c2,\max(a2,b2))\end{bmatrix}=$$
$$=\begin{bmatrix}\max\,(0,\max(0,1))\\ \min\,(1,\max(0,1))\end{bmatrix}=\begin{bmatrix}1\\1\end{bmatrix}=1$$

Let’s consider one-operand operation C=not A:

$$\begin{bmatrix}c1\\c2\end{bmatrix}=\begin{bmatrix}\max\,(c1,1-a2)\\ \min\,(c2,1-a1)\end{bmatrix} \qquad (34)$$

For backtracing:

$$\begin{bmatrix}a1\\a2\end{bmatrix}=\begin{bmatrix}\max\,(a1,1-c2)\\ \min\,(a2,1-c1)\end{bmatrix} \qquad (35)$$

Example 10. Given A$\begin{bmatrix}1\\1\end{bmatrix}$, C$\begin{bmatrix}0\\1\end{bmatrix}$. Execute direct implication for the equation C = not A.

Solution. We receive

$$C=\begin{bmatrix}c1\\c2\end{bmatrix}=\begin{bmatrix}1-a2\\1-a1\end{bmatrix}=\begin{bmatrix}1-1\\1-1\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}=0\,.$$

Let’s consider the same operations with multibit operands. As logic operations are executed bit-by-bit, so, according to that, they will be processed in the same way. Thus the algorithm of performing operands in the range representation will vary. Before procedures of direct and backtracing with input multibit operands the following operations are executed: each multibit vector is represented by a set of bits, which form it, besides the bits do not depend from each other. Then in correspondence with the range method of single bits representation we receive a set of ranged with current numbers. For further processing the ranges with identical current numbers are taken.

Example 11. Execute the direct implication with vectors A=01xx0 and B=xx000 at the operation AND.

Solution. Let’s represent of a vector by ranges

$$A=\left\{\begin{bmatrix}0\\0\end{bmatrix}_1,\begin{bmatrix}1\\1\end{bmatrix}_2,\begin{bmatrix}0\\1\end{bmatrix}_3,\begin{bmatrix}0\\1\end{bmatrix}_4\begin{bmatrix}0\\0\end{bmatrix}_5\right\},$$

$$B=\left\{\begin{bmatrix}0\\1\end{bmatrix}_1,\begin{bmatrix}0\\1\end{bmatrix}_2,\begin{bmatrix}0\\0\end{bmatrix}_3,\begin{bmatrix}0\\0\end{bmatrix}_4\begin{bmatrix}0\\0\end{bmatrix}_5\right\},$$

since the output is not given, we accept $C=\left\{\begin{bmatrix}0\\1\end{bmatrix}_1,\begin{bmatrix}0\\1\end{bmatrix}_2,\begin{bmatrix}0\\1\end{bmatrix}_3,\begin{bmatrix}0\\1\end{bmatrix}_4\begin{bmatrix}0\\1\end{bmatrix}_5\right\}$, C=xxxxx.

Further one after another we select ranges with identical current numbers. Let’s begin with number 1. That is $A\begin{bmatrix}0\\0\end{bmatrix}_1, B\begin{bmatrix}0\\1\end{bmatrix}_1, C\begin{bmatrix}0\\1\end{bmatrix}_1$. We use the formulas for the direct implication. So we receive $C\begin{bmatrix}0\\0\end{bmatrix}_1$. Then we select the following range etc., shaping resulting set of ranges. After all we receive

$$C\left\{\begin{bmatrix}0\\0\end{bmatrix}_1,\begin{bmatrix}0\\1\end{bmatrix}_2,\begin{bmatrix}0\\0\end{bmatrix}_3,\begin{bmatrix}0\\0\end{bmatrix}_4\begin{bmatrix}0\\0\end{bmatrix}_5\right\}.$$

Since each range represents one bit, the transition back to vector representation will be simple and looks like C=0x000.

8. Program implementation

Program implementation of the offered method of data processing includes implementation of the range method and implication of two-place arithmetic operations, comparison operations and logic operations on the language C ++ as Windows API application.

The data are set in two formats: in vector (for cases completely undefined and partially defined unsigned binary vectors) and integer. In the program the possibility of reviewing resulting ranges on all operands, direction of implication and its outcomes is foreseen. The outcomes of the program are used in further for test generation by a method of paths activation in model performance of an operational device.

The execution time for arithmetic operations depends on a number of APRs for each vector. For example, in a case, when all vectors have one range of values, 10 000 operations of matching are fulfilled for 1 second, i.e. execution time of one operation - 100 mcs. If number of APRs for one of vectors is 4, execution time of one operation - 200 mcs. For a vector with 8 APRs execution time of one operation is 1200 mcs. For operations of matching there is a similar dependence of execution time on an amount of APRs. If a vector has only one possible range, the operation is fulfilled for 200 mcs, if 4 - for 500 mcs.

For logic operations time of implementation depends on digit capacity of operands and whether hipping of operand high bits of is required. For example, if all operands are single-digit, then execution time makes 300 mcs, for 5bit operands - 400 mcs.

9. Conclusion

The scientific value and novelty of offered outcomes of researches consists in development of the approach to decomposition of complex models on a number of more simple ones for simplifying the process of testing, namely stage of test generation. A model of a digital device at microprogram level is represented as a set of independent operational devices. Each operational device is a collection of the interacting graphs of three types. The model of a design error is a functional error of an HDL operator. Test generation for all functions testing of a device (with a given set of HDL operators) is carried out according to the updated algorithm of path activation in a structural model performance. Methods of performance and data processing also are offered at implementation of procedures of implication for different types of HDL operators. Their program implementation on the language C ++ is also provided.

The advantages and practical use of the given methods and program realizations are:

1. Getting rid of NP-complete task of finding decision at implication procedures.

2. Linear dependence instead of exponential one at time in program realization of implication procedures of different types including data conversion.

The results of program execution are represented in format of ranges. The outcomes form conditions of path activating in a C-graph.

References: **1.** *Breuer M.A., Fridman A.D.* Functional level primitives in test generation // IEEE Trans. Computers. 1980. N3, vol.C-29.P. 223-234. **2.** *Levendel Y.H., Menon P.R.* Test generation algorithms for computer hardware description languages // IEEE Trans. Computers. 1982. N7, vol.C-31.P. 577-588. **3.** *Thatte S.M., Abraham J.A.* Test generation for microprocessors // IEEE Trans. Computers.- 1980. N6, vol.C-29. P.429-441. **4.** *Shen L., Su S.* A functional testing method for microprocessors // Proc. 14th International Symp. on Fault-tolerant Computing, June 1984. **5.** *Êinosita Ê., Asada Ê., Karatsu Î.* Logic design of VLSI // Ì:Mir, 1988. 307 p. **6.** *Kovalyev Yev.V.* Designing of models of digital automata for test generation in the environment Active-HDL // *Thesis on competition of a scientific degree of the candidate of engineering science*, KhNURE, Ukraine, 2000. P. 128-135. **7.** *Moore R.E.*, Interval Analysis // Prentice Hall, Englewood Cliff s, New Jersey, 1966. **8.** *Rustinov V.A., Syrevitch Yev. Yef., Syrevitch A.V.* Interval method of representing multidigit operands to hold implication procedures during the synthesis of verification tests // *Management Information System and Devices*. 2003, vol. 122. P. 96-103.

ELECTRICAL TEST IS NOT ENOUGH FOR QUALITY

BENGT MAGNHAGEN

JONKOPING UNIVERSITY, SWEDEN

bengt.magnhagen@ing.hj.se

Electrical test means Functional Test (FT), In Circuit Test (ICT) or Boundary Scan Test (BST) or even a combination of these technologies. However, with modern technology, like SMD (Surface Mounted Devices) technology, BGA (Ball Grid Array) components and extremely small component dimensions, electrical test alone does not meet the quality requierments.

Electrical test can not identify bad soldering and bad alignment of components, as examples. Missing decoupling capacitors and so on can not be detected because of it is hard to get physical access for testprobes. Do not forget that digital designs contains a lot of analogue devices!

The tutorial will discuss today test technology with equipment for ICT and BST as well as its pros and cons. And as the addition of this, Inspection. Inspection has traditonally been performed manually but this is not realistic today with board crowded by components. Today Inspection is performed by machine vision. Optical technique named Automated Optical Inspection (AOI) and more advanced X-ray inspection (AXI). AOI and AXI is not the future, it is here today.

EMC /EMI is also a growing challenge and some new ideas will be discussed how to test for these phenomenas.

Підписано до дру*у 9.09.т004. Формат 60*84$^{1}/_{8}$.

Умов. дру*. ар*. ууи4. Облі*.-вид. ар*. у0и0. Зам. № т-875. Тираж №0 прим.

Віддру*овано в навчальнофнау*овому видавничофполіграфічному центрі ХНУРЕ.

6№66иХар*івипросп. ЛенінаиN4.