OPIC
Office de la propriété intellectuelle du Canada

CIPO
Canadian Intellectual Property Office

(12) (19) (CA) Demande-Application

(21) (A1) 2,293,953

(22) 2000/01/05
(43) 2001/07/05

(72) KHMELNIK, SOLOMON, IL
(71) IPROS CORPORATION, CA

(51) Int.Cl.^{7} G06F 7/48

(54) **METHODE ET SYSTEME DE MISE EN OEUVRE D'UN COPROCESSEUR**

(54) **A METHOD AND SYSTEM FOR IMPLEMENTING A COPROCESSOR**

A Method and System for Implementing a Coprocessor

The method of positional coding complex numbers, multi-dimensional vectors, geometrical figures and functions

1. Preface

All processors, developed till present time, operate with numbers. In other words, for available processors the arithmetic language is a natural one, so they very fast execute the operations inherent in human. *It is proposed a new computer arithmetic* that is not characteristic for a human being. Possessing by that, a computer can operate with complicated mathematical objects. The first step in this direction had been made by C. Shannon as long ago as 1950, when he had proposed the arithmetic of negative numbers. The author of the given proposal has developed the arithmetic for complex numbers, multi-dimensional vectors, plane and spatial geometrical figures, functions of one and of many arguments. Operations with such objects are executed in present computers by means of subprograms, that considerably increases expenses of computer resources. The existing methods of increasing speed of such calculations consist in constructing multi-processor systems and development high-speed logical elements. *However, their speed already approaches to the physical limit, therefore the hardware methods of increasing computers' high speed become more and more necessary.*

Thus, for the first time it is proposed an ***original mathematical theory*** that includes the ***method of coding complicated mathematical objects***. This method allows to represent complicated mathematical objects by the binary code, **that is one for the object as a whole**. There have been developed also ***algorithms of the operations*** with such codes. Moreover, there is proposed a realization of these algorithms. This realization is based on the new methods of constructing circuits of specialized processors. These processors contain arithmetical blocks for executing various operations with the mentioned mathematical objects. As such the operations may be arithmetical ones, solving transcendental equations, fast Fourier transformations, geometrical transformations of figures, differentiation and integration of functions, etc.

2. Novelty.

For the first time in the world practice the method of positional coding complex numbers on the various bases has been developed. On the positional codes of complex numbers can be generalized the method "numeral after numeral" for hardware calculation of elementary functions.

Thank to that, it is often enough to have hardware for the realization only exponentiation and taking logarithms, as far as it is possible to express all elementary functions through these functions in the complex area. Besides, this method may be implemented for constructing algorithms of the hardware solving transcendental equations and systems of such equations. By using codes of complex numbers the class of such equations is expanded in comparison with real numbers and algorithms of their solution are essentially simplified.

Then, positional codes of vectors, functions and geometrical figures were developed. It should note that the codes of geometrical figures can be considered as codes of numerical data files and algorithms of effective searching can be constructed for them. The preference being given to positional codes is explained mainly by the simplicity of the execution of the arithmetic operations. So, out of the dependence on an object of coding, the addition of positional codes is related to practicing carries from junior digits to senior ones, while the multiplication consists of shifts (i.e., renumbering digits) and additions. The mentioned above method "numeral after numeral" is generally applied only in combination with the positional coding system (in particular, for solving transcendental equations with complex variables, for example, in navigation calculations).

The computers, operating with complex numbers, three- and many-dimensional vectors, are destined for the numerical solving tasks with functions of complex and vector arguments. These computers are effective in the tasks of management, of representation, of electric power industry.

The computers, operating with geometrical figures, are destined for geometrical transformations, that is necessary in the tasks of recognition, representation, designing.

The computers, operating with functions, are destined for analytical transformations, for solving integral and differential equations; for solving systems of linear, algebraic, transcendental, differential equations. Such computers, like analogous machines, operate with functions as whole ones; for example, they sum up, multiply, differentiate and integrate continuous functions.

The special algebra used in operations with codes of multidimensional vectors and geometrical figures has been developed as well.

3. The position codes of complex numbers and vectors.

The position codes of the multi-dimensional vectors Z are based on their representation as the decomposition

$$Z = \sum_{(m)} r_m \rho^{m}, \qquad (1)$$

where ρ - basis of coding, number or vector,

m - number of the category,

r - category of the decomposition, number or vector, accepting values from the limited set

$$A_R = \{a_0, a_1, a_2, \ldots, a_j, \ldots, a_{R-1}\},$$

containing R of various quantities a_j. The position code of the appropriate vector Z for this decomposition has a form

$$K(Z) = \ldots \sigma_m \ldots,$$

where σ_m - digit designating size r_m.

The formula (1) includes operation of addition and multiplication. For being algorithms of operations with such decomposition (or, it is the some, with position codes) addition and multiplication should be associative and commutative, they should obey distributive law as well. Hence, some set of objects should constitute a ring for being opportunity of position coding this set. The set of a real numbers and the set of multi-dimensional vectors, in whose the operations of addition and multiplication by a quantity have been determined, satisfy such a demand. For the real numbers the position systems are known. For the mentioned set of vectors a position numbering systems with the real basis will be constructed below.

Set of complex numbers makes ring and for it the position numbering systems on the real and complex bases will be constructed too.

It must be determined the operation of multiplying vectors on the vector basis in order to construct the position numbering system of multi-dimensional vectors, this operation obeys laws mentioned above. In other words, one should be determined an algebra in the multi-dimensional vector space. It will be also made.

Algebraic adding complex numbers and vectors is connected with the propagation of transfers. The operations of multiplying and division consist of, as usually, "shift-adding" cycles.

The executed operations with the position codes of vectors are the algebraic adding, the vector, scalar, and special multiplying. The algorithms of these operations contain cycles of the algebraic adding codes of numbers and of vector code shift, in other words, they are easily realized by the technical ways. That may be used by constructing processors operating with the vectors as whole ones. Such a processor needs more simple algorithm for solving problems with vectors, while by given algorithms it works according shorter program and possesses heightened speed of response. It may be pointed out for estimating these quantities that, for instance, the program of vector multiplying vectors, given by three numbers, contains 6 multiplying operations and 3 subtraction operations.

In particular, any complex number is represented in the binary system of coding on the complex base $\rho = \sqrt{2}e^{j\varphi}$, where j - the imaginary unit.

If in the h-dimensional Euclidean space the algebra is defined, then the any point Z of this space is represented in the binary position numbering system on the base $\rho = \pm j\sqrt[h]{2}$, where j - the second vector of this space.

For an illustration and comparison we shall give the binary codes of numbers in all appointed coding systems, including of systems of coding on the real (positive and negative), complex and vector bases - see the table 1.

Table 1. Binary systems of coding.

ρ	φ	**Code of number (2)**	**Code of number (-2)**	**Code of number (-1)**
$\pm j\sqrt{2}$	$\pm\pi/2$	10100	100	101
$-1 \pm j$	$\pm 3\pi/4$	1100	11100	11101
$\sqrt{2}e^{j\varphi}$	$\approx \pm 166^\circ$	1010	110	111
-2	π	110	10	11
2	0	10		
$\pm j\sqrt[3]{2}$ **i,j,k**-unit vectors		1001000 (2i)	1000 (-2i)	1001 (-i)

The appointed coding systems have number merits by comparison with conventional system of coding on positive base. These merits are the next ones:

- It does not need to execute transformation from the direct code into the reverse one (or into the additional one) and backwards,
- The operations with the sign categories are excluded,
- Are simplified the rules of determination of overfilling by algebraic adding,
- Are simplified the algorithms of executing operations with the codes of a changeable length.

4. Coding geometrical figures

There is a great number of problems, in which the operations with the same names are done on the file of numbers. These problems concern, besides usual computational ones, problems of geometrical transforming figures too (they are represented by the set of points with discrete gradation of brightness). The problems arise by tracking after moving object of complicated and changeable configuration, in the designing problems needed sorting out various forms of a good, in the representing devices, and so on. Usually such the transformations are carried out by the computing coordinates of points of the initial figure. However, such a method requires much machine time, as computing coordinates is executed consecutively for all points and needs several operations for an each point (for example, it is necessary in fours of operations of adding and multiplying for computing new coordinates by the affined transforming a flat figure). The solving of the appointed problems may be considerably accelerated by the special coding file of codes.

The coding method is based on the following. Let us examine a binary tree, represented in the fig.1, and let us give a two-digit number (i,k) to each of its vertices (k - tier number, i- vertex number in the k-tier). By that, let us accept that numeration of the tiers passes from right to left, while the numeration of the vertices passes from up to down. We shall designate the vertex with even i-number by $\alpha_{i,k}$ and the vertex with odd i-number by the $\beta_{i,k}$.

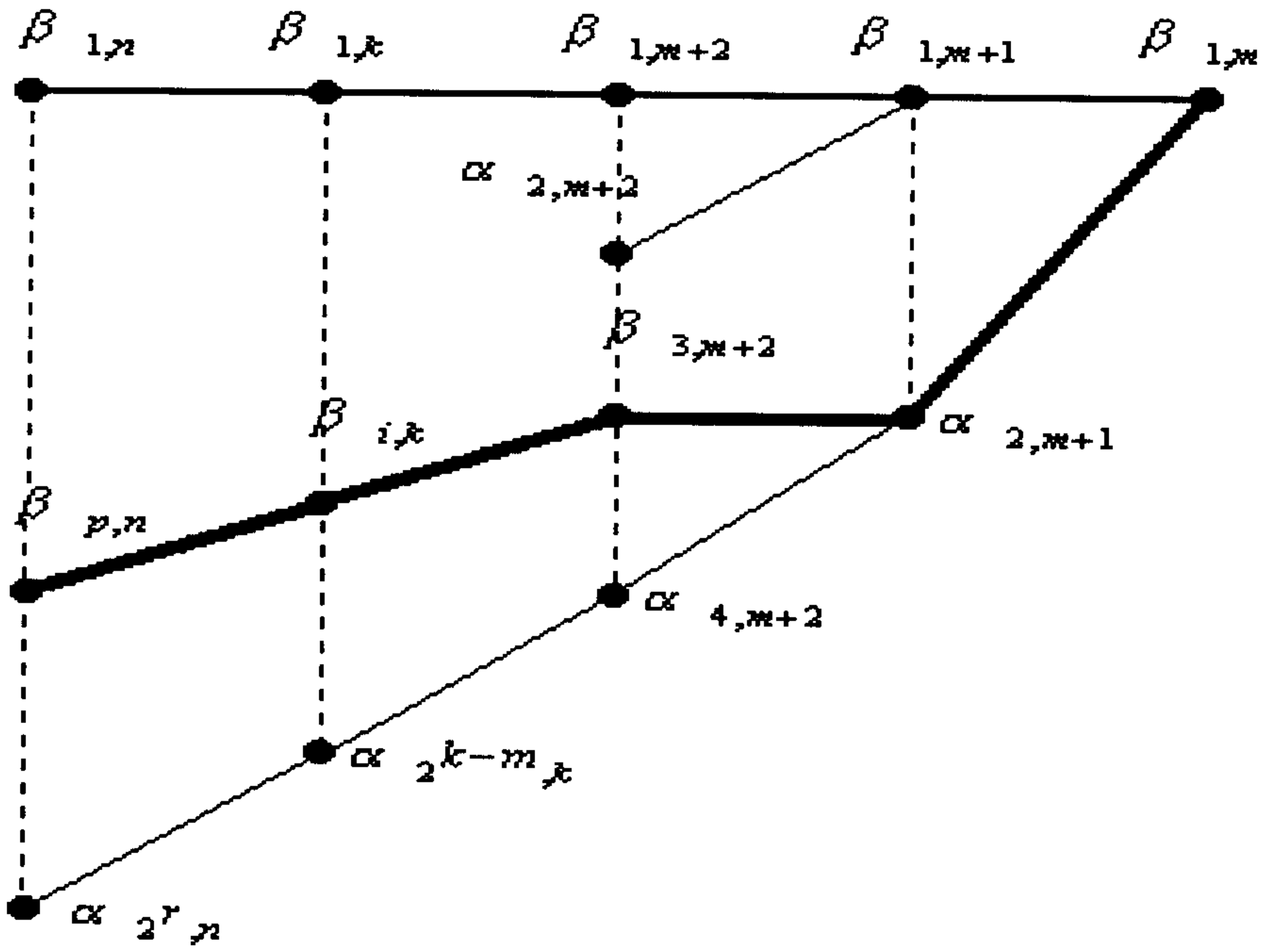


Fig. 1. Code of a tree.

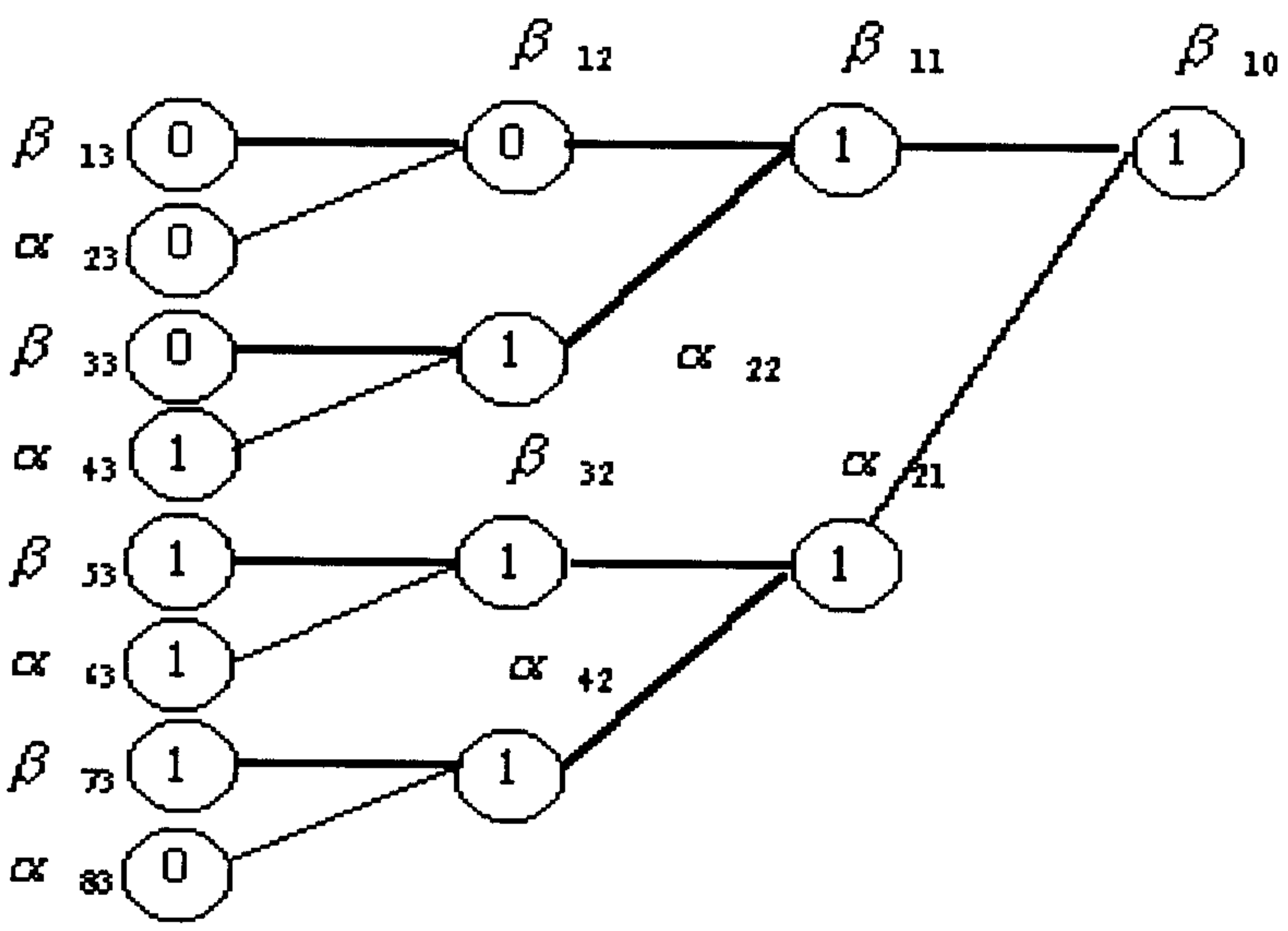


Fig. 2. Examples: a tree of binary categories.

We shall name the tree path joining vertices $\beta_{1,m}$ and $\beta_{p,n}$ as *p-path*. Apparently, each the *p*-path may be represented by a sequence of the symbols **a** and **b**. For example, in the fig.2 is picked out the *p*-path, to which corresponds the sequence

$$\beta_{p,n} \cdots \beta_{i,k} \cdots \beta_{3,m+2}\, \alpha_{2,m+1}\, \beta_{1,m}.$$

We shall name the each symbol $\alpha_{i,k}$ or $\beta_{i,k}$ of the sequence, representing some *p*-path, as *k-category of p-path* or *(i,k-category of the tree)*. If to give in correspondence for each the category of the *p*-path the 1 for the α-category or 0 for the β-category, then *p*-path may be represented by the binary code *K(p)*. In particular, *K(p).* = 0...0...010 in the fig.1. Let us agree now, that α = (0,1) and β = (0,1). We shall name the *p*-path as *open one* , if the quantity of all its categories is equal to 1, and as *closed one*, if the quantity at least of one its category is equal to 0. For illustration in the fig. 2 is represented a tree of the binary categories, in this tree are open 4 paths. It should to pay attention that for the open path, represented in the tree only by the single categories, corresponds the binary code, containing in a general case zero categories too.

We shall name the binary tree of binary categories, built by such a way and representing a set of the binary codes, as *geometrical code*, and the binary codes, composing it, - as *linear codes*.

In particular, if all paths of the tree are open, the tree represents all the binary codes of *r*-categories. The economy of the geometrical code increases in proportion to number of united linear codes. However, merits of the geometrical code are, mainly, that with it one may quite simply execute various operations. Therefore, there is a sense for using geometrical code in a case, when is available the great enough group of binary codes that it is necessary to execute the same, *group operations* (for example, multiplying all the codes by the same number). Besides, by the aid of geometrical code one can represent derivative figures, since exact codes of the vectors exist. Upon that, various transformations of these figures may be interpreted as operations with the geometrical code. These operations are connected with propagating *transfers* from the right, junior, tiers, to the left, senior tiers of the tree.

In the particular cases operations with the geometrical code are equivalent to following transformations of a figure: transfer, turning, pressing, shift of a figure, vector multiplying all vectors of the figure by the basic vector and so on.

Like the codes of three-dimensional figures, geometrical codes of multi-dimensional figures may be constructed, as far as in the ring of multi-dimensional vectors position numbering system of the binary codes also exist. Thus, by the aid of geometrical code one may encode multi-dimensional figure and carry out with it the affined transformations of this figure. One may use the last circumstance, for example, by constructing devices for recognizing images, as far as sins of recognizing objects are often invariant to a certain type of geometrical transformations.

5. Position codes of functions

It is supposed below, that a function is defined by the functional series. By that, a trivial way of coding functions could consist in setting coefficients of this series. However, such a way creates codes of large volume, moreover, it is not effective for multiplying.

Let us give some definitions.

The twofold sum of a form

$$F(x) = \Sigma_{k=0}^{n} \Sigma_{m=0}^{k} \alpha_{mk} R^{k} y^{k-m} (1-y)^{m} ,$$

where α_{mk} - real numbers,

R - positive integer,

$y = f(x)$ - some function of the argument x,

m, k, n - positive integers or zeros.

Such a sum is named as *decomposition of the function F(x) on the base y with the parameter R.*

The triangular matrix, composed from the quantities α_{mk} by such a way that each quantity α_{mk} belongs to k-column and m-row of this matrix, is named as *triangular code of the function F(x) on the base y with parameter R;* this triangular code is designated by the symbols TK$F(x)$ - see fig.3.

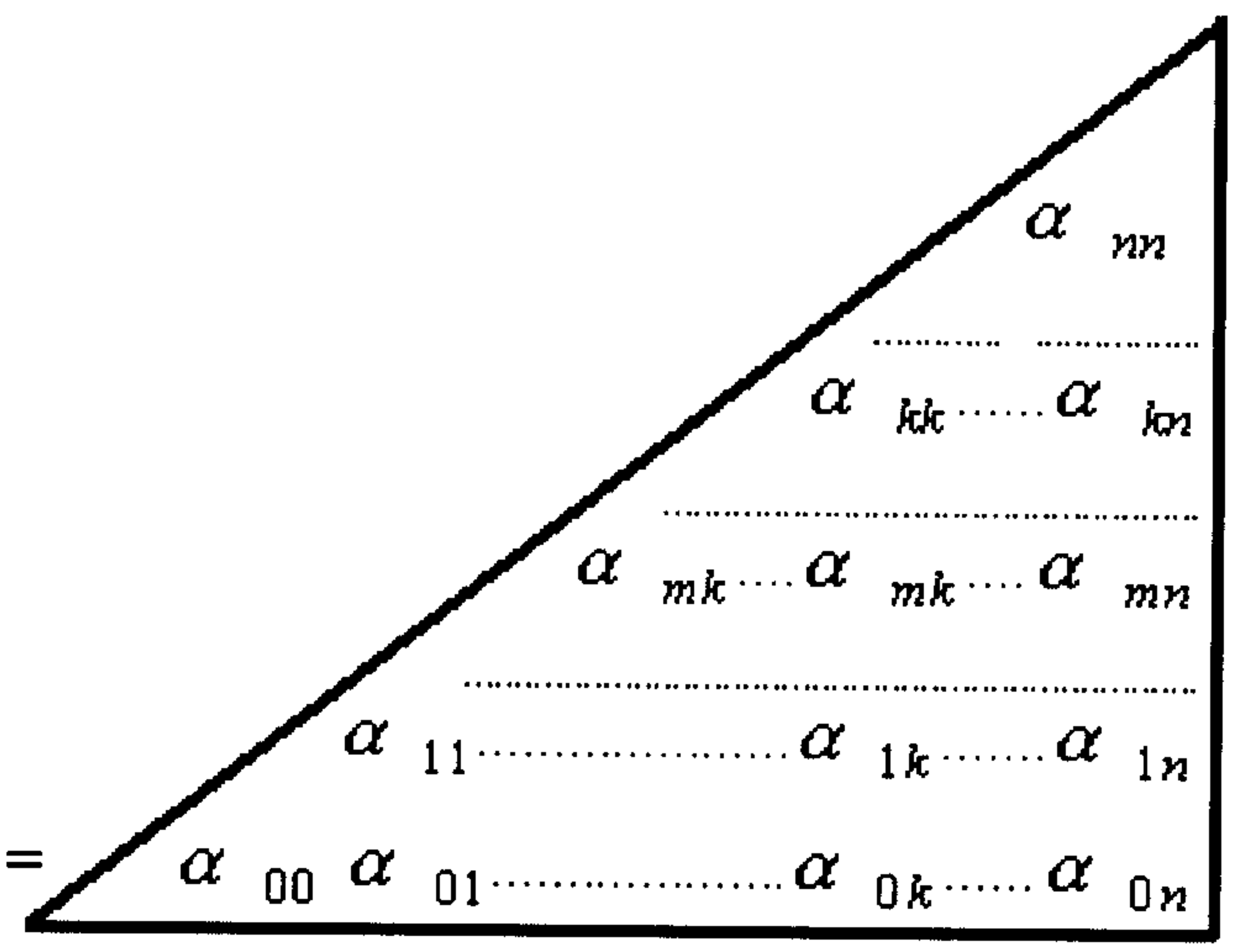
TK$F(x)$ =


Fig. 3. Triangular code of function.

Weight of the categories TK depends on its location. In this sense it may name triangular codes of the functions as position codes as well.

The triangular code is named as ternary one, if the $\alpha_{mk} \in \{-1,0,1\}$, and as the quaternary one, if the $\alpha_{mk} \in \{-2,-1,0,1\}$.

The algebraic adding TK is related with executing operation of the same name on a pair of the equally named categories of the added codes. These operations are related, like as for the usual codes, with the arising transfers in the senior categories. However, while in the usual R-th position codes making transfers is based on the correlation K*(R)* = 10, then in the given case the correlation

$$R\text{-TK}(R) = \begin{matrix} 1 \\ 01 \end{matrix}$$

is used.

Thus, in the triangular codes transfer propagates in two categories of a senior column (<u>the transfer "fork"</u>).

Besides that, with the triangular codes it may execute the operation of multiplying, division by a parameter (for elimination of overfilling), differentiation, integration, shift by an argument.

The considered above properties of the triangular codes do not depend on kind of a base. In particular, the argument $y = x$ itself may be taken as a base. However, it is necessary for application to choose such the bases that allow coding functions of the most general type. Therefore the considerated in detail are the codes on the base $y = Sin^2(x)$ by the aid of which the trigonometrical series and, in particular, Furrier series, may be represented. Moreover, codes of the trigonometrical series have a number of merits by comparison with the general case of coding functions.

The detailed considerating just the trigonometrical triangular codes is related with that the trigonometrical series occupy central place in the theory and practice of functional series. However, if to be distracted from this circumstance, then one more base of coding functions $Ch^2(x)$ may be pointed out; this base has non-less possibilities by comparison with the base $Sin^2(x)$, from the point of view of the formal theory of triangular codes.

Obviously, the properties of the hyperbolical triangular codes are analogous to properties of the trigonometrical triangular codes. The main merit of the hyperbolical triangular codes is that they allow a possibility of ***coding functions of the complex argument.***

There were considered above the codes of functions of one argument. Further generalization will lead us to the concept of codes of functions of several variables. By that, it is naturally to apply the next scheme of reasoning: the code of a number has linear structure; the code of a one argument function is a flat one. Apparently, the code of two arguments' function must be a three-dimensional one, the code of four arguments' function must be a five-dimensional one and so on.

For an example in the fig.4 the code of three arguments' function is represented.

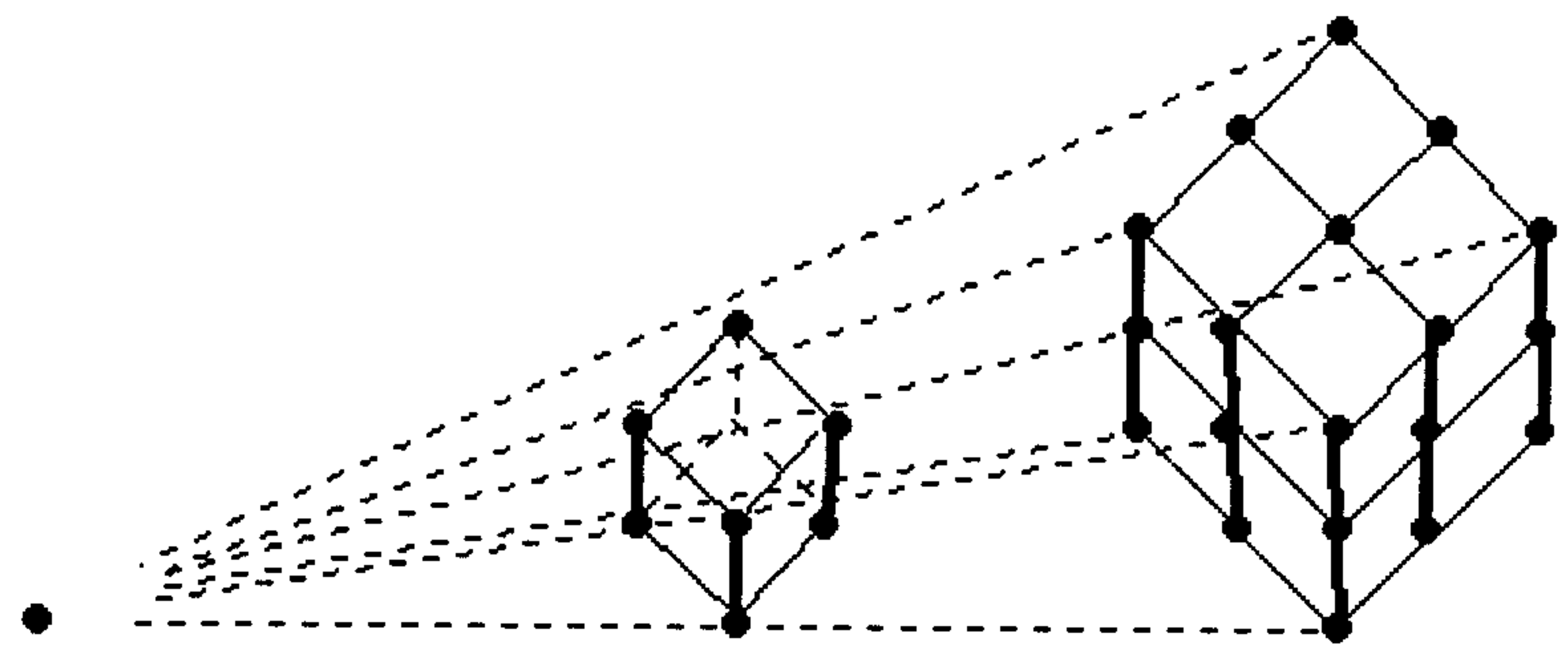


Fig. 4. Code of three arguments' function.

Processors for operations with complex numbers, multi-dimensional vectors, geometrical figures and functions

1. The developed components of processors

The next have been developed on present time

- the mathematical bases (completely)
- the architecture - composition of the units, interconnections between the units, word of length,
- system of the commands - for various variants of the processor,
- circuits of operational units (that execute individual operations) - completely),
- program models - for some operational units,
- algorithms of the intra-processor calculations (that are realized by a hardware),
- formalization of applied problems in the shape that is convenient for a formulation on the proposed processor - for some applied problems,
- analytical comparative analysis of speed.

The making experimental sample, its testing, working out technological documentation for beginning serial production should be a future stage of the developments.

The next things are original in it:

1. the set of machine operations,

2. operational schemes for executing these operations,
3. possibility of working with function as a whole
4. decrease of code length at given accuracy (by comparison with the code length of all coefficients of function series).

2. Architecture

The architecture of the computers, operating with complicated mathematical objects as a whole, and on the hardware realization of the offered algorithms has been developed.

The processor has a conventional architecture as a whole. The main difference is an arithmetical device. The latter has doubled word length, original operational units and a structure that may be rebuilt. That provides processor's working in several conditions:

- the conditions of conventional operations with real numbers of single word length,
- the conditions of conventional operations with real numbers of binary word length,
- the conditions of group operations with real numbers of single word length (by that, the operation of the same name is executed simultaneously with several real numbers),
- the conditions of operations with complex numbers,
- the conditions of operations with three-dimensional vectors,
- the conditions of operations with the plane geometrical figures,
- the conditions of operations with the spatial geometrical figures,
- the conditions of operations with the functions of one real argument,
- the conditions of operations with the functions of many real arguments,
- the conditions of operations with the functions of one complex argument,
- the conditions of operations with the functions of many complex arguments.

3. Machine operations

The proposed processors contain circuits for hardware executing the next operations:

<u>3.1 Machine operations with complex numbers:</u>

1. logical operations with binary codes,
2. arithmetical operations with real numbers,
3. arithmetical operations with complex numbers,
4. taking antilogarithm of complex number,
5. taking logarithm of complex number,
6. determining moduluses of complex number,
7. determining argument of complex number,
8. determining conjugate complex number,
9. trigonometrical functions of complex argument,
10. hyper-trigonometrical functions of a complex argument,

11. raising complex number to complex power,
12. taking logarithm of complex number to complex base,
13. coding pair of real numbers into complex number,
14. coding complex number into pair of real numbers,
15. special multiplying complex number by a pair of complex numbers (in electrical engineering calculations),
16. solving transcendental equations with complex variables (for example, in navigation calculations,
17. operations for a fast Fourier transform.

3.2 Machine operations with vectors:

1. scalar product,
2. vectorial product,
3. multiplication of a vector by number,
4. special multiplication n-dimensional vectors,
5. special component-by-component multiplication n-dimensional vectors,
6. coding and decoding.

3.3 Machine operations with figures:

1. algebraic addition of geometrical and linear codes,
2. multiplying of geometrical and linear codes,
3. dividing geometrical code on a linear code
4. rounding off geometrical code,
5. component-by-component multiplication of geometrical and linear codes,
6. centroaffine transformation of figure,
7. carry of figure,
8. coding and decoding of figure.

3.4 Machine operations with functions:

1. Algebraic addition ,
2. Division by a parameter,
3. Multiplying,
4. Coding and decoding,
5. Differentiating,
6. Integrating,
7. Inverting argument,
8. Displacing ordinates' axis,
9. Truncating.

4. Fields of application

The proposed processors may be used in the following fields:
- managing power systems,
- designing electrical engineering systems,
- treating images,
- underwater acoustics,
- tomography,
- laser holography,
- conform transformations,
- digital filters,
- 3D-Graphics,
- designing air- and hydraulic apparatus and systems,
- navigation and air defense,
- theoretical physics.

5. Merits

By solving problems containing operations with complicated mathematical objects the proposed processors have following merits (by comparison with the present-day processors):
- heightened (roughly by 10 and more times) speed,
- shortening volume of the programs,
- decreasing laboriousness of programming.

The processors' speed increases thank to that that complicated mathematical objects are represented by one binary code; moreover, any operation with the codes of complicated mathematical objects is only one machine operation. In contrast to that, any operation with complicated mathematical objects should be executed by usual processor with the aid of a sub-program.

Example of application.

On expediency of using specialized processors for operations with complex numbers (SPC) in the power systems.

The potential merits of **SPC** is a consequence of that

- the algorithms, represented in complex numbers, are the simpler and shorter ones,
- each operation with the complex numbers is equivalent to few operations with the real numbers,
- all of the elementary functions with complex numbers may be represented through potentiation (the operation is opposite to finding logarithms) and finding logarithms, which are realized on the **SPC** by the instrumental way (like division).

The majority of electrical engineering problems, solved by control of the power system, may be represented in the form of operations with complex numbers. By that, it is shortened several times a number of operations for realizing given algorithm. Indeed, each an operation with the complex numbers is equivalent to several operations with the real numbers. If the executing time for the operation with complex numbers is equal to executing time for the operation with real numbers (by using **SPC**), then computer speed increases several times too.

It is known, moreover, a method of representing algorithms, operating with the complex numbers, in the form of operations with the so called 'cells', that is, with fours of the real numbers or with pairs of the complex numbers. Upon that, a number of the operations are shortened even more. Operations with the cells are also executed on the **SPC** for the time of one operation with the real numbers. On the whole, as it is shown by the analysis, computer speed increases 10 times by solving electrical engineering problem on **SPC**.

It would to be unreasonable to propose a replacement of existing computers on the **SPC**, since, besides of electrical engineering problems in the power systems, a great number of other problems are solved. Therefore, it is proposed to use **SPC** as co-processor. We shall consider this question more in detail on the example of flow distribution problem.

It is known that flow distribution problem should be solved many times when carrying out any electrical engineering calculation in the power system. That takes a great part of the machine time. Moreover, duration of solving just this problem practically does not give a possibility for carrying out technological calculations in the efficient conditions (with a small periodicity).

In connection with that it seems expedient, in addition to a basic computer, to use computer-satellite destined exclusively for solving flow distribution problem. Actually, such a computer could appear in the role of "instrumental sub-program", which is carried out simultaneously with the other calculations. One can achieve even more accelerating calculations by using computer with **SPC** as computer-satellite.

The problems should to be expediently solved by the proposed computers are the following ones: electrical engineering calculations, anti-air-defence problems, transforming images, tomography, laser holography, underwater acoustics, etc. It has been shown that proposed processors *solve* such the problems *much faster* (10 and more times as fast). A *laboriousness of developing* appropriate programs essentially decreases too.

The proposed processor has a traditional architecture. The difference is in constructing arithmetical device. The next things are original in it:

1. the set of machine operations,
2. operational schemes for executing these operations,
3. possibility of working with function

as a whole
4. decrease of code length at given accuracy (by comparison with the code length of all coefficients of function series).

Operational units

6. Operational units for addition of linear codes

Complex numbers and vectors are represented by linear codes. Under addition of linear codes multi-valued carries (in contrast to the ordinary numbers when only two-valued carries) arise. It is proposed two types of carries' propagation's schemes. The first scheme is depicted on the Fig. 5. All carries from each category are transmitted to a next category.

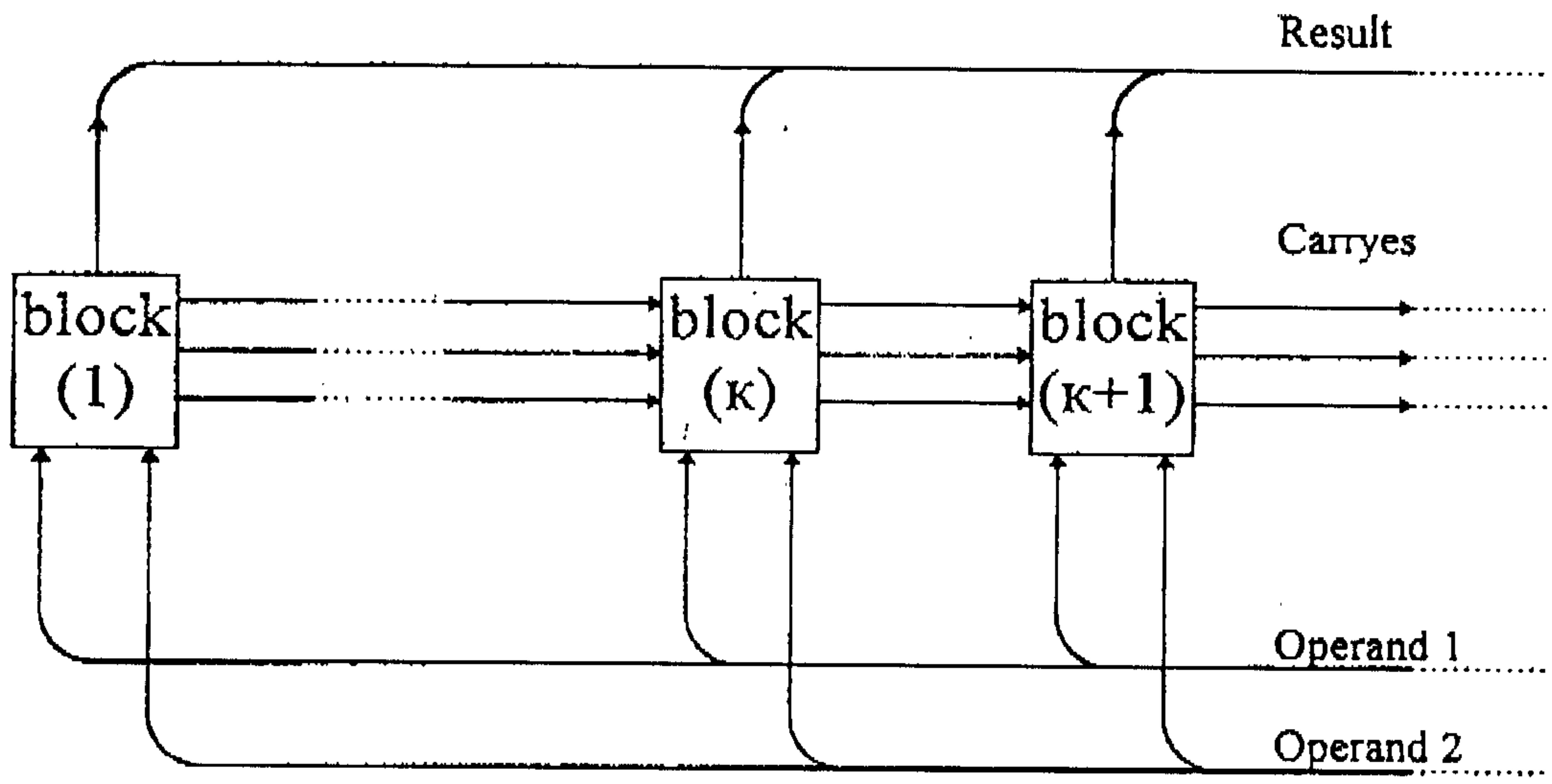


Fig. 5

The second scheme is depicted on the Fig. 6. In it the carries from each category are transmitted to several next categories. At that, the carries from several previous categories enter each category. The schemes of carries forming in each category depend on the accepted for the given codes basis of the coding system.

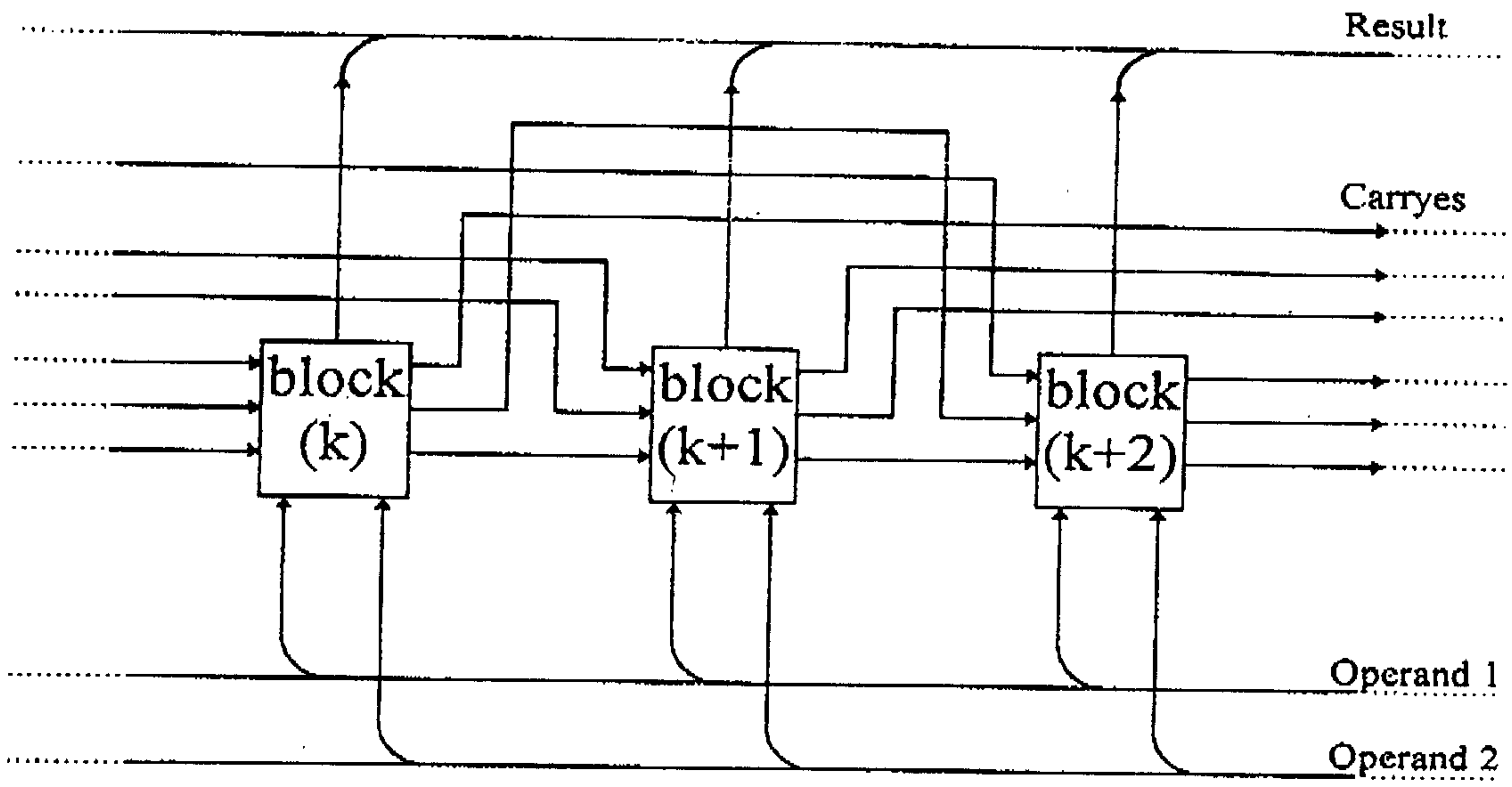


Fig. 6

7. Operational units for addition of geometrical codes of figures

The Fig. 7 depicts the scheme of the carries' propagating in the process of addition of geometrical codes of the plane figures. The relations between categories in the tree of the geometrical codes are shown by dotted lines while the carries are depicted by solid lines. Thus, from each category a carry is transmitted to four categories located in the second level from the given level.

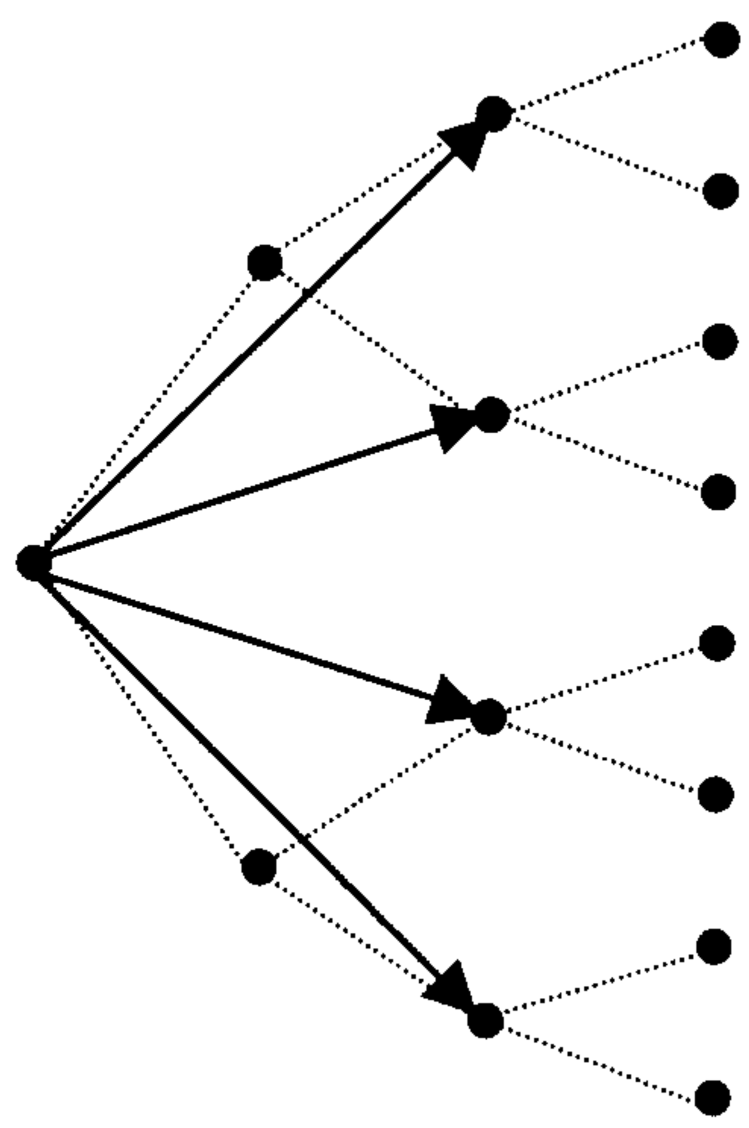


Fig. 7.

In the process of addition of geometrical codes of the three-dimensional figures a carry from the given category is transmitted to sixteen categories located in the fourth level from the given level.

8. Operational units for addition of codes of functions

The scheme of carries in the triangle codes of functions of one argument is depicted on the Fig. 8. In this code the carry from each category is transmitted to two categories of the next level. Accordingly, carries from two categories of the previous level are transmitted to the given category.

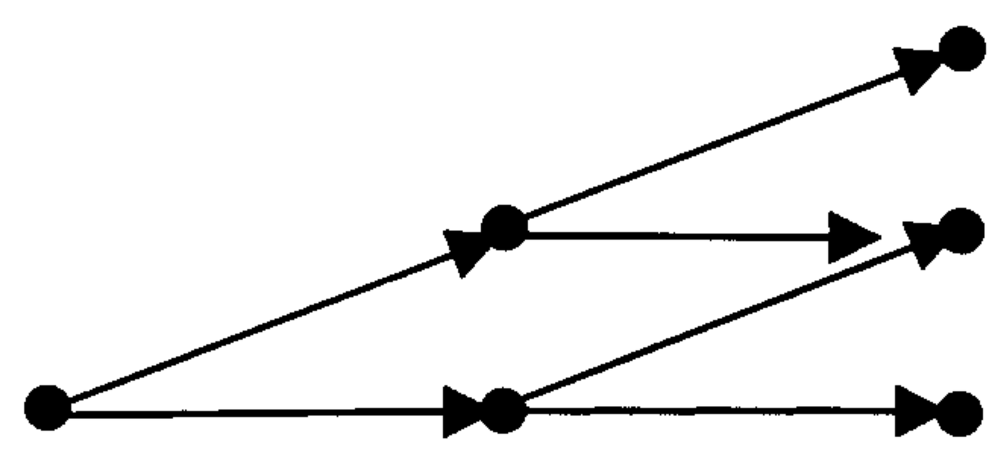


Fig. 8.

In the process of addition of codes of functions of many arguments the carry from each category is transmitted to several categories of the next level. In the process of addition of the functions of two arguments carries propagate as is shown on the Fig. 9.

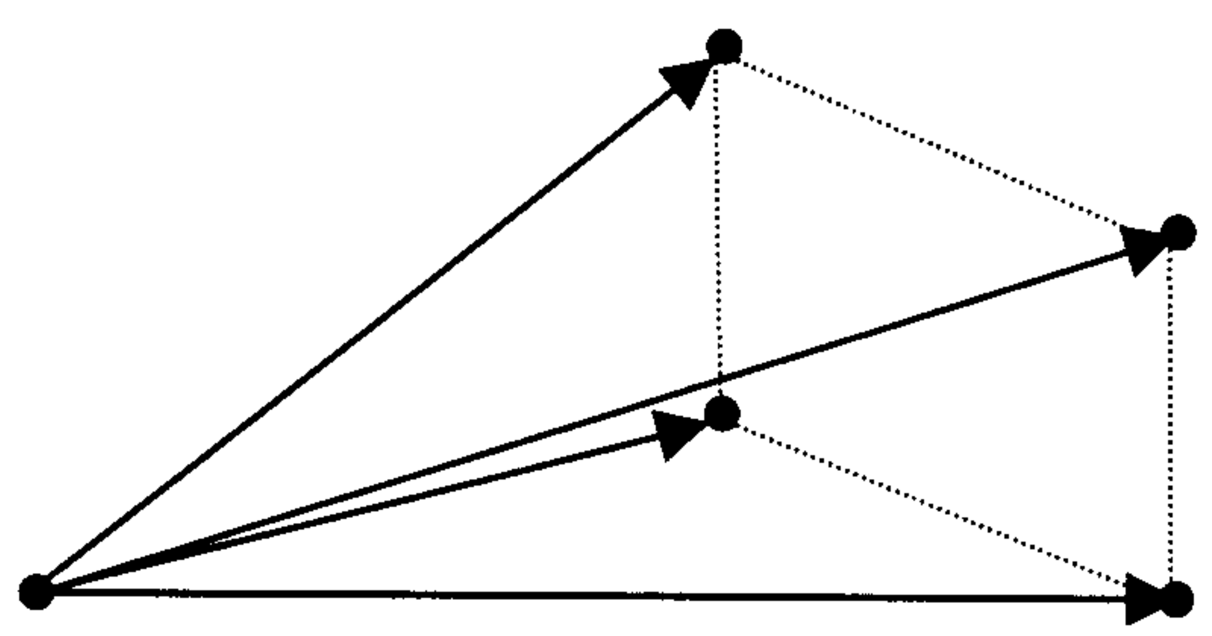


Fig. 9.

In the process of addition of the functions of three arguments carries propagate as is shown on the Fig. 10.

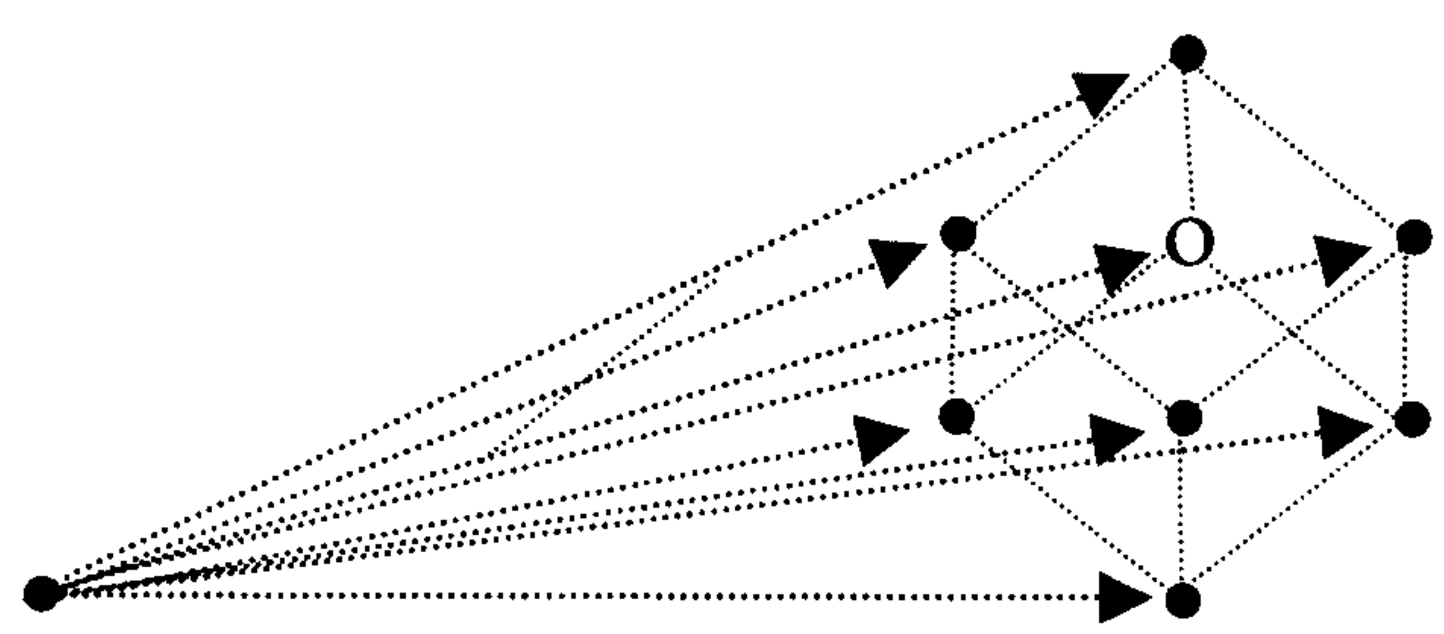


Fig. 10.

9. Operational units for multiplication

The method of the positional coding has such an advantage that the process of multiplication of codes may be executed by means an interchange of addition and shifting. The shifting is equivalent to multiplication of codes on the basis of the coding system. This rule extends on codes of any objects. Thus, multiplication of different objects differs only in the way of execution of the shifting.
The linear codes are shifting along the codes themselves. The geometrical and triangle codes are shifting in the plane of the codes themselves. The pyramidal codes are shifting in the space of the codes themselves.
There are three different types of multiplication of the codes of vectors:

- scalar;
- vectorial;
- per components.

For the geometrical codes operations of multiplication of a figure on a complex number are equivalent to the affined transformation of the figure.

10. Operational units for long operations with complex numbers

The long operations with complex numbers include:

- division;
- logarithming;
- potentiating.

These operations are executed by interchanges of subtractions, shifting and comparison. The schemes of subtraction and shifting are described in the previous chapters. The process of comparison consists in the comparison of the numbers of the senior categories of the compared codes. Subtrahend has the different sense in the different operations. In the process of division the subtrahend is a divisor. In the process of logarithming and potentiating the subtrahend is a function from the extent of the base of the coding system.
All the rest functions of the complex number may be expressed through the above-listed operations.

The sample processor: complex co-processor

1. Overview

The complex co-processor is a powerful mathematical processor intended to be used as a DSP co-processor in a PC environment or in other mathematically intensive applications.

A typical PC-based complex co-processor is shown below:

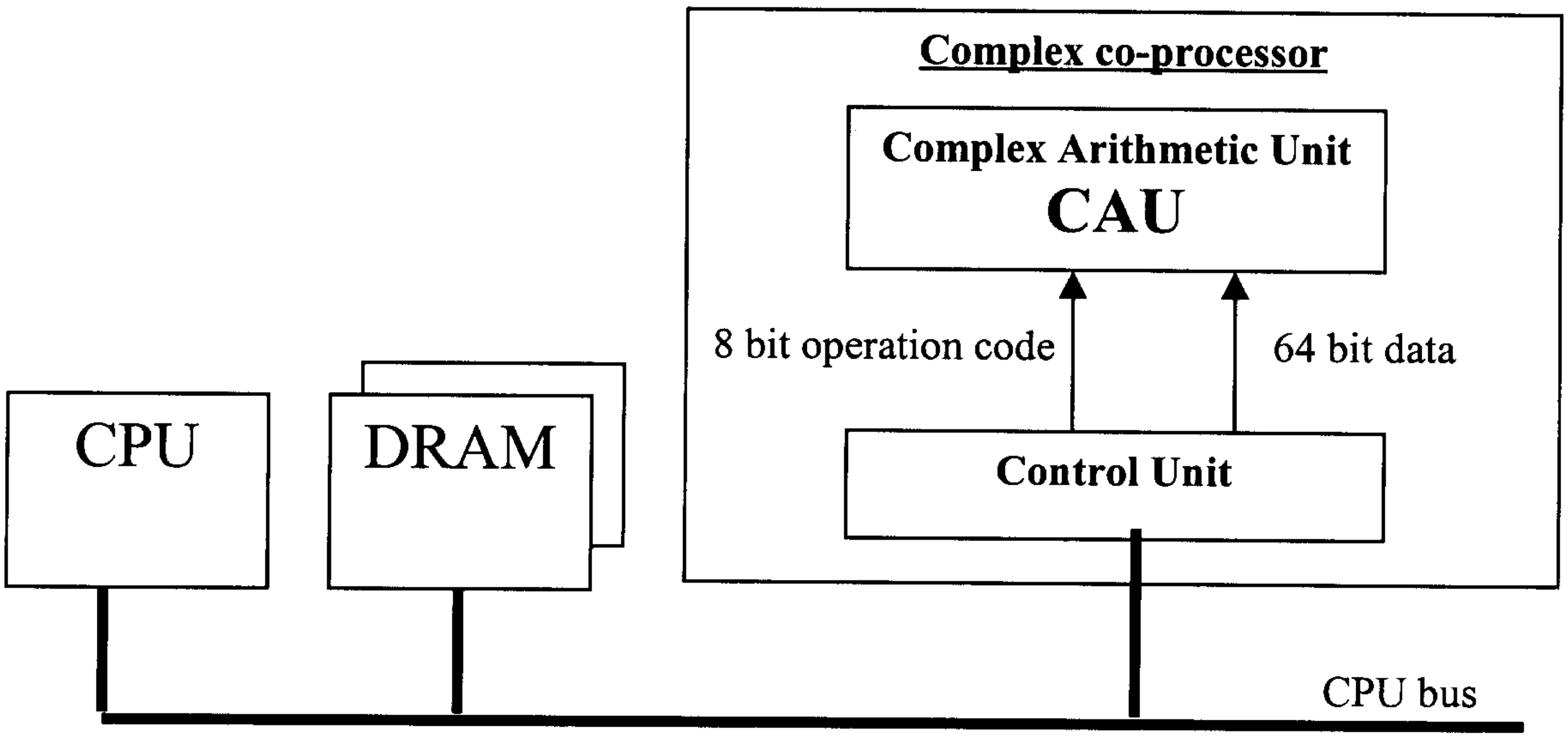


Fig. 11. The System.

2. Complex Arithmetic Unit (CAU)

CAU scheme is shown below:

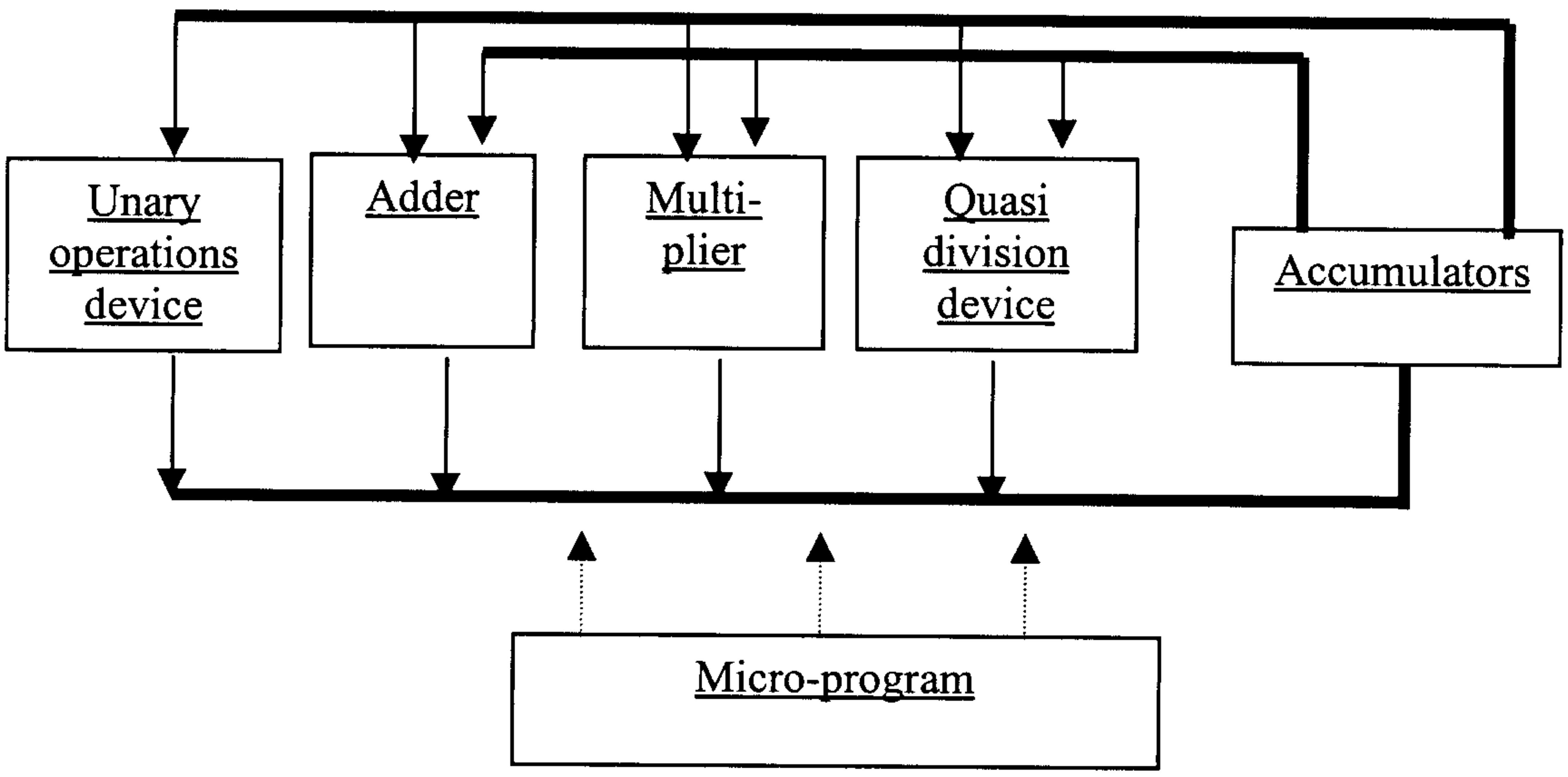


Fig. 12. CAU

2.1 Number Representation

The traditional real floating-point number representation $(-1)^m \cdot (1.M) \cdot 2^{(-1)^e E}$

is shown below:

0 1 2 8 9 31

m	e	Exponent E	Mantissa $1.M$

Fig. 13. Traditional real floating-point number representation

The complex floating-point number representation $\cdot(1.Z)\cdot(-1+j)^{\omega}$
is shown below:

0 9 10 63

Exponent - ω	Mantissa – $1.Z$

Fig. 14. Complex floating-point number representation

The Logarithmic complex number representation $\cdot(1.Z)\cdot(-1+j)^{\omega}$ is shown below:

0 9 10 63

Exponent - ω	Natural Logarithm of Mantissa - $\ln(1.Z)$

Fig. 15. Logarithmic complex number representation

Here complex number *(1.Z)* is represented in number system on basis *(-1+j)*, where *j* – imaginary unit, real number ω is represented in number system on basis (-2).

2.2 Device for unary operations

This device performs the following operations.

1. Coding from "Traditional real floating-point number representation" to "Complex real floating-point number representation",
2. Decoding from "Complex real floating-point number representation" to "Traditional real floating-point number representation",
3. Multiplication with *m = -1, j, -j, 2, -2, (1+j), (-1+j), (1-j), (-1-j),*
4. Addition to *m = 1, -1, j, -j, 2, -2, (1+j), (-1+j), (1-j), (-1-j),*
5. Determination of *Re(A), Im(A), -Re(A), -Im(A),*
6. Determination of positive and negative conjugate numbers.

2.3 Adder

This device performs addition of complex numbers upon the following formulas.

$C = A + m*B$, where $m = 1, -1, j, -j, 2, -2, (1+j), (-1+j), (1-j), (-1-j)$.

2.4 Multiplier

This device performs multiplication with two categories simultaneously:

1. multiplication of complex numbers,
2. centroaffine transformation,
3. multiplication of a cell with a complex number (for solution of linear equation systems with complex numbers, for example, in electric power tasks).

2.5 Quasi-division device

This device performs the following operations with complex numbers that are similar to division by algorithm.

1. Division.
2. Exponentiating $\exp(Z)$
3. Taking logarithm $\ln(Z)$
4. Coding from "Complex real floating-point number representation" to "Logarithmic complex number representation",
5. Decoding from "Logarithmic complex number representation" to "Complex real floating-point number representation",
6. Computing of elementary functions

$$\exp(jZ), \exp(j\tilde{Z}), \exp(-jZ), \exp(-j\tilde{Z}),$$

$$\arg(Z), \sqrt[m]{Z}, {}^{-}\sqrt[m]{Z}, \sqrt[m]{\tilde{Z}}, {}^{-}\sqrt[m]{\tilde{Z}}, \tilde{Z}, |Z|, |\tilde{Z}|, \sqrt{Z}, \sqrt{\tilde{Z}}$$

2.6 Device with micro-programmable architecture

This device performs operations with complex numbers upon micro-programs.
It is important to say that it is possible to add new micro-programs or modify any of the existing micro-program. The following micro-programs are provided.

1. Computing of direct and inverse trigonometrical and hypertrigonometrical functions.
2. Computing of transcendental equations with complex argument, for example, the following equations:

$$Z + A \cdot \exp(j \cdot (\arg(Z))) = B,$$

where A, B are the given complex numbers, Z is the unknown complex number. One of the persecution problems is described with this equation.