MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

KHARKOV NATIONAL UNIVERSITY OF RADIOELECTRONICS

ISBN 966-659-113-8

Proceedings of IEEE East-West Design & Test Workshop (EWDTW'06)

Sochi, Russia, September 15 – 19, 2006

CONTENTS

HES-MV – A Method for Hardware Embedded Simulation

Vladimir Hahanov, Anastasia Krasovskaya, Maryna Boichuk, Oleksandr Gorobets
Kharkiv National University of Radio Electronics, 14 Lenin Ave., 61166, Ukraine
E-mail: hahanov@kture.kharkov.ua

Abstract

Hardware implementation of triadic fault-free simulation method HES-MV – Hardware Embedded Simulation based on Multi-Valued alphabet is proposed. This method uses hardware gate and RTL models for large scale digital designs description. Structure solutions for logic elements models implementation are presented. Logic element has two bits for four values encoding for each input or output line of simulated device.

1. Introduction

Necessity for considerable increase of simulation performance for testing and verification purposes is well-known, it is defined by increasing complexity of digital system-on-chips with millions of gates. Existing simulation tools of leading companies: Cadence, Mentor Graphics, Synopsys, Aldec, spend several hours to analyze design with several millions of gates (PC with 500MHz microprocessor and 512MB RAM). Such costs are very important for end users. Aldec Inc. (www.aldec.com) proposes one of the possible solution: during system verification, separate design model on two parts (hardware H and software S): M={H,S}, H>>S. Moreover software model – a new one, unverified source code. Hardware part is tested IP –cores, implemented into HES (Hardware Embedded Simulator), based on Xilinx's FPGA, connected to the simulation kernel through PCI interface. Thus, Aldec proposed new design flow for world market, it gives possibility to reduce verification time in ten times. But hardware-based simulation excludes possibility for multi-valued simulation mode and transition analysis, hazard simulation, races analysis as well. Proposed approach, along with preserving hardware simulation advantages in performance, allows to simulate signal races and to solve set-up problem by extending hardware model with two-bit signals to identify four states of logic variable. Proposed bus-based primitive and logic elements hardware models may be important on world market of electronic design automation tools for design and test of large scale digital devices.

Object of inquiry – digital circuit, implemented into ASIC of PLD, specified using VHDL language.

Goal of the research – considerable decrease digital device design time (which will be implemented into integrated circuit, containing millions of gates) and extend functional capabilities of fault-free simulation system by multi-valued models hardware implementation, high-performance simulation method for set-up problem solving, race analysis and timing verification of tests under synthesis.

Research problems: 1. Digital circuit models classification. 2. Multi-valued analysis model for hazard detection and set-up problem solving. 3. Creation of software/hardware tools structure to multi-valued fault-free simulation. 4. Software/hardware implementation of multi-valued fault-free simulation method. 5. Testing and verification of hardware/software HES-MV tool.

2. Digital circuit models classification

Several papers laid in the base of current research, which are related to digital devices simulation speed-up based on hardware acceleration [1,2], multi-valued model and circuits for races analysis [3-15], mathematic methods for simulation speed-up [16,17] and synchronous event-based simulation algorithms for digital systems [5-8,18-19]. At fig.1 it is represented improved model classification for fault-free simulation and race analysis.

On the macro-level, we introduce seven qualification characteristics <Form, Structuring, Time, Iterativity, Alphabet, Implementation, Hierarchy>

$$M =< F = \{F^a, F^g, F^t\}, S = \{S^s, S^f\}, T = \{T^s, T^a\}, I =$$

$$= \{I^a, I^i\}, A = \{A^b, A^m\}, P = \{P^h, P^s\}, H = \{H^g, H^r, H^s\} >$$

Each characteristic includes several components (see fig.1).

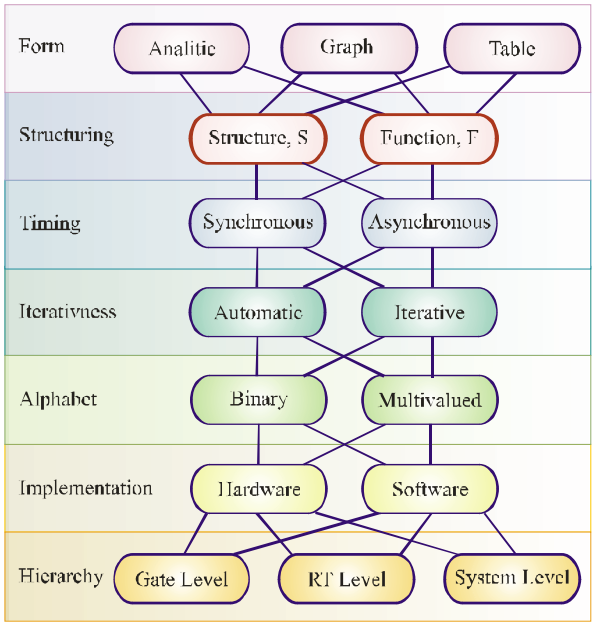

Figure 1. Digital system models

Main components are: *Structuring* specifies degree of details, which can be defined using the rest six classifiers. E.g. seven chosen components $M =< F^a, S^s, T^s, I^i, A^m, P^h, H^g >$ defines analytic, structural, synchronous, iterative, multi-valued, hardware, gate-level model. In general, each specific seven of characteristics defines possible ways to synthesize, analyze and model adequacy, defines maximum computational complexity, precision and variety of forward and backward simulation. This implyies urgency of proposed classification for electronic design automation tools developers. Totally, from presented classification (see fig.1), it exists minimum $|M| = |F| \times |S| \times |T| \times |I| \times |A| \times |P| \times |H| = 3 \times 2 \times 2 \times 2 \times 2 \times 2 \times 3 = 288$ different model implementations! But reality is more multiform, because some existing models are composed in form of vector of own characteristics subset, e.g.:

$$M =< F = \{F^a, F^t\}, S = \{S^s, S^f\}, T = \{T^s\},$$

$$I = \{I^a\}, A = \{A^m\}, P = \{P^h, P^s\}, H = \{H^r\} > .$$

Boundary assessment of number of all possible models is defined as product of Booleans of each characteristic:

$$|M| = (2^{|F|} - 1) \times (2^{|S|} - 1) \times (2^{|T|} - 1) \times (2^{|I|} - 1) \times$$

$$\times (2^{|A|} - 1) \times (2^{|P|} - 1) \times (2^{|H|} - 1) =$$

$$= 7 \times 3 \times 3 \times 3 \times 3 \times 3 \times 7 = 11907.$$

Choosing of concept (subset of parameters of each characteristics) for digital system model creation depends on solving tasks. In case of hardware-based fault-free simulation on gate level, it is necessary to consider model of minimal complexity and sufficient adequacy, which is defined as: analytical, structural, synchronous, iterative, multi-valued, hardware, gate-level:

$$M =< F^a, S^s, T^s, I^i, A^m, P^h, H^g > . \quad (1)$$

This concept, being sufficient, will not be maximum exact, because it has no exact timing delay characteristics, which are peculiar to asynchronous models. Concept of asynchronism increases computational costs for device analysis in ten times.

3. Race analysis models

One may analyze design on the subject of race conditions using methods of synchronous and asynchronous digital systems simulation. Differences of these methods consist in used scale of quantification of time continuum and time delays within components. Specialized model of primitive of device, shown in fig. 2: $\mu =< \varphi, \tau, \lambda, \alpha >$, considers synchronous-asynchronous nature of races and applies both to digital system and to separate components. Four parameters may be refined as follows:

$$\mu =< \varphi = [t_i, t_{i+1}]; \tau = [\tau_{min}, \tau_{max}];$$

$$\lambda = [\lambda_{min}, \lambda_{max}], \alpha = \{0, 1, X, \varnothing\} > . \quad (2)$$

Here we got two types of delays, related to nominal parameters of elements, and dispersion of primary input stimuli applying. Parameter φ defines period of synchronization – timing interval between two consecutive input test or work stimuli, this parameter must be no less than delay of whole circuit's delay. Parameter τ defines nominal element's or circuit's delay dispersion, which identifies procedure of primitive's simulation during asynchronous simulation. Parameter λ defines primary input switch time dispersion, its value identifies algorithms of synchronous, asynchronous and delta-triadic simulation. Parameter α defines signal's alphabet for transition process analysis, at the same time, alphabet must contain three or more symbols.

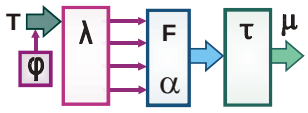

Figure 2. Timing model of element (device)

To simplify asynchronous circuit analysis, model time scale Δt is selected as greatest common divisor of nominal delays τ_i, due to this, all model delays r_i are

integer values. Model delays may be calculated using equation $r_i = \tau_i / \Delta t$. At the same time, all changes on the signals are happened only in time moments divisible by Δt: $0, \Delta t, 2\Delta t, \ldots$ of simulated time. Signal's value on the output $Y_i = g_i(t - \tau_i)$ of element g_i in time t is defined by inputs' states in time t-r_i.

In the following, we will discuss several the most popular race analysis methods, based on variations of parameters of μ.

1. Synchronous races analysis:

$$t_{i+1} - t_i = \text{const}; \quad \alpha = \{0,1,X\}; \\ \tau_{min} = \tau_{max} = 0; \quad \lambda_{min} = \lambda_{max} = \text{const}. \tag{3}$$

It is described as: fixed and in advance defined interval between switching consecutive input stimuli, defined as $\Delta t = t_{i+1} - t_i$; necessity of keeping of correlation for circuit's delay τ and element τ_i: $\Delta t >> \tau >> \tau_i$; fixed value of nominal elements' delay $\lambda_{min} = \lambda_{max} = \text{const}$, and used three-valued alphabet $\alpha = \{0,1,X\}$, where X – state of line, not identified neither as zero, nor as one, which oriented to exposure race conditions.

Synchronous analysis supposes, that time of race existence is considerably greater than circuit's nominal delay. The disadvantage of this method is false race condition identification. Synchronous simulation in $\{0,1\}$ alphabet can't determine race.

2. Asynchronous race analysis:

$$t_{i+1} - t_i \neq \text{const}; \quad \alpha = \{0,1\}; \\ \tau_{min} = \tau_{max} \neq 0; \quad \lambda_{min} = \lambda_{max} = 0 \tag{4}$$

It is described as: not fixed interval between switching consecutive input stimuli; nominal (or model) delays of components are not equal to zero; absence of input's switching time dispersion; usually binary simulation alphabet. It is necessary to know exact components' delays for method's implementation, but this is unavailable sometimes.

3. Δ – triadic race analysis:

$$t_{i+1} - t_i = \text{const}; \quad \tau_{min} = \tau_{max} \neq 0; \\ \lambda_{min} = \lambda_{max} = q; \; \alpha = \{0,1,X,\varnothing\}. \tag{5}$$

In comparison with asynchronous analysis, the parameters of Δ – triadic model are characterized by not fixed interval between switching of test sequences and input signals. The method eliminates disadvantages of binary asynchronous and triadic synchronous simulation. At the same time, in addition to nominal delay $\tau_i = r_i \Delta t$ of each primitive, input signal switching dispersion $q\Delta t$ is added, where q – natural number. We obtain triadic synchronous simulation method, given $q >> \tau$, where τ – maximum delay of the segment. We can increase or decrease adequacy of analysis by varying parameter q, at the same time by respective decreasing or increasing of processing time. But choosing parameter q – is a quite complex problem of expert analysis, related to consideration of wires length (board–chip).

4. Races analysis with growing ambiguity:

$$t_{i+1} - t_i = \text{const}; \; \tau_{min} \neq \tau_{max}; \\ \lambda_{min} = \lambda_{max} = 0; \alpha = \{0,1,X,\varnothing\}. \tag{6}$$

Method guarantees exact race identification, because of using nominal delays interval. Output value $Y_i = g_i(t - r_{max}, t - r_{min})$ in time t is defined by states of corresponding inputs in interval $[t - r_{max}, t - r_{min}]$. And vice versa, ambiguity exists in interval $[t + r_{min}, t + r_{max}]$ when input signal changes its value in time t. Simulation with growing ambiguity removes primitive's simulation delay determinism by including it into interval.

5. Δ-triadic race analysis with growing ambiguity combines advantages of the both methods. Active parameters:

$$t_{i+1} - t_i = \text{const}; \quad \tau_{min} \neq \tau_{max}; \\ \lambda_{min} \neq \lambda_{max}; \; \alpha = \{0,1,X,\varnothing\}. \tag{7}$$

Method gives possibility to obtain previous methods by changing inputs' switch dispersion $q = \{\Delta t, 2\Delta t, \ldots\} \in [r_{min} = \frac{\lambda_{min}}{\Delta t}, r_{max} = \frac{\lambda_{max}}{\Delta t}]$ and interval of ambiguous nominal delays: $0 \leq \frac{\tau_{min}}{\Delta t} \leq \frac{\tau_i}{\Delta t} \leq \frac{\tau_{max}}{\Delta t}$. But universality of such approach is related to complexity of its implementation and low performance. Because of this, weighty arguments must be provided to implement such model.

Presented general model of digital system (2), (7) is oriented to solving almost all tasks of timing verification on the all design stages. But software implementation of simulation algorithm based on (7) is related to considerable time consumptions, which can be a lot of hours. For example, XILINX library, based on VITAL standard [20], allows verifying digital designs after synthesis into ASIC (FPGA, CPLD) in interval from 3 to 8 hours, with chips complexity in 1 000 000 gates. One of the possible solutions to reduce simulation runtime is hardware implementation of models and methods of transition processes analysis and race conditions detection.

4. Hardware approach to race analysis

The main idea of hardware model for multi-valued transition process analysis is development of base elements or primitives, where input and output lines are represented as two-bits busses, able to code four logic states $\alpha = \{0,1,X,\varnothing\}$, which are necessary to identify transition states, different of 0 and 1. Symbol X is able to identify race conditions, hazards [5-9], which lead to unpredicted or unspecified state of digital system. Moreover, symbol X includes all transition processes of more complex alphabets [6, 10], which oriented to combinational circuits simulation. But extension of alphabet gives almost no advantages for sequential circuits' analysis [5], which are taking the most part of modern designs. Thus, it is enough to use triadic logic for solving following tasks:

1. Identifying hazards and race condition on the circuit's lines.

2. Verification of test sequences for the purpose of set-up into determined binary or defined state from undefined triadic one.

Further we consider primitives' models and their hardware implementation into FPGA (CPLD).

FPGA is oriented to truth table implementation, containing 4, 8, 9, 10, 11, 12 inputs. Correspondingly, truth table may contain from $2^4 = 16$ up to $2^{12} = 4096$ one-bit lines. Thus, LUT of FPGA, containing 16 bit and four address inputs, can implement basic logic elements (AND, OR, NOT) using four-valued logic:

And =

$\wedge$	0	1	X	$\varnothing$
0	0	0	0	$\varnothing$
1	0	1	X	$\varnothing$
X	0	X	X	$\varnothing$
$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

$\approx$

$\wedge$	01	10	11	00
01	01	01	01	00
10	01	10	10	00
11	01	10	11	00
00	00	00	00	00

Or =

$\vee$	0	1	X	$\varnothing$
0	0	1	X	$\varnothing$
1	1	1	1	$\varnothing$
X	X	1	X	$\varnothing$
$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

$\approx$

$\vee$	01	10	11	00
01	01	10	11	00
10	10	10	10	00
11	11	10	11	00
00	00	00	00	00

Not =

X_i	0	1	X	$\varnothing$
$\overline{X_i}$	0	1	X	$\varnothing$

$\approx$

X_i	01	10	11	00
$\overline{X_i}$	10	01	11	00

Generalized truth table for two-input primitives, where each input and output is represented by two-bit bus to encode four states, is as the following:

X_1	X_2	$X_1 \wedge X_2$	$X_1 \vee X_2$
00	00	00	00
00	01	00	00
00	10	00	00
00	11	00	00
01	00	00	00
01	01	01	01
01	10	01	10
01	11	01	11
10	00	00	00
10	01	01	10
10	10	10	10
10	11	11	10
11	00	00	00
11	01	01	11
11	10	11	10
11	11	11	11

According to VHDL description (AND, OR four-valued logic functions) [21], represented as listing 1, at fig. 3 post-synthesis implementation of two-input AND element is shown. It consists of 4 inputs, 2 outputs, 6 buffer elements and 2 functional logic blocks.

Listing 1. VHDL description of AND, OR triadic models.

```
-- Defines multi-value logic types and related functions
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_misc.all;
package delta_type is
        type bit2 is array(1 downto 0) of std_logic;
        function "and"( l, r : bit2) RETURN bit2;
        function "not"( l: bit2) RETURN bit2;
        function "or" ( l, r : bit2) RETURN bit2;
        function "xor" ( l, r : bit2) RETURN bit2;
end delta_type;
package body delta_type is
        function "and"( l, r : bit2) RETURN bit2 is
        variable a1, a2, c:std_logic_vector(1 downto 0);
        variable res: bit2;
        begin
a1(1):=l(1); a1(0):=l(0);
a2(1):=r(1); a2(0):=r(0);
c(1):= (a2(1)and a1(1)) or (not a2(0)and a1(1)and not a1(0))
or (a2(1)and not a2(0)and not a1(0));
c(0) := (not a2(1)and a2(0)) or (not a1(1)and a1(0)) or
(a2(0)and a1(0));
        res(1):=c(1); res(0):=c(0);
        return res;
        end function;
        function "not"( l: bit2) RETURN bit2 is
        variable a1, c:std_logic_vector(1 downto 0);
        variable res: bit2;
        begin
        a1(1):=l(1); a1(0):=l(0);
```

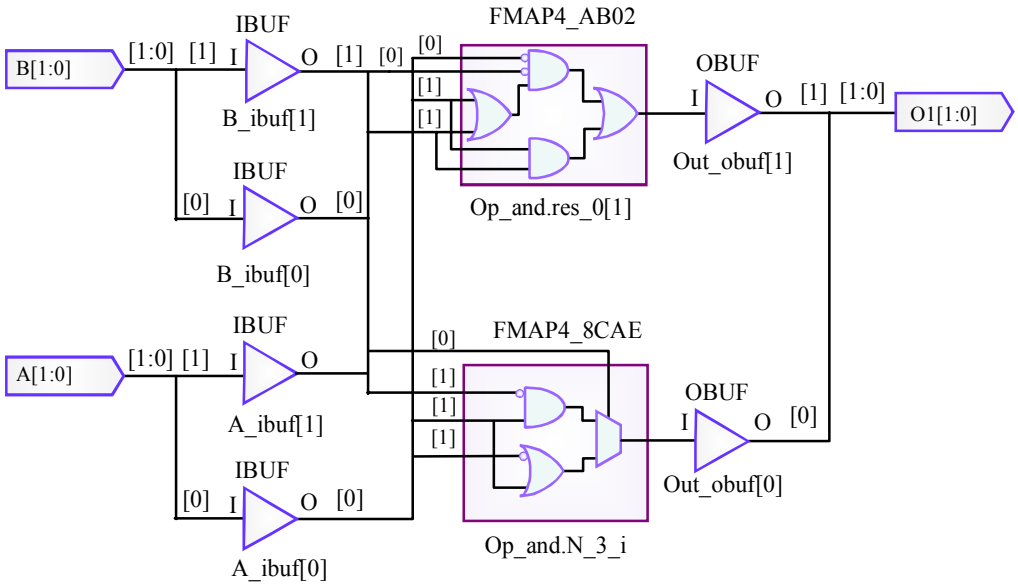

Figure 3. 2AND triadic hardware model

```
            c(1):= a1(1);
            c(0) := a1(1)xor a1(0);
            res(1):=c(1); res(0):=c(0);
            return res;
            end function;
function "or"( l, r : bit2) RETURN bit2 is
variable a1, a2, c: std_logic_vector(1 downto 0);
variable res: bit2;
            begin
a1(1):=l(1); a1(0):=l(0);
a2(1):=r(1); a2(0):=r(0);
c(1):= (a1(1)and a1(0)) or (a2(1)and a2(0)) or (a2(1) and
a1(1));
c(0) := (a2(1) and a2(0)) or (a1(0) and a1(1)) or (not a1(1) and
a1(0)) or (not a2(1) and a2(0));
res(1):=c(1); res(0):=c(0);
            return res;
            end function;
    end delta_type;
```

Implementation of C17 circuit based on multi-valued functional elements is shown in listing 2.

Listing 2. VHDL description of C17 model

```
use work. delta type.all;
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity c17 is
port (gat1, gat2, gat3, gat6, gat7: in bit2;
gat22, gat23: out bit2);
end c17;
architecture c17  of c17  is
signal gat10, gat11,gat16,gat19: bit2;
begin
gat10 <= not (gat1 and gat3);
gat11 <= not (gat3 and gat6);
gat16 <= not (gat2 and gat11);
gat19 <= not (gat11 and gat7);
gat22 <= not (gat10 and gat16);
gat23 <= not (gat16 and gat19);
end c17;
```

Post-synthesis implementation based on Xilinx Virtex2p has 24 LUT and 14 buffer elements. At the same time, primary input/output costs are in two times greater than in base variant (14/7=2); number of logic elements in use are in five times greater (46/8=5,75); Quine estimation shows hardware complexity: 115/14=8,2. One test vector's simulation time for software model is 8 microseconds. For hardware quaternary model this characteristics is 6,236 nanoseconds. Thus, hardware model's performance gain is $\eta = 8 \times 10^{-6} / 6,236 \times 10^{-9} = 1283$ times greater than its software equivalent. This fact allows concluding, that using of multi-valued hardware models for verification of designed digital devices for the purpose of hazards and race conditions identification, is appropriate.

Verification and testing of multi-valued models was performed on tens digital designs. Testing results are shown in tables 1 and 2. First one illustrates hardware complexity of traditional binary logic and quaternary models for transition processes analysis. Estimation of hardware costs increase is no greater than 10 times. Second table shows comparative performance analysis of binary software and multi-valued hardware models during one test vector simulation. Performance gain is greater than three orders of magnitude. Simulation results were obtained using Sigetest simulation and test generation system [22] and Synplicity Synplify Pro 8.1 [23], Intel Pentium workstation, 2.4GHz CPU with 512 MB RAM. Comparative characteristics for basic circuit's parameters are shown in fig. 4 and 5.

Table 1.

Circuits ISCAS'85	Lines number	Inputs number	Binary project	Multiple project	Hardware overhead
c17	11	5	2	22	11
c432	398	36	74	640	8,6
c499	599	41	120	1028	8,56
c1355	1015	41	157	1129	7,19
c1908	1307	33	121	1194	9,86
c3540	2007	50	380	3095	8,14
c6288	4579	32	637	5422	8,5
c20000	19998	72	2045	16973	8,3
c30000	29995	40	3099	28510	9,2
c50000	49996	22	5121	45064	8,8

Table 2.

Circuits ISCAS'85	Lines number	Inputs number	SW time, mks x 1000	HW time, mks x 1000	Simulation overhead
c17	11	5	0,008	0,0000062	1290
c432	398	36	0,15	0,0000707	2121
c499	599	41	0,2	0,0000672	2976
c1355	1015	41	0,86	0,0000693	12409
c1908	1307	33	6,12	0,0000532	115037
c3540	2007	50	9,28	0,0000710	86197
c6288	4579	32	9,01	0,0001687	53408
C20000	19998	72	1,67	0,0002034	8210
C30000	29995	40	4,17	0,0003681	11328
C50000	49996	22	17	0,0005722	29709

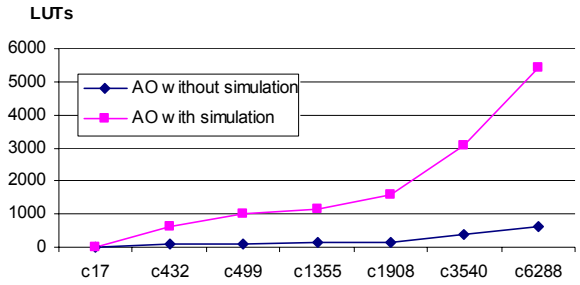

Figure 4. Models' hardware complexity

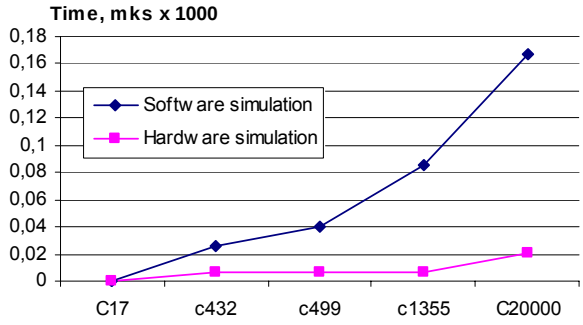

Figure 5. Models' simulation time

5. Conclusions

Scientific novelty of completed research consists in proposal of hardware model for digital devices and primitives, which allows extending functionality of hardware simulation method for transition processes' analysis and solving set-up task during test synthesis.

Practical value is defined by considerable (in 2-3 orders of magnitude) increase of simulation performance and extension of functional capabilities, which gives reasons to implement obtained results into design tools of leading world companies.

Disadvantage of hardware simulation method is 8-10-fold increase of hardware complexity of digital device model, which gives research topic on the decreasing and minimization of multi-valued models' complexity by using cubic coverage of logic and functional primitives [6]. To process minimized truth tables (cubic coverage), it is necessary to add circuit for triadic coverage analysis based on procedure of intersection's union.

6. References

[1] Active-HDL User's Guid. Second Edition. Copyright. Aldec Inc. 1999. 213p.

[2] Riviera 2006.02, DVM. HES. http://www.aldec.com/products/hes/pages/designverification manager/

[3] *Hahanov V.I., Babich A.V., Hyduke S.M.* Test Generation and Fault Simulation Methods on the Basis of Cubic Algebra for Digital Devices. Proceedings of the Euromicro Symposium on Digital Systems Design DSD2001. Warsaw. 2001. P. 228-235.

[4] *Abramovici M., Breuer M.A. and Friedman A.D.* Digital systems testing and testable design. Computer Science Press. 1998. 652 p.

[5] Avtomatizirovannoe proektirovanie ciffovyh ustroistv / S.S.Badulin, U.M.Barnaulov i dr./ Pod red. S.S. Badulina. M.: Radio i svyaz. 1981. 240 s. (In Russian).

[6] Hahanov V.I. Tehnicheskaya diagnostika elementov i uzlov personalnyh komputerov. K.: IZMN. 1997. 308 s. (In Russian).

[7] Baranov S.I., Mayorov S.A., Saharov Yu.P., Selyutin V.A. Avtomatizaciya proektirovaniya cifrovyh ustroistv. L.: Sudostroenie, 1979. 264 s. (In Russian).

[8] Avtomatizaciya diagnostirovaniya elektronnyh ustroistv / Yu.V.Malyshenko i dr. / Pod. red. V.P.Chipulisa. M.: Energoatomizdat, 1986. 216 s. (In Russian).

[9] Eichelberger E.B. Hazard Detection in Combinational and Sequentional Switching Circuits// IBM J. of Res. and Develop. 1965. V.9. No 2. P.30-99.

[10] Fantauzzi G. An Algebraic Model for the Analysis of Logical Circuits// IEEE Trans. on Comput. 1974. C23. No 6. P.576-581.

[11] Hayes J.P. Fault Modelling// IEEE Design and Test of Computers. 1985. V.2. No 2. P.88-95.

[12] Osnovy tehnicheskoi diagnostiki / Pod. red. P.P.Parhomenko. M.: Energiya, 1976. 460 s. (In Russian).

[13] Breuer M.A. A Note on Three-Valued Logic Simulation// IEEE Trans. on Comput. 1972. V.21. No 4. P.401-402.

[14] Shupak Yu.A. Analiz riska sboya v seti asinhronnyh avtomatov. Avtomatika i telemekhanika. 1991. No6. S.141-146 (In Russian).

[15] Kozlov V.N., Hatylev V.V. Principy povysheniya adekvatnosti logicheskogo modelirovaniya cifrovyh integralnyh MDP- i KMDP-shem // Elektronnoe modelirovanie. 1990. No1. S.60-64 (In Russian).

[16] Hahanov V.I. Dvuhtaknoe kubicheskoe ischislenie // ASU i pirbory avtomatiki. Izd-vo Hark. un-ta. 1996. Vyp. 100. S.78-89. (In Russian).

[17] Hahanov V.I., Zaychenko S.A., Yegorov A.A. Analiz bystrodeisrviya bazovyh operacij v deduktivnom metode modelirovaniya neistpravnostei // ASU i pribory avtomatiki. 2004. No 127. S. 138-148 (In Russian).

[18] *B Janick.* Writing testbenches: functional verification of HDL models. Boston. Kluwer Academic Publishers, 2001 354 p.

[19] Yarmolik V.N. Kontrol i diagnostika cifrovyh uzlov EVM. Minsk: Nauka i tehnika, 1998. 234 s. (In Russian).

[20] IEEE Std 1076 - 95. IEEE Standard for Vital ASIC. Modeling Specification. 1995. 81p.

[21] IEEE Std 1076-2002, IEEE Standard VHDL Language Reference Manual.– IEEE Inc., 2002. 124 p.

[22] Hahanov Vladimir, Hahanova Irina, Hyduke Stanley. Topological BDP Fault Simulation Method. Proceedings of the EUROMICRO Systems on Digital System Design. Rennes. 2004. P. 440-443.

[23] Synplify, Synplicity. http://www.synplicity.com/ products/synplifypremier/index.html.

AUTHORS INDEX