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 (47), September – December 2009 **Published 4 times a year**

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

International Editorial Board:

Local Editorial Board:

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

Matrix Implementation of Moore FSM with Encoding of Collections of Microoperations

A. Barkalov, L. Titarenko, O. Hebda, K. Soldatov

***Abstract* — The method is proposed for reduction of hardware amount in logic circuit of Moore finite state machine. The method is oriented on customized matrix technology. It is based on representation of the next state code as a concatenation of code for class of collection of microoperations and code of the vertex. Such an approach allows elimination of dependence among states and microoperations. As a result, both circuits for generation of input memory functions and microoperations are optimized. An example of the proposed method application is given.**

***Index Terms* — Customized matrices, graph-scheme of algorithm, logic circuit, Moore FSM, pseudoequivalent states.**

I. Introduction

The model of Moore finite state machine (FSM) [1] is often used during the digital control systems realization [2, 3]. The development of microelectronics has led to appearance of different programmable logic devices [4], used for implementing FSM circuits. But in the case of mass production, they use ASIC (Application-Specified Integrated Circuits) [6]. In this case the circuit is implemented using customized matrices using the principle of distributed logic [7].

One of the important problems of FSM synthesis with ASIC is decrease of the chip area occupied by its logic circuit. One of the ways to solve this problem is optimal coding of FSM [2]. However this approach does not allow optimization of the circuit generated output signals. In this work some new optimization method is proposed. It is based on representation of the next state code as a concatenation of codes for class of pseudoequivalent states and vertex where this collection is generated. Such an approach allows reducing of hardware amount in both parts of FSM circuits and does not lead to speed loss. A control algorithm to be implemented is represented by the graph-scheme of algorithms [1].

II. The General Aspects and the Basic Idea of Proposed Method

Let Moore FSM be represented by the structure table (ST) with columns [1]: a_m, $K(a_m)$, a_s, $K(a_s)$, X_h, Φ_h, h. Here a_m is an initial state of FSM; $K(a_m)$ is a code of state $a_m \in A$ of capacity $R = \lceil \log_2 M \rceil$, to code the states the internal variables from the set $T = \{T_1,..,T_R\}$ are used; a_s, $K(a_s)$, are a state of transition and its code respectively; X_h is an input, which determines the transition $\langle a_m, \ a_s \rangle$, and equal to conjunction of some elements (or their complements) of a logic conditions set $X = \{x_1,..,x_L\}$; Φ_h is a set of input memory functions for flip-flops of FSM memory, which are equal to 1 for memory switching from $K(a_m)$ to $K(a_s)$, $\Phi_h \subseteq \Phi = \{\varphi_1,..,\varphi_R\}$; $h = 1, \ \dots, \ H$ is a number of transition. In the column a_m a set of microoperations Y_q is written, which is generated in the state $a_m \in A$, where $Y_q \subseteq Y = \{y_1,..,y_N\}$, $q = 1,...,Q$. This table is a basis to form the system of functions

$$\Phi = \Phi(T, X), \quad (1)$$

$$Y = Y(T), \quad (2)$$

which determines an FSM logic circuit. Systems (1)-(2) describe the matrix model of Moore FSM U_1, shown in Fig.1.

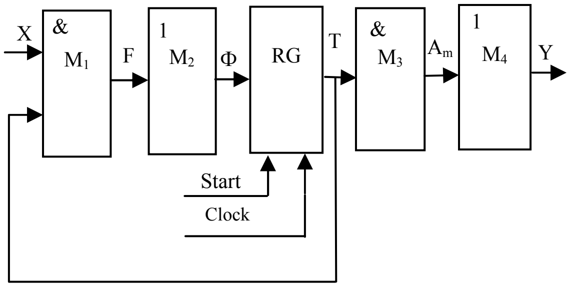


Fig. 1. Matrix implementation of FSM U_1

In FSM U_1 the conjunctive matrix M_1 implements the

Manuscript received October 9, 2009. Matrix Implementation of Moore FSM with Encoding of Collections of Microoperations.

A. Barkalov is with the Institute of Computer Engineering and Electronics, Zielona Góra, Poland (corresponding author to provide phone: 0-683282693; e-mail: A.Barkalov@iie.uz.zgora.pl).

L. Titarenko is with the Institute of Computer Engineering and Electronics, Zielona Góra, Poland (e-mail: L.Titarenko@iie.uz.zgora.pl).

O. Hebda is with the Institute of Computer Engineering and Electronics, Zielona Góra, Poland (e-mail: O.Shapoval@iie.uz.zgora.pl).

K. Soldatov is with the Institute of Computer Engineering Donetsk National Technical University, Donetsk, Ukraine (e-mail: k.soldatov@cs.donntu.donetsk.ua).

system of terms $F = \{F_1,...,F_H\}$; the disjunctive matrix M_2 implements the system (1); the conjunctive matrix M_3 implements the terms A_m ($m = 1,\ldots,M$) corresponding to FSM states; the disjunctive matrix M_4 implements functions (2). The register RG keeps state codes. The matrices M_1 and M_2 forms the block of input memory functions (BIMF) whereas the matrices M_3 and M_4 the block of microoperations (BMO). The area of BIMF can be decreased using the approach of optimal state encoding [8]. It permits to decrease the number of terms in system (1) up to H_0, where H_0 is the number of transitions for equivalent Mealy FSM. The area of BMO can be decreased due to refined state encoding [9]. It is possible some outcome of encoding, when the matrix M_4 is absent. But these methods cannot be used together. In this article the method is proposed permitting mutual area decrease for both blocks of FSM.

One of Moore FSM features is existence of pseudoequivalent states [2], which are the states with the same transitions by the effect of the same inputs. Such states correspond to the control algorithm operator vertices [1], outputs of which are connected with an input of the same vertex.

Let $\Pi_A = \{B_I,..,B_I\}$ be a partition of a set A on classes of pseudoequivalent states. Let us code classes $B_i \in \Pi_A$ by binary codes $K(B_i)$ having R_B bits, where

$$R_B = \lceil \log_2 I \rceil. \qquad (3)$$

Let initial GSA Γ include Q different collections of microoperations (CMO) $Y_q \subseteq Y$. Let us code set Y_q with binary code $K(Y_q)$ having R_Y bits, where

$$R_Y = \lceil \log_2 Q \rceil. \qquad (4)$$

Let $E_1 = \{b_1,\ldots,b_D\}$ be a set of operator vertices from GSA Γ. Let us use the following relation α on this set E_1

$$b_i \alpha b_j \leftrightarrow Y(b_i) = Y(b_j). \qquad (5)$$

In (5), the symbols $Y(b_i), Y(b_j) \subseteq Y$ stand for collections of MO from vertices b_i and b_j ($i, j \in \{1,...,D\}$).The relation α determines the partition $\Pi_\alpha = \{C_1,...,C_\eta\}$. Let us encode each vertex $b_q \in C_j$ by the binary code $K(b_q)$ having

$$R_\alpha = \lceil \log_2 G \rceil \qquad (6)$$

bits. In (6), $G = \max(|C_1|,...,|C_\eta|)$. Let us use variables $z_r \in Z_1$ for this encoding, where $|Z_1| = R_\alpha$. In this case, the code for state $a_m \in A$ can be represented as:

$$K(a_m) = K(Y_q) * K(b_q), \qquad (7)$$

where $b_q \in E_1$ is the operator vertex marked by state $a_m \in A$, $Y_q = Y(b_q)$, and * is the sign of concatenation.

Let us construct the system

$$B = B(A), \qquad (8)$$

which describes the dependence among the classes $B_i \in \Pi_A$ from the states $a_m \in A$. Each function $B_i \in B$ is represented as the following

$$B_i = \mathop{V}_{i=1}^{I} C_{im} A_m (i = 1,...,I), \qquad (9)$$

where the symbol C_{im} stands for Boolean variable equal to 1, $a_m \in B_i$. The proposed matrix implementation of Moore FSM U_2 is shown in Fig. 2.

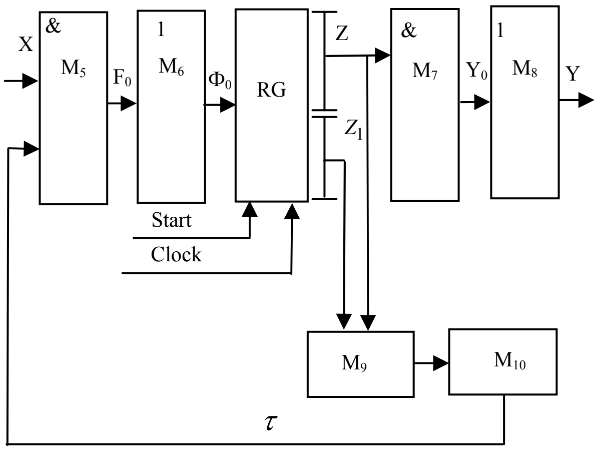


Fig. 2. Matrix implementation of FSM U_2

In FSM U_2, the matrix M_5 implements the system of terms F_0 corresponding to rows of transformed table of transitions and depending on logical conditions $x_l \in X$ and additional variables $\tau_r \in \tau$, used for encoding the classes $B_i \in \Pi_A$, where $|\tau| = R_B$. The matrix M_6 implements the input memory functions

$$\Phi_0 = \Phi_0(\tau, X), \qquad (10)$$

The system (10) includes $R_Y + R_\alpha$ functions; it is the number of flip-flops from RG. The matrix M_7 implements terms Y_0, entering the system $y_n \in Y$ and depending from variables $z_r \in Z$, where $|z| = R_Y$. The matrix M_8 implements functions $y_n \in Y$, depending on terms $\Delta_q \in Y_0$. The matrix M_9 implements the terms A_0 from (9), whereas the matrix M_{10} functions $\tau_r \in \tau$, used for encoding classes $B_i \in \Pi_A$, where $|\tau| = R_B$.

Matrices M_5 and M_6 form the block BIMF, the matrices M_7 and M_8 form the block BMO implementing the functions

$$Y = Y(Z). \qquad (11)$$

Matrices M_9 and M_{10} form the block of code transformer (BCT) generating functions

$$\tau = \tau(z, z_1) . \qquad (12)$$

There are some positive features in the proposed method. Now codes of collections of microoperations do not depend on state codes. It allows encoding of collections $Y_q \subseteq Y$ minimizing the area of BMO. The number of rows in the table of transitions for FSM U_2 is always equal to H_0. It allows such their encoding that diminishes the area occupied by BIMF. As it was mentioned, it is enough

$$R_A = \lceil \log_2 M \rceil \qquad (13)$$

variables for state encoding in case of FSM U_1. The main drawback of U_2 is increase of the number of inputs for BIMF if the following condition is true:

$$R_Y + R_\alpha > R_A . \qquad (14)$$

Besides, the model U_2 includes the block BCT, which requires some area of the chip. But these drawbacks are compensated by area decrease for blocks BIMF and BMO in comparison with the model U_1.

III. Proposed Synthesis Method for Moore FSM

In this work a method of Moore FSM U_2 synthesis using a GSA Γ is proposed. The method includes the next stages:

1. Marking of the GSA Γ and creation of the state set A.
2. Partition of the set A on classes of pseudoequivalent states.
3. Coding of microoperation collections $Y_q \subseteq Y$.
4. Construction of the partition Π_α and encoding of operator vertices $b_q \in E_1$.
5. Encoding the classes $B_i \in \Pi_A$.
6. Construction of transformed table of transitions.
7. Construction of system (12) by the table of BCT.
8. Implementation of matrices $M_5 - M_{10}$.

For the first step implementation the known method [1] is used, when every operator vertex is marked by a unique state. The second step is trivially done by the use of pseudoequivalent states' definition [2]. Remind, that states a_m, $a_s \in A$ are named pseudoequivalent, if marked by them operator vertices of GSA are connected with the input of the same vertex.

The main goal of the third step is maximum decrease for the number of terms in system Y_0. In the best case, each microoperation $y_n \in Y$ is represented by a single term and the matrix M_8 is absent [1]. The fourth step is executed on the base of (5). The codes of states $a_m \in A$ are determined using the formula (7). Classes $B_i \in \Pi_A$ are encoded in such a manner that the number of terms in (12) is maximally decreased. It is reduced to the well-known task of symbolic encoding [3].

The transformed table of transitions includes the columns B_i, $K(B_i)$, a_s, $K(a_s)$, X_k, Φ_k, h. Here $\Phi_h \subseteq \Phi_0$ is a collection of input memory functions equal to 1 to write the code $K(a_s)$ into the register; $h = 1,\ldots,H_0$ is the number of transition. The table of BCT includes the columns a_m, $K(a_m)$, B_i, $K(B_i)$, τ_m, m. Here $\tau_m \subseteq \tau$ is the collection of variables equal to 1 into the code $K(B_i)$ from the m-th line of the table, where $m = 1,\ldots,M$. The last step is discussed in the proposed example.

IV. Example of Application for Proposed Method

Let the symbol $U_i(\Gamma_j)$ means that the GSA Γ_j is interpreted by the model U_i ($\mathrm{i} = 1,2$). Let us discuss the example of design for Moore FSM $U_2(\Gamma_1)$, where GSA Γ_1 is shown in Fig. 3.

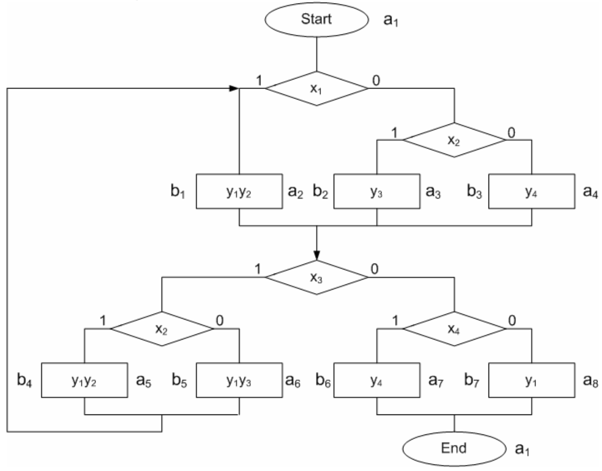


Fig. 3. Initial graph-scheme of algorithm Γ_1

It can be found from GSA Γ_1, that $A = \{a_1,...,a_8\}$, $M = 8$, and $R_A = 3$. There is the partition $\Pi_A = \{B_1,...,B_4\}$, where $B_1 = \{a_1\}$, $B_2 = \{a_2, a_3, a_4\}$, $B_3 = \{a_5, a_6\}$, $B_4 = \{a_7, a_8\}$. It gives us $I = 4$, $R_B = 2$, $\tau = \{\tau_1, \tau_2\}$. There are five different collections of microoperations in GSA Γ_1: $Y_1 = 0$, $Y_2 = \{y_1, y_2\}$, $Y_3 = \{y_3\}$, $Y_4 = \{y_4\}$, $Y_5 = \{y_1, y_3\}$. To encode them, it is enough $R_Y = 3$ variables from the set $Z = \{z_1, z_2, z_3\}$. Let us encode the collections $Y_q \subseteq Y$ as it is shown in Fig. 4.

Z_3 \ Z_1Z_2	00	01	11	10
0	Y_1	*	Y_4	Y_2
1	*	*	Y_3	Y_5

Fig. 4. Codes of collections of microoperations $U_2(\Gamma_1)$

The following system of equations can be obtained using Fig. 3 and Fig.4:

$$\begin{aligned} y_1 &= Y_2 \vee Y_5 = z_2\overline{z_3} = \Delta_1; \\ y_2 &= Y_2 = \overline{z_1}z_2\overline{z_3} = \Delta_2; \\ y_3 &= Y_3 \vee Y_5 = z_1 = \Delta_3; \\ y_4 &= Y_4 = \overline{z_1}z_3 = \Delta_4; \end{aligned} \tag{15}$$

The partition Π_α includes four classes: $C_1 = \{b_1, b_4, b_7\}$, $C_2 = \{b_2\}$, $C_3 = \{b_3, b_6\}$, $C_4 = \{b_5\}$. It gives $G = 3$, $R_\alpha = 2$, $Z_1 = \{z_4, z_5\}$. There is the following system (8) in our example:

$$\begin{aligned} &B_1 = A_1; B_2 = A_2 \vee A_3 \vee A_4; \\ &B_3 = A_5 \vee A_6; B_4 = A_7 \vee A_8. \end{aligned} \tag{16}$$

Let us encode the vertices $b_q \in E_1$ in such a manner that the state codes (7) are determined from Fig.5.

Z₁Z₂Z₃ \ Z₄Z₅	000	001	011	010	110	111	101	100
00	a_1	*	a_4	a_2	*	a_3	*	*
01	*	*	*	a_5	a_6	*	*	*
11	*	*	*	*	*	*	*	*
10	*	*	a_8	a_7	*	*	*	*

Fig. 5. State codes for Moore FSM $U_2(\Gamma_1)$

The following codes can be found from the Karnaugh map (Fig. 5):

$$\begin{aligned} &B_1 = \overline{z_2}; B_2 = z_2\overline{z_4}\,\overline{z_5}; \\ &B_3 = z_5; B_4 = z_4. \end{aligned} \tag{17}$$

Let us encode the classes $B_i \in \Pi_A$ in the following manner $K(B_1) = 01$, $K(B_2) = 00$, $K(B_3) = 10$, $K(B_4) = 11$. The following system can be derived from these codes:

$$\begin{aligned} \tau_1 &= B_3 \vee B_4 = z_5 \vee z_4; \\ \tau_2 &= B_1 \vee B_4 = \overline{z_2} \vee z_4. \end{aligned} \tag{18}$$

The system (18) determines the block BCT, where the matrix M_9 is absent.

Let us construct the system of generalized formulae of transitions for GSA Γ_1:

$$\begin{aligned} &B_1 \to x_1a_2 \vee \overline{x_1}x_2a_3 \vee \overline{x_1}\,\overline{x_2}a_4; \\ &B_2 \to x_3x_2a_5 \vee x_3\overline{x_2}a_6 \vee \overline{x_3}x_4a_7 \vee \overline{x_3}\,\overline{x_4}a_8; \\ &B_3 \to a_2; B_4 \to a_1. \end{aligned} \tag{19}$$

This system together with state codes from Fig. 5 leads to the transformed table of transitions for FSM $U_2(\Gamma_1)$, having $H_0 = 9$ lines (Table 1).

TABLE I
TRANSFORMED TABLE OF TRANSITIONS FOR FSM $U_2(\Gamma_1)$

B_i	$K(B_i)$	a_s	$K(a_s)$	X_h	Φ_h	h
B_1	01	a_2	01000	x_1	D_2	1
		a_3	11100	$\overline{x_1}x_2$	$D_1D_2D_3$	2
		a_4	01100	$\overline{x_1}\,\overline{x_2}$	D_2D_3	3
B_2	00	a_5	01001	x_3x_2	D_2D_5	4
		a_6	11001	$x_3\overline{x_2}$	$D_1D_2D_5$	5
		a_7	01010	$\overline{x_3}x_4$	D_2D_4	6
		a_8	01110	$\overline{x_3}\,\overline{x_4}$	$D_2D_3D_4$	7
B_3	10	a_2	01000	1	D_2	8
B_4	11	a_1	00000	1	-	9

This table is used to derive the system (10). For example, the following functions can be found from Table 1: $D_1 = F_2 \vee F_5 = \overline{\tau_1}\tau_2\overline{x_1}x_2 \vee \overline{\tau_1}\,\overline{\tau_2}x_3\overline{x_2}$; $D_2 = F_2 \vee ... \vee F_8$; $D_3 = F_2 \vee F_3 \vee F_7$; $D_4 = F_6 \vee F_7$; $D_5 = F_4 \vee F_5$. In the case of matrix implementation, there is no need in minimizing these functions. The table for BCT is absent on our example because the system (18) determines the functions (12). Let us find the areas for matrices M_5 – M_{10}, determined as the product for the numbers of inputs and outputs of the matrix. From system of functions we can find the following areas of matrices: $S(M_5) = 2(4+2)*9 = 108$, $S(M_6) = 9*5 = 45$, $S(M_7, M_8) = 5*4 = 20$ and $S(M_9, M_{10}) = 3*2 = 6$. Thus, it is necessary 179 area units [1] to implement the logic circuit of Moore FSM $U_2(\Gamma_1)$. It can be found for FSM $U_1(\Gamma_1)$ that $H = 19$, $S(M_1) = 2(4+3)*19 = 166$, $S(M_2) = 19*3 = 57$, $S(M_3) = 2*3*7 = 42$ and $S(M_4) = 7*4 = 28$. It means that logic circuit of FSM $U_1(\Gamma_1)$ occupies 293 area units. Besides, the circuit of $U_1(\Gamma_1)$ has 4 levels of logic, whereas the circuit for $U_2(\Gamma_1)$ only three (because functions τ and Y are generated in the same time). Thus, application of proposed method for encoding of collections of microoperations with state code presentation in the form (7) allows area decrease for 1,7 times Moore FSM.

V. CONCLUSION

The proposed method of state code presentation targets on area decrease under implementation of Moore FSM logic circuit with customized matrices. This approach allows decreasing the number of terms in the system of input memory functions up to corresponding value of the equivalent Mealy FSM. Besides, this method permits decreasing the number of terms in the system of microoperations due to the lack of dependence among the state codes and codes of collections of microoperations.

Investigation for effectiveness of proposed method was conducted on the standard examples [10]. It shows that the proposed method permits to decrease the average chip area occupied by FSM circuit up to 52% in comparison with the standard FSM implementation. In the same time, it was the increase for FSM performance in 86% of examples. The further direction of our research is application of proposed method for case of FPGA.

REFERENCES

[1] Baranov S., *Logic Synthesis for Control Automata*, Kluwer Academic Publishers, Boston, 1994.

[2] A. Barkalov, L. Titarenko, *Logic Synthesis for FSM-Based Control Units*. Springer - Berlin Verlag Heidelberg, Lectures Notes in Electrical Engineering, 2009, №53.

[3] DeMicheli *G. Synthesis and Optimization of Digital Circuits*, McGraw-Hill, NY, 1994.

[4] Maxfield C. *The Design Warrior's Guide to FPGAs*, Elseveir, Amsterdam, 2004.

[5] Smith M. *Application-Specific Integrated Circuits*, Addison-Wesley, Boston, 1997.

[6] Nababi Z. Embedded *Core Design with FPGA*, McGraw-Hill, NY, 2008.

[7] Yang S. *Logic Synthesis and Optimization Benchmarks user guide. Technical report*, №1991 – IWLS-UG-Saryang.-Microelectronics center of North Carolina.

Prof. dr hab. inż. Alexander BARKALOV, prof. UZ. Prof. Alexander A. Barkalov worked in Donetsk National Technical University (DNTU) from 1976 till 1996 as a tutor . He cooperated actively with Kiev Institute of Cybernetics (IC) named after Victor Glushkov. He got his degree of doctor of technical sciences (Informatics) in 1995 from IC. From 1996 till 2003 he worked as a professor of DNTU. From 2003 he has been working as a professor on Department of Electrotechnics, Informatics and Telecommunications of University of Zielona Gora. The area of his scientific activity is design methods for control units implemented with FPGA, CPLD, and ASIC.

Dr hab. inż. Larysa TITARENKO, prof. UZ. Prof. Larysa Titarenko has got her degree of doctor of technical sciences (Telecommunications) in 2005 from Kharkov National University of Radioelectronics (KNURE). Till September, 2003 she worked as a professor of KNURE. From 2005 she has been working as a professor on Department of Electrotechnics, Informatics and Telecommunications of University of Zielona Gora. The area of her scientific activity is design methods for control units implemented with FPGA, CPLD, and ASIC, as well as their application in telecommunications.

MA Olena Hebda. MA Olena Hebda studied at National aerospace university 'Kharkiv Aviation Institute' from 2003 till 2009 and has received higher education on speciality "Biotechnical and Medical Apparatuses and Systems" and qualification of the research engineer (electronics, telecommunications). In 2009 she has started PhD study on Department of Electrotechnics, Informatics and Telecommunications of University of Zielona Gora. The area of her scientific activity is design methods for control units implemented with FPGA, CPLD, and ASIC.

MA Kirill Soldatov. MA Kirill Soldatov studied at Donetsk National Technical University (DNTU) from 200 till 2010 and has received higher education on speciality "Computer systems and networks". In 2010 he has started PhD study on the chair of computer engineering of DNTU. The area of his scientific activity is design methods for control units implemented with FPGA, CPLD, and ASIC.

Modification of Elementary Operational Linear Chains in Compositional Control Unit with Code Sharing

Aleksander A. Barkalov, *Member, IEEE,* Larysa A. Titarenko, Aleksander N. Miroshkin

Abstract **– The new design method for compositional microprogram control units with code sharing and elementarization of operational linear chains is proposed. The method targets on reduction in the number of LUT-elements in the combinational part of control unit. Some additional control microinstructions containing codes of the classes of pseudoequivalent chains are used for operational linear chains modification. Proposed method is illustrated by an example. Most desirable GSA characteristics for using proposed method were obtained.**

Index Terms **– Circuit synthesis, flow graphs, logic devices, minimization methods.**

I. Introduction

Using of any elementary basis for realization of control unit circuits causes necessity of taking into account not only control algorithm peculiarities but basis features also. The task of hardware amount decrease became very urgent when one need to realize complex of operational and control units on one chip [1]. One of possible way of this task solving is hardware amount decrease in control unit due to using of pseudoequivalent states of Graph-Scheme of Algorithm (GSA). High percentage (>75%) of operational vertices in GSA [2] and presence of embedded memory blocks make it possible of using this synthesize method modification.

Compositional microprogram control unit (CMCU) is reasonable to use in case of linear GSA (percentage of operational vertices in GSA is over 75%) [2]. FPGA (Field-Programmable Gate Arrays) basis is commonly used for realization of control unit circuit [3, 4]. Modification of synthesis method for CMCU with code sharing and modification of operational linear chains (OLC) is proposed in this article.

The main purpose of investigation is simplification of combinational part of CMCU via implementation to GSA of additional vertices containing pseudoequivalent operational linear chain (POLC) class code. The main task of investigation is development of CMCU synthesis method modification that let decrease number of LUT-elements (Look-Up Tables, structural elements of FPGA basis) in Block of Microinstruction Addressing (BMA). Control algorithms are represented as GSA.

II. Main statements

Graph-scheme of control algorithm consists of operational and conditional vertices, making sets E_1 and E_2 accordingly, and the set of arcs E. Let us begin vertex be marked as b_0, end – b_E. Operational vertex $b_q \in E_1$ contain set of microinstructions $Y(b_q) \subseteq Y$, where $Y = \{y_1,...y_N\}$ is the set of output signals of control unit. Conditional vertex $b_g \in E_2$ contains one elements $X(b_g)$ of the logical conditions set $X = \{x_1,...x_L\}$. In case of operational vertices percentage is over 75% from total number of vertices, we talk about linear GSA.

OLC is a sequence of operational vertices of graph-scheme of algorithm. Each OLC α_g has accidental number of inputs I_g^i and only one output Q_g. Formal definitions of OLC, its input and output one can find in [5]. OLC with only one input and one output is called elementary [2].

OLC, outputs of which are connected with the input of the same vertex are called pseudoequivalent operational linear chains (POLC). Such OLCs make the class B_i. All classes are packed into the set $B = \{B_1,...B_I\}$ of POLC classes.

Let GSA contains G elementary OLC α_g that form the set C.

$$R_1 = \lceil \log_2 G \rceil \tag{1}$$

bits are enough for encoding elements of the set C. Number of components in OLC α_g is marked as F_g. Maximum length $Q = \max(F_1,...F_G)$ of linear chain

Manuscript received December 20, 2009. Modification of elementary operational linear chains in compositional control unit with code sharing.

A. A. Barkalov is with University of Zielona Gora, Poland. E-mail: A.Barkalov@iie.uz.zgora.pl

L. A. Titarenko is with University of Zielona Gora, Poland.
E-mail: L.Titarenko@iie.uz.zgora.pl

A. N. Miroshkin is with Donetsk National Technical University, Donetsk, Ukraine. MiroshkinAN@gmail.com.

determines number of bits R_2 in the code for encoding OLC components, where

$$R_2 = \lceil \log_2 Q \rceil. \qquad (2)$$

Elements $\tau_r \in \tau$ and $T_r \in T$ are used for encoding elementary OLC and their components accordingly. It being known that $|\tau| = R_1$ and $|T| = R_2$. Encoding of components is performed in natural order, that is

$$K(b_{gi}) = K(b_{gi-1}) + 1, \qquad (3)$$

where $g = 1, ..., G$, $i = 1, ..., F_g$.

Each operational vertex $b_q \in E_1$ corresponds to microinstruction MI_q storing in control memory (CM) in the cell with address $A(b_q) = A_q$. Code sharing is obtaining of the address A_q as concatenation of OLC code and its component code [2]. Total width of address is

$$R_A = R_1 + R_2. \qquad (4)$$

Structure of compositional microprogram control unit with elementary OLC and code sharing can be used for interpr etation of graph-scheme of control algorithm (Fig. 1). Let us call this structure U_1.

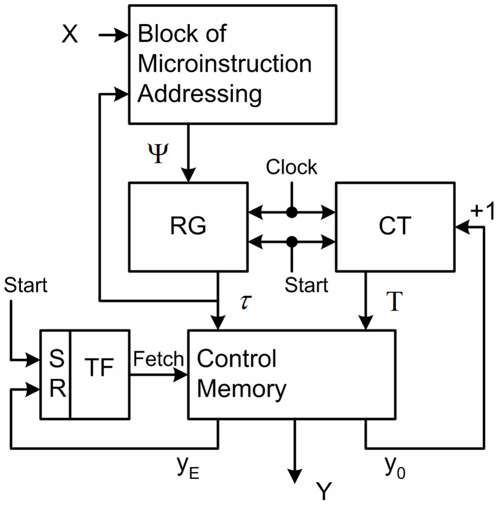


Fig. 1. Structure of compositional microprogram control unit with elementary OLC and code sharing

Block of microinstruction addressing in CMCU scheme realized function of memory excitation for register RG:

$$\Psi = \Psi(X, \tau). \qquad (5)$$

When signal Start is coming initial microprogram address is loaded into RG, zero value is loaded into CT, and flip-flop TF is set to “1” that allows reading microinstructions from control memory. There are two additional internal signals: y_0 and y_E. In case of $y_0 = 1$ content of CT is incremented and next vertex of current operational linear chain is addressed. If $y_0 = 0$ then OLC output is reached and BMA prepares address of next OLC using code of current POLC class. Signal y_E is used at the end of microprogram to reset flip-flop TF. The value “0” of TF output stops access to CM.

Asynchronous reset of counter must be controlled by function $\text{Start} \vee \overline{y_0}$. Signal $\overline{y_0}$ ensures loading zero value to the CT when transition to another OLC performed.

Number of terms in BMA scheme can be decreased by implementation OLC code transformer into POLC class codes [2]. But such realization demands extra FPGA recourses.

In the article complexity of code transformer is proposed to decrease by additional vertices with pseudoequivalent class codes. Free recourses of embedded memory are proposed for storing additional microinstructions.

III. Main idea of proposed method

In initial GSA the set C_1 contains OLC α_g, which are not connected to the end vertex of GSA. All operational linear chains are divided into classes $B_i \in \Pi_C$ of POLC. Binary code $K(B_i)$ of width R_3 is set to each class B_i, where

$$R_3 = \lceil \log_2 I \rceil. \qquad (6)$$

In (6) I is number of POLC classes. After output vertex of each OLC $\alpha_g \in C_1$ additional vertex is added. It contains pseudoequivalent class code $K(B_i)$ of current OLC. For modified OLC encoding $R_2^{'}$ five bits are enough, where

$$R_2^{'} = \lceil \log_2 Q' \rceil. \qquad (7)$$

In (6) Q' is maximum number of vertices in OLC after their modification, therefore $R_2^{'} \geq R_2$.

Embedded memory blocks (EMB) in FPGA can be configured for different task performing. So, in Stratix III chip exists next configurations: 16K×8, 8K×16, 4K×32, 2K×64, 16K×9, 8K×18, 4K×36, 2K×72 [4]. Total number of chip contacts is constant value, that’s why method modification usage restriction presents. If N_c is memory chip contacts number, then

$$n_1 = N_c - R_A^{'} \qquad (8)$$

free contacts of each block can be used for microinstruction generation, where

$$R_A^{'} = R_1 + R_2^{'}. \qquad (9)$$

For all output signals realization N_{EMB} memory blocks are used:

$$N_{EMB} = \left\lceil \frac{N+2}{n_1} \right\rceil. \qquad (10)$$

In (10) N is number of bits for output signals unitary encoding [2]. Constant «2» takes into account additional internal variables y_0 and y_E. Total number of unused memory outputs is obtained according (11):

$$n_2 = n_1 * \left\lceil \frac{N+2}{n_1} \right\rceil - (N+2)\,. \qquad (11)$$

Address width of OLC component can be increased as a result of additional vertices implementation. In this case only $(n_1 - 1)$ outputs of each block are used for output functions realization. If number of used EMB still the same when number of free memory outputs is decremented, usage of proposed method modification is reasonable, i.e.:

$$\left\lceil \frac{N+2}{n_1} \right\rceil * n_1 = \left\lceil \frac{N+2}{n_1 - 1} \right\rceil * (n_1 - 1). \qquad (12)$$

Thereby, condition of reasonability is one of next conditions fulfillment:

$$\left[\begin{array}{l} R'_2 = R_2; \\ \left\lceil \frac{N+2}{n_1} \right\rceil * n_1 = \left\lceil \frac{N+2}{n_1 - 1} \right\rceil * (n_1 - 1). \end{array}\right. \qquad (13)$$

First condition means the same code width for encoding both non-modified and modified OLC components. Second one ensures the same number of used EMB block when codes of non-modified and modified OLC components have different widths. If one or/and two conditions from (13) take place, modification of syntheses method can be done. CMCU structure U_2 is obtained (Fig. 2).

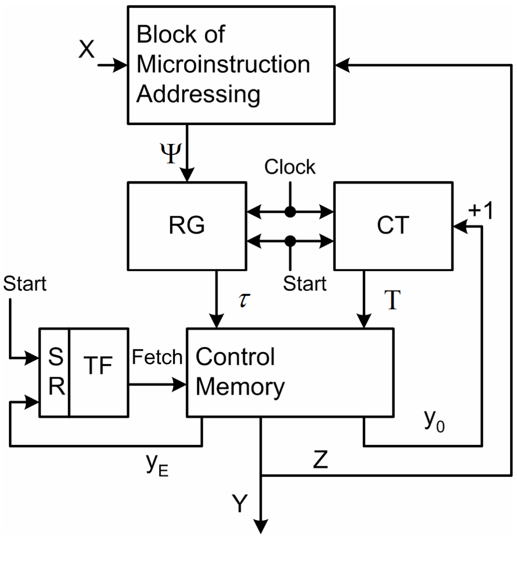


Fig. 2. Structure of compositional microprogram control unit with elementary OLC and code sharing after implementation addition microinstructions

In CMCU U_2 variables $z_r \in Z$, where $|Z| = R_3$, are bits of code $K(B_i)$. Block of microinstruction addressing performed function

$$\Psi = \Psi(Z, X)\,. \qquad (14)$$

Other blocks of CMCU U_2 perform corresponding functions to functions of CMCU U_1 blocks. Let us point out that structural elements BMA, CT, RG, TF is realized in LUT-elements, and CM is implemented in embedded memory.

The following method of CMCU U_2 synthesis is proposed in this article:

1. Construction of the sets C, C_1, and Π_C for a GSA Γ.
2. Implementation of additional vertices with pseudoequivalent class codes $K(B_i)$ to OLC α_g.
3. Encoding of OLC, their components and classes $B_i \in \Pi_C$.
4. Construction of the content of control memory.
5. Construction of CMCU transition table and $\Psi = \Psi(Z, X)$ functions.
6. Synthesis of CMCU logic circuit.

IV. Example of method using

Let GSA Γ_1 (Fig. 3) be characterized by next sets: $C = \{\alpha_1, ..., \alpha_6\}$ – elementary OLC, $C_1 = C \setminus \{\alpha_5, \alpha_6\}$ OLC without connection to the end vertex, $\Pi_C = \{B_1, B_2\}$ – classes of pseudoequivalent operational linear elementary chains, where $B_1 = \{\alpha_1\}$, $B_2 = \{\alpha_2, \alpha_3, \alpha_4\}$.

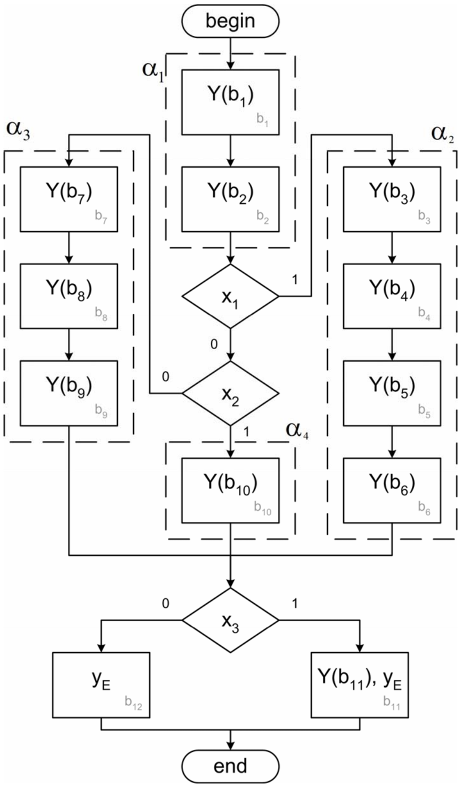


Fig. 3. Initial GSA Γ_1 before implementation of additional vertices with pseudoequivalent class codes

Number of OLC $G=6$, $R_1=3$ bits from the set $\tau=\{\tau_1,\tau_2,\tau_3\}$ are used for their encoding. Maximum length of OLC is $Q=4$, let us use $R_2=2$ variables from the set $T=\{T_1,T_2\}$ for OLC components encoding. Total number of operational vertices is $M=12$, this number demands $R=4$ bit of address in CM. For encoding $I=2$ classes $B_i \in \Pi_C$ of POLC $R_3=1$ bit is used.

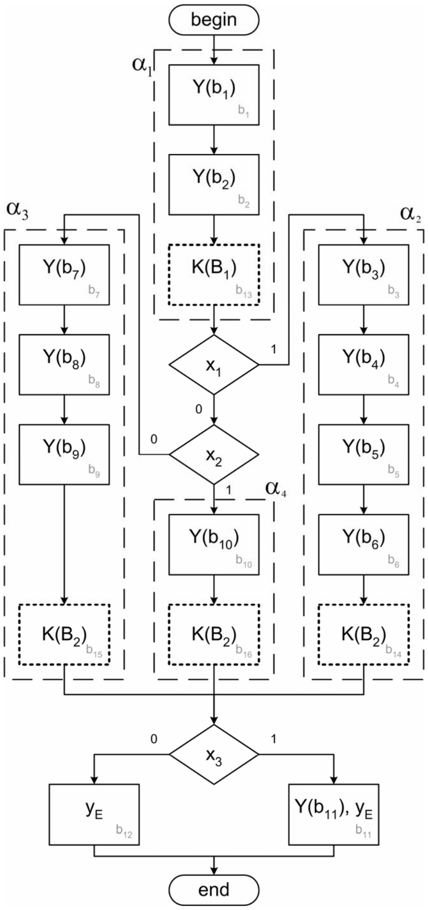


Fig. 4. Initial GSA Γ_1 after implementation of additional vertices with pseudoequivalent class codes

Let us encode OLC $\alpha_g \in C$ and their components in arbitrary manner (3). Addresses $A(b_q)$ of CMCU $U_2(\Gamma_1)$ microinstructions are shown in Table I. Here and after symbol $U_i(\Gamma_j)$ means, that CMCU U_i interprets GSA Γ_j. From Table I one can obtain addresses as concatenation of OLC code and code of component, for example: $A(b_6)=001011$, $A(b_{10})=011000$ and so on. Extra bit is added to the OLC component code according to (6) and (9) because of maximum length of OLC after modification is $Q'=5$.

TABLE I
ADDRESSES OF CMCU MICROINSTRUCTIONS

$(T_0)T_1T_2$ \ $\tau_1\tau_2\tau_3$	000	001	010	011	100	101
(0)00	b_1	b_3	b_7	b_{10}	b_{11}	b_{12}
(0)01	b_2	b_4	b_8	b_{16}	–	–
(0)10	b_{13}	b_5	b_9	–	–	–
(0)11	–	b_6	b_{15}	–	–	–
(1)00	–	b_{14}	–	–	–	–

Codes of classes $B_i \in \Pi_C$ are set as $K(B_1)=0$, $K(B_2)=1$. Microinstruction format of CMCU U_2 in case of $y_0=1$ includes fields y_0, y_E, FY, where field FY contains code of microoperation set; and in other case y_0, y_E, FB – code of class $B_i \in \Pi_C$.

Let EMB block contacts be $N_C=25$ [4], then according to (8), each block has $n_1=25-5=20$ free contacts. Let us suppose number of output functions be $N=30$. For their realization $N_{EMB}=2$ blocks of memory are used (10). According to (11) last block has $n_2=8$ free contacts after realization of all output functions. Second condition of (13) takes place, modification of syntheses method is reasonable. Contents of CMCU $U_2(\Gamma_1)$ control memory is shown in Table II.

TABLE II
CONTENTS OF CMCU CONTROL MEMORY

$A(b_q)$	y_0	FY		y_E
		FB	*	0
$A(b_1)$	1	$Y(b_1)$		0
$A(b_2)$	1	$Y(b_2)$		0
$A(b_{13})$	0	$K(B_1)$	*	0
$A(b_3)$	1	$Y(b_3)$		0
$A(b_4)$	1	$Y(b_4)$		0
$A(b_5)$	1	$Y(b_5)$		0
$A(b_6)$	1	$Y(b_6)$		0
$A(b_{14})$	0	$K(B_2)$	*	0
$A(b_7)$	1	$Y(b_7)$		0
$A(b_8)$	1	$Y(b_8)$		0
$A(b_9)$	1	$Y(b_9)$		0
$A(b_{15})$	0	$K(B_2)$	*	0
$A(b_{10})$	1	$Y(b_{10})$		0
$A(b_{16})$	0	$K(B_2)$	*	0
$A(b_{11})$	*	$Y(b_{11})$		1
$A(b_{12})$	*	-		1

As one can see from Table 2, if $b_q \in E_1$ is output of OLC $\alpha_g \in C_1$, microinstruction has value “0” in y_0 field and code $K(B_i)$ instead of output function set $Y(b_q)$ in the microinstruction word.

Transitions from outputs of OLC $\alpha_g \in C_1$ are expressed by next system of formulae [2]:

$$\begin{aligned} &B_1 \rightarrow x_1 b_3 \vee \overline{x_1} x_2 b_{10} \vee \overline{x_1}\,\overline{x_2} b_7; \\ &B_2 \rightarrow x_3 b_{11} \vee \overline{x_3} b_{12}. \end{aligned} \tag{15}$$

Such system is the base for CMCU U_2 transition table formation. This table consists of next columns: B_i, $K(B_i)$, b_q, $A(b_q)$, X_h, Ψ_h, h. Their purpose became clear from Table III.

TABLE III
FRAGMENT OF CMCU TRANSITION TABLE

B_i	$K(B_i)$	b_q	$A(b_q)$			X_h	Ψ_h	h
	z_1		τ_1	τ_2	τ_3			
B_1	0	b_3	0	0	1	x_1	D_3	1
		b_7	0	1	0	$\overline{x_1}\,\overline{x_2}$	D_2	2
		b_{10}	0	1	1	$\overline{x_1} x_2$	D_2, D_3	3

Addresses of microinstruction is taken from Table 1. Let us point out, that system of memory excitation functions Ψ includes functions $\{D_1, D_2, D_3\}$. Total number of rows $H_2(\Gamma_j)$ in transition table of CMCU $U_2(\Gamma_j)$ is equal to number of terms in system transition formulae. In our example, $H_2(\Gamma_1) = 5$.

System (15) is formed according to transition table. Fragments of system Ψ can be found from Table III:

$$\begin{aligned} &D_2 = \overline{z_1}\,\overline{x_1}; \\ &D_3 = \overline{z_1} x_1 \vee \overline{z_1}\,\overline{x_1} x_2. \end{aligned} \tag{16}$$

For minimization of terms number in (15) classes $B_i \in \Pi_C$ may be encoded with the help of ESPRESSO algorithm, for example.

Realization of logical circuit of CMCU U_2 reduces to implementation of system (16) in base of integrated circuit (FPGA) and realization of control memory on blocks of embedded memory. Modern CAD systems or methods [1, 2] can be used for this purpose.

V. CONCLUSION

Proposed method of OLC modification for compositional microprogram control unit is oriented to LUT-elements decrease in the block of microinstruction addressing. Number of memory blocks in device is the same as for base structure CMCU U_1 with code sharing. Extra clock cycles are used for analysis GSA additional vertices. But at the same time complexity of control unit circuit decreases, that leads to clock signal duration reduction. Conclusion about time characteristics of control unit can be done only for individual case.

Disadvantage of proposed method is in its usage limitation (13).

Scientific novelty of proposed method modification is in usage of POLC classes and free recourses of control memory for LUT-elements number decrease in block of microinstruction addressing. Practical meaning is in chip parameters decrease. It allows realization of device with less cost.

Our future work is directed at development of CAD system for synthesis of control units [5].

REFERENCES

[1] Соловьев В.В. Проектирование цифровых схем на основе программируемых логических интегральных схем. М.: Горячая линия-ТЕЛЕКОМ, 2001. 636 с.

[2] Баркалов А.А. Синтез устройств управления на программируемых логических устройствах. Донецк: ДНТУ, 2002. 262 с.

[3] Virtex-6 FPGAs. Lowest Power High-Performance FPGAs // Available: http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf

[4] Stratix III FPGA: Lowest Power, Highest Performance 65-nm FPGA // Available: http://www.altera.com/products/devices/stratix-fpgas/stratix-iii/st3-index.jsp

[5] Баркалов А.А., Титаренко Л.А. Синтез микропрограммных автоматов на заказных и программируемых СБИС. Донецк: УНИТЕХ, 2009. 336 с.

Aleksander A. Barkalov, Doctor of Science, Professor of DonNTU (Ukraine), Professor of University of Zielona Gora, Poland.
Scientific interests: digital control units, SoPC
Address: Campus A, Budynek Dydaktyczny / A-2 prof. Z. Szafrana str. 2, 65-516 Zielona Gora
E-mail: A.Barkalov@iie.uz.zgora.pl

Larysa A. Titarenko, Doctor of Science, Professor of Kharkiv National Univercity of Radioelectonics (KNURE), Professor of University of Zielona Gora, Poland.
Scientific interests: Digital, adaptive and spatial-time processing of signals in telecommunication. Management and control in communication networks Research of modern digital telecommunication systems and nets.
E-mail: L.Titarenko@iie.uz.zgora.pl

Aleksander N. Miroshkin, Assistant of Donetsk National Technical University.
Scientific interests: digital control units.
E-mail: MiroshkinAN@gmail.com

Comparison of Survivability & Fault Tolerance of Different MIP Standards

Ayesha Zaman , M.L. Palash , Tanvir Atahary and Shahida Rafique

Abstract— **Mobile IP, the current method of internet connectivity is most often found in WLAN environments where users need to carry their mobile devices across multiple LANs with different IP address. This project work first surveys existing protocols for supporting IP mobility and then proposes an extension to mobile IP architecture, called Robust Hierarchical Mobile IP version6 (RH-MIPv6). This architecture attempts to achieve smaller handoff latency in intra-domain mobile network. In RH-MIPv6 a mobile node (MN) registers primary (P-RCoA) and secondary (S-RCoA) regional care of address to two different MAPs simultaneously. In adaptation to this a "MOVING AREA BASED MAP SELECTION SCHEME" is proposed in this paper. A mechanism is developed to enable the mobile node or correspondent node to detect the failure of primary MAP and change their attachment from primary to secondary MAP. With this recovery procedure, it is possible to reduce the failure recovery time. In this paper it is shown that RH-MIPv6 has faster recovery time than MIPv4 and HMIPv6**

Index Terms— **Intra-domain Mobility, Handoff Latency, Triangular Routing, Transmission Control Protocol Sequence.**

I. Introduction

Mobile IP is an Internet Engineering Task Force (IETF) standard communications protocol that is designed to allow mobile device users to move from one network to another while maintaining their permanent IP address. A mobile node registers its location at the home agent. The time taken for this registration process combined with the time taken for a mobile node to configure a new network care-of address in the visiting network, amounts to the overall handoff latency.

The less the handoff latency, the shorter the packet failure recovery time within the MIP standard. Again "Survivability" is used to describe the available performance after a failure [1]. MIPv4 does not consider system survivability and fault tolerance due to triangular routing problem. Again in the HMIPv6, HAs and MAPs are two points of failure and potential performance bottlenecks. Since failure of home agent (HA) or mobile anchor point (MAP) causes service interruption, the Hierarchical Mobile IPv6 (HMIPv6) has only weak survivability [2].

In this paper Robust Hierarchical Mobile IPv6 (RH-MIPv6) is proposed which provides survivability and fault tolerance with the existing HMIPv6.In the HMIPv6, when some failures happen in the mobility agents, an MN re-configures a new RCoA after the detection of the failures. Therefore, in this mechanism a significant amount of time is wasted for the failure detection and the duplicate address detection (DAD). On the other hand, in the RH-MIPv6, multiple RCoAs are configured in advance and are dynamically changed after the failure detection. Thus it is possible to reduce the failure recovery time compared with the HMIPv6.

II. Architecture of Robust Hierarchical Mobile IPv6 (RH-MIPv6)

Here Figure(1) illustrates the architecture of RH-MIPv6.

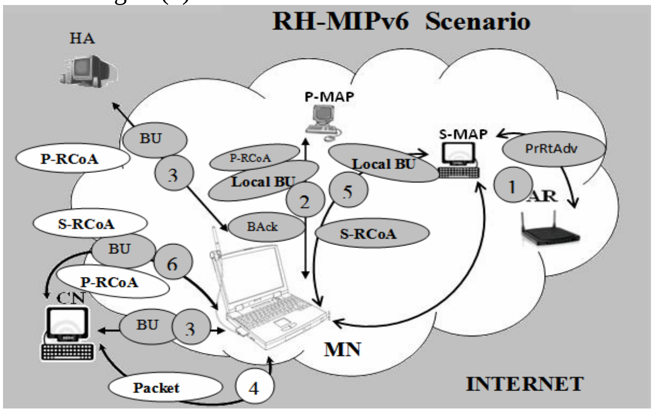


Figure 1. RH-MIPv6 Architecture

Manuscript received Sunday, June 14, 2009. This work was sponsored by the IEEE Computer Society Test Technology Technical Council (TTTC) and financially supported by Cadence, JTAG Technologies, Kaspersky Lab, Synopsis, Mentor Graphics, Tallinn Technical University, Donetsk Institute of Road Transport, Moscow Institute of Electronics and Mathematics, Virage Logic, Echostar, Aldec, Teprocomp, Data Art Lab.

F. A. Ayesha Zaman is a researcher in Dept. of Applied Physics, Electronics and Communication Engineering, University of Dhaka. ph: 01711589484; email: ayeshazaman_gsm@yahoo.com

S. B. M.L. Palash is a lecturer in Department of Electronics and Telecommunication Engineering, Institute of Science & Technology.

T.C. Tanvir Atahary is a lecturer in Department of Electronics and Telecommunication Engineering, University of Liberal Arts Bangladesh, Dhaka Bangladesh. email: tanoy_ece@yahoo.com

Fo. C. Prof. Dr. Shahida Rafique is a Professor in Dept. of Applied Physics, Electronics and Communication Engineering, University of Dhaka. She is also the Dean of Engineering faculty of the same university, email: Shahida.rafique@hotmail.com

III. Primary and Secondary MAP Selection Scheme

A MAP periodically broadcasts a RA message with a MAP option, in order to inform MNs of its presence. The MAP option, which is an IPv6 neighbor discovery extension, is used to disseminate the MAP information throughout a foreign network. For the purpose of MAP selection, the MAP option contains two fields: Distance and Preference fields. The Distance field records the hop distance from the MAP to the MN whereas the Preference field indicates the willingness of the MAP to offer a local registration service. In the furthest MAP selection algorithm, the furthest MAP may be overloaded if all MNs register to the furthest MAP. In addition, if an MN's movement is bounded to the limited area, a nearer MAP can reduce registration delay and signaling overhead than the furthest MAP. For the preference selection, it is difficult to determine how to assign preference values for each MAP. For this a new MAP selection scheme has been proposed which is the "MOVING AREA-BASED MAP SELECTION".

IV. Moving Area-Based MAP Selection

The proposed scheme is called Moving Area-Based MAP selection, in which the MH keeps track of its moving area to determine the best MAP. Figure (2) illustrates the basic idea of the proposed scheme, in which each mobile host selects the closest MAP that covers its moving area. The mechanism for a MH to keep track of its moving area is based on the MAP option periodically transmitted downward by each MAP in the hierarchy. The MH records the total number of MAP options issued by each MAP in its MAP Option Table. When the MH moves to a new subnet, it invokes the MAP selection algorithm to select the MAP with most MAP options received by the MH over a predefined period of time. If there are two or more MAP candidates with the same most MAP options, the MH selects the lowest MAP.

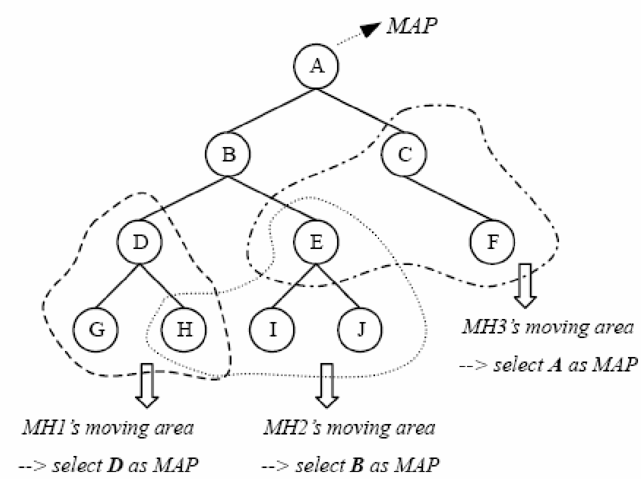
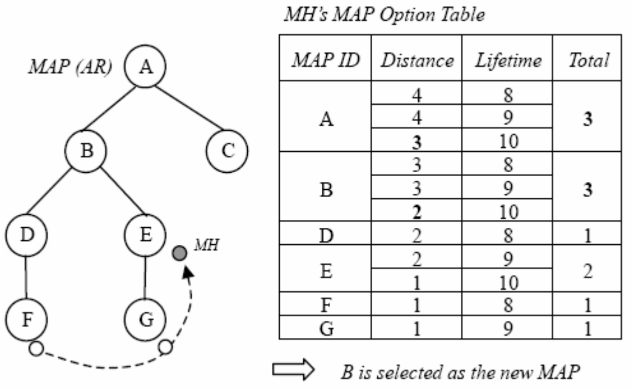


Figure 2. Moving area based MAP selection

For example, the MH in Figure (3) moves from MAP F (the initial position) to MAP G, and then to MAP E. It is assumed that the MH receives only one RA (sent out by MAP A) while it is on each MAP and the lifetime for a new MAP option is set to 10. The MAP Option Table maintained by the MH is shown in figure (3). According to the proposed scheme, MAP B is selected as the new MAP by the MH since it is the closest MAP from which the MH has received the most MAP options. So an MN, receiving multiple RA messages from multiple MAPs, can select two MAPs: the most suitable MAP and the next one. The most suitable MAP is called a primary MAP (P-MAP) and the next one a secondary MAP (S-MAP). In addition, the MN configures a primary RCoA (P-RCoA) and a secondary RCoA (SRCoA) in the P-MAP and S-MAP domains, respectively. After that, the MN registers its (Local Care of Address)L-CoA to the P-MAP/S-MAP.



MH's MAP Option Table

MAP ID	*Distance*	*Lifetime*	*Total*
A	4	8	**3**
	4	9	
	3	10	
B	3	8	**3**
	3	9	
	2	10	
D	2	8	1
E	2	9	2
	1	10	
F	1	8	1
G	1	9	1

⇨ *B is selected as the new MAP*

Figure 3. An example of MAP option table

V. Failure detection and recovery mechanisms

In the HMIPv6 specification, a MAP failure event can be detected by checking a MAP option, which contains an invalid lifetime in the broadcasted RA message. However, it takes too much time for an MN to detect the failure by this passive method because the RA interval is set to a few seconds. Thus, the passive failure recovery mechanism of HMIPv6 results in high packet losses, especially when the MN is communicating with multiple CNs. On the other hand, an MN in the RH-MIPv6 specification detects a MAP failure during packet transmission by utilizing ICMP. Therefore, faster failure detection can be achieved in the presence of active sessions without waiting for any RA message with a coarse grained RA interval. In this section, the failure detection and recovery mechanisms are divided into two cases: those detected by the MN and those by the CN.

A. Failure Detection and Recovery by MN

RH-MIPv6 provides a more active failure detection method than HMIPv6. If an MN is actively sending packets to CN, the MN can detect the MAP failure by receiving ICMP error messages from a router adjacent to the failed P-MAP. Or, when the MN is receiving packets from CN, the MN will receive the encapsulated packets from the S-MAP instead of the P-MAP. This indicates that a P-MAP failure happens. After detection of the P-MAP failure, the MN changes its serving MAP from the P-MAP to the S-MAP. Then, the MN can resume data transmission, if the MN was sending packets to a CN. In this case, the CN receives packets from the MN, which has registered a binding entry of a reset P field during secondary BU(Binding Update) procedure. Then, the CN eliminates primary binding information (i.e. the P field is set) and sets the P field of the secondary binding entry to 1. At the same time, the MN sends BU messages with S-RCoA to the HA and S-MAP as soon as possible. When the BU message arrives at the HA,the HA updates CoA of the MN from the P-RCoA to the S-RCoA. In addition, the S-MAP moves the binding entry of the MN from the backup mapping table to the serving mapping table.

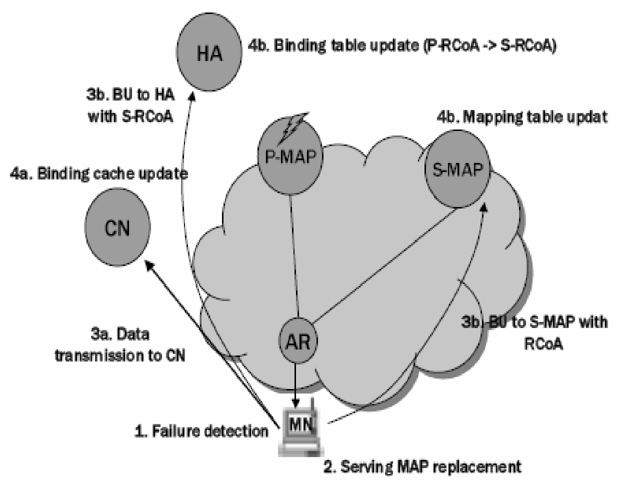

HA
4b. Binding table update (P-RCoA -> S-RCoA)
3b. BU to HA with S-RCoA
4b. Mapping table updat
P-MAP
S-MAP
4a. Binding cache update
CN
3b. BU to S-MAP with RCoA
AR
3a. Data transmission to CN
1. Failure detection
MN
2. Serving MAP replacement

Figure 4.Failure recovery procedure by the MN

B. *Failure Detection and Recovery by CN*

It is assumed that a CN is currently sending data packets to an MN via P-MAP. If the P-MAP fails, the CN receives ICMP error messages (i.e., Host Unreachable) for the sent packets. Then, the CN decides that the P-MAP has failed and rerouting through the S-MAP is then required. Typically, the link loss rate in a wired link is extremely low. Therefore, if the CN determines the MAP failure after receiving a few successive ICMP error messages, a wrong decision can be minimized [7].

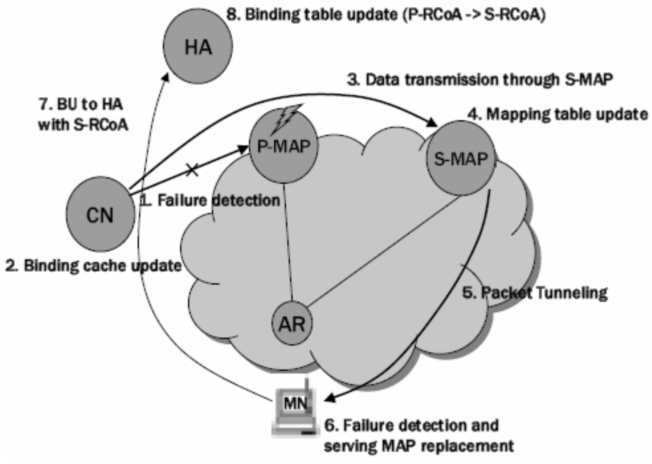


Figure 5. Failure recovery procedure by the CN

Fig.(5) illustrates the failure recovery procedure when a MAP failure is detected by a CN. As mentioned before, the CN regards multiple receptions of ICMP error messages as the indication of a MAP failure. Then, the CN looks for a binding entry with a reset P field in its binding cache, which is updated by the secondary BU procedure.After updating the binding cache, the CN resumes data transmission through the S-MAP and the S-MAP updates its mapping tables. After completion of the mapping table update, the S-MAP tunnels the received packets to the destination MN. If the MN detects a MAP failure, the MN sends a BU message with the S-RCoA, which is configured in the MAP selection procedure in advance, to the HA. After re-BU procedure, the MN can communicate with new CNs, which tries to connect the MN using binding information at the HA, through the S-MAP.

VI. Simulation Based Performance

In this paper IP packet recovery time of three Mobile IP standards has been compared. Here NCTUns-4.0 has been used as the simulator.

A. *TCP sequence number in case if MIPv4*

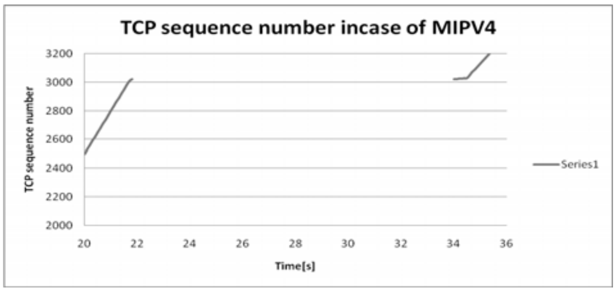


Figure 6. Time vs TCP sequence number for MIPv4

Analysis: When the MN travels from one subnet to another, in between the traversal process handover is halt for sometimes. Then packet transmission from the CN to the MN also halts for a few seconds. Again when the MN registers with a new point of attachment with the n FA, packet transmission continues. The registration process takes some non-zero time to complete as the registration requests propagate through the network. During this period of time the MN is unable to send or receive (IPv4) packets. Here the handover latency is larger due to the requirement of several registrations with different FAs. This is why the packet recovery time is of considerable amount which is shown here about 12 seconds.

B. *TCP sequence number in case of HMIPv6*

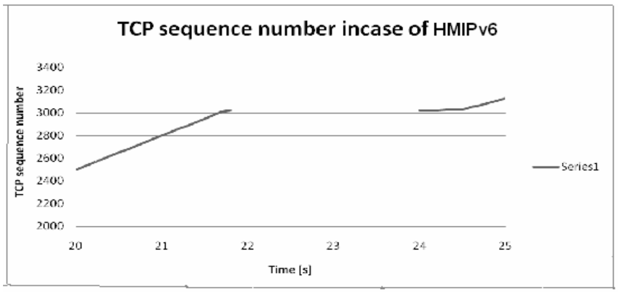


Figure 7. Time vs TCP sequence number for MIPv4

Analysis: In terms of failure recovery, in the case of HMIPv6 , when a failure happens at the MAP, an MN re-configures a new R-CoA after failure detection and the MN registers the new R-CoA to the HA and CNs . Only after the completion of these processes, the MN can resume the suspended sessions. Therefore a significant amount of time which may not be acceptable in real-time applications is wasted for failure detection and recovery in HMIPv6 networks. In this case the failure recovery time is about 2 seconds which is less than that of MIPv4 recovery mechanism.

C. *TCP sequence number in case of RH-MIPv6*

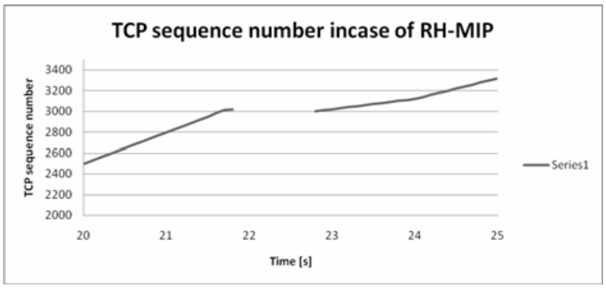


Figure 8. Time vs TCP sequence number for RH-MIPv6

Analysis: In RH-MIPv6, multiple R-CoAs are configured in advance and are dynamically changed after failure detection. Hence it is possible to reduce the failure recovery time than HMIPv6, which is about 1 second as shown in figure above.

VII. Conclusion

One of the most important criteria that affects the scalability property of a mobility management scheme is its signaling load i.e. the bandwidth used by the control messages, such as the Binding Updates to support mobility. In this project MOVING AREA BASED MAP SELECTION SCHEME is proposed where the most suitable MAP had the maximum MAP OPTIONs with minimum number of hop count. This scheme is proposed because in the conventional furthest MAP selection scheme there is a problem of signaling overload. Our future work will be to propose an appropriate load balancing mechanism for the MAPs within a domain in RH-MIPv6 standard.

Acknowledgment

The authors thank Prof. S.Y. Wang of Harvard University for the availability of software package NCTUns 4.0 Network Simulator and Emulator which has high-fidelity of simulating various protocols used in both wired and wireless IP networks.

References

[1] D. Tipper et al, “Providing Fault Tolerance in Wireless Access Networks” *IEEE Communications Magazine*, January 2002.

[2] H. Omar et al., “Supporting for Fault Tolerance in Local Registration Mobile IP Systems,” MIlcom,1998

[3] C. Caetelluccia and L. Bellier, “ Hierarchical Mobile IPv6 Mobility Management (HMIPv6)”, Internet Draft, draft-ietf-mobile ip-hmipv6-08.txt, Work in Progess , Jun, 2003.

[4] K. Kawano, K. Kinoshita, K. Murakami, “ A Mobility –Based Terminal Management in IPv6 Networks”, *ICICE Trans. Commun*. Vol. E85-B, No. 10, Oct, 2002.

[5] S.Pack, T. Kwon, Y. Choi, “ A Comparative Study of Mobility Anchor Point Selection Schemes in Hierarchical Mobile IPv6 Networks”, *ACM MobiWac 2004*, October 2004.

[6] Sangheon Pack, “Robust Hierarchical Mobile IPv6: An Enhancement for Fault-Tolerant Mobile Networks”.

[7] A. Conta, S. Deering, “ Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification”, *IETF RFC 2463*.

First A. Ayesha Zaman was born in Dhaka, Bangladesh on 20th October, 1984. She completed her BSc. Engineering in 2007 and MSc. in 2009 from the department of Applied Physics, Electronics & Communication Engineering, University of Dhaka. She stood 1st class 2nd in BSc. and 1st class 1st in MSc. examination from the same department.

At present she is a research worker at University of Dhaka. She has also worked with neuro-image processing. She has keen interest in Biomedical Engineering also. She has a conference paper in this field; World Academy of Science, Engineering & Technology 57,2009: ISSN 2070-3724, p-p: 230-235; Amsterdam, The Netherlands. She intends to do her further research in the field of communication engineering especially high speed networking.

Investigation of Time Series of Original Values of Currency Rates Measured on Small Time Frames on FOREX Using Methods of Chaos Theory

Olga Dabi-Prashad (MPhil), Ludmila Kirichenko (PhD)
Department of Applied Mathematics
Kharkiv National University of Radio Electronics
Ukraine

***Abstract*—Lately, the linear paradigm with its idea of normal distribution of profits has been replaced with the non-linear approach and Chaos Theory which gives the explanation of the complex behavior of financial markets. It has been discovered that time series of profits measured on long time frames on currency and stock markets (time series of monthly prices etc.) are chaotic. This paper is concentrated on investigation of time series of original values of currency rates measured on short time frames on FOREX (hourly, 4-hourly, daily prices) using methods of Chaos Theory (Time-delay reconstruction method, Grassberger-Procaccia method, estimation of the Lyapunov exponent) in order to define if such time series are chaotic as well.**

***Index Terms*—FOREX, Chaos Theory, memory in financial time series, predictability of financial time series, small (short) time frames.**

I. Introduction

FOREX is the market where currency is sold or bought freely for another currency according to a currency rate. FOREX is also the most liquid market. Many companies and private persons conduct conversion operations on the currency market with various purposes. It is known, that the currency rates on the FOREX market are affected by many factors which makes the currency price movement very complicated [1][2]. It is obvious that to conduct the conversion operations with success it is very important to have a model which would provide a deep understanding of the complicated behavior of the FOREX market. Lately, the linear approach to modeling of the financial markets and the idea of normal distribution of profits (e.g. the portfolio theory, CAPM) [3-5] had been replaced with the non-linear approach and Chaos Theory [6-8]. Within the frame of the non-linear approach, the time series on stock and currency markets have been investigated using the methods of Chaos Theory and it has been shown that the investigated time series adhere to a law of deterministic chaos and are not stochastic as it was stated by the linear paradigm [6,7]. *Chaos* has been defined as a behavior of a deterministic dynamical system which has sensitive dependence on the initial conditions [9,10]. Since the chaotic data are produced with a deterministic system there are non-linear correlations between stages of the system. Thus, if it is discovered that some time series is chaotic, it means there is a memory of the time series about its values in the past and we can predict its values in the future. But still, in the investigations of time series on the FOREX market with the methods of Chaos Theory the time series were used which were constructed on long time frames, such as one day, one month and more. Also, original values of currency rates were transformed to profits which were used in all the calculations instead of original values [6].

Most of conversion operations on FOREX are of speculative character and conducted with the aim to gain profit on the currency rates fluctuations. Such operations are normally conducted using original values of currency rates and short time frames, such as one hour (H1), four hours (H4), one day (D1), one week (1WEEK). Thus, it is useful to investigate time series of original values of currency rates on the FOREX market, which are measured on such small time frames, using methods of Chaos Theory and define if they are chaotic as well.

The main methods which allow to test whether time series is chaotic are [11,12]: reconstruction of attractors with the delay-time reconstruction method, estimation of the correlation dimension in m-dimension space, estimation

Manuscript received December 19, 2009.

Olga Dabi-Prashad (MPhil), has graduated from Wessex Institute of Technology (University of Wales) (e-mail: radharani_ua@yahoo.com).

Dr. Ludmila Kirichenko, Associate professor of Department of Applied Math, Kharkiv National University of Radio Electronics (e-mail: ludmila@kture.kharkov.ua).

of the Lyapunov exponent. In the following sections of the paper these methods will be used to test the time series of original values of currency rates constructed on short time frames on FOREX. Section 2 describes the reconstruction of attractors of dynamical system with the delay-time reconstruction method using one single component. This method allows to estimate visually if the reconstructed attractor is an attractor of a chaotic system. Section 3 describes the test of the considered time series with the Grassberger-Procaccia method. It provides the quantitative estimation of the fractal dimension which saturates at some finite value for a deterministic process. In section 4 it is defined either the considered time series have a sensitive dependence on the initial conditions. The results of the investigation are provided and the paper is concluded in section 5. All estimations in the paper have been conducted using Mathematica 5.0.

II. Reconstruction of attractors with the delay-time reconstruction method

Strange attractor [13] is a set of all the trajectories of a chaotic system, and whatever the initial conditions a trajectory of the chaotic system runs from, it falls on one of the trajectories of the strange attractor. Thus, if a dynamical system adheres to a law of a deterministic chaos, its trajectories run within a strict space, whereas trajectories of a stochastic system look like a cloud of points which tends to fill the entire phase space.

The *delay-time reconstruction method* is based on the idea of the reconstruction of an attractor of a dynamical system using a one-dimensional time series, which are generated by this dynamical system [14]. According to this method, the reconstructed attractor and the original one are topologically equivalent. Vectors of a reconstructed attractor can be formed using the formula:

$$y(t) = (x(t), x(t-\tau), x(t-2\tau), ..., x(t-(m-1)\tau)) \tag{1}$$

where $x = x(t)$ is one-dimensional time series; τ is time delay; m is the dimension of a reconstructed phase space (lag space).

Let us first reconstruct the attractor for a Lorenz system which is described by equations [15]:

$$\begin{cases} \dfrac{dx}{dt} = \partial(y-x), \\ \dfrac{dy}{dt} = x(\rho - z) - y, \\ \dfrac{dz}{dt} = xy - \beta z. \end{cases} \tag{2}$$

where parameters $\partial, \rho, \beta > 0$ and are constant. Let the parameters be defined as follows: $\sigma = 10,\ \rho = 28,\ \beta = 8/3$.

Let us use the measurement of a single variable *y* to reconstruct the Lorenz attractor. Then the reconstructed Lorenz attractor looks as follows:

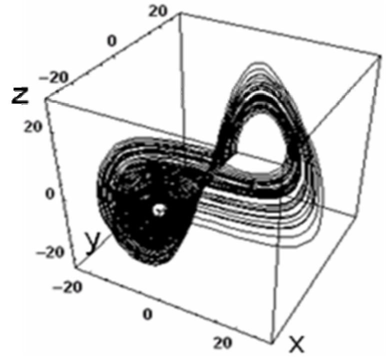


Fig. 1. Reconstructed Lorenz attractor in 3-dimensional lag space and τ =10

It is known, that the original Lorenz attractor looks as follows:

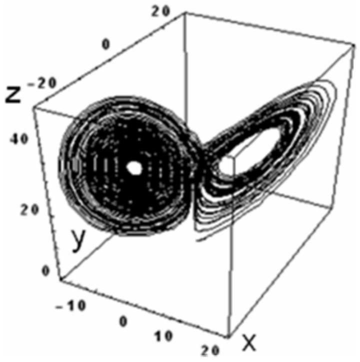


Fig. 2. Original Lorenz attractor.

As we can see from Fig.1 and Fig.2, the reconstructed attractor of the chaotic Lorenz system and the original one look very similar. For chaotic system they are topologically equivalent. Here the value of time delay τ has been chosen experimentally in order to achieve the best result (the reconstructed attractor should not look too stretched or too spread in the lag space).

Let us now reconstruct an attractor for a random system which is described by the autoregression equation $X(t) = \phi \cdot X(t-1) + \varepsilon(t)$, where ϕ is a constant and $\varepsilon(t)$ is a random component which follows Gaussian distribution.

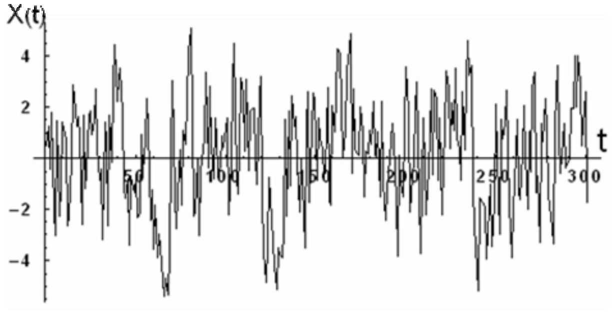


Fig. 3. Realisation of autoregression process for ϕ =0.5

The reconstructed attractor of a random system looks like a cloud of point which tends to fill the entire lag space. This happens for any value of time delay. We can see that in Fig. 4.

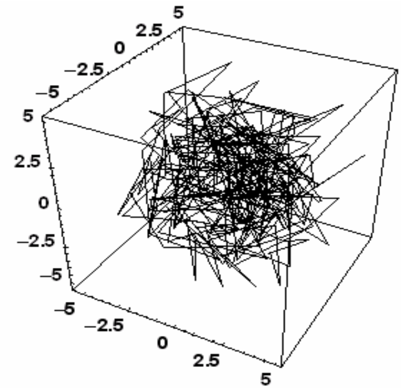


Fig. 4. Reconstructed attractor for autoregression process in a 3-dimensional space, τ =10

Let us now reconstruct attractors for time series of original values of currency rates on the FOREX market, which are measured using small time frames, which are the most common used for conducting of speculative trade operations, i.e. H1, H4, D1, 1WEEK.

Reconstruction of the strange attractor for EUR/USD, time frame H1

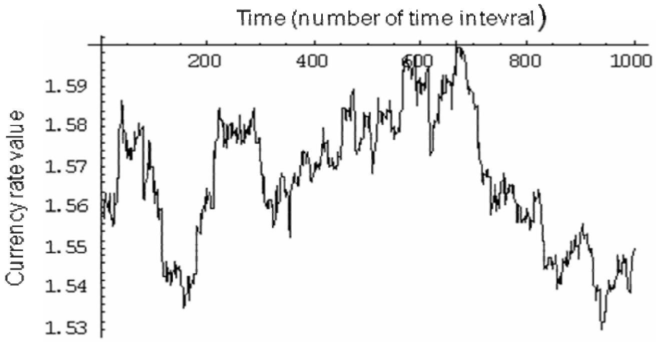


Fig. 5. Realisation of currency rate time series of currency pair EUR/USD, time frame H1

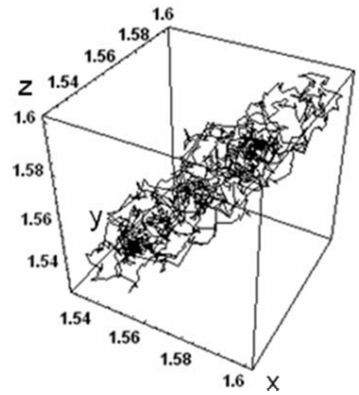


Fig. 6. Reconstructed attractor for EUR/USD in a 3-dimensional lag space, H1, optimal $\tau = 10$

Reconstruction of the strange attractor for NZD/CAD, time frame H4

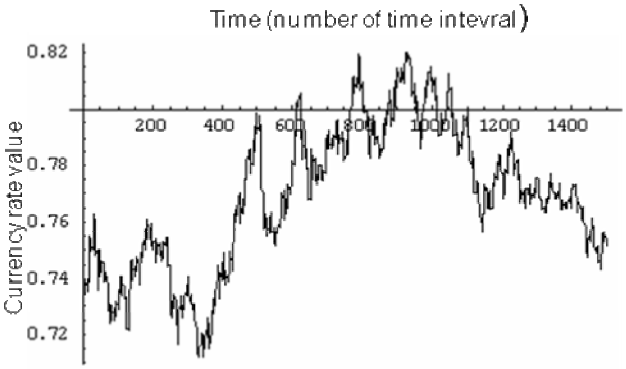


Fig. 7. Realisation of currency rate time series of currency pair NZD/CAD, time frame H4

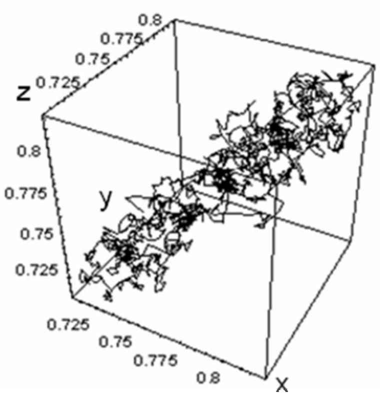


Fig. 8. Reconstructed attractor for NZD/CAD in a 3-dimensional lag space, H4, optimal $\tau = 15$

Reconstruction of the strange attractor for GBP/JPY, time frame D1

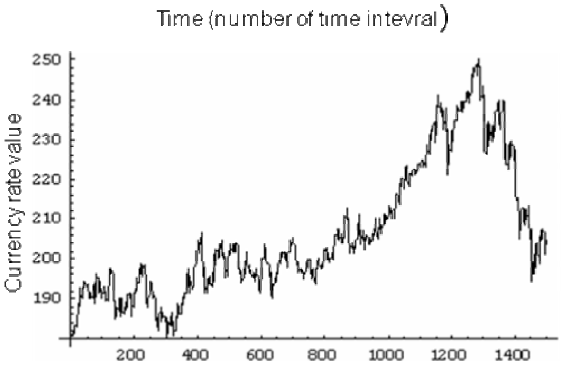


Fig. 9. Realisation of currency rate time series of currency pair GBP/JPY, time frame D1

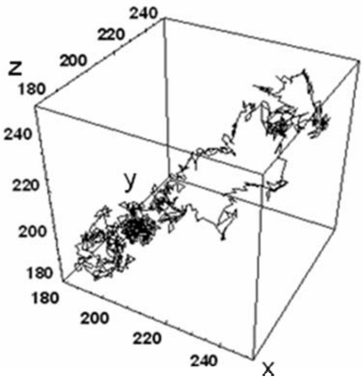


Fig. 10. Reconstructed attractor for GBP/JPY in a 3-dimensional lag space, D1, optimal $\tau = 30$

Reconstruction of the strange attractor for CAD/JPY, time frame 1WEEK

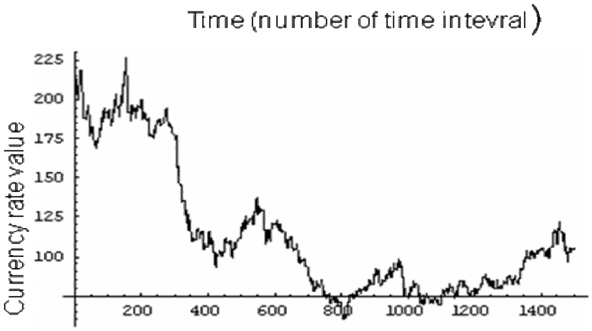


Fig. 11. Realisation of currency rate time series of currency pair CAD/JPY, time frame 1WEEK

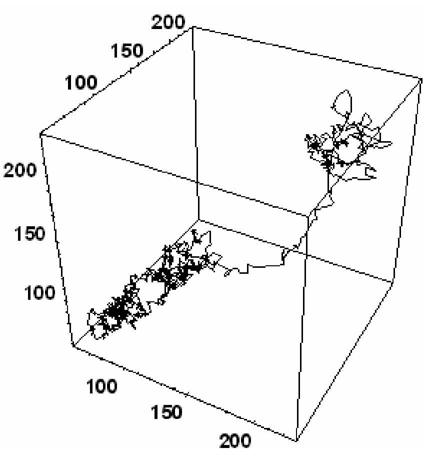


Fig. 12. Reconstructed attractor for CAD/JPY in a 3-dimensional lag space, 1WEEK, optimal $\tau = 20$

As we can see from Fig. 5 – 12, the trajectories of the reconstructed attractors run within strict space and do not look like a cloud of point as it is shown for the random system (Fig. 4). Thus, we can make a preliminary conclusion that the reconstructed attractors are strange attractors and are produced by chaotic systems.

Here the optimal value of time delay τ has been chosen experimentally in order to achieve the best result (the reconstructed attractor should not look too stretched or too spread in the lag space)

III. Estimation of the Correlation dimension in m-dimension space

A strange attractor of a chaotic system is a fractal and has fractal dimension [13][16]. For a true chaotic system its reconstructed attractor saves its dimensionality even if it is embedded into a lag space with a higher dimensionality.

According to the Grassberger-Procaccia method, a good approximation of the fractal dimension of a strange attractor is the *correlation dimension* [17]. Thus, if we reconstruct an attractor using the time-delay reconstruction method for various values of m (here m is called embedding dimension) and estimate the correlation dimension of the reconstructed attractor for various values of embedding dimension, then for a true chaotic system the value of correlation dimension will saturate at its true value which is a finite number. And the correlation dimension for the reconstructed attractor of a random system never stops growing.

If there is time series $x_1, x_2, ..., x_M$ and an attractor is reconstructed in m-dimensional lag space, then the *correlation sum* is the probability of that a pair of points on the reconstructed attractor lie within a distance ε. The correlation sum can be estimated using the formula

$$C(\varepsilon) = \frac{1}{M^2} \cdot \sum_{ij=1}^{M} \theta(\varepsilon - \| x_i - x_j \|), \qquad (3)$$

where $\theta(x)$ is the Heaviside step function:

$$\theta(x) = \begin{cases} 1, \text{ if } & \varepsilon - \| x_i - x_j \| > 0 \\ 0, \text{ if } & \varepsilon - \| x_i - x_j \| \le 0 \end{cases}$$

If $C(\varepsilon)$ is estimated for various ε, then $C(\varepsilon) \sim \varepsilon^{D(m)}$, where D(m) is the correlation dimension. D(m*)* can be estimated as a slope of a line fitted to a small ε-tail of the curve on a log-log plot of $C(\varepsilon)$ against ε.

Let us first estimate the correlation dimension for the Lorenz system:

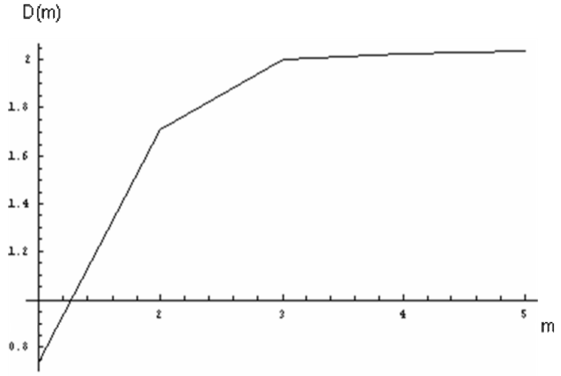


Fig. 13. Lorenz system. Estimation of the correlation dimension

We can see that for a true chaotic system the correlation dimension saturates at a finite value (D(m) = 2.03 for the Lorenz system). The true value of D(m) has been reached at the value of embedding dimension m=3.

Let us now estimated the correlation dimension for the random time series generated by the autoregression equation:

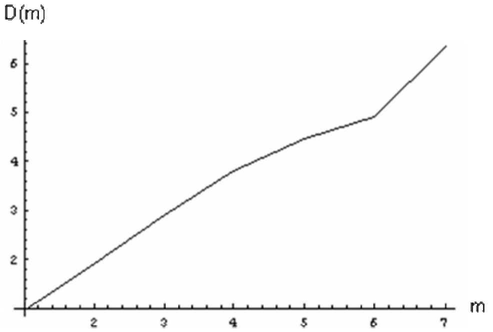


Fig. 14. Autoregression process. Estimation of the correlation dimension

As we can see in Fig. 14, the correlation dimension for a random system never stops growing.

Let us now estimate the correlation dimension for the currency rates time series for which the attractors have already been reconstructed.

Estimation of the correlation dimension for EUR/USD, time frame H1

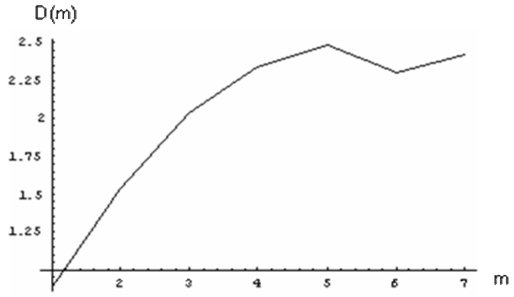


Fig. 15. Currency rates time series for currency pair EUR/USD, time frame H1. Estimation of the correlation dimension

For the considered time series the correlation dimension saturates at the value D(m)=2.5 starting from the embedding dimension m=5.

Estimation of the correlation dimension for NZD/CAD, time frame H4

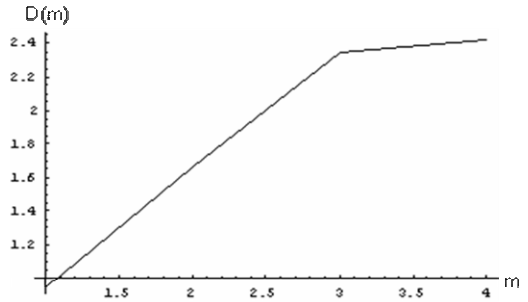


Fig. 16. Currency rates time series for currency pair NZD/CAD, time frame H4. Estimation of the correlation dimension

For the considered time series the correlation dimension saturates at the value D(m)=2.4 starting from the embedding dimension m=3.

Estimation of the correlation dimension for GBP/JPY, time frame D1

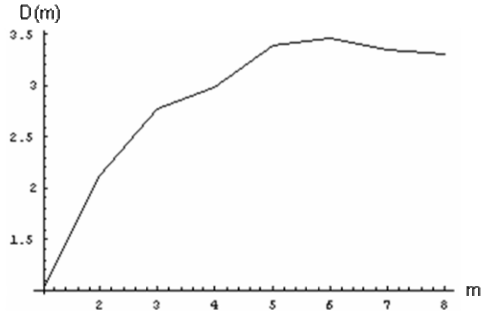


Fig. 17. Currency rates time series for currency pair GBP/JPY, time frame D1.Estimation of the correlation dimension

For the considered time series the correlation dimension saturates at the value D(m)=3.4 starting from the embedding dimension m=5.

Estimation of the correlation dimension for CAD/JPY, time frame 1WEEK

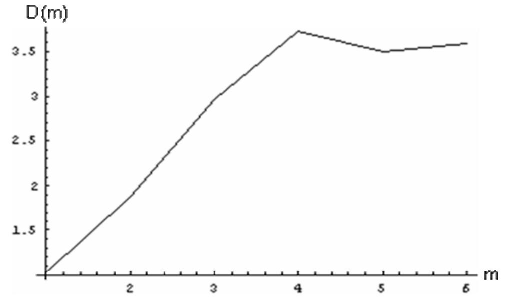


Fig. 17. Currency rates time series for currency pair CAD/JPY, time frame 1WEEK. Estimation of the correlation dimension

For the considered time series the correlation dimension saturates at the value D(m)=3.7 starting from the embedding dimension m=4.

Summary of the results of the estimation of the correlation dimension of the considered data can be found in Table I.

TABLE I
SUMMARY OF THE ESTIMATION OF THE CORRELATION DIMENSION

	D(m)					
m	Lorenz system	Random system	EUR/USD H1	NZD/CAD H4	GBP/JPY D1	CAD/JPY W1
1	0.7352	0.9812	0.8913	0.9427	1.0266	1.0174
2	1.7111	1.9102	1.5353	1.6627	2.1257	1.8776
3	2.0212	2.8898	2.0287	2.3553	2.7628	2.9508
4	2.0246	3.8084	2.3383	2.4235	2.9858	3.7092
5	2.038	4.4817	2.4801		3.3863	3.4996
6		4.8971	2.2998		3.4693	3.5864
7		6.3489	2.4148		3.3494	
8					3.3198	

For the estimation of the correlation dimension for all the considered samples of time series we have used the optimal values of time delay τ which have been obtained during the procedure of the attractor reconstruction. Also, the values of ε should not bee too small in order to obtain enough point for the statistical estimation of the correlation sum [17]. On other hand, ε should not be too big, otherwise it becomes comparable to the size of the attractor. For the estimation of the correlation dimension for EUR/USD we have used M=1000 points. And for all other currency pairs M=1500 points have been used. According to the Eckmann and Ruelle estimation [18] the maximum value of correlation dimension which is allowed to estimate using number of point M is defined by the formula:

$$D_{\max} \cong 2\lg M \qquad (4)$$

Thus, for M=1000 it is allowed to estimate the correlation dimension $D(m) \leq 6$. For all the considered samples of time series the estimated correlation dimension does not exceed D(m) = 3.5. Thus, the estimation of the correlation dimension is reliable for the given samples.

We have estimated the correlation dimension for currency rates time series of various currency pairs considered on small time frames, such as H1, H4, D1, 1WEEK. For all them the correlation dimension saturates at some finite value. It shows that on small time frames the currency rates time series are produced with deterministic system.

IV. Estimation of the Lyapunov Exponent

Deterministic chaos can appear in a dynamical system only if there is a sensitive dependence on its initial conditions. Such sensitivity is measured by the largest Lyapunov exponent (it is often called just the Lyapunov exponent). The Lyapunov exponent measures the divergence of initially close trajectories. For chaotic systems the Lyapunov exponent is always $\Lambda > 0$ [19].

The Lyapunov exponent has been estimated for the currency rates times series for which the attractors have been reconstructed and the correlation dimension has been estimated above in this paper. To estimate the Lyapunov exponent the Wolf algorithm has been used [18], which allows to estimate the exponent just using a one-dimensional time series. Also, to prove the efficiency of the estimation method the Lyapunov exponent has been estimated for the Lorenz system. The results of the estimation can be found in Table II:

TABLE II
THE RESULTS OF ESTIMATION OF THE LYAPUNOV EXPONENT FOR THE CURRENCY RATES TIME SERIES

	EUR/USD H1	NZD/CAD H4	GBP/JPY D1	CAD/JPY W1	Lorenz system
Λ	0.37	0.39	0.42	0.4	1.37

As it can be seen in Table II, all the investigated time series have the Lyapunov exponent greater than zero and thus, they are chaotic.

Since the Lyapunov exponent measures the divergence of initially close trajectories, then the less is the value of Λ, the less the initially trajectories diverge. Thus, the rate $1/\Lambda$ defines the predictability of a system.

V. Conclusion

In this paper we investigate the time series of original values of currency rates on the FOREX market which have been measured on small times frames such as 1 hour (H1), 4 hours (H4), 1 day (D1), 1 week (1WEEK). Such time frames are the most common used by traders during speculative trade operations. As a result of the investigation of the considered time series with the methods of Chaos Theory, it has been found that the time series of original values of currency rates on the FOREX market are chaotic on small time frames. Thus, such time series have a memory about its values in the past and we can predict its values in the future. This can be used in development of more efficient technical indicators for predicting price movement on FOREX which will help to traders to conduct the conversion operations with better success. Also, this allows to consider use of time series measured on small time frames in other prediction methods where only long time frames have been used before.

The fact, that the time series of original values of currency rates are chaotic, gives the opportunity to develop new prediction methods based on using original values of currency rates without pre-processing of initial data (transformation it to profits etc.). This can save time and resources while using such prediction methods. This is even more essential when the forecast needs to be done in the real-time mode (e.g. while conducting trade operations on FOREX)

References

[1] *Eric Naiman*. Short Trader's Encyclopedia. Alpina Business Books. (2007)

[2] FOREX Online User Manual for Successful Trading. (www.instantforexincome.com)

[3] *Markowitz H.M.* Portfolio Selection, Journal of Finance 7 (1953)

[4] *Litner, J.* The Valuation of Risk Assets and the Selection of Risky Investments in Stock Portfolios and Capital Budgets. Rev. Econ. Statist. (Feb 1965)

[5] *Mossin J.* Equilibrium in a Capital Asser Market. Econometrica 34 (1966)

[6] *Edgar E. Peters*. Chaos and Order in the Capital Markets: A New View of Cycles, Prices, and Market Volatility. Wiley; 2 edition (August 1996)

[7] *Edgar E. Peters*. Fractal Market Analysis: Applying Chaos Theory to Investment and Economics. Wiley; 1 edition (January 12, 1994)

[8] *A.S. Soofi & L. Cao*. Modelling and forecasting financial data: techniques of nonlinear dynamics. Norwell, Massachusetts: Kluwer academic publishers (2002

[9] *F. Mun*. Chaotic Vibrations: Wiley, New York (1987)

[10] *Shuster, H. G.* Deterministic Chaos: An Introduction, second revised edition. VCH Publishers, New York (1988)

[11] *M. Corazza & A.G. Malliaris*. Multi-Fractality in Foreign Currency Markets, Multinational Finance Journal, 6(2), 65-98 (2002)

[12] *Gian Italo Bischi, Carl Chiarella, Laura Gardini*. Nonlinear Dynamics in Economics, Finance and the Social Sciences: Springer; 1st Edition. edition (January 12, 2010)

[13] *D. Ruelle*. Small random perturbations of dynamical systems and the definition of attractors. Communications of Mathematical Physics 82: 137–151. (1981)

[14] *F. Takens*, *D. A. Rand* and *L.-S. Young*. Detecting strange attractors in turbulence. Dynamical Systems and Turbulence, Lecture Notes in Mathematics, vol. 898: 366–381, Springer-Verlag (1981)

[15] *Edward Lorenz*. The Essence of Chaos. University of Washington Press. University of Washington Press (April 1996)

[16] *Benoît B. Mandelbrot.* The Fractal Geometry of Nature. New York: W. H. Freeman and Co., (1982)

[17] *P. Grassberger, T. Schreiber, and C. Schaffrath, Int. J.* Bifurcation and Chaos 1, 521 (1991)

[18] *Eckmann J.-P., Kamphorst S., Ruelle D., Ciliberto S.* Lyapunov exponents from a time series //Phys.Rev.A. -1986. -V.34. -P.4971-4980

[19] *V. I. Oseledets*. Multiplicative ergodic theorem: Characteristic Lyapunov exponents of dynamical systems, Trudy MMO 19, 179–210 (1968)

Olga Dabi-Prashad was born in 1983, Ukraine.

In 2004 graduated from Kharkiv National University of Radio Electronics, Ukraine, with Bachelor's degree in Applied Mathematics.

In 2009 defended her thesis entitled "Using Chaos Theory and Oscillators for Predicting Price Movement in FOREX" at the Wessex Institute of Technology (University of Wales), the United Kingdom, and has been awarded the Degree of Master of Philosophy in Engineering.

Ms. Dabi-Prashad's scientific interests focus on quantitative analysis in financial markets, Chaos Theory, fractal and multifractal analysis.

Ludmila Kirichenko was born in 1964.

In 1986 graduated from Kharkiv Institute of Radio Electronics, Ukraine, with Master's Degree in Applied Mathematics.

In 2000 defended the thesis for a Candidate Degree at Kharkov State Technical University of Radioelectronics, Ukraine, and has been awarded the Candidated Degree in Technical Science (specialization: mathematical modelling and computing methods).

At present time, Dr. Kirichenko holds the position of Associate professor in Department of Applied Math, Kharkiv National University of Radio Electronics.

Dr. Kirichenko's scientific interests focus on deterministic and chaotic systems, stochastic self-similar processes, time series modelling and forecasting, fractal and multifractal analysis. She is the author of over 50 publications on the listed subjects.

Structural Analysis Technique and Bad Synchronization Styles

Melnik D., Lukashenko O.

***Abstract* — This paper discusses early detection of potentially missing synchronizers on clock domain crossing paths, using structural static analysis.**

***Index Terms* — Clock Domain Crossing, Metastabillity, Static Verification**

I. Introduction

THE number of independent clock domains found on the typical today's device is continuously growing. According to the latest industry research, the average number of clock domains on a single device is >15—20 and it becomes higher and higher from day to day. The CDC-related design flaws are also growing exponentially, appearing to be very dangerous as the roots of intermittent chip failures (can be found only in the silicon). Static CDC verification is considered as one of the first de-facto steps in today's SoC design methodology; only static techniques can work as soon as the RTL starts taking shape [1].

The sections of logic elements that driven by clocks coming from different sources are called clock domains [2]. The signals that interface between asynchronous clock domains are called the clock domain crossing (CDC) signals (see Figure 1). The DATA_A signal is considered as an asynchronous signal into the receiving clock domain (no constant phase and time relationship exists between CLK_A and CLK_B).

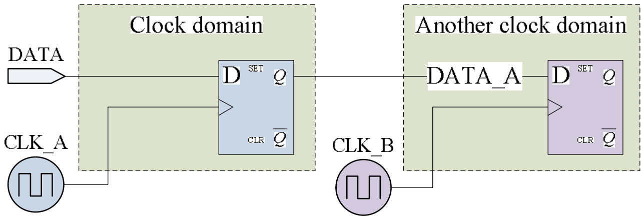


Fig. 1. Clock domains and CDC signal

The nature of CDC bugs is intermittent; it simply means that a test suite can be successfully completed on a chip in the morning, but the same tests will complete with errors for the same chip in the afternoon [3]. Consider the simplest flip-flop example: such a flip-flop is located anywhere in the chip; the data signal for this flip-flop comes from the domain #A but the clock signal — from the domain #B... so whenever the setup or hold condition is violated, the flip-flop can go to one or to zero and it cannot be predicted (see Figure 2).

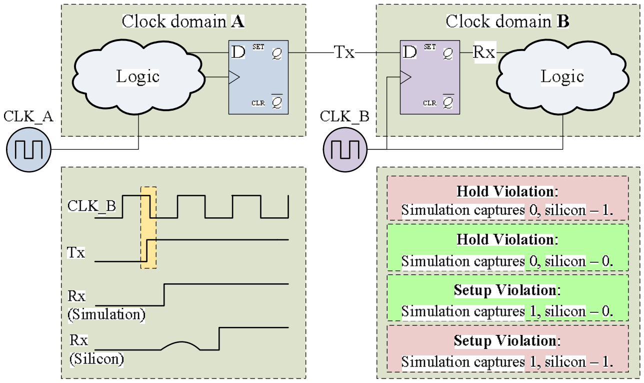

Clock domain A
Clock domain B
Logic
Logic
D SET Q
CLR Q̄
Tx
Rx
CLK_A
CLK_B
CLK_B
Tx
Rx (Simulation)
Rx (Silicon)
Hold Violation: Simulation captures 0, silicon – 1.
Hold Violation: Simulation captures 0, silicon – 0.
Setup Violation: Simulation captures 1, silicon – 0.
Setup Violation: Simulation captures 1, silicon – 1.

Fig. 2. Possible metastability effects

The metastability term is used to describe what happens in digital circuits when the clock and data inputs of a flip-flop change values at approximately the same time. As shown in the Figure 2, it leads to the flip-flop output oscillating and not settling to a value within the appropriate delay window [4]. Such glitches happen in every design wherein two or more discrete systems communicate (the number of clock domains is greater than two).

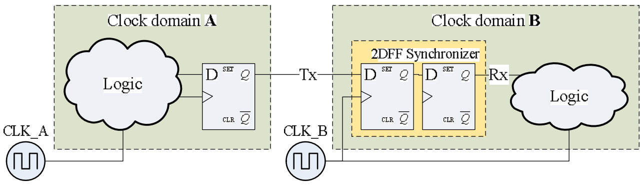


Fig. 3. Simplest synchronizer comprising 2 DFFs in series

Designers have actually found a solution to this and most of them is aware that metastability can be controlled using synchronizers on CDC signals (outputs of metastable registers are isolated so that the metastable value does not propagate to downstream logic) [5, 6]. Whenever there is a

Manuscript received .November 9, 2009

Melnik D. is with Kharkiv National University of Radioelectronic, Kharkiv, Lenin avenue, 14, Ukraine; (e-mail: melnyk.dima@gmail.com).

Lukashenko O. is with Kharkiv National University of Radioelectronic, Kharkiv, Lenin avenue, 14, Ukraine; (e-mail: olga.lukashenko@gmail.com).

domain crossing signal, two flip-flops are placed one next to the other clocked by the same clock (see Figure 3). Such a synchronization structure decreases the MTBF (Formula 1, where the f_{clk} – clock frequency, f_{in} – input signal frequency, t_d – duration of critical time window) from hours to thousands years [4].

$$MTBF = \frac{1}{f_{clk} \times f_{in} \times t_d} \quad (1)$$

Formula (1) means (average) time between failures. Recent trends have been in favor of using static analysis tools [1]. But the biggest disadvantage of this approach is that it comes pretty late in the game — after the design has been synthesized, and the gate-level netlist is available (finding a CDC at this stage — which needs to be fixed — could set the design schedule totally off). So there is a need in static analysis tool that:

1. Performs lightweight synthesis (netlist synthesis emulation) directly from the RTL description — alongside with Verilog, VHDL or SystemVerilog compilation.
2. Reports domain crossing paths with potentially missing synchronizers, thus providing an obvious advantage in the form of early checking.

II. Automatic Clock Domains Extraction

A. Detect Global Clock in a Design

Clock domains extraction with further synchronizers detection is illustrated by the dataflow that is shown in the Figure 4. It involves several steps, starting with the compilation of the RTL description and creation of the database with netlist elements (lightweight synthesis), proceeding with special attributes assignment and their propagation through a design hierarchy (global clocks detection), and further manipulations with global clocks (clock domains look up).

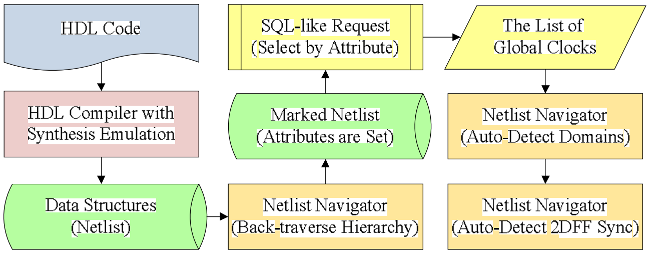

HDL Code
HDL Compiler with Synthesis Emulation
Data Structures (Netlist)
Netlist Navigator (Back-traverse Hierarchy)
Marked Netlist (Attributes are Set)
SQL-like Request (Select by Attribute)
The List of Global Clocks
Netlist Navigator (Auto-Detect Domains)
Netlist Navigator (Auto-Detect 2DFF Sync)

Fig. 4. Clock domains and synchronizers detection dataflow.

Attributes are distributed through the design hierarchy (netlist): “DESIGN_CLOCK” attribute is back-propagated from each flip-flop clock pin. Since all the netlist elements were added to the database, it further can be used for selection by the particular attribute(s) presence (SQL-like request).

- The back-propagation of the attribute is terminated on the storage elements (flip-flops and latches) and tri-states. While back-propagation is stopped, it means that the signal which feeds the flip-flop clock pin is not an external input signal and thus it cannot be considered as a global clock.
- However, if the attribute reaches an external input pin (passes only through combinatorial logic, buffers and inverters), it is considered as a global clock – added to the list of global clocks (see Figure 5).
-

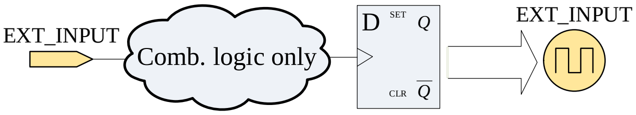


Fig. 5. Global clock is auto-detected by direct connection to external input pin

B. Extract Clock Domains

Clock domains can be detected when the list of global clocks is available; each global clock creates at least one separate clock domain. In order to detect clock domains, global clocks should be propagated through the design hierarchy (external input pins marked with the “DESIGN_CLOCK” attribute):

- Transparent logic. Combinatorial logic, latches and tri-states that happen on the attribute propagation path are considered as transparent objects.
- Flip-flops consideration. Each flip-flop that happens on the propagation path is added to the appropriate clock domain if “DESIGN_CLOCK” attribute reaches its clock input pin; if a flip-flop clock pin is driven by the output of another flip-flop which already belongs to a clock domain, the flip-flop is also added to the same clock domain (Figure 6)

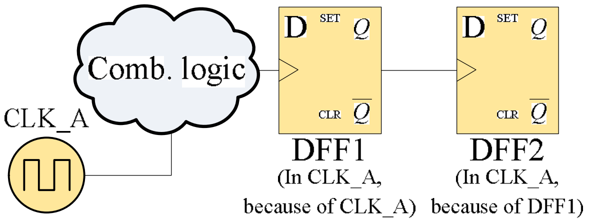


Fig. 6. Flip-flops are added to corresponding domains

- Derived domains. If two or more clock signals are propagated through the same combinatorial logic or multiplexer then the output of this logic or multiplexer derives a new clock signal that correspondingly results in a new clock domain for subsequent connections (see Figure 7). Also if a clock signal is connected to the multiplexer select pin then the output of this multiplexer derives a new clock signal.

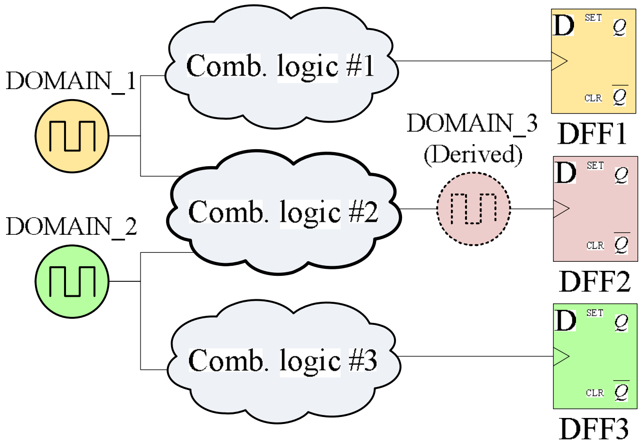


Fig. 7. Derived domains

Design is considered to be in a single clock domain if clock domains were not detected.

III. Detecting potentially non-synchronized paths

While the netlist is marked with clock domain-related attributes, the data about each flip-flop membership is available, it becomes possible to go further and detect synchronized and potentially non-synchronized CDC paths.

A. Detecting missing 2DFF synchronizer

In order to be considered as 2DFF synchronizer, a pair of flip-flops should comply with the following restrictions:

1. Each flip-flop should receive the data only from the same clock domain (correct case – FF#1 receives data from domain A and transmits it to FF#2; incorrect case – FF#1 receives data from domain A, FF#2 receives data from domain B).

2. The outputs of the first and second flip-flops should not be connected to external design output(s) (in each case, the propagation should be blocked by non-clock input of another flip-flop(s) from the same domain).

It should be noted, that for some very high speed designs, the MTBF of a two-flop synchronizer is too short and a third flop is added to increase the MTBF to a satisfactory duration of time [7].

The paths which does not pass through a 2DFF synchronizer upon arrival into the new clock domain can be considered as potentially non-synchronized and reported as the design rule violations (synchronization errors class, see Figure 8).

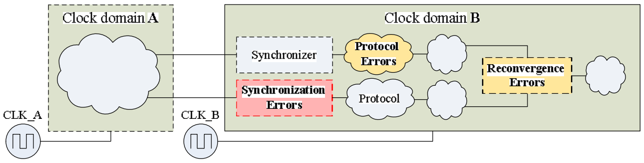


Fig. 8. Synchronization and other potential CDC errors

B. Detect hazardous transfers

But other incorrect design patterns should be detected when transmitting the data between asynchronous clock domains – except the direct transfers (missing synchronizer).

A simple synchronizer comprises two flip-flops in series without any combinatorial circuitry between them. This approach ensures that the first flip-flop exits its metastable state and its output settles before the second flip-flop samples it. For proper work of such synchronization, the signal crossing a clock domain should pass from flip-flop in the original clock domain to the first flip-flop of the synchronizer without passing through any combinatorial logic. First flip-flop of a synchronizer is sensitive to glitches that combinatorial logic produces (glitch that occurs at the correct time could meet the setup-and-hold requirements of the first flip-flop in the synchronizer, causing the synchronizer to pass a pseudo-valid signal to the rest of the logic in the target clock domain). Therefore, combination logic should not be located between asynchronous clock domains, because it significantly increases the risk to propagate pseudo-valid value to downstream logic.

The following pattern is forbidden:

1. Data input of the FF#1 in the receiving domain is fed by combinatorial logic output

2. Any of the logic inputs is(are) fed by data from FF of the transmitting domain.

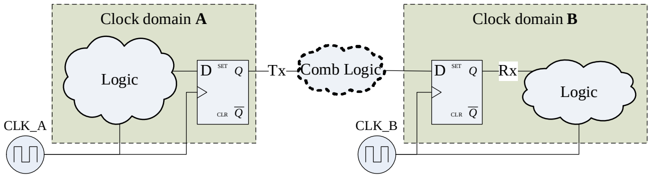


Fig. 9. Combinatorial logic between domains.

If combinatorial logic is located between two synchronizing flip-flops, the second flip-flop becomes sensitive to glitches produced by combinatorial logic – a possibility to propagate pseudo-valid value to downstream logic increases significantly and synchronizer could become useless in this case. Following conditions are characterized the situation:

1. FF#1 of the receiving domain feeds some combinatorial logic

2. Any of the logic outputs feeds receiving domain FF.

Combinatorial logic placed as described at the Figure 10 may also be treated as synchronizer that consists of one FF only. Above description explains that it is not enough to ensure metastability convergence.

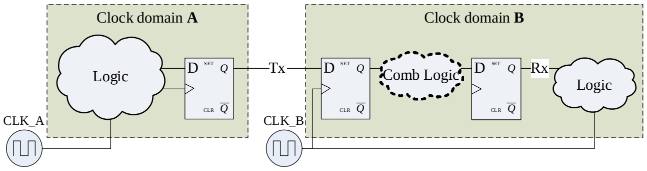


Fig. 10. Combination logic between after FF#1 in receiving domain.

C. Detect hazardous reset lines

Section 2.1 describes global clock signals detection. The same algorithm may be used to detect global reset signals using the netlist database. “DESIGN_RESET” attribute is back-propagated from each flip-flop reset pin.

- The back-propagation of the attribute is terminated on the storage elements (flip-flops and latches) and tri-states. While back-propagation is stopped, it means that the signal which feeds the flip-flop reset pin is not an external input signal and thus it cannot be considered as a global reset.
- However, if the attribute reaches an external input pin (passes only through combinatorial logic, buffers and inverters), it is considered as a global reset.

The global reset leading edge is safe because it set all the circuits to a known starting state. While reset trailing edge is not so harmless [8]. During the global reset all the clocks are started. But when the reset is removed it may happen simultaneously with the sampling edge of one of the clocks. Thus some FFs may enter metastable state. To prevent this situation synchonizer should be used for the global reset trailing edge. The proper synchronization circuit is shown at Figure 11. The leading edge is transferred directly and trailing edge is synchronized properly.

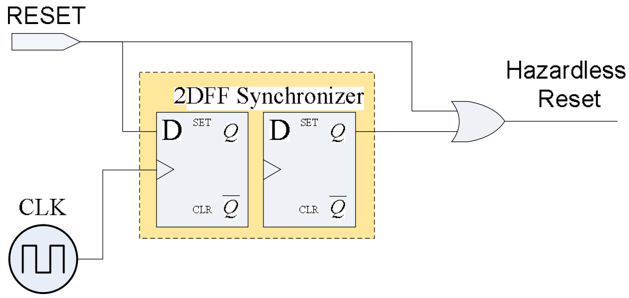


Fig. 11. Global reset synchronization

Internally generated asynchronous reset (set) signal may also be transferred from one clock domain to another. Thus it combines both situations described above. The signal can lead to asynchronous domains related problems as it crosses domain boundaries. On the other hand removal of the reset may coincide with receiving clock sampling edge and so also lead to metastability.

The both methods can be used to solve the problem.

- Asynchronous reset line is synchronized with 2DFF synchronizer
- Asynchronous reset trailing edge is synchronized (Figure 12)

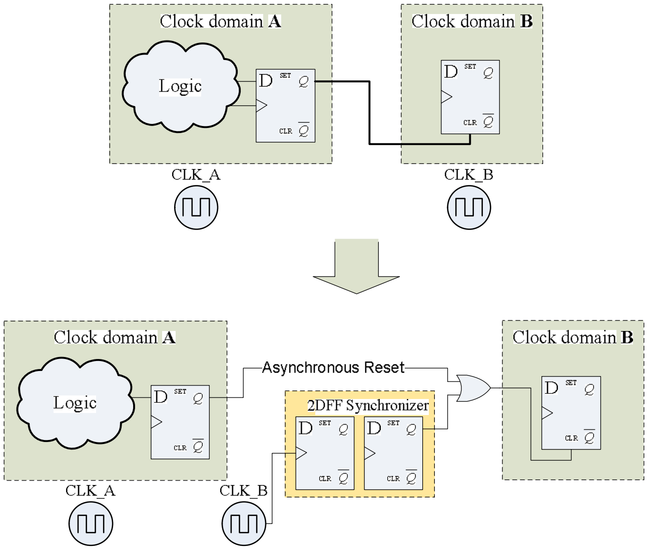


Fig. 12. Synchronized-trail asynchronous clear.

IV. Conclusion

Proposed structural analysis technique includes building of a netlist of the target design (lightweight synthesis is performed alongside with compilation) and performing further static analysis on this netlist. The novelty of the approach concerns propagation of various attributes through a design hierarchy: once the database with “netlist element”—”attribute(s)” relations is prepared, it can be used for SQL-like selections by attribute. The result of analysis is a summary of CDC paths in the design where the synchronization is potentially missing or incorrectly implemented (data and reset transfers). Proposed technique deals only with the first of CDC problems list which can be detected with static analysis [8]:

1. Missing and incorrectly implemented synchronizers.
2. Correctly implemented synchronizer.
3. Complex synchronizers that require protocol verification.
4. Potential reconvergence problems.

To perform more complete CDC verification, formal analysis techniques should be used alongside with structural analysis (see Figure 9).

During the structural verification stage, it is possible to generate monitors for CDC transfer protocols. At the stage of formal verification [9], a simple reset sequence is used and cycle-based design analysis is performed (requires knowledge about clock periods of the asynchronous clocks). While a monitor is proven, it means that CDC protocol is followed.

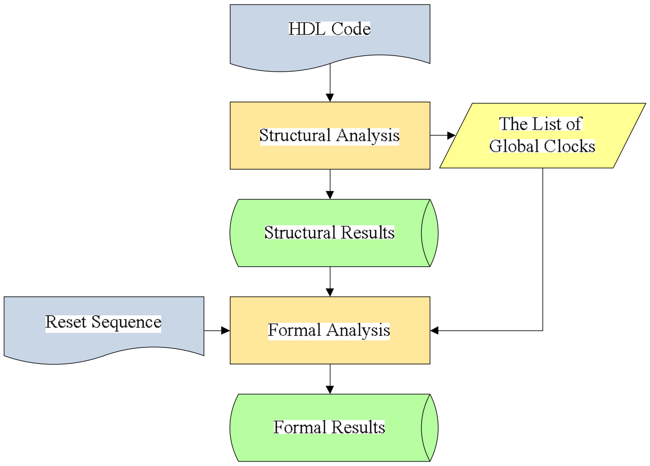


Fig. 13. Structural and formal analysis

References

[1] Sanjay Churiwala and Sapan Garg of Atrenta, and Chirag Gupta and Paresh Joshi of Texas Instruments, “Verification of Clock Domain Crossing in SoCs: Part One — Tools and Needs”. Downloadable from www.chipdesignmag.com.

[2] Cadence technical paper, “Clock Domain Crossing. Closing the Loop on Clock Domain Functional Implementation Problems”. Downloadable from w2.cadence.com/whitepapers/cdc_wp.pdf.

[3] Mentor Graphics technical webinar, “Finding and Eliminating CDC Errors with 0-In CDC Verification”. Downloadable from www.mentor.com/player/2007/zero_in_cdc/index.html.

[4] Michelle Lange of Mentor Graphics, "Automating Clock-Domain Crossing Verification for DO-254 (and other Safety-Critical) Designs". Downloadable from www.do254.com/documents/Papers/Mentor_CDC-for-DO254.pdf.

[5] Clifford E. Cummings, SNUG-2001, “Synthesis and Scripting Techniques for Designing Multi-Asynchronous Clock Designs”. Downloadable from www.sunburst-design.com/papers.

[6] Tai Ly, “The Need for an Automated Clock Domain Crossing Verification Solution”. Downloadable from www.mentor.com/fv.

[7] Clifford E. Cummings, SNUG-2008, "Clock Domain Crossing (CDC) Design & Verification Techniques Using SystemVerilog". Downloadable from www.sunburst-design.com/papers.

[8] Ran Ginosar, “Fourteen Ways to Fool Your Synchronizer”, Proceedings of the Ninth International Symposium on Asynchronous Circuits and Systems (ASYNC'03)

[9] Ping Yeung of Mentor Graphics, “Five Steps to Quality CDC Verification”. Downloadable from www.mentor.com/products/fv/techpubs/.

[10] Mentor Graphics, “Formal Verification User Guide V2.5”. Feb 2007.

Melnik Dmitry, Ph.D student of Design Automation department, KNURE. Scientific interests: multiple clock domains verification in complex hardware designs, safety-critical systems design and verification. Address: KNURE, Kharkov, 61166 14 Lenin Avenue.

Lukashenko Olga, Ph.D student of Design Automation department, KNURE. Scientific interests: functional verification, multiple clock domains verification, coverage-driven verification. Address: KNURE, Kharkov, 61166 14 Lenin Avenue.

Brain-Like Computer Structures

Vladimir Hahanov, *Member, IEEE,* Svetlana Chumachenko, Ngene Christopher Umerah, Tiecoura Yves
Computer Engineering Faculty, Kharkov National University of Radioelectronics, Kharkov, Ukraine
hahanov@kture.kharkov.ua

Abstract − **High-speed multiprocessor architecture for brain-like analyzing information represented in analytic, graph- and table forms of associative relations to search, recognize and make a decision in n-dimensional vector discrete space is offered. Vector-logical process models of actual applications, where the quality of solution is estimated by the proposed integral non-arithmetical metric of the interaction between binary vectors, are described.**

I. Introduction

The goal is to remove arithmetic from computer and transform free resources to the brain-like infrastructure of associative logic simulating the brain functionality that makes possible making the right decision every moment. The brain and the computer have the same technological basis in the form of primitive logical operations: and, or, not, xor. With experience, the brain and the computer create more complex functional space-time logic converters using the above primitive operations. Specialization of computer, focused on using only logical operations, enables to approximate to the associative logic human thinking, and thus considerably (x100) improve the performance of solving nonarithmetic problems.

Removing arithmetic operations, leveraging the parallelism of the vector logic algebra, and multiprocessor architecture provide an efficient infrastructure, which combines mathematical and technological culture to solve applied problems.

Brain-likeness of multiprocessor digital system-on-a-chip is the concept of making an architecture and models of computational processes to implement typical brain nonarithmetic associative logic functionalities on today's digital platform by using vector logical operations and criteria for search, pattern recognition and decision-making problems. Market appeal of logical associative multiprocessor (LAMP) is determined by thousands of old and new logical problems, which now are solved ineffectively by redundant universal computers with high-performance arithmetic processor. Here are some problems relevant to the IT-market: 1. Analysis and synthesis of syntactic and semantic language structures (abstracting, error correction, analysis of the text quality). 2. Video and audio pattern recognition by means of their representation by vector models of essential parameters in discrete space. 3. Use of Infrastructure IP for complex technical products to ensure their manufacturability and lifetime reliability. 4. Knowledge testing and expert appraisal of objects or parties to determine their validity. 5. Identification of the object or process to make a decision under uncertainty. 6. Exact information retrieval in the Internet, if information is given by a vector of parameters. 7. Target designation of fighter or aircraft autoland system functioning in microsecond time. 8. Air traffic control or optimization of municipal traffic control infrastructure to avoid conflicts. Practically all these problems are solved in real time; they are isomorphic by the logical structure of the process models, based on a set of interrelated associative tables. To solve them it is necessary quick and dedicated hardware platform (LAMP), focused on the concurrent execution of search, recognition and decision-making procedures, estimated by means of the integral nonarithmetic quality criterion.

Our goal in this article is to increase considerably (x100) the speed of search, recognition and decision-making procedures by means of multiprocessor and concurrent implementation of associative logic vector operations for the analyzing graph and tabular data structures in discrete Boolean space without the use of arithmetic operations.

The problems are: 1) Developing nonarithmetic metric for estimating the associative logic solutions. 2) Creation of data structures and process models for solving the applied problems. 3) Designing the architecture of logical associative multiprocessor. 4) Implementation of LAMP.

Essence of the research is the infrastructure for expert servicing of requests in real time integrating multiprocessor system-on-chip with associative-logical data structures to obtain a deterministic solution, the validity of which is estimated by nonarithmetic integral interaction quality criterion of a query and given discrete space.

References: 1. Hardware platform for associative logical information analysis [1-2]. 2. Associative logical data structures for solving the information problems [3-4]. 3. Models and methods for discrete analyzing and synthesizing [5-6]. 4. Multiprocessors for solving information-logical problems [7-10]. 5. Brain-like and intelligent logical computing [11-12].

Manuscript received October 19, 2009.

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).

Chumachenko Svetlana 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).

II. Integral Metric For Solution Estimation

Infrastructure of brain-like multiprocessor includes models, methods and associative logical data structure, focused on hardware support of search, recognition and decision-making processes [11-12] on the basis of vector nonarithmetic operations.

Evaluation of problem solution is determined by the vector-logical criterion of interaction quality between a query (a vector m) and a system of associative vectors (associators). The query processing results in generating a positive response in the form of one or more associators, as well as the numerical grade of membership characteristic (quality function) of an input vector m to the obtained solution: $\mu(m \in A)$. The input vector $m = (m_1, m_2, ..., m_i, ..., m_q)$, $m_i \in \{0,1,x\}$ and the matrix A_i of associators $A_{ijr} (\in A_{ij} \in A_i \in A) = \{0,1,x\}$ have the same dimension that is equal to q. Below the membership grade of m-vector to A is designated by $\mu(m \in A)$.

There are 5 types of set-theoretic (logical) Δ - interaction of two vectors $m \cap A$ defined in Fig. 1. They form all primitive reactions of the generalized SRM systems (SRM – Search, Pattern Recognition and Decision Making) on the input request vector. In the technological field of knowledge – Design & Test – this sequence of actions is isomorphic to the route: fault finding, fault locating, decision-making for repairing. All three stages of technological route require the metric for estimating solutions to choose the optimal variant.

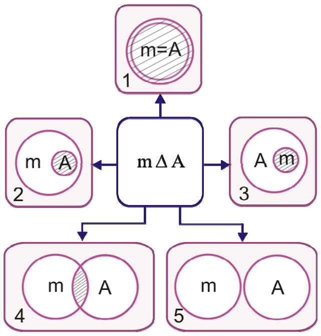


Fig. 1. The results of the intersection of two vectors

Definition. Integral set-theoretic metric for the estimating query quality is a function of the interaction of multivalued vectors $m \cap A$, which is determined by average sum of three normalized parameters: code distance $d(m, A)$, membership function $\mu(m \in A)$ and membership function $\mu(A \in m)$:

$$Q = \frac{1}{3}[d(m,A) + \mu(m \in A) + \mu(A \in m)],$$

$$d(m,A) = \frac{1}{n}[n - \mathrm{card}(m_i \bigcap_{i=1}^{n} A_i = \varnothing)];$$

$$\mu(m \in A) = 2^{\mathrm{card}(m \cap A) - \mathrm{card}(A)} \leftarrow \mathrm{card}(m \cap A) =$$

$$= \mathrm{card}(m_i \bigcap_{i=1}^{n} A_i = x) \,\&\, \mathrm{card}(A) = \mathrm{card}(\bigcup_{i=1}^{n} A_i = x); \qquad (1)$$

$$\mu(A \in m) = 2^{\mathrm{card}(m \cap A) - \mathrm{card}(m)} \leftarrow \mathrm{card}(m \cap A) =$$

$$= \mathrm{card}(m_i \bigcap_{i=1}^{n} A_i = x) \,\&\, \mathrm{card}(m) = \mathrm{card}(\bigcup_{i=1}^{n} m_i = x).$$

Explanations. The normalization of parameters makes it possible to estimate the level of vector interaction in the interval [0,1]. If it is fixed the limiting maximum value of each parameter equal to 1, it means the vectors are equal. The minimal estimation $Q = 0$ is fixed if the vectors are not coincided by all n coordinates. If intersection power $m \cap A = m$ is equal to half of A vector space, membership and quality functions are equal respectively:

$$\mu(m \in A) = \frac{1}{2};\ \mu(A \in m) = 1;\ d(m, A) = 1;$$

$$Q(m, A) = \frac{5}{2 \times 3} = \frac{5}{6}.$$

The same value will be setting for Q parameter if the power of intersection $m \cap A = A$ is equal to half of the vector space m. If the power of intersection $\mathrm{card}(m \cap A)$ is equal to half of the power of vector spaces A and m, membership functions are the following:

$$\mu(m \in A) = \frac{1}{2};\ \mu(A \in m) = \frac{1}{2};\ d(m, A) = 1;$$

$$Q(m, A) = \frac{4}{2 \times 3} = \frac{4}{6} = \frac{2}{3}.$$

It should be noted, if the intersection of two vectors is equal to the empty set, then the power of number 2 from the symbol "empty" is equal to zero: $2^{\mathrm{card}(m \cap A) = \varnothing} = 2^{\varnothing} = 0$. It really means that the number of common points in the intersection of two spaces is zero.

The aim of a new vector logical criterion of solution quality is improving considerably the performance of calculating the quality Q of interaction between the components m and A, when analyzing the associative data structures by using the vector logical operations only. The arithmetic criterion (1) without the averaging membership functions and code distance can be transformed to the form:

$$\begin{aligned}
&Q = d[m, A_{i(j)}] + \mu[m \in A_{i(j)}] + \mu[A_{i(j)} \in m],\\
&d(m, A_{i(j)}) = \mathrm{card}[m \bigoplus_{i(j)=1}^{n(m)} A_{i(j)} = 1];\\
&\mu(m \in A_{i(j)}) = \mathrm{card}[A_{i(j)} = 1] - \mathrm{card}[m \bigwedge_{i(j)=1}^{n(m)} A_{i(j)} = 1];\\
&\mu(A_{i(j)} \in m) = \mathrm{card}[m = 1] - \mathrm{card}[m \bigwedge_{i(j)=1}^{n(m)} A_{i(j)} = 1].
\end{aligned} \quad (2)$$

The first component of the criterion forms the degree of mismatch between n-dimensional vectors – the code distance, by performing xor operation, second and third ones determine the degree of non-membership of conjunction result to a set of "1" each of two interacting vectors. The notions of membership and non-membership are complementary, but calculating non-membership is more technological. Thus, the ideal criterion of quality is equal to zero, if two vectors are equal. The estimation of the interaction quality between two binary vectors is decreasing with increasing criterion from 0 up to 1. To finally remove arithmetic operations, when counting a vector quality criterion, it is necessary to transform the expressions (2) to the form:

$$\begin{aligned}
&Q = d(m, A) \vee \mu(m \in A) \vee \mu(A \in m),\\
&d(m, A) = m \oplus A;\\
&\mu(m \in A) = A \wedge \overline{m \wedge A};\\
&\mu(A \in m) = m \wedge \overline{m \wedge A}.
\end{aligned} \quad (3)$$

Here the criteria are not numbers, but vectors, which determine the interaction of components m, A. The increasing quantity of 0 in three quality vectors improves the criterion, and 1's indicate loss of interaction quality. To compare the estimations it is necessary to determine the power of 1's in each vector without performing addition operation. This can be done using the register [9-10] (Fig. 2), which makes it possible to perform left shifting and compacting all 1 coordinates of n-bit binary vector for one clock cycle.

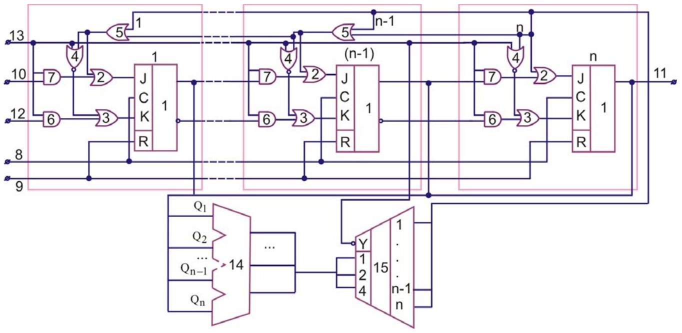


Fig. 2. Register for shifting and compacting 1's

After compacting procedure right unit bit number of compacted set of 1's determines the index of interaction quality for vectors. For binary sets $m = (110011001100), A = (000011110101)$ the determining their interaction quality by formulas (3) is shown in the following form (zero coordinates are marked by dots):

m	1	1	.	.	1	1	.	.	1	1	.	.
A	.	.	.	.	1	1	1	1	.	1	.	1
$m \wedge A$	.	.	.	.	1	1	.	.	.	1	.	.
$\overline{m \wedge A}$	1	1	1	1	.	.	1	1	1	.	1	1
$d(m, A) = m \oplus A$	1	1	.	.	.	.	1	1	1	.	.	1
$\mu(A \in m) = m \wedge \overline{m \wedge A}$	1	1	.	.	.	.	.	.	1	.	.	.
$\mu(m \in A) = A \wedge \overline{m \wedge A}$	.	.	.	.	.	.	1	1	.	.	.	1
$Q = d(m, A) \vee \mu(m \in A) \vee \mu(A \in m)$	1	1	.	.	.	.	1	1	1	.	.	1
$Q(m, A) = (6/12)$	1	1	1	1	1	1	.	.	.	.	.	.

It is formed not only the estimation of vector interaction that is equal to $Q(m, A) = (6/12)$, but the most importantly, the unit coordinates of the row $Q = d(m, A) \vee \mu(m \in A) \vee \mu(A \in m)$ identify all essential variables for which there is low-quality vector interaction. To compare two solutions obtained by logical analysis, compressed quality vectors Q are used; and vector procedure including the following vector operations is performed:

$$Q(m,A) = \begin{cases} Q_1(m,A) \leftarrow \mathrm{or}[Q_1(m,A) \wedge Q_2(m,A) \oplus Q_1(m,A)] = 0;\\ Q_2(m,A) \leftarrow \mathrm{or}[Q_1(m,A) \wedge Q_2(m,A) \oplus Q_1(m,A)] = 1. \end{cases} \quad (4)$$

Vector-bit or-operator of devectorization determines a binary bit solution on the basis of application a logical OR operation to n bits of an essential variables vector of quality criterion. A circuit design for decision

$$Q = \begin{cases} Q_1 \leftarrow Y = 0\\ Q_2 \leftarrow Y = 1 \end{cases}$$

and analytic process-model include three operations, shown in Fig. 3.

$$Y = \vee[(Q_1 \wedge Q_2) \oplus Q_1]$$

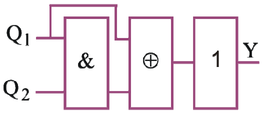


Fig. 3. Process-model of decision

For binary vectors which are quality criteria the procedure for choosing the best one on the basis of expression (4) is presented below:

$Q_1(m, A) = (6,12)$	1	1	1	1	1	1	.	.	.	.	.	.
$Q_2(m, A) = (8,12)$	1	1	1	1	1	1	1	1	.	.	.	.
$Q_1(m,A) \wedge Q_2(m,A)$	1	1	1	1	1	1	.	.	.	.	.	.
$Q_1(m, A) \oplus Q_1(m, A) \wedge Q_2(m, A)$	.	.	.	.	.	.	.	.	.	.	.	.
$Q(m, A) = Q_1(m, A)$	1	1	1	1	1	1	.	.	.	.	.	.

Vector logical criteria of interaction quality for associative sets enable to obtain estimation of the search, pattern recognition and decision-making with high-speed parallel logic operations, which is especially important for critical real-time systems.

The quality criterion Q uniquely determines three forms of interaction between any two objects in the n-dimensional vector logical space: the distance, and two membership functions. Taking into account that all three estimates included in the integral criterion form are joined by the function OR, simplification of vector interaction gives the result

$$\begin{aligned} & Q = d(m,A) \vee \mu(m \in A) \vee \mu(A \in m) = \\ & = m \oplus A \vee A \wedge \overline{m \wedge A} \vee m \wedge \ \overline{m \wedge A} = \\ & = m \oplus A \vee [A \wedge (\overline{m} \vee \overline{A})] \vee [m \wedge \ (\overline{m} \vee \overline{A})] = \\ & = m \oplus A \vee [A\overline{m} \vee A\overline{A} \vee m\overline{m} \vee m\overline{A}] = \\ & = (A\overline{m} \vee m\overline{A}) \vee [A\overline{m} \vee A\overline{A} \vee m\overline{m} \vee m\overline{A}] = \\ & = A\overline{m} \vee m\overline{A} \vee A\overline{m} \vee A\overline{A} \vee m\overline{m} \vee m\overline{A} = \\ & = m \oplus A. \end{aligned}$$

The quality criterion $Q = m \oplus A$ conforms to the metric for estimating distance or interaction in vector logical space, as well as it has a trivial computational form for estimating many solutions related to the analysis and synthesis of information. In fact, a logical vector space should not use the metric distance and quality criteria, including scalar arithmetic operations. WFC determines not only the distance between disjoint objects vector logical space, but also their mutual affiliation: if they intersect. Vector logic criterion (VLC) determines not only the distance between disjoint objects of a vector-logical space, but also their mutual membership: $d(a,b) \vee \mu(a,b) \vee \mu(b,a)$, if they overlap.

Vector discrete logical (Boolean) space defines the interaction of objects through the use of three axioms (identity, symmetry, and triangle), which form a non-arithmetic B-metric of vector dimension:

$$B = \begin{cases} d(a,b) = a \oplus b = (a_i \oplus b_i), i = \overline{1,n}; \\ d(a,b) = [0 \leftarrow \forall i(d_i = 0)] \leftrightarrow a = b; \\ d(a,b) = d(b,a); \\ d(a,b) \oplus d(b,c) = d(a,c), \\ \oplus = [d(a,b) \wedge \overline{d}(b,c)] \vee [\overline{d}(a,b) \wedge d(b,c)]. \end{cases}$$

Vertices of a transitive triangle are the vectors, which identify the objects of n-dimensional Boolean B-space. Sides of the triangle d(a, b), d(b, c), d(a, c) are the distances between the vertices. They are also vectors of the dimension n, where each bit is defined in the same alphabet as the coordinates of vertex vectors.

Vector transitive triangle is perfect analogy with the numerical distance of the metric M-space, which is determined by the system of axioms defining the interaction of one, two and three points in any space:

$$M = \begin{cases} d(a,b) = 0 \leftrightarrow a = b; \\ d(a,b) = d(b,a); \\ d(a,b) + d(b,c) \geq d(a,c). \end{cases}$$

The specific of metric triangle axiom is numerical (scalar) comparison of the distances for three objects, where the interval uncertainty of the expression “two triangle sides can be greater or equal to third one” is not usable to determine the exact length of the last side. Elimination of this disadvantage is able to be done only in logical vector space, which characterized by determinate representation of each parameter of the state for process or phenomenon. Then the numerical uncertainty of third triangle side in a vector logical space takes the form of the exact binary vector, which characterizes the distance between two objects and is calculated on the basis of information about the distances of other two triangle sides:

$$d(a,b) \oplus d(b,c) = d(a,c).$$

We can transpose the right component to the left side, which makes it possible to compress any closed space to a zero vector

$$d(a,b) \oplus d(b,c) = d(a,c) \to d(a,b) \oplus d(b,c) \oplus d(a,c) = 0.$$

Convolution of space to a zero vector is of interest for many practical problems, including: 1) Diagnosis and error correction when the transmitting the information via communication channels. 2) Fault detection in digital products based on two-valued and multivalued fault detection tables. The theoretical justification of space convolution is presented below. It includes the proving of correctness of the use vector Boolean space metric to determine the interaction between logical structures, including point, line, plane.

Axiom 1. For Boolean variables the logical expression is valid: $a = b \to a \oplus b = 0$. In fact, modulo 2 sum is a function of nonequivalence, which is true or unit value when variable values are not coincided $a \neq b \to a \oplus b = 1$ and zero – when arguments are coincided.

Definition 1. The distance between two points of n-dimensional space is a vector, calculated on the basis of the XOR-sum of the same name coordinates:

$$d(a,b) = a_i \overset{n}{\underset{i=1}{\oplus}} b_i.$$

Definition 2. The vector distance between two points (objects) of n-dimensional logical space is zero (one), if all components of the vector are zero (one):

$$d(a,b) = 0(1) \leftarrow \forall i[a_i \overset{n}{\underset{i=1}{\oplus}} b_i = 0(1)].$$

Definition 3. A simple chain is a sequence of vector distances and points in a space, not including equal components: distances or points

Definition 4. Vector logical cycle D is a set of vector distances $d_i \in D$ between points of a space, forming a closed simple chain, where the first and last points are coincided.

Theorem 1. Xor-sum of vector distances, forming a cycle, between two points in n-dimensional space is equal to zero: $d(a,b) \oplus d(b,a) = 0$.

This follows from the axioms of symmetry (commutativity) that the vector distances between any two points in n-dimensional space $d(a,b) = d(b,a)$ are equal. But, according to axiom 1, the transposition of the right side of equality to the left one is accompanied by the regulation of relations between the components using xor-operation, the result of which is equal to zero: $d(a,b) \oplus d(b,a) = 0$ in view of the equality of vector distances.

Theorem 2. Xor-sum of vector distances or two sides of the transitive triangle is equal to the third one.

Proof. In general, the metric defines the interaction of three points (a, b, c) (the triangle sides) in the space by means of the forming three distances $d(a,b), d(b,c), d(a,c)$. Assume that there are two distances $d(a,b), d(b,c)$, which, according to Theorem 1, define equality $d(a,b) \oplus d(b,c) = 0$ that has three variants of point interaction:

1) $d(a,b) = d(b,c) = 0$ – there is one point marked in this case by identifiers a,b,c, the distance between them is zero;

2) $d(a,b) = d(b,c) \to d(a,b) \oplus d(b,c) = 0$ – there are two points {b,a = c}, which form two identical distances, creating a cycle, in accordance with the principle of symmetry or commutativity;

3) $d(a,b) \neq d(b,c) \to d(a,b) \oplus d(b,c) \neq 0 \to d(a,b) \oplus d(b,c) = d(a,c)$ – there are two unequal distances $d(a,b) \neq d(b,c)$, which are possible only when the interaction of three points in space a,b,c for determining third distance is realized by vector operation $d(a,b) \oplus d(b,c) = d(a,c)$. In this case, the vector specified in the right side of equality will never be equal to one in the terms of the left side, because $d(a,b) \neq d(b,c)$. Thus, the relationship of any three points in a vector logical space can be reduced to a formal interaction, specified by the equality $d(a,b) \oplus d(b,c) = d(a,c)$, which degenerating regulates the interaction of two and a single point on itself.

Theorem 3. Xor-sum of vector distances in the transitive triangle is equal to zero:

$$d(a,b) \oplus d(b,c) = d(a,c) \to d(a,b) \oplus d(b,c) \oplus d(a,c) = 0 .$$

The proving is based on application of Axiom 1 to the transitive closure expression, obtained in Theorem 2.

Theorem 4. Xor-sum $\overset{n}{\underset{i=1}{\oplus}} d_i = 0$ of vector distances $d_i \in D$ in cycle D, defined by finite quantity of nodes (n), is equal to zero:

$$\begin{aligned} &1)\ d_1 = 0; \\ &2)\ d_1 \oplus d_2 = 0; \\ &3)\ d_1 \oplus d_2 \oplus d_3 = 0; \\ &4)\ d_1 \oplus d_2 \oplus d_3 \oplus d_4 = 0; \\ &5)\ d_1 \oplus d_2 \oplus ... \oplus d_i \oplus ... \oplus d_n = 0. \end{aligned}$$

The proving is based on application Theorems 1-3 to the distances between the points (nodes), forming closed cycles. In the first case the transitive closure distance of point on itself is took place. In the second one it is the distance between two transitive closed points. In third one – between three points of a space. In forth one – between four points. The fifth case generalizes the presence of zero distance in the sum of transitive closures of any points in n-dimensional vector space.

Consequence 1. Xor-sum of any binary codes of the same length is equal to zero, if they form a cycle.

Consequence 2. Metric β of vector logic space is defined by a single equality that forms zero xor-sum of the distances between nonzero and finite quantity of points, closed in a cycle:

$$\beta = \overset{n}{\underset{i=1}{\oplus}} d_i = 0 .$$

Definition 5. Cyber Space is vector logic space, specified by β-metric, where the xor-sum of distances between a finite number of cycle points is equal to zero-vector.

The metric β of vector logical space is focused not on elements of the set, but the relationship, thereby reducing the axioms from three to one formula and extend it to arbitrary complex structures of n-dimensional space.

Example. There are five points of a vector space: (000111, 111000, 101010, 010101, 110011). The closure of these points in the cycle gives the following side-distances of the pentagon: (111111, 010010, 111111, 100110, 110100). Coordinatewise addition of all vectors gives the result: (000000). The practical significance of this fact lies in the possibility of determining any distance of a closed cycle, if (n-1) sides of a figure are known. For a triangle, this means the possibility of determining third side by two known ones. If create a triangle closed logical space, we can gain 66% of data, which generate all distances in the logical space.

III. Process Model For Searching, Recognition And Decision Making

The quality metrics represented in (3), makes it possible to evaluate the proximity between spatial objects, as well as the interaction of vector spaces. As practical example of the usefulness of an integral quality criterion we can consider the firing at a target, which is illustrated by the diagrams of vector interaction described above (see Fig. 1):

1) A shell hit right on a target;

2) The target is hit by a shell of unreasonably large caliber;

3) The shell caliber is not enough to hit a large target;

4) Inefficient and inaccurate shot by large-caliber shell;

5) Shell flew past the target. A process model for the interaction $P(m,A)$ corresponds to the integral quality criterion that evaluate not only hit or miss, but also the efficiency of utilizing the shell caliber.

The analytical form of a generalized process model for choosing the best interaction between the input query m and the system of logic associative relations is presented in the following form:

$$\begin{gathered}
P(m,A)=\min Q_i(m\ \underset{i=1}{\overset{n}{\Delta}}\ A_i)=\vee[(Q_i\ \underset{j=1,n}{\overset{j\neq i}{\wedge}}\ Q_j)\oplus Q_i]=0;\\
Q(m,A)=(Q_1,Q_2,...,Q_i,...,Q_n);\\
A=(A_1,A_2,...,A_i,...,A_n);\\
\Delta=\{and,or,xor,not,slc,nop\};\\
A_i=(A_{i1},A_{i2},...,A_{ij},...,A_{is});\\
A_{ij}=(A_{ij1},A_{ij2},...,A_{ijr},...,A_{msq});\\
m=(m_1,m_2,...,m_r,...,m_q).\\
Q_i=d(m,A_i)\vee\mu(m\in A_i)\vee\mu(A_i\in m),\\
d(m,A_i)=m\oplus A_i;\\
\mu(m\in A_i)=A_i\wedge\overline{m\wedge A_i};\\
\mu(A_i\in m)=m\wedge\overline{m\wedge A_i}.
\end{gathered}\tag{5}$$

Comment: 1) The functionality $P(m,A)$ specifies the analytical model for computational process in the form of statement, minimizing the integral quality criterion. 2) Data structures are presented as nodes-tables of the graph $A=(A_1,A_2,...,A_i,...,A_m)$, which logically interact each other. 3) A graph node is described by the ordered set of the vector-rows of an associative table $A_i=(A_{i1},A_{i2},...,A_{ij},...,A_{is})$ for explicit solutions, where the row $A_{ij}=(A_{ij1},A_{ij2},...,A_{ijr},...,A_{msq})$ is true proposition. Since the functional presented in tabular form has no time-constant input and output variables, this structure differs from sequential von Neumann's machine, defined by finite automata Miles and Moore. Equivalence of all variables in the vector $A_{ij}=(A_{ij1},A_{ij2},...,A_{ijr},...,A_{msq})$ creates conditions for their existence that means the invariance of the problem solving for direct and inverse implication in the space $A_i\in A$. The associative vector A_{ij} is an explicit solution, where each variable is defined in the final, multi-valued and discrete alphabet $A_{ijr}\in\{\alpha_1,\alpha_2,...,\alpha_i,...,\alpha_k\}=\beta$. The interaction $P(m,A)$ between the input vector-query $m=(m_1,m_2,...,m_r,...,m_q)$ and the graph $A=(A_1,A_2,...,A_i,...,A_m)$ generates a set of solutions and makes it possible to choose the best ones by minimum quality criterion:

$$P(m,A)=\min Q_i[m\wedge(A_1\vee A_2\vee...\vee A_i\vee...\vee A_m)].$$

The concrete interaction between the graph nodes generates the functionality $A=(A_1,A_2,...,A_i,...,A_m)$ that can be realized by the following structures: 1) A single associative table that includes all solutions of a logic problem explicitly. The advantage is maximum speed of parallel associative searching for a solution by the table. The disadvantage is the highest hardware complexity of memory allocation for large-scale table. 2) Tree (graph) structure of binary relations between the functional primitives, each of them generates the truth table for small numbers of variables. The advantage is the smallest hardware complexity of problem solving. The disadvantage is minimum speed of sequential associative searching for a solution by tree. 3) The compromise graph structure of logically understandable to the user relations between primitives, each of them generates the truth table for logical strongly connected variables. The advantage is high speed of concurrent associative searching for solutions by minimal number of the graph tables, as well as relatively low hardware complexity of problem solving. The disadvantage is decrease in speed because of sequential logic processing of the graph structure for explicit solutions found in the tables.

Partitioning a single table (associative memory) on k parts allows reducing hardware cost, expressed in components (LUTs – Look Up Table) of programmable logic array [8,9]. Each memory cell is created by 4 LUTs. Taking into account the associative matrix can be represented by a square of side n, the total memory hardware cost Z(n) for storing of data and the time T(n) for analyzing the logic associative graph are functionally dependent on the number n of table partitions or the number of nodes:

$$\begin{gathered}
Z(n)=k\times\frac{1}{4}\times\left(\frac{n}{k}\right)^2+h=\frac{n^2}{4\times k}+h,\ (h=\{n,const\};\\
T(n)=\frac{4\times k}{t_{clk}}+\frac{4}{t_{clk}}=\frac{4}{t_{clk}}(k+1),\ (t_{clk}=const).
\end{gathered}\tag{6}$$

Here h is cost of the general control circuit for the system of associative memory. Consequences of reducing hardware is reduction the speed for processing the memory structure or increasing the time for analyzing the system components. The period of processing a single associative memory is a cycle of 4 clock pulses. The number of partitions k increases proportionally the number of cycles in the case of the worst serial connection of memory. The summand $\frac{4}{t_{clk}}$ determines the time needed to prepare data on the system input, as well as their decoding on the output of computer structure. The functional dependences of the hardware cost and the time for analyzing a graph of associative memory on the number of nodes or partitions are presented in Fig. 4.

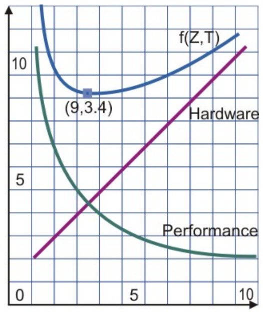


Fig. 4. The dependences of hardware and time on the number of nodes

The generalized function for the efficiency of graph structure on the number of nodes

$$f[Z(n),T(n)]=Z(n)+T(n)=\left(\frac{n^2}{4\times k}+h\right)+\left(\frac{4}{t_{clk}}(k+1)\right)\tag{7}$$

allows determining the optimal partitioning for the total and specified volume of associative memory [6]. In the case shown in Fig. 4, the best partitioning is the minimum of additive function, which is determined by value k, reversing the function derivative to zero:

$$n \times n = 600 \times 600,\ h = 200,\ t_{clk} = 4,\ k = 4.$$

The proposed process model for analyzing the graph of associative tables, as well as the introduced solution quality criteria are the basis of the developing a dedicated multiprocessor architecture focused on the concurrent vector logic operations.

IV. Architecture Of Logic Associative Multiprocessor

To analyze large information volumes of logical data, there are several technologies focused to the practical application: 1. Use a workstation for serial programming, where the cost and time of problem solving are very high. 2. Development of a dedicated concurrent processor based on the PLD. The high concurrency of information processing compensates for the relatively low clock rate in comparison with CPU. Such reprogrammable circuit design is the best solution regarding performance. Disadvantage is lack of flexibility the software methods for solving logic problems and high cost of implementing the system-on-a-chip PLD under large production volumes. 3. The best solution is to leverage advantages CPU, PLD and ASIC concurrently [8,9]. This is due to the flexibility of programming, the possibility of correcting the source code, the minimum command set, and simple circuit designs for hardware multiprocessor implementation, the parallelization of logic procedures by the structure of bit processors. The implementation of a multiprocessor in ASIC allows to obtain the maximum clock rate, the minimum chip cost for large product volumes, and low power consumption. Combining the advantages of the technologies above determines the basic configuration of the LAMP, which has spherical multiprocessor structure (Fig. 5), consisting of 16 vector sequencers. Each sequencer together with the boundary elements is connected with eight contiguous ones. The processor PRUS [9], developed by Dr. Stanley Hyduke (CEO Aldec, USA), is the LAMP prototype.

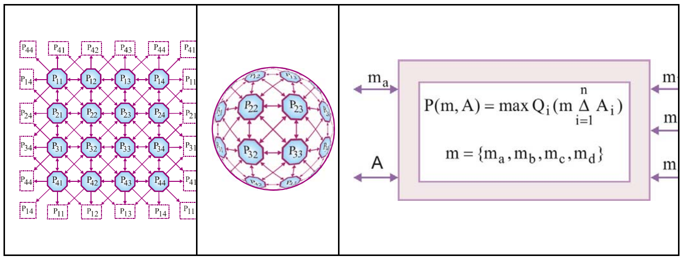


Fig. 5. LAMP macroarchitecture and interface

Entering information in the processor is realized like the classical design flow, except the stage "place and route" that is replaced by the operation of distributing software modules and data among all logical bit processors running concurrently. The compiler provides the placement of data among processors, sets the time of searching for solutions at the output each of them, and also plans transfer the results to another processor. LAMP is an effective processor network, which processes the data and provides the exchange of information between network components when searching for solution. The simple circuit engineering of each processor can effectively process very large arrays with millions bits of information, expending time in hundreds the times less compared with general-purpose processor. Basic cell (vector processor for LAMP) can be synthesized by using 200 gates, which makes it possible to implement network containing 4096 computers in ASIC, using advanced silicone technology. Taking into account that memory costs for data storage are very small, LAMP may be applied for the designing of control systems in the areas of human activity, such as: industry, medicine, information protection, geology, weather forecasting, artificial intelligence, space science. LAMP is of particular interest for digital data processing, pattern recognition and cryptanalysis. If LAMP functioning is considered, its main purpose is obtaining quasi-optimal solution of the integrated problem of search and / or pattern recognition by using infrastructure components focused to the performing vector logical operations:

$$P(m, A) = \min Q_i(m \overset{n}{\underset{i=1}{\Delta}} A_i),\ m = \{m_a, m_b, m_c, m_d\}.$$

System interface, corresponding to this functional, is presented in Fig. 5. All components $\{A, m_a, m_b, m_c, m_d\}$ can be input and output. Bidirectional interface specification is related to the invariance of relation for all variables, vectors, A-matrix, components and infrastructure inputs and / or outputs. Therefore, the structural model of LAMP can be used to solve any problems of direct and inverse implication in discrete logical space, and it emphasizes its difference from the automaton model concept of computer with explicit inputs and outputs. The components or registers $m = (m_a, m_b, m_c, m_d)$ are used for solution in the form of buffer, input and output vectors, as well as for identification of quality estimation for request performance. One of the variants multiprocessor architecture LAMP is a structure shown in Fig. 6. The main its component is a multiprocessor matrix $P = [P_{ij}], \mathrm{card}(4 \times 4)$, containing 16 vector-processors, each of them is designed for performing 5 logic vector operations with data memory contents, described by a table of dimension $A = \mathrm{card}(m \times n)$.

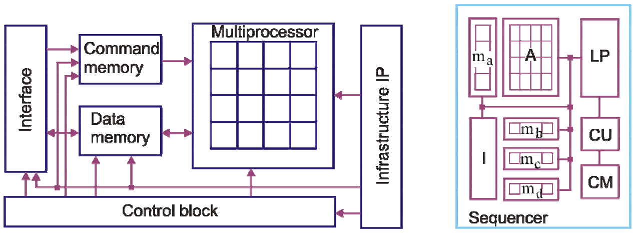


Fig. 6. LAMP architecture and sequencer structure

Interface is used for data exchange and data loading to the appropriate memory commands. The control unit initializes the executing commands of logical data processing and synchronizes the functioning all components of a multiprocessor. Infrastructure IP [1] is designed for servicing all modules, diagnosing faults and repairing functionality of components and device in whole. Elementary logic associative processor or sequencer (see Fig. 6) is a part of the multiprocessor and contains: logical processor (LP), associative (memory) A-matrix for concurrent executing basic operations, block of vectors m, designed for concurrent processing rows and columns of A-matrix, as well as data exchange when computing, direct access memory (CM) for the storing commands of data processing software, automaton (CU) for logic operations execution control, interface (I) for the connecting sequencer and other elements of a multiprocessor. Logical Processor (LP) (Fig. 7) provides the implementation of five operations (and, or, not, xor, s1s - shift left bit crowding), which are the basis for the creating algorithms and procedures of information retrieval and evaluation of solutions. LP module has a multiplexer at the input to select one of five operands, which is passed to the selected logic vector operator. By using a multiplexer (element or), a result is entered in one of four operands, which are selected by appropriate address.

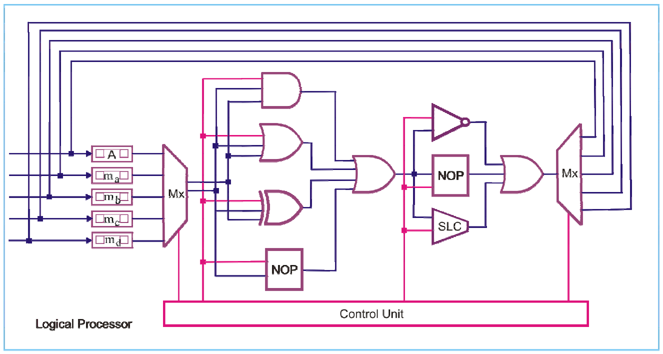


Fig. 7. Structure of logic calculations

Implementation features of the logical processor is use of three binary (and, or, xor) and two unary (not, slc) operations. The last ones can be added to the cycle of processing register data by selecting one of three operations (not, slc, nop – no-operation). To improve the efficiency of logical unit, two elements with empty operation are included. If it is necessary to perform a unary operation only, the selecting nop at the level of binary commands should be done, that almost means the transfer data through a follower to second level of unary operations. All LP operations are register or register-matrix. The last ones are designed for the analyzing vector-rows of a table using input m-vector as a request for exact information retrieval. The following combination of operators and operands are acceptable in a unit for logic calculating:

$$C = \begin{cases} \{m_a, m_b, m_c, m_d\} \Delta A_i; \\ \{m_a, m_b, m_c, m_d\} \Delta \{m_a, m_b, m_c, m_d\}; \\ \{not, nop, slc\} \{m_a, m_b, m_c, m_d, A_i\}. \end{cases}$$

$$\Delta = \{and, or, xor\}.$$

Realization of all vector operations for logic calculating by using a single sequencer in Verilog environment and followed implementation in PLD chip gives the results:

Logic Block Utilization:
Number of 4 input LUTs: 400 out of 9,312 4%
Logic Distribution:
Number of occupied Slices: 200 out of 4,656 4%
Number of Slices only related logic: 200 out of 200 100 %
Total Number of 4 input LUTs: 400 out of 9,312 4%
Number of bonded IOBs: 88 out of 320 29%
Total equivalent gate count for design: 2400

Clock rate of register operations for Xilinx's Virtex 4 is 100 MHz that by order of magnitude higher than similar procedures for a computer with clock rate 1GHz.

V. Infrastructure For Vector Logic Analyzing

Infrastructure is a set of models, methods and data definition languages, data analysis and synthesis tools for solving the functional problems. Model (system model) is a set of interrelated components defined in space and time, which describe the process or phenomenon with specified adequacy, and used for achieving the aim under constraints and metric for evaluating of the solution quality. Here, the constraints are the hardware costs, the time-to-market, which are have to be minimized. Metric for the evaluating solution by using the model is defined by a binary logic vector in the discrete Boolean space. The conceptual computer model is presented by an aggregate of control and operational automata. The system functionality model LAMP uses GALS (Global Asynhronus Local Synchronus) [8] technology for the creating hierarchical digital systems with the local synchronization of individual modules and simultaneous global asynchronous of the entire device.

To detail the structure of the vector processor and sequencer the analytical and structured process models are presented below. They are reduced to the analysis of the A-matrix by columns or rows. The first one is shown in Fig. 8 and it is designed for determining a set of feasible solutions relatively the input query m_b .

$$m_{ai}^{m} = \overline{\vee} [(m_b \bigwedge_{i=1}^{n} A_i) \oplus m_b];$$

$$A_i = (m_b \bigwedge_{i=1}^{n} A_i).$$

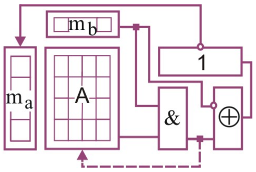


Fig. 8. Searching all feasible solutions

The second structure (Fig. 9) searches for optimal solution on a set of ones, found in the first process model, by analyzing rows. In addition, the second model has a separate application, focused on the finding single-valued and multi-valued solutions, for example, when searching for faults in digital systems-on-chips.

$$m_b^s = (\bigwedge_{\forall m_{ai}=1} A_i) \wedge (\overline{\bigvee_{\forall m_{ai}=0} A_i})$$

$$m_b^m = (\bigvee_{\forall m_{ai}=1} A_i) \wedge (\overline{\bigvee_{\forall m_{ai}=0} A_i})$$

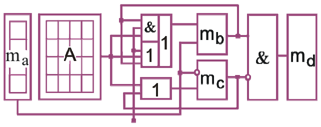


Fig. 9. Structure for searching the optimal solution

All operations are presented by two process models are vector. Process model for analyzing rows (see Fig. 8) generates the vector m_a for identification of feasible $m_{ai} = 1$ or contradictory $m_{ai} = 0$ solutions relatively the input condition m_b for n cycles of processing all m-bit vectors of the table $A = \text{card}(m \times n)$. The quality (validity) of the decision is determined for each interaction between the input vector m_b and the row $A_i \in A$ by the disjunction (devectorization) block. The matrix A can be modified by its intersecting with an input vector on the basis of the operation $A_i = (m_b \overset{n}{\underset{i=1}{\wedge}} A_i)$, if it is necessary to remove from the A-table all insignificant for the solution coordinates and vectors, marked by unit values of the vector m_a. An interesting solution for the problems of diagnosis by analyzing table rows shown in Fig. 9, should be interpreted as follows. After performing the diagnostic experiment the binary output response vector m_a is made, which masks the A-table of faults to detect single or multiple faults. Vectors m_b and m_c are used to accumulate the results of conjunction and disjunction operations. Then the logical subtraction the contents of the second vector m_c from the first register m_b and subsequent saving the result in register m_d is performed. To implement the second equation, which generates a multiple solution, element AND is replaced by the function OR. The circuit has also a variable for the choosing the solution search mode: single or multiple. The process model uses as input condition a vector m_a, which controls the choice of vector operation AND, OR for processing unit $A_i(m_{ai} = 1) \in A$ or zero $A_i(m_{ai} = 0) \in A$ A-table rows. The result of n cycles is accumulation of unit and zero solutions relatively coordinate values of the vector m_a in the registers A_1, A_0, respectively. A priori, the vectors of 1's and 0's are entered in these registers: $A_1 = 1, A_0 = 0$. After processing all n rows of A-table for n cycles the vector conjunction for the contents of register A_1 and the inversion of the register A_0 are performed, which generates the result in the form of the vector m_b, where unit coordinates determine a solution. When analyzing a fault table of digital device the columns identified with the numbers of faults or faulty blocks to be repaired correspond to unit coordinates of the vector m_b. Within the bounds of the Infrastructure IP the optimization repairing problem can be solved by using an universal structure of vector logic analysis. It is necessary to cover all faults found in the cells by minimum number of spare rows and/or columns, such as memory. The technological and mathematical culture of vector logic in this case provides a simple and interesting circuit solution for obtaining a quasioptimal coverage, Fig. 10. The advantages are: 1) The computational complexity of the procedure is $Z = n$ of vector operations, equal to the number of table rows. 2) The minimum hardware costs, which are a table and two vectors m_b, m_a for storing intermediate coverages and the accumulating result in the form of unit coordinates, corresponding to table rows, which contain a quasioptimal coverage. 3) There is no need for the classical splitting the coverage problem for searching a coverage core and a complement. 4) There is no need for complicated procedures for manipulating rows and columns. The disadvantage is obtaining not always optimal coverage that is costs for the efficiency of vector procedure, shown in Fig. 10.

$$\begin{cases} m_b = (m_b \vee A_i); \\ m_{ai} = \overset{n}{\underset{i=1}{\vee}} [(m_b \vee A_i) \wedge \overline{m}_b]. \end{cases}$$

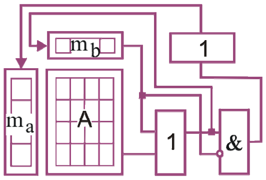
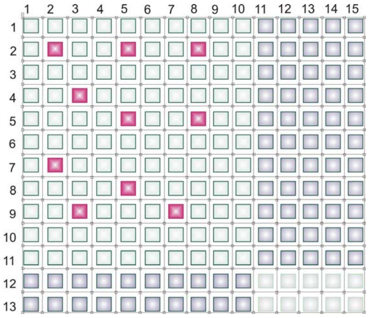


Fig. 10. Process model for searching quasioptimal coverage

There is the devectorization operation, which at the last stage transforms the vector result in a bit m_{ai} of the vector m_a by the function OR $m_{ai} = \vee[(m_b \vee A_i) \wedge \overline{m}_b]$. In general, in the algebra of vector operations the devectorization operation is written in the notation: <binary operation> <vector>: $\vee A_i$, $\wedge m$, $\overline{\wedge}(m \vee A_i)$. The inverse vectorization operation is the concatenation of Boolean variables: $m_a(a, b, c, d, e, f, g, h)$. In the process for coverage searching a priori the vectors $m_b = 0, m_a = 0$ are nulled. The quasioptimal coverage is accumulated in the vector m_a for n cycles by serial shifting. Bits, entered in the register m_a, are formed by the circuit OR, which realizes devectorization by analyzing the input result $[(m_b \vee A_i) \wedge \overline{m}_b]$ on the presence of 1's. The next example is characterized by functionally completeness of the diagnosis cycle, when this information is used to repair faulty memory cells after obtaining the quasioptimal coverage [9]. The dimension of a memory module 13x15 cells does not influence on the computational complexity of obtaining a coverage for ten faulty cells by using spare rows (2) and columns (5) (Fig. 11).



1	.	.	.	.	.	1	.	.	.
.	.	.	1	.	.	.	.	1	.
.	1	.	.	1	.	.	1	.	.
.	.	.	.	.	.	.	.	.	1
.	.	1	.	.	1	.	.	.	.
1	1	1	.	.	.	.	.	.	.
.	.	.	1	.	.	.	.	.	.
.	.	.	.	1	1	.	.	.	.
.	.	.	.	.	.	1	.	.	.
.	.	.	.	.	.	.	1	.	.
.	.	.	.	.	.	.	.	1	1

Fig. 11. Memory module with spare and coverage table

To solve the optimization problem a coverage table for faulty cells is generated (see Fig. 11); it contains rows (spares) for the covering faults:

$$(C_2, C_3, C_5, C_7, C_8, C_2, R_2, R_4, R_5, R_7, R_8, R_9).$$

The columns are faults of cells $(F_{2,2}, F_{2,5}, F_{2,8}, F_{4,3}, F_{5,5}, F_{5,8}, F_{7,2}, F_{8,5}, F_{9,3}, F_{9,7})$ to be repaired. Here the columns match the coordinates of faulty cells, and rows identify the spare components (rows and columns), which can repair the faulty coordinates. The process model (Fig. 9) makes it possible to obtain the optimal solution in the form $m_a = \boxed{1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0}$, which corresponds to the coverage: $R = \{C_2, C_3, C_5, C_7, C_8\}$. It is one of three possible minimum solutions $R = C_2, C_3, C_5, C_7, C_8 \vee C_2, C_3, C_5, C_8, R_9 \vee C_2, C_5, C_8, R_4, R_9$ for a fault detection table. The technological model for embedded diagnosing and repairing memory is shown in Fig. 12.

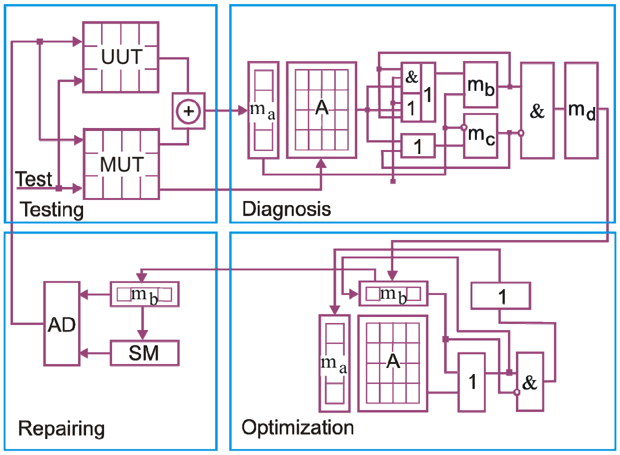


Fig. 12. Model for embedded testing and repairing memory

It includes four components: 1) Testing a module (UUT – Unit Under Test) by using the reference model (MUT – Model Under Test) to generate the output response vector m_a, dimension of which corresponds to the number of test patterns. 2) Fault diagnosis based on analysis of the fault detection table A. 3) Optimization of fault coverage by spares (rows and columns) based on the analysis of the table A. 4) Repairing memory by the readdressing (AD – Address Decoder) faulty rows and columns of the vector m_a by spare components SM – Spare Memory [9].

The process model for embedded servicing is functioned in real time and allows maintaining a digital system-on-a chip without human intervention that is an interesting solution for the critical technologies related to the remote maintenance of a product. The proposed process model for the analyzing associative tables, as well as the imposed quality criteria for logical solutions allows solving the problems for quasi-optimal covering, diagnosing software faults and/or hardware modules. The model of vector calculations provided the basis for the developing dedicated multiprocessor architecture focused to searching, pattern recognition and decision making by using associative tables.

Performance evaluation, Fig. 13, of design solution based on the specialization Sp and standardization St requires use three discrepant parameters: quality Y, time T, hardware cost H:

$$E = F(Y, T, H),$$
$$Y = (1 - P)^{n(1-Q)};$$
$$T = \frac{1}{f} \times S \times d;$$
$$H = 2(H^s \times n).$$

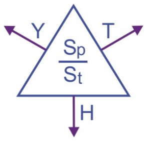


Fig. 13. Performance evaluation for process model

The parameter Y depends on the testability Q of a design, the probability P of existence faulty areas in the chip and the number n of undetected faults.

The time of problem solving is determined by the follows: $S \times d$ structural depth of a circuit multiplied by the average delay of the primitive that enter into the maximum logical path and divided by clock speed f of a device. Hardware costs (see Fig. 6) are a function of the complexity: $H^s, n, H^u, H^m, H^d, H^i$ – sequencer, number of logic processors, control unit, command and data memory, Infrastructure IP for diagnosis and communication interface. In order to simplify the formula of efficiency it can be assumed that the matrix of logic processors is equivalent by the complexity the rest part of the LAMP $(H^s \times n) = H^u + H^m + H^d + H^i$. Analysis of the efficiency E^n the proposed process model for diagnosing and repairing the memory block (see Fig. 12) based on the LAMP with respect to the basic E^b realization on the universal computer uses the evaluation:

$$\eta = \frac{E^b}{E^n} = \frac{T^b}{T^n} + \frac{H^b}{H^n} = \frac{(S \times d)^b}{(S \times d)^n} + \frac{(H^s \times n)^b}{(H^s \times n)^n} =$$
$$= \frac{200}{20} + \frac{1000000 \text{ gates}}{2(800 \times 16) \text{ gates}} = 10 + 40 = 50.$$

Here it is supposed that the clock speed of basic and new products, their quality, as well as delays of primitives are equal and they eliminated from the formula for calculating result. The structural depth of hardware variants is equal to 200 and 20, the number of equivalent gates – 1000000 and 25600. Additive evaluation of the efficiency for using infrastructure to solve the problems of diagnosis and memory repair gives a result equal to 50. Multiplicative evaluation is almost by order of magnitude greater.

VI. Conclusion

Existing software analogs do not provide a purely vector-logical pathes for searching, pattern recognition and decision-making in a discrete information spaces [3,8]. Almost all of them use a universal command system of modern expensive CPU with math coprocessor. On the other hand the hardware dedicated tools for logical analysis, which can be considered as prototypes [1,3], typically focused on bitwise or nonvector information processing.

To eliminate the disadvantages of software analogs and hardware prototypes it is proposed the new approach for vector logic processing the associative data with complete exclusion of arithmetic operations, which influence on the performance and hardware complexity. It was successfully implemented on the basis of modern microelectronic devices in the form of multiprocessor digital system-on-a-chip.

Actual implementation of the approach is based on the proposal of infrastructure, which includes the following components: 1. Process models for the analyzing associative tables based on the use of vector logical operations for searching, pattern recognition, decision making in the vector discrete Boolean space. Models are focused on high-performance concurrent vector logical analysis of information and calculating of solution quality criteria on the basis of proposed beta-metric of cyber space. 2. A multiprocessor architecture for concurrent solving associative logic problems by using a minimal set of vector logical operations and total exclusion of arithmetic instructions. It provides high performance, minimal cost and low power consumption of LAMP, implemented in a chip of programmable logic. 3. Novel vector logical process model for embedded diagnosing digital systems-on-chips and searching for quasioptimal coverage based on the logic associative multiprocessor, parallel operations for computing processes and calculating quality criteria.

The veracity and practical significance of the obtained results are confirmed by the creation of multiprocessor infrastructure for diagnosing and repairing memory components of digital system-on-a-chip, the theoretical proof of the metric for vector logical space and the quality criteria for estimating solutions.

Further research are focused on the developing a prototype of logic associative multiprocessor in order to solve the topical problems of searching, pattern recognition and decision making by using the proposed infrastructure of vector logical analysis.

References

[1] Y. Zorian, “Test Strategies for System-in-Package” *Plenary Paper of IEEE East-West Design & Test Symposium (EWDTS’08)*, 2008.

[2] L. Smith, “3D Packaging Applications, Requirements, Infrastructure and Technologies”, *Fourth Annual International Wafer-Level Packaging Conference,* September, 2007.

[3] M.F. Bondarenko, “Brain-Like Computers”, Radielectronics & Informatics, No2, pp. 89–105, 2004.

[4] M.F. Bondarenko, “Predicate Algebra”, Bionica intellecta, No1, pp. 15 – 26, 2004.

[5] A. Acritas, “Fundamental of computer algebra with applications”, M.: Mir, 1994, 544 p.

[6] A.V. Attetkov, “Optimization Methods”, *Bauman Moscow State Technical University,* 2003, 440 p.

[7] J. Bergeron, “Writing Testbenches Using SystemVerilog”, *Springer Science and Business Media, Inc*., 2006, 414 p.

[8] D. Densmore, “A Platform–Based taxonomy for ESL Design”, *Design & Test of Computers*, pp. 359–373, 2006.

[9] V.I. Hahanov, “Digital System-on-Chip Design and Test”, *Kharkov:Novoye Slovo*, 2009, 484 p.

[10] V.I. Hahanov, “Digital System-on-Chip Design and Verification”, *Kharkov:Novoye Slovo*, 2010, 528 p.

[11] A.A. Cohen, “Addressing architecture for Brain-like Massively Parallel Computers”, *Euromicro Symposium on Digital System Design (DSD'04)*, pp. 594-597, 2004.

[12] O.P. Kuznetsov, “Simulating High-Speed Intelligent Processes of Ordinary Thinking”, *Intelligent Systems*,Vol.2, Issue. 1-4, 1997.

A CLOSER LOOK AT MICROPROCESSORS THAT HAVE SHAPED THE DIGITAL WORLD

Christopher U. Ngene[1] *Student member IEEE*, Manish Kumar Mishra[2]

[1]*Computer Engineering Faculty, Kharkov National University of Radioelectronics, Kharkov, Ukraine*
unchris.ua@gmail.com.
[2]*Department of Computer Engineering University of Maiduguri, Nigeria; mishrasoft1@gmail.com*

Abstract – **If you have been following the development in the microprocessor world you would attest to the fact that things have dramatically changed since the introduction of the first world acclaimed microprocessor Intel 4004 in 1971. What were the changes that have been made to these processors that have actually improved our lots, especially how we perceive the world around us and improve our productivity at work? In this study, we investigate different general purpose processors with a view to enlightening consumers and enthusiasts alike, determine which of the myriads of processors will be most appropriate for their tasks and the choice of which makes more economic sense. We have been able to explore in relative detail that processor speed is not the only determinant of processor performance but of most significant is the architecture. This study reveals that new technology is not the only factor that determines whether a new processor is actually new, but most importantly marketing considerations have been the driving force.**

I. INTRODUCTION

Since the introduction of the first commercial integrated circuit in 1961 and the introduction of the first microprocessor in 1971, the semiconductor industry has experienced a healthy growth. Anyone involved in electronic design or electronic design automation (EDA), marketing or analysis of electronic devices knows that things are becoming evermore complex as the years go by and microprocessors are no exception to this rule. By comparison today's microprocessors are more complex than the 1960's and '70s mainframe's central processing units. Most importantly these processors outperform those mainframe CPUs and are cheaper and affordable. Architectures have also evolved to the extent that we no longer talk about CISC or RISC but a combination of both. Other architectures that enhance performance like EPIC (Explicitly Parallel Instruction computing) based on VLIW in conjunction with pipelining, super-scaling and hyper-threading have boosted performance to unimaginable level.

The fate of both Intel and Microsoft was dramatically changed in 1981 when IBM introduced the IBM PC, which was based on a 4.77MHz Intel 8088 16-bit processor running the Microsoft Disk Operating System (MS-DOS) 1.0. Was there any one chip that propelled Intel into the Fortune 500? Intel says there was: the 8088 [6]. Since that fateful decision was made, computers became affordable and found their ways into our homes. The focus of this paper is on Intel and AMD processors. Even at that we have discussed selected number of processors that have made their marks in the industry. The obvious reason for this selective approach is because there are myriads of processors in the inventory of these vendors which the limited space at our disposal will not be enough to accommodate.

Available publications on microprocessors have focused mainly on the historical development, specific processor reviews and benchmarking [1][2][18]. We agree that these are necessary. But numerous users of these processors may not quite appreciate some of the technical jargons employed in some of these publications. With respect to the above this paper has taken a wider perspective to give our readers a holistic view. This has dictated the approach we have taken and have encapsulated all vital information regarding processor within the pages of this paper. In this paper we have given a brief historical background of microprocessors and how the Moore's law has been holding out as a result of innovations in chip fabrication enabling smaller and smaller feature sizes. The rest of this paper is presented as follows: section 2 presents some historical backgrounds on the development of transistors, integrated circuits and subsequently microprocessors. In this section we have also looked at the basic processor architectures. Section 3 presents the different types of processors and microprocessor feature trends. Special purpose processors – microcontrollers, graphic and digital signal processors were discussed, but a detailed description of these processors have been left out for future work. Section 4 presents methods of chip fabrication, foundries, process yield and process technologies and an evaluation of the relationships between die sizes, lithography and transistor count per die.

2. BACKGROUND INFORMATION

Before we proceed further a little background would suffice in order to appreciate where we are now by knowing where we came from. The development of computer systems is closely tied with processors and subsequently

Manuscript received November 3, 2009.

Christopher U. Ngene is with the Kharkov National Uuniversity of Radioelectronics, *Computer Engineering Faculty*, Lenin Prosp., 14, Kharkov, 61166, Ukraine (corresponding author, phone: +38057-7021326), unchris.ua@gmail.com.
Manish Kumar Mishra is with Department of Computer Engineering University of Maiduguri, Nigeria; mishrasoft1@gmail.com

microprocessors. Processor (Central Processing Unit - CPU) in conjunction with the memory is the brain of the computer. Data processing (arithmetic and logic operations take place in the CPU). Early computers which were mainly mainframes have very large CPUs. Early CPUs were implemented as discrete components and numerous small integrated circuits (ICs) on one or more circuit boards. Microprocessors, on the other hand, are CPUs manufactured on a very small number of ICs; usually just one. The overall smaller CPU size as a result of being implemented on a single die means faster switching time because of physical factors like decreased gate parasitic capacitance. Prior to the advent of machines that resemble today's CPUs, computers such as the ENIAC had to be physically rewired in order to perform different tasks. These machines are often referred to as "fixed-program computers," since they had to be physically reconfigured in order to run a different program. Since the term "CPU" is generally defined as a software (computer program) execution device, the earliest devices that could rightly be called CPUs came with the advent of the stored-program computer. CPU deals with discrete states and thus employs switching elements for change of states. Before the discovery of transistors, electrical relays and vacuum tubes (thermionic valves) were commonly used as switching elements. The electromechanical relays and vacuum tubes have the problems of contact bounce and heat respectively. They generally have a slow switching capability. They are considered to be very unreliable for the above reasons. Tube computers like EDVAC are generally faster than electromechanical computer (Harvard Mark I) but are less reliable. EDVAC tended to average eight hours between failures, whereas relay computers (Harvard Mark I) failed rarely.

2.1 Evolution and Direction of Development

Let us start by examining the different switching elements and subsequent technologies that characterises the generations of microprocessor. Vacuum tubes and electromechanical relays were used in the first generation Processors. One other important drawback of these early processors was that programming was done using machine language. The discovery of transistor by three bell laboratory scientists - J. Bardeen, H. W. Brattain, and W. Shockley launched the Second generation processors. Transistors revolutionised electronics in general and computers in particular. Transistors were much smaller than vacuum tubes, consumed less energy, faster switching and more reliable. Programming of the CPU was done in assembly languages (symbolic languages) and followed by high level languages such as FORTRAN and COBOL. Standardization trend generally began in the era of discrete transistor CPUs. With this improvement more complex and reliable CPUs were built onto one or several printed circuit boards containing individual components. After the deployment of transistors as a switching element, CPUs were still large and occupies several circuit boards. The needs to reduce the size of components were primary preoccupation of engineers and scientists. A method of manufacturing many transistors in a compact space was developed. This method is known as Integrated Circuit (IC). An IC is a complete electronic circuit on a small chip of silicon. Beginning in 1965 ICs began to replace transistors in CPUs. In 1959, Jack Kilby and Robert Noyce independently invented a means of fabricating multiple transistors on a single slab of semiconductor material.

2.2 Scale of Integration

Dimensions on an IC are measured in units of micrometers, with one micrometer (1 μm) being one millionth of a meter. To serve as a reference point, a human hair is roughly 100 μm in diameter. Each year, researchers and engineers have been finding new ways to steadily reduce these feature sizes to pack more transistors into the same silicon area. There are different levels of integration. This has to do with the number of digital components that are placed on a single chip. The early ICs contained only one building block (logic gates) such as AND gates etc. CPUs based on this sort of IC are known as Small Scale Integration (SSI) devices. Such ICs contained tens of transistors. To build an entire CPU out of SSI ICs required thousands of individual chips, but still consumed much less space and power than earlier discrete transistor designs.

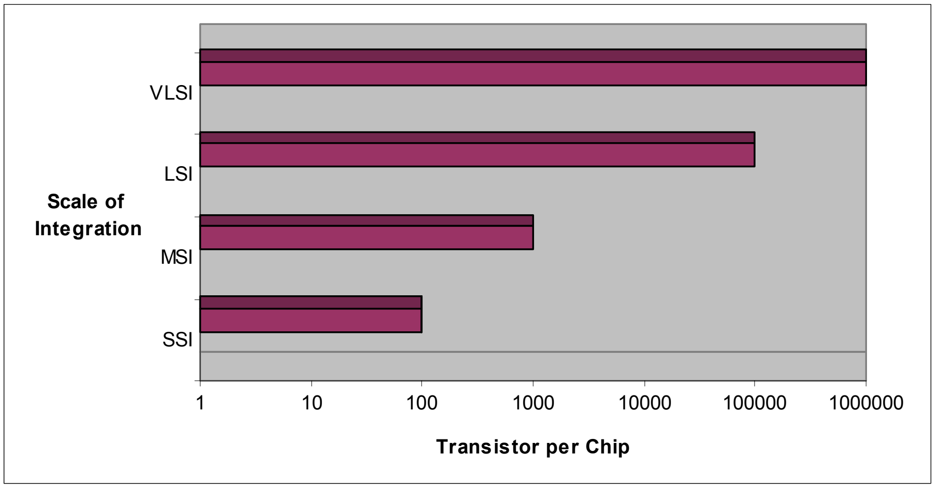


Fig. 1 Scale of Integration

With the advance in microelectronic technology more and more transistors are placed on a single chip. This resulted in the reduction of the number of ICs required for a complete CPU. Other levels of integration include Medium Scale Integration (MSI), Large Scale Integration (LSI) and Very Large Scale Integration (VLSI). The increase in the scale of integration results in increase in transistor counts to hundreds, tens of thousands and tens of millions, see figure 1. Lithography (method of tiny writing) is a technology that brought the possibility of squeezing 4 billion transistors on chip whose size is not more than a postage stamp [1].

The introduction of microprocessors marked the beginning of the fourth generation CPUs. MP is a complete CPU on a single chip of silicon wafer. This is actually an extension of the third generation CPU technology. If the early CPUs were designed for specific purposes (applications) the MP is actually a general purpose CPU on a single chip. The first commercially available MP was the Intel 4004, produced in 1971. It contained 2300 PMOS transistors [4]. **Moore's Law:** As the number of transistors on a single chip kept on increasing, Fairchild Semiconductor's director of Research & Development and later co-founder of Intel Gordon Moore observed in his paper titled "Cramming More Components onto Integrated Circuits" in1965 that the density of elements in ICs was doubling every 18 months, and predicted that the trend would continue for the next ten years. See section 3 for more details. With certain amendments, this came to be known as Moore's Law. By 1971 when the first MP was produced Moore's law was found to be an accurate predictor of the growth of the number of transistors on a single chip. Lithography is the reason why Moore's Law endures after 44 years. First demonstrated in September of 2007, the 32nm SRAM test chip is a testament to the health of not only the 32nm process, but also of the health of Moore's law. More information on photolithography can be found in section 4 of this paper.

2.3. CISC and RISC Architecture

CISC (Complex Instruction Set Computer) and RISC (Reduced Instruction Set Computer) are dominant processor architecture paradigms. Computers of the two types are differentiated by the nature of the data processing instruction sets interpreted by their CPUs. They both have advantages and drawbacks, which are detailed below. The IBM 360 system, created in 1964, was probably the first modern processor system, which initiated the idea of computer architecture in computer science and adopted micro-coded control. Micro-coded control facilitated the use of complex instruction sets and provided flexibility, thus appeared CISC. CISC was primarily motivated by a desire to reduce the "semantic gap" between the machine language of the processor and the high-level languages in which people were programming, the theory was that such a processor would have to execute fewer instructions and thus would have better performance [7]. To improve performance, CISC systems try to reduce the number of instructions programs must call. To do this, they have large sets of microcode instructions that cover a broad range of tasks. A single microcode instruction, in turn, when translated in the CPU, may become several tasks the processor performs. As a consequence, instructions are of variable length and often require more than one clock cycle to complete. However, according to the 20-80 rules, 20% of the available instructions are likely to be used 80% of the time, with some instructions only used very rarely. Some of these instructions are very complex, so creating them in silicon is a very arduous task. Instead, the processor designer uses microcode instead of hardwired control unit [8]. CISC processors are characterised by few number of registers, large instruction set (some simple and some complex), variable length instruction, instructions generally take more than 1 clock cycle to execute, microcode control etc. The processors that employ CISC architecture include DEC VAX, Motolora 68K and 680x0, x86 families of processors, Pentium MMX, to Pentium III.

In 70's, John Cocke at IBM's T.J Watson Research Centre provided the fundamental concepts of RISC, the idea came from the IBM 801 minicomputer built in 1971 which is used as a fast controller in a very large telephone switching system. This chip contained many traits a later RISC chip should have: few instructions, fix sized instructions in a fixed format, execution on a single cycle of a processor and a Load / Store architecture. These ideas were further refined and articulated by a group at University of

California Berkeley led by David Patterson, who coined the term "RISC" [9]. They realized that RISC promised higher performance, less cost and faster design time. RISC systems, on the other hand, seek to improve performance by reducing the number of clock cycles required to perform tasks. They have small sets of simplified instructions, doing away with microcode altogether in most cases. While this means that tasks require more instructions, instructions are all of the same length and usually require only one clock cycle to complete. Because of this, RISC systems are capable of processing instructions in parallel in a process called pipelining. RISC processors are characterised by large number of registers, fixed length instruction, pipeline, instructions generally take less than 1 clock cycle to execute, hardwired control, complexity pushed to the compiler etc. At the theoretical level a RISC chip has to execute more instructions to complete a given task, but it does this so fast that it ends up being faster than an equivalent CISC chip. This is because they only have to do a few simple tasks, so they can concentrate on doing them at really high speed. The early RISC processors are RISC I and RISC II from the University of California at Berkeley and the MIPS from Stanford University. Other typical RISC system includes HP PA-RISC, IBM RT-PC, IBM RS6000, Intel's i860 and i960, MIPS R2000 (and so on), Motorola's 88K, Motolora/IBM's PowerPc, and Sun's SPARC etc.

Currently, the difference between RISC and CISC chips is getting smaller and smaller. RISC and CISC architectures are becoming more and more alike. Many of today's RISC chips support just as many instructions as yesterday's CISC chips. The PowerPC 601, for example, supports more instructions than the Pentium. Yet the 601 is considered a RISC chip, while the Pentium is definitely CISC. So, the difference between RISC and CISC is no longer one of instruction sets, but of the whole chip architecture and system. The designations RISC and CISC are no longer meaningful in the original sense. What counts in a real world, are always how fast a chip can execute the instructions it is given and how well it runs existing software [10]. The biggest threat for CISC and RISC might not be each other, but a new technology called EPIC. EPIC stands for Explicitly Parallel Instruction Computing. Like the word parallel already says EPIC can do many instruction executions in parallel to one another. EPIC was created by Intel and is in a way a combination of both CISC and RISC. This will in theory allow the processing of Windows-based as well as UNIX-based applications by the same CPU.

III. Types of MP

The discussion of MP will not be complete without looking at the different types and the direction of development. As designers found more and more applications for MP they pressured MP manufacturers to develop devices with architectures and features optimized for doing certain types of tasks. In response to the expressed needs, MP have evolved in two major directions during the last 15 years- Special and General purpose processors [4].

3.1 Special Purpose Processors

These types of processors are designed to perform specific functions like controlling a robot arm, controlling a production process. Some of them are optimized to handle graphics and multimedia. The processor used in play station is a good example of a processor optimized to handle streaming graphic information.

Embedded Controllers (Microcontrollers): Microcontrollers are frequently used in automatically controlled products and devices, such as telephones, clocks, automobile engine control systems, office and domestic machines, appliances, power tools, and toys. In contrast to general-purpose CPUs, microcontrollers may not implement an external address or data bus, because they integrate RAM and non-volatile memory (EEPROM) on the same chip as the CPU. Because they need fewer pins, the chip can be placed in a much smaller, cheaper package. It emphasises high integration, low power consumption, self-sufficiency and cost-effectiveness. Microcontrollers often operate at very low speed compared to modern day microprocessors, but this is adequate for typical applications [18].

An embedded system may have minimal requirements for memory and program length. Input and output devices may be discrete switches, relays, or solenoids. An embedded controller may lack any human-readable interface devices at all. For example, embedded systems usually don't have keyboards, screens, disks, printers, or other recognizable I/O devices of a personal computer. The number of microcontrollers that is in use today is far greater than the number of general purpose processors. For instance in a typical home there may be say two desktop systems and a laptop, this implies that there is only 3 general purpose processors. In such homes you may have about 4 handsets, a refrigerator, a deep freezer, a microwave oven, TV set, DVD player and two toys. This translates to about 10 embedded processors which is far greater than the number of general purpose processors in the same home. It is important to note that our desktop PCs also contain microcontrollers that handle input/output operations. The major characteristic of embedded processors is Low power. Consequently low power processors have also found their ways in cell phones and GPS receivers. Examples of such processors are low power Z-80 and 80386. Top Microcontroller manufacturers are; Texas Instruments, Atmel, National Semiconductor, Silicon Laboratories, NXP Semiconductors, NEC, Microchip etc.

Graphic Processors: A Graphics Processing Unit (GPU) is a processor attached to a graphics card dedicated to calculating floating point operations and the like. It is a specialized processor that offloads 3D graphics rendering from the general purpose microprocessor. It is used in embedded systems, mobile phones, personal computers, workstations, and game consoles. Modern GPUs are very efficient at manipulating computer graphics, and their highly parallel structure makes them more effective than general-purpose CPUs for a range of complex algorithms. In a personal computer, a GPU can be present on a video card, or it can be on the motherboard. More than 90% of new desktop and notebook computers have integrated GPUs, which are usually far less powerful than those on a video card [18]. CPUs compute, while GPUs let you experience. Both are important, but most computers today are shipped with an underpowered Graphics Processing Unit. Whether you're editing photos, watching videos, playing a game, or just using the latest operating systems, a powerful GPU will help your desktop or notebook PC run smoothly with jaw-dropping visuals. As the processing power of GPUs has increased, so has their demand for electrical power. High performances GPUs often consume more energy than current CPUs.

Many companies have produced GPUs under a number of brand names. In 2008, Intel, NVIDIA and AMD/ATI were the market share leaders, with 49.4%, 27.8% and 20.6% market share respectively. However, those numbers include Intel's very low-cost, less powerful integrated graphics solutions as GPUs. Not counting those numbers, NVIDIA and AMD control nearly 100% of the market. VIA Technologies/S3 Graphics and Matrox also produce GPUs [18]. GPUs can be Dedicated or Integrated. A dedicated graphics cards have RAM that is dedicated to the card's use. Integrated graphics solutions, or shared graphics solutions are graphics processors that utilize a portion of a computer's system RAM rather than dedicated graphics memory. Computers with integrated graphics account for 90% of all PC shipments. These solutions are cheaper to implement than dedicated graphics solutions, but are less capable. Historically, integrated solutions were often considered unfit to play 3D games or run graphically intensive programs such as Adobe Flash. It is entertainment applications that are driving CPU performance now and for games and sound and graphics processing, you need floating point performance. Nvidia is a leader in the design and manufacture of graphic processors.

Digital Signal Processors (DSP): Digital Signal Processing is carried out by mathematical operations. In comparison, word processing and similar programs merely rearrange stored data. This means that computers designed for business and other general applications are not optimized for algorithms such as digital filtering and Fourier analysis.

Digital Signal Processors are microprocessors specifically designed to handle Digital Signal Processing tasks. Digital signal processing algorithms typically require a large number of mathematical operations to be performed quickly on a set of data. Signals are converted from analog to digital, manipulated digitally, and then converted again to analog form. However, the last forty years have shown that computers are extremely capable in two broad areas, (1) data manipulation, such as word processing and database management, and (2) mathematical calculation, used in science, engineering, and Digital Signal Processing. All microprocessors can perform both tasks; however, it is difficult (expensive) to make a device that is optimized for both. Since DSPs are optimised for high speed arithmetic operations its instruction set is much smaller than that of a desktop microprocessor—perhaps no more than 80 instructions. This means that the DSP needs only a slimmed-down instruction-decode unit and fewer internal execution units. Moreover, any execution units that are present are geared toward high-performance arithmetic operations. DSP uses a Harvard architecture (maintaining completely physically separate memory spaces for data and instructions) so the chip's fetching and execution of program code doesn't interfere with its data processing operations. General purpose processors use Von Neumann architecture, whereby data and instructions are stored in the same memory but of course in different locations. Digital signal processors have far fewer transistors than a CPU, thus they consume less power, which makes them ideal for battery-powered products. Their simplicity also makes them inexpensive to manufacture, thus they're well suited for cost-sensitive applications. Texas Instrument is the top supplier of chips for cellular handsets, as well as the number one producer of digital signal processors and analog semiconductors [11].

3.2 General Purpose Processors

These are the type of processors that powers laptops, desktops, workstations and servers. They are named general purpose because they accomplish a wide variety of tasks including those tasks specialized processors are designed for. They are characterized by large amount of registers and addressable memory locations. The tables below (table 1 to table 5) [12] show the major characteristics of these processors and the various improvements accomplished over the years by designers/manufacturers. There are so many companies manufacturing microprocessors but for the purposes of this paper we will concentrate on those processors manufactured by Intel and Advanced Micro Devices (AMD) and little mention of Motorola. There is no doubt that Intel and AMD have monopolized the microprocessor industry and have engaged in strict competition over who will produce the best performing processors. We shall take a look shortly on how they have fared all these years. The third column on the table shows the size of the general purpose registers (GPR) and floating point registers (FPR or FP unit). The GPR usually determines the number of bits the ALU can handle at a time. Thus when we say 8-bit or 32-bit processor we are saying that the integer unit or ALU and GPR are 8 or 32 bits wide. The floating point unit (FPU) is also known as math processor or coprocessor. It has its own instruction set separate from the main processor and it is optimized to handle arithmetic operation on very small and very large numbers, that is real numbers.

The tables Table 1 through to Table 5 were organized in accordance with the various Intel families of processors to aid readability and understandability. The first column of these tables gives the type of processor and year it was first introduced. In this paper we have considered selected types of processors for a given year because there are so many types of processors that were produced each year. Some of the new ones are not actually better than their elder brothers. Marketing rather than technical considerations becomes ever more important, the once-clear distinctions between different products become blurred. Consequently we have chosen those processors that have actually showcased new technology and can be said to be distinct from their older cousins. These companies use a naming convention that is aimed at deceiving the buyers who always think that newer processors are better than the older ones without considering their needs. For instance Intel original Pentium III was actually Pentium II SSE with Streaming SIMD. Sometime the trick is in the number (clock speed).The Celeron 266 actually ran at 266MHz, but it performed like a Pentium MMX at 200MHz. AMD's response to this was to revive the old and rather unpopular PR rating. The AMD Athlon XP family were introduced at 1333, 1400, 1466, and 1533MHz, but are sold as the 1500+, 1600+, 1700+ and 1800+ [19]. In some cases the manufacturers deliberately remark their chips to lower speeds even as the chip can operate at a higher speed. This was done for marketing purposes only, as lower speed ones were cheaper and sold at cheaper prices. When the buyers got hint of this, over-clocking of processors was born.

TABLE 1
SELECTED CPUS: X486 AND EARLIER PROCESSORS

Processors and Date introduced	Clock Speed Internal/External – FSB (MHz)	Register Width (Bits)	External Data Bus/ Address Bus Width (Bits)	Transistor Count per Die	On-Die Caches	Die Size/Technology (mm^2/µm)
4004 – Nov 1971	0..1 / 0.1	4	4/640 Bytes	2,300	None	24/10 PMOS
8080 – Apr 1974	2 / 2	8	8/8 – 256B	6,000	None	20/ 6µm NMOS
Z-80 – July 1976	2.5-12 /2.5-12	8	8/16 – 64KB	6,000	None	
8085 –Mar 1976	5/5	8	8/16 – 64KB	6,500	None	/3 NMOS
8086 – June 1978	10, 8, 4.77 / same	16	16/20 – 1MB	29,000	None	33/3.2µm
MC68000 – Sep 1979	50	16	/24 - 16MB	68,000		

80286 – Feb 1982	12, 10, 6 / same	16	32/24 – 16MB	134,000	None	1.5μm
80386 DX – Oct 1985	16, 20, 25, 33/same	32	32 / 32 – 4GB	275,000	None	1.0 -1.5
80486 DX – Apr ‘89	25, 3, 50, /20-50 75 - 100	32, 80FPU	32/32 – 4GB	1,200,000	L1-8KB Unified	/1, 0.8 (50 MHz) 345/0.6
80486DX - Mar’94				1,600,000		

TABLE 2
PENTIUM PROCESSOR FAMILY

Processors and Date introduced	Clock Speed Internal/External Bus - FSB (MHz)	Register Width (Bits)	External Data Bus/ Address Bus Width (Bits)	Transistor Count per Die (Millions)	On-Die Caches	Die Size/Process Tech. (mm^2/nm)
Pentium Processor family						
Pentium – Mar 1993 – Mar-94 (Desktop)	60- 200/50-66	32, 80FPU	64/32 – 4GB	3.1 3.3	L1-2x8KB	294/800 148/600, 350
Pentium Pro- Nov 1995 – Jan-96 (Desktop)	150-200/60-66	32, 80FPU	64/32 – 4GB	5.5	L1- 2x8KB L2- 256,512KB	202/600, 196/350
Pentium with MMX–Oct-96 – Sep-97 (Desktop/Notebook)	166 -300/60 – 66	32, 80FPU, 64MMX	64/32 – 4GB	4.5	L1-16KB	140/350, 250
Pentium Extreme EditionApr-05 – Jan-06 (840,955,965)	3,2GHz/800MHz 3.2, 3.46, 3.73 /1.066GHz	32	64/36 – 64GB 64/36 – 64GB	230 376	L2 – 2 MB L2 – 4 MB	203/90 162/65
Pentium Dual Core –Jan-07 Mobile T2130, T2060,T2080 Jul-07 Desktop E2140, E2160	1.6, 1.73/ 533 1.6, 1.8/800MHz	32	64/36 – 64GB Same	176 Same	L2 – 1MB Same	90.3/65 Same
Pentium II –Jan 1997 –Aug-98 (Desktop) (Server)	233-300/66 333,350, 450 /66, 100	32, 80FPU, 64MMX	64/36 – 64GB same	7.5 same	L1- 2x16KB L2-256KB or 512KB L2 – 512KB	202/350 104/250
Pentium II Xeon –Jun-98 (Servers)	400-450/100	32	64/36	7.5	L1- 512KB L2- 2MB	/250
Pentium III Mobile Celeron Jan-99 – Sep-02 (Mobile)	266,333, 366-466/ 450-850/100 1.6-2.5GHz /133, 400	32	64/36	18.9	L2 – 128KB L2 – 128KB L2 – 256KB	/250 104/180 /130
Pentium III Xeon Oct-99-May-00 (High End)	0.6 -1GHz/ 100, 133 500-550	32	64/36-64GB	28 9.5	L1- 256, 512 L1- 2MB same	104/180 /250
Pentium III with Speedstep tech. Jan-00 (Mobile PC)	0.750 – 1GHz /100MHz	32	64/36	28	L2 – 256KB Advanced xfer cache	/180
Pentium III Mobile Jul-01 – Sep-02 (Mobile PC) Server Jan-02	1 – 1.33GHz /133MHz 1.4GHz/133	32	64/36	44 same	L2 – 512KB Same with advanced xfer cache	/130 same
Pentium III Celeron Mar-04 – Mar-06 (Notebook)_	0.9 – 1.7Ghz/400 1.46- 2GHz /400		64/36-64GB	140 151	L2– 512K, 1MB L2- 1MB	/130, 90 /65
Pentium M						
Pentium M Mar-03 (Mini-Notebook)	0.9 – 1.7/400	32	/32 – 4GB	77M	L2 – 1MB	83/0.13
Pentium M May-04- Jan-05 (Full size Mobile PC)	1 -2.13/400, 533	32	/32 – 4GB	144M	L2 – 2MB	87/90nm
Pentium 4						
Pentium 4 Nov-00 Aug-01 – Mar-02 With HTT Nov-02 Feb-Jun-04 Feb-05 (Desktop/Gaming)	1.3 – 2G /400 2 - 2.8 /400-533 3.06 – 3.40 /533 – 800 2.8 – 3.8/800 3 – 3.6/800, 1.066	32 32 64 64	64/32 – 4GB Same same	42 55 55 125 169	L2 – 256KB L2 – 512KB L2 – 512KB L2 – 1MB L2 – 2MB	217/180 131/130 131/130 112/90 135/90
Pentium 4 Jun-02 – Sep-03 (Appied comp/Mobile)	1.7 – 3.2G /400, 533	32		55	L2 – 512KB	131/130
Pentium 4 with HTT 518, 532(Mobile)Jun-04 –Sep-04	2.8- 3.46 G /533	32		125	L2 – 1MB	112/90
Pentium 4 Extreme Edition HTT Nov-03Nov-04(Gaming)	3.2 – 3.46/800, 1.066	64	64/36-64GB	178 178	L2 – 512KB L3 – 2MB	/130
Xeon Processors						
Xeon MP Sep-08	2.13 – 2.66 /1.066	64	64/40 – 1TB	1.9B	L2 – 8MB	503/45

E7420, 30, 40 (Server)				Same same	12MB 16M	503/45 same
Pentium D						
Quad Core Xeon Jan-08 X3320 (Server)	2.50 – 2.83 /1333		64/36- 64GB	456 820	L2 – 6 MB 12MB	164/45 214/45
Pentium D May-05 (Desktop PC)	2.8 -3.2/800	64	64/36-64GB	230M	L2 -1MB	206/90nm
Pentium D Jan-06(Desktop)	2.8-3.733/800, 1.066	64	64/36-64GB	376M	L2 – 2MB	280/65nm

The second column of these tables indicates the processor and the front side bus (FSB) speed. For some processors like the Core i7 family the second column indicates the processor speed and the QPI (Quick Path Interconnect). Early processors have very low computational power. The first processor was never used in a personal computer; it was designed for use in a calculator. Most of these processors go for a microcontroller. Interestingly the Z-80 was the most popular processor of its time. It was developed by a group of ex-Intel engineers who set out to improve on the 8080 but still maintain compatibility with it. Zilog pushed it to faster and faster clock speeds over the years. It was more than a match for the clumsy first-generation 16-bit CPUs like the Intel 8086 and can still do useful work. Incredibly, versions of the 30-year-old 8-bit Z-80 are still in production. Early microprocessors did not incorporate FPUs, but real number crunching is done using high level or low level programs written for the main processor. This reduces the speed of processing and it explains why these processors perform poorly on graphic applications. The 8086 incorporates 8087 math processor which is used in parallel with the main processor. Intel 386 DX a high-performance chip of choice for a long, long time. The DX-33 was never a volume seller because of the AMD386DX-33's entry into the market. This was a mixture of AMD's own design work and the Intel technologies covered under the existing ten year cross-license agreement. In 1976 and 1982 Intel signed a technology sharing contract with AMD which allowed AMD to use Intel intellectual property to make 80286 processors. But AMD wanted to make the 386 Intel objected and the legal battle began. AMD declared that they will use the parts in their position to make the 386. And in late 1991 AMD won the case and the AMD386DX-33 was born. This also marked the beginning of Clone wars. This generally helped to bring down the cost of these class and subsequent classes of processors at the time. It was the buyers that actually benefitted immensely from this situation. AMD made and sold a moderate number of its own 386DX-33, but soon moved on to that all-time great chip, the 386DX-40 [19].

The 32-bit 80486 uses the same processor used in the 386, but this time the coprocessor 80387 was integrated in the same die as the main processor. The math processor being integrated on the chip allows it to execute math instructions about 3 times as fast as 386/387 combination [4]. The 486 brought a number of mainframe techniques into the x86 worlds for the first time: internal cache, rudimentary branch prediction, integrated FPU, and a five-stage pipeline. A 386 does incorporate pipeline technique. Instructions are loaded one after another into the RAM end of the pipeline and the CPU just takes them from the other end as needed. All modern CPUs are heavily pipelined.

After the 486 processor, Intel came up with the next generation of the IA-32 family of processors with the Pentium processor in 1993. It has internal data paths of 128 and 256 bits to handle multiple 32 bit size simultaneously for processing. It incorporates an advanced programmable interrupt controller (APIC) and multimedia capability – MMX technology. The number of processors per die varies tremendously depending especially on the number of caches that are on the die, this also enhances the speed of execution of instructions by the processors and thus its performance. The main new feature in the fifth-generation P5 Pentium processors was the superscalar architecture, in which two instruction execution units could execute instructions simultaneously in parallel. The P6 family of processors are Pentium pro, P2, P2-Xeon, Celeron, P3, P3-Xeon. The P6 architecture upgrades the superscalar architecture of the P5 processors by adding more instruction execution units and by breaking down the instructions into special micro-ops. This is where the CISC instructions are broken down into more RISC commands. Original P6 processor includes 256KB, 512KB, or 1MB of full-core speed L2 cache. P6 with 512KB L2 cache runs at half-core speed. The P6 Celeron has no L2 cache but Celeron-A has 128KB of on-die full-core speed L2 cache. The P2's L2 cache runs at half speed, the Xeon's runs at full speed, and is available from 512KB to 8MB. The first P3 has 512KB of half-core speed L2 cache and subsequent ones run at full core speed.

Pentium 4. The release of Pentium 4 marks Intel's first all-new x86 design since the Pentium Pro. It is the successor of the P6 family with an entirely new architecture. The Pentium 4 is a single-core mainstream desktop and laptop central processing units (CPUs) introduced in November 20, 2000. Pentium 4s were shipped last on August 8, 2008. They had the 7th-generation microarchitecture, called NetBurst, which includes features such as Hyper Pipelined Technology, Rapid Execution Engine and Execution trace cache which are firsts in this particular micro-architecture with high speed internal data paths. The execution trace cache of L1 stores decoded micro-operations, so that when executing a new instruction, instead of fetching and decoding the instruction again, the CPU can directly access the decoded micro-ops from the trace cache, thereby saving a considerable amount of time. Moreover the micro-ops are cached in their predicted path of execution, which means that when instructions are fetched by the CPU from the cache, they are already present in the correct order of execution. Astonishingly, the Pentium 4 did *not* improve on the old P6 design in either of the normal two key performance measures: integer processing speed or floating-point performance. It was a surprise to the computing community. At 1.5GHz, the Pentium-4 was not only inferior to the Athlons, it couldn't beat the Pentium-III. The Pentium 4 design sacrificed orthodox performance in order to gain two things: clock speed and Streaming SIMD Extension (SSE) performance. While it did quite a lot less per clock-tick than an Athlon or a Pentium-III, it ticked over faster – 1.5GHz on introduction, 2GHz inside the year, and 3 to 4GHz within the next year or so after that. The Pentium 4's SSE unit was exceedingly fast — easily faster than the equivalent SIMD units (be they MMX or SSE or 3Dnow) in any of the AMD chips or in the Pentium 2s and 3s.

Pentium M. which was designed by the Intel's center in Israel was a truly excellent little chip, based on the old Pentium III but heavily revised was designed as a low-power notebook part [22]. The Pentium M blows the P4 into the weeds, and gave an Athlon XP a very good run for its money.

The **Pentium D** brand refers to dual-core desktop microprocessors. It was first introduced in 2005. The Pentium D processor is a multi-core processor, consisting of two Pentium 4 processors and each core sits on its own die. The Intel Pentium D processor features the first desktop dual-core design with two complete processor cores, that each run at the same speed, in one physical package. There are so many Pentium D revisions and the maximum clock speed reached is 3.7 GHz for the extreme edition. It is a desktop processor with core 8xx and 9xx series.

Itanium Processors - The first 64 bit processor which uses the IA-64 was Itanium code named Merced delivered in 2001. Itanium architecture provides for 128 64-bit general purpose registers, 128 82-bit floating-point registers 64 1-bit predicate registers The Intel Architecture – 64-bit (IA-64) is a unique combination of innovative features, such as explicit parallelism (instruction-level parallelism, in which the compiler makes the decisions about which instructions to execute in parallel.), predication, speculation etc. That architecture is designed to be highly scalable to fill the ever increasing performance requirements of various server and workstation market segments. The IA-64 architecture features a revolutionary 64-bit instruction set architecture (ISA) which applies a new processor architecture technology called EPIC (Explicitly Parallel Instruction Computing). This architecture is compatible with the IA-32 instruction set.

TABLE 3
INTEL ITANIUM PROCESSORS

Processors and Date introduced	Clock Speed Internal/External Bus - FSB (MHz)	Register Width (Bits)	External Data Bus/ Address Bus Width (Bits)	Transistor Count per Die (Millions)	On-Die Caches	Die Size/Process Tech. (mm^2/μm)
			Itanium			
Itanium May-01 (Enterprise)	733, 800/266		64/44-16TB	25M	L1 – 2x16K L2 – 256KB L3 – 2, 4MB off die	/0.18
Itanium 2 Jul-02 – Apr-04	0.9 -1 G/400	64	64/40-1TB	220M	L2 – 1,5,	421/180
Nov-04	1.3 – 1.6G/400			410M	3MB	374/130
(Servers)	1.5, 1.6G/				1.5- 6MB	
						374/130
					3, 4, 6, 9MB	
				592M		432/130
Itanium 2 Jul-05	1.6/667M		64/40 – 1TB	1.72B	L2 – 6-9MB	596/90nm
Mid-06						
(Enterprise)	1.6/400		same	same	12MB	same
Dual Core Itanium 2 Jul-06	1.4 -1.6/400		64/40 – 1TB	1.72B	L3 – 16MB	596/90nm

The birth of EPIC started when Hewlett Packard and Intel announced that they were forming an alliance to jointly develop a new 64-bit architecture using existing Very Long Instruction Word (VLIW) technology as a starting point. At that point the 64-bit x86 processor Intel was developing under the code name P7 was quietly dropped in favour of the particular flavour of VLIW that HP researchers had been quietly working on for about five years. Itanium was one of the biggest technological flops in the history of computing. Its sales were so disastrously beneath expectations. Only a few thousand systems using the original *Merced* Itanium processor were sold, due to relatively poor performance, high cost and limited software availability. Recognizing that the lack of software could be a serious problem for the future, Intel made thousands of these early systems available to independent software vendors (ISVs) to stimulate development. HP and Intel brought the next-generation Itanium 2 processor to market a year later.

Core Processors – use Core architecture. It is a multi-core processor microarchitecture that was a replacement for Intel's NetBurst microarchitecture, which has been in use across desktop, mobile and server platforms since its release in the Pentium 4 range of processors. The extreme power consumption of NetBurst-based processors and the resulting inability to effectively increase clock speed was the primary reason Intel abandoned the NetBurst architecture. The Intel Core Microarchitecture was designed by the Intel Israel (IDC) team that previously designed the Pentium M mobile processor. Yonah is the first multicore chip made by Intel using the new 65 nanometer process. AMD which is the closest rival to Intel was almost a year behind in adapting to this technology. Yonah was Intel's first multicore processor that featured both cores on a single die. Although Yonah (Core Duo and Core Solo) features the new Core logo, it doesn't actually make use of the new Core microarchitecture see table 4. When manufacturing the Core Duo processors, Intel encountered defects that rendered one of the two cores inoperable. Since one of the cores was perfectly functional, Intel decided to disable the second core and call the processor the Core Solo. Like the Core Duo, the Core Solo is used exclusively for mobile computers and is not a 64-bit processor. When you begin looking at the Core processors, keep in mind that the clock speeds are down a bit from the Pentium line. This is because the Core technology focuses on improving performance per clock cycle rather than on improving performance by increasing clock cycles. The first chip that carried the new core architecture was Woodcrest, the first eighth generation server and workstation chip, which was released on June 19, 2006. After Woodcrest, was Conroe, the first eighth generation desktop chip that was released on July 23, 2006. Finally, Merom, the first eighth generation mobile chip, that was released in August 2006. The Conroe and Merom feature the new Core 2 Duo logo. Like Intel, AMD has several processors in its dual-core line-up. However, before AMD made the move to dual-core processor technology, it focused on integrating 64-bit technology into the Athlon XP processors. In doing so, AMD was able to take into account the transition to dual-core processors earlier in the design process.

TABLE 4
SELECTED INTEL CORE PROCESSORS

.Processors and Date introduced	Clock Speed Internal/External Bus - FSB (MHz)	Register Width (Bits)	External Data Bus/ Address Bus Width (Bits)	Transistor Count per Die	On-Die Caches	Die Size/Process Tech. ($mm^2/\mu m$)
Core Processors						
Core Solo U1400/U1500 Mar-06 (Mobile PC)	1.2-1.3/533	32	64/31 – 2G	151M	L2 – 2MB	90/65nm
U1300 (Mini & Thin)	1.06/533		same	152M	same	same
Core Solo T1300/T1400 Aug-96	1.66 – 1.83/667	32	64/31 – 2G	152M	L2 – 2MB	90/65nm
Core Duo		32				
Feb-06 T2…. (Mobile)	1.6 – 2/533		64/31 – 2GB	151M	L2 – 2MB	90/65nm
Mar-06 (Mini, Thin)	1.06-1.20/533		64/31 – 2GB	151M	L2 – 2MB	90/65nm
Core i7 Nov-08 Quad Core	2.66 – 2.93/4.8GT/S*	64	64/36-64GB	731M	L3– 8MB	263/45nm
956 Extreme (Desktop)	3.20/		64/36-64GB	731M	L3- 8MB	263/45nm
Core 2 Processor						
Core 2 Solo Jan-06 U2100… (Mobile PC)	1.2, 1.06/533	64	64/36-64GB	291	L2 – 1MB	143/65nm
Core 2 Duo Jul-06		64				
U7…..	1.2, 1.6/533		64/36-64GB	167M	L2 – 2MB	143/65
L7…..	1.33 - /667, 800		64/36-64GB	291M	L2 – 4MB	143/65
T5…	1.66-1.83/667		64/36-64GB	291M	L2 – 2MB	143/65
T7… Jul-07 (Mobile PC)	2- 2.6/667, 800		64/36-64GB	291M	L2 – 4MB	143/65
Core 2 Duo		64				
E6… Jul-06	1.8-2.66/1.066		64/36-64GB	167M, 291M	L2 – 2, 4MB	111/65
E4….Apr-07 (Desktop)	1.8-2.2/800		64/36-64GB	167M	L2 – 2MB	143/65
Core 2 Quad Q6… Jan-07 (Desktop)	2.4-/1066,1333	64	64/36-64GB	582M	L2 – 8MB	286/65
Core 2 Extreme QX9… Nov-07	3GHz/1333	64	64/36-64GB	820M	L2 – 12MB	214/45
Core 2 Quad Q9…Jan-08 (Desktop)	2.5, 2.83/1333	64	64/36-64GB	820M	L2 – 6, 12MB	214/45
Core 2 Duo P9….Dec-08 (Mobile PC)	2.53, 2.66/1066	64	64/36-64GB	410M	L2 – 6MB	107/45

*Intel Quickpath Interconnect – replaces the FSB in Core i7

Core 2 Duo is designed for both desktops and laptops. The Core 2 Duo processor's model number system uses the 6000 series for desktops and the 5000 and 7000 series for laptops. The higher the model number, the higher the clock speed. The processor that powers the computer on which this paper was prepared is Intel Duo 2 Core T5450. A 2 MB Level 2 cache is on die bringing the total number of processors on the die to 291 millions. Each of the processors runs at 1.66 GHz with FSB speed of 667MHz. The process technology is 65nm with a die size of 143mm^2 and a maximum TDP of 35W.

Intel Core i7 is a family of several Intel desktop x86-64 processors, the first processors released using the Intel Nehalem micro-architecture and the successor to the Intel Core 2 families. It has quickpath interconnect which replaces the FSB. This means that the Northbridge chipset is not required. It is a Quad-Core processor with on-die memory controller: the memory is directly connected to the processor. The following caches: 32 KB L1 instruction and 32 KB L1 data cache per core; 256 KB L2 cache (combined instruction and data) per core; 8 MB L3 (combined instruction and data) "inclusive", shared by all cores; Single-die device: all four cores, the memory controller, and all cache are on a single die. It features 8 simultaneous threads with hyper-threading. All Core i7 processors are used in desktop PCs.

Intel's Atom Processor: The fact that processor manufacturers are migrating from the megahertz to low power processors cannot be expressed in a better way than the release of the Atom processor by Intel. As Intel's smallest and lowest power processor, the Intel® Atom™ processor is a single core processor which enables the latest Mobile Internet Devices (MIDs), and another new category of devices for the internet called netbooks (Internet-centric mobile computing devices)" and nettops (basic Internet-centric desktop PCs). Table 1.6 shows that the processor was manufactured using 45nm process and has a core speed of 1.6 GHz and capable of addressing 4 GB of physical memory. The processor remains software compatible with previous 32-bit Intel® architecture and complementary silicon. Intel Atom processor N270 at 1.6 GHz core speed with 533 MHz front-side bus (FSB) has 2.5 watts thermal design power2 (TDP) [13].

TABLE 5
ATOM PROCESSOR

Processors and Date introduced	Clock Speed Internal/External Bus - FSB (MHz)	Register Width (Bits)	External Data Bus/ Address Bus Width (Bits)	Transistor Count per Die	On-Die Caches	Die Size/Process Tech. (mm^2/µm)
			Atom Processor			
Atom 230- single core Q2'08 (Nettop)	1600/533	64		47	L1-32KB Instr 24 Data L2- 512KB	26/45
Atom 330 -2 cores Q3'08 (Nettop)	1600/533	64	64/32-4GB	47	L1-32KB Instr 24 Data L2- 2x512KB	26/45
Atom Jun-08 N270 (Netbook)	1600/533	32	64/32-4GB	47M	L1 – 32KB Instr and 24KB WBD L2 – 512KB	26/45nm
Atom Z5xx Apr-08 (MIDs)	800 -1860/400, 533	32	64/32-4GB	47M	L2 – 512KB	26/45nm

Westmere Processor family: This 32nm chip will debut in the last quarter of 2009. Intel had already upgraded its fabs in the United States to meet the rollout plan. The new 32-nm chips, developed under the code name Westmere, offer increased performance without an increase in the thermal envelope. Mobile and desktop processor production will begin in the fourth quarter of 2009, with an unspecified rollout date to follow. The 32-nm chips will feature two processing cores and four instructional threads, with integrated graphics. Chips for mainstream desktops are being developed under the code name Clarkdale, while the processors for thin and light notebooks are code-named Arrandale.

Microprocessor feature trends are summarised in Table 6. These include microprocessor clock rate trends, increasing number of processors per die, etc. over the last 16 years. The exponentially rising clock rate indicates several changes in testing over the next 10 years. Transistor feature sizes on a VLSI chip reduce roughly by 10.5% per year, resulting in a transistor density increase of roughly 22.1% every year. An almost equal amount of increase is provided by wafer and chip size increases and circuit design and process innovations [14]. This can be seen in Figure 2, which shows a nearly 44% increase in transistors on microprocessor chips every year, approximately doubling every two years as stated by Moore's law.

TABLE 6
FEATURE TRENDS OF PROCESSOR CHIPS

Year	1997 – 2001	2003-2006	2007-2012
Feature Size, µm	0.25-0.15	0.13-0.10	0.07-0.022
Millions of transistors/cm^2	4-10	18-39	84-180
Number of wiring layers	6-7	7-8	8-9
Die size, mm^2	50-385	60-520	70-750
Pin count	100-900	160-1475	260-2690
Colck rate, MHz	200-730	530-1100	840-1830
Voltage, V	1.2-2.5	0.9-1.5	0.5-0.9
Power, W	1.2-1.6	2-96	2.8-109

The current 45nm process will be followed by 32nm going by Intel's assertion in September 2007 that it will ramp up performance and energy efficiency in its microprocessors by using a 32-nanometer process technology starting in 2009. During a keynote at the Intel Developer Forum in San Francisco, Intel president and CEO Paul Otellini showed a 300mm wafer built using the 32-nm manufacturing technology. The chip will house more than 1.9 billion transistors and its increased performance will enable "true to life entertainment and real-life graphics capabilities," Otellini said in his keynote. The chips will be an upgrade over processors built using the 45-nm process. It is important to note that earlier in 2007 a group of chipmakers led by IBM agreed to further collaborate to jointly develop 32-nm semiconductor production technology. Other companies in the collaboration include Freescale Semiconductor, Chartered Semiconductor Manufacturing, Infineon Technologies, and Samsung Electronics. However, Intel is the first company to demonstrate working 32nm processors [15].

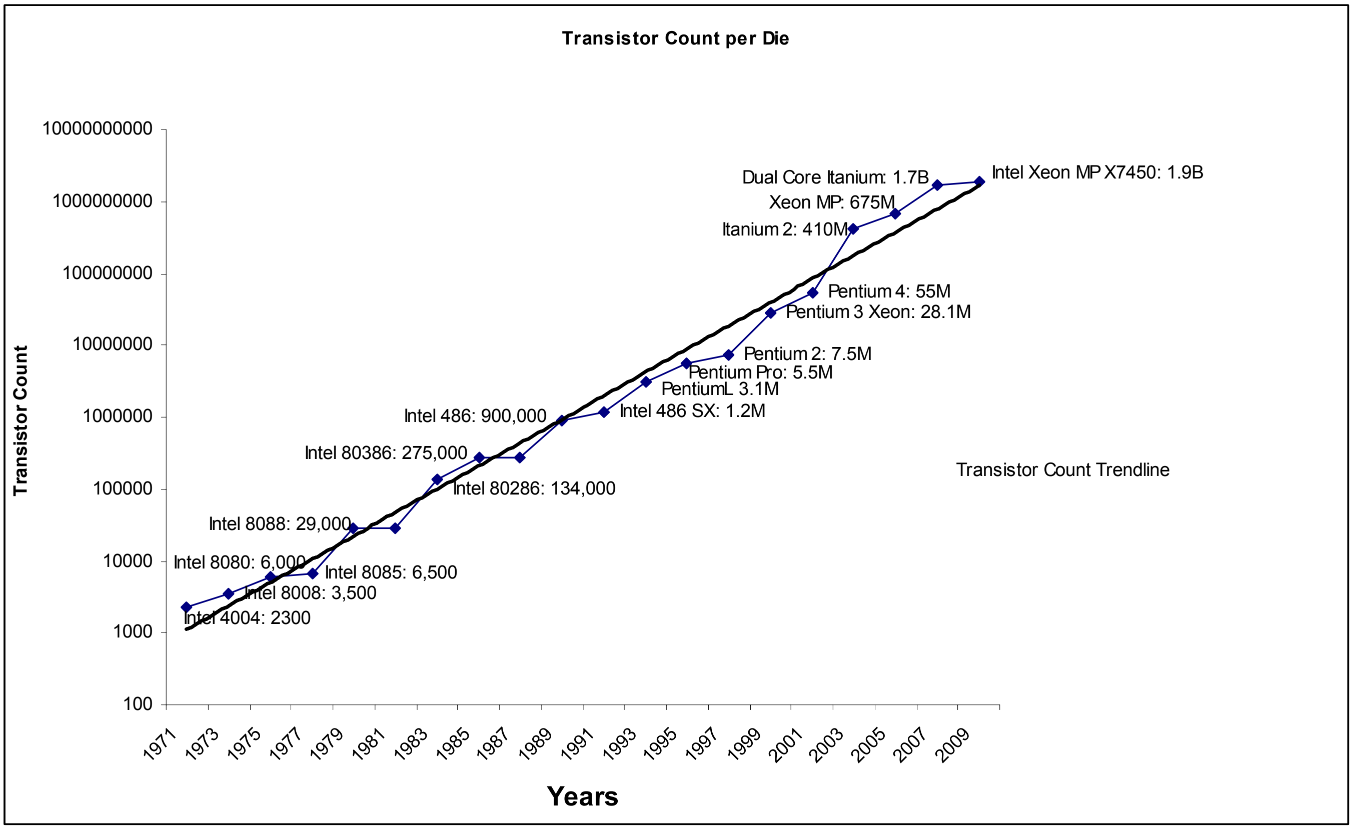


Fig. 2. Transistor Count Trends

Naming and Numbering Convention. For a very long time core speed has been the accepted benchmark used when comparing processors and their performance, processor technology has evolved far beyond the speed of the core and the multiplier alone. After it become evident that performance can no longer be measured by CPU speed alone, Intel moved to a new strategy – processor numbers. Intel opined that the sum of all the features of a processor is far greater than the speed rating alone. Prior to this new strategy processor were named according to its speed, for example, Pentium 4 with HTT 3.40 GHz. And older processors will continue to use the old naming convention.

Intel processor numbers are based on a variety of features that may include the processor's underlying architecture, cache, FSB, clock speed, power and other Intel technologies. A processor number represents a broad set of features that can influence overall computing experience but is not a measurement of performance. Once you decide on a specific processor brand and type, compare processor numbers to verify the processor includes the features you are looking for. Intel's processor number system is used with the following brands: Core processors, Pentium processors Celeron processors, Atom processors, Xeon and Itanium processors [15]. A higher number within a processor class or family generally indicates more features, including: cache, clock speed, Front Side Bus, Intel® QuickPath Interconnect, new instructions, or other Intel technologies. In 2004, Intel introduced processor numbers for desktop and mobile systems with the goal of allowing customers to quickly differentiate among comparable processors and consider more than one processor feature during the selection process. A higher processor number may also have more of one feature and less of another. Let us now look at the numbering scheme for each of the family. The numbering scheme is shown in table 7.

Chipset and motherboard numbers are aligned to the appropriate processor and chipset respectively for ease of matching. Chipset numbers and are indicated by a one-letter suffix. For example, in the name Intel® 5000X, the X identifies this as a chipset and 5000 identifies it as a chipset for the Intel Xeon Processor 5000 series. Again, the letter suffix does not have any inherent meaning — for instance, "A" is not necessarily more powerful than "B." Board numbers are based on the chipset number and are indicated by a 2-letter suffix following the chipset suffix. For example, Intel® Server Board S5000PSL is the server board for the volume segment (Star Lake) family based on the Intel 5000P chipset [20]. Generally Intel processor names consist of two parts, the brand name and the processor number. Show below is an example.

$$\overbrace{\underbrace{\textbf{Intel® Core™ 2 Quad processor}}_{\text{Brand}}\qquad\underbrace{\textbf{Q9550S}}_{\text{Number}}}^{\text{Processor Name}}$$

Core processor: The processor number consists of alpha prefixes followed by four digit numbers. The alpha prefixes indicate whether the processor is for a desktop, server or mobile (laptop). Processor numbers for the Intel® Core™2 processor family brands are categorized with an alpha prefix followed by a four digit numerical sequence. The alpha prefixes also indicate the maximum range of TDP for a given processor. The following alpha prefixes are used to indicate processors for mobile systems: T, P, L, U.

TABLE 7
NUMBERING SCHEME

Alpha Prefix	Description
	Mobile
T	Highly energy efficient processors with TDP 30-39W
P	Highly energy efficient processor with TDP 20-29 W
L	Highly energy efficient with TDP 12-19W
U	Ultra high energy efficient with TDP less than or equal to 11.9W
S	Small form-factor with 22x22 BGA package
T1xxx, U1xxx	Core Solo
T2xxx, L2xxx and U2	Core Duo
T5xxx, T7xxx, P, L, U and S	Core 2 Duo
	Desktop (E, X, Q, and QX)
E	Desktop energy efficient dual-core processors with TDP greater than or equal to 55W (E6xxx)
X	Desktop or mobile dual-core extreme performance processors
Q	Quad-core high performance processors
QX	Quad-core extreme performance processors
920, 940, 965. Core i7-965	**Core i7** high performing processors for extreme gamers and enthusiasts.
E2xxx, E5xxx, E6xxx and E7xxx	**Pentium Dual Core:** TDP that is greater than or equal to 65W
	Intel Server Processors (Xeon and Itanium)
X	High performance
E	Mainstream (rack optimized)
L	Power optimised

Please note that the alpha prefixes X and QX are also used for mobile dual-core and quad core extreme performance processors respectively. Processor number for the Core 2 Quad family has in addition to the alpha prefix and four digit numerical sequence is further identified by an “S” suffix which represents processors having a lower TDP. Itanium Processor 9000 → Multi-processor and dual-processor; Xeon Processor: 7000 series → Multi-Processor; Xeon Processor 5000 series → Dual-processor; Xeon Processor 3000 series → Single-Processor.

Like Intel AMD uses processor numbers to differentiate different processor families. Athlon 64 X2 is the brand name for AMD's line of 64-bit, dual-core processors for desktop computers. Athlon 64 X2 processor's model number system starts with 3800+ and move up to 5600+. The higher the model number, the higher the clock speed. Athlon 64 FX is a version of the Athlon 64 processor aimed at gamers and digital media creation professionals. It has a higher clock speed than Athlon 64 X2, as well a host of other features specifically aimed at enhancing graphics, video, and animation. Athlon 64 FX is available in dual- and quad-core versions: Athlon 64 FX-60 is the dual-core version; Athlon 64 FX-70, Athlon 64 FX-72, and Athlon 64 FX-74 are the quad-core versions. Turion 64 X2 is AMD's 64-bit, dual-core, low-power processor designed for laptops. Currently crop of Turion 64 X2 processor's model number system consists of two letters followed by two numbers: They start with TL-50 and move up to TL-60.

Processors (sometimes in conjunction with chipsets and/or server board components) may also contain other Intel technologies and capabilities that affect performance and may be reflected in incremental processor numbers. These include features such as Hyper-Threading Technology1 , 64-bit technology, Intel® Virtualization Technology, Intel® I/O Acceleration, Technology, Intel® Active Management Technology2 , etc. [20].

Performance Leadership between Intel and AMD Processors. Both AMD and Intel manufacture microprocessors based on the x86 architecture. The war between these two organizations has probably been the most prolonged war in the computer world. The corporate rivalry dates back to 1969, with the setting up of the Advanced Micro Devices Corporation, just one year after the establishment of Intel Corporation. In January 1995, both the organizations settled their litigations, but the processor wars continued. This paper is not targeted at elaborate comparisons between these two giants of the microprocessor world. However it is important to mention a few of the differences between the processors of these giants, especially their earlier processors. So far as cost-no-object performance goes, Intel dominated the mainstream CPU market from the time they started it all in 1971 through until around the end of the 21st century. Until the Athlon arrived to change the landscape, only four times had other chip makers succeeded in making a faster mainstream CPU than Intel's best: Zilog produced their immortal Z-80 in 1976 and dominated the market with it until the rise of the 16-bit 8086/8088 twins in the early '80s. From that time on, Intel had an incredible unbroken run of 18 straight years at the top until it stumbled with their lack-lustre Pentium Pro.

When it comes to the overall power consumption of a system, performance to cost ratio, 3D Gaming, MP3 and

Video Encoding, graphics AMD is a winner when its processors are compared to Intel's previous processors i.e. processors before the debut of the Core 2 and Quad-core processors. AMD processors were cheaper as compared to Intel's previous processors. Experiments have also proved that a machine running on an Intel Core 2 Duo processor consumed, at least 7W more power, than an AMD Sempron. When it comes to cooling and productivity computers working on Intel's Core 2 Duo processor and Quad-core processors definitely have superior cooling features and better heat sinks, when compared to the AMD machines. Not only this, but the Core 2 Duo processors could reach to a speed of 3.2 GHz on proper cooling. Office Productivity and Multitasking (word processing, spreadsheet, internet browsing speed) , which is an important feature that worries every computer user will not be the same for 32-bit and 64-bit processors. A 64-bit processor will definitely outperform 32-bit processors on all platforms. We can say that the situation between processors of these two organisations is fairly balanced.

IV. Fabrication of Microprocessor

An IC is made **from** layers of doped silicon, polysilicon, metal and silicon dioxide, built on top of one another, on a thin silicon wafer. Some of these layers form transistors, and others form planes of connection wires. The basic step in IC fabrication is to construct a layer with a customized pattern, a process known as ***lithography.*** Today's IC device technology typically consists of 10 to 15 layers, and thus the lithography process has to be repeated 10 to 15 times during the fabrication of an IC. The trend in wafers has moved from 200mm (eight-inch) diameter to a bigger, 300mm (12-inch) diameter wafer. This has increased surface area dramatically over the smaller 200mm design and boost chip production to about 675 chips per wafer. Intel and other manufacturers are already using 300mm wafer production.

4.1. Measuring units

In the table1.2 through 1.8 on selected microprocessors discussed above we have made mention of the technology process and die size. It is useful to review the size scale of the chips being discussed to further aid us appreciate how small these chips are. One micrometer is one millionth of a meter and one nanometer is one billionth of a meter. At the micrometer scale is the thickness of a human hair and size of bacteria, viruses are nanometer scale in size and atoms are picometers in size [16]. The smallest processes currently in production up to the third quarter of 2009 are 45nm processes and some of these processes actually have features smaller than 45nm.

4.2. Lithography

Optical lithography is the most important technology employed in chip making and is the pillar that will be very difficult to topple. Lithography is technology that uses radiation with about half the wavelength of purple light. The future chip shrinking will depend on the future of lithography. In both logic and memory chips each of the vast profusion of transistors acts like a switch that allows electrons to flow through the device. A metal-oxide semiconductor field-effect transistor (MOSFET) which is virtually used in all modern chips has three main parts: a source, a drain and a gate. A voltage applied to that gate lets the electrons to flow from source to drain. Physically the gate sits between the source and the drain. The industry sold microprocessors based on how fast the chips could process instructions, and that rate was pretty much directly related to how small the gate width was [1]. This war for a higher processor speed drove the shrinking, with the result that the gate width got smaller than the half-pitch. A Pitch is a distance between metal plates at the source and the drain. Because microprocessor speed was largely determined by the dimensions of the gate, by 2000 the gate had become the smallest feature produced in the semiconductor industry. For logic devices the gate length became the smallest feature, but for memory the half-pitch remained the smallest feature [1]. For memories speed is also a key parameter, but there was no similar war between memory manufacturers seeking to drive up the clock frequency. They rather concentrated on reducing the size of each memory cell on their chips so that they could squeeze ever

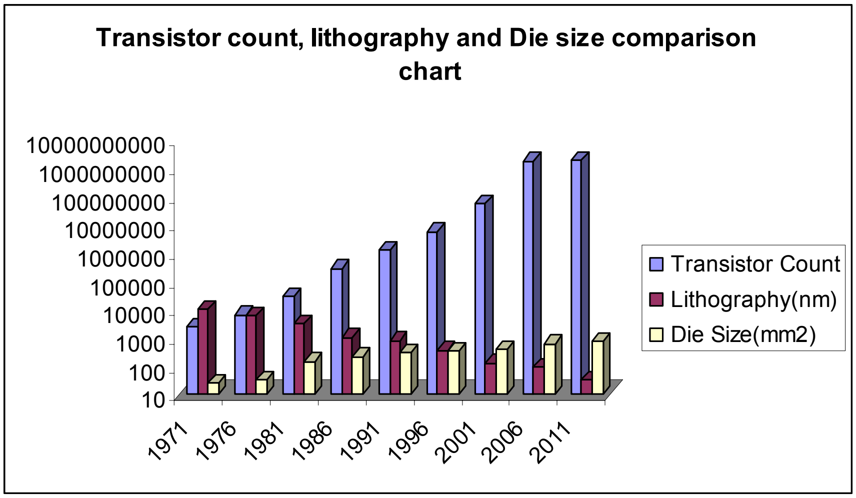
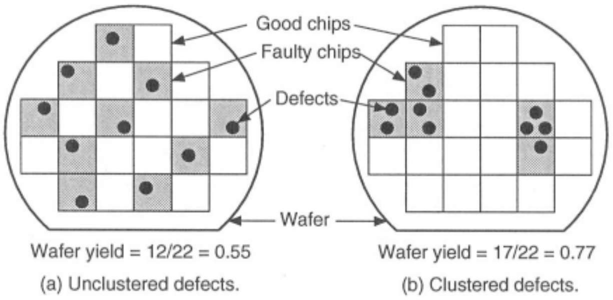


Fig. 3. Technology Trends

more bits into less and less real estate. It is important to note at this juncture that the shrink rate of memories is higher than that of logic. And it is likely that memories will embrace the new technology that will replace optical lithography – EUV (Extreme Ultra-Violet lithography) [17]. EUV is also uses optics, since it uses electromagnetic radiation but has a wavelength of about one-fifteenth in length. Other next generation lithography are electron beam and imprint lithography. For more detailed information on lithography process technology refer to see [1] [16]. The impact of the process technology on the overall processor count per die is shown in figure 3. It is evident that the smaller the process technology (lithography) the larger the number of transistors per die and consequently the speed, increases and TDP reduces. With increased transistor count per die has led to an increase in die size. Moore's law is holding up due to the reduced feature sizes which are possible as a result of optical process technology. The feature sizes will continue to decrease as the semiconductor roadmap moves to the 22nm process technology as year 2011 approaches.

4.3 Yield, Chip testing and grading

As each 8 inch or 12 inch wafer rolls off the production line, it is split up into its individual chips after undergoing a special feature tests. Ideally, they would be all exactly identical, but in practice there are tiny variations between them. Process variations, such as impurities in wafer material and chemicals, dust particles on masks or in the projection system, mask misalignment, incorrect temperature control, etc., can produce defects on wafers. The term *defect* generally refers to a physical imperfection in the processed wafer [2]. The stuck-at fault model is the most commonly used model in VLSI testing and fault tolerance schemes. In this model, a physical defect manifests itself as a signal consistently having a certain value (either zero or one) independent of the input [3]. These defects affect the process yield. The *process yield* of a manufacturing process is defined as the fraction (or percentage) of acceptable parts among all parts that are fabricated (The ratio of good to bad chips on a wafer). See figure 4



Source [2]

Fig. 4. Defect modelling for yield estimation

Typical defects are broken conductors, missing contacts, bridging between conductors, missing transistors, incorrect doping levels, and many other phenomena that can cause the circuit to fail. Some defects are observable through the optical or electron microscope. Others are not visible and can only be detected by electrical tests. To estimate the VLSI yield, defects are modelled as random phenomena. For more details refer to [2].

Yields well under 50 percent are common when a new chip starts production; however, by the end of a given chip's life, the yields are normally in the 90 percent range. Most chip manufacturers guard their yield figures and are very secretive about them because knowledge of yield problems can give their competitors an edge. After testing each die (chips) on the wafer, the bad ones are marked. All the dies are cut from the wafer and each retested, packaged and retested again. The packaging process is also referred to as bonding, because the die is placed into chip housing where a special machine bonds fine gold wires between the die and the pins on the chip. The package is the container for the chip die, and it essentially seals it from the environment. This testing (pressures, temperatures, and speeds) after packaging is to detect the point at which the chip will stop working. At this point, the maximum successful speed is noted and the final chips are sorted into bins with those that tested at a similar speed. For example, the Pentium III 750, 866, and 1000 are all exactly the same chip made using the same process. They were sorted at the end of the manufacturing cycle by speed. The paradox is that Intel often sells a lot more of the lower-priced 933 and 866MHz chips, so it will just dip into the bin of 1000MHz processors and label them as 933 or 866 chips and sell them that way. People began discovering that many of the lower-rated chips would actually run at speeds much higher than they were rated, and the business of overclocking was born. Overclocking describes the operation of a chip at a speed higher than it was rated for [21]. With the birth of overclocking vendors who reap where they did not sow starting remarking slower chips and reselling them as if they were faster. Because most of the Intel and AMD processors are produced with a generous safety margin—that is, they will normally run well past their rated speed—the remarked chips would seem to work fine in most cases. Of course, in many cases they wouldn't work fine, and the system would end up crashing or locking up periodically [21]. Intel and AMD have stopped the trend (remarking fraud) by building overclock protection in the form of a multiplier lock into most of its newer chips.

4.4 Chip foundries

Having discussed microprocessors up to this point it becomes necessary to know where these chips are manufactured; the location of the foundries. Chip foundries fabricate anyone's chips on contract basis. The question is "Where are the foundries (fabs) located? It will interest you to know that most of the foundries are located in the Far East countries – Taiwan, South Korea, Philippines, China, Singapore. Taiwan for instance is home to several chip foundries, the two biggest being Taiwan Semiconductor Manufacturing Co. and United Microelectronics Corp., both headquartered in Hsinchu [1]. It is home to 6 of the 10 major DRAM manufacturers – Inotera, Nanya, Powerchip, ProMos, Rexchip and Winbond [5]. Some of the semiconductor companies no longer do any of their own fabrication. Some who used to make their own chips have become fab-lite. They now retain some facilities to develop the initial technology but then send their volume business to foundries.

This is done purely from economic point of view as it allows these companies to do rapid development under their own control and avoid the cost of massive fabs for volume. Building a new fab is quite expensive and can cost about $5 billion. This explains why more and more American and European countries take this approach [1].. Integrated device makers like Intel, Samsung, Toshiba and IBM design and build all their own chips. AMD, Freescale and Texas Instruments have recently gone to the fabs. IBM Microelectronics in East Fishkill N.Y. has become a dominant foundry for game processors. Fujitsu also maintains a 65nm fab in Japan. After Intel and Samsung Texas Instrument is the third largest manufacturer of semiconductors worldwide.

Conclusion

This paper has examined most of the processors by Intel and AMD that have made their marks in the last three decades. Some of them did not live up to expectations as vendors continually played tricks on the consumers especially with their first releases of their new line of processors. In most cases new processors have been found to perform lower or operate at the same level of performance as their older cousins. Rushing to purchase a new processor may not be the right thing to do at the moment. But for how long shall one continue to wait for the second release. We should try to endure if our current processor is still serving us well. We have revealed that Moore's law will continue to endure as long as new process technologies reduce the feature sizes on the die. The use of EUV process will certainly reduce feature sizes further. We have seen that the better the process technology the more transistors we can squeeze on the die, the faster the processor and the less thermal design power and ultimately the cheaper they become.

References

[1] Bill Arnold, "Shrinking Possibilities", IEEE Spectrum, pp. 23~25, April 2009.

[2] Michael L. Bushnell and Vishwani D. Agrawal, Essentials of Digital Testing for Digital, Memory and Mixed-Signal VLSI Circuits: ISBN 0-306-47040-3, Kluwer Academic Publishers, 2002.

[3] Fred A. Bower, Sule Ozev, and Daniel J. Sorin, "Autonomic Microprocessor Execution via Self-Repairing Arrays", IEEE Transactions on Dependable and Secure Computing, Vol. 2, No. 4, pp. 297~310, Oct-Dec 2005.

[4] Douglas V. Hall, Microprocessors and Interfacing Revised 2nd ed., ISBN: 0-07-060167-4, New Delhi, Tata McGraw-Hill, 2007.

[5] Yu-Tzu and Samuel K. Moore, "Taiwan's Troubled DRAM Plan", IEEE Spectrum, pp. 9~10, June 2009.

[6] Brian R. Santo, "25 Microchips That Shook the World", IEEE Spectrum May 2009.

[7] Keith Diefendorff, "History of the PowerPC Architecture", Communication of the ACM 37, 6, pp.28~33, June 1994.

[8] Indiana University Information technology Services Knowledge Base, "What are CISC and RISC technologies, and how do they compare?",

[9] Patterson, D.S. and Ditzel, D.R. "The case for the reduced instruction set computer", Computer Architecture News 8:6 pp. 25-33, Oct.15, 1980.

[10] Jeff Prosise, "RISC vs. CISC: The Real Story – What makes the PowerPC a RISC processor and the Pentium a CISC? ", http://www.zdnet.com/pcmag/pctech/content/14/18/tu1418.001.html.

[11] Texas Instruments Datasheets and Cross Reference" http://www.supplyframe.com/datasheet-pdf/manufacturer/texas+instruments?id=2477

[12] Intel Corporation, "Microprocessor Quick Reference Guide", http://www.intel.com.

[13] Intel, "Embedded and Communications", Product Brief, intel.com/go/embedded.

[14] Harriott L. R., "A New Role for E-Beam: Electron Projection," IEEE Spectrum, vol. 36, no. 7, pp. 41–45, July 1999.

[15] Intel Corporation, "White Paper Introduction to Intel's 32nm Process Technology", www.intel.com.

[16] Scotten W. Jones, Introduction of Integrated Circuit Technology, 4th ed, IC Knowledge LLC.

[17] Chris A. Mack, "Seeing Double", IEEE Spectrum, pp. 123, November 2008.

[18] Wikipedia

[19] Red Hill Technology, "The Red Hill CPU Guide", http://www.redhill.net.au/index.html.

[20] Intel, "New numbering system for Interl Server Processors", www.intel.com.

[21] Scott Mueller and Mark Edward Scoper, "Microprocessor Types and Specifications", InfromIT Network, file://J:\MacmillanComputerPublishing\chapters\JW003.html 3/22/01].

[22] Intel Pentium M power data, 1.6 GHz, 0.13μm technology, 1 MB L2 cache

[23] http://www.intel.com/design/intarch/pentiumm/pentiumm.htm

Optic Link System

Filippenko Igor Olegovich, Vladimir Ivanovich Hahanov
Computer Engineering Faculty, Kharkov National University of Radioelectronics, Lenin Ave. 14,
Kharkov, Ukraine, 61166, phone: (057) 70-21-421, (057) 70-21-326
E-mail: hahanov@kture.kharkov.ua; kiu@kture.kharkov.ua; fios@inbox.ru

Abstract – **In given article is overviewed the digital optic link system and the problem of encoding analog information in communication systems. This system can be used as warless computer networking with out paying for radio band. As optic transmitter used red semiconductor laser.**

I. Introduction

At present optical channels of information transfer received wide enough application. Examples of the given systems can be remote controls, the infra-red transceivers in embedded systems and opt link computer and telephone networks.

Thus, it was decided to make communication system with optic link communication channel with throughput up speed near 10 Mbps.

The optic link systems are useful, if needed to set communication channel for high distance without using high cost radio LAN systems as WiMAX or WiFi. The sermon puss of this system – you need no pay for radio band and components used on it are very chip.

II. Overview

For construction of the given system and evident demonstration of its serviceability the working model was assembled. In a role of the transmitted information was chosen digital speech stream. Physically all system can be divided into two blocks which are carrying out the following tasks:

- Coding speech in a digital stream
- Transfer digital signal by optical channel

the schematic diagram of this system is given below on Figure 1.

Manuscript received October 2, 2009.

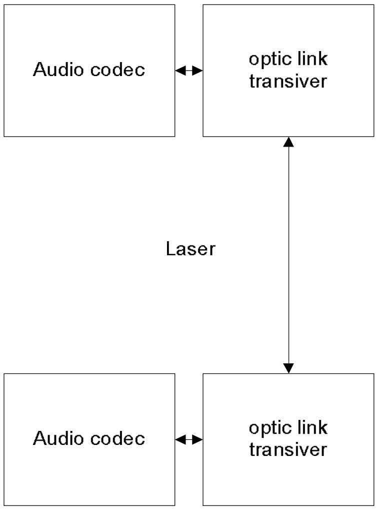


Fig. 1. Optic link system

During researches, it was revealed, that the transfer on the optical communication channel of information with presence of a constant component is not possible without special transformations. The reason to that is noise immunity. For example, in a case over light of the photo receiver an extraneous light source can be understood as a useful signal, thus, not being by those. In a consequence complete, loose of the useful information (payload).

For avoidance of it, it was undertaken to build in the reception module the site filtering a constant component of a signal. Thus the useful signal will be transferred as a code without a constant component. There are some kinds of such code, in this case was chosen one of most widespread - Manchester code.

It represents a meander with frequency twice above than base signal (absence of a constant component). The phase of a signal changes only at transition from unit in a zero and back.

The given principle is used in ports of Irda standard, and in local computer networks. On the Figure 2. Manchester code is given.

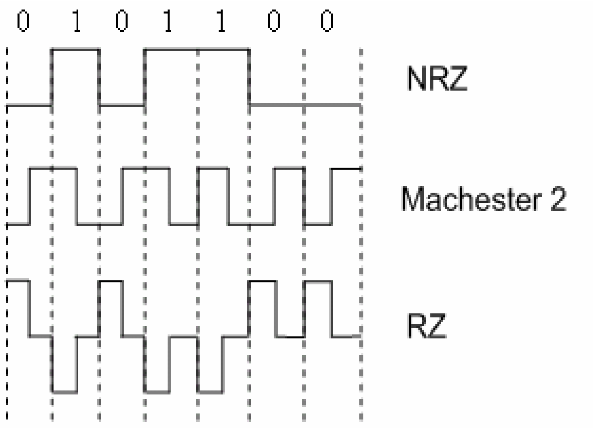


Fig. 2. Manchester 2

III. Coding of speech

For simplification in the article described only two analog channel multiplexing in one digital. Two analog

Channels multiplexing is not limit for the system and if it needed number of channels can by increase. As we can see from figure 1, digital communication system consist of two channels, named: left and right. Both analog channels are connected to ADC via analog multiplexor. Time domain multiplexing is realized by alternating multiplexor switching.

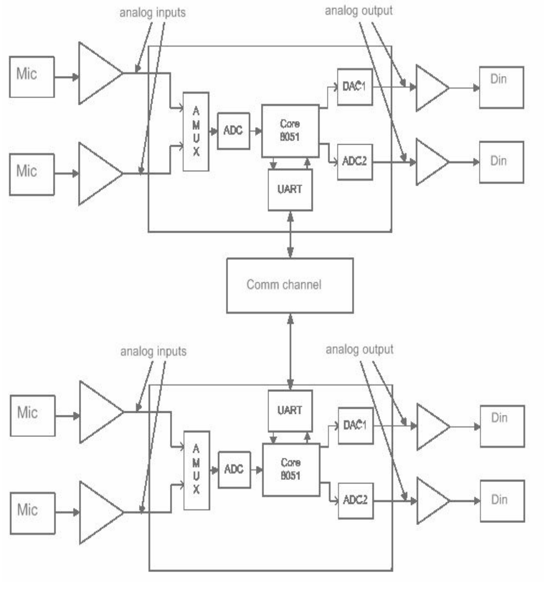


Fig. 3. Digital communication system

As shown at Fig 3 the low frequency analog signal for example from microphone comes to preamplifier. Preamplifier is needed for amplification of weak signal from the microphone to value (In this case 2.5 V), which needed for correct processing in ADC and signal to noise ratio increasing. Preamplifier is implemented on chip LM324.

After preamplification, analog signal comes to input of analog multiplexor (AMUX) which integrated into chip 8051F042. Further conversions are provided in this chip. From the multiplexor output, left and right analog channel signals are sequentially comes to the ADCs input. After analog to digital conversion signal comes to central processor unit (CPU).

The architecture, core speed and instruction set allow to realize all needed algorithms, such as amplitude normalization, signal compression, variable gain amplification, code domain channel interleaving.

Amplitude normalization based on pseudo instant companding algorithm. The essential of this method is a delaying signal during approximately 10 mS, measuring of maximum level of signal at this duration, calculation of coefficient of amplification and multiplication of delayed signal by coefficient of amplification. In common view this method de pictured on Fig 4.

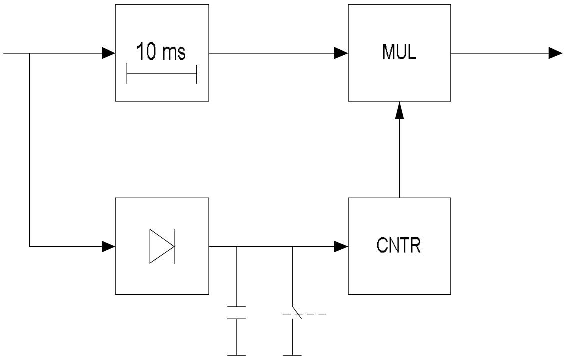


Fig. 4. Amplitude formulizer in common view

Variable gain amplifier allows to adjust level of transmitted audio signal according commands from remote receiver.

Before sending thru communication channel processed digital signal is encoded to packets and send them in Manchester code.

Packet includes: preamble, address, payload and check sum. Besides, packet also includes additional information markers, which make possible correct code demultiplexing by reviser. As result of code demultiplexing by receiver there are two digital streams which after processing by CPU (un compression, etc.) flows to DAC.

DAC provide restoring digital streams to analog signal signals.

IV. Optic link transceiver

The optical transceiver is submitted as two independent blocks: transmitter and receiver. The transmitter inverts incoming digital stream and transfers it to semiconductor laser diode. Using semiconductor lasers is very effective for it high coefficient of useful work and high speed reaction.

The receiver amplifies the signal, received by the photo diode, amplified it up to amplitude in 3.3 volts, filters a constant component, comparing the current meaning with previous(noise immunity), inverts it. On the output we have a restored digital stream.

V. Conclusion

There is variation of adoptive digital communication system with multiplexing a several analog channels in to one digital and optic link communication channel is offered. However components used for design implementation allows realize needed analog processing algorithm.

1. The way described in this article allows to make cost-effective Optic link system.

2. Given system shows high efficacy during testing. Testing time – 1 hour, errors – 0,2%, speed – 10 Mbps.

VI. Future implementation

In future this system can be worked on speed up to 100 Mbps and used in optic coir communication, warless LAN connections e. t. c

References

[1] User's reference manual to C8051F04x Silicon Laboratories Inc. 2005. p 29-180.

[2] Frunze A. V. Microcontrollers? It is easy!, M.: «OOO ID SKIMEN», 2002. p 6-240.

[3] Fedorkov B. G., Telec V.A. DAC and ADC microshems: functionality, parameters, implementation M.: ENERGOIZDAT, 1990. p 63-90.

[4] Goroshkov B.I. Radio electronic devises M.: Radio and communication, 1984. p 6-184.

[5] G. Shild C: The Complete Reverence M.: Williams Publishing, 2002. p 40-190.

Descriptor Neural Networks with Arbitrary Characteristic Index

Hahanov V.I., Rutkas A.A.

Abstract – **We consider a difference descriptor system and its modeling with the help of a neural network. The corresponding descriptor network is a special connection of dynamic and static neurons. The network configuration is defined by the Weierstrass's normal form of regular matrix sheaf.**

Keywords – **Descriptor system, neural network, Weierstrass's normal form.**

A descriptor control system is described by differential-algebraic equations, and vector equation of the states of the system contains a singular matrix at the vector of derivatives [1]. A transition from derivatives to finite differences generates a vector difference algebraic equation [2,3]

$$Ax(k+1)+Bx(k)=f_k(x(k)),\quad k=0,1,2,... \qquad (1)$$

Here A,B - square ($n\times n$) matrices, and the criterion of the descriptor property of the system is the noninvertibility A ($\det A=0$). If the characteristic pencil $\lambda A+B$ is regular $\det(\lambda A+B)\not\equiv 0$, then it turns to the normal form of K. Weierstrass [4], and resolvent matrix-function $(\lambda A+B)^{-1}$ exists for large λ and satisfies the power estimate [6]

$$\left\|(\lambda A+B)^{-1}\right\|\le C|\lambda|^{p-1},\ |\lambda|>r \qquad (2)$$

The minimal integer $p\ge 0$ such that estimate (2) is valid is called *the index* of the matrix pencil $\lambda A+B$. Also we call the integer p the characteristic index of system (1). If the matrix A is invertible, in particular when $A=E$, then the index $p=0$ and system (1) is explicit difference system. Thus, if $p\ge 1$, then system (1) is descriptor.

In [5], there is considered a discrete neural network, which models descriptor system (1) of index $p=1$. Here we construct and analyze an artificial neural network, which is described by equations (1) with arbitrary index $p\ge 1$. It is natural to call the built network the descriptor neural network of index p, (Fig. 1). Its construction is determined by the index p and the normal form of the pencil of matrices $\lambda A+B$. Suppose that the matrices A,B in (1) have the block-diagonal normalized form (compare with [4, 5 and 6])

$$A=\begin{pmatrix}E_m & 0\\ 0 & H\end{pmatrix},\ B=\begin{pmatrix}J & 0\\ 0 & E_{n-m}\end{pmatrix},\ H^p=0 \qquad (3)$$

The index p of the pencil $\lambda A+B$ coincides with the nilpotency index of the matrix block H: $H^{p-1}\ne 0$, $H^p=0$. Here E_m designates single ($m\times m$)-matrix, J - any ($m\times m$) matrix. In the general case, the matrix H can be a block-diagonal matrix, containing any amount of nilpotent Jordan cells of sizes $s_k\le p$, so that at least one cell has the maximum size p and $\sum_k s_k=n-m$. To find a model of neural network structure, it is enough to consider the case of $p=n-m$ such that H is a unique nilpotent Jordan cell with size p. In accordance to the approach in the theory of neural networks there are nonlinear vector functions $f_k(x)$ in (1) are chosen in the form $f_k(x)=\Psi(Wx+\Theta(k))$, where elements w_{ik} of the matrix W are interpreted as synaptic weights, components $\Theta_i(k)$ of the vector $\Theta(k)$ - as depositions (external influences at k-th step). In accordance to breaking up of matrices on blocks (3), the vectors x,Ψ have the representations:

$$x=\begin{bmatrix}v\\ h\end{bmatrix},\ \Psi=\begin{bmatrix}\varphi\\ \vartheta\end{bmatrix};\ v=\begin{bmatrix}x_1\\ ...\\ x_m\end{bmatrix},\ \varphi=\begin{bmatrix}\psi_1\\ ...\\ \psi_m\end{bmatrix}.$$

Then the vector equations of the states (1) are rewritten in the form:

$$v(k+1)+Jv(k)=\varphi(Wx(k)+\Theta(k)) \qquad (4)$$

$$\left.\begin{aligned}x_{m+2}(k+1)+x_{m+1}(k)&=\psi_{m+1}(Wx(k)+\Theta(k))\\ x_{m+3}(k+1)+x_{m+2}(k)&=\psi_{m+2}(Wx(k)+\Theta(k))\\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\ x_n(k+1)+x_{n-1}(k)&=\psi_{n-1}(Wx(k)+\Theta(k))\end{aligned}\right\} \qquad (5)$$

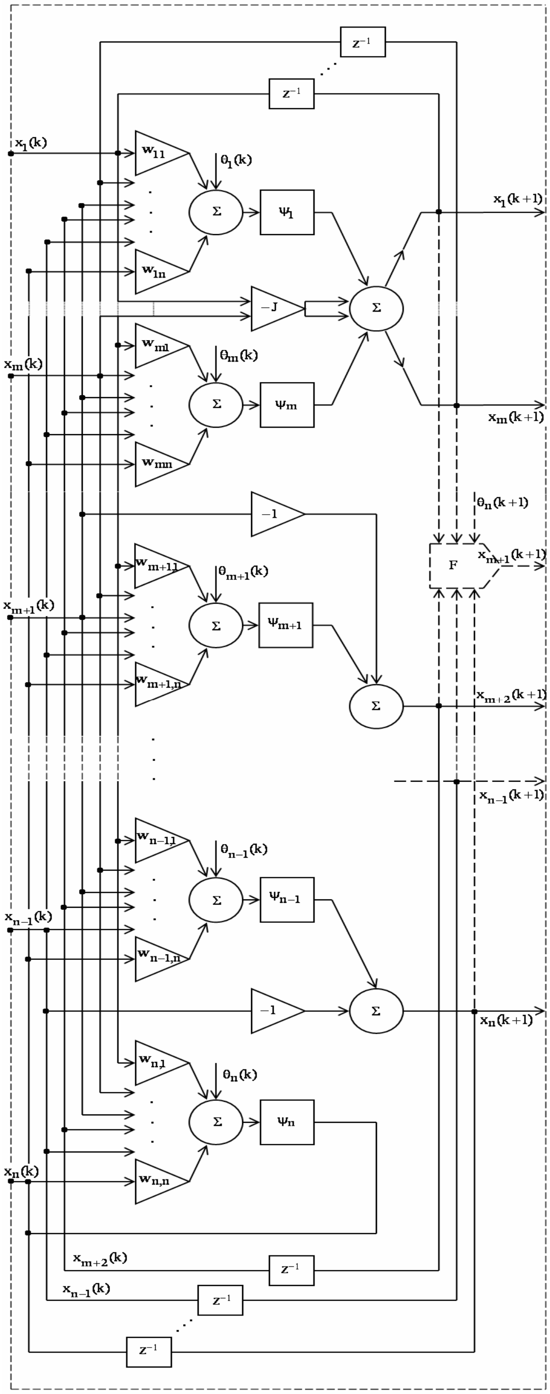
$$x_n(k)=\psi_n(Wx(k)+\Theta(k)) \qquad (6)$$

The main dynamic block (4) of m equations can be realized as Hopfield vector dynamic neuron with additional block of multiplying by the matrix ($-J$).

Manuscript received October 30, 2009.

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).
Rutkas Andrey is with the Kharkov National University of Radioelectronics, Ukraine, 61166, Kharkov, Lenine Prosp., 14.

In Figure the corresponding dynamic subnet with entrance $v(k)=[x_1(k),...,x_m(k)]^{tr}$ and output $v(k+1)=[x_1(k+1),...,x_m(k+1)]^{tr}$ is represented in the case of *nonmutual* activation functions $\psi_i(u)=\psi_i(u_i)$, $u=Wx+\Theta$, depending on the component u_i of the vector of internal state $u=u(k)$. Therefore, m Hopfield classical dynamical neurons are used in the network realization of equations (4).



Descriptor network of index $p=n-m$

The descriptor part of the neural network in Figure transforms the part $(x_{m+1}(k),...,x_n(k))$ of the entrance vector into the vector $(x_{m+2}(k+1),...,x_n(k+1),0)$, which is a result of the left shift of the vector $(x_{m+1}(k+1),...,x_n(k+1))$. For this purpose, the special connection of $(n-m-1)$ dynamical neurons and McCuloch-Pitts static neuron with activation function ψ_n is used. Static (or algebraic) equation (6) has the following form for ($k+1$)-st step:

$$\psi_n(\sum_{j=1}^{n} w_{n,j}x_j(k+1)+\Theta_n(k+1))-x_n(k+1)=0 \quad (7)$$

Under the conditions

$$w_{n,n+1}\neq 0,\ \frac{d\psi_n(u_n)}{du_n}\neq 0,\ \forall u_n\in R,$$

equation (7) can be explicitly solved in the components

$$x_{m+1}(k+1)=F[(x_1,...,x_m;x_{m+2},...,x_n;\Theta_n)(k+1)] \quad (8)$$

The dotted block F in Fig. 1 corresponds to solution (8). Equations (4),(5),(8) determine the recurrence operator $S_{k+1}(x(k))=x(k+1)$. The operator S_{k+1} depends on the parameters $\Theta(k),\Theta(k+1)$ and it is defined on the manifold $\Lambda_k=\{x(k)\}$ of vectors $x(k)\in R^n$ which satisfy the scalar equation (6).

REFERENCES

[1] Bender D.J., Laub A. *The linear-quadratic optimal regulator for descriptor systems*, IEEE Transactions on Automatic Control. 1987. Vol. AC-32, №6. P.2062-2077.

[2] Bender D.J. Laub A. The linear-quadratic optimal regulator descriptor systems: discrete-time case. Automatica 23 (1987), P.71-85.

[3] Bondarenko M.F., Rutkas A.G. On a class of implicit difference equations, Dopovidi NAN Ukraine. 1998. №7. P. 11-15.

[4] Gantmacher F.R. Theory of matrices. M.: Nauka, 1966. 576p.

[5] Rutkas A. Descriptor Neural Networks And Singular Implicit Dynamic Systems, Proceedings of IEEE East-West Design & Test Synposium (EWDTS'08), Lviv, Ukraine, October 9-12, 2008, P.429-430.

[6] Vlasenko L.A. Evolution models with implicit and degenerate differential equations, Dniepropetrovsk: System technologies, 2006. 273p.

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.

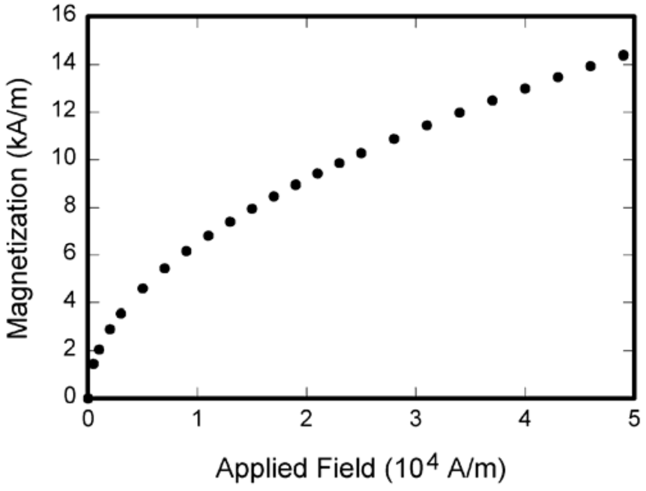


Fig. 1. Magnetization as a function of applied field. Note that "Fig." is abbreviated. There is a period after the figure number, followed by two spaces. It is good practice to explain the significance of the figure in the caption.

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 → 10^{-8} Wb = 10^{-8} V·s
B	magnetic flux density, magnetic induction	1 G → 10^{-4} T = 10^{-4} Wb/m^2
H	magnetic field strength	1 Oe → $10^3/(4\pi)$ A/m
m	magnetic moment	1 erg/G = 1 emu → 10^{-3} A·m^2 = 10^{-3} J/T
M	magnetization	1 erg/(G·cm^3) = 1 emu/cm^3 → 10^3 A/m
$4\pi M$	magnetization	1 G → $10^3/(4\pi)$ A/m
σ	specific magnetization	1 erg/(G·g) = 1 emu/g → 1 A·m^2/kg
j	magnetic dipole moment	1 erg/G = 1 emu → $4\pi \times 10^{-10}$ Wb·m
J	magnetic polarization	1 erg/(G·cm^3) = 1 emu/cm^3 → $4\pi \times 10^{-4}$ T
χ, κ	susceptibility	1 → 4π
χ_ρ	mass susceptibility	1 cm^3/g → $4\pi \times 10^{-3}$ m^3/kg
μ	permeability	1 → $4\pi \times 10^{-7}$ 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 → 10^{-1} J/m^3
N, D	demagnetizing factor	1 → $1/(4\pi)$

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: 27.12.2009. Format 60×84 1/8.
Relative printer’s sheets: 5,93. Circulation: 300 copies.
Published by SPD FL “Prostor”.
Lenin ave, 14, Kharkov, 61166, Ukraine

Рекомендовано Вченою радою Харківського національного
університету радіоелектроніки (протокол № 4 від 27.1209.2009)
Підписано до друку 27.12.2009. Формат 60×841/8.
Умов. друк. арк. 5,93 . Тираж 300 прим. Ціна договірна.
Віддруковано у ПП «Простор»
61166, Харків, просп. Леніна, 14.