TEST TECHNOLOGY TECHNICAL COUNCIL (TTTC) OF THE IEEE COMPUTER SOCIETY
EDUCATION AND SCIENCE MINISTRY OF UKRAINE

KHARKOV NATIONAL UNIVERSITY
OF RADIOELECTRONICS

ISSN 1563-0064

RADIOELECTRONICS & INFORMATICS

Scientific and Technical Journal **Founded in 1997**

№ 4 (51), September – December 2010 **Published 4 times a year**

Sertificate of the State Registration КВ № 12097-968 ПР 14.12.2006

Address of journal edition: Ukraine, 61166, Kharkiv, Lenin avenu, 14, KNURE, Design Automation Department, room 321, ph. (0572) 70-21-326, d-r Hahanov V.I.
E-mail: ri@kture.kharkov.ua; hahanov@kture.kharkov.ua,
http://www.ewdtest.com/ri/

CONTENTS

Metastability Testing at FPGA Circuit Design using Propagation Time Characterization

Branka Medved Rogina, Peter Škoda, Karolj Skala, Ivan Michieli

Abstract—**This paper describes the measurement method and experimental technique with advanced instrumentation setup for analysing the metastability behavior and performance measurement of flip-flops used in programmable logic devices. In order to demonstrate this testing approach, the results for metastable characteristics parameters of one FPGA digital circuit fabricated commercially in 90 nm CMOS process are presented. The same test methods can also be used for evaluation of timing reliability in digital circuits as well.**

Index Terms—**FPGA, metastability, testing, propagation time**

I. Introduction

METASTABILITY is a central issue in the synchronization of two or more asynchronous signals. The standard method for accomplishing this task is to employ a D flip-flop (FF) as the synchronizing element. Metastability failure appears if data and clock input signals violate setup and hold times, as the output signal of the FF then becomes unpredictable [1]. The increased size of structural and functional complexity of today's digital circuits, such as FPGAs (Field Programmable Gate Array), has given rise to circuit designs with high numbers of asynchronous clock domains, where metastability problems occur most often, when the signal is transferred between circuitry in unrelated or completely asynchronous clock domains [2]. Concerning the reliability of digital circuit design, metastability is a problem that cannot be avoided It should be noted that manufacturers of digital circuits rarely give information about these parameters.

Different experimental procedures have been employed to accurately determine metastability characteristics, but the number of results cannot be compared because of different excitation and recording methods, so there is no standard test method. The results are very often presented through the MTBF value (Mean Time Between Failure) due to metastability, as an estimate of the average time between instances when signal transfers could cause metastability problems and design failure [3,4].

We have already presented a practical measurement technique to determine the MTBF characteristics of nonprogrammable and programmable logic devices using high accuracy time interval measurement setup [5,6,7]. Here we present a subsequent approach based on integrated propagation time characteristics, using a high speed digital storage oscilloscope for the acquisition, measurement and statistical timing of data analysis of metastable FF in FPGA devices. Although input (IOB) FFs are normally used to synchronize asynchronous input signals, the tests are performed both for FFs located in IOB (Input Logic Block) and CLB (Configurable Logic Block) areas .

II. Characterizing Metastability

A. Metastability equation

It is well known that the input to a synchronizing FF must be stable for a minimum time before the clock edge (setup time t_{SU}) and for a minimum time after the clock edge (hold time t_{HD}), in order to ensure reliable operation. The synchronizer output is then available after a specified clock-to-output propagation time (t_{CO}), as shown in Fig. 1.

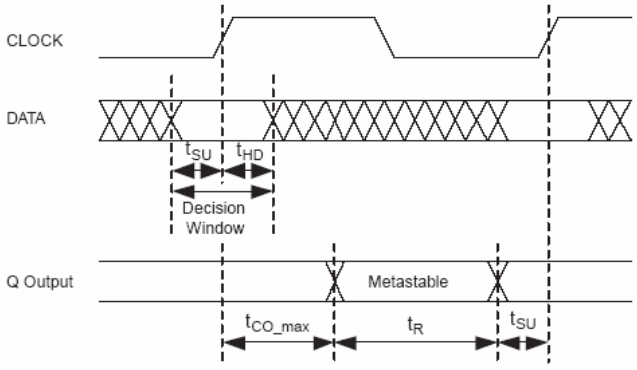


Fig. 1 A synchronizing FF must be stable for a minimum time before the clock edge (setup time t_{SU}) and for a minimum time after the clock edge (hold time t_{HD}), in order to ensure reliable operation

The problem with metastable events is not merely their occurrence, but when the event causes inconsistent values to

Manuscript received December 1, 2010.

Branka Medved Rogina (corresponding author) is with the Ruđer Bošković Institute, 10000 Zagreb, Croatia, Phone: +385998038703, Fax: +38514680090, E-mail: medved@irb.hr

Peter Škoda is with the Ruđer Bošković Institute, 10000 Zagreb, Croatia, Phone: +38514561023, Fax: +38514680090, E-mail: pskoda@irb.hr

Karolj Skala is with the Ruđer Bošković Institute, 10000 Zagreb, Croatia, Phone: +38598209816, Fax: +38514680212, E-mail: skala@irb.hr

Ivan Michieli, Ruđer Bošković Institute, 10000 Zagreb, Croatia, Phone: +38514561023, Fax: +38514680090, E-mail: michieli@irb.hr

Maja Vlah is with the KONČAR – Electrical Engineering Institute, 10000 Zagreb, Croatia, Fallerovo šetalište 22, Phone: + 38513667315, Fax: + 38513667317, E-mail: mvlah@koncar-institut.hr

Siniša Marijan is with the KONČAR – Electrical Engineering Institute, 10000 Zagreb, Croatia, Fallerovo šetalište 22, Phone: + 38513667315, Fax: + 38513667317, E-mail: smar@koncar-institut.hr

be latched into subsequent flip-flops. One manifestation of metastability behavior is excessive propagation time of this transition [8]. This increased delay will normally cause timing violations in the subsequent (synchronous) circuit if the output has not resolved itself by the time that it must be valid for use, for example, as an input to another stage. Because of the greater densities and more aggressive clocking strategies applied today, FPGAs have become more susceptible to these delay faults [9].

The metastability failure rate is usually calculated using the equation that relates mean time between two failures with operating conditions [10]:

$$\mathrm{MTBF} = \frac{10^{\frac{t_R}{\tau_d}}}{2 \cdot f_c \cdot f_d \cdot w} \qquad (1)$$

where:

- t_R is the resolve time or the maximum time a digital output can remain in a metastable state without causing a synchronization failure,
- w is the metastable window and represents the likelihood that a device enters into a metastable state,
- τ_d is the metastability resolution time constant and describes the speed at which the metastable condition is resolved,
- f_c is the frequency of the system clock,
- f_d is the frequency of the input signal.

Available resolve time is calculated in the following manner:

$$t_R = (1/f_c - t_{CO\_max} - t_{SU}) \qquad (2)$$

Therefore, the amount of resolve time t_R allowed for a device to settle plays a significant role in calculating its failure rate. In our practical measurement technique, we determine device depending characteristics, the metastability resolution time constant τ_d and the metastable window w, so that circuit's failure rate for different resolving time values at specific frequencies of clock and data signals can/may be calculated.

B. Metastability measurement method

Common methods for exploring MTBF use the stimulating data and clock signal out of phase, representing asynchronous signals that are most frequently used in real systems today [11]. However, the probability of an FF going into a metastable state will be higher if the input signal more often violates the setup time or hold time of the FF. Therefore, we use synchronous data and clock input signals to deliberately induce metastability in testing FF. However, it is necessary to consider the following. A MTBF parameter is used to calculate the mean time between synchronization failure events and is valid for random events, input events that are uniformly distributed over the clock period. Since we use a history-dependent method of generating input events, we must take care that it does not affect the data and still obtain the results that correspond to the excitation of random, asynchronous input signals.

As for the analysis of the results, we do not predetermine the time (after the clock signal) at which the output state of an FF is analyzed, as in the LTD (Late Transition Detection) method [11]. We simply measure clock-to-output propagation time at various setup, hold time values and store the data for later processing.

For example, Fig. 2 shows a propagation time characteristic measured for one general purpose FPGA circuit, Xilinx's 90 nm FPGA Spartan-3 within various setup and hold time relationships. It is evident that the clock-to-output propagation time starts to increase (black line) as the time interval between data and clock signals becomes less than 0.2 ns (setup time). At the same time, the number of output signals with increased propagation time but of regular voltage level (logical correct events) decreases (blue line). The largest measured increase in propagation time is 1.3 ns, with only 10 % of transitions occurred. The measurement results correspond to data listed in the manufacturer's specifications [12].

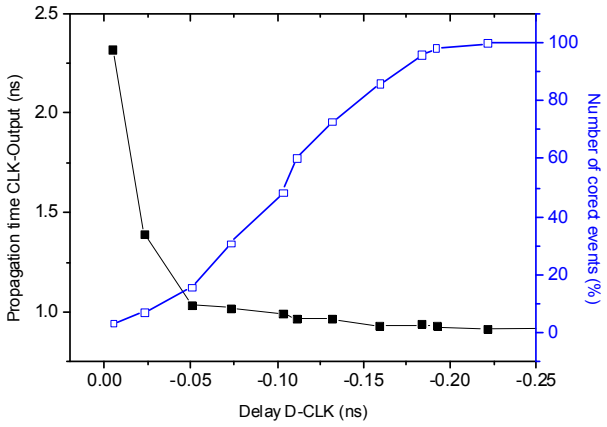


Fig. 2 A propagation time characteristic and number events measured for one general purpose FPGA circuit (Xilinx's 90 nm FPGA Spartan-3) within various setup and hold time relationships

To experimentally determine the metastability resolution time constant τ_d and the metastable window w, the edge of the data signal is set at the centre of the metastable window. This is usually adjusted, so the output signal of a FF favors both high and low logic data equally, corresponding to 50 % logically correct events [1]. (Using this Foley method type of excitation is simpler than the principle used in our original method [5].) The exact number of events is determined by DSO counting, or can be estimated upon the intensity of traces on the oscilloscope screen (as in Fig. 3).

Based on all measured results, it is possible to determine the decay function of this metastable state over time from the histogram of time propagation data. The metastability resolution time constant and the metastable window are calculated by applying the principle described by [13,14].

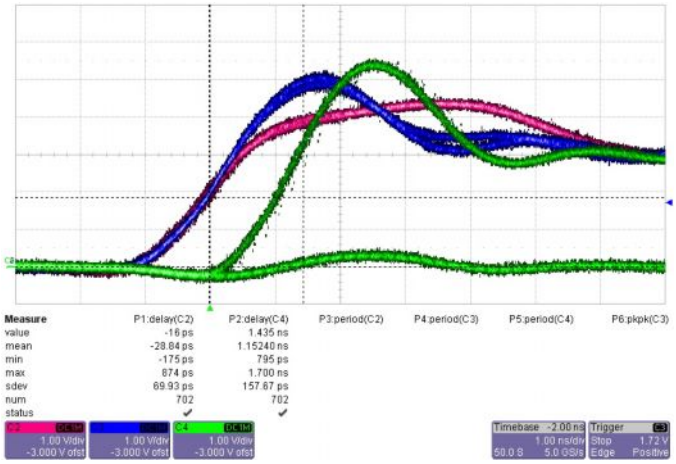


Fig. 3 Digital storage oscilloscope representation of data, clock and output signals waveforms and statistical data of propagation time, for IOB FF in Spartan-3 FPGA

III. Experimental Test Setup and Measurement Results

In our first measurement setup, time intervals are measured based on the start-stop principle, using a TAC (Time-to-Amplitude Converter). The time resolution of this analog method is very high (3.5 ps). Analog methods of time interval measurement give an excellent opportunity to analyze high-speed data signals with ps timing resolution [15]. For determining only the decay of the metastability state, such a sophisticated measurement setup might not always be necessary. Therefore, we propose a metastability test setup based on digital storage oscilloscope (DSO) for data acquisition. The DSO is capable of continuous measured data accumulation, storage and a statistical data mining analysis of the results [7].

The main parts of the measurement setup are shown in Fig. 4.

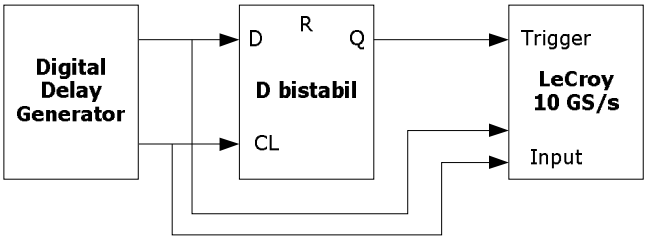


Fig. 4 Experimental metastability test setup based on digital storage oscilloscope for data acquisition

A digital delay generator (Stanford Research DG353) provides programmable clock and data fast pulse input signals for FF under testing. The trigger output signal of the generator is used as an external trigger input for a 10 GS/s digital storage oscilloscope (LeCroy WaveRunner 6100). The resolution of the signal generator (5 ps) is the only sizable limitation of the accuracy of the measurement method.

The logical connection of IOB and CLB test FFs in Spartan-3 FPGA is shown in Fig. 5. In each device, the same implementation tests the IOB and CLB FFs simultaneously.

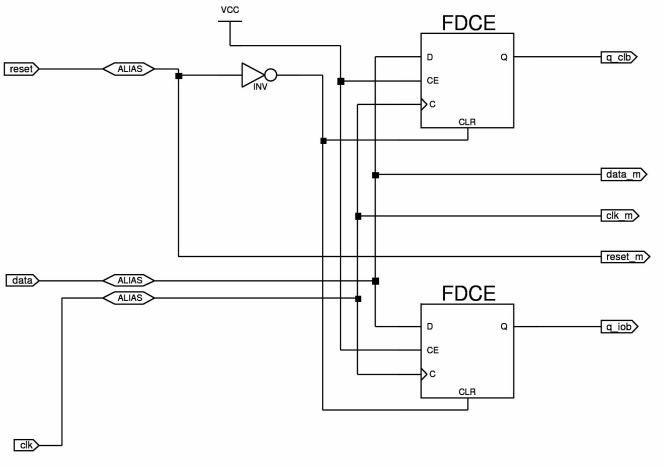


Fig. 5. Logical connection of IOB and CLB test FFs in Spartan-3 FPGA for metastability test

The time waveform diagram of reference signals for metastability test is shown in Fig. 6.

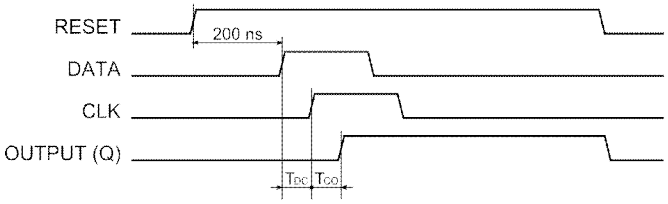


Fig. 6. The time waveform diagram of reference signals for metastability test

A. *Evaluating the metastability parameters*

In order to determine the statistical characteristics of logically correct events, the oscilloscope is triggered by the output signal of tested FF. A characteristic histogram of the measured propagation time of approximately 25 x 10^3 events is presented in Fig. 7. The horizontal time base represents the time from the moment of triggering the oscilloscope "back" to the edge of the clock signal; increasing settling time values are shown "from right to left."

In Fig. 8 the X-axis represents the time from triggering Q output back to the clock edge and, therefore, increasing metastability time is shown from left to right. The value of τ for the metastable region can be measured from the histogram. The slope of the metastable region starts at about -1.0 ns and ends at -1.15 ns. For this instance, τ is about 28 ps and is obtained from the reciprocal of the slope of the histogram. This particular device is quite fast and

metastable events cannot be observed as well as in slower devices.

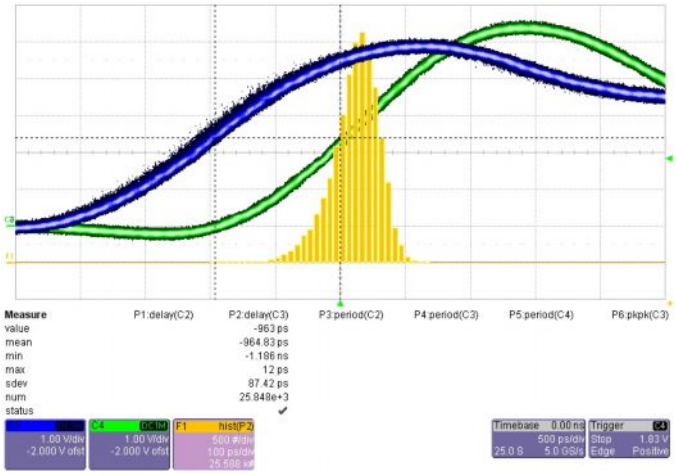


Fig. 7. Digital storage oscilloscope representation of propagation time histogram for IOB FF in Spartan-3 FPGA in metastability test

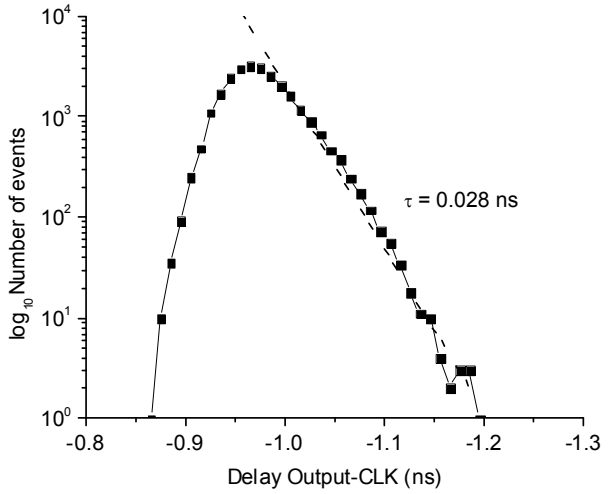


Fig. 8. Determination of resolution time constant τ for IOB FF in Spartan-3 FPGA in metastability test

The width of the metastable window w is determined for different propagation time values from the same histogram. It is first necessary to convert the vertical axis named by numbers of metastable events to a time scale representing the effective size of the metastable window for a given propagation time. In order to make this conversion, it is necessary to locate a point on the vertical axis which can be mapped to a propagation time parameter on the x-axis. Following the procedure in [14], events with normal propagation time (measured value 0.966 ns) are assigned to a metastable window that is equal to setup time (0.21 ns). The width of the metastable window (from t_{SU} to w) is reduced in the same proportion as the number of events (with larger propagation time). This is the basic idea that determines the parameters necessary to calculate the metastable window from the propagation time histogram. By applying this approach to the measured data histogram, metastable window values of 78.75, 7.88 and 0.65 ps are calculated for events with propagation time 50, 120 and 180 ps larger than the normal value. As expected, events with a large increase of propagation time are due to smaller metastable windows. For high speed technologies, like FPGAs, only events that occur as a result of small metastable window lead to "deep" metastability, while those for larger values constitute the area of "deterministic" metastability [13]. The results of these calculations for Spartan-3 FPGA IOB FF are shown in Fig. 9.

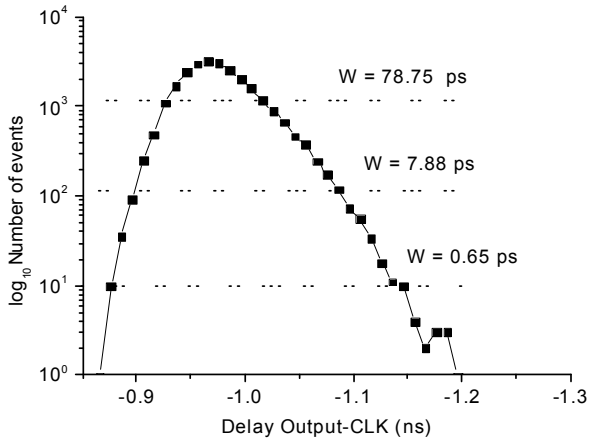


Fig. 9. Determnation of width of the metastable window T_w for IOB FF in Spartan-3 FPGA in metastability test

We present propagation time histogram results both for IOB and for CLB FFs in Spartan-3 FPGAs tested within three working regimes defined by the relation number of logical correct events and total number of events (100 %, 50 % and 10 %). The working regime is defined by setup time. The 100 % regime is derived from the 50 % regime by increasing its setup time by 2 ns, which puts the FF into an area of reliable/safe operation. The 10% regime is the area of deep metastability, with a small number of logically correct events, which are characterized by significantly increased propagation time. Approximately 5000 data points (propagation time) were collected for each working regime. The Spartan-3 FPGA showed little difference between IOB and CLB FF output behavior (Fig. 10. and Fig 11). the average value (mode) of propagation times is noted in the upper right corner of each histogram. The histograms clearly show an increase is propagation time and significant skewing of their distribution as FFs are driven into a metastable region of operation.

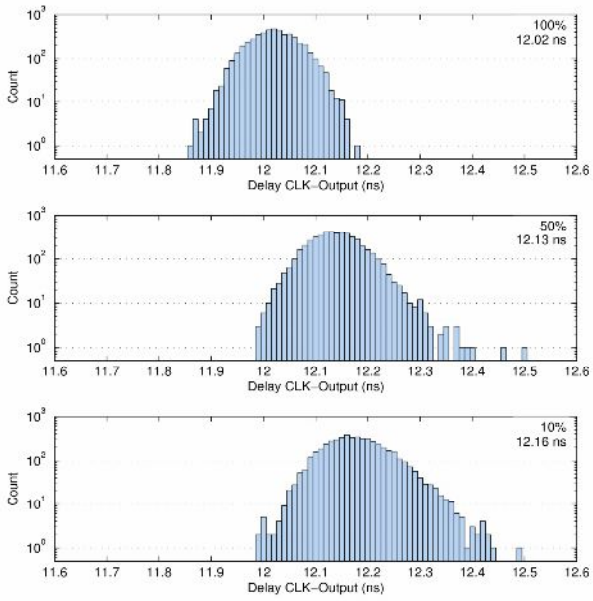


Fig. 10. Propagation time histogram for IOB FF in Spartan-3 FPGA at different stimulus (100%, 50%, 10%) in metastability test

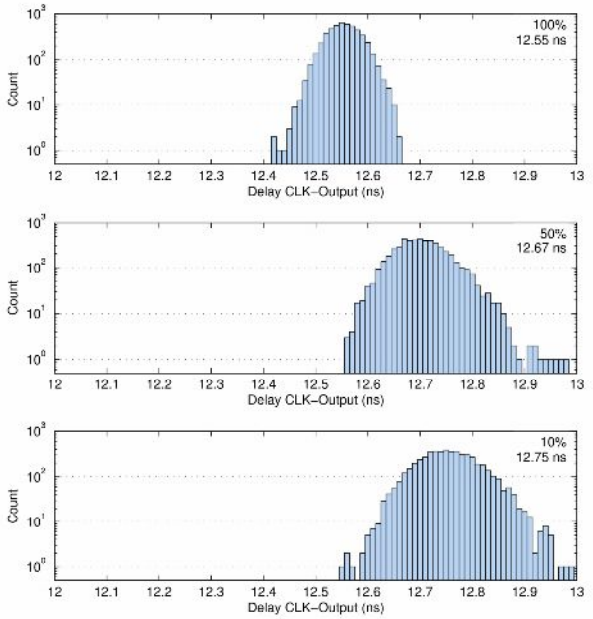


Fig. 11. Propagation time histogram for CLB FF in Spartan-3 FPGA at different stimulus (100%, 50%, 10%) in metastability test

IV. Discussion and Conclusion

In this paper, we have presented a modification of our experimental method for the measurement of metastability characteristics of synchronizing devices like FF using a digital storage oscilloscope for data measurement and analysis. By purposely changing parameters of input signals and by measuring the propagation time, one can monitor the entry of the FF into a metastable state, which is almost impossible in real-life (true) operation of digital circuits. The failures that we do detect provide us with a scaled estimate of the number of true failures. For the purpose of demonstration, the results are presented for IOB and CLB FF outputs fabricated in one process (90 nm) of FPGA devices.

Using the proposed experimental test setup based on a deliberately induced metastability method, we found the resolution time constant of Xilinx's 90 nm FPGA Spartan-3 to be less than 30 ps. In addition, there is no significant difference for the IOB and CLB FF outputs. The effective size of the metastable window depends on the propagation time of the metastable events, and changes by dropping from 0.1 ns to 1 ps.

Measurements on Spartan-3 using the LTD principle (to be announced) resulted in a 25 % higher value of τ. (Measurements were carried out with f_d = 10 MHz and f_c < 90 MHz signals and suggest an increase of MTBF with factor 2.4 x 10^6 for every additional 500 ps delay between the clock signal – At lower clock and asynchronous data frequencies MTBF increases inversely proportional to the product of these two frequencies).

Available data for FPGA devices fabricated in 90 nm processes yielded values of 20 to 50 ps (40 ps measured for Xilinx) and can be used for a comparison/verification with the results measured in our laboratory [16].

Acknowledgment

The authors wish to express their gratitude and satisfaction for being given free licenses from Mentor Graphics Inc. for the use of its advanced software package for rapid time analysis through the support of Academic Research Program.

References

[1] C. Foley, "Characterizing metastability", Proc. of the 2nd International Symposium on Advanced Research in Asynchronous Circuits and Systems, 1996, pp. 175-184.

[2] J. Stephenson, „Don't Let metastability cause problems in your FPGA-based design",(http://www.embedded.com/columns/technicalinsights/220300539?cid=RSSfeed_embedded_news).

[3] P. Afke, "Metastable recovery in Virtex-II Pro FPGAs", (http://china.xilinx.com/support/documentation/application_notes/xapp094.pdf).

[4] "Understanding metastability in FPGAs", (http://www.altera.com/literature/wp/wp-01082-quartus-ii-metastability.pdf).

[5] B. Medved Rogina, B. Vojnović, "Metastability evaluation method by propagation delay distribution measurement", Proc. of the Fourth Asian Test Symposium, 1995, pp. 40-44.

[6] B. Medved Rogina, K. Skala, B. Vojnović, "Metastability characteristics testing for programmable logic design", Lecture Notes in Computer Science 1142, Springer Verlag, 1996, pp. 381-388.

[7] B. Medved Rogina, I. Michieli, „Metastability paremeters evaluation by DSO aquisition", Proc. of the 12th International Workshop on Systems, Signals and Image Processing, 2005, pp. 397-40.

[8] A. Steininger, „Error containment in the presence of metastabiliy", Proc. Dagstuhl Seminar on Distributed Algorithms in VLSI Chips, 2009.

[9] C. Rockwood, "Keeping time with clock domain crossings in FPGAs", (http://chipdesignmag.com/display.php? articleId=903).

[10] R. Ginosar, „MTBF of a multi-synchronizer system on chip", (http://webee.technion.ac.il/~ran/papers/MTBFmultiSyncSoc.pdf).

[11] P. Kalisz, Z. Jachna, „Metastability tests of flip–flops in programmable digital circuits", Microelectronics Journal, (37), 2006, pp. 174-180.

[12] „Spartan-3 FPGA data sheets", (http://www.xilinx.com/support/documentation/spartan-3_data_sheets.htm).

[13] C. Dike, E. Burton, Miller and noise effects in a fynchronizing flip-flop, IEEE Journal of Solid.state Circuits, 34 (1999) 849-855.

[14] A. Cantoni, J. Walker, T.D. Tomlin, Characterization of a flip-flop metastability measurement method, IEEE Transactions on Cuircuits and Systems 54 (2007) 1032-1040.

[15] J. Kalisz, "Review of methods for time interval measurements with picosecond resolution", Metrologia (41), 2004, pp. 17-32.

[16] S. Beer, R. Ginosar, M. Priel, R. Dobkin and A. Kolodny, „The Devolution of Synchronizers", (http://webee.technion.ac.il/~ran/papers/The%20devolution%20of%20synchronizers%20ASYNC%202010.pdf.

Effects of Level Quantization and Threshold Clipping of the Signal and Basis Functions of Discrete Fourier Transform

Gamlet S. Khanyan, *Member, IEEE*

***Abstract*—The paper studies the influence of signal quantization levels number on accuracy of the results of spectral analysis. The overflow effect (signal threshold clipping due to shortage of the quantizing device bits) is also considered. A formula is derived for transforming a real number to its nearest quantization level. Numerical modeling of the quantized realizations of harmonic signal (pure one and mixed with noise) as well as its Fourier transform's basis functions is performed to construct characteristics – dependencies between program-assigned signal parameters and those measured in the course of digital processing under various quantization and clipping conditions.**

***Index Terms*—Analog-to-digital conversion, discrete and fast Fourier transform, level quantization, threshold clipping.**

I. Introduction

Level quantization is an inherent procedure of digital signal processing (DSP). Besides, discrete variability is the basis of most physical and biological world, and of various mathematical constructions (energy levels in quantum mechanics, DNA encoding, Boolean algebra, Walsh functions, etc.). Yet, wherever the reality idealization is based on the description of continually changing values, level quantization is considered as a distorting phenomenon. This is also true of spectral analysis based on discrete Fourier transform (DFT) method, where the transformed series of numbers, which are supposed to be continual in theory, are level quantized in practice.

Level quantization takes place not only during signal acquisition – when it undergoes analog-to-digital conversion (ADC), but also during its further processing by a computing device with finite number of memory cells' and processor registers' bits. In the latter case it is convenient to study, as an example, the level quantization of basis functions (sines and cosines) of DFT calculated in a direct (but slow) way, and by a rather sophisticated in mathematical description method of fast Fourier transform (FFT), having multiple versions of its implementation algorithms [1]. These versions (data decimation in time or frequency domain with their prior or post processing by the "butterfly" scheme, etc.) "shuffle" differently the interim results and are not commutative in their ultimate calculation with the level quantization operation.

Level quantization is often accompanied by clipping the signal on thresholds established, e.g., for protection against overload.

Questions about the effects of these and other ways of signal restriction on the accuracy of the Fourier transform are relevant to the metrology of spectral analysis as a means of measuring the amplitude-phase-frequency characteristics of oscillatory processes of different nature. Answers to those questions will allow manufacturers and customers of microelectronic devices to coordinate more precisely the actual performance of the industry's products with features envisaged during their design stage.

Level quantization of the signal has been subject of a large number of works – from the earliest to the present time of DSP development history [2]–[7]. However, effects related to it cannot be considered to have been fully studied. For instance, well-known manuals on theory and practice of signal processing [2]–[4] confine themselves to describing the quantization phenomenon and rationale of probability distributions of the rounding noise (error), in particular the assumption of its uniform distribution. Among the quoted (by no means complete) list of the literature the papers [5]–[7] can be highlighted where quantization effects are studied as applied to one of the vast areas of DSP – digital filtering.

As for another major DSP area – discrete spectral analysis, one can feel a noticeable lack of references here, and the present work is aimed, if not to fill this gap, but to demonstrate a possible research direction of the problem, which occupies a prominent place among other problems of measurements and signal processing.

I. Mathematical background

Let us consider an analog signal $s(t)$ describing a physical process, and its digital realization s_n of duration T s and length of N samples, obtained (starting at a time moment t_0) with sampling frequency F Hz. The signal is assumed to be clipped on constant thresholds A and B:

Manuscript received December 14, 2010.

G. S. Khanyan is with the Central Institute of Aviation Motors named after P. I. Baranov, 2, Aviamotornaya st., Moscow, 111116, Russian Federation (phone: +7 906 099 9958; fax: +7 495 552 4847; e-mail: dep007@rtc.ciam.ru, khanian@mail.ru).

$$\begin{cases} s_n = \begin{bmatrix} A, & s(t_n) \le A \\ s(t_n); & A < s(t_n) < B \\ B, & s(t_n) \ge B \end{bmatrix} \\ t_n = t_0 + n/F; \quad n = 0,1,...,N-1; \quad N = TF. \end{cases} \tag{1}$$

Before digital processing, such a signal undergoes level quantization, which can be represented as a chain of operators affecting s_n and transforming it into a quantized number

$$\bar{s}_n = \mathbf{P}^{-1}\mathbf{Q}\mathbf{P}s_n . \tag{2}$$

At first, the sample s_n, possessing a physical dimension of process $s(t)$, is affected by scaling (calibration) operator **P**, which transforms it into a dimensionless number contained in the range with fixed integer boundaries L_A=**P**A and L_B=**P**B corresponding to the thresholds A and B. The operation of this is linear

$$s'_n = \mathbf{P}s_n = Cs_n + D , \tag{3}$$

with the coefficient C and the displacement D, equal

$$C = (L_B - L_A)/(B - A); \quad D = (L_A B - L_B A)/(B - A) . \tag{4}$$

Then quantization operator **Q** itself comes into effect and rounds up the scaling result to the nearest integer number:

$$s''_n = \mathbf{Q}s'_n = [s'_n + 1/2] . \tag{5}$$

Finally, the obtained integer (5) is descaled, i.e., returned to the original physical dimension by applying the inverse operator **P**:

$$\bar{s}_n = \mathbf{P}^{-1}s''_n = (s''_n - D)/C . \tag{6}$$

Thus, we arrive at the function, depending on the single variable s_n and four parameters A, B, L_A, L_B:

$$\bar{s}_n = \left[\frac{(s_n - A)L_B - (s_n - B)L_A}{B - A} + \frac{1}{2} \right] \frac{B - A}{L_B - L_A} + \frac{L_B A - L_A B}{L_B - L_A} . \tag{7}$$

Real ADC devices contain an even number of quantization levels L=2^l, where l is the number of binary digits. The levels are numbered from L_A= –L/2 to L_B=L/2–1. Of interest is an odd number of levels L with the boundaries L_A= –(L–1)/2, L_B=(L–1)/2. Both cases of L are easy to combine:

$$L_A = -[L/2], \quad L_B = [(L-1)/2]; \quad L \ge 2 . \tag{8}$$

It turns out that formula (7) is invariant under the displacement by an integer constant L', i.e., it does not change when replacing L_A and L_B by L_A+L' and L_B+L'. In particular, we can set L_A=0, L_B=L–1 in (7) and simplify it:

$$\bar{s}_n = \left[\frac{(s_n - A)(L-1)}{B - A} + \frac{1}{2} \right] \frac{B - A}{L - 1} + A . \tag{9}$$

Considering now an important case of threshold symmetry B= –A>0 and normalizing the signal per that threshold (σ_n=s_n/B), we obtain a formula of quantization and clipping

$$\bar{\sigma}_n = \begin{cases} q(\sigma_n) = \dfrac{2}{L-1}\left[\dfrac{(L-1)\,\sigma_n + L}{2} \right] - 1, & |\,\sigma_n\,| < 1 \\ \operatorname{sgn}\sigma_n , & |\,\sigma_n\,| \ge 1, \end{cases} \tag{10}$$

which depends on a single variable σ_n and contains a single parameter – the number of quantization levels L.

Formula (10) looks most simply for the cases of two-level (L=2) and three-level (L=3) quantization

$$\bar{\sigma}_n = \begin{bmatrix} +1, & \sigma_n \ge 0 \\ -1, & \sigma_n < 0 \end{bmatrix}; \qquad \bar{\sigma}_n = \begin{bmatrix} +1, & \sigma_n \ge +1/2 \\ 0, & -1/2 \le \sigma_n < +1/2. \\ -1, & \sigma_n < -1/2 \end{bmatrix} \tag{11}$$

Note that the quantization function q is properly defined on clipping thresholds (equal to ±1): putting σ_n= ±1 in (10) we get $\bar{\sigma}_n$= ±1= q(±1). But for an even L the zero value of σ_n shifts:

$$\bar{\sigma}_n(0) = 1/(L-1), \quad L \bmod 2 = 0; \quad \bar{\sigma}_n(0) = 0, \quad L \bmod 2 = 1 . \tag{12}$$

An important property of the $q(\sigma)$ function lies in the fact that an integer constant K can be moved beyond the brackets:

$$q(\sigma + K) = q(\sigma - J) - Jq(0) + K, \quad J = (KL - K) \bmod 2 . \tag{13}$$

If K is even or L is odd, then J=0 and $q(\sigma)$ becomes periodical.

Difficulties of further research lie in the fact that it is impossible to obtain an analytical expression for the Fourier transform of any type of quantized signal, except for the trivial $s(t)$=*const*.

The exact formulas for the spectra of the amplitudes and phases are derived for time-limited pure harmonic signal under the assumption of continuity of its samples [8]. They are the basis of special methods of spectral analysis that increase accuracy of estimating harmonic oscillation's parameters, and are applied in the present paper. Here, despite the fact that the quantized signal can be described by means of piecewise constant function (10), to define the coordinates of those constants' boundaries on the horizontal axis seems to be not possible. Hence the impossibility of the DFT calculated sum fragmentation. Therefore, the only way to study the problem is numerical simulation. A shortcoming of this approach is the need to consider a lot of special cases, whose classification is unlikely to be carried out comprehensively.

II. Description of Numerical Experiments

Numerical experiments to determine the influence of level quantization and threshold clipping on the accuracy of spectral analysis were performed using the dynamic measurements digital signal processing program *quatrix.exe*® [9], in which the procedures described by formulas (1)–(10) were included.

As a model of physical process (1) the analog harmonic signal of frequency f_0, amplitude a_0, and initial phase φ_0 mixed with noise and observed in a window with initial time t_0= –T/2 was taken:

$$s(t) = a_0 \cos(2\pi f_0 t + \varphi_0) + r(t);\ -T/2 \le t < T/2\ . \quad (14)$$

Harmonic signal, by virtue of the superposition principle (the linearity of Fourier transform), serves as a structural component of various types of dynamic processes (vibration, pulsation, etc.). Therefore, the results obtained in a study of this model can be rightfully extended to the complex physical processes that have polyharmonic nature and occur in reality.

The noise in model (14) was formed by the software generator of uniformly distributed pseudorandom numbers, *r′=random*, so that the noise component's samples

$$r_n = r(t_n) = b_0(2r' - 1);\quad 0 < r' < 1 \qquad (15)$$

varied within limits $\pm b_0$, whereas the harmonic component was between $\pm a_0$. Hence, samples s_n were limited and normalized per threshold $B=1+b_0$ so that for $a_0>1$ signal clipping was imitated.

The objective of each experiment was to build characteristics, that is a plot of a harmonic signal's parameter under measurement depending on the threshold or the number of quantization levels of either the signal or the basis functions of DFT or FFT when that parameter is fixed as a given constant in the model (14). Effect of quantization or clipping was assessed by comparing the experimentally measured signal parameters – frequency *f,* amplitude *a*, and phase φ with the program-assigned values f_0, a_0, φ_0.

Signal parameters were estimated on the basis of processing the results of discrete Fourier transform

$$S_m = \frac{1}{N}\sum_{n=0}^{N-1} \bar{\sigma}_n\, e^{-i2\pi mn/N};\quad m = 0,1,\ldots,N-1 \qquad (16)$$

that calculates the spectral function S_m of the integer frequency variable (bin) *m.*

The first half of samples S_m (hermitic conjugate with the second one) is used (assuming N to be an even number) to establish both the spectra of amplitudes A_m and phases Φ_m

$$A_m = 2\,|\,S_m\,|,\ \ \Phi_m = \arg S_m;\ \ m = 0,1,\ldots,m_0,\ldots,N/2 \qquad (17)$$

of the signal's digital realization s_n on the frequencies $f_m=m/T$. Here the bin m_0 is singled out, which is the address of the maximum peak in the amplitude spectrum searched by sorting out and comparing components of the array of numbers A_m.

A. Frequency measurement

It is clear that m_0 is a rough estimate of the process (14) harmonic component's dimensionless frequency $f_0T=m_0+\mu_0$.

Fractional adjustment μ_0 is estimated by formula

$$\mu_0 = (A_{m_0+1} - A_{m_0-1})/(A_{m_0+1} + A_{m_0-1})\,, \qquad (18)$$

possessing, as was shown in [8], good accuracy for moderate noise level b_0 and for m_0 location sufficiently distanced from spectrum edges $m=0$ and $m=N/2$.

This formula, in which the nearest (left and right) neighbors of the amplitude spectrum maximum peak are presented, was used to measure frequencies with accuracy exceeding the spectral resolution.

B. Amplitude measurement

The main problem that arises when evaluating the amplitude is its lowering at non-integer value of the dimensionless frequency f_0T (maximum peak of amplitude spectrum turns out to be $a_0\sin(\pi\mu_0)/(\pi\mu_0)$ instead of a_0). This phenomenon accompanied by the appearance of false sidelobe components, is known in the literature as a leakage effect (see, e.g., [2], [4]). There is no leakage at $\mu_0=0$, and the leakage is maximum at $\mu_0=1/2$ (in the latter case, the peak amplitude is ~64% of the harmonic oscillation's amplitude's true value).

To smooth the leakage effect, focusing method proposed in [8] was applied. The essence of the method consists in summing the square of amplitude spectrum maximum peak with squares of the related sidelobe components, and taking the square root *a* of that sum for evaluation of the oscillation's amplitude a_0.

C. Phase measurement

To assess phase φ_0, alternating method [8] was applied, which eliminates phase distortion – its false shift in the $\pi\mu_0$ taking place in the traditional spectral analysis (provided in time window $0 \le t < T$ alleged "by default"). The essence of the alternating method (provided in time window $-T/2 \le t < T/2$) is to swap the first ($0 \le n < N/2$) and second ($N/2 \le n < N$) halves of the signal's digital realization s_n before performing DFT. And then the phase φ_0 is estimated with accuracy $\pi\mu_0/N$ as the value of phase Φ_m at the address m_0 of the amplitude spectrum peak.

The results of phase measurements were outputted in degrees (within ± 180°).

III. Signal Quantization and Clipping Research Results

Digital realizations of process (14) with duration of T=1 s and length of N samples were formed with sampling frequency F Hz, so that the spectral resolution $F/N=1/T$ was 1 Hz and the dimensionless frequency f_0T coincided with the dimensioned one f_0.

A. Appearance of the oscillograms and spectrograms

Prior to discussing the results, it is of interest to regard the external look of examined signals and their amplitude spectra.

Fig. 1*a* indicates the first half of the oscillogram drawn with points and the amplitude spectrum of pure harmonic signal (a_0=1 V, b_0=0) of a semi-whole dimensionless frequency 256.5 drawn with solid line. The realization length is N=2048 samples, and the number of quantization levels is L=4096, which is typical of 12-digit ADC-board often used in practice.

With this number of levels accepted to be "infinite", as is shown in further analysis, the samples of signal can be considered as continuum numbers and both direct and fast Fourier transform methods – as identical in terms of their precision.

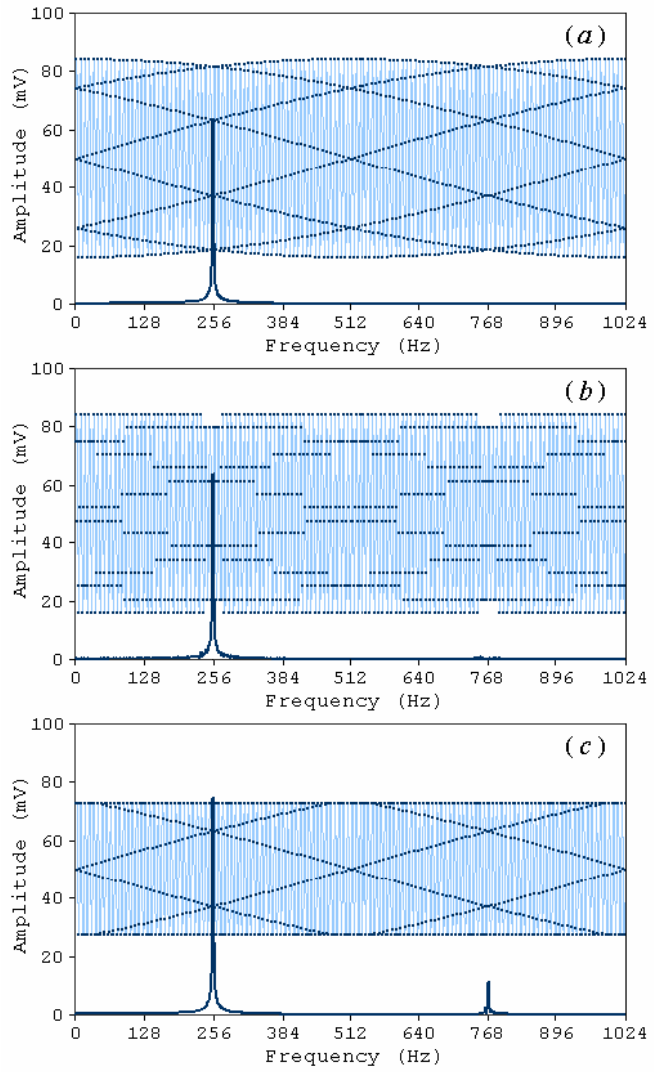


Fig. 1. Waveform and amplitude spectrum of a sinusoid under various conditions of quantization and clipping: *a* – large, *b* – small number of signal quantization levels without clipping; *c* – signal clipping with a large number of its quantization levels

The same signal quantized with a small number of levels L=16 is shown in Fig. 1*b*, and in Fig. 1*c* – with L=4096 levels, but clipped on amplitude (a_0=1.5).

A comparison of these illustrations shows how little the amplitude spectrum view is affected by the strong oscillogram view change. Spectrum distortion is more noticeable in case of the signal clipping, which is proved by the harmonic occurring at frequency 767.5 Hz.

Such a weak effect of level quantization on the amplitude spectrum (inadequate degree of waveform distortion) is certainly a positive fact, and at the same times an unexpected paradox.

B. Amplitude-frequency characteristics

Fig. 2 shows amplitude-frequency characteristics obtained for a short length N=32 of the signal's digital realization. Measurements were made at a constant phase φ_0=90°.

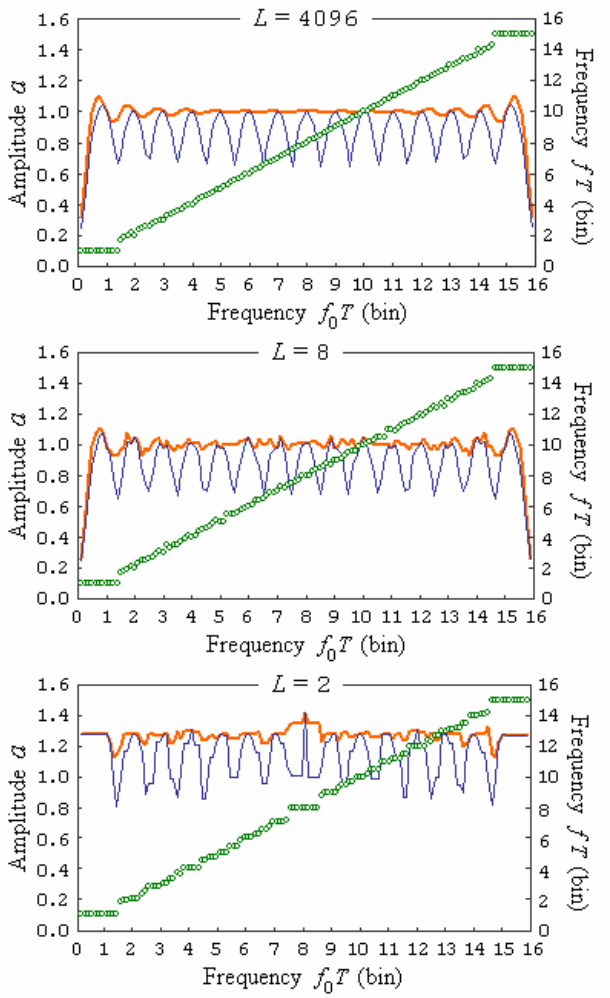


Fig. 2. Amplitude-frequency characteristics of pure harmonic signal for different number L of its quantization levels

Variable for each of the specified quantization levels L was the dimensionless signal frequency $f_0T=m_0+\mu_0$ incrementing by step 1/8 from spectrum origin to Nyquist frequency N/2. Amplitude was built depending on it: the orange line – using focusing method, the blue line – without adjustment (peak value of amplitude spectrum at the address m_0). Green circles show measured signal frequencies adjusted according to (18).

Amplitude measurement result close to the ideal one a=1 is ensured for $L\geq 8$ through focusing application. Amplitude oscillations calculated without adjustments are explained by leakage lowering it at semi-whole frequencies to ~64% of a_0.

The frequency measurement plot practically coincides with the program-assigned line, except in the spectrum central zone for two-level quantization and at the spectrum edges for any number of quantization levels. In the latter case, the so-called edge effect holds for the amplitude, being smoothed only if L=2.

An extremely small number of levels (L<8) raises the amplitude measurement result (by 20–30 %), practically not touching the nature of genuine spectral deficiencies (leakage and edge effects).

C. *Phase characteristics of quantization*

Fig. 3 shows phase characteristics of harmonic signal – results of phase measurement depending on the number of quantization levels L.

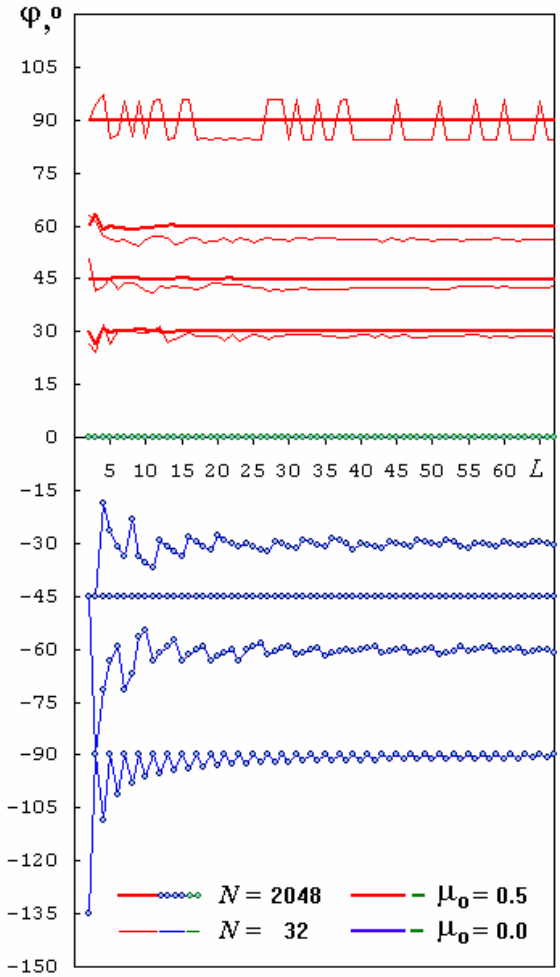


Fig. 3. Phase characteristics of pure harmonic signal with N=32 and 2048 realization length measured in the absence and with the utmost leakage effect

Characteristics were taken in the spectrum center (on bin $m_0=N/4$) with constant phase φ_0 taken from the row 0°, ±30°, ±45°, ±60°, ±90°. Using this sign symmetry to analyze the leakage effect for $\varphi_0 \geq 0$ $\mu_0=1/2$ was set, and for $\varphi_0 \leq 0$ – $\mu_0=0$. The signal's realization length was assigned to be large (N=2048, thick line and points) and small (N=32, thin line).

Results of phase measurement for long and short realizations are the same when there is no leakage (as it is evident for the curves in the lower part of the figure). It is surprising that distortion is either totally absent (for phases 0°, –45°), or for small L it is large and can have either "sawtooth" behavior (for phases –30°, –60°) or appear only when the number of quantization levels is even (for phase –90°).

If leakage is present (see the upper part of the figure), phase distortion depends on realization's length. It is minor for N=2048 and proportional to the phase itself for N=32.

In any case the phase measurement result can be considered to be approaching the asymptotic limit value for L>64.

D. *Noise influence*

To assess the effect of noise the harmonic signal was mixed with uniformly distributed noise of the same intensity ($b_0=a_0=1$), so that the ideal result of amplitude measurement was a=0.5. In the characteristics in Fig. 4 N=512, φ_0=0, f_0T=128.

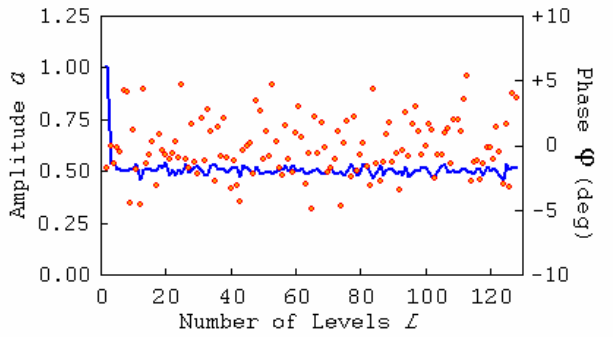


Fig. 4. Amplitude a (solid curve) and the phase φ (points) of harmonic signal mixed with noise depending both on the number of quantization levels L

A strong (twofold) overestimation of the amplitude (focused) is observed only for two-level quantization. For L>2 the noise-caused scattering of both phase (by several degrees) and amplitude (by some percentage) is not sensitive to quantization.

It can be argued that the noise is a distorting factor which is almost independent of the quantization (as well as of leakage, anyway).

E. *Signal clipping*

To research the signal threshold clipping effect, characteristics given in Fig. 5 were taken.

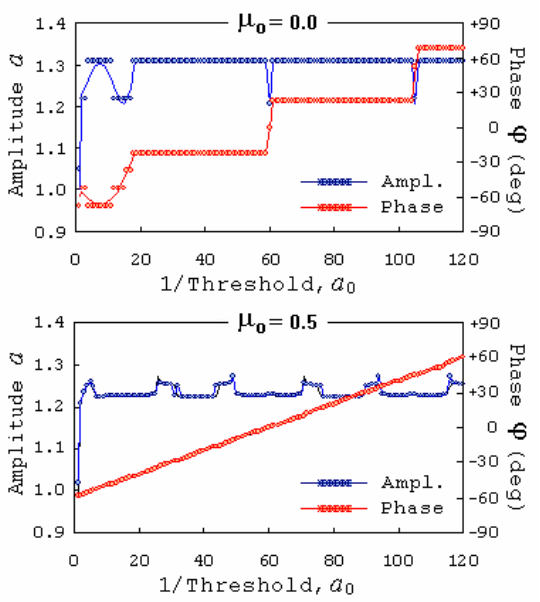


Fig. 5. Amplitude a and phase φ of a pure harmonic signal depending on the signal clipping threshold

The variable on X-axis is "overload factor" – parameter a_0 of model (14) showing how many times the signal amplitude is larger than unity threshold in (10). Y-axes feature amplitude

a and phase φ of the pure harmonic signal with length of N=512 samples and of frequency 128+μ_0, where μ_0 adopted values 0 and ½ corresponding to the minimum and maximum leakage effects. In each case a large (L=4096, thin line) and a small (L=4, points) number of quantization levels were taken. Phase φ_0 was set to vary linearly: from –60° to +60°.

The experimental results radically differ for different cases of μ_0. Paradoxically, for the maximum leakage phase measurements fit perfectly on program-assigned line, whereas in the absence of leakage, they are "ragged," piecewise constant, differing for small values of a_0 by cases of L (this difference is also clearly seen for amplitude measurements).

As for the amplitude, clipping leads to its overestimation of about 30 %. The clipped signal with unlimited growth of a_0 approximates the signal with 2–3 quantization levels.

IV. Fourier Transform's Basis Functions Quantization Research Results

We simulated and compared three situations related to the Fourier transform's basis functions level quantization.

In the first of them samples of signal (14) with amplitude a_0=1, equal to threshold B=1 (which ensures absence of signal clipping), were quantized. The second situation was created by quantization of the DFT (16) basis functions' samples during the cycles passing on m and n in the course of their direct calculation, and in the third situation the FFT sines and cosines table's entries during their preliminary calculation were quantized.

As trigonometric functions vary within limits ±1, quantization in the two latter cases, according to (10), was also carried out without threshold clipping.

A. *Appearance of the spectrograms*

In Fig. 6 signal is almost continual – it has L=4096 quantization levels, but both the samples of DFT and FFT basis functions have only L=8 levels.

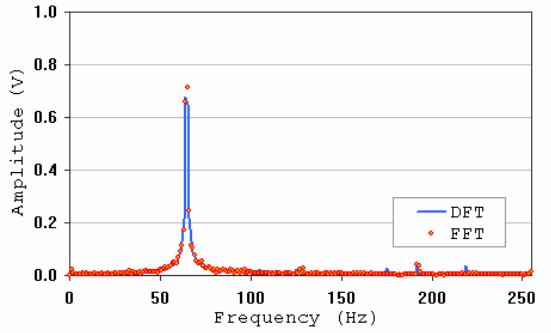


Fig. 6. Amplitude spectrum of a sinusoid with L=8 quantization levels of both DFT and FFT basis functions

We see a generally weak, but noticeable difference in the peak amplitude of the DFT and FFT spectra from each other, which already indicates the impact of quantization on the metrological properties of fast computational algorithms.

B. *Frequency characteristics of the quantization*

Fig. 7 shows the frequency characteristics plotted in the absence of leakage and when it has the maximum effect.

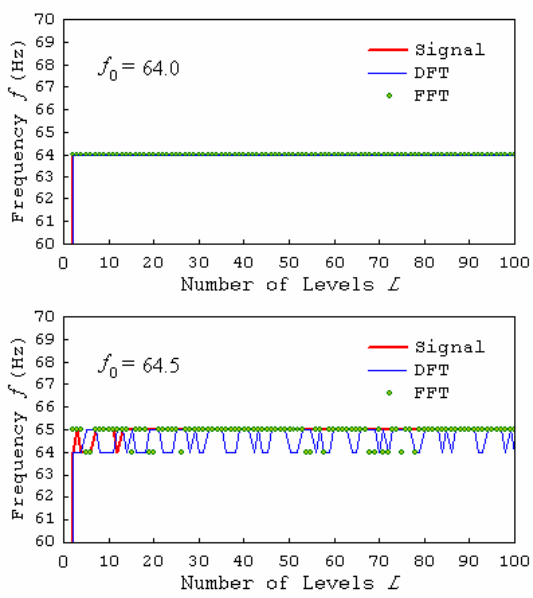


Fig. 7. The results of measuring an integer and a half-integer frequency of sine wave depending on the number of quantization levels of signal and basis functions of DFT and FFT

In the first case (f_0T=64) the ideal outcome measure $f=f_0$ is obtained at the two phase values (φ_0=0°, φ_0=45°) beginning from number of quantization levels L=3 (both for signal and for basis functions of the Fourier transform).

In the second case (f_0T=64.5, φ_0=0°) the value of fT varies between the nearest bins 64 and 65 giving them "equal preference" for the direct method of calculating the spectral function (16), which is a more plausible outcome compared to the fast method.

C. *Amplitude characteristics of the quantization*

Fig. 8 shows the amplitude characteristics obtained for φ_0=0° on the integer and half-integral signal frequency.

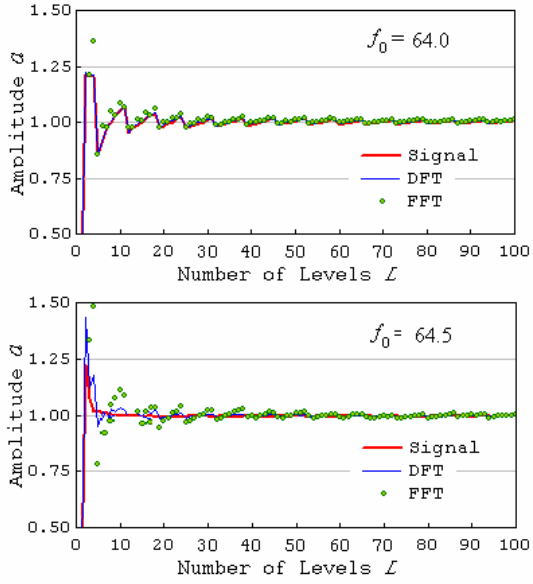


Fig. 8. The amplitude of the integer and half-integer frequency sine wave depending on the number of quantization levels of signal and basis functions of DFT and FFT

It is evident that the behavior of curves in all the three quantization modes depends strongly on the leakage effect. Closer to each other are results given by quantization of signal and DFT basis functions. However, the quantization of the FFT basis functions increases the existing error of the amplitude *a*.

D. Phase characteristics of the quantization

Fig. 9 shows the results of measuring the zero initial phase $\varphi_0=0$ of a harmonic signal obtained in the absence of leakage and when it has the maximum effect.

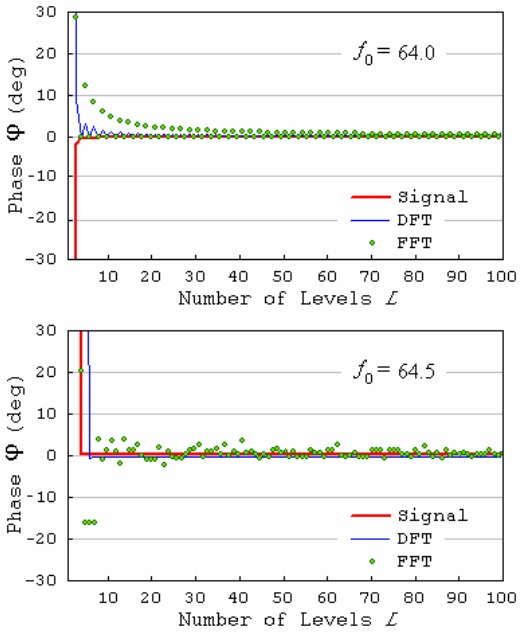


Fig. 9. Measurements of zero-phase of an integer and a half-integer frequency sine wave depending on the number of quantization levels of signal and basis functions of DFT and FFT

Just as in the measurement of the amplitude, closer results are obtained by quantization of signal and DFT basis functions.

However, the quantization of the FFT basis functions significantly increases the error in phase φ, that indicates the downside of fast algorithm application for small *L*.

V. Conclusion

Level quantization, which dramatically changes the signal appearance, has little effect on the precision of spectral analysis results. Frequencies of process harmonics are determined with a sufficient accuracy for $L>2$ signal quantization levels, amplitudes – for $L>8$ levels, and phases – for $L>64$ levels. The amplitude and phase spectra distortions are more noticeable in case of the signal clipping.

Level quantization of discrete Fourier transform's basis functions produces by means of the fast calculation method a markedly greater error of spectral estimates of the parameters of the harmonic components of the varying process as compared to the direct calculation method. Therefore, FFT and other fast algorithms of digital signal processing, implemented in microelectronic devices with a small number of memory cells' bits, should undergo metrological certification as a means of measuring the parameters of processes.

In those DSP areas where spectral analysis methods can be applied, processing a large amounts of data from many sources (sensors) of data acquisition (aviation engines stand testing, meteorology, monitoring of environment and engineering facilities of great length: railways, oil and gas pipelines, etc.) can be carried out with multiple data contraction by selecting a small number of quantization levels for the signals. This justifies demand for designing, producing and purchasing electronic and computing devices with a small number of bits of data provided.

The proposed method of investigating the effects of quantization and clipping can be used in various fields of digital signal processing and related disciplines (wavelet analysis, sequential analysis, etc.).

References

[1] R. E. Blahut, *Fast algorithms for digital signal processing*. MA: Addison-Wesley, 1984.

[2] L. R. Rabiner, and B. Gold, *Theory and application of digital signal processing*. NJ: Prentice-Hall, 1975.

[3] I. S. Gonorovsky, *Radio-engineering circuits and signals*. Moscow: Sovetskoye Radio, 1977.

[4] A. V. Oppenheim, and R. W. Schafer, *Discrete-time signal processing*. NJ: Prentice-Hall, 1989.

[5] A. Yurdakul, and G. Dündar, "Statistical Methods for the Estimation of Quantization Effects in FIR-Based Multirate Systems," *IEEE Trans. Signal Process.*, vol. 47, no. 6, pp. 1749–1753, Jun. 1999.

[6] Yu. A. Bryuhanov, "Quantization and overflowing effects in digital recursive first-order filters with round-off under constant external influence," in *Proc. of 8th Int. Conf.* "*Digital Signal Processing and its Applications*", Moscow, 2006, vol. 1, pp. 191–194.

[7] J. Paduart, J. Schoukens, and Y. Rolain, "Fast measurement of quantization distortions in DSP Algorithms," *IEEE Trans. Instrum. Meas.*, vol. 56, no. 5, pp. 1917–1923, Oct. 2007.

[8] G. S. Khanyan, "Analytical investigation and estimation of errors involved in the problem of measuring the parameters of a harmonic signal using the Fourier transform method," *Measurement Techniques*, NY: Springer, Aug. 2003, vol. 46, pp. 723–735 [*Izmeritel'naya Tekhnika*, Russia, 2003, no. 8, pp. 3–10].

[9] G. S. Khanyan, and N. V. Sheina, "A digital signal processing system for data support of GTE stand tests," in "*Scientific Contribution to the Creation of Aviation Motors*", vol. 2, V. A. Skibin, and V. I. Solonin (ed.), Moscow: Mashinostroenie, 2000, pp. 534–536.

Gamlet S. Khanyan (M'10) – Was born in 1950. Graduated from the Moscow Physical-Technical Institute in 1974. Works at the Central Institute of Aviation Motors in the Department of Dynamic Measurements and Signal Processing as a Senior Research Fellow. In 2004, defended his thesis for a Ph.D. degree in Technical Sciences on the subject "Development of the Discrete Spectral Analysis of Fast Varying Dynamic Processes as Applied to the Information Support of Gas Turbine Engines Tests". Member of the Russian A. S. Popov Society for Radioengineering, Electronics and Communications since 2009.

Performance Evaluation of AGLETS and JADE Mobile Agent Using Encryption and Decryption Time

Dada E. G., Joseph S. B., and M. K. Mishra

Abstract — **The mobile agent approach is a relatively new concept in the distributed systems environment. The agents migrate from Client to server in a network where the state of the running program is saved, transported to the new host, and are stored, allowing the program to continue from the point where it stopped. In this paper, we evaluate the performance of the JADE and Aglet mobile agents. We developed a simulation program to evaluate the performance of the two mobile agents using the Encryption time, Decryption time and file transfer time. Our findings revealed that there is no significant difference between the performances of these two mobile agents using the parameters mentioned before.**

Index Terms — **Aglets, Decryption, Encryption, JADE, Mobile Agent.**

I. INTRODUCTION

MOBILE agents are autonomous programs that move about the network on behalf of their owners while searching for information or even negotiating with other agents. Mobile agents can also be defined as those agents that possess the characteristics of mobility. This means that the agents have the ability to migrate from one host computer to another. This may seem like a trivial characteristic, but the advantages to mobility are both subtle and important. Mobile agents are also known as programs that are able to migrate in a network in order to optimize their consumption of resources, such as network bandwidth, or to adapt to a changing environment.

A mobile agent migrates from one Host to another Host where the data is sourced. This agent could be a control system (in the simplest case, a thermostat), that reads the source data and then interacts with the system to make adjustments. In this model, the mobile agent interacts with the data collection agent at the source. Moreover, the agent could collect and filter data, and then return to the original host. This type of agent could be useful in situations where full-time connections are not always practically possible (such as satellites in low-earth orbit).

The advent of mobile agents' technology attracts a lot of interest from the fields of distributed systems, information retrieval, electronic commerce and artificial intelligence. The emergence of Java, with its support for mobile code, led to heightened research activity in this area. Java is the language of choice for mobile agent systems such as Concordia, JADE, Odyssey, Aglets, Tracy and Voyager. Java also supports development of mobile agents that are tightly integrated with the Web [1].

Mobile agents have been used in a variety of applications including process control and network monitoring. Network monitoring is an ideal application for mobile agents. An agent is provided details of data collection, and then disbursed into a network. The agent collects data, and either communicates the data back to a central server, or migrates back to itself with its data. Process control is another interesting application. Instead of purely collecting data from remote servers, the agents must also monitor and control the devices to which they're attached. Prior to migrating, the agents can be configured for their particular destination. From this perspective, mobile agents are an interesting deployment method for distributed systems.

II. MOBILE AGENT ARCHITECTURE

The mobile agent architectural pattern introduces the ability for agents to migrate themselves between hosts. The agent architecture includes the mobility element, which allows an agent to migrate from one host to another. An agent can migrate to any host that implements the mobile framework. The mobile agent framework provides a protocol that permits communication between hosts for agent migration. This framework also requires some kind of authentication and security, to avoid a mobile agent

Manuscript received November 18, 2010. Performance evaluation of AGLETS and JADE mobile agent using encryption and decryption time.

Dada E. G. is with the Department of Computer Engineering, University of Maiduguri, Nigeria (e-mail: gbengadada2004@ yahoo.com).

Joseph S. B. is with the Department of Computer Engineering, University of Maiduguri, Nigeria (e-mail: sjbassi74@ yahoo.co.uk).

M. K. Mishra is with the Department of Computer Engineering, University of Maiduguri,Borno State,Nigeria, (Corresponding author phone: +234-8065578635 e-mail: mishrasoft1@gmail.com).

framework from becoming a conduit for viruses. Also implicit in the mobile agent framework is a means for discovery. For example, which hosts are available for migration, and what services do they provide? Communication is also implicit, as agents can communicate with one another on a host, or across hosts in preparation for migration. The mobile agent architecture is advantageous as it supports the development of intelligent distributed systems that is dynamic, and whose configuration and loading is defined by the agents themselves (see Figure 1).

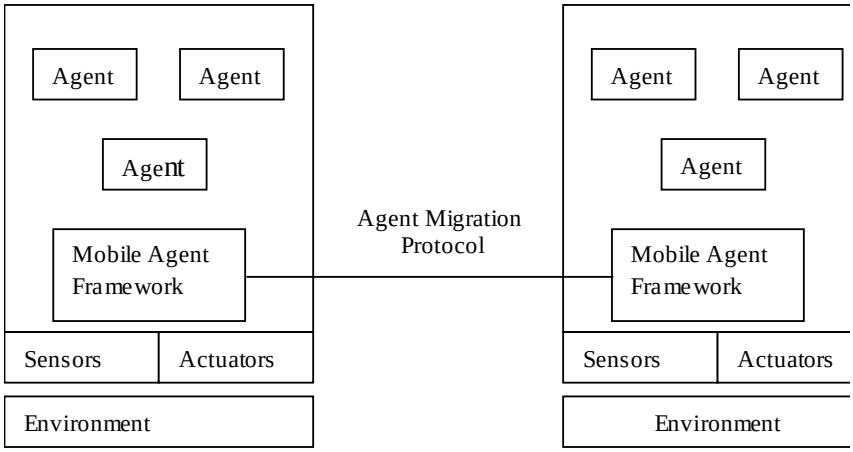


Fig. 1. The mobile agent framework supports agent mobility [5]

A. Technical Obstacles in the Development of Mobile Agents

The Mobile agent paradigm is a promising technology and a new method of communication amongst network nodes. Despite a number of successful mobile agent applications, still there are some barriers preventing this technology from spreading out to a wider range of enterprise and individual users. This is due to many reasons such as, lack of standard in both software and hardware products (e.g. programming languages, protocols and devices). To overcome this, a number of initiatives are underway which may help developers in building their applications based on mobile agent technology as in [3].

Researchers and the developers also find it difficult to define the real concept of mobile agent technology and the tasks that Mobile Agent should perform is another contributing problem. Furthermore, the current infrastructure is not ready to support and integrate with mobile agent technology. Another major concern made by researchers is the security issue, for example, when using MAs whether in E-commerce or M-commerce fields, to act on behalf of their users to handle transactions over the net.

Other unresolved issues include privacy, trust and integrity. Privacy is lost since the agent must have access to the user profile, which may contain sensitive information about the user, and may be shared with other agents in the working environment. In addition, this information may be modified during the transaction (by a hacker for example).

TABLE I
TECHNICAL IMPLICATIONS OF MOBILE AGENTS

Technical Issue	Implication for Mobile Agents
Bandwidth	MAs conserve bandwidth, especially for networks which have low bandwidth capacity (e.g. wireless network). By replacing continuous communication with an agent directly at the point of information generation, the bandwidth use can be reduced. Instead of sending dozens or even hundreds of queries across the network, sending one agent on a single request the agent can manage this process locally at the remote side.
Fault-tolerance	MAs can act or respond on errors that may be encountered within their contexts because of their adaptive and ragged attributes.
Flexibility	MAs can give greater flexibility, because new tasks and codes can be added to the system without the need for a fixed code-base.
Interaction	Mobile agents enable new types of interaction, such as negotiating agents that travel to vendors' sites/servers seeking for the best deal such as comparing prices (e.g. e-commerce application).
Protocols	MAs are able to move (relocate itself) to remote hosts in order to establish "channels" based on proprietary protocols.
Scalability	MAs can carry out their function well (without disruption) when the host system or environment changes in size or volume in order to meet a new user's need.
Self-contained tasks	MAs can carry out tasks which require variable degrees of independence such as, network management, software updates, etc.
Weak coverage	MAs fit perfectly into a disconnected environment where the signal coverage is frequently lost (being disconnected); MAs will then migrate from one node to another when the coverage becomes available.

These issues are trivial but it has to be considered in the mobile agent applications. In other words, MAs need to be protected against hosts, and hosts need to be protected against MAs. In summary, the Table I briefly explains the implications of using MAs for 8 identified significant technical issues.

B. JADE

Java Agent DEvelopment Framework (JADE) is a software framework originally developed by TILAB, Italy and it is totally written in Java. It is an enabling technology, a middleware for the development and run-time execution of peer-to-peer applications which are based on the agents' paradigm. It also simplifies the implementation of multi-agent systems through a middleware that complies with the Foundation for Intelligent Physical Agents (FIPA) specifications. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required.

The conceptual model of JADE dwells mainly on distributed system topology with peer-to-peer networking, and software component architecture with agent paradigm. The network topology affects how the various components are linked together, whereas the component architecture specifies what the components are supposed to expect from one another. The intelligence, initiative, information, resources and control are fully distributed on mobile terminals as well as on computers in the fixed network. Agents otherwise called "peer" evolve dynamically in JADE, appearing and disappearing in the system according to the needs and the requirements of the application environment. Communication between the peers, regardless of whether they are running in the wireless or wired network, is completely symmetric with each peer being able to play both the initiator and the responder role.

The development of JADE according to [4] is based on the following driving principles:

Interoperability: JADE is compliant with the FIPA specifications. As a consequence, JADE agents can interoperate with other agents, provided that they comply with the same standard.

Uniformity and portability: JADE provides a homogeneous set of APIs that are independent from the underlying network and Java version. More in details, the JADE run-time provides the same APIs both for the J2EE, J2SE and J2ME environment. In theory, application developers could decide the Java run-time environment at deploy-time.

Easy to use: The complexity of the middleware is hidden behind a simple and intuitive set of APIs.

Pay-as-you-go philosophy: Programmers do not need to use all the features provided by the middleware. Features that are not used do not require programmers to know anything about them; neither do they add any computational overhead.

Architectural Model: JADE includes both the libraries (i.e. the Java classes) required to develop application agents and the run-time environment that provides the basic services and that must be active on the device before agents can be executed. Each instance of the JADE run-time is called container (since it "contains" agents). The set of all containers is called platform and provides a homogeneous layer that hides to agents (and to application developers also) the complexity and the diversity of the underlying tires (hardware, operating systems, types of network, JVM).

As seen in Figure 2, JADE is compatible with the J2ME

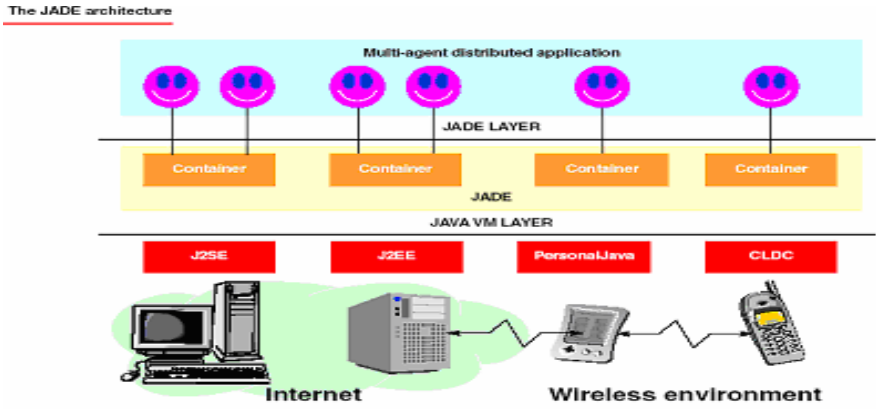


Fig. 2. JADE architectural model (Source: Bellifemine F., JADE a White Paper, Exp Journal)

CLDC/MIDP1.0 environment. It has already been tested on the fields over the GPRS network with different mobile terminals among which include Nokia 3650, Motorola Accompli008, Siemens SX45, PalmVx, Compaq iPaq, Psion5MX, HP Jornada 560. The JADE run-time memory footprint, in a MIDP1.0 environment, is around 100 KB, but can be further reduced until 50 KB using the ROMizing technique i.e. compiling JADE together with the JVM. JADE is extremely versatile and therefore, not only does it fit the constraints of environments with limited resources, but it has already been integrated into complex architectures such as .NET or J2EE where JADE becomes a service to execute multi-party proactive applications. The limited memory footprint allows installing JADE on all mobile phones provided that they are Java-enabled.

C. *Aglets*

Aglet is a mobile Java object that visits aglet enabled hosts in a computer network. It is autonomous, since it runs in its own thread of execution after arriving at a host, and reactive, because of its ability to respond to incoming messages [2]. Aglet agent framework was designed by IBM Tokyo in the 1990s. Aglets is based on the Java programming language, as it is well suited for a mobile agents framework. First, the applications are portable to any system (both homogeneous and heterogeneous) that is capable of running a Java Virtual Machine (JVM). Second, a JVM is an ideal platform for migration services. Java supports serialization, which is the aggregation of a Java application's program and data into a single object that is restartable. In this case, the Java application is restarted on a new JVM. Java also provides a secure environment (sandbox) to ensure that a mobile agent framework doesn't become a virus distribution system [5].

The Aglets framework is shown in Figure 3.

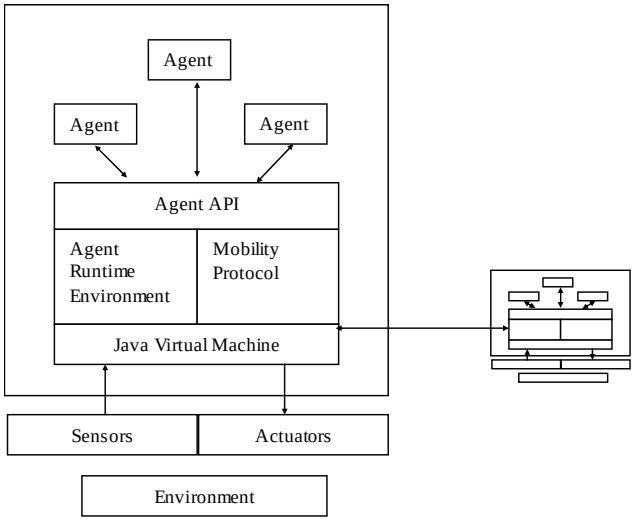


Fig. 3. Aglets mobile agent architecture [5]

At the bottom of the framework is the JVM (the virtual machine that interprets the Java bytecodes). The agent runtime environment and mobility protocol are next. The mobility protocol, called Aglet Transport Protocol (or ATP), provides the means to serialize agents and then transport them to a host previously defined by the agent. The agent API is at the top of the stack, which in usual Java fashion provides a number of API classes that focus on agent operation. Finally, there are the various agents that operate on the framework. The agent API and runtime environment provide a number of services that are central to a mobile agent framework. Some of the more important functions are agent management, communication, and security. Agents must be able to register themselves on a given host to enable communication from outside agents. In order to support communication, security features must be implemented to ensure that the agent has the authority to execute on the framework. Aglets provides a number of necessary characteristics for a mobile agent framework, including mobility, communication, security, and confidentiality. Aglets provide weak migration, in that the agents can only migrate at arbitrary points within the code (such as with the dispatch method).

III. Discussion

We developed an application for the implementation of two mobile agent systems on win32 platform that is, windows XP SP2. Five dummy files whose size ranges from 500kb to 1mb were created on the client system. The results of the comparison encryption and decryption time between Aglets and JADE can be view from the server. The time is measured in milliseconds while the size of the files is measured in bytes. From the experiments performed, it was observed that there is difference in encryption and decryption time between Aglets and JADE with the latter giving a better performance than Aglets in all test cases. The time difference was not much, it was also observed that the time taken to encrypt, decrypt and transfer files increases as the file sizes increases.

IV. Analysis of Results

The Table II is a comparison between the encryption time of each file from 500kb – 1mb for JADE and Aglets (see Figure 4).

TABLE II
Encryption Time Comparison between JADE and Agelets

	Encryption Time (ms)	
Size (kb)	JADE	Aglet
500	760	765
600	1681	1686
700	1700	1705
800	1761	1766
900	1971	1977
1000	2052	2058

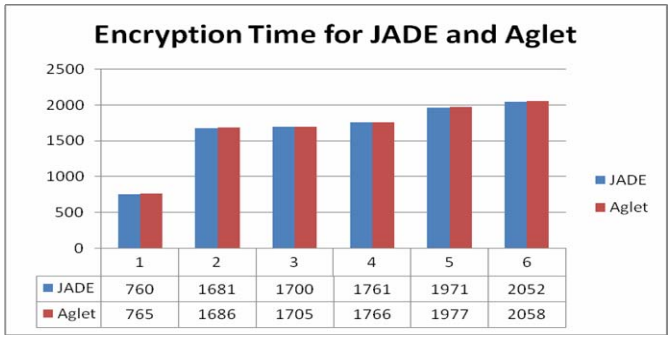


Fig. 4. Graph showing encryption time comparison between JADE and Aglets

The Table III is a comparison between the decryption time of each file from 500kb – 1mb for JADE and Aglets (see Figure 5).

The Table IV shows the comparison of JADE and Aglet in terms of the time it takes to send files from one computer system to another one (see Figure 6).

It was observed from the Table IV that the rate at which JADE transfers the encrypted files is to some extent faster than that of Aglets.

TABLE III
DECRYPTION TIME COMPARISON BETWEEN JADE AND AGELETS

	Decryption Time (ms)	
Size (kb)	JADE	Aglet
500	302	300
600	455	451
700	510	504
800	612	1766
900	667	661
1000	730	731

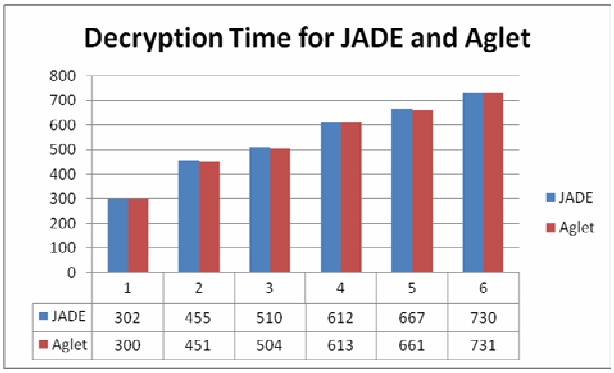


Fig. 5. Graph showing decryption time comparison between JADE and Aglets

TABLE IV
FILE TRANSMISSION TIME COMPARISON BETWEEN JADE AND AGELETS

	Time taken to send files from one system to another (ms)	
Size (kb)	JADE	Aglet
500	169,337,778,715	169,337,778,713
600	169,337,778,775	169,337,778,773
700	169,337,779,790	169,337,779,788
800	169,337,780,037	169,337,780,034
900	169,337,780,318	169,337,780,315
1000	169,337,780,500	169,337,780,497

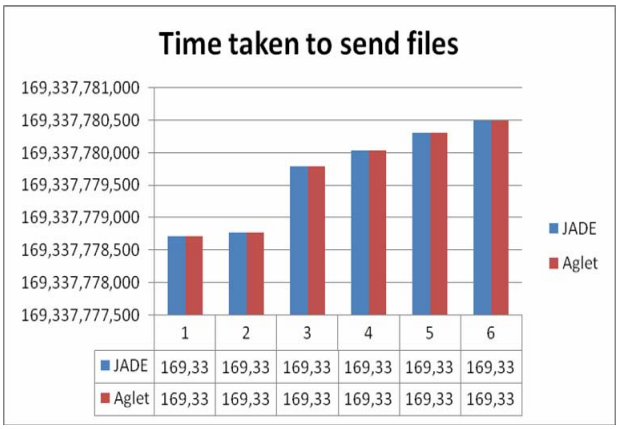


Fig. 6. Graph showing file transmission time comparison between JADE and Aglets

V. CONCLUSION

The results obtained showed that there is only some slight difference in performance between JADE and Aglet in terms of Encryption time, Decryption time and file transfer. The little differences could be due to the fact that all information exchanged by JADE complies with FIPA specification and hence include only the information required by the transport layer unlike Aglet that exchanges all data. Another factor is that JADE support skeletons that are implemented as abstract classes that relief the programmer the burden of solving synchronization, timeouts and other challenges. Also JADE uses the asynchronous method of messaging, which puts the Agent Communication Language (ACL) into consideration and supports multiple agent execution and interaction. This makes it more preferred in multi agent distributed environments. Aglets use API for transfer of agent and RMI for exchange of messages which is not in line with FIPA regulations that JADE is using. Our future work will focus on using memory utilization and fault tolerance to test the performance of mobile agents across different platform.

REFERENCES

[1] Wong, D., Paciorek, N. and Moore D., "Java-based mobile agents," Communications of the ACM 42(3), 1999, pp. 92-105.

[2] Lange, D. B. and Oshima, M., "Programming and deploying Java mobile agents with aglets," Addison-Wesley, 1998.

[3] FIPA, "Foundation for intelligent physical agents", 2004, URL: http://www.fipa.org/

[4] Bellifemine, F., "JADE – a white paper. EXP. 3.3", Retrieved October 1, 2010 from http://exp.telecomitalialab.com, , 2003

[5] Jones Tim M., "Artificial intelligence: A System Approach.," Infinity Science Press LLL, 2008.

A Low-Cost Optimal Time SIC Pair Generator

I. Voyiatzis[1], H. Antonopoulou[2], C. Efstathiou[1]
[1]*Department of Informatics, Technological Educational Institute of Athens, Greece*
[2] *Technological Educational Institute of Patras, Computer Technology Institute, Patras, Greece*
voyageri@otenet.gr

***Abstract* - The application of Single Input Change (SIC) pairs of test patterns is very efficient for sequential, i.e. stuck-open and delay fault testing. In this paper a novel implementation for the application of SIC pairs is presented. The presented generator is optimal in time, in the sense that it generates the n-bit SIC pairs in time n×2n, i.e. equal to the theoretical minimum. Comparisons with the schemes that have been proposed in the open literature which generate SIC pairs in optimal time reveal that the proposed scheme requires less hardware overhead**

***Keywords* - Built-In Self Test, Two-Pattern Testing, Delay Fault Testing, Stuck-Open Testing**

I. INTRODUCTION

WHILE every VLSI design project has its own unique set of goals, there is a fundamental need for reliability in the finished product. Built-In Self Test (BIST) [1] constitutes an attractive and practical solution. Advantages of BIST include the possibility of performing at-speed testing, very high fault coverage, elimination of test generation effort and less reliance on expensive external testing equipment for applying and monitoring test patterns. Therefore BIST reduces the cost of testing. With the increasing complexity of today's VLSI devices (with millions of gates) BIST schemes for embedded modules are increasingly becoming a necessity.

Despite of the fact that exhaustive single-pattern testing provides for 100% fault coverage of detectable single and multiple stuck-at faults without the need for fault simulation or deterministic test pattern generation, it is widely known that a large class of physical defects can not be modeled as stuck-at faults. For example, a transistor stuck-open fault in a CMOS circuit can convert a combinational Circuit Under Test (CUT) into a sequential one [2] while a delay fault (although it does not affect the steady-state operation) may cause circuit malfunction at clock speed [3]. Detection of such faults requires two-pattern tests.

In the literature, two largely known types of two-pattern tests have been investigated, Multiple Input Change (MIC) and Single Input Change (SIC) pairs. SIC pairs are pairs of patterns in which the first pattern differs from the second one in exactly one bit. The utilization of SIC pairs for the detection of stuck-open and delay faults holds some very interesting properties and has been studied by a number of researchers both theoretically [11] and experimentally [29], [31]-[38]. In the theoretical field, Smith [11] proved that SIC tests are sufficient to detect all robustly detectable path delay faults. In the experimental field, Wang and Gupta [6] proved that SIC pairs provide higher pseudorandom robust path delay fault coverage than MIC pairs. In other words, if a certain number of pairs is applied to the inputs of a Circuit Under Test (CUT), if the pairs are SIC, the achieved fault coverage will be higher than the case in which the pairs are MIC. Gizdarski [40] utilized SIC sequences in order to test delay faults in the address decoders of RAM memories. The above-referenced results, as well as a number of related works [31-38] indicate that the utilization of SIC pairs for testing delay and stuck-open faults compares favorably to the utilization of MIC pairs, since it results in higher fault coverage with fewer test vectors.

In this paper a novel technique is presented for the generation of SIC pairs of patterns. The number of cycles required to generate the SIC pairs is n×2n, i.e. equal to the theoretical minimum. Comparisons with schemes proposed previously for the application of SIC pairs in optimal time indicate that the proposed scheme requires less hardware overhead.

The paper is organized as follows. In Section 2 the proposed scheme is introduced. In Section III the hardware implementation is presented. In Section IV the techniques presented in the literature for the generation of SIC pairs in optimal time are compared. Finally, in Section V we conclude the paper.

II. IMPLEMENTATION OF THE PROPOSED SCHEME

Definition 1. We define by $G_n = (g_{n-1}, g_{n-2}, \ldots, \ldots, g_1, g_0)$ the 2^n-row by n column matrix that is the output of a binary-reflected gray code.

For example, for n=3, $G_3 = (g_2, g_1, g_0)$ is the 8-row 3 column matrix presented in the sequel.

000
001
011
010
110
111
101
100

Definition 2: We define by $G^i = T^i(G)$, $1 \le i < n$ the 2^n-row by n column matrix that is generated from G by cyclically shifting the columns one position to the right and inverting the high- and low-order columns.

For example, for G= (g_2, g_1, g_0), $G^1 = T^1(G) = (g_0', g_2, g_1')$, $G^2 = T^2(G) = T^1(T^1(G)) = (g_1, g_0', g_2')$ and $G^3 = T^3(G) = T^1(T^2(G)) = T^1(T^1(T^1(G))) = (g_2\ g_1\ g_0) = G$. In the following table, we present the matrices G, G^1 and G^2 for n=3.

G = (g2 g1 g0)	$G^1 = T^1(G) = (g_0', g_2\ g_1')$	$G^2 = T^2(G) =$ (g1 g0' g2')
000	101	011
001	001	001
011	000	101
010	100	111
110	110	110
111	010	100
101	011	000
100	111	010

It is trivial to show that for any value of n, $G_n^n = T^n(G_n) = G_n$.

Hayes [41] proposed a procedure to construct all SIC pairs within $n \times 2^n$ cycles by applying the n sequences G_n, G_n^1, G_n^2, … G_n^{n-1}, and presented an intuitive proof for the correctness of the construction. The proposed generator for the generation of the SIC pairs in optimal time is presented in Figure 1.

The module depicted in Figure 1 comprises an n-stage gray counter (an n-stage counter and n-1 2-input XOR gates), an n-stage barrel shifter, a k-stage counter, a k-input OR gate, a k-to-n decoder with enable and a series of 2-input XOR gates. It operates as follows. Initially, both counters are reset to 0. The n-stage counter starts increasing, hence the sequence G_n is generates at the A[n-1:0] outputs of the generator. When the n-stage counter reaches 2^n-1, the k-stage counter increases to 1. Hence, the outputs of the n-stage counter are shifted one position to the right by the barrel shifter, the output of the OR gate is 1 and the decoder output is 00…01. Therefore, the sequence $G_1 = (g_0'\ g_{n-1}\ g_{n-2} \cdots g_2\ g_1')$ is generated. When this completes, the k-stage counter is increased again, the shifter shifts the outputs of the gray counter two positions to the right, the output of the OR gate becomes 1 again and the decoder output becomes 00…010; therefore, the sequence $G_2 = (g_1\ g_0'\ g_{n-1}\ g_{n-2} \cdots g_2')$ is generated and so on.

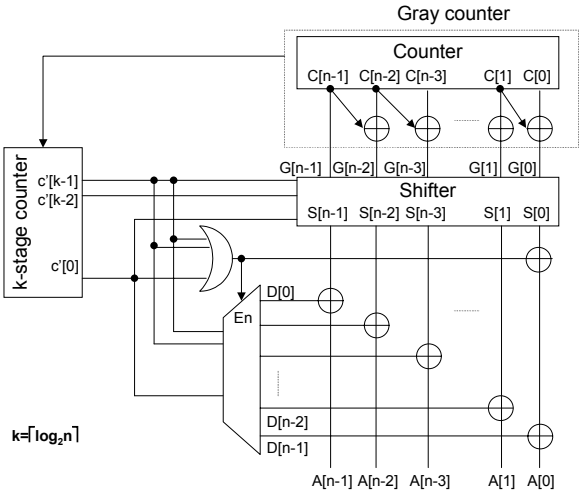
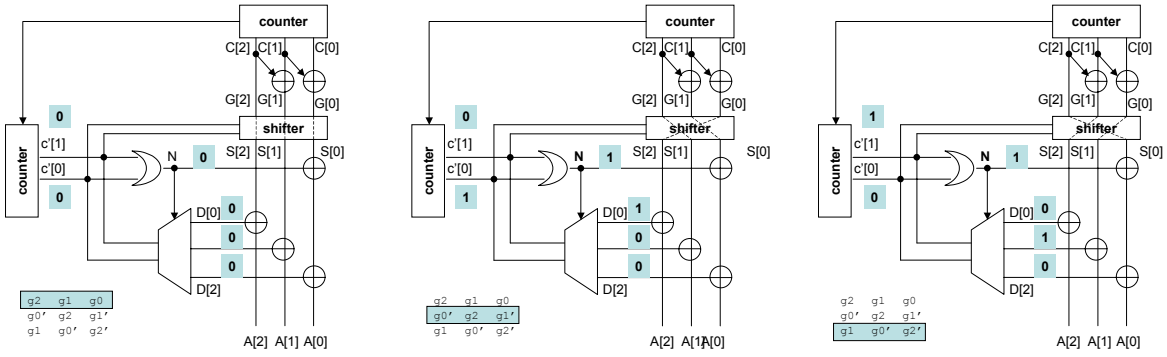


Fig. 1. The proposed generator

In order to exemplify the operation of the proposed generator, in Figure 2 we present the operation for the case n=3. During the first phase, Figure 2 (a), the $G_3 = (g_2\ g_1\ g_0)$ sequence is applied to the A[2:0] outputs. During the second phase, Figure 2 (b), the sequence $G_3^1 = (g_0'\ g_2\ g_1')$ is applied to the A[2:0] outputs; finally, during the third phase, the sequence $G_3^2 = (g_1, g_0'\ g_2')$ is applied.



a b c

Fig. 2. Operation of the proposed generator for n=3

III. CALCULATION OF THE HARDWARE OVERHEAD OF THE PROPOSED SCHEME

In the application of the proposed scheme, implementing the Gray generator requires an n-bit counter accompanying (n-1) XOR gates; also, the n-bit barrel shifter is required (n×$\log_2$n flip flops) the k-stage counter (k=$\log_2$n), the k-to-n decoder with enable, a k-input OR gate, and (n+1) additional XOR gates are required. To calculate the hardware overhead of the k-to-n decoder, we follow a reasoning similar to that used in [27].

A *k*-to-*K* decoder can be implemented as follows. Let $k_1=\lfloor \frac{k}{2} \rfloor$ and $k_2=\lceil \frac{k}{2} \rceil$. Then $k_1+k_2=k$. A *k*-to-*K* decoder ($K=2^k$) with enable input can be implemented using two subdecoders with enable (k_1-to-K_1) and (k_2-to-K_2) and *K* 2-input NOR gates. The first k_1-to-K_1 subdecoder is denoted by D_a; its inputs are denoted by d_{a0} to $d_{ak1\text{-}1}$ and its outputs are denoted by D_{a0} to $D_{aK1\text{-}1}$; the second subdecoder is denoted by D_b; its inputs are denoted by d_{b0} to $d_{bK2\text{-}1}$; its outputs are denoted by D_{b0} to $D_{bK2\text{-}1}$. All outputs of the two subdecoders are inverted using K_1+K_2 inverters. Each one of the *K* gates takes two inputs: the first is an output of the first decoder; the second is an output of the second decoder as follows. $D_0=\overline{D_{a0}+D_{b0}}$; $D_1=\overline{D_{a0}+D_{b1}}$, ..., $D_{K\text{-}1}=\overline{D_{aK1-1}+D_{bK2-1}}$. For example, in Figure 3 we present a 3-to-8 decoder using the above-mentioned procedure.

For the proposed scheme, only *n* out of 2^k outputs are implemented (n≤2^k). In Table I the Hardware Overhead (in transistors) for various values of *n*, the inputs of the CUT is presented.

TABLE I
Patterns generated by the module in Figure 1

c'[1:0]	C[2:0]	G[2:0]	S[2:0]	N	D[0:2]	A[2:0]
00	000	000	000	0	000	000
	001	001	001			001
	010	011	011			011
	011	010	010			010
	100	110	110			110
	101	111	111			111
	110	101	101			101
	111	100	100			100
01	000	000	000	1	100	101
	001	001	100			001
	010	011	101			000
	011	010	001			100
	100	110	011			110
	101	111	111			010
	110	101	110			011
	111	100	010			111
10	000	000	000	1	010	011
	001	001	010			001
	010	011	110			101
	011	010	100			111
	100	110	101			110
	101	111	111			100
	110	101	011			000
	111	100	001			010
00	000	000	000	0	000	000

In Table I, in the first column we present the value of n and in the second column the value of k=log2n; in the third and fourth columns we present the k1 and k2 values such that k1+k2=k; in the seventh and eighth columns the hardware overhead of the two sub-decoders is presented; in the ninth column the overhead of the n 2-input gates is presented. In the tenth column, the hardware overhead of the decoder for each value of *n* is presented. This value will be considered for the calculation of the hardware overhead of the proposed scheme. For the calculations, an *m*-input NAND/NOR gate is considered to have 2*m* transistors and an *m*-input AND 2*m*+2 transistors [23]. The hardware overheads of the 2x4, 3x8 and 4x16 decoders are 26, 66, and 116 transistors respectively.

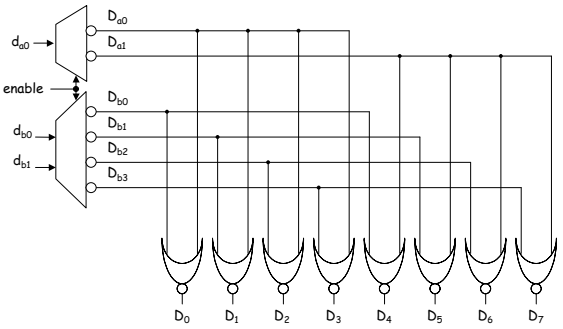


Fig. 3. Implementation of 3-to-8 decoder utilizing smaller decoders and 2-input gates

TABLE II
Calculation of *k*-to-*n* decoder (*n*≤2^k) with enable hardware overhead (in transistors)

n	k	Dec_1	Dec_2	Dec_1 H/W	Dec_2 H/W	n 2-input gates	Dec H/W
12	4	2x4	2x4			48	100
16					26	64	116
20	5		3x8			80	172
24						96	188
28						112	204
32				26		128	220
36	6	3x8				144	276
40						160	292
44						176	308
48						192	324
52						208	340
56						224	356
60						240	372
64					66	256	388
68	7		4x16			272	404
72						288	470
76						304	486
80						320	502
84						336	518
88						352	534
92						368	550
96						384	566
100						400	582
104						416	598
108						432	614
112						448	630
116						464	646
120						480	662
124						496	678
128				66	116	512	694

IV. COMPARISONS

In this section, the proposed scheme will be compared with the techniques proposed hitherto for the generation of SIC pairs in optimal time, i.e. in exactly n×2n clock cycles [5], [37], [40] in terms of the required hardware overhead.

In PEAT [5], an *n*-stage NFSR, an *n*-stage *shift register* and an *n*-stage *shift register with flip capability* are utilized to generate the SIC pairs within $(n+1)\times 2^n$ clock cycles. To implement the technique, the NFSR and *n scan flip-flops with flip capability* are implemented. Furthermore, the *n* flip-flops of the existing register are substituted by *scan flip-flops*.

In [37], Das *et al* presented an optimal solution to the problem of generating SIC pairs, in the sense that the pairs are generated within time equal to the theoretical minimum, i.e. $n\times 2^n+1$. However, the hardware overhead of [37] is rather high, thus the value of the scheme lies mainly on its high theoretical significance. The hardware overhead of the scheme is, according to [37], 3n+2 flip flops, n XOR gates (2-input), (2*n*-1) OR gates (2-input), n+1 AND gates (2-input) and 1 NOT gate.

Gizdarksi [40] utilized the algorithm proposed by Hayes in order to generate the SIC pairs to the inputs of the address decoder of a RAM. Gizdarski utilized two sequences, called TS1 and TS2 in [40]; TS2 is utilized in order to detect additional faults in the address decoder (and its generation is more complicated and hardware intensive). Hence the generator for the TS1 sequence is considered for our comparisons. The required hardware includes control logic, an n-bit binary counter, an n-bit register, n 2-input gates, and $n\times\log_2 n$ 2-to-1 multiplexers in a barrel shifter. Since no information is provided in [40] for the control logic, we shall not take it into account in our comparisons.

For the comparisons, the following are taken into account [23]. A 2-input NAND/NOR gate requires 4 transistors; a 2-input AND requires 6 transistors and a 2-input XOR gate can be implemented by 4 CMOS transistors [42]. The memory elements used are considered to have set/reset capability. Thereby, the flip-flop requires 26 transistors, the *scan flip-flop* requires 34 transistors and the *scan flip-flop with flip capability* [5] requires 46 transistors.

In Table III we present, for each one of the optimal SIC pair generation techniques (first column) the formulas used for the calculation of the hardware overhead (second column) and the hardware overhead (in transistors, third column). In Figure 4 we present, for various values of n, the ratio of the hardware overhead (in transistors) over n, the number of CUT inputs. From Figure 4 it can be concluded that the proposed scheme presents the least hardware overhead of the schemes that have been proposed in the open literature.

V. CONCLUSION

In this paper a novel generator for Single Input Change two-pattern tests has been presented. The number of cycles required to generate the SIC pairs is $n\times 2^n$, i.e. equal to the optimal (minimum) time required. Comparisons with the techniques that have been proposed in the literature for the generation of SIC pairs in optimal time revealed that the proposed scheme requires less hardware overhead.

TABLE III
Optimal-time SIC pair generation techniques: Comparison

	Hardware Overhead	
Technique	**Modules**	**Transistors**
Peat [5]	$n\times$(DFF+NOR)+$n\times DFF_{scanwithflip}$+ $n\times DFF_{scan}$	$110\times n$
Gizdarski [40]	Control + n×DFF + n×DFF + n×AND_2 + n×$\log_2$n×MUX_{21}	n×(58+log2n)
DAS [37]	(2n+2)×DFF + n×XOR + (2n-1)×OR_2 + (n+1)×AND_2 + NOT	84×n+40
Proposed	n×DFF + (n-1)×XOR + $\log_2$n×DFF + n×$\log_2$n ×MUX_{21} + log2n-to-n Dec + (n+1)×XOR + $\log_2$n-input OR	$n\times(38+6\times\log_2 n)+20\times\log_2 n$

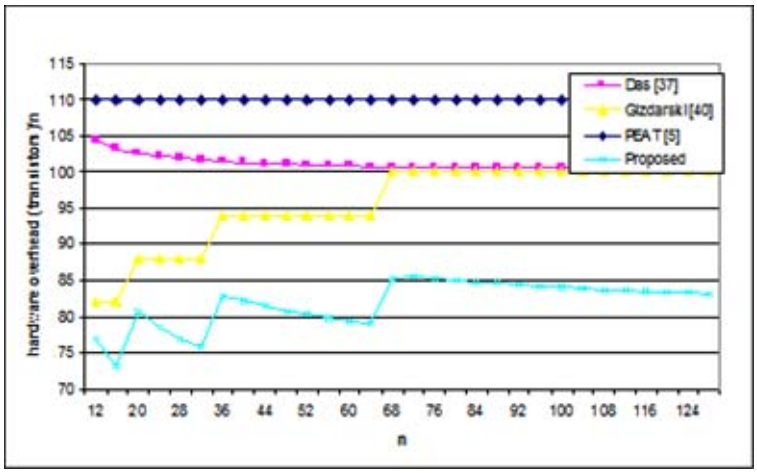


Fig. 4. Optimal time SIC pair generation schemes: Comparison

REFERENCES

[1] Abramovici M., Breuer M., Freidman A., *"Digital Systems Testing and Testable Design"*, Computer Science Press, 1990.

[2] Wadsack R. L., *"Fault Modeling and logic simulation of CMOS and MOS integrated circuits"*, Bell Syst. Tech. J., vol. 57, no. 5, pp. 1449-1474, May-June 1987.

[3] Woods M.H., "*MOS VSI Reliability and Yield Trends",* Proceedings of the IEEE, vol. 74, no. 12, Dec. 1986, pp. 1715-1729.

[4] C. Dufaza, Y. Zorian, On the Generation of Pseudo-Deterministic Two-Pattern Test Sequences with LFSRs, Proceedings of the European Design and Test Conference, Paris, France, March 17-20 1997, pp. 69-76.

[5] Craig G., Kime Ch., *"Pseudo-Exhaustive Adjacency Testing: A BIST Approach for Stuck-open Faults"*, IEEE International Conference, pp. 126-137, 1985.

[6] Wang W., Gupta S., *"Weighted Random Robust Path Delay Testing of Synthesized Multilevel Circuits"*, IEEE VLSI Test Symposium, pp. 291-297, 1994.

[7] Girard P., Landrault C., Moreda V., Pravossoudovitch S., "*An Optimized BIST Test Pattern Generator for Delay Testing*", in. Proc. VLSI Test Symposium, 1997, pp. 94-100.

[8] Voyiatzis I., Kranitis N., Gizopoulos D., Paschalis A., Halatsis C., "An Accumulator-based Built-In Self-Test Generator for Robustly Detectable Sequential Fault Testing", IEE Proceedings, Computers and Digital Techniques, vol. 151, no. 6, pp. 466-472, November 2004.

[9] S. Nandi, B. Vamsi, S. Chakraborty, P.Pal Chaudhuri, *Cellular Automata as a BIST structure for testing CMOS circuits*, IEE Proc.-Com. Digit. Tech., Vol. 141, No. 1, pp. 41-47, January 1994.

[10] Voyiatzis I., Paschalis A., Nikolos D., Halatsis C., *"An Efficient Built-In Self Test Method for Robust Path Delay Fault Testing"*, Journal of Electronic Testing: Theory and Applications, pp. 219-222, June 1996.

[11] Smith G. L., *"Model for Delay Faults based Upon Paths"*, IEEE International Test Conference, pp.309-314, 1984.

[12] E. Blokken, H. de Keulenaer, F. Catthoor, H. J. de Man, *A flexible module library for custom DSP applications in a Multiprocessor Environment*, IEEE J. Solid-State Circuits, vol. 25, no. 3, pp. 720-729, June 1990.

[13] R.J. Higgins, *Digital Signal Processing in VLSI*, Englewood Cliffs, NJ: Prentice Hall, 1990.

[14] Rajski J., Tyszer J., *"Accumulator-Based Compaction of Test Responses",* IEEE Transactions on Computers, vol. 42, no. 6, pp.643-650, June 1993.

[15] Rajski J., Tyszer J., *"Test Responses Compaction in Accumulators with Rotate Carry adders",* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 12, no. 4, pp. 531-539, April 1993.

[16] A.P. Stroele, "*Test Response Compaction Using Arithmetic Functions*", Proceedings of the 14th VLSI Test Symposium, pp. 380-386, 1996.

[17] J. Rasksi and J. Tyszer, *Arithmetic Built-In Self Test for Embedded Systems*, Prentice Hall PTR, Upper Saddle River, New Jersey, 1998.

[18] Gupta S., Rajski J., Tyszer J., *"Arithmetic Additive Generators of Pseudo-Exhaustive Test Patterns",* IEEE Transactions on Computers, vol 45, no.8, pp.939-949, August, 1996.

[19] A.P. Stroele, "*BIST Pattern Generators Using Addition and Subtraction Operations*", Jetta, 11, pp. 69-80, 1997.

[20] K. Radecka, J. Rajski, J. Tyszer, "*Arithmetic Built-In Self Test for DSP Cores", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*", vol. 16, no. 11, pp. 1358-1369, November 1997.

[21] I. Voyiatzis, A. Paschalis, D. Nikolos, and C. Halatsis. "*Accumulator-Based BIST Approach for Stuck-Open and Delay Testing*", Proc. of the European Design and Test Conference 1995, March 1995, pp. 431-435.

[22] Mano M., "*Digital Design*", Prentice Hall Int., 3rd ed., 2002.

[23] Weste N., Eshraghian K., "*Principles of CMOS VLSI Design: A Systems Perspective*", Addison Wesley Company 1985.

[24] C.W. Starke, *Built-In Test for CMOS Circuits*, Proceedings of the IEEE Int. Test Conf., pp. 309-314, Oct. 1984.

[25] A. Vuksic, K. Fuchs, *A New BIST Approach For Delay Fault Testing*, Proceedings of the European Design and Test Conference, pp. 284-288, March 1994.

[26] Ch. Chen, S. K. Gupta, "*BIST Test Pattern Generators for Stuck-open and Delay Testing",* Proceedings of the European Design and Test Conference, pp. 289-296, March 1994.

[27] Voyiatzis I., Th. Haniotakis, C. Halatsis, "A Novel Algorithm for the Generation of SIC Pairs and its Implementation in a BIST environment", IEE Proc. Circuits, Devices & Systems, vol. 153, Issue 5, October 2006, pp. 427-432.

[28] J. Rajski, J. Tyszer, "*Recursive Pseudoexhaustive Test Pattern Generation*", IEEE Transactions on Computers, vol. 42, no. 12, December 1993, pp.1517-1521.

[29] Girard P., Landrault C., Pravossoudovitch S., Virazel A., "*Comparison between random and pseudorandom generation for BIST of delay, stuck-at and bridging faults*", IEEE On-Line Testing workshop, pp. 121-126, 2000.

[30] Voyiatzis I., Paschalis A., Nikolos D., Halatsis C., "*Exhaustive and Pseudoexhaustive Built-In Two-Pattern Generation for Datapaths*", IEEE International On-Line Test Workshop, 1998.

[31] David R., Girard P., Landrault C., Pravossoudovitch S., Virazel A., "*On using Efficient Test sequences for BIST*", VLSI Test Symposium 2002.

[32] Virazel A., David R., Girard P., Landrault C., Pravossoudovitch S., "*Delay Fault Testing: Choosing between Random SIC and Random MIC sequences*", 2000, IEEE European Test Workshop, pp. 9-14.

[33] Rahaman H, Das D., Bhattacharya B., "*Transition count based BIST for Detecting Multiple Stuck-open Faults in CMOS circuits*", The Second IEEE Asia Pacific Conference on ASICs, Aug. 28-30, 2000.

[34] Crepaux-Motte, Jacomino M., David R., "*An algebraic Method for Delay Testing*", 14th VLSI Test Symposium, 1996, pp. 308-315.
[35] Gharaybeh M., Bushnell M., Agrawal V., "*Parallel Concurrent Path Delay Fault Simulation Using Single-Input Change Patterns*", 9th IEEE International Conference on VLSI Design, pp.426-431, Jan. 1996.
[36] Gharaybeh M., Bushnell M., Agrawal V., "*A parallel-vector Concurrent fault Simulator and Generation of Single-Input-Change Tests for Path-delay Faults"*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 17, no. 9, September 1998.
[37] Das D., Chaudhuri I., Bhattacharya B., "*Design of an Optimal Test pattern Generator for Built-In Self Testing of Path Delay Faults*", Proc. VLSI-97, pp. 205-210, 1997.
[38] S. Lu, M. Lu, "*Testing Iterative Logic Arrays for Delay Faults with a constant number of patterns*", 2002 Int'l Symposium on Electronic Materials and Packaging, pp.492-498.
[39] Brglez F., Bryan D., Kozminski K., "*Combinational Profiles of Sequential Benchmark Circuits*", in Proceedings of the International Symposium on Circuits and Systems (ISCAS), pp. 1229-1234, IEEE, 1989.
[40] Gizdarski E., "*Detection of Delay Faults in Memory Address Decoders*", Journal of Electronic Testing, Volume 16, Number 4, August 2000, pp. 381 – 387.
[41] J. P. Hayes, "*Testing Memories for Single-Cell Pattern Sensitive Faults*", IEEE Trans. on Computers, vol. C-29, no. 3, March 1980.
[42] H. T. Bui, A. K. Al-Sheraidah, Y. Wang, "New 4-transistor XOR and XNOR designs", in Proceedings of the Second IEEE Asia Pacific Conference on ASICs, 2000, pp. 25-28.

Component Architecture with Runtime Type Definition

E. M. Grinkrug, A. R. Shakurov

Abstract — **The component-based approach to software design and development is being focused on. By analyzing the main ideas of this approach, their currently existing implementations, their limitations and promising lines of development we suggest a new component architecture which extends capabilities of existing component technologies. The main principles for building such architecture are described.**

Index Terms — **Runtime environment, software architecture, software engineering, software reusability**

I. Introduction

In the field of software development, the idea of **code reuse** has always been of utmost importance [6], reducing costs, minimizing errors, making the code maintainable. The first examples are program libraries, design patterns [4] and application frameworks. Object-oriented and generic programming [3] also contains this idea at its core.

But the most promising manifestation of the idea of code reuse is probably the component-based software engineering [7]. The term **"component"** is usually used to refer to a program entity that holds data and implements some functionality, which are hidden by a well-defined interface (cf. [8], [9]). A more formal definition is given at sec. IV.

The concept of interface varies from one technology to another. It may be a number of "properties" (or attributes, members etc.) [11] or an independent indivisible entity [1]. There're another options. Below an approach that we consider optimal is presented and justified.

Combining components into a working system is usually though of as a relatively simple procedure (e.g. [2]). The structure of the system, however, is manipulated differently in different technologies. Our goal here is to find the optimal way of organizing component interaction, introducing a good balance between flexibility and ease of use.

Manuscript received September 27, 2010.

E. M. Grinkrug is with the Business Informatics Department, Higher School of Economics, Moscow, Russia (e-mail: egrinkrug@hse.ru).

A. R. Shakurov is with the Business Informatics Department, Higher School of Economics, Moscow, Russia (e-mail: egrinkrug@hse.ru).

II. Limitations of component models

Despite the advantages of the component approach its current implementations have a number of substantial limitations. The core difficulty here is to introduce a component that provides a necessary functionality but doesn't exceed it, bloating the system under development.

One way to strike such compromise is to separate the designtime work with a component from its runtime use. The need for such distinction is realized and implemented to some extent (see [11] e.g.), but we believe more can be done here.

For example, let us suppose we're designing a GUI application and we want to change a text on a button (that already has its functionality somehow connected to the application). The change is considered to be permanent: the text won't be changed during the runtime. The procedure is quite obvious (one has only to set the needed value to the corresponding property of the button), but its implementations in various frameworks share a common flaw: the variable property is left variable for the lifetime of the component. Although it's known for a fact the label is a subject to change only during design time (and at runtime it's constant), we have no means of expressing that.

This problem can be treated in a different way. Difficulties in deep adjustment of a component to the context of its use (the design/runtime opposition is just an example) are rooted in the fact that essentially we're dealing with the need of defining **a new type** of data. And since any activities concerning a component implying the use of its functionality and therefore its execution, the process of defining a new type must take place at runtime (and without recompiling a program).

There are several bypass routes such as source code, bytecode or binary code generation, runtime compiler calls etc. This routes, however aren't always an option (in embedded systems, for example, the compiler is usually unavailable). That's why a system capable of building new types from user-configured components without stepping over the bounds of its component model is of great interest.

III. Data organization and control flow management

Let us concisely consider the main aspects of existing object-oriented programming languages and component technologies. Analysis of their advantages and limitations has defined the main features of the suggested component model described in the next section.

A comprehensive consideration of specific languages and technologies, however, is beyond the scope of this work (see [10] for such review). We base on a generalized object-oriented concept, engaging other technologies when it's necessary since the requirement of runtime data type definition leads to a component technology to which designtime and runtime aren't clearly distinguished.

Any development and execution environment can be looked at from the two points of view, viz. the principles and mechanisms of data organization and control flow management. From the first point of view, one of the greatest strengths of the object-oriented paradigm is the hierarchical data organization. It's a well-tried remedy for handling a growing complexity of software systems, so the suggested component model is made to be capable of combining components into interacting groups, constituting new components (strictly speaking, combined into groups are *types of components,* and only then composite components are created, see sec. IV).

From the managing control flow point of view, object-oriented approach provides us with methods. We believe the concept of method to be too flexible and complex, which overcomplicates the structure of the programs using it. A method may have a return value, or it may return nothing. The return value itself may be a reference to some internal class member, or it may be a defensive copy. A method is allowed to have an arbitrary number of parameters of arbitrary types. This list can be continued (consider overloading and overriding, for example). The concept of method is not specific enough to allow the desired formalization of object interaction.

The concept of property presented by some frameworks (C#, JavaBeans) is more apposite. It combines features of both object field (well-defined type and a fixed set of operations: reading, writing) and method (behind access operations a programmer-defined functionality can be hidden). The property-based component interaction model is simple and formal because of the limited number of characteristics of the "property" concept. However, properties (the way they are implemented in C#, for example) can't compete with methods in capabilities. If a property can only be accessed for read/write operations it's impossible to efficiently implement well-known callback mechanism. An operation of **binding** is needed (see below).

Following the two key principles we've pointed out (hierarchical organization of components and property-based interaction) is, in our view, necessary to create a viable component model. Along with the runtime type definition requirement, these principles form the basis of the suggested solution, which we will now describe.

IV. Model

Let us describe the component model that will allow one to keep the strengths on the existing technologies and models, while eliminating the drawbacks mentioned above.

A. Component

The main idea of component-based software development is the distinction between an interface and an implementation, which lets one to implement and use a component separately. By component we mean a "black box", i.e. data and functionality hidden behind an interface. Therefore component is described by its interface, implementation and state (data incorporated in it and changed during its lifetime).

The interface of the component is described by the set of its properties (named attributes with a given value type and access permissions). **The implementation** defines a behavior of the component and is different for **primitive, composite** and **compiled** components

Interface

The unit cell of the interface part of the component is its **property,** an entity with a specified type that represents some aspect or attribute of the component. Some of the three operations may be applicable to the property: reading, writing and binding. The applicability is governed by the access permissions (defined in the component type, subsec. IV.B).

The value of the property (when it can be read) is defined by the internal state of the component, which is changed by the property writing operations. The operation of binding property *A* to property *B* (*A* and *B* may be owned by different components) allows the owner of *B* to receive notifications of change of *A*'s value in form of the new value being written to *B*.

Implementation

Sticking to the idea of hierarchical organization of components, we distinguish three kinds of them: **primitive, composite** and **compiled.**

Primitive components are similar to variables of primitive types in programming languages. They hold data of most common types (numbers, characters, text string, logical values etc.), are indivisible from the model's point of view and don't have any properties. Primitive components are so-called value-variables, whose values are given at initialization and are immutable. The prime characteristic of such component is the distinction of its identity.

Compiled components are implemented with use of off-

site means (i.e. outside the component model). Having this kind of components around is required to integrate thirds-party technologies (e.g. JavaBeans). Both compiled and primitive components are not introspected by the model. The main differences between them are that the latter is an immutable value-variable without properties and that the former has a default state (i.e. it can be created without any context, while a primitive component requires at least it's initial value to be given during its creation).

Composite component is a set of other components (called supercomponent and subcomponents respectively) interconnected by **event connections** and **shared properties.** An event connection is created when two properties are bound (see above) to each other. A shared property connection implies that a subcomponent uses its supercomponent's property (of the same type) instead of creating a new property of its own. The reference to the shared property is provided to the subcomponent by the supercomponent at initialization time (subsec. IV.B describes the implementation of this mechanism). Since the same property may be shared by several properties of subcomponents, their interaction is flexibly adjustable.

This way of "projecting" interface onto its implementation serves an apposite compromise between a trivial (giving access to an internal variable) and a too complex, model-breaking (writing in a programming language) approaches to implementing properties' functionality.

Since subcomponents are allowed to be composite components, the hierarchy can have an arbitrary depth (limited only by available hardware). Leafs of the tree are primitive (and maybe compiled) components.

Let us now describe component types and mechanisms of their creation.

B. Component type

A component type is a named entity that specifies the way the component of this type can be built. The concept of type is similar to that of class in object-oriented languages. The type describes the interface, the implementation and their interconnection for the components of this type. The interface part is comprised of **property descriptors,** each specifying a name, a type, access permissions and a default value for the property of the future component. The implementation part is different for primitive, compiled and composed types (corresponding to the three kinds of components described above). For primitive type, the implementation is a variable holding the current value of the component. For the compiled type, it's the instructions for obtaining an implementation of the component and connecting it to the interface. To create a JavaBeans component, for example, one has to provide the name of the corresponding java class (the rest is done by the Java reflection mechanism [5]).

A composite type includes a set of **subcomponent descriptors,** each specifying a type of the corresponding subcomponent of the future component and its initial value. This information is completed with connection specifications. For every property of every subcomponent it's specified which property of the supercomponent it shares (if sharing takes place). The list of necessary event connections is also stored within the type.

Let us consider the component construction process. First, references to properties are set up to point to either preliminary created properties or the shared properties of supercomponent accordingly to property descriptors. Second, subcomponents are created (one for each subcomponent descriptor) and the references to shared properties (if any) are given to them. Third, necessary event connections are established.

After a certain type was instantiated, the resulting component has all the corresponding properties and subcomponents. And all the restrictions that are being complied with during its functioning (type and access control etc.) are governed by the metainfo of the type.

C. Runtime type definition

Let us consider a process of defining a new type at runtime for which the model suggests the following course of action.

First of all, the user chooses a component from a type library (that contains predefined primitive types along with composite types defined earlier) and creates **a type editor** – a special entity that holds functionality for defining new types from existing ones. It receives the chosen type as its input data and allows performing actions listed below.

Second of all, the user is allowed to configure the resulting structure. He can change property descriptors (names, default values and access permissions) as well as implementation metadata: add or remove subcomponents, event connections and shared properties. It's important to notice that instances of a newly defined type adjust deeply to its requirements. For instance, changing a property from being random-accessed to read-only switches the underlying implementation from a variable to a constant. As another example, making a property unbindable entirely removes change listener-related functionality from the component.

Finally, when the necessary modifications are done, the newly created composite type can be added to the type library and be used on equal terms with other types.

The type editor also allows wrapping an arbitrary structure into a composite component. This is important at the beginning of development process when there're only primitive types in the library.

The type editor thereby provides access to the internal data of the type and allows modifying its **copy** in order to be able to create modified versions of types that have instances in the system without tracking changes in every type and spreading them to its instances. This makes the

implementation of the type editor relatively simple and we won't concentrate on the question in this work.

V. Conclusion

We have described the core principles of the new component architecture that extends capabilities of existing component models. We have tried to demonstrate their flexibility and versatility. We believe that applications following the architecture will be able to extend, evolve and adapt to the changing requirements faster and more actively than the traditional software.

References

[1] Bruneton, E., Coupaye, T., Stefani, J.B., *The Fractal Component Model specification. Version 2.0-3*, The ObjectWeb Consortium, 2004.

[2] Costa Seco, J., Silva, R., Piriquito, M., "ComponentJ: A Component-Based Programming Language with Dynamic Reconfiguration", *Computer Science and Information System*, ComSIS Consortium, Novi Sad, Serbia, 2008, pp. 63-86.

[3] Dos Reis, G., Järvi, J., "What is Generic Programming?" *Library-Centric Software Design*, Montréal, Québec, Canada, 2005, pp. 1-11.

[4] Gamma, E., Helm, R., Johnson, R., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1994.

[5] Gosling, J., Joy, B., Steele, G., *The Java™ Language Specification. 3rd ed.*, Addison Wesley, 2005.

[6] Krueger, C.W., "Software reuse", *ACM Comput. Surv. Vol. 2,* ACM, New York, 1992, pp. 131-183.

[7] McIlroy, M.D., "Mass produced software components", *Naur P., Randell B., "Software Engineering, Report on a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7th to 11th October 1968"*, Scientific Affairs Division, NATO, Brussels, 1969, pp. 138-155.

[8] Object Management Group, *The Common Object Request Broker: Architecture and Specification. Version 3.1. Part 3 - Components*, OMG document formal/2008-01-08, 2008.

[9] Redmond, F.E., *DCOM: Microsoft Distributed Component Object Model*, IDG Books Worldwide, Inc., Foster City, 1997.

[10] Stiemerling, O., *Component-Based Tailorability*, Bonn University, Bonn, 2000.

[11] Sun Microsystems Inc. *The JavaBeans™ API specification. Version 1.01-A*, Sun Microsystems Inc., 1997.

Security Risks and Modern Cyber Security Technologies for Corporate Networks

Alexander Adamov, Vladimir Hahanov, IEEE Computer Society Golden Core Member, Wajeb Gharibi

***Abstract*—The article aims to highlight current trends on the market of corporate antivirus solutions. Brief overview of modern security threats that can destroy IT environment is provided as well as a typical structure and features of antivirus suits for corporate users presented on the market. The general requirements for corporate products are determined according to the last report from av-comparatives.org [1]. The detailed analysis of new features is provided based on an overview of products available on the market nowadays. At the end, an enumeration of modern trends in antivirus industry for corporate users completes this article. Finally, the main goal of this article is to stress an attention about new trends suggested by AV vendors in their solutions in order to protect customers against newest security threats.**

***Index Terms*—Antivirus technologies, corporate security, corporate network, malicious software, protection, threats, trojan.**

I. Introduction

MOST companies think of defeating itself against potential security attacks, but only a few of them really imagine a set of security threats that can danger the company. Many of them described in corporate in security standards thus helping the companies to organize IT security defense system. In such context antivirus protection plays the vital part of whole security area. Moreover contemporary antivirus solutions become more advanced and mature. Nowadays they include not only antivirus engine for workstations and an administration console, but many additional features, like antivirus for a mail protection system, a gateway, a database of incidents and enhanced report and logging system. Nonetheless, an implementation of many of such solutions is far from solving all corporate security issues. That is why it is not enough to install only personal antivirus products within a corporate network, but whole corporate suite to cope with all threats at different levels of a network. This will help to construct a corporate secure IT environment.

Manuscript received August 18, 2010. This work was supported in part by the Kingdom of Saudi Arabia under CoEIA Project.

V.I. Hahanov is with Kharkov National University of Radio Electronics, Dean of Computer Engineering Faculty, Kharkov, 61166 Ukraine (phone: (+380)-50-2782198; fax: (+380)-57-7021326; e-mail: hahanov@kture.kharkov.ua).

A.S. Adamov is with Kharkov National University of Radio Electronics, Kharkov, 61166 Ukraine (e-mail: alexander.adamov@ dnt-lab.com).

Wajeb Gharibi is with Jazan University, P. O. Box 4425 Arrawabi, Unit #1, Jazan 82822-6694, KSA. Mobile: +966 508 2232 64, E-mail: gharibi@jazanu.edu.sa, gharibiw@hotmail.comz

II. The Risk of Malware and Internet Threats

The main risks for companies in area of information security comprise infections by viruses, trojans, worms, exploits and other malicious code that can reveal the corporate secrets by stealing confidential data and be the reason of serious data leakage. Also phishing and online banking fraud can be a serious problem for IS managers.

Taking in consideration that corporate IT infrastructure mainly consists of domain-joined computers it can be more likely to encounter worms. The main propagation vectors of worms are opened file shares, removable drives, e-mail and IM channels. These are commonly used within companies' networks as a corporate communication and can be a potential threat. According to Microsoft Security Intelligence Report [13] 4 of the top 10 malware families detected on domain-joined computers are worms.

The most popular families are Autorun worms that can spread through removable drives, and network worm Kido/Kido/Conficker/Downadup which was appeared on November 2008 and caused a global world epidemic. The worm has struck more than 10 million computers, using vulnerability in service "Server" (MS08-067).

The worm sent to the remote machine specially crafted RPC-request on TCP port 445 (MICROSOFT_DS|SMB) which caused the buffer overflow by calling wcscpy_s() function in NetpwPathCanonicalize() (library netapi32.dll). The given malicious program applied a wide spectrum of methods to hide the presence in the system: files view settings in Explorer, disabling the services, responsible for system security. It was used several ways of distribution: the admin shared folders, removable devices, downloading the updates from websites, domain addresses of which were generated by special algorithm. As a result it has received a wide proliferation all over the Internet. The detailed

description of the worm you can find in malware encyclopedia [14].

III. Security Risks in Hardware

A. Overview

Recently Dell Company, the leading computer system manufacturer, announced that in its servers' line PowerEdge malicious program has been found embedded in a flash memory of a motherboard [15].

Thus, the computer industry has been faced with the threat of computers' infection with malicious software, but at the level of firmware. The topic of malicious inclusions in hardware is becoming more importance due to the fact that most of our systems on chips are fabricated in Southeast Asia, although under the brand names of major U.S. companies. This can be explained by reducing production costs and increasing market competitiveness. Another side of a coin is losing a trust during a fabricating process. Especially, when it comes to development for military purposes, which may result in decommissioning weapon systems.

A model of compromised system is represented in Fig. 1.

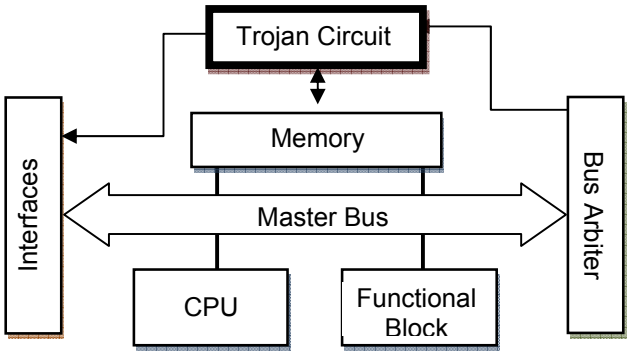


Fig. 1. Trojan insertion embedded in system on a chip

The trojan can be activated by a special value on Master Bus, for instance, it can be memory address where stored targeted data. Once trojan circuit is triggered, the payload can be one of the following: disabling system, transmitting interested data to third party by means of embedded interfaces, collecting accessed information in the memory for further utilization, rising security privileges for a current process running in the system.

B. A Formal Model of Hardware Trojan (HT)

Let us consider a formal model of HT by introducing several abstract concepts. *Trojan* (T_i) is a malicious component that can provide an access to *System* (S_i) in certain moment with the appropriate condition.

The pairs (T_i, O_i) are bound by the set of specified actions A_s. This set is defined according to security policy and specification of the vendor and is a subset of the whole set A of all possible actions for each pair.

At the same time, pairs (T_i, O_i) can communicate by a set of malicious actions A_m. It is obvious that $A = A_s \bigcup A_m$. The purpose of security verification is identifying actions from the set A_m.

The task of malicious circuit detection is getting more complicated when HT can take advantage of a set of specified actions, that can gain an access to a computer system or its component, from the set A_s, such as $A_s \bigcap A_m \neq \varnothing$. As a result, it is needed to verify system considering a whole set of actions A. It is a hard verification task even for small systems on a chip because of searching within a set of all possible input vectors.

C. Hardware Trojan Detection Task

The danger hidden in complex system on a chip nowadays is underestimated. The trojan circuit can be easily embedded to a system on a chip and hardly detected taking in consideration the size of the modern digital system [18].

The formal view to the problem of malicious insertions proves that the task of trojan detection in complex digital system is difficult.

The solution can be found in the area of high level testing methodology in order to cope with the complexity of the task. Nowadays there are powerful methods that are provided by researchers that can help in trojan detection and analysis, such as in [19] and [20], but still there is no mature solution that can provide universal methodology for fables companies and governments.

IV. Assessing the Losses of the Company From Security Threats

The breaches in corporate environment may cause undesirable data leakage and will lead to suspending business processes of the company. In such scenario it may lose important customers and business partners because company which cannot protect itself from this attacks is faithless over the unforeseen costs like malicious programs influence, information drain, attacks on computer networks, etc [16].

The result is that when the number of personal computers is growing and communication channels capacity is increasing malware epidemic's scope and losses are growing correspondingly. Therefore the company management has to think about the information security.

In modern world the probability of malicious programs get into a computer system is constantly growing. It may cause not only short-term fault in the network, but a complete stopping the company. Losses by malicious programs are estimated as billions of dollars around the world annually and continue to increase.

According to [17] the cost of the average caused by malware attack in a corporate network can be calculated as in:

$$\begin{aligned} DELAY = &(comp\_num \times fix\_time \times adjuster\_hour\_payment) \\ &+ additional\_expenses + \\ &+ \left(\frac{items\_day \times product\_price \times fix\_time \times comp\_num}{8 \times adjuster\_num} \right) + \\ &+ \left(\frac{salary \times comp\_num \times fix\_time}{8 \times 22 \times adjuster\_num} \right), \end{aligned} \quad (1)$$

where *comp_num* – nmber of computers within a network; *fix_time* – time in hours for fixing a fault; *adjuster_hour_payment* – payment for adjusting a computer per hour; *adjuster_num* – number of such specialists; *additional_expenses* – additional expenses for network repairing and buying new devices; *product_price* – price of a product; *items_day* – number of product items per day; *salary* – salary of an employee per month.

V. Analysis of Corporate Antiviruses

According to latest report from Av-Comparatives Lab [1] the main players at corporate security market are Avira, Eset, G Data, Kaspersky, Sophos, Symantec. In this article we will overview functional diversity of existed corporate suits and take a look to nearest future of corporate antivirus suits which seem to become a total security solution for corporate users.

The typical structure of corporate suite:

1) Administration console – provides useful managing and configuration environment for administrators of big networks.
2) Antivirus for workstation – actually the antivirus engine with all features peculiar to workstation antivirus. Provides centralized protection of user's system on a corporate network against all types of malware, network attacks, spam.
3) Mail server antivirus – protects the mail server against spam and malware delivered by email channels.
4) File server antivirus –protects data on servers under Microsoft Windows operation system control against all types of malware. Designed mostly for high-performance corporate servers.

Analyzing all products options it has been distinguished the main features of modern corporate antivirus:

1) Easy Installation and Deployment – simple and fast way to deploy the solution into a big corporate network, supporting Active Directory technology.
2) Usability and Management – console provides useful management interface with real-time monitoring and logging features.
3) Scalability – solution works with networks of different size from small business to enterprise scale with thousands of computers distributed geographically around many offices.
4) Technical Support and Updates – regularly delivers antivirus updates and helps to solve all unforeseen security issues of a company with a short response time. Also website and online services are important points.
5) Cross-Platform Security – ability to protect systems with different types of operational systems, such as Linux, MacOS, mobile platforms, etc.

VI. Nowadays Trends and Suggestions

As for future in area of corporate users' security protection a growing trend is including more sophisticated administration interface that provides detailed information about the real-time status of the network. It can be represented as advanced graphical interface with diagrams or even as a separate product. It can be an intelligent agent that can handle huge amount of information from thousands of computers and hints the administrator what to do in that case.

For instance, Blue Medora designed a special agent for Symantec corporate solution which results in "less complexity, more uniform operations management, and a significant reduction in costs due to the elimination of redundant infrastructure and multiple platform-specific tools" [2]. It proves the idea that there is an area for further improvement of corporate antivirus solution even for the outstanding vendor.

Among extensible features are the following:

1) Improved monitoring of incidents with malware.
2) Improved monitoring of the user's intrusion into the antivirus key processes.
3) Monitoring of failures in updates and malware scanning tasks.

The new features to be included into the product:

1) Real-time status and availability monitor.
2) Log monitors.
3) Report and take-action system that would help administrator to perform necessary actions to any type of threats.

The main idea is to raise a sensitivity level of the persons who are responsible for corporate network security and reduce the time of reaction to the emerging danger. Therefore, useful and exhausted data representation can really help in struggling against malware.

Except logging and monitoring an essential part of security solution is integrity. Modern corporate antivirus solutions comprise not only a bunch - Antivirus, Antispyware, Firewall, Antispam with Managing System, but many additional features, such as Backup systems, Password and Key Managers and Encryption Utilities to organize safe confidential data storage.

This trend is peculiar to home solutions as well. Thus, Kaspersky Pure for home users provides besides malware protection also password management system to keep in safe all family's identities [3]. Another example, Norton

Online Family also allows observing the kids activity on computer [4].

As for enterprise suits, Symantec provides Protection Suite for Endpoints where gathered encryption, confidential data storage and others features aimed to maintain IT security in a company [5]. One more interesting product came from Sophos [6]. Endpoint Security and Data Protection has Integrated DLP (Data Loss Prevention) and Encryption tools in its package.

Also mobile and non-Windows platforms should be supported within a corporate solution because of huge diversity of working devices: laptops, PDAs, smart phones, etc. Many antivirus vendors have such solutions in a product line.

The important point to be considered is Security-as-a-Service. A security is not only software, but a state of a system. It is important to have 24/7 technical support service to solve a newest security issues, such as new versions of malware, zero-day exploits. Often proactive defense cannot cope with a huge variety of new malware modification released every day by hacker's generators. The same way administrator cannot keep all software up-to-date with new patches installed. In such context deploying vulnerability searching system is desirable to reveal software breaches and notify to install new updates in time.

Here the problem of support service's quality has been raised. It is not a secret that a high quality support service can be granted only by the team of qualified malware experts not by "sandbox" robots [here we can put a reference to our research in "Sandbox Comparatives"]. Many companies provide malware analysis column on their web sites or even separate security domains where the descriptions of most popular threats are published, like it is done at virusradar.com by Eset and securelist.com by Kaspersky Lab.

Another side of the coin is an ability to remove consequences of an infection. Not all antivirus engines allow proper disinfection of the system or network after an incident that already has taken a place. In that case special removal utilities and scripts are released by analysts to help administrators in cleaning their IT farms. There are such services from Symantec [7] and AVZ tool from Kaspersky Lab [8].

Phishing is becoming a serious problem for all users in the cyber world. What antivirus vendors can suggest in protecting corporate users against this problem except of standard anti-phishing modules that block dangerous web sites from black list? The interesting solutions have been introduced within Kaspersky Internet Security 2011 – Geo Filter and Online Banking modules. According to information from official site: "Geo Filter provides the user with an option to block domains related to specific countries. Online Banking controls requests to Online Banking services while processing confidential data" [9]. Those modules could be helpful in keeping a communication with financial institutions more safe which could be essential in corporate environment.

Finally, a following to corporate security standards is what some AV vendors do. The big companies try to organize corporate IT security according to policies compliant to security standards. Among them:

1) X509 – is ITU-T standard specifies formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm [10],
2) LDAP (Lightweight Directory Access Protocol) – is an application protocol for querying and modifying data using directory services running over TCP/IP [11],
3) Microsoft IWA (Integrated Windows Authentication) – provides authentication connections between Microsoft IIS, Internet Explorer, and other Active Directory aware applications [12].

VII. Conclusion

To sum up, in this short review the current security threats have been briefly presented. According to them an analysis of antivirus solution for corporate users was proposed. The general features and structure of corporate suit were enumerated based on the latest report from av-comparatives.org. In the last part of the article we considered modern trends in current antivirus solutions from most popular AV vendors, such as Eset, Symantec, Sophos and Kaspersky.

It is obvious that the corporate products represent quite powerful solutions for enterprise networks but they could become better by adopting new standards, technologies and a high level of support services. The corporate suit is becoming a heavy package of tools aimed to fight against malware, network attacks, spam, phishing. It gives to administrators a control under a huge corporate network that allows monitoring a real-time activity and react to an existing situation as soon as possible. A corporate security is a multifactor system that consists of security software, services, policies and a human factor. None of them should be missed in a building process of secure corporate environment.

References

[1] "Review of IT Security Suites for Corporate Users", May 2009. Available: www.av-comparatives.org.

[2] Blue Medore Agent for Symantec Endpoint Protection, Available: http://www.bluemedora.com/product/page/40

[3] "Kaspersky Pure. Ultimate Protection for Your Digital Life", Available: http://www.kaspersky.com/kaspersky-pure

[4] "Norton™ Online Family. A smarter way to keep your kids safe online". Available: https://onlinefamily.norton.com/familysafety/loginStart.fs

[5] "Symantec Protection Suite Enterprise Edition for Endpoints". Available: http://www.symantec.com/business/protection-suite-enterprise-edition-for-endpoints

[6] Sophos, "Endpoint Security and Data Protection". Available: http://www.sophos.com/products/enterprise/endpoint/security-and-control/

[7] Symantec, “Spyware and Virus Removal”. Available: http://www.symantec.com/norton/nortonlive/spyware-virus-removal.jsp
[8] “How to scan your computer, save the log and run a script using the AVZ utility? “. Available: http://support.kaspersky.com/faq/?qid=208279710
[9] Kaspersky Internet Security 2011 Manual. Available: http://support.kaspersky.com/kis2011?level=2
[10] Wikipedia, http://en.wikipedia.org/wiki/X.509
[11] Wikipedia, http://en.wikipedia.org/wiki/LDAP
[12] Wikipedia, http://en.wikipedia.org/wiki/Integrated_Windows_Authentication
[13] Microsoft Security Intelligence Report. Volume8. July-Dec2009, Available: http://www.microsoft.com/security/about/sir.aspx
[14] Malware Encyclopedia. Available: http://www.totalmalwareinfo.com/eng/Net-Worm.Win32.Kido_/_Conficker.A-C_Worm
[15] “PC giant warns of hardware trojan“, NewScientist, 22 July 2010.
[16] Filatov, Kozlovskih, Cvetkova, Planning, Finance, Management of Enterprise. – Finance and Statistics, 2005, 384 p.
[17] M.V. Bocharinkova, A. S. Saprykin, V.A. Kiktenko, A. S. Adamov, “Developing methods to assess damage from the spread of malware at enterprises”, IT-Security Conference for New Generation, Moscow, Russia, 28-29 April 2009.
[18] X. Wang, M. Tehranipoor and J. Plusquellic, "Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions", International Workshop on Hardware Oriented Security and Trust, 2008, pp. 15-22.
[19] I. Verbauwhede, P. Schaumont, "Design methods for Security and Trust", DATE'07, 2007.
[20] F. Wolff, C. Papachristou, S. Bhunia, R. S. Chakraborty, "Towards Trojan-Free Trusted ICs: Problem Analysis and Detection Scheme", DATE'08, 2008.

Alexander Adamov was born in Kharkov, Ukraine on August, 05, 1984. Master of Science in Computer Engineering, Kharkov, Ukraine, Kharkov National University of Radio Electronics in 2006. Master of Philosophy in Data Mining and Knowledge Discovery, University of Wales, UK in 2008. PhD study in Kharkov National University of Radio Electronics from 2006 to 2009. The major field of study is information security in hardware.

He worked as Software Developer in Aldec Inc., Security Analyst in Design and Test Lab, Ltd. The current job is Manager of Malware Research Lab in Design and Test Lab, Ltd., Kharkov, Ukraine. Lecturer and Trainer in Kharkov National University of Radio Electronics. Last publication is “Security Risks in Hardware: Implementation and Detection Problem”, EWDTS’2010, St. Petersburg, Russia, 17-20 Sep 2010. His research interests are information security, computer threats, digital system verification and education.

Vladimir Hahanov, Dr. of Sc., Professor, Dean of the Computer Engineering Faculty (1650 students, 102 professors and scientists), Kharkov National University of Radio Electronics. Work Experience: Scientific advisor of 2 doctors of science and 20 PhD’s; General Chair of the International symposium “IEEE East-West Design and Test”. Program committee member of 20 International Conferences in area of HW/SW Design & Test. Published 559 papers and 7 books. IEEE Computer Society Golden Core member. Member of IEEE Computer Society since 2001. 3 Outstanding contribution awards from IEEE Computer Society President.

Wajeb Gharibi, Dr., Associate Professor, Coordinator, Dept. of Computer Networks, College of Computer Science & Information Systems, Jazan University, P. O. Box 4425 Arrawabi, Unit #1, Jazan 82822-6694, KSA. Mobile: +966 508 2232 64, E-mail: gharibi@jazanu.edu.sa, gharibiw@hotmail.comz

Models and Methods for Verification and Diagnosis of SoC HDL-code

Vladimir Hahanov, *Senior Member, IEEE* , Wajeb Gharibi, Eugenia Litvinova, *Member, IEEE*,
Svetlana Chumachenko, *Member, IEEE*

Abstract — **Xor-metrix for object relations in a vector logic space and a structural testing model are proposed. Assertion-based models and methods for the verification and diagnosis of HDL-code functional failures, which make possible to reduce considerably time-to-market of software and hardware, are developed. An architectural model of multimatrix reduced logical instruction set processor for embedded diagnosing is offered.**

I. Introduction

Recent trends in creating new communications, computing and information services, useful to the human, are development of dedicated gadgets, which have important advantages over PCs and laptops: power consumption, compactness, weight, cost, functionality, and friendliness of interface. Practically the top ten dedicated products 2010 (Apple iPad, Samsung Galaxy S, Apple MacBook Air, Logitech Revue, Google Nexus One (HTC Desire), Apple iPhone 4, Apple TV, Toshiba Libretto W100, Microsoft Kinect, Nook Color) is realized as digital systems-on-chips. By 2012 the mobile and wireless communication market will move to 20 nm (results of the January 2011 Technology Forum of Common Platform Alliance). Further development of the technologies by year: 2014 – 14 nm, 2016 – 11 nm. In 2015 more than 55% of mobile phones will be smartphones, tablet PCs will replace laptops and netbooks. Superfones (Nexus-1, Google) will unite all devices and services. The transition from the computing platform to mobile devices with small size results in considerable reduction in power consumption worldwide. The next computerization wave, entitled "Internet of things", is being accelerated. It will lead to widespread sensor networks, including their integration into the human body. The world market of the above devices and gadgets today involves about 3 billion products. For their effective designing, manufacturing and exploitation the new technologies and Infrastructures IP are created. One of the possible steps in this direction is represented below in the form of verification technology T^v: M^t is metrics and model for testing, H^c is HDL-code of a design, G^t is synthesis of software transaction graph, $\{M^f, M^s\}$ determine creating two verification models for HDL-code (functional failure table and software activation matrix), $\{D^c, D^r, D^m\}$ determine developing three methods for diagnosing the functional failures (for analyzing rows, columns and whole matrix), which use the assertion engine (assertion is a logical statement for detecting the semantic errors in software), P^m is architecture of multimatrix processor for parallel analyzing tabular data, R is implementation of models, methods and tools in the system Riviera, Aldec Inc.:

$$T^v = M^t \to H^c \to G^t \to \begin{Bmatrix} M^f \to \begin{Bmatrix} D^c \\ D^r \end{Bmatrix} \\ M^s \to D^m \end{Bmatrix} \to P^m \to R.$$

The objective of the research is to reduce time-to-market and improve the quality of digital systems-on-chips by developing the assertion-based infrastructure, models and methods for verification and diagnosis HDL-code. The information, needed for detecting failures at the functional blocks, is formed during simulation (execution) of software code. Design effectiveness for digital product is determined as the average and normalized in the range [0,1] integral criterion:

$$E = F(L,T,H) = \min[\frac{1}{3}(L+T+H)],\ Y = (1-P)^n;$$

$$L = 1 - Y^{(1-k)} = 1 - (1-P)^{n(1-k)};$$

$$T = [(1-k) \times H^s]/(H^s + H^a);\ H = H^a/(H^s + H^a).$$

The criterion takes into account the following: the error level L, the verification time T, software-hardware redundancy, determined by the assertion engine and Infrastructure IP tools H. The parameter L, as a complement of the parameter Y (yield), depends on the testability k of a

Manuscript received December 3, 2010.

Vladimir Hahanov is with the Kharkov National University of Radioelectronics, Ukraine, 61166, Kharkov, Lenin Prosp., 14, room 321 (corresponding author to provide phone: (057)7021326; fax: (057)7021326; e-mail: hahanov@ kture.kharkov.ua).

Wajeb Gharibi is with Jazan University, P. O. Box 4425 Arrawabi, Unit #1, Jazan 82822-6694, KSA. Mobile: +966 508 2232 64, E-mail: gharibi@jazanu.edu.sa, gharibiw@hotmail.comz

Eugenia Litvinova is with the Kharkov National University of Radioelectronics, Ukraine, 61166, Kharkov, Lenin Prosp., 14, room 378 (phone: (057)7021326; fax: (057)7021326; e-mail: ri@ kture.kharkov.ua).

Chumachenko Svetlana is with the Kharkov National University of Radioelectronics, Ukraine, 61166, Kharkov, Lenin Prosp., 14, room 321 (phone: (057)7021326; fax: (057)7021326; e-mail: ri@ kture.kharkov.ua).

design, the probability P of existence of faulty components, and the quantity of undetected errors n. The time of verification is determined by the testability of a design k [3,4], multiplied by the structural complexity of hardware-software functionality, divided by the total complexity of a design in code lines. The software-hardware redundancy depends on the complexity of assertion code and other costs, divided by the total design complexity. At that software or hardware redundancy has to provide the specified diagnosis depth for functional errors and time-to-market, defined by customer.

The problems are: 1) Creation of a metrics and structural-analytical model for testing digital systems-on-chips. 2) Improvement of the models and methods for detecting functional failures, based on assertion engine, to increase the speed of HDL-code verification and diagnosis. 3) Development of the architectural model of multimatrix processor for diagnosing.

References are: 1. Models of the problems for technical diagnosis are presented in [1-6]. 2. Diagnosis and verification of digital systems-on-chips are described in [9-17, 22-15]. 3. Hardware and matrix processors for increasing the speed of testing are proposed in [18-21].

II. A MODEL FOR TESTING AND VERIFICATION

The effective process models and methods for diagnosing the functional failures in software and/or hardware are offered. The register or matrix (tabular) data structures, focused to parallel execution of logic operations, are used for detecting the faulty components.

The problem of synthesis or analysis of system components can be formulated in the form of interaction (symmetrical difference is an analog of xor-operation on the Boolean) of its model F, input stimuli T and responses L in a cybernetic space:

$$f(F,T,L)=\varnothing \rightarrow F\Delta T\Delta L=\varnothing .$$

A cyberspace is a set of information processes and occurrences, which use computer systems and networks as a carrier. Particularly, a space component is represented by k-dimensional (tuple) vector $a=(a_1,a_2,...,a_j,...,a_k)$, $a_j=\{0,1\}$ in a binary alphabet. Zero-vector is k-dimensional tuple, all coordinates of which are equal to zero: $a_j=0,\ j=\overline{1,k}$.

Metrics β of cybernetic (binary) space is defined by a single equality that forms zero-vector for xor-sum of the distances d_i between nonzero and finite quantity of points, closed in a cycle:

$$\beta=\bigoplus_{i=1}^{n} d_i=0.$$

The Hamming distance between two objects (vectors) a and b is determined as derived vector: $d_i=d(a,b)=a_j\bigoplus_{j=1}^{k} b_j$. Otherwise: the metrics β of a vector logic binary space is xor-sum of the distances (it is equal to zero) between finite quantity of graph points (nodes), closed in a cycle. The sum of n-dimensional binary vectors, specifying the coordinates of cycle points, is equal to zero-vector. This metrics definition uses relations that allow reducing the axiom system from three up to one and extending it on any constructions of n-dimensional cyberspace. The classical metrics definition for determining interaction of one, two and three points in vector logic space is a particular case of β-metrics when $i=1,2,3$ respectively:

$$M=\begin{cases} d_1=0\leftrightarrow a=b; \\ d_1\oplus d_2=0\leftrightarrow d(a,b)=d(b,a); \\ d_1\oplus d_2\oplus d_3=0\leftrightarrow d(a,b)\oplus d(b,c)=d(a,c). \end{cases}$$

The metrics β of cybernetic multiple-valued space, where each coordinate of vector (object) is determined in the alphabet that is the Boolean on universe of primitives by the power p: $a_j=\{\alpha_1,\alpha_2,...,\alpha_r,...,\alpha_m\}, m=2^p$, is the symmetric difference (it is equal to $\varnothing$-vector by all coordinates) of the distances between finite quantity of points, closed in a cycle:

$$\beta=\mathop{\Delta}_{i=1}^{n} d_i=\varnothing . \qquad (1)$$

Equality empty vector the symmetric difference of coordinatewise set-theory interaction (1) emphasizes the equivalence of the components (distances), which form the equation with a single coordinate operation $d_{i,j}\Delta d_{i+1,j}$, used, for instance, in four-digit Cantor's model. It is defined by the corresponding Δ-table:

Δ	0	1	x	$\varnothing$
0	$\varnothing$	x	1	0
1	x	$\varnothing$	0	1
x	1	0	$\varnothing$	x
$\varnothing$	0	1	x	$\varnothing$

$\cap$	0	1	x	$\varnothing$
0	0	$\varnothing$	0	$\varnothing$
1	$\varnothing$	1	1	$\varnothing$
x	0	1	x	$\varnothing$
$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

$\cup$	0	1	x	$\varnothing$
0	0	x	x	0
1	x	1	x	1
x	x	x	x	x
$\varnothing$	0	1	x	$\varnothing$

a	0	1	x	$\varnothing$
$\tilde{a}$	1	0	$\varnothing$	x

(2)

The truth tables for other basic set-theory operations are represented in (2). A number of primitive symbols, formed closed alphabet relative to the set-theory coordinate operations, can be increased. At that the power of alphabet (Boolean) is determined by the expression $m=2^p$, where p is a number of primitive symbols. This metrics is not only of theoretical interest, but has a practical focus on generalization and classification of technical diagnosis problems by creating a model for xor-relations on the set of four main components. The procedures of test synthesis, fault simulation and detection can be reduced to xor-relations on a full interaction graph (Fig. 1) for four nodes (functionality, unit, test, faults) $G=\{F,U,T,L\}$.

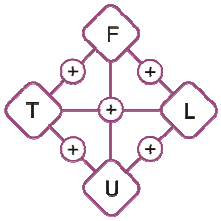


Fig. 1. Graph of interaction between technical diagnosis components

The graph creates four basic triangles, which form 12 triads of relations for the problems of technical diagnosis:

$T\oplus F\oplus L=0$	$T\oplus L\oplus U=0$	$T\oplus F\oplus U=0$	$F\oplus L\oplus U=0$
1) $T=F\oplus L$	4) $T=L\oplus U$	7) $T=F\oplus U$	10) $F=L\oplus U$
2) $F=T\oplus L$	5) $L=T\oplus U$	8) $F=T\oplus U$	11) $L=F\oplus U$
3) $L=T\oplus F$	6) $U=T\oplus L$	9) $U=T\oplus F$	12) $U=F\oplus L$

Insertion of the node U in the graph of interaction between technical diagnosis components extends the functionality of the model; new properties of the resulting system appear. Introduction the new node in the structure has to have strong arguments of its advisability. Concerning the graph, represented in Fig. 1, all problems can be classified into groups as follows.

Group 1 involves the theoretical experiments (on the functionality model), without the device: 1) test synthesis by using the functionality model for a specified fault list; 2) development of the functionality model, based on a given test and fault list; 3) fault simulation for functionality by using given test.

Group 2 – real experiments (by device) without functionality model: 4) test synthesis by physical fault simulation in the device; 5) fault list generation for the device by means of diagnostic experiment; 6) test and faults verification by means of the experiment on a real device.

Group 3 – test experiments (verification) without faults: 7) test synthesis by means of comparing the model simulation results and real device; 8) functionality synthesis by using a real device and a given test; 9) verification of test and functionality model by using the real device with existing faults.

Group 4 – experiments during operation with real inputs: 10) check of correct behavior of a real device on the existing or specified faults; 11) test the device on the existing model in the operation; 12) verification of the functionality and fault list relative to the behavior of a real device.

The most popular problems of the above list are: 1, 3, 5, 8, 9. Another classification of the problem types can be introduced. It allows defining by the graph $G=(F,U,T,L)$ all the conceptual solutions of target problems: test synthesis, functionality model definition, fault model generation and designing of a device:

1) $T=F\oplus L$; 4) $F=T\oplus L$; 7) $L=T\oplus F$; 10) $U=T\oplus L$;
2) $T=U\oplus L$; 5) $F=U\oplus L$; 8) $L=T\oplus U$; 11) $U=T\oplus F$;
3) $T=F\oplus U$; 6) $F=T\oplus U$; 9) $L=F\oplus U$; 12) $U=F\oplus L$.

All constructions, used in a relationship, have the remarkable property of reversibility. Component, calculated using the other two, can be used as an argument to determine any of the two original ones. Thereby, transitive reversibility of each relation triad on complete graph is occur, when by using any two components it is always possible to restore or to determine the third one. At that the format for each component must be identical in structure and dimension (vectors, matrices). Fault diagnosis methods, based on the proposed metrics and testing models, are considered in more detail below.

III. Model for Detecting Functional Failures in Software

The space equation $f(F,T,L,U)=0\rightarrow F\oplus T\oplus L\oplus U=0$ is used. It is transformed to the form $L=(T\oplus F)\oplus(T\oplus U)$. Fault (functional failures) diagnosis is reduced to comparison of simulation $(T\oplus F)$ and full-scale $(T\oplus U)$ results, which generates a functional failure list L, detected in the diagnosed unit. Model-formula for searching the functionally faulty block F_i is reduced to solving by determining xor-interaction between three components:

$$L=F_i\leftarrow[(T\oplus F_i)\overset{p}{\underset{i=1}{\oplus}}(T\oplus U_i)]=0.$$

An analytic model for verification of HDL-code by using temporal assertion engine (additional observation lines) is focused to achievement the specified diagnosis depth and presented as follows:

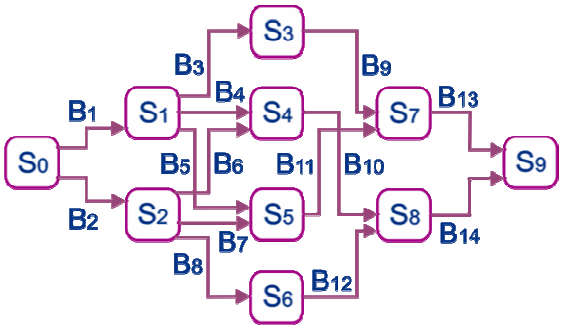
$$\begin{aligned} &M=f(F,A,B,S,T,L), && F=(A*B)\times S;\ S=f(T,B);\\ &A=\{A_1,A_2,...,A_i,...,A_n\};\ && B=\{B_1,B_2,...,B_i,...,B_n\};\\ &S=\{S_1,S_2,...,S_i,...,S_m\};\ && S_i=\{S_{i1},S_{i2},...,S_{ij},...,S_{ip}\};\\ &T=\{T_1,T_2,...,T_i,...,T_k\};\ && L=\{L_1,L_2,...,L_i,...,L_n\}. \end{aligned} \quad (3)$$

Here $F=(A*B)\times S$ is functionality, represented by Code-Flow Transaction Graph – CFTG (Fig. 2); $S=\{S_1,S_2,...,S_i,...,S_m\}$ are nodes or states of software when simulating test segments. Otherwise the graph can be considered as ABC-graph – Assertion Based Coverage Graph. Each state $S_i=\{S_{i1},S_{i2},...,S_{ij},...,S_{ip}\}$ is determined by the values of design essential variables (Boolean, register variables, memory). The oriented graph arcs are represented by a set of software blocks

$$B=(B_1,B_2,...,B_i,...,B_n),\ \overset{n}{\underset{i=1}{\cup}}B_i=B;\ \overset{n}{\underset{i=1}{\cap}}B_i=\varnothing,$$

where the assertion $A_i\in A=\{A_1,A_2,...,A_i,...,A_n\}$ can be put in correspondence to each of them. Each arc B_i – a

sequence of code statements – determines the state of the node $S_i = f(T, B_i)$ depending on the test $T = \{T_1, T_2, ..., T_i, ..., T_k\}$. The assertion monitor, uniting the assertions of node incoming arcs $A(S_i) = A_{i1} \vee A_{i2} \vee ... \vee A_{ij} \vee ... \vee A_{in}$ can be put in correspondence to each node. A node can have more than one incoming (outcoming) arc. A set of functionally faulty blocks is represented by the list $L = \{L_1, L_2, ..., L_i, ..., L_n\}$.



$$B = (B_1B_3B_9 \vee (B_2B_7 \vee B_1B_5)B_{11})B_{13} \vee$$
$$\vee ((B_1B_4 \vee B_2B_6)B_{10} \vee B_2B_8B_{12})B_{14} =$$
$$= B_1B_3B_9B_{13} \vee B_2B_7B_{11}B_{13} \vee B_1B_5B_{11}B_{13} \vee$$
$$\vee B_1B_4B_{10}B_{14} \vee B_2B_6B_{10}B_{14} \vee B_2B_8B_{12}B_{14}.$$

Fig. 2. Example of ABC-graph for HDL-code

The model for HDL-code, represented in the form of ABC-graph, describes not only software structure, but test slices of the functional coverage, generated by using software blocks, incoming to the given node. The last one defines the relation between achieved on the test variable space and potential one, which forms the functional coverage as the power of state i-th graph node $Q = \mathrm{card}C_i^r / \mathrm{card}C_i^p$. In the aggregate all nodes have to be full coverage of the state space of software variables, which determines the test quality, equal to 1 (100%): $Q = \mathrm{card}\bigcup_{i=1}^{m} C_i^r / \mathrm{card}\bigcup_{i=1}^{m} C_i^p = 1$. Furthermore, the assertion engine $< A, C >$ that exists in the graph allows monitoring arcs (code-coverage) $A = \{A_1, A_2, ..., A_i, ..., A_n\}$ and nodes (functional coverage) $C = \{C_1, C_2, ..., C_i, ..., C_m\}$. The assertions on arcs are designed for diagnosis of the functional failures in software blocks. The assertions on graph nodes carry information about the quality of test (assertion) for their improvement or complement. The Code-Flow Transaction Graph makes possible the following: 1) use the testability design to estimate the software quality; 2) estimate the costs for creating tests, diagnosing and correcting the functional failures; 3) optimize test synthesis by means of solving the coverage problem by the minimum set of activated paths of all arcs (nodes). For instance, the minimum test for the above mentioned ABC-graph has six segments, which activate all existent paths:

$$T = S_0S_1S_3S_7S_9 \vee S_0S_1S_4S_8S_9 \vee S_0S_1S_5S_7S_9 \vee$$
$$\vee S_0S_2S_4S_8S_9 \vee S_0S_2S_5S_7S_9 \vee S_0S_2S_6S_8S_9.$$

Tests can be associated with the following program block activization matrix:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}
T_1	1	.	1	.	.	.	.	.	1	.	.	.	1	.
T_2	1	.	.	1	.	.	.	.	.	1	.	.	.	1
T_3	1	.	.	.	1	.	.	.	.	.	1	.	1	.
T_4	.	1	.	.	.	1	.	.	.	1	.	.	.	1
T_5	.	1	.	.	.	.	1	.	.	.	1	.	1	.
T_6	.	1	.	.	.	.	.	1	.	.	.	1	.	1

The activization matrix shows the fact of indistinguishability of the functional failures on a test in the blocks 3 and 9, 8 and 12, which constitute two equivalence classes if there is one assertion (monitor) in the node 9. To resolve this indistinguishability it is necessary to create two additional monitors in the nodes 3 and 6. As a result, three assertions in the nodes $A = (A_3, A_6, A_9)$ allow distinguishing all the blocks of software code. Thus, the graph enables not only to synthesize the optimal test, but also to determine the minimum number of assertion monitors in the nodes to search faulty blocks with a given diagnosis depth.

Increasing the number of assertion monitors leads to modification of an activization table. Otherwise, on a given test and the assertion engine it is necessary to solve uniquely the diagnosis problem for functional failures of the software code with the depth up to a software module. At that the number of assertions and test segments to be minimum acceptable for the code identification of all the blocks:

$$|T| + |A| \geq \log_2 |B| = \mathrm{card}T + \mathrm{card}A \geq \log_2 \mathrm{card}B.$$

Initially, the number of monitors-assertions is equal to the number of test segments. The activization table for software modules makes it possible to identify code blocks with functional failures by the generalized output response vector (assertion monitoring)

$$V = (V_1, V_2, ..., V_i, ..., V_n), V_i = \{0,1\}, V_i = T_i \oplus B_j, \forall j(B_{ij} = 1)$$

The vector coordinate $V_i = T_i \oplus B_j = 1$ identifies the nonpassage of the test segment on a subset of activated modules. In accordance with the vector V, defined on the activization table subject to the above rule for calculating its coordinates:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}	V
T_1	1	.	1	.	.	.	.	.	1	.	.	.	1	.	0
T_2	1	.	.	1	.	.	.	.	.	1	.	.	.	1	1
T_3	1	.	.	.	1	.	.	.	.	.	1	.	1	.	0
T_4	.	1	.	.	.	1	.	.	.	1	.	.	.	1	1
T_5	.	1	.	.	.	.	1	.	.	.	1	.	1	.	0
T_6	.	1	.	.	.	.	.	1	.	.	.	1	.	1	1

a logical function of software functional failures can be constructed, which is simplified using the coordinates of the output response vector V:

$$B=(\overline{T}_1 \vee B_1 \vee B_3 \vee B_9 \vee B_{13}) \wedge (\overline{T}_2 \vee B_1 \vee B_4 \vee B_{10} \vee B_{14}) \wedge$$
$$\wedge (\overline{T}_3 \vee B_1 \vee B_5 \vee B_{11} \vee B_{13}) \wedge (\overline{T}_4 \vee B_2 \vee B_6 \vee B_{10} \vee B_{14}) \wedge$$
$$\wedge (\overline{T}_5 \vee B_2 \vee B_7 \vee B_{11} \vee B_{13}) \wedge (\overline{T}_6 \vee B_2 \vee B_8 \vee B_{12} \vee B_{14});$$
$$\{V,T\}=(010101)\rightarrow$$
$$B=(0 \vee B_1 \vee B_4 \vee B_{10} \vee B_{14}) \wedge (0 \vee B_2 \vee B_6 \vee B_{10} \vee B_{14}) \wedge$$
$$\wedge (0 \vee B_2 \vee B_8 \vee B_{12} \vee B_{14}) =$$
$$=(B_1 \vee B_4 \vee B_{10} \vee B_{14}) \wedge (B_2 \vee B_6 \vee B_{10} \vee B_{14}) \wedge$$
$$\wedge (B_2 \vee B_8 \vee B_{12} \vee B_{14}) =$$
$$=B_1B_2 \vee B_4B_2 \vee ... \vee B_3B_6B_{12} \vee ... \vee B_{14}.$$

After transformation the conjunctive normal form (CNF) to disjunctive normal form the obtained terms include all possible solutions in the form of unit coordinate coverage for the output response vector by single or multiple software functional failures. Choosing the best solution is made by determining DNF term of the minimum length.

In this example, the optimal solution is a term containing a single block $B = B_{14}$, which covers three units in the output response vector $V = (010101)$. This fact is also evident from comparison of the last two columns of the activation matrix B.

IV. A METHOD FOR VECTOR LOGIC ANALYZING COLUMNS

Methods for detecting the functional failures (FF) in the statement blocks use previously generated functional failure table $B = [B_{ij}]$, where a row is relation between a test segment and subset of activated (on this segment) software blocks $T_i \approx (B_{i1}, B_{i2}, ..., B_{ij}, ..., B_{in})$. A column forms the relation between software block and test segments $B_j \approx (T_{1j}, T_{2j}, ..., T_{ij}, ..., T_{pj})$, which activate it. Otherwise, a column is an assertion vector, detecting the functional failure in corresponding block. On simulation stage the response $m = (m_1, m_2, ..., m_i, ..., m_p)$ of the assertion engine on a test is identified by means of generating each bit $m_i = (A_1 \vee A_2 \vee ... \vee A_i \vee ... \vee A_k),\ A_i = \{0,1\}$ as response of assertions on the test segment T_i. Searching FF's is based on the definition of xor-operation between the vector of assertion states and columns of the functional failure table $m \oplus (B_1 \vee B_2 \vee ... \vee B_j \vee ... \vee B_n)$. The solution is determined by the vector B_j with minimum quantity of 1 coordinates, which determine the functionally faulty software blocks, checked by the test segments. Diagnosis by the functional failure table on the basis of the response $m = (m_1, m_2, ..., m_i, ..., m_n), m_i = \{0,1\}$ is reduced to the methods for vector logic analyzing columns or rows.

The first one is based on use vector xor-operation between m-response of the functionality on the test, formally considered as an input vector-column, and columns of the fault detection table $m \oplus (B_1 \vee B_2 \vee ... \vee B_j \vee ... \vee B_m)$. To determine the interaction quality of vectors $Q_j(m \oplus B_j)$ and to choose the best solution the columns with minimal quantity of 1's for resultant vector are identified. They forms the functionally faulty blocks, checked by test patterns. The analytic model for solving the diagnosis problem and obtaining the list of functionally faulty software blocks is represented in the following form:

$$L = L \underset{j=1}{\overset{n}{\vee}} B_j \leftarrow \sum_{i=1}^{k} (B_{ij} \underset{i=1}{\overset{k}{\oplus}} m_i) = (0 \vee \min). \quad (4)$$

Here an output response vector is input one for subsequent analyzing of the functional failure table

$$m = f(A,B) \oplus f^*(A,B,L)\ . \quad (5)$$

And it is a result of test experiment – comparison of the functional (output states) for model under test $f(A,B)$ and unit under test $f^*(A,B,L)$ with the faults L on the test patterns A. In second case if a set of faults $L > 1$, it means existence of equivalent functional failures on given test and assertion engine.

A process model for searching the best solution with minimum quantity of 1 coordinates from 2 or more alternatives is shown in Fig. 3. It involves the following operations: 1) Initially, in all coordinates (the worst solution) of the vector Q, where the best solution is stored, 1 values are entered; and simultaneously left slc operation with compaction of 1's is performed for given vector Q_i. 2) Comparing of two vectors is performed: Q and the next estimation Q_i from the solution list. 3) Vector operation And $(Q \wedge Q_i)$ is performed. The result is compared with vector Q, which allows changing it, if the vector Q_i has less quantity of 1 values. 4) The procedure for searching the best solution is repeated by n times.

$$Q = Q\overline{(\vee((Q \wedge Q_i) \oplus Q))} \vee Q_i(\vee((Q \wedge Q_i) \oplus Q));$$
$$Y = \vee((Q \wedge Q_i) \oplus Q);$$
$$Q = Q\overline{Y} \vee Q_i Y.$$

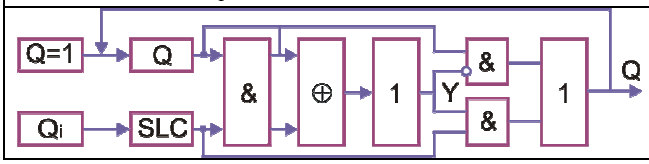
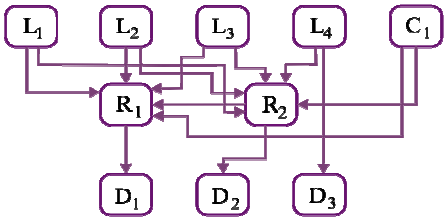


Fig. 3. Process-decision model

An advantage of the method for vector logic analyzing columns is the choice of the best solution from all possible single and multiple faults. Actually, such single functional failures are included in the fault list, which when logical multiplying them by output response vector give a result in the form of vector-column. Disjunction of all columns, generating a solution, is equal to the output response vector

$$\overset{r}{\underset{j=1}{\vee}}(B_j \in B) = m.$$

An example for analyzing the functional failure table FFT of the module Row_buffer (Fig. 4) is represented below.



Test	B_1 L_1	B_2 L_2	B_3 L_3	B_4 L_4	B_5 C_1	B_6 R_1	B_7 R_2	B_8 D_1	B_9 D_2	B_{10} D_3	m_1	m_2
T_1	1	0	0	0	0	1	0	1	0	1	0	1
T_2	0	1	0	0	0	0	1	0	1	0	1	1
T_3	0	0	1	0	0	1	1	1	0	1	0	1
T_4	0	0	0	1	0	0	1	0	1	0	1	0
T_5	0	0	0	0	1	1	1	1	0	1	0	0
T_6	1	0	0	0	0	0	1	0	1	0	1	1
T_7	0	1	0	0	0	1	0	1	0	0	0	1
T_8	0	0	1	0	0	1	0	1	0	0	0	1
T_9	0	0	1	0	0	0	1	0	1	0	1	0
T_{10}	0	0	0	1	0	1	1	1	0	0	0	0
T_{11}	0	0	0	1	0	0	0	0	0	1	0	0
T_{12}	0	0	0	0	1	0	1	0	1	0	1	0
T_{13}	0	0	0	0	1	1	1	1	0	0	0	0

Fig. 4. Row_buffer transaction graph and table FFT

On the basis of the diagnosis procedure (4) and tables FFT (see Fig. 3) the faulty components can be determined by analysis of FFT columns. Here the vectors m_1, m_2 define the diagnosis results, performed by the procedure (5). The diagnosis result for single and multiple functional failures is following:

$$L^s(m_1) = m_1 \wedge (\overset{10}{\underset{j=1}{\vee}} B_j) = B_9 \to D_2;$$
$$L^m(m_2) = m_2 \wedge (\overset{10}{\underset{j=1}{\vee}} B_j) = B_1 \vee B_2 \to L_1 \vee L_2;$$
$$Q(m_1, D_2) = 1;\ Q[m_2, (L_1 \vee L_2)] = \frac{1}{3}(\frac{4}{13} + \frac{1}{4} + 1) = 0{,}52.$$

In the first case, the diagnosis is defined as a single faulty module D_2 that present in the transactional graph; the solution quality is equal to 1. In the second case, the diagnosis procedure detects two faulty modules $L_1 \vee L_2$, the quality estimation of which is not the optimal. Nevertheless, the solution is the best among all the possible, which is maximally approximate to the output response vector by the membership criterion $Q[m_2, (L_1 \vee L_2)]$. The computational complexity of the method for analyzing columns is determined by the following dependence: $Z^c = 3n^2 + n^2 = 4n^2$; $Z^r = 3n + n = 4n$. Here, the first estimate takes into account the implementation of coordinate operations on the matrix of the dimension $n \times n$. The second estimate determines the computational complexity of the register parallel operations to compute quality criteria and process the matrix, respectively.

V. Method for Vector Logic Analysis of Rows

The method is designed for determination of fault or functional failure (FF) location in software code and consists of two procedures: 1) determining the logical product of the conjunction of lines, marked by unit values of the vector $T_i(m_i = 1)$, by the negation of disjunction of zero rows $T_i(m_i = 0)$ for single faulty modules; 2) determining the logical product of disjunction of unit lines by the negation of the disjunction of zero rows for multiple faulty modules:

$$L^s = (\underset{\forall m_i = 1}{\wedge} T_i) \wedge (\overline{\underset{\forall m_i = 0}{\vee} T_i});$$
$$L^m = (\underset{\forall m_i = 1}{\vee} T_i) \wedge (\overline{\underset{\forall m_i = 0}{\vee} T_i}); \qquad (6)$$

The formulas are interesting, because they are not related to the diagnosis quality criteria and operate only two components: FFT table and output response vector. Performing the diagnosis procedure by the formulae (4) for the output response vector $m_1 = (0101010010010)$, specified in the last table FFT, forms the result: $L^s(m_1, T) = D_2$, which is not worse than previously obtained by the method for analyzing columns. For the output response vector $m_2 = (1110011100000)$ the diagnosis result is: $L^m(m_2, T) = L_1 \vee L_2$. Computational complexity of the method for analyzing rows is determined

by the following dependence: $Z^c = n^2$; $Z^r = n$. The first estimate is designed to count the number of coordinate operations, the second one determines the computational complexity of processing, based on the register parallel operations. The proposed methods for diagnosing functional failures in software and hardware are the most important components of the Infrastructure IP.

Formulae (6) can be modified if the following designations are introduced:

$$a = (\underset{\forall m_i=1}{\wedge} T_i);\ b = (\underset{\forall m_i=0}{\vee} T_i);\ c = (\underset{\forall m_i=1}{\vee} T_i);$$

$$L^s = a\bar{b} = a \oplus ab = a(a \oplus b) = a(b \oplus 1);$$

$$L^m = c\bar{b} = c \oplus cb = c(c \oplus b) = c(b \oplus 1);$$

$$L = \begin{cases} a\bar{b} = a\bar{b} = a \oplus ab = a(a \oplus b) = a(b \oplus 1); \\ c\bar{b} = c\bar{b} = c \oplus cb = c(c \oplus b) = c(b \oplus 1) \leftarrow a\bar{b} = 0 \end{cases}$$

Any right side expression of the equations can be used to detect functional failure in the software or hardware. The difference lies in the presence or absence of inversion, which is replaced by xor-operation, more preferable for diagnosis and pattern recognition. In this case, the process model for diagnosing single (using a-component) or multiple (b-component) faults (functional failures) based on analyzing the table FFT has an effective vector-oriented computing technology:

$$L = (b \oplus 1)(a \vee c),$$

embedded Infrastructure IP of software/hardware. According to set theory, this means determining the result of set-theory subtraction $L = (a \vee c) \setminus b = (a \setminus b) \vee (c \setminus b)$ in the algebra-logic vector space. For such operations the multimatrix processor is needed, which is strictly focused on the parallel execution of several logic operations on data matrices.

VI. Matrix Method for Detecting the Functional Failures in Software

Further to the software transaction graph (3) a method for diagnosing functional failures in software uses the triad of matrices of the same format:

$$M = B \oplus A \oplus L = 0,$$

$$L = B \oplus A \leftarrow L_{ij} = B_{ij} \oplus A_{ij} \leftarrow \{B_{ij}, A_{ij}, L_{ij}\} = \{0,1\};$$

$$B = [B_{ij}],\ A = [A_{ij}],\ L = [L_{ij}],$$

$$i = \overline{1,n};\ j = \overline{1,m};\ \oplus = a\bar{b} \vee \bar{a}b.$$

Here matrices form: B – block activization on test segments during simulation; A – activity of assertions, corresponding to blocks, on test segments and during simulation; L – faulty blocks, obtained as result of xor-operation on two above matrices. Coordinate-wise analyzing the matrices uses binary xor-operation, such as (see Table I).

Obtained result $L = B \oplus A$ in the form of L-matrix $[L_{ij}] = (T \times B \times \{0,1\})$, all coordinated of which are equal to zero, indicates absence functional failures in software relatively the proposed verification plan in the format (test – functional blocks – activization $[B_{ij}] = (T \times B \times \{0,1\})$, test – assertions – response $[A_{ij}] = (T \times A \times \{0,1\})$. Another model experiment indicates presence the functional failures $L = \{B_1, B_2, B_3, B_5, B_6\}$ in software code (see Table II).

Here are the results of vector operations on all rows of two tables $\vee L_i$=11101100 and $\vee A_i$=11011111. Logical conjunction of them with the preliminary inversion of the first vector gives the coordinates of blocks with functional failures, marked by units. In this example, the vector forms only one block (00100000)&(11101100)=(00100000).

Tables I, II

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	.	.	.	.	.	.	.	.
T_2	.	.	.	.	.	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	.	.	.	.	.	.	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	.	.	.	.	.	.

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	1	.	.	1	.	.	1	.
T_2	.	0	0	.	0	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	0	.	0	.	.	0	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	0	1	1	.	.	.
$\vee A_i$	1	1	0	1	1	1	1	1

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	.	.	.	.	.	.	.	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	1	.	1	.	.	1	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	1	.	.	.	.	.
$\vee L_i$	1	1	1	0	1	1	0	0

What is the reason for the reduction of faulty blocks? If to assume that in compliance with the verification plan the verification of the first block has to detect faults on first and sixth test, which is not satisfied, so block 1 can be excluded from the fault list. Similarly, modules 2, 5, 6 can be excluded. Then the corrected result will have only one block with the functional failures: $L = \{B_3\}$. The procedure for refining the diagnosis result can also be formalized in the following form: $L = B_j \leftarrow B_j \wedge L_j = B_j, j = \overline{1,m}$. If the comparison result is negative $B_j \oplus L_j = 0$, it means the code is incorrect, assertion or test failed, including functional coverage. For the diagnosis code in accordance with the process model of the form

$$L(B,T) = (B \oplus A) \rightarrow L(B) = (\overline{\vee_{i=1,n} A_i}) \wedge (\vee_{i=1,n} L_i),$$

it is necessary to consider the following items:

1. Coverage is any metric for choosing test and determining its confidence. Code coverage is test metric, focused on the confirmation of execution of all code lines. Decomposition of software code into blocks is performed $B = \{B^s, B^t\} \leftarrow B^s \cap B^t = \varnothing, B^s \cup B^t = B$. Each block belongs to one of two types: the sequence of statements without a branch or time delay circuit $B_i \in \{B^s, B^t\}$. Location of assertion monitors is carried out for block activity on test at the beginning of the branch or in the first timer cycle of a time delay circuit. In the modeling process assertions form an activization matrix for software blocks on each test segment $B_{ij} = T_i \oplus B_j \in \{0,1\}$. If the block is active (assertion passed) on the test (testbench), matrix coordinate is equal to 1, otherwise – $B_{ij} = 0$. Testbench is input conditions for testing the HDL-code and corresponding output responses, which define transformations of the device under test in the functional subspace.

2. Functional coverage is test metric that ensures the accessability of all essential states in the software variable and function definition space. Decomposition of software functionality in control and transaction graphs is performed: $F = \{F^c, F^t\} \leftarrow F^c \cap F^t = \varnothing, F^c \cup F^t = F$. This makes it possible to considerably reduce the dimension of coverage problem that defines the domain for the control variable and data flow. Test generation and the subsequent coverage driven verification use the above mentioned graphs with constraints, taken from the specification. Synthesized test for the control graph allows activation of all logic and arithmetic variables involved in initiation of software transaction. Way of variable activation or test synthesis consists of pseudo-random or deterministic (algorithmic) generating test inputs, as well as hand-writing input stimuli. Forms of coverage definition are an abbreviated truth table, Boolean equation, binary decision diagrams, the flowgraph. Test for the second graph handles data flows, which at the system level not always have to be checked because of the absence of faults, such as short circuits between the variables or constant faults in them. Transaction graph can be used to create a verification plan for essential interface parameters of software. To do this it is necessary to use interface assertions operating by global variables.

2. Assertion matrix for software blocks has a form similar to the structure of block activation $A = [A_{ij}]$. Here format of assertion as logic statement, using the essential variables of software block $f(X) = A_{ij} = \{0,1\}$, responses for running the corresponding activated on the test module $B_{ij} = 1$. Several statements can be in the block, separated to increase the diagnosis depth or united by function or. In last case assertion responses for correct functioning of the block. Assertion has two values: 1 – block operates fault-free, 0 – there are functional failures. Assertions are represented by two hierarchy levels: interface and block ones $A = \{A^i, A^b\}$. The first ones are focused on testing the essential parameters of the specifications, which are common for the software and external for it. Second ones are built into software block, which don't have branches. Power of commands or code lines – up to 20 – is determined by the number of statements to be placed on the screen. Such block can contain time or event delay statements.

VII. Implementation of Models and Methods in the Verification System

Practical implementation of models and verification methods is integrated into the simulation environment Riviera of Aldec Inc., Fig. 5. New assertion and diagnosis modules, added in the system, improved the existing verification process, which allowed 15% reduction the design time of digital product.

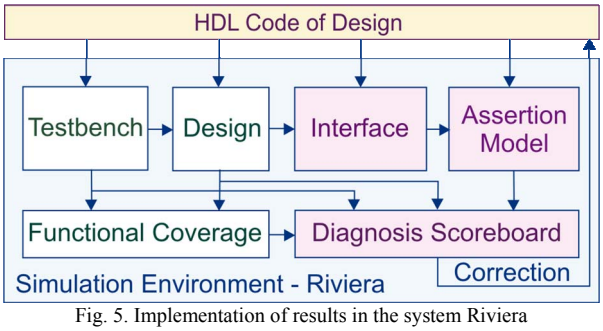


Fig. 5. Implementation of results in the system Riviera

Actually, application of assertions makes possible to decrease the length of test-bench code and considerably reduce (x3) the design time (Fig. 6), which is the most expensive. Assertion engine allows increasing the diagnosis

depth of functional failures in software blocks up to level 10-20 HDL-code statements.

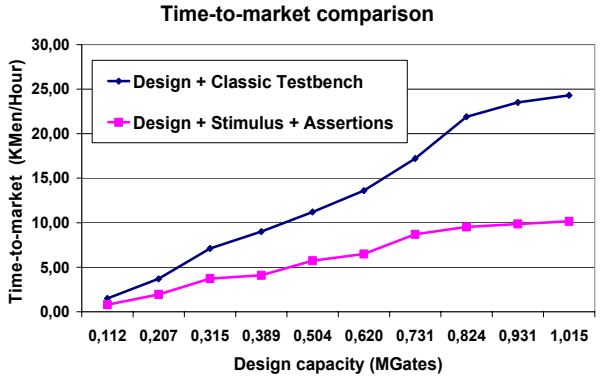


Fig. 6. Comparative analysis of verification methods

Due to the interaction of simulation tools and assertion engine, automatically placed inside the HDL-code, an access of diagnosis tools to the values of all internal signals is appeared. This allows quickly identifying the location and type of the functional failure, as well as reducing the time of error detection in the evolution of product with top-down design. Application of assertion for 50 real-life designs (from 5 thousand up to 5 million gates) allowed obtaining hundreds of dedicated solutions, included in the verification template library VTL, which generalizes the most popular on the market EDA (Electronic Design Automation) temporal verification limitations for the broad class of digital products. Software implementation of the proposed system for analyzing assertions and diagnosing HDL-code is part of a multifunctional integrated environment Aldec Riviera for simulation and verification of HDL-models.

High performance and technological combination of assertion analysis system and HDL-simulator of Aldec company is largely achieved through integration with the internal simulator components, including HDL-language compilers. Processing the results of the assertion analysis system is provided by a set of visual tools of Riviera environment to facilitate the diagnosis and removal of functional failures. The assertion analysis model can also be implemented in hardware with certain constraints on a subset of the supported language structures. Products Riviera including the components of assertion temporal verification, which allow improving the design quality for 3-5%, currently, occupies a leading position in the world IT market with the number of installations of 5,000 a year in 200 companies and universities in more than 20 countries on the world.

VIII. Multimatrix processor of binary operations and verification infrastructure

To implement effective computational processes by time and cost of associated with the diagnosis of functional failures it is necessary processor of the simple architecture with minimum instruction set, where the operands are not only Boolean variables, but also more complex structures such as registers and matrices. Such processor should execute in parallel mode operations over all bits of the regular operands, not requiring special compilers for paralleling computing processes.

Multimatrix processor (MMP) is a minimum architecture of instruction primitives, where each of them focused on the parallel execution of only one operation (and, or, xor, slc) over the corresponding matrix (two-dimensional data array). The number of command-oriented matrix primitives creates a system – a heterogeneous multimatrix processor of binary operations with buffer M, Fig. 7.

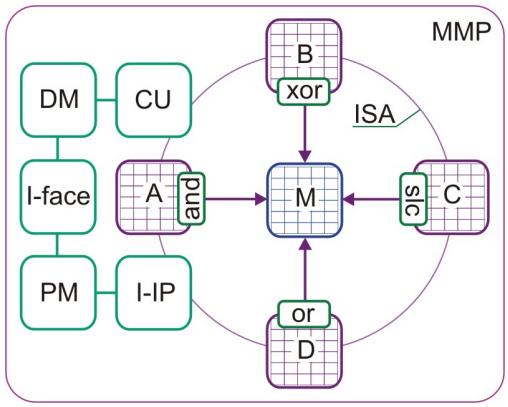


Fig. 7. Multimatrix processor of binary operations

The standard blocks are shown here: data DM and program PM memory, control unit CU, interface I-face and infrastructure I-IP, as well as multimatrix processor, including 4 memory blocks with embedded operations (A – and, B – xor, C – or, D – slc – shift left crowding) and buffer memory M. Multimatrix processor (MMP) is focused on parallel execution one of four instructions (ISA – Instruction Set Architecture) for processing matrices of binary data of the same dimension $M = M\{\text{and}, \text{or}, \text{xor}, \text{slc}\}\{A, B, C, D\}$ and saving the result in the buffer M. Feature of MMP is that each instruction has data matrix for parallel processing (not matrix cell has instruction set of 4 operations) to simplify the control structure and device in whole. The complexity of MMP is focused on data structures, matrix memory has a single hardware embedded instruction that enables to implement primitive control system for parallel computing (SIMD – Single Instruction Multiple Data). Proposed MMP architecture is adapted to execution of logic instructions by the operands of register level. MMP prototype is integrated in the hardware acceleration board for simulation and verification HES™, Aldec Inc.

On the basis of multimatrix (register) processor an infrastructure for verification HDL-code (Fig. 8) is developed. It is modification of I-IP standard IEEE 1500 SECT [3, 4, 11, 14]. There are 4 process models: testing on the simulation stage, diagnosis of functional failures, diagnosis optimization, repairing.

1. Process model for testing involves HDL-model, assertion engine, testbench and coverage. Last one estimates test quality for all design states. In simulating the

activization matrix B for software blocks and assertion response matrix A on test segments are generated. Matrix A can be transformed to assertion state vector m by application of the function Or to vector-columns of A-matrix.

$$\begin{cases} B = (T \oplus F); \\ m = \bigvee_{j=1}^{m} A_j \leftarrow A = (T \oplus A^c). \end{cases}$$

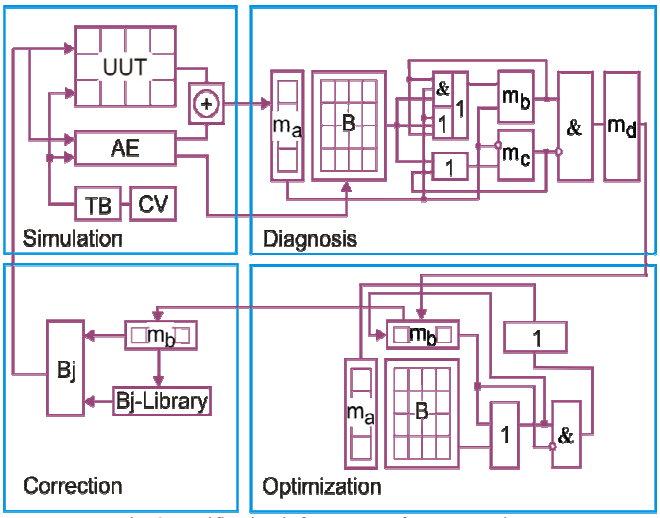


Fig. 8. Verification infrastructure for HDL-code

2. The last two components are used in the second process model for diagnosing blocks of HDL-code. Diagnosis is fault vector, which forms a subset of blocks m_d with functional failures. At that the errors can be in testbench and in assertion statements, which are designed for testing and monitoring software blocks. If exact identification of the block is absent when comparing the columns of activization matrix and assertion responses, triple diagnosis uncertainty $D = \{B_j, T_i, A_{ij}\}$ arises.

3. The third block solves the problem of minimizing the number of blocks, in which functional failures can be, up to one of them. At that a block activization matrix and the diagnosis m_d, obtained in the previous process model, are used.

4. Correction of functional failures is focused on manual searching errors in a software block, presented by the vector m_b. Automated correcting errors in the block is possible, if there is a library of diversion software modules of the similar functionality in the verification infrastructure.

The proposed infrastructure is one of steps towards the creation of verification automaton for software blocks. An example of diagnosing the functional failure, based on using the activization matrix, is represented below. The vector of assertion responses is obtained from the matrix $A_{ij} = \{1 \rightarrow \text{failed}, 0 \rightarrow \text{passed}\}$ by disjunctive union of rows content:

$$m_i = \bigvee_{j=1}^{m} A_{ij} =$$

T	m
T_1	.
T_2	1
T_3	.
T_4	1
T_5	.
T_6	.
T_7	.
T_8	1

=

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	.	.	.	.	.	.	.	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	1	.	1	.	.	1	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	1	.	.	.	.	.

Subsequent implementation of xor-operation between the assertion vector and activization matrix columns allows obtaining the best solution, which is determined by the minimum code distance

$$L = L \vee B_j \leftarrow \sum_{i=1}^{n} (B_{ij} \overset{n}{\underset{i=1}{\oplus}} m_i) = (0 \vee \min):$$

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

T	m
T_1	.
T_2	1
T_3	.
T_4	1
T_5	.
T_6	.
T_7	.
T_8	1

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	1	.	.	1	.	1	1	1
T_3	.	.	.	.	.	1	1	1
T_4	.	1	.	1	1	.	1	1
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	1	1	.	.	.	1	1	1
$d(A,B_j)$	4	4	0	4	2	4	6	6

Diagnosis is block 3 has functional failures, because three assertions are failed on the test segments 2,4 and 8, which in this combination activate only block number 3. If assertion matrix (not vector) is used for diagnosing, searching for faulty blocks is the following:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	.	.	.	.	.	.	.	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	1	.	1	.	.	1	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	1	.	.	.	.	.

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	.	.	.	.	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	.	.	.	.	.	.	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	.	1	1	.	.	.
$d(A,B_j)$	2	2	0	3	2	2	3	3

Diagnosis is similar to the previous one: block 3 has functional failures, because the code distance is equal to zero only for the column number 3.

IX. Conclusion

The following results are proposed in the paper:

1. A structural model for relations on the set of four main components of technical diagnosis (functionality, unit, test and faults), which is characterized by complete xor-interaction of all the graph nodes and transitive reversibility of each relation triad that allows defining and classifying the ways of solving practical problems, including test synthesis, fault simulation and fault detection.

2. A new model of software in the form of Code-Flow Transaction Graph, as well as a new matrix method for diagnosing functional failures, which are characterized by adaptability of data preparation when detecting faulty

blocks, are proposed. They allow considerably reducing the design time of digital systems on chips.

3. Methods for searching functional failures, which differ in parallel execution of vector operations on the rows of a functional failure table, are improved. They allow substantially (x10) increasing the performance of computational procedures associated with diagnosis and repair of software and hardware.

4. The architecture of multimatrix processor, focused to increasing the speed of embedded diagnosis of functional failures in the software or hardware product, which differs using parallel logic vector operations and, or, xor, slc that enables to increase considerably (x10) the speed of diagnosing single and/or multiple faults (functional failures).

5. The infrastructure for verification and diagnosis of HDL-code for design digital systems-on-chips, which involves four process models for testing, diagnosing, optimization and correcting errors, closed in a cycle, that makes it possible to reduce the time of code debugging, when creating a design.

6. Practical implementation of models and verification methods is integrated into the simulating environment Riviera of Aldec Inc. New assertion and diagnosis modules improved the existing verification process, which allowed 15% reduction in overall design time of digital products.

REFERENCES

[1] Technical diagnosis basics / Editor P.P. Parchomenko.- M.: Energy.– 1976.– 460 p.

[2] Parchomenko P.P., Sogomonyan E.S. Technical diagnosis basics (Optimization of diagnosis algorithms, hardware tools) / Editor P.P. Parchomenko.– M.: Energy.– 1981.– 320 p.

[3] Infrastructure for brain-like computing / M.F. Bondaryenko, O.A. Guz, V.I. Hahanov, Yu.P. Shabanov-Kushnaryenko.– Kharkov: Novoye Slovo.– 2010.– 160 p.

[4] Design and Verification of digital systems on chips / V.I. Hahanov, I.V. Hahanova, E.I. Litvinova, O.A. Guz.– Kharkov: Novoye Slovo. – 2010. – 528 p.

[5] Semenets V.V., Hahanova I.V., Hahanov V.I. Design of digital systems by using VHDL language.– Kharkov: KHNURE.– 2003.– 492 p.

[6] Hahanov V.I., Hahanova I.V. VHDL+Verilog = synthesis for minutes. – Kharkov: KHNURE.– 2006.– 264 p.

[7] Hahanov V.I. Technical diagnosis of digital and microprocessor structures: Manual.- K.: ISIO, 1995.- 242 p.

[8] Skobtsov Yu.A. Logic simulating and testing digital devices / Yu.A. Skobtsov, V.Yu. Skobtsov. –Donetsk: IPMM NSA of Ukraine, DonNTU.– 2005.– 436 p.

[9] IEEE Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture IEEE Std 1149.7-2009. – 985 p.

[10] Da Silva F., McLaurin T., Waayers T. The Core Test Wrapper Handbook. Rationale and Application of IEEE Std. 1500™. –Springer.– 2006.– XXIX.– 276 p.

[11] Marinissen E.J., Yervant Zorian. Guest Editors' Introduction: The Status of IEEE Std 1500.–IEEE Design & Test of Computers.– 2009.– No26(1).– P.6-7.

[12] IEEE Std 1800-2009 IEEE Standard for System Verilog-Unified Hardware Design, Specification, and Verification Language. http://ieeexplore.ieee.org/servlet/opac?punumber= 5354133

[13] Marinissen E.J. Testing TSV-based three-dimensional stacked ICs // DATE 2010.– 2010.– P.1689-1694.

[14] Benso A., Di Carlo S., Prinetto P., Zorian Y. IEEE Standard 1500 Compliance Verification for Embedded Cores // IEEE Trans. VLSI Syst.– 2008.– No 16(4).– P. 397-407.

[15] Ubar R., Kostin S., Raik J. Embedded diagnosis in digital systems // 26th International Conference “Microelectronics”, MIEL 2008. – 2008.– P. 421-424.

[16] Elm M., Wunderlich H.-J. Scan Chain Organization for Embedded Diagnosis // Design, Automation and Test in Europe, DATE '08.– 2008.– P. 468–473.

[17] Bulent I. Dervisoglu. A Unified DFT Architecture for Use with IEEE 1149.1 and VSIA/IEEE P1500 Compliant Test Access Controllers. Proceedings of the Design Automation Conference. – 2001. – P. 53-58.

[18] Chenlong Hu, Ping Yang, Ying Xiao, Shaoxiong Zhou. Hardware design and realization of matrix converter based on DSP & CPLD // 3rd International Conference Power Electronics Systems and Applications.– 2009.– P. 1-5.

[19] Dave N., Fleming K., Myron King, Pellauer M., Vijayaraghavan M. Hardware Acceleration of Matrix Multiplication on a Xilinx FPGA // 5th IEEE/ACM International Conference Formal Methods and Models for Codesign.– 2007.– P.97-100.

[20] Loucks W.M., Snelgrove M., Zaky S.G. A Vector Processor Based on One-BitMicroprocessors // IEEE Micro.–Volume 2, Issue 1.– 1982.– P. 53-62.

[21] Hilewitz Y., Lauradoux C., Lee R.B. Bit matrix multiplication in commodity processors // International Conference Application-Specific Systems, Architectures and Processors.– 2008.– P. 7-12.

[22] Soon, J.L.K.; Low Ching Ling; DEV. Design explorer for verification. Integrated Circuits, ISIC '09. Proceedings of the 2009 12th International Symposium: 2009, Page(s): 413 – 416.

[23] Rafe, V.; Rafeh, R.; Azizi, S.; Miralvand, M.R.Z.; Verification and Validation of Activity Diagrams Using Graph Transformation. Computer Technology and Development, 2009. ICCTD '09. 2009 , Page(s): 201-205.

[24] Xiaoxi Xu; Cheng-Chew Lim; Using Transfer-Resource Graph for Software-Based Verification of System-on-Chip. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume: 27, Issue: 7. 2008, Page(s): 1315 – 1328.

[25] Zhongjun Du; Zhengjun Dang; A New Algorithm Based Graph-Search for Workflow Verification. Information Engineering and Computer Science (ICIECS), 2010 2nd International Conference: 2010, Page(s): 1–3.

[26] Gorbatov V.A., Gorbatov A.V., Gorbatova M.V. Discrete mathematics.– M: High School, 2006.– 448 p.

A Method of High-Level Synthesis and Verification with SystemC Language

Volodymyr Obrizan, Kharkov National University of Radio Electronics, Ukraine

***Abstract*—This paper presents a method for automatic RTL-interface synthesis for a given C++ function as well as for a given SystemC-interface. This task is very im-portant in High-Level Synthesis design flow where design entry is usually done in some abstract language (e.g. C++). As a source high-level description targets different SoC architectures or protocols, so it is needed to generate relevant pin-level interfaces and protocols automatically.**

***Index Terms*— Computer languages. High level synthesis, System-level design, System testing.**

I. Introduction

A system level interface can be mapped to different RTL (or pin-accurate) interfaces. The mapping depends on a selected architecture and protocol.

The goal of the presented research is to reduce design time and human efforts needed to generate a pin-accurate interface and a protocol for a given arbitrary high-level description.

The research tasks are:

a) to research the state of the art;

b) to research a mapping between high-level inter-faces and RTL interfaces;

c) to develop a method for generation of a communication protocol;

d) to test the proposed solution.

This paper is organized as follows. State of the art is presented in the second section. The third section defines the prerequisites of the proposed method: a) a mapping between system-level and RT-level interfaces; b) a way how the communication protocol is specified. We conclude in the fourth section.

II. State Of The Art

There are several methods for interface synthesis. The interface synthesis task is defined as follows: to generate an interface between two processes with arbitrary protocols [1]. In [2] authors present a method of interface generation for a given waveform. In [3] authors propose a method to generate interface circuits. The proposed solution produces flexible microarchitectures from FSM descriptions. The FSM descriptions are derived from a formal language called SIMPLE. In [4] authors present a method to generate hardware interfaces and protocols depending on a VHDL description. In [5] authors report on a formal language to specify protocols for further synthesis. None of the above methods solve the problem, if a system model is described in C++.

Among modern System-on-Chip communication protocols we can outline the following: AMBA [6], Wishbone [7], CoreConnect, Open Core Protocol, Avalon [8]. The main disadvantage of these specifications is that they provide the description of the protocols in verbal form with waveforms. There are no algorithms, FSMs, or transactors.

III. A Format Of A Source Description

Let's consider prerequisites which make this method possible. We will consider two main points:

a) a definition of an interface;

b) a definition of a communication protocol.

A. A definition of an interface

Let F — is a function defined with a high-level description language, X = (x1, x2, …, x_i) — is a vector of arguments , Y = (y_1, y_2, …, y_k) — is a function's output vector. Then, a function to be synthesized looks like:

$$Y = F(X). \quad (1)$$

On later design stages, the low-level interface of a module must be defined. However, a number of different low-level interfaces can be associated with a single high-level interface.

In a source model written in C++, a module interface can be defined in the following ways:

a) as a function or member function declaration (SystemC-interface);

b) as a SystemC module (SC_MODULE) with a SystemC pin-accurate interface;

c) in some other way, different from the above.

A function (member function) declaration looks like the following:

```
return_type function_name (argument_list);
```

Manuscript received December 15, 2010.

Volodymyr Obrizan is with Kharkov National University of Radio Electronics, Kharkov, Ukraine (e-mail: volodymyr.obrizan@gmail.com).

here return_type — is a function return value type, argument_list — is a list of arguments, including each argument's type and optional argument's identifier. A C++ function can return values of arbitrary built-in types and user types, pointers or references to them. There is one exception — type void means function doesn't return a value. A mapping between C++ and VHDL types is shown in Table 1.

TABLE I
A MAPPING BETWEEN C++ AND VHDL TYPES

C++ type	Width, bit	VHDL type
char, unsigned char, signed char, bool	8	std_logic_vector (7 downto 0)
unsigned short int, float, short int	16	std_logic_vector (15 downto 0)
int, unsigned int, unsigned long int, long int, double	32	std_logic_vector (31 downto 0)
*T (pointer to T)	32	std_logic_vector (31 downto 0)
long long, long double	64	std_logic_vector (63 downto 0)
wchar_t, void	—	n/a, non-synthesizable

Thus, a high-level synthesis program has the following options: 8, 16, 32, and 64-bit types without intermediate values. These types will be translated into registers and buses of respective widths. Sometimes, it may result in unreasonable hardware expenses. For example, let's suppose that we need to encode 10 values, so we use a variable of type char mapping to an 8-bit register. The register can encode 256 values at the most. But to encode 10 values it is needed only 4 bits. Thus, a half of the register will not be used. In general case, if we need to encode n values, then number of required bits (k) is:

$$k = \lceil \log_2 n \rceil. \quad (2)$$

To address this issue, SystemC provides two type sets to work with arithmetic values with arbitrary precision: from 1- to 64-bit types, 64-bit and above types. Classes sc_int (signed integer) and sc_uint (unsigned integer) model integer arithmetic types in a range from 1 to 64 bits. These classes are recommended for use only for synthesis if unambiguity between high-level model simulation and RTL simulation is needed. Sole-ly for high-level simulations, it is better to use native int type, because its simulation speed is higher. Classes sc_bigint and sc_biguint model 64-bit types and above.

Let's consider the greatest common factor function. Its SystemC-interface is defined as a virtual member function (see below). Then, one needs to inherit this interface and implement its behavior. (This task isn't considered in this paper.)

```
class find_gcf_if : public sc_interface
{
public:
   virtual int find_gcf (int a, int b) = 0;
}
```

A synthesis program can extract number and types with relative bit-widths of the parameters using such description. Fig. 1 shows the result of such synthesis. We see the well-defined RTL interface.

```
int find_gcf(int a, int b);
```

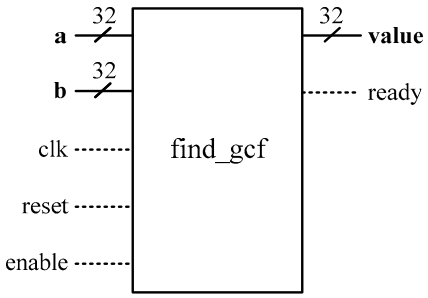


Fig. 1. A function declaration and its RTL interface

There are three groups of ports:

a) informational: a, b, value — the number, names and sizes depend on a given C++ function decla-ration;

b) global control: clk, reset — usually the same for any function.

c) protocol: enable, ready — the number and mean-ing depend on a given protocol.

B. A definition of a communication protocol

Usually, a communication protocol is given as a textual description of rules which one must follow to successfully communicate with a device. Also, such description is supplemented with waveforms. However, these kinds of description aren't good for automatic translation, so a designer should manually specify a communication protocol in VHDL or Verilog.

Let's consider a simple communication protocol. The waveform is shown in fig. 2. The communication rules are the following.

1. Set up the parameters to the inputs 'a' and 'b'.
2. Set up 'enable' signal.
3. Wait until 'ready' signal is '1'.
4. Read the result of the calculation at 'value' port.

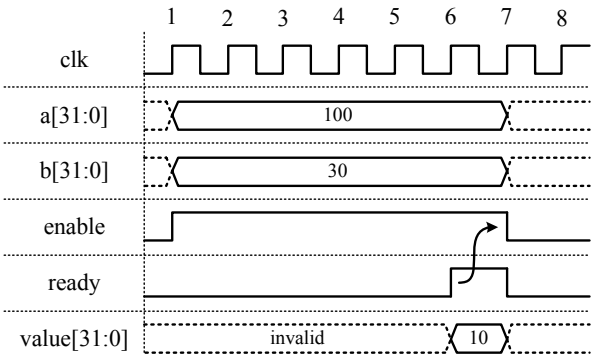


Fig. 2. A simple communication protocol

On the slave's side these rules are defined as follows.

1. Wait until 'ready' signal is set.
2. Read the inputs 'a' and 'b'.
3. Perform the calculation.
4. When the calculation is done, set 'value' output and 'ready' signal.
5. Wait one clock cycle and reset 'ready' signal.

We can represent these rules in a form of the following algorithms. There are two algorithms: one for the master process (fig. 3) and one for the slave process (fig. 5). There are corresponding finite-state machines for these algorithms: fig. 4 and fig. 6 respectively.

The loop at the state a1 in the master process' FSM means that the master process must keep the arguments and 'enable' signal unchanged till 'ready' signal is set. This makes the master process to wait until the slave process is ready with calculations and the result is stable.

Actually, at the FSM in the fig. 5, the state a_1 is a group of states, because most calculations will be done in more than one clock cycle. So this group state can consist of many operation and decision vertices needed to implement the function being synthesized.

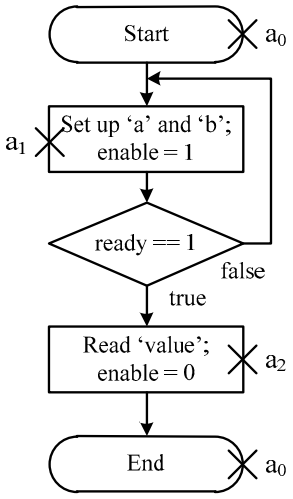


Fig. 3. An algorithm of the master process

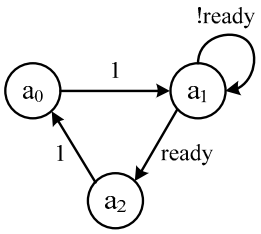


Fig. 4. An FSM of the master process

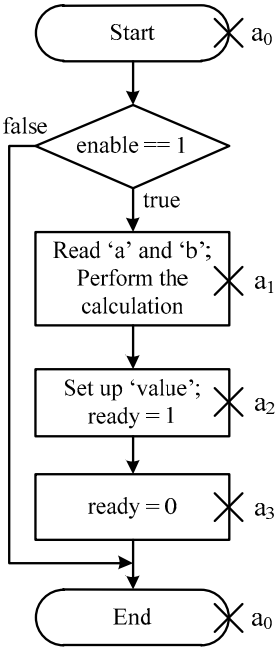


Fig. 5. An algorithm of the slave process

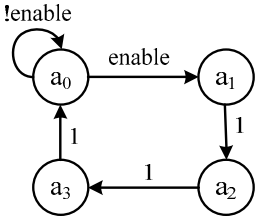


Fig. 6. An FSM of the slave process

In general case, these algorithms depend on a number of arguments and body logic of the function. So it is not a complex task to make the method work for arbitrary function declarations.

These algorithms can be easily synthesized into FSMs and further into RTL interfaces and corresponding protocol logic.

Also, the master process' algorithm can be used to generate a transactor to verify the results of synthesis. This transactor converts higher-level requests (function calls) to lower-level events (logic signals activations). To do so, it is needed to implement the master process algorithm as a SystemC-transactor. For example, for the given function the

master transactor will look as follows:

```
class find_gcf_trans : public find_gcf_if,
   public sc_module {
public:
   // RTL-interface
   sc_out<int> a, b;
   sc_in<int> value;
   sc_in<bool> clock, reset, ready;
   sc_out<bool> enable;

   SC_CTOR(find_gcf_trans) { }

   // High-level interface
   virtual int find_gcf(int a, int b) {
      wait(clock->posedge_event());
      this.a = a;
      this.b = b;
      enable = true;
      while(!ready)
         wait(clk->posedge_event());
      enable = false;
      return value;
   }
};
```

The complete system of transactors and modules is shown in fig. 7. The 'Testbench' module is a high level test-bench written using SystemC language. It has a single port of the *find_gcf_if* type (see section III). The 'H2L' module is a transactor converting high-level calls to low-level binary signals. It works accordingly with the algorithm shown in fig. 3. and the FSM shown in fig. 4. The 'L2H' module is a transactor converting low-level binary signals to high-level system calls. It works accordingly with the algorithm shown in fig. 5 and the FSM shown in fig. 6. The 'find_gcf' module is a high-level module written in SystemC language. However, in this system, the L2H module can be substituted with an RTL or gate-level module. So, the high-level test-bench can be reused on all levels of abstraction.

It should be noted, that the test-bench is a clock-less module. The notion of time is specified in transactors only.

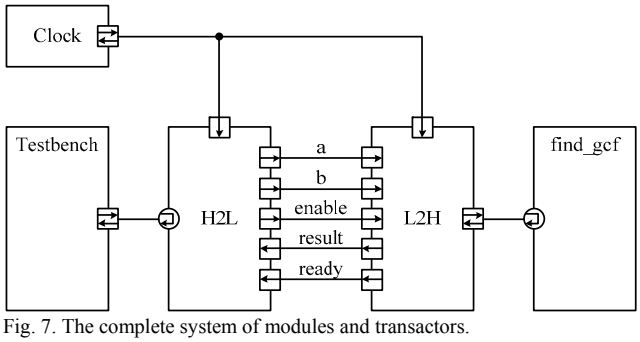


Fig. 7. The complete system of modules and transactors.

IV. Conclusion

A method to generate RTL-interfaces with the simple communication protocol for a given function declaration is presented. This method defines the required informational, control and protocol ports and their bit-widths. We propose to specify communication protocols in a form of algorithms to simplify the interface synthesis task. The method implemented as a part of high-level synthesis tool significantly reduces time to RTL and designer efforts.

Also, the proposed method is useful for automatic generation of transactors to verify synthesized solutions. This enables reusing of high-level tests.

This method doesn't consider a case when it is needed to access data in a shared memory (pointers).

Further research lays in analyzing and defining algorithms and FSMs for popular on-chip protocols (AMBA, Open Core Protocol, CoreConnect, etc.). Also, it is needed to refine the method to handle shared memory access via pointers.

References

[1] Narayan, S. and Gajski, D. D. "Interfacing incompatible protocols using interface process generation". In *Proceedings of the 32nd ACM/IEEE Conference on Design Automation*, ACM, San Francisco, California, United States, June 12 - 16, 1995, pp. 468-473.

[2] Chung, K., Gupta, R. K., and Liu, C. L. "An algorithm for synthesis of system-level interface circuits". In *Proceedings of the 1996 IEEE/ACM international Conference on Computer-Aided Design*, IEEE Computer Society, San Jose, California, United States, November 10 - 14, 1996, pp. 442-447.

[3] Yun, C., Kang, D., Bae, Y., Cho, H., and Jhang, K. "Automatic interface synthesis based on the classification of interface protocols of IPs". In *Proceedings of the 2008 Conference on Asia and South Pacific Design Automation*, IEEE Computer Society Press, Seoul, Korea, January 21 - 24, 2008, Los Alamitos, pp. 589-594.

[4] Smith, J. and De Micheli, G. "Automated composition of hardware components". In *Proceedings of the 35th Annual Conference on Design Automation*, ACM, San Francisco, California, United States, June 15 - 19, 1998, pp. 14-19.

[5] Madsen, J. and Hald, B. "An approach to interface synthesis". In *Proceedings of the 8th international Symposium on System Synthesis*, ACM, Cannes, France, September 13 - 15, 1995, pp. 16-21.

[6] "AMBA Open Specifications". arm.com/products/system-ip/amba/amba-open-specifications.php

[7] "Specification for the: WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores". Revision: B.3, Released: September 7, 2002 opencores.org/cdn/downloads/wbspec_b3.pdf

Ecological Corporative System Concept in Solving Problems of Ecological Estimation and Ecological Hygienic Normalization

T.V. Kozulia, N.V. Sharonova, M.M. Kozulia

Abstract — **Approach of methodical application of concept of ecological corporative system (CES) in the system of ecological monitoring is shown in the article. Appropriateness of corporative approach application was indicated for problem solving of standardization of state parameters of environmental objects. The appropriate algorithms of the assessment of the health risk and of the ecological standardization problems were developed on basis of analysis of existent approaches of determination of system status and appropriateness of introduction of the risk-analysis for problem solving of state identification of CES.**

Index Terms — **The ecological corporative system, the risk-analysis, the ecological standardization, the ecological – hygienic assessment.**

I. Introduction: relevance of the work and relation with scientific practical tasks

ENVIRONMENT is the economics basis, means of subsistence, source, of national wealth. As the world is being industrialized, urbanized and the resource base usage is growing, intellectual resource management is getting necessary. Ecological estimation (EE) is one of the ways attaining the goal.

The EE aim is to provide taking into account environment state and ecosystem durability in the development and approval processes of projects, plans for development, programs, and policies. The EE itself is the planning process. It to use prognosis analysis and interpretation of significant affects upon environment, which will be caused by planned activity, and providing information, which is needed during decision-making management.

Manuscript received December 2, 2010.

Kozulia T.V. is with the National Technical Univercity "Kharkov Polytechnic Institute", Kharkov, 61142 Ukraine (corresponding author to provide phone: 3-8057-7076474, 3-8057-7135189; e-mail: kozulia@kpi.kharkov.ua).

Sharonova N.V. is with National Technical Univercity "Kharkov Polytechnic Institute", Kharkov, 61142 Ukraine (phone: 3-8057-7076460, 3-8057-3375982; e-mail: sharonova@kpi.kharkov.ua).

Kozulia M.M. is with the Kharkov National Univercity named after V. N.Karazin, Kharkov, 61142 Ukraine (phone: 3-8057-7076474, 3-8057-7135189; e-mail: kozulia@kpi.kharkov.ua).

The EE is used to prevent and minimize unfavourable affect. It also simultaneously estimates real resource potential and maximizes advantages in order to:

- change and improve planned activity project;
- ensure effective resource usage;
- improve social aspects of planned activity;
- determine ways of monitoring and management;
- assist well-grounded decision-making.

The EE process is connected with characteristic and analysis of the whole influence spectrum which includes biophysical (environmental influence), social, in population health affect, economic, risk and uncertainty aspects are taken into account.

Risk always means the result uncertainty. At the same time risk is frequently understood as the loss possibility, although it is described as probability of getting a different result. Risk as a rule combines event probability the event effect and the consequence of the event caused by it. Risk is a progressing contributor of the process that has a negative influence potential upon main process flow according to Rational Unified Process (RUP) point of view [4].

Statistical risk often comes to the possibility of some undesirable event. As a rule such event possibility and its harm estimation are combined into one verisimilar result. Such result combines risk possibilities set, loss and reward into the expected present result value.

So statistical decision-making theory says that risk estimation function $\delta(x)$ for parameter θ calculated for some observed x is determined as expected loss function L value:

$$R(\theta, \delta(x)) = \int L(\theta, \delta(x)) \times f(x \mid \theta) dx , \qquad (1)$$

where: $\delta(x)$ is estimation, θ is estimation parameter.

As it is known the harmful factor a effect is determined by the biological response spectrum of the organism to any harmful factor affect. Death, disease, physiological disease symptoms, functional changes of indistinct biological importance, pollutants or their metabolic products accumulation in organs and tissues are possible responses. Three zones are distinguished during harmless pollution levels determination. The first zone (absence of factor affect) is named sub-threshold level. The third zone (toxic influence) is characterized by pathological organism changes caused by the pollutant (disease or its sings). The

second zone lies between them. It is the zone of organism changes which are of unclear biological value. Probably such change is connected with protective adjustments. It indicates the environment deviation from the biological optimum.

According to leading hygienists, existing hygienic pollutants normalization principles are widely practically realized. However, theoretical basis of this problem and its methodology are still widely discussed in literature. Concepts of "thresholdlessness", "risk realization inadmissibility", "benefit-harm" and "justified risk" are meant at the moment. All these concepts contradict the main approach established in USSR which substantiated hazardous substances MPC with "harmless" and "medical aspect domination" criterion.

It is necessary to work out a universal approach to solve problems caused by economic activity influence upon environment, by natural and social environment influence upon living organisms and human health.

II. Aims and problems of investigation

Economic, social and natural systems interaction is characterized as versatile. The systems themselves are to be developed and the consequences of this development effect upon living organisms and human health can hardly be predicted. A new concept of solving the above mentioned problem is proposed. It is aimed at harmonizing "economic-social-environment" system. The latter is presented as a corporative system with ecological principles priority and thus it is named the corporative ecological system (CES) [5, 6]. On the basis of the suggested concept and introducing risk theory it is possible to solve the following problems: common developing systems interaction estimation within the in framework of the CES, commonly acceptable managerial decision and healthy human environment conditions determination. To achieve this aim the following tasks touched upon in the present work:

1) to determine main approaches to general system state (CES subsystems) estimation in their present condition their development and external influence;

2) to form methodical approaches to qualitative and quantitative health risk estimation in the sphere of normalization and decision-making concerning secure influence upon human beings.

III. Research materials and scientific results

Corporative ecological system (CES) concept has been proposed in the ecological monitoring system. This concept considers corporative interconnection between three macrosystem components, their independent progress. This approach gives an opportunity to determine every subsystem state and also the state of the system as a whole. Riskless CES and subsystems state is proposed to estimate on the basis of thermodynamics and synergetic [5, 6]. Ecological risk according to thermodynamic ecosystem structure is conditioned by entropy increasing due to disturbances in it. Equilibrium state is characterized by the minimal entropy level.

Managerial decision is estimated according to the CES general state and depending on every CES subsystem entropy and informative ness. It means that CES state entropy is action result parameter. Current decision realization within some time interval $[t_0\ T]$ produces incompatible effect series. Incompatibility is conditioned by thermodynamic flow connection between CES subsystems of local level. Thermodynamic equilibrium maintenance is an optimal functionality criterion of such a system. This means $\Delta S \to \min \to 0$ or $S = \max$ state in relation to other states.

Generally the CES state as a macrosystem is determined by its components macro state. Different CES subsystems macrostates are established together with their different microstates. Certain system state is connected with one of the realization of possible E_n energy levels. Each energy level has its own statistical weight Ω_n. If the internal system energy with entropy S_n is close to E_n then:

$$\Omega_n = e^{(E_n - A)/kT},$$

where A is free energy, k is Boltzmann constant, T is temperature.

The $P_{in} = 1/\Omega_n$ value is corporative system macrostate possibility provided that all system macrostates with energy E_n are equally possible. The CES state from the thermodynamics point of view is determined by the following parameters:

$$S_{CES} = S_{EES} + S_{CS} + S_{ES},\ \Omega = \Omega_{EES} \times \Omega_{CS} \times \Omega_{ES}, \quad (2)$$

where S_{EES}, S_{CS}, S_{ES} are spaces of natural ecological, social and economic systems accordingly; $\Omega_{EES}, \Omega_{CS}, \Omega_{ES}$ statistical weight of CES subsystem macrostate realization.

General CES state realization is characterized by the value:

$$S(t) = \int_V S_{dV}^{(p,e,s)}(x_1(t), x_2(t)...x_n(t))dV, \quad (3)$$

where $S^{(p,e,s)}$ is entropy of natural (ecological), economic and social system state accordingly; dV is a macrosystem size (volume) characteristic.

Entropy change is determined as $\Delta S_{dV} \geq 0$ and corresponds to entropy intensity production in CES. This takes place in the case of irreversible processes in the corporative system components which are physical chemical systems:

$$\sigma(t) = \frac{dS_{dV}}{dt} \geq 0 \text{ if } t \to 0. \quad (4)$$

This value is connected with new CES state probability realization with changes that happen in time. This means that changes are connected with the macrosystem entropy production:

$$P(t) = \frac{dS_{dV}}{dt} = \int_V \sigma(x_1(t), x_2(t)...x_n(t))dV. \quad (5)$$

$P(t) \to \min$ CES is a condition to be met for the system to stay within stationary process limits and managerial influence not to cause entropy production.

Microscopic state equations are received in synergetic approach to the thermodynamic description (model making) realization of three complex systems inside one corporative system. CES equilibrium loss is determined by small amount of collective modes. These modes are order parameters and describe macroscopic structure. These macroscopic variables represent microscopic system components behaviour according to principle of the subjecting at the same time. Order parameters allow the system to turn into structure which has a stable state.

Information about order parameters changes near instability points and subordinate modes information does not change:

$$P(\xi_u, \xi_s) = \prod_s P_s(\xi_s \mid \xi_u) f(\xi_u), \quad (6)$$

where ξ_u is order parameters probability; ξ_s is subordinate modes amplitude probability.

Macrosystem stability state normalization condition looks like this:

$$\sum_{\xi_s} P_s(\xi_s \mid \xi_u) = 1. \quad (7)$$

Equilibrium system state break is connected with entropy production $P = \frac{d_i S}{dt}$. Equilibrium state for CES as a quasi-isolated system is determined as P=0. Equilibrium state breaks the condition of $P = \frac{d_i S}{dt} \geq 0$. When disturbance is small inside CES, entropy production is as follows according to Prigozhin I. R. theorem: $dP \leq 0$ is evolution condition; $P = \min$, $dP = 0$ is equilibrium or stationary state condition.

It is proposed to characterize infringement risk estimation in the macrosystem by integrated index in case of CES. Integrated index takes into account changes inside each subsystem and ecological risk value for each subsystem – ecological (natural), social and economic ones. Such risk index value represents deviation from normative stable state (risk). It also promotes such managerial decision-making which could keep each systems right to develop with keeping balance inside the system and in CES as a whole (Fig. 1).

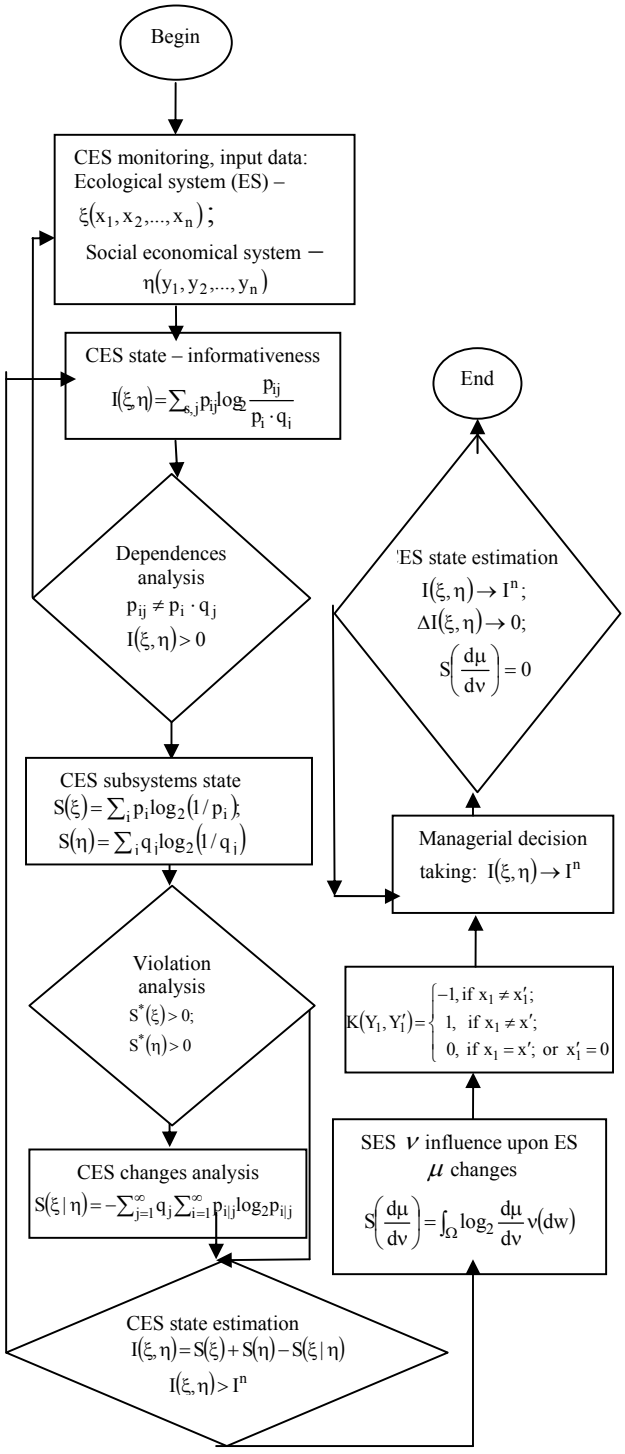


Fig. 1. CES state estimation scheme in the form entropy and comparator K: ξ is probabilistic observation $(x_1, x_2, ..., x_n)$ with probability $(p_1, p_2, ..., p_n)$; η is probabilistic observation $(y_1, y_2, ..., y_m)$ with probability $(q_1, q_2, ..., q_m)$; p_{ij} is probability of interaction of operation $\xi = x_i; \eta = y_j$; S is entropy of systems state; μ is ecological changes to man-made ν argument (in accordance with Radona-Nicodima) ($S_t w = w(t + \tau)$; I^n is unconfigured state, equilibrium position; * is determination S provided that $I^n = 0$

Coordination factors of characteristics and parameters interconsistency supply are given priority when determining CES state. These are characteristics of complicated corporative systems along with thermodynamical parameters of identifying corporative system equilibrium, integrity and evolution harmonization. The situation analysis as to ecological risk estimation and people health risk estimation is proposed to consider from a new point of view. Such a position bars further system development with existing violations in environment and negative influence upon components and elements of natural ecosystems. According to CES concept the estimation of considered object is carried out equilibrium besides population health state estimation in the form of positive risk and boundaries normalization of permissible influence upon natural ecosystems and human beings is carried out too.

Existing risk estimation methods are rather tedious and great amount of information is needed to be processed. On the basis of each factor (relevance) from each of the 5 database blocks a characteristic is retrieved which reflects 75–80% of all initial data and provides a practical analysis of output variables structure.

The present approach does not take into account related systems state (economic and natural systems) and doesn't let to estimate risk level in these systems and its influence upon population health risk level.

The aim of approaches main to consider to ecological risk estimation formation for territories and the population health the research is state. Corporative concept advantages were determined regarding corporative ecological system and positive risk estimation.

Analyzed indicator groups are distributed among three factor blocks according to the CES concept of corporative approach. Population health belongs to the 1st block; the way of life, social economic environment belong to the 2nd block; biological, chemical, physical and social environment factors belong to the 3rd block (Fig. 2).

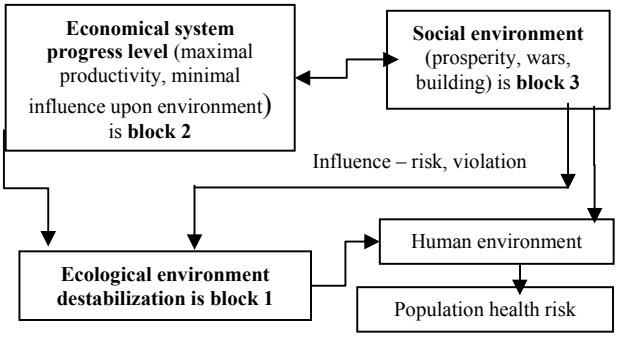


Fig. 2. Block schema of population health assessment

Such work has been carried to determine sanitary epidemiological welfare. Considerable amount of factors are analysed in each block of the multidimensional health risk determination system. There are 33 factors in the 1st block (A), 29 factors in the 2nd (B), 12 factors in the 3rd one. That is why this work provides factors ranking according to their health influence. Bigger factor gets lower rank value if it affects health. Factor value is to be decreased if the direction of the vector is negative. And a bigger factor corresponds a higher rank value in the opposite case.

Rank points are summed for each block for all territories examined. Standardized factor (SF) [10] is calculated for each factor block for each territory. A standardized index is arithmetic mean of the paints sum for each block. It takes into account factors frequencies with positive and negative vectors:

$$SF_i = \sum P_i \cdot (P_{fon})^{-1},\ P_{fon} = \left(\sum P_i^- \cdot f_i^- + \sum P_i^+ \cdot f_i^+\right) \cdot \left(\sum f_i^- + \sum f_i^+\right)^{-1}, \quad (8)$$

where P_i is the sum of points for all indices of each block for each factory (it is calculated by adding factor ranking results with negative (P_i^-) and positive (P_i^+) direction vectors of influence upon health); f_i^- is relative group index frequency, ranked according to negative and positive (f_i^+) vectors influence upon health.

The SF_i values lead to the probabilistic quantities according to the model [9]:

$$P_{risk_i} = 1 - \exp\left[0{,}5 \cdot \left(-H\Pi_i^2\right)\right], \quad (9)$$

where P_{risk} is probabilistic value, which characterizes isolated influence risk prevention of corresponding blocks factor groups (n_j).

It is safe to say two problems appear to an ecological and economic sense within the framework of corporative approach. They concern population health estimation.

1. Research object state, violation probability, destabilizing process risk determination that can be confirmed by entropy value.
2. Forming of standard as estimated values on the influence minimization basis and according to destabilizing factors effect upon living organism (Fig. 3).

This corporative approach realization was considered when solving the problem of ecological security violation detection during Dergachivsky refuse dump and water objects monitoring according to the following scheme:

$$\begin{cases} X = x_i \ / \tilde{A}\ddot{A}\hat{E},\ \sigma(X) = \sqrt{\dfrac{\sum\limits_{i=1}^{\text{length}(X)-1} (x_i - \text{mean}(X))^2}{\text{length}(X)-1}}, \\ F(X,i) = \dfrac{1}{\sqrt{2\pi}\sigma(X)} \cdot \exp\left[\dfrac{-1}{2(\sigma(X))^2}(x_i - 1)^2\right]. \end{cases} \quad (10)$$

Influence risk as informational and influence entropy, if PI doesn't exceeds x_2=1:

$$\text{Risk} = S;\ S(I) = -P(X, x_1, x_2) \cdot \ln[P(X, x_1, x_2)],$$

where $P(X, x_1, x_2), \cdot PP(X, x_1, x_2)$ **are state, influence possibility:**

$$P(X,x_1,x_2)=\frac{1}{\sqrt{2\pi}\sigma(X)}\cdot\left[\begin{array}{l}\frac{-1}{2}\mathrm{erf}\left[\frac{1}{2}\sqrt{2}\frac{(-x_2+\min(X))}{\sigma(X)}\right]\frac{\pi}{2}\sqrt{2}\sigma(X)+\\ +\frac{1}{2}\mathrm{erf}\left[\frac{1}{2}\sqrt{2}\frac{(-x_1+\min(X))}{\sigma(X)}\right]\frac{\pi}{2}\sqrt{2}\sigma(X)\end{array}\right],$$

$$P(X,x_1,x_2)=\frac{1}{\sqrt{2\pi}\sigma(X)}\cdot\left[\begin{array}{l}\frac{-1}{2}\mathrm{erf}\left[\frac{1}{2}\sqrt{2}\frac{(-x_2+1)}{\sigma(X)}\right]\frac{\pi}{2}\sqrt{2}\sigma(X)+\\ +\frac{1}{2}\mathrm{erf}\left[\frac{1}{2}\sqrt{2}\frac{(-x_1+1)}{\sigma(X)}\right]\frac{\pi}{2}\sqrt{2}\sigma(X)\end{array}\right].$$

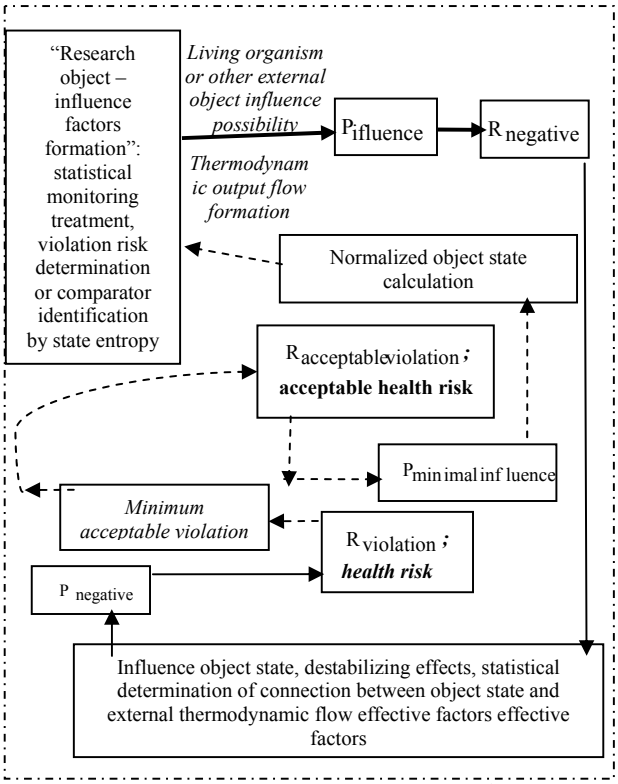


Fig. 3. Risk analysis problems with CES realization: → direction and sequence of the 1st problem solving; - - - ▶ ranking problem decision concerning influence object

Ecological security parameters were calculated with the help of Mathcad 2001 according to influence objects pollution indices (PI). The objects are refuse dump and surface water according to the system (4).

Research objects	Determined parameters
Microbiological factors-B1 (MK)	max(M, K) = 4.3 σ(MK) = 1.048 PP(MK, x1, x2) = 0.292 P(MK, x1, x2) = 0.034 S = 0.362 or *Risk* SS = 0.36 or $S(I)$
Organoleptic factors -B2 (L_1–L_6)	max(L1, L2, L3, L4, L5, L6) = 29.75
Toxicological factors B3 (N_1–N_2)	Values21 := stack(L1, L2, L3, L4, L5, L6) σ(Values21) = 4.714 P(Values, x1, x2) = 0.075 ln((P(Values, x1, x2))) = −2.596 PP(Values21, x1, x2) = 0.074 SS = 0.193 max(N1, N2) = 25 Values31 := stack(N1, N2) σ(Values31) = 10.002 P(Values2, x1, x2) = 0.035 ln((P(Values2, x1, x2))) = −3.344 PP(Values21, x1, x2) = 0.074 SS = 0.193
Polluting anions (B_1), Heavy metals (B_2), Organic matter (B_3)	max(stack(B1, B2, B3)) = 350 Values3 := stack(B1, B2, B3) σ(Values3) = 78.022 P(Values3, x1, x2) = 4.531×10^{-3} ln((P(Values3, x1, x2))) = −5.397 PP(Values3, x1, x2) = 4.531×10^{-3} S1 = 0.024

So water indices have the least probability to stay within norm limits. Organic matter and heavy metals will have the most influence in the case of norm excess. This is confirmed by situation analysis at the refuse dump:

X := Values3 x1 = 0 x2 := max(B1, B2, B3) mean(X) = 33.985 σ(X) = 78.022 x2 = 350

PP(Values3, x1, x2) = 0.592 P(Values3, x1, x2) = 0.445 x2 = 350 PP(Values3, x1, x2) = 0.592

S1 := −PP(Values3, x1, x2)·ln((PP(Values3, x1, x2))) ln((PP(Values3, x1, x2))) = −0.524

S1 = 0.31

Each pollutant block on the basis of corporative approach showed the following results. The thermodynamic analysis results were taken into account:

PP(B1, x1, x2) = 0.438 PP(B2, x1, x2) = 4.969×10^{-3}

PP(B3, x1, x2) = 3.673×10^{-3}

SS(B1) := −PP(B1, x1, x2)·ln((PP(B1, x1, x2))) ln((PP(B1, x1, x2))) = −0.826

SS(B1) = 0.362

SS(B2) := −PP(B2, x1, x2)·ln((PP(B2, x1, x2))) ln((PP(B2, x1, x2))) = −5.304

SS(B2) = 0.026

SS(B3) := −PP(B3, x1, x2)·ln((PP(B3, x1, x2))) ln((PP(B3, x1, x2))) = −5.607

SS(B3) = 0.021

Heavy metals, dangerous organic matters, anions most probably have an effect within normal PI according to the results above. Transformation processes possibility is foreseen according to their relatively big entropy rates. Statistical processing has shown water properties change

dependence on refuse dump ingredients' effect probability. The same dependence was observed between effects in influence system (fig. 4).

N=4	Regression Summary for Dependent Variable: microbiol (S R= ,48593048 R?= ,23612844 Adjusted R?= ----- F(1,2)=,61824 p<,51407 Std.Error of estimate: ,71622				
	Beta	Std.Err. of Beta	B	Std.Err. of B	t(2)
Intercept			1,667948	0,451186	3,696807
BM	0,485930	0,618010	0,002539	0,003229	0,786283

Variable	Correlations (Spreadsheet6sta.sta)			
	anion	BM	organic	microbiol
anion	1,000000	0,611485	-0,608468	0,079973
BM	0,611485	1,000000	-0,171451	0,485930
organic	-0,608468	-0,171451	1,000000	-0,532610
microbiol	0,079973	0,485930	-0,532610	1,000000

N=4	Regression Summary for Dependent Variable: toxcolog (Spr R= ,99843848 R?= ,99687940 Adjusted R?= ,99063819 F(2,1)=159,73 p<,05586 Std.Error of estimate: ,65584				
	Beta	Std.Err. of Beta	B	Std.Err. of B	t(1)
Intercept			13,82016	0,512073	26,9887
organic	-0,799504	0,056702	-0,26831	0,019029	-14,1001
BM	-0,750641	0,056702	-0,03973	0,003001	-13,2384

N=20	Regression Summary for Dependent Variable: PPI R= ,61181109 R?= ,37431281 Adjusted R?= ,30070255 F(2,17)=5,0851 p<,01858 Std.Error of estimate: ,25121			
	Beta	Std.Err. of Beta	B	Std.Err. of B
Intercept			0,442659	0,115734
Ind(mikrobiol)	-0,565826	0,193112	-0,003616	0,001234
IdZ(tocsical)	-0,306203	0,193112	-0,008937	0,005636

Fig. 4. Statistical influence results of certain pollution source upon water quality parameters

IV. Conclusion

A Theoretical And Practical Work Significance

Main research areas were determined by corporative approach introduction into ecological monitoring system. The aim was to determine CES components definite state estimation and the possibility of making managerial decision. As a result ecological, social and economic optimal system equilibrium and their harmonious progress is obtained. Using risk theory allows conforming CES state determination results obtained with the help of thermodynamical functions and with ecological security risk method. It also allows to give information about population health danger due to the heath risk estimation. Risk analysis gives a new point of view upon ranking in ecology, hygiene and sanitation

B Summary

Corporative approach appropriateness for organoleptic water properties quality risk estimation has been determined. The case of chemical factors influence based on the example of water objects analysis has been considered. This was made by advantages estimation of risk analysis for environment objects quality, “environment—human health” homeostasis disbalance possibility estimation and by health risk estimation application in rate setting. The results of analytical research and water state risk analysis calculation application are as follows:

1. CES state determination algorithm (see fig. 1) has been devised according to the CES and their components thermodynamic (entropic probabilistic) nature (eq.1-7). Ecological hygienic health risk scheme and chemical factor permissible level have provided according to CES realization and minimal health risk estimation conditions (see fig. 2).
2. Complex probability thermodynamic system (see fig.3) for source and influence object ecological state estimation has been determined.

Suggested corporative approach realization appropriateness was shown on the basis of example of ecologically dangerous territory state analysis. The analysis aim is to determine ecological danger points and to work out the most constructive managerial decision. The most probable influence factors and thermodynamically conditioned negative influence processes (eq.10, fig. 3) determination is the analysis possibility condition.

References

[1] Ragozin A.L. Obschie zakonomernosti formirovaniya i kolichestvennaya ocenka prirodnyh riskov na territorii Rossii. //Voprosy analiza riska. – 1999. – T.1. – № 2–4. – P. 28-47

[2] Aziev R.G., Kuz'min I.I., Men'shikov V.V. Kolichestvennyi podhod k ocenke bezopasnosti. – M.: izd-vo MGU, 1989.

[3] Menshikov V.V. Konceptualnye osnovy ocenki ekologicheskogo riska. – M.: MNEPU, 2001. – 44 s.

[4] Holton, Glyn A. Defining Risk // Financial Analysts Journal. – 2004. – 60 (6). – P. 19-25.

[5] Kozulya T. V. Teoretichni aspekti stvorennya korporativnoї sistemi ekologichnogo upravlinnya. //Optiko-elektronni informaciino-energetichni tehnologiї. – 2005. – №2(10). – S 193-197.

[6] Kozulya T. V., Sharonova N. V. Modelirovanie struktury i identifikaciya sostoyaniya korporativnoi ekologicheskoi sistemy (KES) // Problemi informaciinih tehnologii. – 2007. – № 1. – S. 178–185.

[7] Prigojin I. R., Stengers I. Poryadok iz haosa. – M.: Nauka, 1986.– 432 p.

[8] Problemy ocenki riska zdorov'yu naseleniya ot vozdeistviya faktorov okrujayuschei sredy / Pod red. Yu. A. Rahmanina, G. G. Onischenko. – M.: NII ECh i GOS, 2004. – P. 157–160.

[9] Unificirovannye metody sbora dannyh, analiza i ocenki zabolevaemosti naseleniya s uchetom kompleksnogo deistviya faktorov okrujayuschei sredy: Metodicheskie rekomendacii. – M., 1996.

[10] Petrov A. N., Zhilyakov A. M., Druzhinin M. V. Ocenka riska zdorov'yu pri opredelenii sanepidblagopoluchiya naseleniya //Gigiena i sanitariya. – 2006. – № 5. – P. 49-50.

Kozulia T. V. Chemic production engineer in National Technical University “Kharkov Polytechnic Institute”. Candidate of geographical science, Kharkov National University after N. V. Karazin, Ukraine, 1999. Major field of study: nonorganic substance technology.
Research assistant, Scientific Technical Complex “Monocrystal Institute”, Kharkov, Ukraine. Post-graduate, Kharkov National University after N. V. Karazin, Ukraine. Associate professor, Kharkov National University after N. V. Karazin, Ukraine. Associate professor, Automatic Operating Systems Department, Informatics and Management School, National Technical

University “Kharkov Polytechnic Institute”, Kharkov, Ukraine. Scientific interests: decision making in the system of ecological monitoring, modeling physical chemical processes in natural systems.

Sharonova N. V. Applied mathematics in Kharkov National University of Radio Electronics. Candidate of technical science, Kharkov National University of Radio Electronics, Ukraine, 1983. Doctor of technical science, Kharkov National University of Radio Electronics, Ukraine, 1993. Major field of study: applied mathematics. Research assistant, Kharkov National University of Radio Electronics. Pro-rector, Humanitarian Academy. Professor, Automatic Operating Systems Department, Informatics and Management School, National Technical University “Kharkov Polytechnic Institute”, Kharkov, Ukraine. Head of Computer Intellectual System Department, National Technical University “Kharkov Polytechnic Institute”, Kharkov, Ukraine. Scientific interests: applied mathematics, mathematical modeling, computer systems, computer linguistics.

Kozulia M. M. Student of the Kharkov National Univercity named after V.N. Karazin, Kharkov, Department of Physics & Technics.

Preparation of Papers for IEEE TRANSACTIONS and JOURNALS (May 2007)

First A. Author, Second B. Author, Jr., and Third C. Author, *Member, IEEE*

***Abstract*—These instructions give you guidelines for preparing papers for IEEE TRANSACTIONS and JOURNALS. Use this document as a template if you are using Microsoft *Word* 6.0 or later. Otherwise, use this document as an instruction set. The electronic file of your paper will be formatted further at IEEE. Define all symbols used in the abstract. Do not cite references in the abstract. Do not delete the blank line immediately above the abstract; it sets the footnote at the bottom of this column.**

***Index Terms*—About four key words or phrases in alphabetical order, separated by commas. For a list of suggested keywords, send a blank e-mail to keywords@ieee.org or visit** http://www.ieee.org/organizations/pubs/ani_prod/keywrd98.txt

I. INTRODUCTION

THIS document is a template for Microsoft *Word* versions 6.0 or later. If you are reading a paper or PDF version of this document, please download the electronic file, TRANS-JOUR.DOC, from the IEEE Web site at http://www.ieee.org/web/publications/authors/transjnl/index.html so you can use it to prepare your manuscript. If you would prefer to use LATEX, download IEEE's LATEX style and sample files from the same Web page. Use these LATEX files for formatting, but please follow the instructions in TRANS-JOUR.DOC or TRANS-JOUR.PDF.

If your paper is intended for a *conference*, please contact your conference editor concerning acceptable word processor formats for your particular conference.

When you open TRANS-JOUR.DOC, select "Page Layout" from the "View" menu in the menu bar (View | Page Layout), which allows you to see the footnotes. Then, type over sections of TRANS-JOUR.DOC or cut and paste from another document and use markup styles. The pull-down style menu is at the left of the Formatting Toolbar at the top of your *Word* window (for example, the style at this point in the document is "Text"). Highlight a section that you want to designate with a certain style, then select the appropriate name on the style menu. The style will adjust your fonts and line spacing. **Do not change the font sizes or line spacing to squeeze more text into a limited number of pages.** Use italics for emphasis; do not underline.

To insert images in *Word*, position the cursor at the insertion point and either use Insert | Picture | From File or copy the image to the Windows clipboard and then Edit | Paste Special | Picture (with "float over text" unchecked).

IEEE will do the final formatting of your paper. If your paper is intended for a conference, please observe the conference page limits.

II. PROCEDURE FOR PAPER SUBMISSION

A. Review Stage

Please check with your editor on whether to submit your manuscript as hard copy or electronically for review. If hard copy, submit photocopies such that only one column appears per page. This will give your referees plenty of room to write comments. Send the number of copies specified by your editor (typically four). If submitted electronically, find out if your editor prefers submissions on disk or as e-mail attachments.

If you want to submit your file with one column electronically, please do the following:

--First, click on the View menu and choose Print Layout.

--Second, place your cursor in the first paragraph. Go to the Format menu, choose Columns, choose one column Layout, and choose "apply to whole document" from the dropdown menu.

--Third, click and drag the right margin bar to just over 4 inches in width.

Manuscript received October 9, 2001. (Write the date on which you submitted your paper for review.) This work was supported in part by the U.S. Department of Commerce under Grant BS123456 (sponsor and financial support acknowledgment goes here). Paper titles should be written in uppercase and lowercase letters, not all uppercase. Avoid writing long formulas with subscripts in the title; short formulas that identify the elements are fine (e.g., "Nd–Fe–B"). Do not write "(Invited)" in the title. Full names of authors are preferred in the author field, but are not required. Put a space between authors' initials.

F. A. Author is with the National Institute of Standards and Technology, Boulder, CO 80305 USA (corresponding author to provide phone: 303-555-5555; fax: 303-555-5555; e-mail: author@ boulder.nist.gov).

S. B. Author, Jr., was with Rice University, Houston, TX 77005 USA. He is now with the Department of Physics, Colorado State University, Fort Collins, CO 80523 USA (e-mail: author@lamar.colostate.edu).

T. C. Author is with the Electrical Engineering Department, University of Colorado, Boulder, CO 80309 USA, on leave from the National Research Institute for Metals, Tsukuba, Japan (e-mail: author@nrim.go.jp).

The graphics will stay in the "second" column, but you can drag them to the first column. Make the graphic wider to push out any text that may try to fill in next to the graphic.

B. *Final Stage*

When you submit your final version (after your paper has been accepted), print it in two-column format, including figures and tables. You must also send your final manuscript on a disk, via e-mail, or through a Web manuscript submission system as directed by the society contact. You may use *Zip* or CD-ROM disks for large files, or compress files using *Compress, Pkzip, Stuffit,* or *Gzip.*

Also, send a sheet of paper or PDF with complete contact information for all authors. Include full mailing addresses, telephone numbers, fax numbers, and e-mail addresses. This information will be used to send each author a complimentary copy of the journal in which the paper appears. In addition, designate one author as the "corresponding author." This is the author to whom proofs of the paper will be sent. Proofs are sent to the corresponding author only.

C. *Figures*

Format and save your graphic images using a suitable graphics processing program that will allow you to create the images as PostScript (PS), Encapsulated PostScript (EPS), or Tagged Image File Format (TIFF), sizes them, and adjusts the resolution settings. If you created your source files in one of the following you will be able to submit the graphics without converting to a PS, EPS, or TIFF file: Microsoft Word, Microsoft PowerPoint, Microsoft Excel, or Portable Document Format (PDF).

D. *Electronic Image Files (Optional)*

Import your source files in one of the following: Microsoft Word, Microsoft PowerPoint, Microsoft Excel, or Portable Document Format (PDF); you will be able to submit the graphics without converting to a PS, EPS, or TIFF files. Image quality is very important to how yours graphics will reproduce. Even though we can accept graphics in many formats, we cannot improve your graphics if they are poor quality when we receive them. If your graphic looks low in quality on your printer or monitor, please keep in mind that cannot improve the quality after submission.

If you are importing your graphics into this Word template, please use the following steps:

Under the option EDIT select PASTE SPECIAL. A dialog box will open, select paste picture, then click OK. Your figure should now be in the Word Document.

If you are preparing images in TIFF, EPS, or PS format, note the following. High-contrast line figures and tables should be prepared with 600 dpi resolution and saved with no compression, 1 bit per pixel (monochrome), with file names in the form of "fig3.tif" or "table1.tif."

Photographs and grayscale figures should be prepared with 300 dpi resolution and saved with no compression, 8 bits per pixel (grayscale).

Sizing of Graphics

Most charts graphs and tables are one column wide (3 1/2 inches or 21 picas) or two-column width (7 1/16 inches, 43 picas wide). We recommend that you avoid sizing figures less than one column wide, as extreme enlargements may distort your images and result in poor reproduction. Therefore, it is better if the image is slightly larger, as a minor reduction in size should not have an adverse affect the quality of the image.

Size of Author Photographs

The final printed size of an author photograph is exactly 1 inch wide by 1 1/4 inches long (6 picas × 7 1/2 picas). Please ensure that the author photographs you submit are proportioned similarly. If the author's photograph does not appear at the end of the paper, then please size it so that it is proportional to the standard size of 1 9/16 inches wide by 2 inches long (9 1/2 picas × 12 picas). JPEG files are only accepted for author photos.

How to create a PostScript File

First, download a PostScript printer driver from http://www.adobe.com/support/downloads/pdrvwin.htm (for Windows) or from http://www.adobe.com/support/downloads/ pdrvmac.htm (for Macintosh) and install the "Generic PostScript Printer" definition. In *Word,* paste your figure into a new document. Print to a file using the PostScript printer driver. File names should be of the form "fig5.ps." Use Open Type fonts when creating your figures, if possible. A listing of the acceptable fonts are as follows: Open Type Fonts: Times Roman, Helvetica, Helvetica Narrow, Courier, Symbol, Palatino, Avant Garde, Bookman, Zapf Chancery, Zapf Dingbats, and New Century Schoolbook.

Print Color Graphics Requirements

IEEE accepts color graphics in the following formats: EPS, PS, TIFF, Word, PowerPoint, Excel, and PDF. The resolution of a RGB color TIFF file should be 400 dpi.

When sending color graphics, please supply a high quality hard copy or PDF proof of each image. If we cannot achieve a satisfactory color match using the electronic version of your files, we will have your hard copy scanned. Any of the files types you provide will be converted to RGB color EPS files.

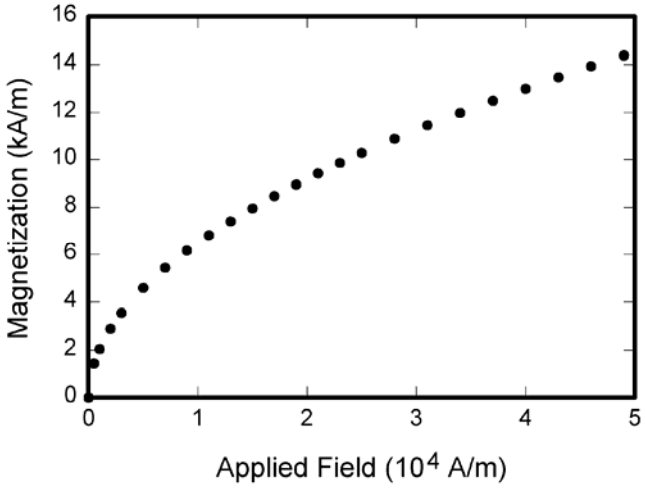


Fig. 1. Magnetization as a function of applied field. Note that "Fig." is abbreviated. There is a period after the figure number, followed by two spaces. It is good practice to explain the significance of the figure in the caption.

Web Color Graphics

IEEE accepts color graphics in the following formats: EPS, PS, TIFF, Word, PowerPoint, Excel, and PDF. The resolution of a RGB color TIFF file should be at least 400 dpi.

Your color graphic will be converted to grayscale if no separate grayscale file is provided. If a graphic is to appear in print as black and white, it should be saved and submitted as a black and white file. If a graphic is to appear in print or on IEEE Xplore in color, it should be submitted as RGB color.

Graphics Checker Tool

The IEEE Graphics Checker Tool enables users to check graphic files. The tool will check journal article graphic files against a set of rules for compliance with IEEE requirements. These requirements are designed to ensure sufficient image quality so they will look acceptable in print. After receiving a graphic or a set of graphics, the tool will check the files against a set of rules. A report will then be e-mailed listing each graphic and whether it met or failed to meet the requirements. If the file fails, a description of why and instructions on how to correct the problem will be sent. The IEEE Graphics Checker Tool is available at http://graphicsqc.ieee.org/

For more Information, contact the IEEE Graphics H-E-L-P Desk by e-mail at graphics@ieee.org. You will then receive an e-mail response and sometimes a request for a sample graphic for us to check.

E. Copyright Form

An IEEE copyright form should accompany your final submission. You can get a .pdf, .html, or .doc version at http://www.ieee.org/copyright. Authors are responsible for obtaining any security clearances.

TABLE I
UNITS FOR MAGNETIC PROPERTIES

Symbol	Quantity	Conversion from Gaussian and CGS EMU to SI [a]
Φ	magnetic flux	1 Mx $\rightarrow 10^{-8}$ Wb = 10^{-8} V·s
B	magnetic flux density, magnetic induction	1 G $\rightarrow 10^{-4}$ T = 10^{-4} Wb/m^2
H	magnetic field strength	1 Oe $\rightarrow 10^3/(4\pi)$ A/m
m	magnetic moment	1 erg/G = 1 emu $\rightarrow 10^{-3}$ A·m^2 = 10^{-3} J/T
M	magnetization	1 erg/(G·cm^3) = 1 emu/cm^3 $\rightarrow 10^3$ A/m
$4\pi M$	magnetization	1 G $\rightarrow 10^3/(4\pi)$ A/m
σ	specific magnetization	1 erg/(G·g) = 1 emu/g $\rightarrow$ 1 A·m^2/kg
j	magnetic dipole moment	1 erg/G = 1 emu $\rightarrow 4\pi \times 10^{-10}$ Wb·m
J	magnetic polarization	1 erg/(G·cm^3) = 1 emu/cm^3 $\rightarrow 4\pi \times 10^{-4}$ T
χ, κ	susceptibility	1 $\rightarrow 4\pi$
χ_ρ	mass susceptibility	1 cm^3/g $\rightarrow 4\pi \times 10^{-3}$ m^3/kg
μ	permeability	1 $\rightarrow 4\pi \times 10^{-7}$ H/m = $4\pi \times 10^{-7}$ Wb/(A·m)
μ_r	relative permeability	$\mu \rightarrow \mu_r$
w, W	energy density	1 erg/cm^3 $\rightarrow 10^{-1}$ J/m^3
N, D	demagnetizing factor	1 $\rightarrow 1/(4\pi)$

Vertical lines are optional in tables. Statements that serve as captions for the entire table do not need footnote letters.

[a]Gaussian units are the same as cgs emu for magnetostatics; Mx = maxwell, G = gauss, Oe = oersted; Wb = weber, V = volt, s = second, T = tesla, m = meter, A = ampere, J = joule, kg = kilogram, H = henry.

III. MATH

If you are using *Word*, use either the Microsoft Equation Editor or the *MathType* add-on (http://www.mathtype.com) for equations in your paper (Insert | Object | Create New | Microsoft Equation *or* MathType Equation). "Float over text" should *not* be selected.

IV. UNITS

Use either SI (MKS) or CGS as primary units. (SI units are strongly encouraged.) English units may be used as secondary units (in parentheses). **This applies to papers in data storage.** For example, write "15 Gb/cm^2 (100 Gb/in^2)." An exception is when English units are used as identifiers in trade, such as "3½-in disk drive." Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity in an equation.

The SI unit for magnetic field strength H is A/m. However, if you wish to use units of T, either refer to magnetic flux density B or magnetic field strength symbolized as $\mu_0 H$. Use the center dot to separate compound units, e.g., "A·m^2."

V. Helpful Hints

A. Figures and Tables

Because IEEE will do the final formatting of your paper, you do not need to position figures and tables at the top and bottom of each column. In fact, all figures, figure captions, and tables can be at the end of the paper. Large figures and tables may span both columns. Place figure captions below the figures; place table titles above the tables. If your figure has two parts, include the labels "(a)" and "(b)" as part of the artwork. Please verify that the figures and tables you mention in the text actually exist. **Please do not include captions as part of the figures. Do not put captions in "text boxes" linked to the figures. Do not put borders around the outside of your figures.** Use the abbreviation "Fig." even at the beginning of a sentence. Do not abbreviate "Table." Tables are numbered with Roman numerals.

Color printing of figures is available, but is billed to the authors. Include a note with your final paper indicating that you request and will pay for color printing. Do not use color unless it is necessary for the proper interpretation of your figures. If you want reprints of your color article, the reprint order should be submitted promptly. There is an additional charge for color reprints. **Please note that many IEEE journals now allow an author to publish color figures on Xplore and black and white figures in print. Contact your society representative for specific requirements.**

Figure axis labels are often a source of confusion. Use words rather than symbols. As an example, write the quantity "Magnetization," or "Magnetization *M*," not just "*M*." Put units in parentheses. Do not label axes only with units. As in Fig. 1, for example, write "Magnetization (A/m)" or "Magnetization ($\mathrm{A \cdot m^{-1}}$)," not just "A/m." Do not label axes with a ratio of quantities and units. For example, write "Temperature (K)," not "Temperature/K."

Multipliers can be especially confusing. Write "Magnetization (kA/m)" or "Magnetization (10^3 A/m)." Do not write "Magnetization (A/m) × 1000" because the reader would not know whether the top axis label in Fig. 1 meant 16000 A/m or 0.016 A/m. Figure labels should be legible, approximately 8 to 12 point type.

B. References

Number citations consecutively in square brackets [1]. The sentence punctuation follows the brackets [2]. Multiple references [2], [3] are each numbered with separate brackets [1]–[3]. When citing a section in a book, please give the relevant page numbers [2]. In sentences, refer simply to the reference number, as in [3]. Do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] shows ... ." Please do not use automatic endnotes in *Word*, rather, type the reference list at the end of the paper using the "References" style.

Number footnotes separately in superscripts (Insert | Footnote).[1] Place the actual footnote at the bottom of the column in which it is cited; do not put footnotes in the reference list (endnotes). Use letters for table footnotes (see Table I).

Please note that the references at the end of this document are in the preferred referencing style. Give all authors' names; do not use "*et al*." unless there are six authors or more. Use a space after authors' initials. Papers that have not been published should be cited as "unpublished" [4]. Papers that have been accepted for publication, but not yet specified for an issue should be cited as "to be published" [5]. Papers that have been submitted for publication should be cited as "submitted for publication" [6]. Please give affiliations and addresses for private communications [7].

Capitalize only the first word in a paper title, except for proper nouns and element symbols. For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [8].

C. Abbreviations and Acronyms

Define abbreviations and acronyms the first time they are used in the text, even after they have already been defined in the abstract. Abbreviations such as IEEE, SI, ac, and dc do not have to be defined. Abbreviations that incorporate periods should not have spaces: write "C.N.R.S.," not "C. N. R. S." Do not use abbreviations in the title unless they are unavoidable (for example, "IEEE" in the title of this article).

D. Equations

Number equations consecutively with equation numbers in parentheses flush with the right margin, as in (1). First use the equation editor to create the equation. Then select the "Equation" markup style. Press the tab key and write the equation number in parentheses. To make your equations more compact, you may use the solidus (/), the exp function, or appropriate exponents. Use parentheses to avoid ambiguities in denominators. Punctuate equations when they are part of a sentence, as in

$$\int_0^{r_2} F(r,\varphi)\, dr\, d\varphi = [\sigma r_2 / (2\mu_0)] \cdot \int_0^{\infty} \exp(-\lambda\, |z_j - z_i|)\, \lambda^{-1}\, J_1(\lambda r_2)\, J_0(\lambda r_i)\, d\lambda\,. \tag{1}$$

Be sure that the symbols in your equation have been defined before the equation appears or immediately following. Italicize symbols (*T* might refer to temperature,

[1]It is recommended that footnotes be avoided (except for the unnumbered footnote with the receipt date on the first page). Instead, try to integrate the footnote information into the text.

but T is the unit tesla). Refer to "(1)," not "Eq. (1)" or "equation (1)," except at the beginning of a sentence: "Equation (1) is ... ."

E. Other Recommendations

Use one space after periods and colons. Hyphenate complex modifiers: "zero-field-cooled magnetization." Avoid dangling participles, such as, "Using (1), the potential was calculated." [It is not clear who or what used (1).] Write instead, "The potential was calculated by using (1)," or "Using (1), we calculated the potential."

Use a zero before decimal points: "0.25," not ".25." Use "cm^3," not "cc." Indicate sample dimensions as "0.1 cm × 0.2 cm," not "0.1 × 0.2 cm^2." The abbreviation for "seconds" is "s," not "sec." Do not mix complete spellings and abbreviations of units: use "Wb/m^2" or "webers per square meter," not "webers/m^2." When expressing a range of values, write "7 to 9" or "7-9," not "7~9."

A parenthetical statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.) In American English, periods and commas are within quotation marks, like "this period." Other punctuation is "outside"! Avoid contractions; for example, write "do not" instead of "don't." The serial comma is preferred: "A, B, and C" instead of "A, B and C."

If you wish, you may write in the first person singular or plural and use the active voice ("I observed that ..." or "We observed that ..." instead of "It was observed that ..."). Remember to check spelling. If your native language is not English, please get a native English-speaking colleague to carefully proofread your paper.

VI. Some Common Mistakes

The word "data" is plural, not singular. The subscript for the permeability of vacuum μ_0 is zero, not a lowercase letter "o." The term for residual magnetization is "remanence"; the adjective is "remanent"; do not write "remnance" or "remnant." Use the word "micrometer" instead of "micron." A graph within a graph is an "inset," not an "insert." The word "alternatively" is preferred to the word "alternately" (unless you really mean something that alternates). Use the word "whereas" instead of "while" (unless you are referring to simultaneous events). Do not use the word "essentially" to mean "approximately" or "effectively." Do not use the word "issue" as a euphemism for "problem." When compositions are not specified, separate chemical symbols by en-dashes; for example, "NiMn" indicates the intermetallic compound $Ni_{0.5}Mn_{0.5}$ whereas "Ni–Mn" indicates an alloy of some composition Ni_xMn_{1-x}.

Be aware of the different meanings of the homophones "affect" (usually a verb) and "effect" (usually a noun), "complement" and "compliment," "discreet" and "discrete," "principal" (e.g., "principal investigator") and "principle" (e.g., "principle of measurement"). Do not confuse "imply" and "infer."

Prefixes such as "non," "sub," "micro," "multi," and "ultra" are not independent words; they should be joined to the words they modify, usually without a hyphen. There is no period after the "et" in the Latin abbreviation "*et al.*" (it is also italicized). The abbreviation "i.e.," means "that is," and the abbreviation "e.g.," means "for example" (these abbreviations are not italicized).

An excellent style manual and source of information for science writers is [9]. A general IEEE style guide and an *Information for Authors* are both available at http://www.ieee.org/web/publications/authors/transjnl/index.html

VII. Editorial Policy

Submission of a manuscript is not required for participation in a conference. Do not submit a reworked version of a paper you have submitted or published elsewhere. Do not publish "preliminary" data or results. The submitting author is responsible for obtaining agreement of all coauthors and any consent required from sponsors before submitting a paper. IEEE Transactions and Journals strongly discourage courtesy authorship. It is the obligation of the authors to cite relevant prior work.

The Transactions and Journals Department does not publish conference records or proceedings. The Transactions does publish papers related to conferences that have been recommended for publication on the basis of peer review. As a matter of convenience and service to the technical community, these topical papers are collected and published in one issue of the Transactions.

At least two reviews are required for every paper submitted. For conference-related papers, the decision to accept or reject a paper is made by the conference editors and publications committee; the recommendations of the referees are advisory only. Undecipherable English is a valid reason for rejection. Authors of rejected papers may revise and resubmit them to the Transactions as regular papers, whereupon they will be reviewed by two new referees.

VIII. Publication Principles

The contents of IEEE Transactions and Journals are peer-reviewed and archival. The Transactions publishes scholarly articles of archival value as well as tutorial expositions and critical reviews of classical subjects and topics of current interest.

Authors should consider the following points:

1) Technical papers submitted for publication must advance the state of knowledge and must cite relevant prior work.
2) The length of a submitted paper should be commensurate with the importance, or appropriate to

the complexity, of the work. For example, an obvious extension of previously published work might not be appropriate for publication or might be adequately treated in just a few pages.

3) Authors must convince both peer reviewers and the editors of the scientific and technical merit of a paper; the standards of proof are higher when extraordinary or unexpected results are reported.
4) Because replication is required for scientific progress, papers submitted for publication must provide sufficient information to allow readers to perform similar experiments or calculations and use the reported results. Although not everything need be disclosed, a paper must contain new, useable, and fully described information. For example, a specimen's chemical composition need not be reported if the main purpose of a paper is to introduce a new measurement technique. Authors should expect to be challenged by reviewers if the results are not supported by adequate data and critical details.
5) Papers that describe ongoing work or announce the latest technical achievement, which are suitable for presentation at a professional conference, may not be appropriate for publication in a TRANSACTIONS or JOURNAL.

IX. CONCLUSION

A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

APPENDIX

Appendixes, if needed, appear before the acknowledgment.

ACKNOWLEDGMENT

The preferred spelling of the word "acknowledgment" in American English is without an "e" after the "g." Use the singular heading even if you have many acknowledgments. Avoid expressions such as "One of us (S.B.A.) would like to thank ... ." Instead, write "F. A. Author thanks ... ." **Sponsor and financial support acknowledgments are placed in the unnumbered footnote on the first page, not here.**

REFERENCES

[1] G. O. Young, "Synthetic structure of industrial plastics (Book style with paper title and editor)," in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.

[2] W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.

[3] H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.

[4] B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.

[5] E. H. Miller, "A note on reflector arrays (Periodical style—Accepted for publication)," *IEEE Trans. Antennas Propagat.*, to be published.

[6] J. Wang, "Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication)," *IEEE J. Quantum Electron.*, submitted for publication.

[7] C. J. Kaufman, Rocky Mountain Research Lab., Boulder, CO, private communication, May 1995.

[8] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interfaces (Translation Journals style)," *IEEE Transl. J. Magn.Jpn.*, vol. 2, Aug. 1987, pp. 740–741 [*Dig. 9th Annu. Conf. Magnetics* Japan, 1982, p. 301].

[9] M. Young, *The Techincal Writers Handbook.* Mill Valley, CA: University Science, 1989.

[10] J. U. Duncombe, "Infrared navigation—Part I: An assessment of feasibility (Periodical style)," *IEEE Trans. Electron Devices*, vol. ED-11, pp. 34–39, Jan. 1959.

[11] S. Chen, B. Mulgrew, and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," *IEEE Trans. Neural Networks*, vol. 4, pp. 570–578, Jul. 1993.

[12] R. W. Lucky, "Automatic equalization for digital communication," *Bell Syst. Tech. J.*, vol. 44, no. 4, pp. 547–588, Apr. 1965.

[13] S. P. Bingulac, "On the compatibility of adaptive controllers (Published Conference Proceedings style)," in *Proc. 4th Annu. Allerton Conf. Circuits and Systems Theory*, New York, 1994, pp. 8–16.

[14] G. R. Faulhaber, "Design of service systems with priority reservation," in *Conf. Rec. 1995 IEEE Int. Conf. Communications*, pp. 3–8.

[15] W. D. Doyle, "Magnetization reversal in films with biaxial anisotropy," in *1987 Proc. INTERMAG Conf.*, pp. 2.2-1–2.2-6.

[16] G. W. Juette and L. E. Zeffanella, "Radio noise currents n short sections on bundle conductors (Presented Conference Paper style)," presented at the IEEE Summer power Meeting, Dallas, TX, Jun. 22–27, 1990, Paper 90 SM 690-0 PWRS.

[17] J. G. Kreifeldt, "An analysis of surface-detected EMG as an amplitude-modulated noise," presented at the 1989 Int. Conf. Medicine and Biological Engineering, Chicago, IL.

[18] J. Williams, "Narrow-band analyzer (Thesis or Dissertation style)," Ph.D. dissertation, Dept. Elect. Eng., Harvard Univ., Cambridge, MA, 1993.

[19] N. Kawasaki, "Parametric study of thermal and chemical nonequilibrium nozzle flow," M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.

[20] J. P. Wilkinson, "Nonlinear resonant circuit devices (Patent style)," U.S. Patent 3 624 12, July 16, 1990.

[21] *IEEE Criteria for Class IE Electric Systems* (Standards style), IEEE Standard 308, 1969.

[22] *Letter Symbols for Quantities*, ANSI Standard Y10.5-1968.

[23] R. E. Haskell and C. T. Case, "Transient signal propagation in lossless isotropic plasmas (Report style)," USAF Cambridge Res. Lab., Cambridge, MA Rep. ARCRL-66-234 (II), 1994, vol. 2.

[24] E. E. Reber, R. L. Michell, and C. J. Carter, "Oxygen absorption in the Earth's atmosphere," Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0200 (420-46)-3, Nov. 1988.

[25] (Handbook style) *Transmission Systems for Communications*, 3rd ed., Western Electric Co., Winston-Salem, NC, 1985, pp. 44–60.

[26] *Motorola Semiconductor Data Manual*, Motorola Semiconductor Products Inc., Phoenix, AZ, 1989.

[27] (Basic Book/Monograph Online Sources) J. K. Author. (year, month, day). *Title* (edition) [Type of medium]. Volume (issue). Available: http://www.(URL)

[28] J. Jones. (1991, May 10). Networks (2nd ed.) [Online]. Available: http://www.atm.com

[29] (Journal Online Sources style) K. Author. (year, month). Title. *Journal* [Type of medium]. Volume(issue), paging if given. Available: http://www.(URL)

[30] R. J. Vidmar. (1992, August). On the use of atmospheric plasmas as electromagnetic reflectors. *IEEE Trans. Plasma Sci.* [Online]. *21(3).* pp. 876–880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar

First A. Author (M'76–SM'81–F'87) and the other authors may include biographies at the end of regular papers. Biographies are often not included in conference-related papers. This author became a Member (M) of IEEE in 1976, a Senior Member (SM) in 1981, and a Fellow (F) in 1987. The first paragraph may contain a place and/or date of birth (list place, then date). Next, the author's educational background is listed. The degrees should be listed with type of degree in what field, which institution, city, state, and country, and year degree was earned. The author's major field of study should be lower-cased.

The second paragraph uses the pronoun of the person (he or she) and not the author's last name. It lists military and work experience, including summer and fellowship jobs. Job titles are capitalized. The current job must have a location; previous positions may be listed without one. Information concerning previous publications may be included. Try not to list more than three books or published articles. The format for listing publishers of a book within the biography is: title of book (city, state: publisher name, year) similar to a reference. Current and previous research interests end the paragraph.

The third paragraph begins with the author's title and last name (e.g., Dr. Smith, Prof. Jones, Mr. Kajor, Ms. Hunter). List any memberships in professional societies other than the IEEE. Finally, list any awards and work for IEEE committees and publications. If a photograph is provided, the biography will be indented around it. The photograph is placed at the top left of the biography. Personal hobbies will be deleted from the biography.

Camera-ready was prepared in Kharkov National University of Radio Electronics
Approved for publication: 27.12.2010. Format 60×84 1/8.
Relative printer's sheets: 6,5. Circulation: 300 copies.
Published by SPD FL Andreev K.V.
Lenin Ave, 14, Kharkov, 61166, Ukraine

Рекомендовано Вченою радою Харківського національного
університету радіоелектроніки (протокол № 4 від 27.12.2010)
Підписано до друку 27.12.2010. Формат 60×841/8.
Умов. друк. арк. 6,5 . Тираж 300 прим. Ціна договірна.
Віддруковано у ФОП Андреєв К.В.
61166, Харків, просп. Леніна, 14.