prace naukowo-badawcze

Instytutu Maszyn Matematycznych

A. WITKOWSKI: SRT division with a normalized divisor

S. MAJERSKI: Dzielenie liczb binarnych z redundancyjnym zapisem kolejnych reszt i ilorazu

Zjednoczenie Przemysłu Automatyki i Aparatury Pomiarowej "MERA"

Instytut Maszyn Matematycznych

prace naukowo-badawcze Instytutu Maszyn Matematycznych

w zeszycie zamieszczono:

Warszawa 1980

Adres Redakcji: Instytut Maszyn Matematycznych
Branżowy Ośrodek INTE
ul. Krzywickiego 34, 02-078 Warszawa
tel. 28-37-29 lub 21-84-41 w. 244

Prace naukowo-badawcze Instytutu Maszyn Matematycznych

SRT DIVISION WITH A NORMALIZED DIVISOR

by Andrzej WITKOWSKI

A method of the quotient digit selection for SRT division is presented. This paper shows that the precision of inspection of the divisor and partial remainder, which is required to select quotient digit, depends on the range of the divisor. The optimal range of the divisor is found /the range for which the precision of inspection of the divisor and partial remainder is the lowest/. This range is: $1-1/r \leq |D| \leq 1$. The way of reducing the divisor to the optimal range: $1-1/r \leq |D| \leq 1$ is presented. The proposed method permits to implement high speed division devices.

INTRODUCTION

This paper presents the technique of the quotient digit selection which is specially suited for implementation in computers executing operations: a·b or a·b+c with a high speed i.e., comparable with the speed of operation a+b.

High speed division can be effected choosing sufficiently large r /radix of the division/. The point is that quotient digit q_i is a function of divisor D and partial remainder rA_{i+1}. Generally speaking, q_i depends on all bits of words d and ra_{i+1}; words d and ra_{i+1} represent numbers D and rA_{i+1}.

However, it turns out that the quantity of bits to be inspected can be radically reduced. Such a solution was proposed by D.E. Atkins [1]. He showed that, using SRT algorithm, q_i can be calculated using only several bits of d and ra_{i+1}. He also showed that, the greater r, the more bits must be inspected.

This paper tries to develop the Atkins method. It shows the way of calculating q_i which considerably decreases the precision of inspection of d and ra_{i+1} compared with the Atkins' method.

For the method presented here, a high speed device executing operation a·b or a·b+c is needed. As long as the operation a·b or a·b+c lasted much longer than the operation a+b, this method was applicable only occasionally. Nowadays, when the operation a·b or a·b+c is effected by LSI, integrated circuits its applicability can be much wider.

1. DEFINITION OF THE DIVISION ALGORITHM

The present section defines the process of the division. A denotes the dividend, D the divisor. Dividend A and divisor D are represented by words a and d. In the division process quotient Q and remainder A_{-n} are being looked for.

The following conditions have to be fulfilled:

- quotient Q is represented by m digits q_i

$$Q = \sum_{i=-1}^{-m} q_i \cdot r^i, \qquad /1/$$

the digits q_i fulfil the condition:

$$|q_i| < r; \quad r \geq 2 \qquad /2/$$

- quotient Q, remainder A_{-m}, dividend A and divisor D /D≠0/ are related by the following relationships:

$$\frac{A}{D} = Q + \frac{A_{-m} \cdot r^{-m}}{D} \qquad /3/$$

$$|A_{-m}| \leq k \cdot |D| \qquad /4/$$

where k is a positive constant,

- digits q_i are determined in such a way that:

$$|A_i| \leq k \cdot |D| \qquad /5/$$

where A_i denotes the i-th partial remainder:

$$A_i = rA_{i+1} - q_i \cdot D \qquad /6/$$

$$A_0 = A \qquad i = -1, \ldots, -m$$

Note that in the division process defined above, digits of the quotient Q are integers. Quotient Q can be obtained in the redundant form i.e., in the form in which the number of digits q_i is greater than r. Obviously, in the division process mentioned above, Q can also be found in a univocal form.

Note also that quotient Q approximates the real quotient A/D with an error not greater than $k \cdot r^{-m}$ for, as it follows from /3/ and /4/:

$$\left| \frac{A}{D} - Q \right| \leq k \cdot r^{-m} \qquad /7/$$

According to /5/ and /6/, q_i is defined basing on A_{i+1} and D, so that:

$$r \cdot \frac{A_{i+1}}{D} - k \leq q_i \leq r \cdot \frac{A_{i+1}}{D} + k \qquad /8/$$

It follows from /8/ that for a given value of A_{i+1}/D, digits q_i can assume all integer values from the interval $\Delta q = \langle rA_{i+1}/D-k,\ rA_{i+1}/D+k \rangle$. The length Δq of this interval is

$$\Delta q = 2k$$

In order to make the division process feasible, the interval of the length of Δq must contain at least one integer. Therefore, its length cannot be smaller than 1:

$$\Delta q \geq 1$$

$$\text{or} \quad k \geq 1/2 \qquad /9/$$

Note that dependence /8/ does not define the value of q_i in a univocal way. For a typical value of k = 1, within the interval of the length of $\Delta q = 2$, two or three integers being the digits q_i can be found. For instance: when $rA_{i+1}/D = 1$ the proper digits q_i will be 0, 1 and 2.

2. SRT ALGORITHM

In this section the SRT algorithm will be described and one of its graphic interpretation will be given. SRT algorithm was elaborated by Sevensy, Robertson and Tocher; see [4],[6]. The feature of this algorithm is that quotient Q is formed in the redundant form. In the SRT algorithm for r > 2 digits q_i can assume values from the following set:

$$q_i \in \{-n, -(n-1), \ldots 0, \ldots, n-1, n\} \qquad /10/$$

where n is an integer so that:

$$1/2\cdot(r-1) \leq n \leq r-1 \qquad /11/$$

The restriction /11/ on the set of digits q_i results from the following conditions:

- at least r different digits are necessary to represent number of radix r. Since quantity of the set $\{n, \ldots, 0, \ldots, n\}$ is 2n+1, the condition $2n+1 \geq r$ or $n \geq 1/2\cdot(r-1)$ must be satisfied
- inequality $n \leq r-1$ results from the assumption /2/.

Note that, if $n > 1/2\cdot(r-1)$, the quotient Q will be represented in the redundant code. Obviously, the greater n, the greater the redundancy of the quotient code.

Now, the redundancy of the quotient Q will be related with k taken from /4/. According to /6/, for a given q_i, A_i/D is a linear function of A_{i+1}/D, thus A_i/D has a maximum when A_{i+1}/D is maximum. Since $(A_i/D)_{max} = (A_{i+1}/D)_{max} = k$ /see /5// for $q_i = n$ we have:

$$k = r\cdot k - n$$

or

$$k = \frac{n}{r-1} \qquad /12/$$

From /11/ and /12/ the interval of the variation of k can be determined:

$$1/2 \leq k \leq 1 \qquad /13/$$

Dependence /13/ is in agreement with the dependence /9/ obtained above.

Value k defined by /12/ is a measure of the redundancy of the quotient code; the smaller the difference 1-k, the greater the redundancy. In the next section the effect of k on the price of the device calculating q_i will be considered.

To end this section, one of the graphic interpretations of the division process will be demonstrated. It is called the A-D plot. This plot will be very helpful for us in the next section.

A-D plot illustrates the connection between the divisor D, the partial remainder rA_{i+1}, the coefficient k and the digit q_i. These connections result from the dependence /5/ and /6/. Equation /6/ may be rewritten as

$$rA_{i+1} = A_i + q_i\cdot D$$

Note that for given q_i quantity rA_{i+1}, as a function of D, has a maximum when A_i is maximum. Since $|A_i| \leq k\cdot|D|$, $(A_i)_{max} = k\cdot D$ and:

$$(rA_{i+1})_{max} = (k+q_i)\cdot D \qquad /14/$$

Likewise, for given q_i rA_{i+1}, as a function of D, has a minimum when $A_i = (A_i)_{min} = -k\cdot D$ and

then:

$$(rA_{i+1})_{min} = (q_i - k) \cdot D \qquad /15/$$

In order to obtain the A-D plot, lines /14/ and /15/ should be drawn on the $/rA_{i+1}, D/$ plane with a digit q_i as a parameter assuming values from the set of the quotient digits /10/. For given q_i, the area between lines $/rA_{i+1}/_{max}$ and $/rA_{i+1}/_{min}$ is called the q_i area. This area is a set of all pairs $/rA_{i+1}, D/$ for which the proper quotient digit is q_i. Obviously, there are as many areas as quotient digits. As an example the A-D plot for r = 4 and k = 2/3 is shown /Fig. 1/.

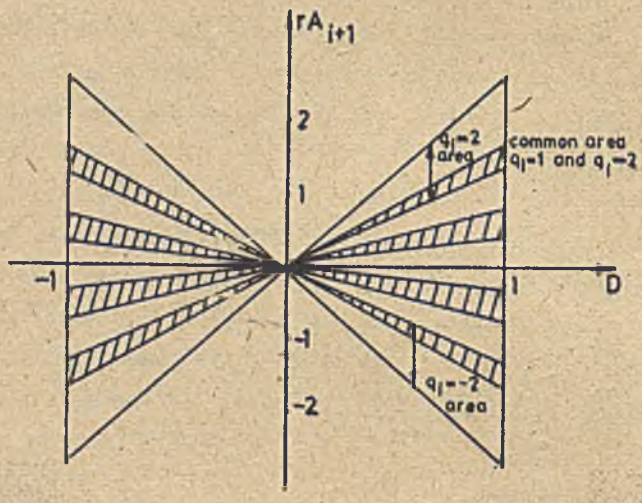


Fig. 1. A-D plot with r = 4 and k = 2/3

As it was stressed in the previous section, digits q_i could be chosen arbitrarily - see /8/. It can be noticed that in the A-D plot the q_i areas overlap in such a way, that some pairs $/rA_{i+1}, D/$ determine points which belong to two q_i areas. For these pairs both digits q_i are correct. The set of points $/rA_{i+1}, D/$ for which two digits q_i are correct will be called a common area of q_i and q_{i+1}. In the Fig. 1 these areas are lined.

Note that the A-D plot is symmetrical relative to each of the axes of the co-ordinate system. It results from the fact that the equations /14/ and /15/ define lines passing through the centre of the co-ordinate system $/rA_{i+1}, D/$. Therefore, in what follows, the A-D plot made in one quarter of the co-ordinate system $/rA_{i+1}, D/$ will be used.

In terms of the A-D plot, the process of the formation of Q can be defined as follows: the given values of rA_{i+1} and D define a point in one of the q_i areas. Digit /digits/ q_i corresponding to this area, it the next quotient digit and it can be used to calculate next partial remainder according to /6/.

/

3. METHOD OF THE q_i SELECTION

In this section the method of the q_i selection will be given. Both the way of determining of q_i and the precision of inspection of ra_{i+1} and d, in order to determine q_i, will be described. Furthermore, the block scheme of the device determining digits q_i will be suggested.

In the method presented the following assumptions must be fulfilled:

1. Quotient is being formed in a redundant form according to the SRT algorithm.

2. Divisor D is initially reduced to the form:

$$|D| \geq D_k \quad \text{and} \quad |D| < 1,$$

$$\text{where} \quad D_k > 0 \qquad /16/$$

The method presented will be illustrated by A-D plots. Referring to them, the term "the division field" will be used. This term denotes area comprised between lines: $D = D_k$, $D = 1$, $rA_{i+1} = (k+n) \cdot D$ and $rA_{i+1} = (k-n) \cdot D$.

The idea of the method.

The "division field" can be divided into equal rectangles of the sides ΔrA and ΔD in such a way that each of the rectangles will be entirely included in one of the q_i areas. In the Figs 2 and 3 two exemplary divisions of the "division field" are demonstrated. In both examples each rectangle is entirely included in a certain q_i area. All the points of the rectangle which is entirely included in the $q_i = j$ area correspond to next quotient digit equal to j. Remember that co-ordinates of these points represent quantities rA_{i+1} and D.

Let the division of the A-D plot into rectangles be established. Formation of the quotient Q consists in locating pairs $/rA_{i+1}$, D/ in the proper rectangle and reading digits q_i corresponding to this rectangle.

Since in this procedure rA_{i+1} and D must be located within intervals of the length of ΔrA /height of the rectangle/ and ΔD /width of the rectangle/, the words ra_{i+1} and d can be inspected with a limited precision. Precision of the inspection of ra_{i+1} and d depends on quantities ΔrA and ΔD; the greater they are, the less the is required precision the smaller the number of bits of words ra_{i+1} and d should be inspected/. Further considerations are intended to establish ΔrA and ΔD as large as possible /the precision of examination as low as possible/.

The following observations can be made:

1. For given values r, k, D_k the A-D plot can be divided into rectangles in many ways. These ways differ from each other in quantities ΔrA and ΔD. It is illustrated in the Figs 2 and 3. For the Fig. 2 the rectangles are greater and the precision of the inspection of ra_{i+1} and d can be lower.

2. Maximum values of ΔrA and ΔD are limited by the size of common areas q_i and q_{i+1}. Namely sectors of the length of ΔrA and ΔD /vertical and horizontal sides of the rectangles, respectively/ must be entirely included in the common areas. Otherwise the rectangles with the sides ΔrA and ΔD could not be included in the q_i areas. Let $\Delta rA_i'$ denotes the height of the common area of q_i and q_{i+1} /distance between lines /14/ and /15/ along

rA_{i+1} axis/, and $\Delta D_i'$ denotes the width of common area of q_i and q_{i+1} /distance between lines /14/ and /15/ along D axis/. Now, if we want the rectangle with the sides ΔrA and ΔD to be entirely included in the q_i areas, the following conditions must be fulfilled:

$$\Delta rA \leq \Delta rA_i' \qquad /17/$$
$$\Delta D \leq \Delta D_i'$$

Condition /17/ must be fulfilled for all q_i areas, and the restriction for the size of rectangles has finally the following form:

$$\Delta rA \leq \Delta rA' \qquad /18/$$
$$\Delta D \leq \Delta D',$$
$$\text{where } \Delta rA' = \min\{\Delta rA'_{/n-1/}, \ldots \Delta rA'_{n-1}\}$$
$$\Delta D' = \min\{\Delta D'_{/n-1/}, \ldots, \Delta D'_{n-1}\}$$

3. As follows from the observation 2^0, the most interesting are the rectangles in common areas. In the Figs 2 and 3 the thick line separates rectangles which define different quotient digits. Note that this line forms a broken line of the characteristic step-like structure; height and width of every step is a multiple of quantities ΔrA and ΔD.

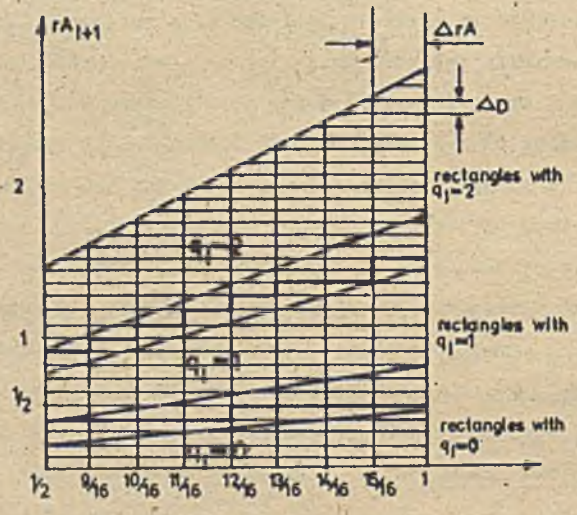


Fig. 2. A-D plot with r = 4, k = 2/3

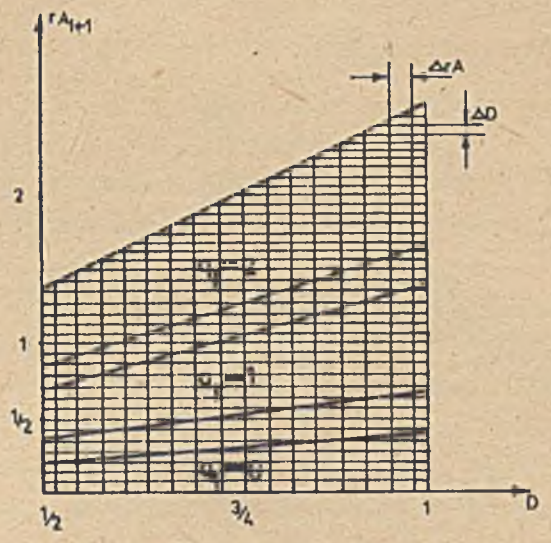


Fig. 3. A-D plot with r = 4, k = 2/3

4. It is worth while to point out the dependence of $\Delta rA'$ and $\Delta D'$ on D_k. The greater D_k, the larger size of rectangles covering the A-D plot. It is illustrated in the Figs 4 and 5. Both of them were drawn for the case r = 4, k = 1. For the Fig. 4 the "division field" is limited from the left side with the line D = 1/2 and for the Fig. 5 with the line D = 3/4. It is easy to note, that for the case of the Fig. 5 rectangles are larger and the required precision of inspection of ra_{i+1} and d is lower. For the Fig. 4, in order to determine q_i, 4 bits of the words ra_{i+1} and d should be inspected, while for the Fig. 5, 3 and 1 bits, respectively. Let $\Delta rA1'$ and $\Delta D1'$ denote the height and width of the common area for $D1_k$, and $\Delta rA2'$, $\Delta D2'$ the height and width of the common area for $D2_k$. One can easily demonstrate that:

$$\frac{D2_k}{D1_k} = \frac{A2'}{A1'} = \frac{D2'}{D1'} \qquad /19/$$

Therefore, the height and width of common areas are directly proportional to D_k. To summarize the range of variation of D influences essentially the precision of inspection of ra_{i+1} and d, hence the price of devices which determine digit q_i.

As follows from the observation 2^0, the size of rectangles is defined by $\Delta rA'$ and $\Delta D'$. Now, the influence of parameters k and D_k on $\Delta rA'_j$ and $\Delta D'_j$ will be considered. For a given common area defined by digits $q_i = j$ and $q_i = j-1$, the quantities $\Delta rA'_j$ and $\Delta D'_j$ are expressed by the following formulae /see Fig. 6/:

$$\Delta D'_j = D'' - D' = \frac{rA''_{i+1}}{-k+i} - \frac{rA''_{i+1}}{k+i-1} =$$

$$= rA''_{i+1} \cdot R \cdot \frac{2n - R}{R^2 i^2 - R^2 i + nR - R^2} \qquad /20/$$

where R = r-1

$$\Delta rA'_j = rA''_{i+1} - rA'_{i+1} = (k+i-1)\cdot D'' - (-k+i)\cdot D'' =$$
$$= (2k-1)\cdot D'' = \left(\frac{2n}{r-1} - 1\right) \cdot D'' \qquad /21/$$

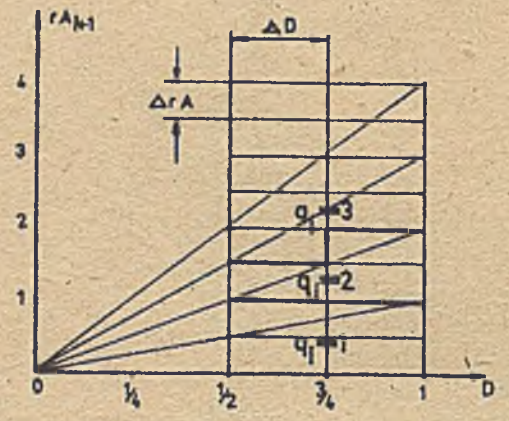


Fig. 4 A-D plot with r = 4, $|D| \geq 1/2$, k = 1

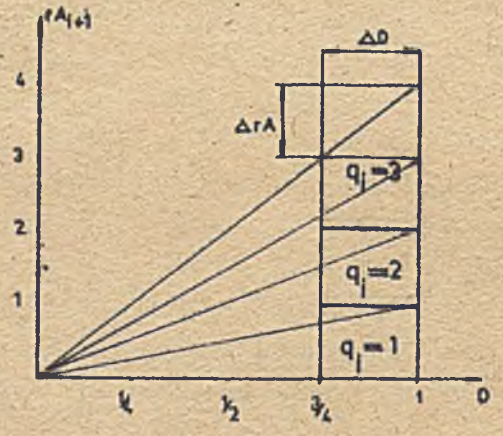


Fig. 5. A-D plot with r = 4, $|D| \geq 3/4$, k = 1

The above formulae have been given by D.E. Atkins in 1968 [V]. He noticed that the greater the redundancy of quotient code /the greater n/, the greater $\Delta rA'_j$ and $\Delta D'_j$ /the less precision of inspection of ra_{i+1} and d/. For given j and D_k, the quantities $\Delta rA'_j$ and $\Delta D'_j$ are the greatest when k = 1, because then the redundancy is the highest.

It is easy to note that $\Delta D'_j$ is minimum when j = n and $D = D_k$. However, $\Delta rA'_j$ is minimum when $D = D_k$. Remember that $(\Delta D'_j)_{min} = \Delta D'$ and $(\Delta rA'_j)_{min} = \Delta rA'$. So, the sides of the rectangles $\Delta rA'$ and $\Delta D'$ are defined by the left border of common area $q_i = n$ and $q_i = n-1$. In what follows only a part of the A-D plot which is defined by the lines $D = D_k$ $rA_{i+1} = (k+n-1)\cdot D$ and $rA_{i+1} = (-k+n)\cdot D$ will be considered /see Fig. 7/.

Now, let us consider in detail the influence of D_k on precision of inspection of ra_{i+1} and d. We search for such a value D_k, for which $\Delta rA'$ and $\Delta D'$ will be the greatest.

Note, that $\Delta D'$ -horizontal side of rectangle - is the longest, when it stretches from right-down border of the common area $q_i = n$ and $q_i = n-1$ /point A in the Fig. 7/ to its left

border /point B in the Fig. 7/. Co-ordinate of the point A is the required value D_k.

Value D_k can be determined in an analytical way:

$$-k+n = (k+n-1)\cdot D_k$$

$$D_k = \frac{-k+n}{k+n-1} \qquad /22/$$

Similar result can be obtained in another way, namely, by looking for a maximum of $\Delta D'_j$ which is defined by /20/. For given restrictions $|D| \geq D_k$ and $|D| \leq 1$ and for $j = n$ the quantity $\Delta D'$ reaches maximum for $D_k = (-k+n)/(k+n-1)$.

In what follows, we shall restrict outselves only to cases for which $k = 1$ /$n = r-1$/. As mentioned before, for these cases $\Delta rA'$ and $\Delta D'$ are the greatest.

For $k = 1$ the equation /22/ will take the form:

$$D_k = \frac{n-1}{n} = 1 - 1/n,$$

The devices that determine digits q_i will be designed basing on the value of D_k, so it would be convenient to bring D_k to an easily representable form. Since $r > n$:

$$D_k = 1 - \frac{1}{n} < 1 - \frac{1}{r}$$

Hence a new condition on the form of D can be obtained:

$$|D| \geq 1 - \frac{1}{r} \qquad /23/$$

When the condition /23/ is satisfied, precision of inspection of ra_{i+1} and d is still the lowest.

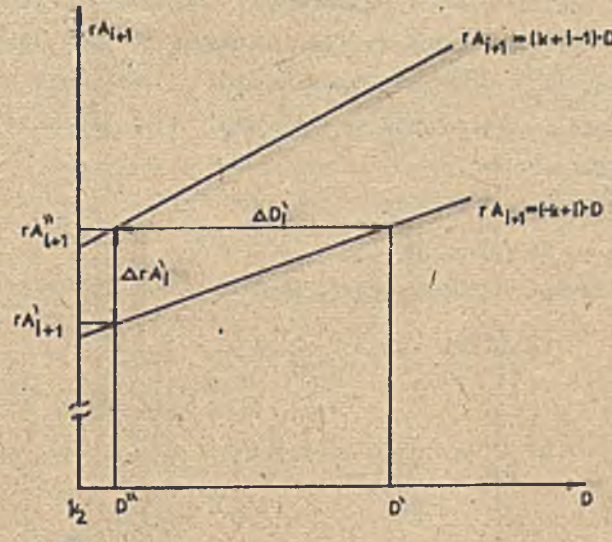


Fig. 6. Height $\Delta rA'_i$ and width $\Delta D'_i$ of common area $q_i = i$ and $q_i = i-1$

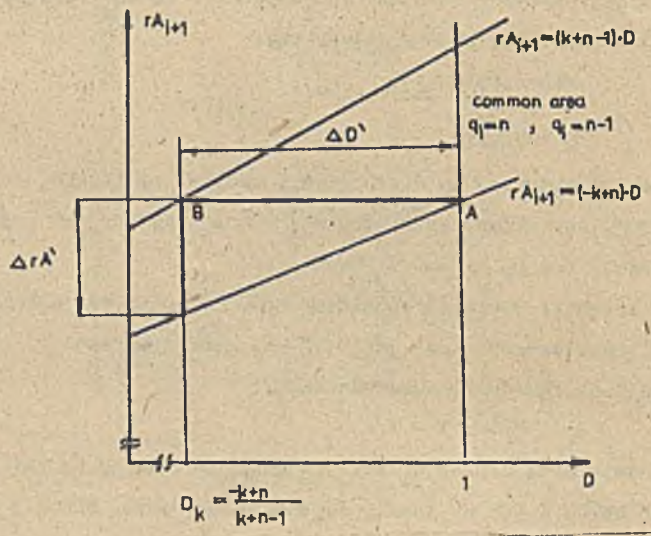


Fig. 7. Common area $q_i = n$ and $q_i = n-1$ for $|D| \geq D_k$

Now, precision of inspection of ra_{i+1} and d necessary to determine q_i will be found. With the restriction /23/, the A-D plot can be divided into rectangles, as shown in the Fig. 8. Vertical sides of rectangles are defined by the lines: $D = 1-1/r$, $D = 1$, and horizontal sides by the lines: $rA_{i+1} = n-2$, $rA_{i+1} = n-3$, $rA_{i+1} = n-4$ etc.

In order to determine q_i, the pair (rA_{i+1}, D) must be located in one of these rectangles. So, the value of rA_{i+1} must be located in one of the following intervals: $\langle n-1, n-2)$, $\langle n-2, n-3)$, ... $\langle -n+2, -n+1)$ while the value of D in an interval $\langle 1-1/r, 1\rangle$ or $\langle -1+1/r, -1\rangle$. For such location, we can inspect ra_{i+1} with the precision limited to $1+\log_2 r$ bits /bits of the integer part of ra_{i+1}/. Since the condition /23/ is satisfied, it is enough to inspect in the word d only bit of the sign. Hence:

$$NA = 1 + \log_2 r$$
$$ND = 1, \qquad /24/$$

where NA, ND - number of bits to be inspected in ra_{i+1}, d to determine the digit q_i.

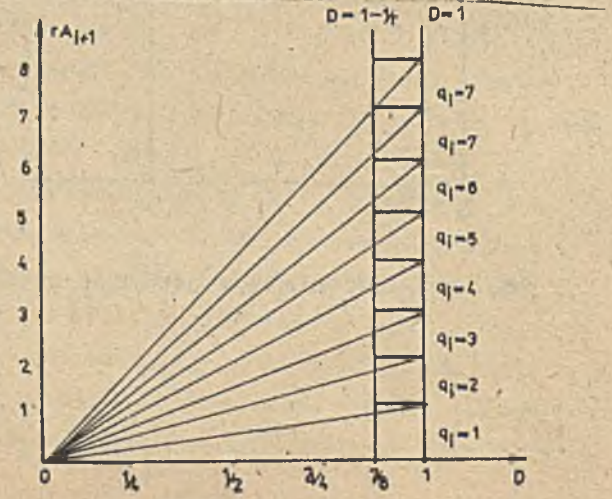


Fig. 8. Division of the A-D plot into rectangles when $|D| \geq 1-1/r$, $r = 8$

Further decrease of the number NA is impossible, because, when inspecting NA-1 bits in ra_{i+1}, we should differentiate between numbers, differing by 2, and the rectangle of the height of 2 would not be entirely included in any q_i area.

Note that in the method presented here, rectangles form vertical column defined by the lines D = 1-1/r and D = 1. This division of the A-D plot is much simpler than that demonstrated in the Figs 1, 2, 3; the step line does not appear. Simplicity of the described method is achieved at the cost of rigorous requirement put on the form of the divisor D; in the Figs 1, 2, 3 $|D| \geq 1/2$ and for presented method the form of the divisor is defined by /28/. The problem of bringing D to the form /28/ is discussed in the next section.

Table I illustrates the difference between division methods with fixed restriction on the form of D /e.g. $|D| \geq 1/2$/ and the method presented here with the restriction /28/. Note that the restriction /28/ depends on the value of r. Three ranges of variation of D: $|D| \geq 1/16$, $|D| \geq 1/2$, $|D| \geq 1-1/r$ are demonstrated in the Table 1. For each range the precision of inspection of ra_{i+1} and d for different r is given.

Required precision of inspection of ra_{i+1} and d — Table 1*

r	Joint quantity of bits must be inspected to select q_i or NA+ND		
	$\|D\| \geq 1/16$	$\|D\| \geq 1/2$	$\|D\| \geq 1-1/r$
4	16	7	4
8	18	10	5
16	21	15	6
32	23	17	7
64	25	19	8
246	29	23	10

One can see from the Table 1 that when the divisor is brought to the form /28/ the problem of determining q_i is simplified considerably. For example: function of 10 variables permits to determine q_i with the radix 8 in the case of $|D| \geq 1/2$, and with the radix 246 in the case of $|D| \geq 1-1/r$. It can affect substantially the cost and speed of the division.

A simple device can be designed for the method of the q_i selection proposed here. Denote by $r\hat{a}_{i+1}$ the word ra_{i+1} cut to NA most significant bits. Let $r\hat{A}_{i+1}$ denotes a number represented by the word $r\hat{a}_{i+1}$. It is easy to note - see Fig. 8 - that the digit q_i is greater by 1 than $r\hat{A}_{i+1}$ /for the first quarter on the A-D plot, for others the signs of D and rA_{i+1} must be taken into consideration/. The above observation does not concern the case when $r\hat{A}_{i+1} = n$; for this case $q_i = r\hat{A}_{i+1}$.

To sum up, one can write for the first quarter of the A-D plot:

$$q_i = \begin{cases} r\hat{A}_{i+1} + 1 & \text{when } r\hat{A}_{i+1} < n \\ r\hat{A}_{i+1} & \text{when } r\hat{A}_{i+1} = n \end{cases}$$

For the whole A-D plot the principle of the q_i selection is given by expression:

$$q_i = \begin{cases} (r\hat{A}_{i+1} + \text{sign}\,(r\hat{A}_{i+1})) \cdot \text{sign}\,(D) & \text{when } |rA_{i+1}| < n \\ rA_{i+1} \cdot \text{sign}\,(D) & |rA_{i+1}| = n, \end{cases} \qquad /26/$$

where $\text{sign}\,(D) = |D|/D$.

*Two first columns of the Table 1 result from the method of selection of q_i proposed by Atkins; see [1].

The device which implements /25/ has a very simple structure. It consists of binary adder and device which blocks adding the value of sign $(r\hat{A}_{i+1})$ when $|r\hat{A}_{i+1}| = n$. The block scheme of such device is shown in the Fig. 9.

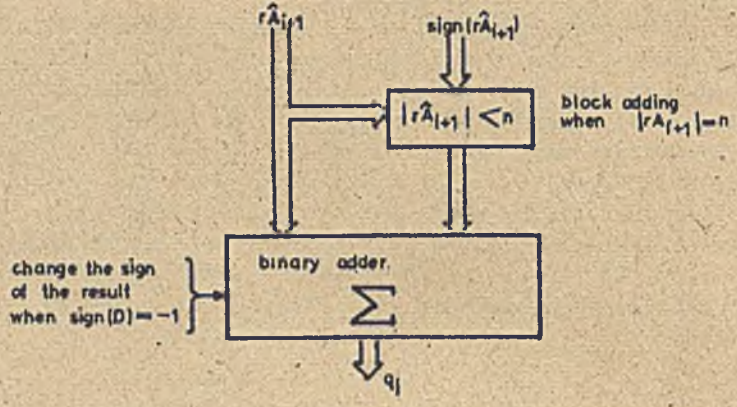


Fig. 9. Block scheme of the device determining the digit q_i.

The device from the Fig. 9 determines q_i for any radix of the SRT division and for $|D| \geq 1-1/r$. In the next section the way of reduction of D to the form $|D| \geq 1-1/r$ is presented.

4. NORMALIZATION OF THE DIVISOR

In the previous section a method of q_i selection was proposed. This method puts the following restrictions on the variation range of the divisor:

$$|D| \leq 1 \qquad /26/$$
$$|D| \geq 1-1/r$$

In this section the way of reducing divisor to the interval /26/ is shown. This process will be called normalization. We assume that the divisor is initially normalized so that:

$$|D| \leq 1 \qquad /27/$$
$$|D| \geq 2^{-p}$$

where p is an integer.

When p = 1, the divisor is binary-normalized and when p = 4, the divisor is hexadecimaly-normalized. Note that many modern computers are equipped with the devices which reduce numbers to the interval /27/. It concerns particularly the floating-point numbers. For example: some IBM computers normalize mantissas of the floating-point numbers to the interval $\langle 1/16, 1)$.

Let us examine the variation range of |D|, i.e. the interval $\langle 0,1\rangle$. Note that when:

$|D| \in \langle 0, 2^{-p})$ according to /27/, this event can not happen
$|D| \in \langle 2^{-p}, 1-1/r)$ divisors from that range should be reduced to the interval /26/
$|D| \in \langle 1-1/r, 1\rangle$ divisors from that range fulfil /26/.

Consequently, only divisors from the interval $\langle 2^{-p}, 1-1/r)$ should be transformed. Now, the method of reducing them to the interval /26/ will be given.

The interval $\langle 2^{-p}, 1-1/r)$ can be divided into equal parts of the length of b. In this way, we get a sequence of partitions: $\alpha_1, \alpha_2, \ldots, \alpha_m$.

$\alpha_1 = (1-1/r,\ 1-1/r-b\rangle$
$\alpha_2 = (1-1/r-b,\ 1-1/r-2b\rangle$
.
.
.
$\alpha_m = (1-1/r-(m-1)b,\ 1-1/r-bm\rangle$,

where

$$m = \frac{1-1/r-2^{-p}}{b}$$

For each of the intervals α_i one can assign such a number /factor/ a_i, that

$$|D| \in \alpha_i \Rightarrow |D| \cdot a_i \in \langle 1-1/r, 1\rangle$$

In other words, factors a_i transform values from intervals α_i into values from interval $\langle 1-1/r, 1\rangle$. So, the process of transformation of the divisor into the form /26/ consists in locating a given value $|D|$ within one of the intervals α_i and then multiplying D by an adequate factor a_i.

Note that to locate $|D|$ within intervals of the length of b, it is sufficient to inspect word d with the limited precision. This precision decreases with the increasing length of the interval α_i /value b increases/.

Now, the values of b and a_i will be calculated. Let y denote a co-ordinate of the end of the interval α_i. The following inequalities must be fulfilled to transform the value from interval $\langle y-b, b\rangle$ to the value from interval $\langle 1-1/r, 1\rangle$:

$$\begin{cases} y \cdot a_i \leq 1 \\ y \cdot a_i \geq 1-1/r \\ (y-b) \cdot a_i \geq 1-1/r \\ (y-b) \cdot a_i \leq 1 \\ y \geq 1-2^{-p} \\ y \leq 1 \\ a_i > 0 \\ b > 0 \\ r \geq 2 \end{cases} \qquad /28/$$

Now, a maximum value b_{max} is needed, for which /28/ will be fulfilled for every value y from the interval $\langle 2^{-p}, 1-1/r\rangle$. After solving /28/ it turns out that this value is $b_{max}=1/(r \cdot 2^p)$. One can see that b_{max} is easily binary-representable. In accordance with the calculated value b_{max} the number of the intervals α_i is:

$$m = \frac{1-1/r-2^{-p}}{b_{max}} = 2^p \cdot (r-1) - r$$

For a given value of p the intervals α_i are as follows:

$$\alpha_1 = \langle 1-1/r,\ 1-1/r-1/(r \cdot 2^p)\rangle$$
$$\alpha_2 = \langle 1-1/r-1/(r \cdot 2^p),\ 1-1/r-2/(r \cdot 2^p)\rangle$$
$$\vdots$$
$$\alpha_m = \langle 1-1/r-(m-1)/(r \cdot 2^p),\ 1-1/r-m/(r \cdot 2^p)\rangle .$$

To calculate factors a_i, each of the intervals

$$\alpha_i = \langle 1-1/r-(i-1)/(r \cdot 2^p),\ 1-1/r-1/(r \cdot 2^p)\rangle$$

must be reduced to the interval $\langle 1-1/r, 1\rangle$, and the following relations must be fulfilled:

$$(1-1/r-(i-1)/r \cdot 2^p) \cdot a_i \leq 1$$
$$(1-1/-r-i/(r \cdot 2^p)) \cdot a_i \geq 1-1/r$$
$$i = 1,2,\ldots,m,$$

or

$$a_i \leq \frac{r \cdot 2^p}{2^p r - 2^p - (i-1)} \qquad /29/$$

$$a_i \geq \frac{r-1}{2^p r - 2^p - i}$$

Note that factors a_i are not defined in a univocal way. They can be selected from the intervals defined by /29/. It permits us to design freely a device which calculates a_i.

Let us determine now the precision of inspection of D. Modulus of the divisor D should be located within intervals of the length of $b_{max}=1/(r \cdot 2^p)$. For this location d must be inspected with the precision restricted to n bits:

$$n = \log_2 (r \cdot 2^p) + 1 = \log_2 r + p + 1$$

(the +1 bit is the bit of the sign)

Example

Table 2 illustrates the above considerations for the case p=1, r=8. Columns of the Table 2 contain:

- division of the variation range of D into intervals α_i,
- intervals in which factors a_i must be included,
- chosen factors a_i

It is easy to calculate that for r=8 and p=1, b_{max} equals 1/16 and m equals 5.

Normalizing factors a_i for p=1, r=8 — Table 2

intervals α_i	intervals in which factors a_i must be included	chosen factors a_i
⟨0,1/16) ⟨1/16,2/16) : ⟨7/16,8/16)	divisors from these intervals do not appear because we assumed p=1 and $\|D\| \geq 1/2$	
⟨8/16,9/16)	⟨1.750,1.778⟩	1.7500
⟨9/16,10/16⟩	⟨1.556,1.600⟩	1.5625
⟨10/16,11/16)	⟨1.400,1.455⟩	1.4375
⟨11/16,12/16)	⟨1.272,1.330⟩	1.3135
⟨12/16,13/16)	⟨1.166,1.231⟩	1.1875
⟨13/16,14/16]	⟨1.080,1.144⟩	1.1250
⟨14/16,15/16)	divisors fulfil /26/	1.0000
⟨15/16,16/16)		1.0000
analogous for negative divisors.		

Conclusions and observations

1. In the method of q_i selection described in sections 3 and 4, first the normalization $D.a_i$ should be executed. Obviously, the multiplication $D.a_i$ must last much shorter than the division, if we want the normalization to be profitable. Otherwise, any profit of simplification of the device calculating q_i, will be lost due to extending the division execution time. So the normalization is worth using only in devices which perform the multiplication operation very quickly, i.e, comparably with the adding operation.

2. Apart from normalization $D \cdot a_i$, operation $A \cdot a_i$ should be executed to obtain the correct value of Q. It is obvious because:

$$\frac{A}{D} = \frac{A \cdot a_i}{D \cdot a_i} = Q$$

The correct value of Q can be also obtained if Q' is multiplied by a_i', where: $Q' = A/(D \cdot a_i)$, $a_i' = 1/a_i$. This method is even more practical, because $a_i' < 1/a_i$ is always greater than 1 - see /29//.

3. Normalizing process to a required interval ⟨1-1/r,1) can take place in several stages i.e., D can be brought in succession into intervals: $\langle 1-1/r_1,1\rangle, \langle 1-1/r_2,1\rangle, \ldots, \langle 1-1/r,1\rangle$, where $r_1 < r_2 < \ldots < r$. For example: hexadecimaly-normalized D /$|D| \in \langle 1/16,1\rangle$/ can be reduced into the interval ⟨7/8,1⟩ in two stages: first into the intermediate interval ⟨1/2,1⟩ and then into the target interval ⟨7/8,1⟩. This process considerably decreases the number of factors a_i what allows to simplify corresponding devices. For example: the process of reducing directly $|D| \in \langle 1/16,1\rangle$ into the interval ⟨7/8,1⟩ needs 104 different factors a_i, while performing that in two stages /first to ⟨1/2,1⟩, next to ⟨7/8,1⟩/ needs only 20 factors a_i. Moreover precision of the inspection of D decreases for the multistage normalization, hence the cost of the device calculating a_i decreases.

4. Different versions of the q_i selection are possible. For example the divisor D can be normalized to the interval $\langle 1-1/r_1,1\rangle$ and the radix of the division can be $r_2 > r_1$. Advantages of this procedure are the following:

- reduction of D to the interval $\langle 1-1/r_1,1\rangle$ decreases precision of the inspection of ra_{i+1} and d, necessary to determine q_i
- reduction of D to the interval $\langle 1-1/r_1,1\rangle$ demands less precision of the inspection of d, than reduction of D to the interval $\langle 1-1/r_2,1\rangle$ /$r_1 < r_2$/.

In other words, when we decrease precision of the inspection of d in the normalizing process,

we increase it during the process of choosing q_i. This version of method gives a designer much freedom in choosing between the cost of the implemented algorithm and its speed. In the Fig. 10 the division of the A-D plot into rectangles is presented. In this figure D is normalized to $\langle 7/8,1\rangle$ and the radix of the division is r=16.

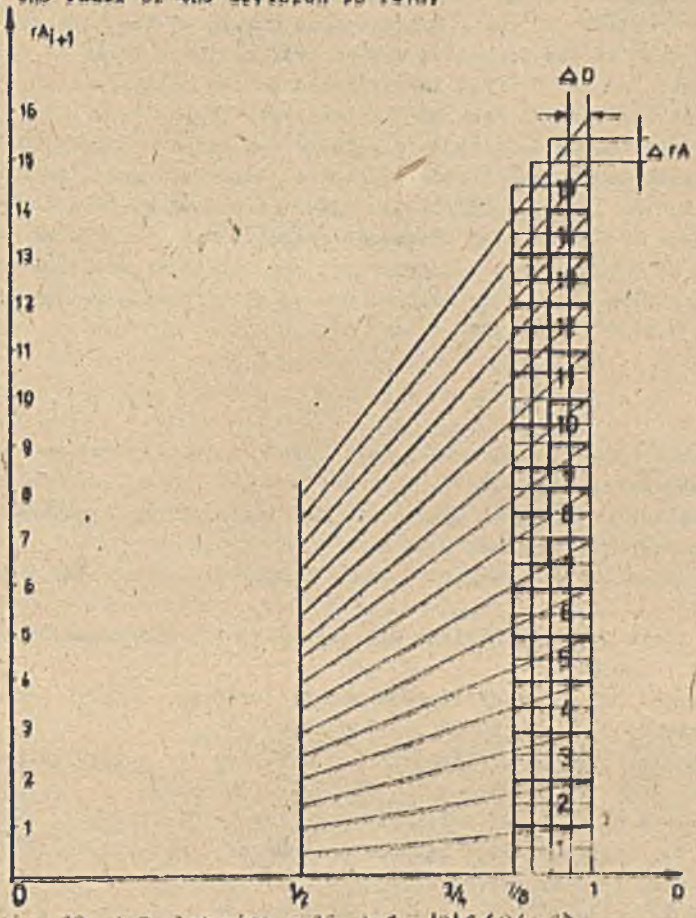


Fig. 10. A-D plot with r=16, k=1, $|D| \in \langle 7/8,1\rangle$

5. Sufficiently great r should be chosen to accelerate the process of division. Great r cannot be chosen when D is not normalized, or is normalized to an interval too wide, e.g. $\langle 1/16,1\rangle$. In this case, the precision of inspection of ra_{i+1} and d is too high to construct complicated devices calculating q_i /for r=32, q_i is a function of 25 variables when $|D| \in \langle 1/16,1\rangle$/. The method described above, considerably reduces cost of the device calculating q_i, as it permits to decrease considerably the required precision of inspection of ra_{i+1} and d/ for r=32, q_i is a function of 7 variables only. So, using this method, very rapid division algorithms can be implemented. But one should bear in mind that in this method

- simplification of the q_i selection is reached at the cost of additional devices calculating factors a_i
- high-speed arithmetical unit executing operation of the multiplication is needed.

5. APPLICATION OF THE DIVISION WITH THE NORMALIZED DIVISOR

SRT division with the normalized divisor is to be used in high-speed processor being designed in the Institute of Mathematical Machines in Warsaw. According to the design processor called INMAS is to execute operation $a \cdot b+c$ where a,b,c are 64-bit floating-point numbers.

IMMAS processor co-operates with IBM computers as the input-output channel. It is equipped with own small local memories and memory for constant values. The speed of IMMAS amounts to about $33 \cdot 10^6$ operations per second. Such a high speed was reached as a result of the pipelined mode of the data processing. The IMMAS processor is equipped with a device which normalizes the modulus of mantissa of the floating-point number to the interval $\langle 1/16, 1\rangle$.

In the implementation of the presented method r=32 has been taken as a radix of the division. The normalization of the divisor is performed in two stages: first, the divisor is reduced to the interval $\langle 1/2, 1\rangle$ and then to the interval $\langle 31/32, 1\rangle$.

In sum the process needs 44 factors a_i. These factors are stored in the memory for constant values. A simple device calculates address of a factor basing on 6 bits of d. After reducing the divisor to the interval $\langle 31/32, 1\rangle$ /after executing $a_i \cdot D$/ and after multiplying the dividend by adequate factors, IMMAS processor executes the division according to /6/.

Owing to the use of a high-speed processor and the method of division presented here, it is expected that the time of the division of the 64-bit floating-point numbers will amount to about 600 ns i.e., $1.7 \cdot 10^6$ divisions per second.

References

[1] ATKINS D.E.: Higher-radix division using estimates of the divisor and partial remainders, IEEE Trans. on Computers, 1968 t.C-17, s. 925-934.

[2] PAPLIŃSKI A.: Analiza sumacyjnych algorytmów dzielenia, Prace Instytutu Informatyki Politechniki Warszawskiej, Warszawa 1977

[3] FREIMAN C.V.: Statistical analysis of certain binary division algorithms, Proc.IRE, 1961, t. 49, s. 91-103

[4] ROBERTSON J.E.: A new class of digital division methods, IRE Trans. Electronic Computers, 1958, t. EC-7, s. 218-222.

[5] AVIZIENIS A.: Signed digit number representation for fast parallel arithmetic, IRE Trans. Electronic Computers, 1961, t. EC-10, s. 389-400.

[6] TOCHER T.D.: Techniques of multiplication and division for automatic binary computers, Quarter. J. Mech. Appl. Math, 1958, t. 2, Cz. 3, s. 364-384.

[7] FLORES I.: The Logic of Computer Arithmetic, Englewood Cliffs N.J.: Prentice-Hall, 1963.

[8] ROBERTSON J.R.: The correspondence between methods of digital division and multiplier recoding procedures, IEEE Trans. on Computers, 1970, t. C19, s. 692-701.

[9] BAŃKOWSKI J.: Zapisy redundancyjne a szybkość operacji w maszynach cyfrowych, Arch. Aut. i Telemechaniki, Warszawa 1965, t. 10, z. 3.

Prace naukowo-badawcze Instytutu Maszyn Matematycznych

DZIELENIE LICZB BINARNYCH Z REDUNDANCYJNYM ZAPISEM KOLEJNYCH RESZT I ILORAZU

Stanisław MAJERSKI

W pracy opisano algorytmy dzielenia binarnego przeznaczone dla szybkich równoległych układów cyfrowych. W algorytmach tych wyeliminowano odejmowania i dodawania z propagacją przeniesień, występujące w klasycznej metodzie dzielenia nieresztytucyjnego. Działania te zastąpiono szybszym trójskładnikowym dodawaniem z pamiętaniem przeniesień, które dla odróżnienia od zwykłego dodawania nazwano redukcją równoległą trzech składników do dwóch. W kolejnych krokach dzielenia otrzymuje się, w wyniku redukcji równoległej, kolejne dwuskładnikowe reszty chwilowe. Na podstawie kilku bitów takich reszt wyznacza się kolejne cyfry ilorazu w zapisie redundancyjnym. Zapis ten może być na bieżąco przekształcany w zwykły zapis binarny. Dla trzech omawianych algorytmów przyjęto odpowiednio zapisy redundancyjne ilorazu o cyfrach -1,0,+1, cyfrach 0,1,2, oraz cyfrach od -3 do +3. W trzecim z algorytmów jednemu krokowi dzielenia i jednej cyfrze ilorazu odpowiadają dwa jego bity.

1. WSTĘP

Przedmiotem niniejszej pracy są algorytmy dzielenia przeznaczone do stosowania w układach cyfrowych szybkiego równoległego dzielenia binarnego o dużej precyzji. Określenie "szybkie równoległe dzielenie" zostało tu użyte z tego względu, że przedstawione algorytmy opracowano pod kątem uzyskania możliwie najkrótszego czasu wykonania dzielenia. Umożliwiają one mianowicie przyjęcie równoległej struktury układu dzielenia, zapewniającej prawie całkowitą eliminację procesów szeregowego przetwarzania informacji, takich jak propagacja przeniesień. Określenie "duża precyzja" związane jest z tym, że algorytmy są bardzo efektywne, gdy wymagana jest duża dokładność obliczeń.

Ogólnie znane algorytmy dzielenia równoległego, a w szczególności powszechnie stosowany w układach cyfrowych algorytm równoległego dzielenia nieresztytucyjnego, opisane zostały między innymi w [1]. Algorytmy te obejmują określoną sekwencję odpowiednio od siebie uwarunkowanych odejmowań i dodawań, przy czym wybór określonego z tych działań zależy od wyniku poprzedniego działania, a ściślej od znaku tego wyniku. O czasie trwania dzielenia decyduje głównie łączna liczba odejmowań i dodawań oraz czas trwania pojedynczego odejmowania i dodawania, zależny zwykle między innymi od liczby bitów odejmowanych lub dodawanych liczb.

Jeśli wynik dzielenia liczb w zapisie binarnym może być przedstawiony w zapisie redundancyjnym, wówczas, dla określenia kolejnej cyfry ilorazu i kolejnego odejmowania czy dodawania, nie jest konieczna dokładna znajomość kolejnej reszty chwilowej, ale wystarczy odpowiednie jej oszacowanie, co można wykorzystać dla skrócenia poszczególnych kroków dzielenia. Algorytm takiego dzielenia podany został niezależnie przez D. Sweeney'a z IBM, J.E. Robertsona [2] i T.D. Tocchera [3], oraz od nazwisk tych autorów nazwany algorytmem dzielenia SRT. Uogólnienie tego algorytmu na rozwinięcia liczb o wyższych podstawach oraz szczegółowa analiza, jaka części kolejnej reszty powinna być badana dla wyznaczenia kolejnej cyfry ilorazu i określenia następnego kroku dzielenia, przedstawione zostały przez D.E. Atkinsa w [4], przy czym założono, że cyfry redundancyjnego zapisu ilorazu reprezentują liczby z przedziału symetrycznego względem zera.

W niniejszej pracy omówiono algorytmy dzielenia binarnego, w których iloraz przedstawiony jest w pozycyjnym zapisie redundancyjnym o podstawie rozwinięcia 2 i cyfrach -1,0,+1, o podstawie rozwinięcia 2 i cyfrach 0,1,2, jak również o podstawie rozwinięcia 4 i cyfrach -3,-2,-1,0,+1, +2,+3. Istotną cechą wszystkich trzech algorytmów jest tworzenie kolejnych reszt chwilowych w postaci dwóch składników, które następnie łącznie z trzecią liczbą, będącą odpowiednią wielokrotnością dzielnika, są "redukowane równolegle" z zachowaniem ich sumy do dwóch składników kolejnej reszty chwilowej. Określenie "redukcja równoległa" wprowadzono tu dla "trójskładnikowego dodawania z pamiętaniem przeniesień". Powodem było zarówno wyraźniejsze rozróżnienie takiej operacji od zwykłego dodawania i odejmowania, jak również to, że praca niniejsza jest jedną z cyklu prac autora, w których dla zwiększenia szybkości układów cyfrowych wykorzystuje się "redukcję równoległą" również innej liczby składników,np. redukcję z 7 do 3 składników. Prace te, nawiązujące do zgłoszonych w ostatnich latach patentów [5] - [12] zostaną w niedługim czasie złożone do opublikowania.

Redukcja równoległa, zastępująca odejmowania i dodawania, jest wykonywana całkowicie równolegle, bez propagacji przeniesień wzdłuż redukowanych liczb, w czasie praktycznie niezależnym od liczby ich bitów. Dla każdego z omawianych w pracy algorytmów podano jakie minimalne grupy bitów składników kolejnych reszt należy badać dla wyznaczenia cyfr ilorazu i jakie cyfry ilorazu jakim kombinacjom bitów należy przyporządkować. Układy cyfrowe dzielenia oparte na tych algorytmach są przedmiotem zgłoszenia patentowego [10] .

Zmiana zapisu redundancyjnego ilorazu na binarny zapis nieredundancyjny może być sprowadzona do wykonania jednego dodawania binarnego dwóch liczb w zapisie nieredundancyjnym, jak również może odbywać się na bieżąco w trakcie otrzymywania cyfr ilorazu, począwszy od cyfr najbardziej znaczących, w sposób opisany na przykład w [4] .

Ze względu na dużą objętość pracy wydaje się celowe podanie następującego komentarza do poszczególnych rozdziałów pracy.

Czytelnik poszukujący algorytmów dzielenia tylko w celu wyboru odpowiedniego rozwiązania projektowanego przez siebie układu dzielenia może ograniczyć się do przeczytania rozdziałów 5,7,8. Każdy z nich opisuje jeden "szybki" algorytm dzielenia, przy czym rozdziały 5 i 7 dotyczą dzielenia, w którym jednemu krokowi odpowiada wyznaczenie 1 bitu ilorazu, a rozdz. 8 dotyczy dzielenia bardziej złożonego, w którym jednemu krokowi odpowiadają 2 bity ilorazu. Algorytm z rozdz. 5 jest prostszy niż z rozdz. 7, ten ostatni jednak może mieć pewne zalety, w przypadku gdy ten sam układ cyfrowy jest wykorzystany do wyznaczania wartości bardziej złożonych funkcji /porównaj [11] i [12]/.

Rozdziały 4 i 6 umożliwiają dokładniejsze zapoznanie się i zrozumienie zasady działania algorytmu z rozdz. 5, a pośrednio i dwóch pozostałych algorytmów, natomiast rozdz. 3 stanowi wprowadzenie do wszystkich trzech algorytmów.

Rozdział 2 nie jest w zasadzie związany tylko z dzieleniem, ale również z wyznaczaniem wartości innych bardziej złożonych funkcji. Dotyczy on mianowicie założeń do całego, wspomnianego wcześniej, przygotowywanego cyklu opracowań poprzedzonego patentami [5] - [12], a poświęconego zwiększaniu szybkości złożonych układów cyfrowych poprzez eliminację w możliwie najwyższym stopniu procesów propagacji przeniesień wzdłuż przetwarzanych liczb.

W rozdziale 9 omówiono natomiast przekształcenie redundancyjnego zapisu liczb o cyfrach -1,0,+1, o cyfrach 0,1,2, jak również o cyfrach od -3 do +3 na nieredundancyjny zapis binarny /porównaj [4] /.

Wyjaśnienia wymaga jeszcze przyjęcie w niniejszej pracy większych zakresów dzielnej i reszt chwilowych niż zakres dzielnika. Autor uznał za bardziej komunikatywny opis algorytmów dla przypadku gdy dzielnik mieści się w dzielnej, lub reszcie chwilowej więcej niż jeden raz. Po zapoznaniu się z algorytmem nie powinno natomiast nikomu nastręczać trudności przesunięcie przecinka w dzielnej i dzielniku o dowolną liczbę pozycji binarnych i odpowiednie uwzględnienie tego w ilorazie.

2. DODAWANIE I RÓWNOLEGŁA REDUKCJA LICZB JAKO ELEMENTY BARDZIEJ ZŁOŻONYCH OBLICZEŃ

Wykonywanie bardziej złożonych operacji arytmetycznych, takich jak mnożenie czy dzielenie, jak również wyznaczanie wartości bardziej skomplikowanych funkcji jedno lub wieloargumentowych, w przeznaczonych specjalnie do tego celu układach cyfrowych, sprowadza się zwykle do wykonania określonej sekwencji odpowiednio uwarunkowanych dodawań liczb, negowań bitów liczb, oraz dekodowań mniejszych lub większych grup bitów. W przypadku tradycyjnie stosowanych binarnych zapisów liczbowych, o minimalnej liczbie bitów koniecznej do przedstawienia liczb o określonej precyzji, duży wpływ na czas trwania całej złożonej operacji ma występujący w trakcie dodawań proces propagacji przeniesień. Dotyczy to zwłaszcza liczb o dużej liczbie bitów. Procesu tego mającego w dużym stopniu cechy przekształcenia szeregowego można uniknąć stosując redundancyjne zapisy liczb. Prowadzi to jednak z kolei do bardziej złożonej struktury układów cyfrowych i konieczności dysponowania większymi zasobami pamięciowymi. W przypadku wysokich wymagań na szybkość wykonywania obliczeń, rozwiązań optymalnych należy poszukiwać wśród rozwiązań wykorzystujących w odpowiedni sposób zarówno nieredundancyjne zapisy liczbowe, charakteryzujące się minimalną liczbą bitów, jak i zapisy redundancyjne. W szczególności można pobierać z pamięci i wpisywać do pamięci liczby w zapisie nieredundancyjnym, natomiast w obliczeniach operować również na liczbach w zapisach redundancyjnych. Jako elementy składowe bardziej złożonych obliczeń mogą występować wtedy szybkie dodawania, dające jako wyniki pośrednie obliczeń liczby w zapisie redundancyjnym, jak i wolniejsze dodawania, dające jako wyniki końcowe obliczeń, liczby w zapisie nieredundancyjnym. Stosunek jednych do drugich decyduje w dużym stopniu o "wydajności" obliczeń. Narzuca się stąd wniosek, że dla zwiększenia tej wydajności należy zastępować proste operacje zawierające niewiele elementarnych dodawań, bardziej złożonymi operacjami obejmującymi możliwie długie sekwencje dodawań. W wielu przypadkach tylko jedno, a mianowicie ostatnie z tych dodawań, dające wpisywany do pamięci wynik w zapisie nieredundancyjnym nie może być realizowane w sposób całkowicie równoległy.

W konsekwencji powyższego rozumowania należy rozważyć trzy możliwe rodzaje dodawań dwuargumentowych, zależnie od istnienia argumentów w zapisie nieredundancyjnym i redundancyjnym, gdzie wynikiem tych dodawań jest suma przedstawiona w zapisie redundancyjnym, a następnie omówić również sposób przekształcenia zapisu takiej sumy w zapis nieredundancyjny. Przed omówieniem tych trzech rodzajów dodawań poświęcimy trochę uwagi niektórym zapisom liczbowym.

Załóżmy najpierw, że liczby pobierane z pamięci przedstawione są w zapisie nieredundancyjnym, a mianowicie w zapisie binarnym uzupełnieniowym. Zapis dwóch takich liczb /dogodnie pisać jedną taką liczbę nad drugą/, można uważać za zapis jednej liczby, a mianowicie ich sumy, która jest przedstawiona w zapisie redundancyjnym o podstawie rozwinięcia 2 i zakodowanych binarnie cyfrach 0,1,2 z wyróżnioną pozycją znakową o zakodowanych binarnie cyfrach 0,-1,-2. Aby umożliwić całkowicie jednorodne traktowanie cyfr na wszystkich pozycjach binarnych, przyjmiemy następującą interpretację liczb, przedstawionych w zapisie binarnym uzupełnieniowym i pamiętanych w rejestrach maszyny cyfrowej. Będziemy uważać, że najbardziej znacząca pozycja binarna rejestrów maszyny nie jest pozycją znakową, ale, że pozycja znakowa znajduje się już poza rejestrem na dowolnie dalekiej pozycji /w zależności od potrzeby/. Przy takiej interpretacji jedynka na najbardziej znaczącej pozycji rejestru świadczy wprawdzie o tym, że liczba jest ujemna, ale jedynce tej można przypisać wagę dodatnią. Pozwala to na jednorodne traktowanie wszystkich pozycji binarnych liczb pamiętanych w rejestrach maszyny cyfrowej.

Przykładami zapisów uzupełnieniowych liczb całkowitych w rejestrach pięciobitowych i sum liczb w zapisie redundancyjnym o podstawie 2 i cyfrach 0,1,2 zakodowanych binarnie są zapisy

$$\left.\begin{array}{ll} 0\,0\,1\,0\,1 & +5 \\ 1\,1\,0\,1\,1 & -5 \end{array}\right\} 0$$

$$\left.\begin{array}{ll} 0\,1\,1\,0\,1 & +13 \\ 1\,1\,0\,1\,1 & -5 \end{array}\right\} +8$$

$$\left.\begin{array}{ll} 0\,0\,1\,0\,1 & +5 \\ 1\,1\,0\,0\,1 & -7 \end{array}\right\} -2$$

gdzie wszystkie jedynki traktuje się jako jedynki dodatnie.

Jak łatwo wywnioskować z podanych przykładów przekształcenie takich redundancyjnych zapisów sum w tradycyjne zapisy nieredundancyjne sprowadza się do wykonania klasycznego dodawania dwóch dodatnich liczb binarnych /przy odpowiednim założeniu ograniczającym zakres dodawanych liczb/.

W przypadku, gdy w wyniku dodawania dwóch liczb przedstawionych w zapisie uzupełnieniowym chcielibyśmy otrzymać sumę w zapisie redundancyjnym o podstawie rozwinięcia 2 i zakodowanych binarnie cyfrach -1,0,+1, wówczas należy tylko zanegować bity jednej z dodawanych liczb, przypisując otrzymanym bitom "wagi" ujemne, oraz dodatkowo zapamiętać ujemną jedynkę na najmniej znaczącej pozycji w dodatkowym rejestrze jednopozycyjnym/. Należy pamiętać przy tym, że np. zapis 11110 odpowiada wtedy liczbie -2 w przypadku przypisanych poszczególnym pozycjom binarnym wag dodatnich, a +2 w przypadku wag ujemnych.

Analogicznie, odwrotne przekształcenie, a mianowicie przekształcenie zapisu o zakodowanych binarnie cyfrach -1,0,+1 w zapis o cyfrach 0,1,2 wymaga zanegowania bitów "składnika ujemnego" oraz zapamiętania dodatniej jedynki na najmniej znaczącej pozycji dodatkowego rejestru jednopozycyjnego.

Przechodząc do dodawania liczby w zapisie uzupełnieniowym do liczby w omówionym "dwuskładnikowym" zapisie redundancyjnym o kodowanych binarnie cyfrach 0,1,2 napotykamy /przy przyjętej wyżej interpretacji zapisu uzupełnieniowego/ na problem "redukowania równoległego trzech liczb dodatnich" w zapisie binarnym do dwóch liczb "dodatnich" w tym zapisie z zachowaniem ich sumy. Taka redukcja równoległa jest niczym innym jak trójskładnikowym dodawaniem binarnym z zachowaniem przeniesień, które realizować można w szeregu niezależnie od siebie działających sumatorów jednopozycyjnych. Każdy z nich "redukuje" trzy bity redukowanych składników do dwóch bitów dwuskładnikowego wyniku. Aby redukcja równoległa nie kojarzyła się czytelnikowi wyłącznie z sumowaniem i sumatorami, wymienimy jako inny jej przykład redukcję równoległą siedmiu składników do trzech składników, realizowaną przez szereg koderów, z których każdy redukuje siedem bitów, z tej samej pozycji binarnej siedmiu składników, do trzech bitów, zajmujących trzy kolejne pozycje binarne w zredukowanych składnikach.

Przechodząc z kolei do dodawania dwóch liczb przedstawionych w omówionym "dwuskładnikowym" zapisie redundancyjnym o kodowanych binarnie cyfrach 0,1,2 dochodzimy do problemu redukcji z czterech do dwóch składników w binarnym zapisie uzupełnieniowym z zachowaniem sumy tych składników. Redukcję taką można wykonać równolegle w dwóch szeregach jednopozycyjnych sumatorów binarnych, w czasie nie przekraczającym czasu propagacji sygnału przez dwa sumatory jednopozycyjne.

Przedstawione wyżej rozumowanie stało się punktem wyjścia dla szeregu opracowań autora, których wspólną cechą jest, jak to już wspomniano we wstępie, dążenie do eliminacji szeregowego przetwarzania informacji przez zastępowanie dodawań i odejmowań redukcją równoległą składników.

3. ITERACYJNY PROCES WYZNACZANIA CYFR ILORAZU

Wykonanie dzielenia dzielnej a przez dzielnik b polega na wyznaczeniu ilorazu q i reszty r spełniających równanie

$$r = a - bq \qquad /1/$$

gdzie reszta r mieści się w określonym zakresie, np. spełniając warunek $0 \leq r < b$.

W układach cyfrowych dzielenie jest realizowane zazwyczaj w trzech etapach, obejmujących:

- czynności wstępne, takie jak normalizacja, przez którą rozumiemy sprowadzenie dzielnej i dzielnika do odpowiednich przedziałów liczbowych,
- iteracyjny proces wyznaczania kolejnych cyfr ilorazu,
- czynności końcowe, takie jak np. zmiana postaci reszty, czy zmiana zapisu lub korekta ilorazu.

Najbardziej czasochłonnym, a zatem mającym największy wpływ na efektywność działania układu dzielenia jest zwykle iteracyjny proces wyznaczania kolejnych cyfr ilorazu.

W przypadku wyznaczania ilorazu w zapisie pozycyjnym o podstawie rozwinięcia 2, w kolejnym kroku iteracji dobiera się kolejną cyfrę q_i ilorazu q na podstawie równania:

$$a_{i+1} = 2\,(a_i - bq_i) \qquad /2/$$

gdzie a_i jest kolejną, odpowiednio przesuniętą względem dzielnej resztą chwilową /nieprzesuniętą resztą chwilową jest $a_i 2^{-i}$/ przy czym dzielną a można uważać za początkową resztę chwilową a_0. Dla zbieżności tego procesu przyjmuje się przy tym odpowiedni warunek ograniczający zakres wartości a_i.

Istotnie, mnożąc obie strony relacji /2/ przez 2^{-i-1} oraz sumując stronami od i=0 do i=n-1 dostajemy

$$a_n\, 2^{-n} = a_0 - b \sum_{i=0}^{n-1} q_i\, 2^{-i} \qquad /3/$$

czyli otrzymujemy równanie /1/ przy założeniu, że

$$q = \sum_{i=0}^{n-1} q_i\, 2^{-i} \qquad /4/$$

$$r = a_n\, 2^{-n} \qquad /5/$$

Równanie /4/ przedstawia zapis ilorazu rozwiniętego przy podstawie 2, przy czym w przypadku liczby różnych cyfr q_i większej od dwóch przedstawia ono redundancyjny zapis ilorazu. Znaczna część niniejszej pracy poświęcona jest algorytmom dzielenia, w których cyfry ilorazu w zapisie redundancyjnym o podstawie rozwinięcia 2 reprezentują wartości -1,0,+1, lub wartości 0,1,2.

Jeśli dzielną i dzielnik, przedstawione w zapisie binarnym, będziemy traktować jako liczby w zapisie czwórkowym kodowanym binarnie i jeśli w miejsce relacji /2/ przyjmiemy

$$a_{i+1} = 4\,(a_i - bq_i) \qquad /6/$$

wówczas liczba kroków dzielenia zmniejszy się w przybliżeniu dwukrotnie, przy czym liczba różnych wartości cyfr q_i nie może być mniejsza od czterech. W miejsce /3/, /4/, /5/ można wówczas napisać

$$a_m\, 4^{-m} = a_0 - b \sum_{i=0}^{m-1} q_i\, 4^{-i} \qquad /7/$$

$$q = \sum_{i=0}^{m-1} q_i\, 4^{-i} \qquad /8/$$

$$r = a_m\, 4^{-m} \qquad /9/$$

W niniejszej pracy zajmiemy się również dzieleniem liczb przedstawionych w zapisie binarnym, traktowanym jako kodowany binarnie zapis czwórkowy, w którym iloraz przedstawiony jest w redundancyjnym zapisie czwórkowym o cyfrach -3,-2,-1,0,+1,+2,+3.

Dla wszystkich proponowanych w niniejszej pracy algorytmów dzielenia, punktem wyjścia dla iteracyjnego procesu wyznaczania cyfr ilorazu jest jedno z równań /2/, /6/. W kolejnych krokach tego procesu redukowane są równolegle trójki liczb do par liczb z zachowaniem ich sumy. Dwie z trzech redukowanych liczb są składnikami dzielnej lub składnikami kolejnej reszty chwilowej przedstawionymi w binarnym zapisie uzupełnieniowym. Trzecią redukowaną liczbą przedstawioną w takim samym zapisie, jest zero, dzielnik lub odpowiednia jego wielokrotność, zależnie od grupy bardziej znaczących bitów, wziętych z obu składników kolejnej reszty, a w pewnych przypadkach również od jednego, dwóch lub trzech bitów dzielnika. W wyniku redukcji w kolejnym kroku dzielenia wspomnianych trzech liczb do dwóch otrzymuje się dwa składniki następnej reszty chwilowej. W każdym z takich kroków, równocześnie z wyznaczeniem trzeciej redukowanej liczby, której w równaniach /2/ i /6/ odpowiada człon $-bq_i$, wyznaczona jest na podstawie tej samej grupy bitów kolejna cyfra ilorazu q_i.

4. KONSTRUKCJA ALGORYTMU DZIELENIA O CYFRACH ILORAZU -1,0,+1.

Opracowanie algorytmu dzielenia binarnego o ilorazie w zapisie redundancyjnym, sprowadza się w zasadzie do określenia jakie bity kolejnej reszty chwilowej i dzielnika należy analizować dla wyznaczenia kolejnej cyfry ilorazu i jakim kombinacjom bitów odpowiadają poszczególne cyfry ilorazu. Zmniejszenie liczby badanych bitów jest przy tym bardzo istotne, gdyż badanie jednej dodatkowej pozycji binarnej dwuskładnikowej reszty chwilowej powoduje około czterokrotne zwiększenie analizującego układu dekodującego.

Punktem wyjścia do określenia grupy analizowanych bitów kolejnej reszty chwilowej i dzielnika, koniecznych do wyznaczania kolejnej cyfry $q_i \in \{-1,0,+1\}$ ilorazu q oraz do określenia kryteriów wyznaczenia tej cyfry, jest równanie /2/. Zakładając zbieżność opisanego tym równaniem iteracyjnego procesu wyznaczania cyfr ilorazu /czyli zakładając ograniczenie zakresu wartości liczb a_i/ otrzymujemy warunek

$$|a_i| \leq 2|b| \qquad /10/$$

Z warunku tego wynikają trzy zakresy wartości kolejnej reszty chwilowej

$$-2|b| \leq a_i \leq 0 \qquad -|b| \leq a_i \leq |b| \qquad 0 \leq a_i \leq 2|b| \qquad /11/$$

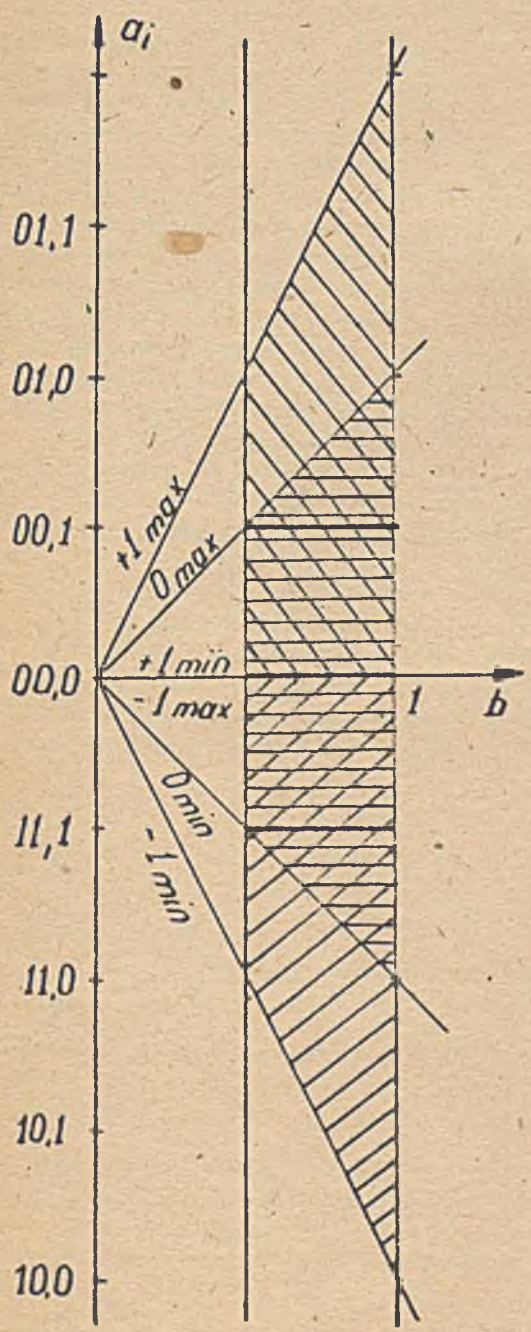
którym wolno przyporządkować cyfry -1,0,+1 ilorazu. W przypadku dodatniego dzielnika b podanym wyżej kolejnym zakresom reszt przyporządkowuje się odpowiednio cyfry -1,0,+1; dla ujemnego dzielnika kolejność przyporządkowania jest dokładnie odwrotna.

Jeśli na osi odciętych prostokątnego układu współrzędnych odkładać będziemy wartości dzielnika, a na osi rzędnych wartości kolejnej reszty, wówczas każdej parze liczb, jaką stanowi kolejna reszta i dzielnik, można przyporządkować punkt na płaszczyźnie. Nierównościom /11/ odpowiadają w szczególności na tej płaszczyźnie sektory kątowe, którym można przyporządkować odpowiednie cyfry ilorazu. Dla uproszczenia dalszego rozumowania ograniczymy się do dodatnich wartości dzielnika /nie zawężając przez to ogólności problemu, gdyż zmiana znaku dzielnika ma wpływ tylko na zmianę znaku cyfr +1, -1 ilorazu/. Załóżmy, że znana jest określona grupa kolejnych bardziej znaczących bitów reszty chwilowej i odpowiednie bity dzielnika. Na wspomnianej płaszczyźnie wyznaczony jest wówczas obszar prostokątny, któremu, o ile leży on w całości w jednym sektorze kątowym, można przyporządkować cyfrę ilorazu odpowiadającą temu sektorowi. Taka metoda graficzna wyznaczania cyfr ilorazu podana została przez D.E. Atkinsa w [4], gdzie przeanalizowano dokładnie algorytmy dzielenia z ilorazem w zapisie redundancyjnym o symetrycznym względem zera zakresie cyfr. Proponowane w niniejszej pracy algorytmy dzielenia, różniące się od innych znanych algorytmów dzielenia dwuskładnikowymi resztami chwilowymi, opracowano, stosując analogiczną metodę graficzną do podanej w [4]. Omówimy kolejno dwa wykresy pokazane na rys. 1a i 1b, które dla dzielnika z przedziału

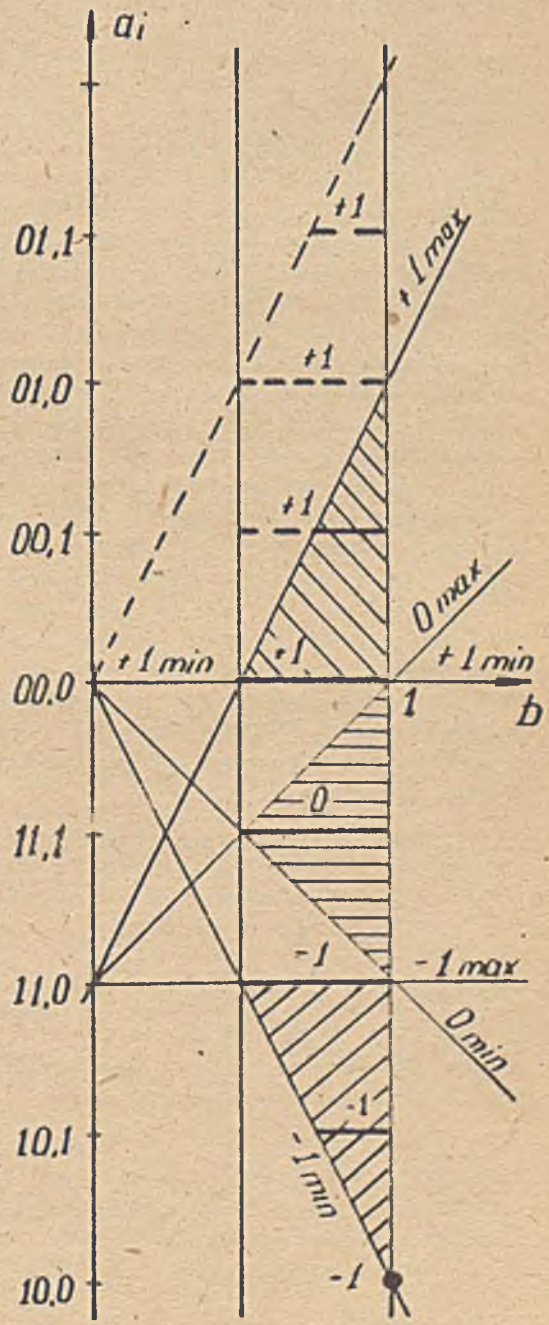
$$\frac{1}{2} \leq b < 1, \qquad /12/$$

służą do wyznaczania zakresów kolejnych reszt chwilowych, jakim przyporządkowuje się cyfry ilorazu -1,0,+1.

Wykres pierwszy jest szczególnym przypadkiem wykresu D.E. Atkinsa dla przyjętego wyżej zakresu dzielnika i cyfr ilorazu. Oś odciętych jest osią wartości dzielnika, a oś rzędnych osią wartości kolejnej reszty. Przy prostych ukośnych i poziomej, które stanowią ograniczenia sektorów, podano oznaczenia minimalnych i maksymalnych wartości zakresu cyfr -1,0,+1. Zakres dzielnika ograniczają dwie proste pionowe, między którymi obszar zakreskowany ukośnie ponad osią odciętych odpowiada cyfrze ilorazu +1, obszar zakreskowany ukośnie pod osią odciętych - cyfrze ilorazu -1, a obszar zakreskowany poziomo - cyfrze ilorazu 0. Tłuste odcinki poziome odpowiadają minimalnym i maksymalnym wartościom kolejnej reszty dla poszczególnych cyfr -1,0,+1, które można przyjąć dla dowolnej wartości dzielnika z przedziału /12/. Odległości między tymi odcinkami stanowią przedziały w ramach których można dla opracowywanego algorytmu dzielenia przyjąć najdogodniejsze wartości graniczne kolejnej reszty dla każdej z cyfr -1,0,+1.


a_i
01,1
01,0
00,1
00,0
11,1
11,0
10,1
10,0
+1 max
0 max
+1 min
-1 max
0 min
-1 min
1
b

Rys. 1a


a_i
01,1
01,0
00,1
00,0
11,1
11,0
10,1
10,0
+1
+1 max
+1
+1
+1 min
+1
0 max
+1 min
1
b
0
-1
-1 max
0 min
-1
-1 min
-1

Rys. 1b

Wykres drugi, na rys. 1b, jest stosowaną przez autora modyfikacją omówionego wyżej wykresu D.E. Atkinsa dostosowaną do reszt dwuskładnikowych. Wykres ten dotyczy przypadku, gdy od analizowanej grupy bardziej znaczących bitów dwóch składników kolejnej reszty chwilowej odcięta została /jedno miejsce po przecinku/ mniej znacząca część obu składników reszty. Wartość odciętej mniej znaczącej części obu składników reszty mieści się wówczas w przedziale $\langle 0,1)$. O górną granicę tego przedziału zostały zatem obniżone na wykresie 1b proste oznaczone przez -1_{max}, 0_{max} i $+1_{max}$ w stosunku do odpowiednich prostych z wykresu D.E. Atkinsa. W wyniku tego zabiegu zmniejszyły się znacznie zakreskowane na wykresie obszary, którym przyporządkować wolno odpowiednie z cyfr $-1,0,+1$. Powstały nawet między nimi trójkąty, którym nie można przyporządkować żadnej cyfry. Należy jednak pamiętać, że po odcięciu mniej znaczącej części składników reszty chwilowej, wartości analizowanej dla wyznaczenia cyfr ilorazu grupy bitów obu składników tej reszty stanowią całkowitą wielokrotność liczby $\frac{1}{2}$, a zatem interesujący nas obszar jest tylko zbiorem zaznaczonych na wykresie poziomych odcinków i jednego oddzielnego punktu. Warto tu jednak podkreślić, że chociaż obszarem wyjściowym dla proponowanego algorytmu dzielenia powinny być wartości bardziej znaczącej grupy bitów dzielnej, odpowiadające tłustym odcinkom na wykresie, to w trakcie dzielenia mogą występować również wartości bardziej znaczącej grupy bitów kolejnej reszty odpowiadające przerywanym odcinkom poziomym na wykresie. Wynika to stąd, że otrzymywane wartości kolejnych reszt ograniczone są od góry dopiero przez pokazaną na wykresie przerywaną prostą ukośną, a nie przez obniżoną w stosunku do niej o jedność prostą $+1_{max}$. Podane na wykresie liczby nad poziomymi odcinkami i przy zaznaczonym na dole wykresu punkcie wskazują, które z cyfr $-1,0,+1$ ilorazu przyporządkowuje się podanym na osi rzędnych wartościom, odpowiadającym analizowanej grupie bitów dzielnej lub kolejnej reszty chwilowej. Na tej podstawie można podać tabelę zależności cyfr $-1,0,+1$ ilorazu od wartości bardziej znaczących części kolejnych reszt chwilowych, a tym samym określić odpowiedni algorytm dzielenia. Tabela ta podana zostanie w następnym rozdziale.

Omówiony wykres z rys. 1b sporządzony został, jak już wspomniano, dla przypadku odcięcia mniej znaczących bitów składników kolejnej reszty chwilowej do jednego miejsca po przecinku. Jest to dla przyjętego zakresu dzielnika odcięcie optymalne ze względu na minimalną liczbę badanych bitów. Można tego dowieść analizując dwa dodatkowe wykresy, analogiczne do wykresu z rys. 1b, ale różniące się od niego miejscem odcięcia bitów kolejnej reszty o jedną pozycję binarną w prawo i w lewo. Wykresów tych i ich analizy nie będziemy podawać w niniejszej pracy.

5. ALGORYTM DZIELENIA O CYFRACH ILORAZU $-1,0,+1$

Zakładając, że dzielnik b spełnia warunek

$$\frac{1}{2} \leq |b| < 1 \qquad /13/$$

oraz, że dzielna i kolejne reszty chwilowe mieszczą się w przedziale

$$-2\,|b| \leq a_i \leq 2\,|b| \qquad /14/$$

można, jak to podano w poprzednim rozdziale, przyporządkować cyfry ilorazu $-1,0,+1$ następującym trzem zakresom kolejnych reszt

$$-2\,|b| \leq a_i < 0 \qquad -|b| \leq a_i \leq |b| \qquad 0 < a_i \leq 2\,|b| \qquad /15/$$

Kolejność przyporządkowania cyfr $-1,0,+1$ podanym zakresom reszt jest przy tym, w przypadku dzielnika dodatniego zgodna z podanym wyżej porządkiem cyfr i nierówności, a w przypadku dzielnika ujemnego dokładnie odwrotna. Pamiętając o tej jedynej różnicy, przyjmiemy w dalszych rozważaniach niniejszego rozdziału, że dzielnik b jest dodatni i spełnia warunek

$$\frac{1}{2} \leq b < 1 \qquad /16/$$

co oczywiście nie ogranicza zakresu stosowalności algorytmu.

Można wykazać, że przy tych założeniach, dla wyznaczenia kolejnej cyfry ilorazu potrzeba i wystarcza przeanalizować bity z trzech kolejnych pozycji binarnych obu składników kolejnej reszty chwilowej. W pierwszym kroku dzielenia analizuje się oczywiście trójkę bitów dzielnej lub trójki bitów ze składników dzielnej, przy czym bierze się je z dwóch pozycji binarnych przed przecinkiem i jednej po przecinku /patrz rys. 1b/. Analizowane trójki bi-

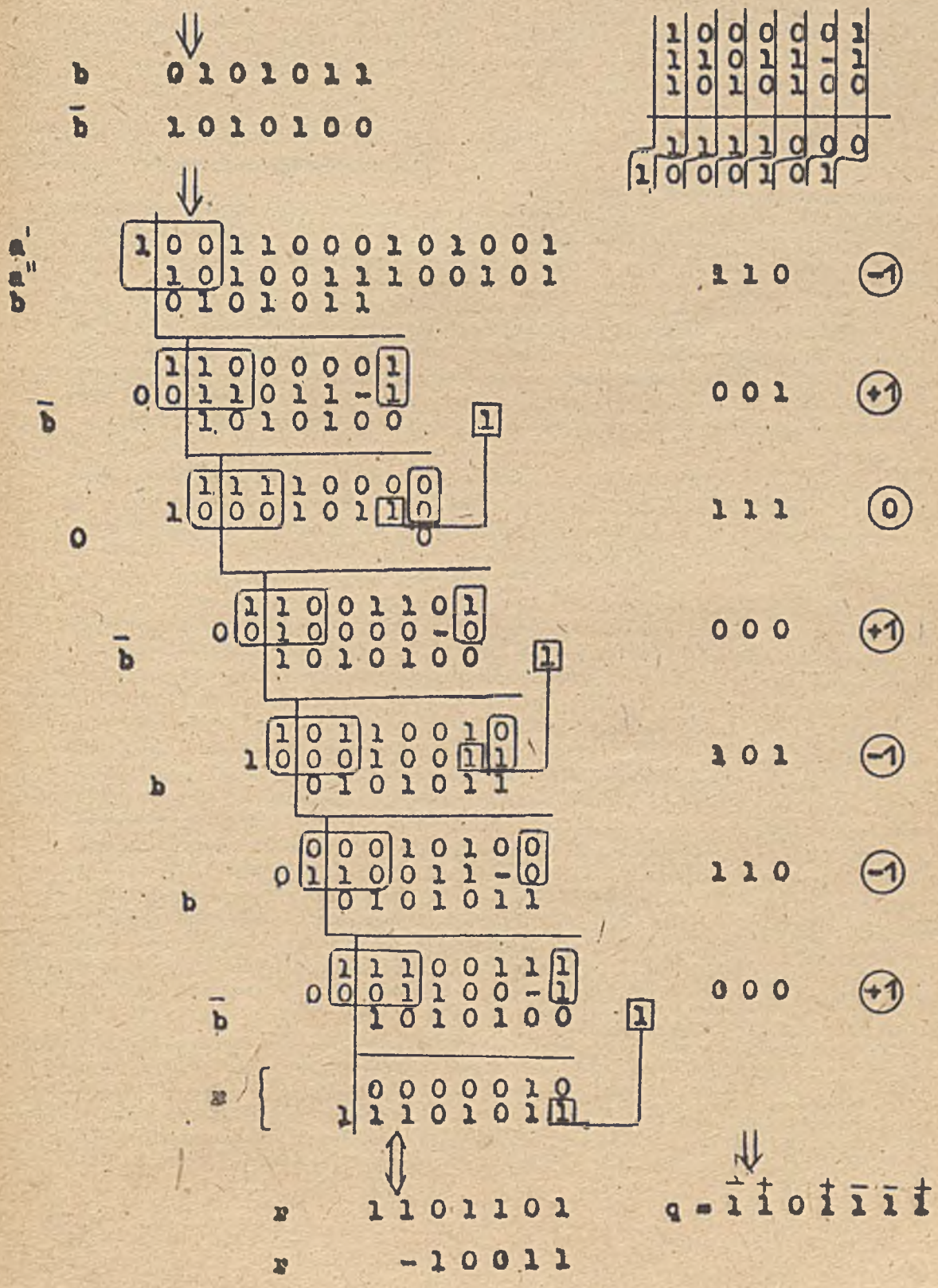

b 0 1 0 1 0 1 1
b̄ 1 0 1 0 1 0 0
a'
a''
b
b̄
b
b̄
b
b̄
z
x 1 1 0 1 1 0 1
x -1 0 0 1 1
q = 1 1 0 1 1 1 1
1 1 0
0 0 1
1 1 0
0 0 1
1 1 0

Рис. 2

tów można traktować jako dwie liczby trójbitowe, które po dodaniu modulo 2^3 dają jedną liczbę trójbitową, reprezentującą przybliżoną wartość dzielnej lub kolejnej reszty chwilowej w zapisie binarnym uzupełnieniowym. Spełnia ona warunek

$$-2 \leq a_i < 2 \qquad /17/$$

Otrzymanym w ten sposób liczbom trójbitowym odpowiadają /zgodnie z rys. 1b/ następujące cyfry ilorazu

10,x	-1
11,0	-1
11,1	0
0x,x	+1

/18/

przy czym bity oznaczone przez x nie mają wpływu na dekodowaną cyfrę. Tabela /18/ obowiązuje, jak już wcześniej wspomniano, dla dzielnika dodatniego. W przypadku dzielnika ujemnego należy zmienić tylko znaki cyfr ilorazu na przeciwne.

Zamiast dodawania dwóch liczb modulo 2^3 i dekodowania otrzymanej w wyniku trójki bitów zgodnie z /18/, można oczywiście dekodować od razu dwie trójki bitów. Odpowiednią tabelę obejmującą 64 badane kombinacje zerojedynkowe można wypisać bezpośrednio na podstawie /18/. Nie będziemy jej tutaj podawać.

Dla uniknięcia dekodowania dodatkowo znaku dzielnika przy wyznaczaniu cyfr ilorazu można w przypadku dzielnika ujemnego przyjąć na początku dzielenia negację dzielnej zamiast dzielnej, uwzględniając oczywiście korekcyjną dodatnią jedynkę na najmniej znaczącej pozycji dzielnej.

Kolejne reszty, mieszczące się w zakresie określonym relacją /14/, są już odpowiednio przesunięte względem dzielnika. Zakres kolejnych reszt przed przesunięciem jest dwukrotnie mniejszy w stosunku do dzielnika, z czego wynika, że również reszta końcowa spełnia warunek

$$-|b| \leq a_n \leq |b| \qquad /19/$$

Jeśli na resztę końcową narzucony jest inny warunek, dzielenie wymaga dodatkowego kroku korekcyjnego lub kilku dodatkowych kroków korekcyjnych.

W podanym dotychczas opisie występuje pewna formalna nieścisłość. Powinno się mianowicie wyraźniej odróżniać chwilową resztę "przesuniętą" a_i od chwilowej reszty $a_i 2^{-i}$ /jak również dzielnik b_i od dzielnika "przesuniętego" $b_i 2^{-i}$/. Opis algorytmu byłby wtedy jednak mniej przejrzysty dla projektanta układów cyfrowych, który przyzwyczajony jest do tego, że w układzie dzielenia operuje się zwykle przesuniętą resztą chwilową i nie przesuniętym dzielnikiem. Formalnie w większości podanych wzorów należałoby dopisywać przy liczbach a_i, b odpowiednie potęgi dwójki /pozwoliłoby to między innymi uniknąć niezgodności zakresów we wzorach /14/ i /19//.

Opisany algorytm dzielenia o cyfrach ilorazu -1,0,+1, ilustruje podany na rys. 2 przykład liczbowy. Symbolami a', a", r oznaczono na nim składniki dzielnej i resztę, symbolami b i $\bar{b}$ oznaczono dzielnik i jego negację, a symbolami q i r iloraz i resztę końcową. Strzałkami podwójnymi pokazano miejsce przecinków binarnych zgodnie z ustalonymi zakresami dzielnej i dzielnika. W przypadku pominięcia przecinków można traktować ten przykład jako przykład dzielenia liczb całkowitych, gdzie dzielna a=1266, jej składniki a'=-6615, a"=8349, dzielnik b=+43, iloraz q=29, a dwuskładnikowa reszta końcowa r=19. W przypadku istnienia przecinków w miejscach pokazanych przez podwójne strzałki, pierwsze trzy z tych liczb są $2^{12}=4096$ razy mniejsze, a pozostałe trzy liczby $2^6=64$ razy mniejsze. Kolejne trójki liczb redukowanych w kolejnych krokach dzielenia rozdzielono na rys. 2 kreskami poziomymi, a bity redukowanych liczb, nie brane pod uwagę przy redukcji jako nie mające już wpływu na dalszy przebieg dzielenia, oddzielono kreskami pionowymi. Pary trójek bitów składników dzielnej i pary trójek bitów składników kolejnych reszt chwilowych, od których należą cyfry ilorazu i wybór trzeciej redukowanej liczby objęto linią zamkniętą.

W ten sam sposób zaznaczono pojedyncze pary bitów ze składników dzielnej dopisywane w kolejnych krokach do reszt chwilowych. W przypadku, gdy trzecią z redukowanych liczb jest negacja dzielnika, przy redukcji uwzględnia się jedynkę korekcyjną na najmniej znaczącej pozycji binarnej. Takie korekcyjne jedynki pokazano na rys. 2 w małych kwadracikach dwukrotnie: raz z prawej strony zanegowanych dzielników, drugi raz we wskazanych pojedynczymi strzałkami miejscach, w których zostaną wzięte pod uwagę przy kolejnej redukcji. Sposób redukowania trzech liczb do

dwóch liczb wyjaśnia powtórzona w prawej górnej części rys. 2 redukcja z drugiego kroku dzielenia. Pokazano tam jakie pary bitów liczb po redukcji przyporządkowuje się poszczególnym trójkom bitów liczb sprzed redukcji. Dwustopniowe wyznaczanie cyfr ilorazu na podstawie par trójek bitów objętych na rys. 2 liniami zamkniętymi pokazano z prawej strony tablicy. W wyniku dodania dwóch liczb trójbitowych modulo 2^3, otrzymuje się mianowicie przedstawione z prawej strony rysunku trójki bitów, którym następnie przyporządkowuje się podane w kółkach cyfry $-1,0,+1$ ilorazu. Omówiony sposób wyznaczania cyfr ilorazu można oczywiście zastąpić jednostopniowym wyznaczaniem cyfr w większym dekoderze, o sześciobitowym wejściu. Otrzymanym cyfrom $-1,0,+1$ ilorazu przyporządkowuje się w następnym kroku dzielenia, w przypadku dodatniego dzielnika, jako trzeci redukowany składnik odpowiednio: dzielnik, zero i negację dzielnika. Redundancyjny zapis ilorazu można ewentualnie przekształcić w zwykły zapis binarny, co zostanie opisane w rozdz. 9.

6. ANALOGIE Z KLASYCZNYM ALGORYTMEM DZIELENIA

Dla pokazania niektórych różnic między opisanym w poprzednich rozdziałach algorytmem dzielenia o cyfrach ilorazu $-1,0,+1$, a ogólnie znanym klasycznym algorytmem binarnego dzielenia, oraz dla uwypuklenia pewnych cech charakterystycznych opisanego algorytmu dzielenia o cyfrach ilorazu $-1,0,+1$, omówimy sposób modyfikacji powszechnie stosowanego algorytmu dzielenia binarnego nieresztytucyjnego, prowadzący do otrzymania opisanego algorytmu dzielenia o cyfrach $-1,0,+1$.

Klasyczny algorytm binarnego dzielenia nieresztytucyjnego, opisany między innymi w [1], obejmuje sekwencję odejmowań od dzielnej i dodawań do dzielnej dzielnika, przesuwanego po każdej z tych operacji o jedną pozycję binarną w prawo względem dzielnej, co zresztą jest zwykle realizowane w układach cyfrowych przez przesuwanie otrzymywanych w wyniku tych działań kolejnych reszt chwilowych w lewo. Cyfry 0,1 otrzymywanego w ten sposób ilorazu zależą przy tym, dla danego znaku dzielnika, od znaków kolejnych reszt.

Jeśli chcielibyśmy wynik dzielenia otrzymać nie w zwykłym zapisie binarnym, ale w zapisie pozycyjnym o podstawie rozwinięcia 2 i cyfrach $-1,+1$, który jest również zapisem jednoznacznym nieredundancyjnym, należałoby tylko sekwencji odejmowań i dodawań dzielnika, określonej wspomnianym klasycznym algorytmem dzielenia nieresztytucyjnego, przyporządkować odpowiednią sekwencję cyfr $-1,+1$ /w zależności od znaku dzielnika/. Uwzględnienie tego wyniku wprost w wzorów /2/ i /3/ w rozdziale 3 dla $q_i \in \{-1,+1\}$.

Rozważmy teraz taką modyfikację omawianego algorytmu nieresztytucyjnego, w której w niektórych krokach dzielenia występowałoby tylko przesunięcie kolejnej reszty względem dzielnika, bez wykonywania odejmowania lub dodawania. Takim zmodyfikowanym krokom dzielenia należałoby przyporządkować cyfry $q_i=0$, co również wynika z wzorów /2 i /3/. Nasuwają się pytania: czy i kiedy można wprowadzić takie kroki dzielenia i jaki cel można przez to uzyskać.

Jak łatwo zauważyć kroki dzielenia polegające na przesunięciu reszty i wyznaczeniu cyfry $q_i=0$ można stosować tylko wtedy, gdy po ich wykonaniu zostaje zachowany warunek /10/ z rozdziału 4, wynikający z wymagań na zbieżność procesu dzielenia. Zachodzi to, jak już wiadomo, w przypadku, gdy spełniona jest druga z nierówności /11/.

Wprowadzenie trzech rodzajów kroków dzielenia i odpowiadających im trzech różnych cyfr ilorazu $-1,0,+1$, komplikuje dość znacznie algorytm dzielenia i realizujące ten algorytm układy cyfrowe. Rekompensatą tego może być tylko zwiększenie szybkości wykonywania dzielenia. W tradycyjnym algorytmie binarnego dzielenia nieresztytucyjnego wynik odejmowania lub dodawania w kolejnym kroku dzielenia decyduje o wyborze jednej z tych operacji dla następnego kroku. W przypadku dzielenia liczb o dużej precyzji czas wykonania tych operacji, ze względu na propagację przeniesień, może być dość długi. W zmodyfikowanym algorytmie o cyfrach ilorazu $-1,0,+1$ do określenia następnej operacji nie jest konieczna dokładna znajomość wyniku poprzedniej operacji, tylko jego odpowiednie oszacowanie. Dzięki temu pełne wykonanie operacji odejmowania lub dodawania nie jest niezbędne i może być zastąpione mniej czasochłonnymi przekształceniami równoległymi, takimi jak równoległa redukcja składników z zachowaniem ich sumy. Celem zastosowania algorytmu wyznaczania ilorazu w zapisie redundancyjnym o cyfrach $-1,0,+1$ jest więc skrócenie czasu dzielenia uzyskane przez zastąpienie efektywnego wykonywania odejmowań i dodawań w kolejnych krokach dzielenia mniej czasochłonną redukcją równoległą składników.

7. ALGORYTM DZIELENIA O CYFRACH ILORAZU 0,1,2

Zajmiemy się obecnie dzieleniem liczb przedstawionych w zapisie binarnym z ilorazem w zapisie redundancyjnym o podstawie rozwinięcia 2 i cyfrach 0,1,2.

Zgodnie z założeniami przyjętymi na początku pracy, w kolejnych krokach dzielenia odbywa się redukcja równoległa trzech liczb do dwóch liczb z zachowaniem ich sumy. Pierwsze dwie z trzech redukowanych liczb są składnikami kolejnej reszty chwilowej /w pierwszym kroku składnikami dzielnej, lub negacjami tych składników uzupełnionymi korekcyjnymi jedynkami na najmniej znaczącej pozycji/. Trzecia z redukowanych liczb może być zerem, dzielnikiem, lub jego negacją /z korekcyjną jedynką/. Z założenia, że iteracyjny proces wyznaczania cyfr 0,1,2 ilorazu przebiega zgodnie z równaniem /2/ i że zbieżności tego procesu wynika jednakowy znak sumy dwóch pierwszych redukowanych liczb we wszystkich krokach dzielenia i również ustalony, przeciwny znak trzeciej redukowanej liczby /jeśli nie jest ona zerem/.

Omówimy kolejno dwa algorytmy dzielenia, najpierw przy założeniu dodatnich kolejnych reszt, następnie ujemnych. Algorytmy te różnią się od siebie dość istotnie. Wynika to stąd, że przy przyjętym zapisie uzupełnieniowym liczb, odcięta część mniej znaczących bitów składników kolejnej reszty przedstawia zawsze liczbę nieujemną, niezależnie od znaku całej reszty, ma więc znak zgodny z bardziej znaczącą częścią reszty w przypadku pierwszego z algorytmów, a przeciwny w przypadku drugiego z nich.

Dla pierwszego algorytmu przyjęliśmy zatem, że suma dwóch pierwszych redukowanych liczb jest dodatnia, natomiast trzecia liczba jest ujemna /jeśli nie jest zerem/.

Zakładając zbieżność iteracyjnego procesu wyznaczania cyfr ilorazu /tzn. przyjmując ograniczenie zakresu wartości liczb a_i/ otrzymujemy z równania /2/ dla cyfr $q_i \in \{0,1,2\}$ warunek

$$0 \leq a_i \leq 4\,|b| \qquad /20/$$

Przy powyższym założeniu na podstawie /2/ i /20/ trzem następującym zakresom wartości kolejnej reszty chwilowej

$$0 \leq a_i \leq 2\,|b| \qquad |b| \leq a_i \leq 3\,|b| \qquad 2\,|b| \leq a_i \leq 4\,|b| \qquad /21/$$

można przyporządkować odpowiednio cyfry ilorazu 0,1,2.

Dla dodatniego dzielnika z zakresu

$$\frac{1}{2} \leq b < 1 \qquad /22/$$

wyznaczenie kolejnej cyfry ilorazu odbywa się na podstawie bitów wziętych z czterech pozycji binarnych składników dzielnej lub kolejnej reszty, w tym z dwóch pozycji binarnych przed przecinkiem i z dwóch pozycji binarnych po przecinku, oraz na podstawie dwóch bitów dzielnika, a mianowicie drugiego i trzeciego bitu po przecinku. W podanej niżej tablicy przyporządkowano cyfry 0,1,2 ilorazu liczbom czterobitowym, otrzymanym w wyniku dodawania modulo 2^4 dwóch liczb czterobitowych wybranych jako czwórki bitów ze składników dzielnej lub składników kolejnej reszty, oraz podkreślonym w tablicy parom bitów dzielnika. Bity oznaczone w tablicy przez x nie mają wpływu na wybór cyfry ilorazu

a_i / b	00,0x	00,10	00,11	01,00	01,01	01,1x	1x,xx	
0,1<u>00</u>	0	0	1	1	2	2	2	1,0<u>11</u>
0,1<u>01</u>	0	0	1	1	1	2	2	1,0<u>10</u>
0,1<u>1x</u>	0	0	0	1	1	1	2	1,<u>00x</u>

W wyniku dzielenia wykonywanego zgodnie z opisanym algorytmem otrzymuje się dwuskładnikową resztę z zakresu

$$0 \leq a_n \leq 2\,|b| \qquad /23/$$

Zmniejszenie tego zakresu może wymagać wykonania jednego lub kilku dodatkowych kroków dzielenia obejmujących również korektę ilorazu.

Dla ujemnego dzielnika z zakresu

$$-1 < b < -\frac{1}{2} \qquad /24/$$

wyznaczenie kolejnej cyfry ilorazu może odbywać się na podstawie tej samej podanej wyżej tablicy, przy czym zamiast badanych bitów dzielnika należy wziąć pod uwagę ich negacje. Czwórki najbardziej znaczących bitów ujemnego dzielnika odpowiadające poszczególnym wierszom podanej tablicy podano na prawo od tablicy, podkreślając badane bity.

Wszystkie powyższe stwierdzenia odnoszące się do dzielnika ujemnego dotyczą oczywiście dzielnika w zapisie uzupełnieniowym. W przypadku dzielnika ujemnego w postaci znak-moduł postępuje się oczywiście identycznie jak dla dzielnika dodatniego.

Analizowania bitów dzielnika można uniknąć, zawężając jego zakres, w przypadku dodatniego dzielnika do

$$\frac{3}{4} \leq b < 1 \qquad /25/$$

a w przypadku ujemnego dzielnika do

$$-1 < b \leq -\frac{3}{4} \qquad /26/$$

czyli do przypadków, gdy trójka najbardziej znaczących bitów ma postać

0,11 lub 1,00

Analogiczne zawężanie zakresu dzielnika omówione jest przez A. Witkowskiego w [33]. Wyznaczenie kolejnej cyfry ilorazu odbywa się wtedy wyłącznie na podstawie bitów wziętych z czterech pozycji binarnych składników dzielnej lub kolejnej reszty. Poniżej przyporządkowano cyfry ilorazu liczbom czterobitowym otrzymanym w wyniku dodawania modulo 2^4 dwóch liczb czterobitowych reprezentowanych przez czwórki bitów wzięte ze składników dzielnej lub składników kolejnej reszty.

00,xx	0
01,xx	1
1x,xx	2

Zawężenie zakresu dzielnika umożliwia zatem nie tylko ograniczenie się do analizowania grupy bitów kolejnych reszt, ale upraszcza ponadto znacznie tę analizę.

Zmiany zakresu dzielnika z /22/ lub /24/ na /25/ lub /26/ można dokonać mnożąc dzielną i dzielnik przez $\frac{3}{4}$ lub $\frac{3}{2}$, czyli dodając do dzielnej i dzielnika dzielną i dzielnik przesunięte o jedną lub dwie pozycje binarne w prawo. Z uwagi na możliwość stosowania dwuskładnikowej dzielnej dodawanie odpowiednio przesuniętej dzielnej można zastąpić dopisaniem przesuniętej dzielnej jako drugiego składnika.

Mnożenie dzielnej i dzielnika przez $\frac{3}{4}$ stosuje się jeśli czwórka najbardziej znaczących bitów dzielnika ma postać

0,101 lub 1,010

a mnożenie przez $\frac{3}{2}$ stosuje się dla tejże czwórki bitów dzielnika w postaci

0,100 lub 1,011.

Warto wzrócić uwagę, że w konsekwencji pomnożenia dzielnej i dzielnika przez $\frac{3}{4}$ lub $\frac{3}{2}$ otrzymuje się w wyniku dzielenia również resztę pomnożoną przez odpowiednią z tych liczb.

Przejdziemy obecnie do drugiego algorytmu dzielenia, w którym suma dwóch pierwszych redukowanych liczb jest ujemna, a trzecia liczbą nieujemną.

W miejsce wzorów /20/ - /23/ otrzymujemy dla tego algorytmu wzory

$$-4\,|b| \leq a_i \leq 0 \qquad /27/$$

$$-2\,|b| \leq a_i \leq 0 \qquad -3\,|b| \leq a_i \leq -|b| \qquad -4\,|b| \leq a_i \leq -2\,|b| \qquad /28/$$

$$\frac{1}{2} \leq b < 1 \qquad /29/$$

$$-2\,|b| \leq a_n \leq 0 \qquad /30/$$

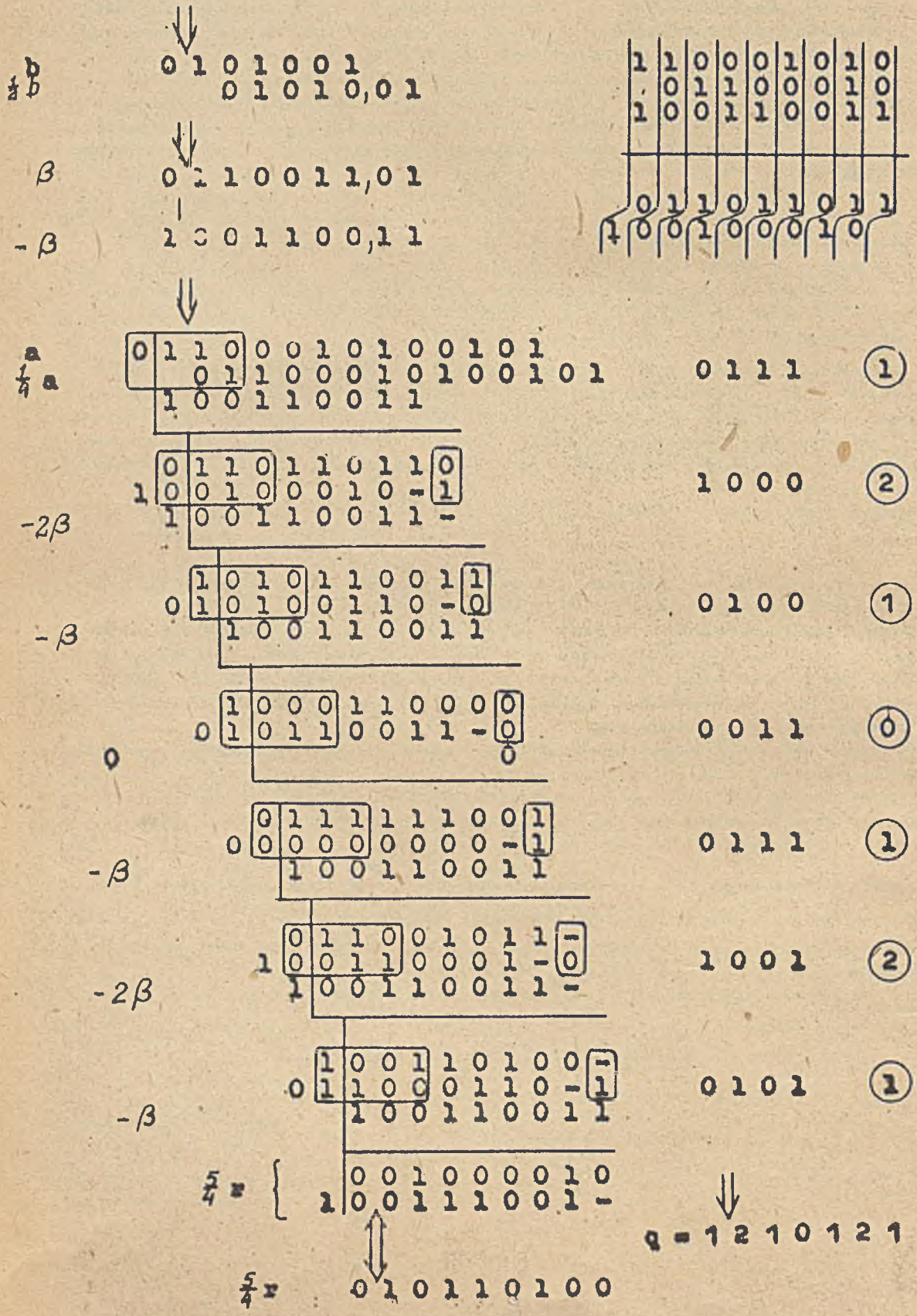

b
½b
0 1 0 1 0 0 1
0 1 0 1 0,0 1
β
0 1 1 0 0 1 1,0 1
-β
1 0 0 1 1 0 0,1 1
a
¼a
0 1 1 0 0 0 1 0 1 0 0 1 0 1
0 1 1 0 0 0 1 0 1 0 0 1 0 1
1 0 0 1 1 0 0 1 1
-2β
-β
0
-β
-2β
-β
5/4 x
0 0 1 0 0 0 0 1 0
1 0 0 1 1 1 0 0 1 -
5/4 x
0 1 0 1 1 0 1 0 0
0 1 1 1
1
1 0 0 0
2
0 1 0 0
1
0 0 1 1
0
0 1 1 1
1
1 0 0 1
2
0 1 0 1
1
q = 1 2 1 0 1 2 1

Рис. 3

a w miejsce tablicy podanej poprzednio tablicę

b \ a_1	1,11,xx	1,10,1x	1,10,01	1,10,00	1,01,11	1,01,10	1,01,0x	1,00,xx	
0,100	0	1	2	2	2	2	2	2	1,011
0,101	0	1	1	2	2	2	2	2	1,010
0,11x	0	0	1	1	1	2	2	2	1,00x

w której, podobnie jak poprzednio, przez x oznaczono bity nie mające wpływu na wynik dekodowania. Analizowane bity w tablicy podkreślono.

Dla zawężonego zakresu dzielnika, a mianowicie zakresu

$$\frac{3}{4} \leq b < 1 \qquad /31/$$

lub

$$-1 < b < \frac{3}{4} \qquad /32/$$

czyli dla przypadków, gdy trójka najbardziej znaczących bitów ma postać

0,11 lub 1,00

wyznaczenie kolejnej cyfry ilorazu odbywa się, podobnie jak w poprzednio opisanym algorytmie, wyłącznie na podstawie bitów wziętych z czterech pozycji binarnych składników dzielnej lub kolejnej reszty. Poniżej przyporządkowane cyfry ilorazu liczbom czterobitowym otrzymanym w wyniku dodawania modulo 2^4 dwóch liczb czterobitowych reprezentowanych przez czwórki bitów wziętych ze składników dzielnej lub składników kolejnej reszty. Liczby te podkreślono, dopisując dodatkowo z lewej strony jedynkę symbolizującą ujemny bit znakowy analizowanej części dzielnej lub kolejnej reszty.

1,11,xx	0
1,10,1x	0
1,10,0x	1
1,01,11	1
1,01,10	2
1,01,0x	2
1,00,xx	2

Porównując opisane dwa algorytmy /o cyfrach 0,1,2 ilorazu/ można stwierdzić, że algorytm dzielenia o dodatniej sumie dwóch pierwszych redukowanych liczb jest algorytmem prostszym i pod każdym względem lepszym od algorytmu o ujemnych resztach chwilowych. Oba te algorytmy są jednak bardziej złożone od omawianego w poprzednich rozdziałach algorytmu dzielenia o cyfrach ilorazu -1,0,+1. Nie wydaje się zatem celowe konstruowanie opartych na nich układów cyfrowych przeznaczonych wyłącznie do dzielenia. Okazuje się jednak, że algorytmy takie, po odpowiednich modyfikacjach ujawniają pewne zalety w przypadku wykorzystania ich w konstrukcji układów cyfrowych służących /oprócz dzielenia/ do wyznaczania wartości niektórych funkcji elementarnych metodą cyfra po cyfrze /patrz [11] i [12]/.

Opisany algorytm dzielenia o cyfrach 0,1,2 ilorazu oraz o dodatnich kolejnych resztach chwilowych, ilustruje podany na rys. 3 przykład liczbowy. Strzałkami podwójnymi oznaczono na nim, podobnie jak na rys. 2, miejsca przecinków binarnych, zgodnie z ustalonymi zakresami dzielnej i dzielnika. Można jednak pominąć przecinki i traktować ten przykład jako przykład dzielenia liczb całkowitych, mianowicie dzielenia całkowitej dodatniej dzielnej a_0=6309 przez dzielnik b_0=41. Z uwagi na to, że trójką najbardziej znaczących bitów wartości bezwzględnej tego dzielnika /0101001/ ma postać 101, dzielenie poprzedzone jest /w celu zawężenia zakresu dzielnika/ wyznaczeniem $\frac{3}{4}$-krotności dzielnika, oznaczonej na rys. 3 przez β_1, gdzie $\beta = \frac{3}{4}b$. W wyniku dzielenia otrzymuje się iloraz q=153 oraz $\frac{3}{4}$-krotność reszty $\frac{3}{4}r$ =45, co odpowiada wartości reszty r=36. Wszystkie występujące na rys. 3 oznaczenia graficzne, takie jak linie i strzałki zostały opisane już w rozdziale 5 w odniesieniu do rys. 2. W górnym prawym rogu rys. 3 pokazano sposób redukowania trzech liczb do dwóch dla redukcji z pierwszego kroku dzielenia.

8. ALGORYTM DZIELENIA O CYFRACH ILORAZU OD -3 do +3

Zajmiemy się obecnie dzieleniem liczb przedstawionych w zapisie ósemkowym kodowanym binarnie, w którym iloraz przedstawiony jest w zapisie redundancyjnym o podstawie rozwinięcia 4 i cyfrach $-3,-2,-1,0,+1,+2,+3$.

Na podstawie równania /6/ dla cyfr $q_i \in \{-3,-2,-1,0,+1,+2,+3\}$, zakładając zbieżność iteracyjnego procesu wyznaczania cyfr ilorazu /czyli przyjmując ograniczenie zakresu wartości liczb a_i/, otrzymujemy warunek

$$|a_i| \leq 4\,|b| \qquad /33/$$

Przyjmując podobnie jak w poprzednich rozdziałach zakres dzielnika

$$\frac{1}{2} \leq |b| < 1 \qquad /34/$$

otrzymujemy na podstawie /6/ i /33/ następujące zakresy kolejnej reszty chwilowej oraz cyfry ilorazu jakie mogą być przyporządkowane tym zakresom

$$\left.\begin{array}{ll}
-4\,|b| \leq a_i \leq -2\,|b| & -3 \\
-3\,|b| \leq a_i \leq -|b| & -2 \\
-2\,|b| \leq a_i \leq 0 & -1 \\
-|b| \leq a_i \leq |b| & 0 \\
0 \leq a_i \leq 2\,|b| & +1 \\
|b| \leq a_i \leq 3\,|b| & +2 \\
2\,|b| \leq a_i \leq 4\,|b| & +3
\end{array}\right\} \qquad /35/$$

Podana kolejność przyporządkowania cyfr ilorazu poszczególnym zakresom kolejnych reszt obowiązuje dla dzielnika dodatniego. W przypadku dzielnika ujemnego znaki cyfr ilorazu należy zmienić na przeciwne. Pamiętając o tej różnicy przyjmiemy w dalszym ciągu niniejszego rozdziału, że dzielnik b jest dodatni i spełnia warunek

$$\frac{1}{2} \leq b < 1 \qquad /36/$$

co oczywiście nie ogranicza zakresu stosowalności algorytmu.

Przy podanych założeniach, postępując analogicznie jak to opisano dla ilorazu o cyfrach $-1,0,+1$, otrzymujemy niżej opisany algorytm dzielenia.

Wyznaczenie kolejnej cyfry ilorazu odbywa się na podstawie bitów wziętych z pięciu pozycji binarnych składników dzielnej lub składników kolejnej reszty chwilowej, w tym z trzech pozycji binarnych przed przecinkiem i z dwóch pozycji binarnych po przecinku oraz na podstawie dwóch bitów dzielnika, mianowicie drugiego i trzeciego bitu po przecinku. W podanej niżej tablicy przyporządkowano cyfry ilorazu liczbom pięciobitowym, otrzymanym w wyniku dodawania modulo 2^5 dwóch liczb pięciobitowych wybranych jako piątki bitów ze składników dzielnej lub składników kolejnej reszty, oraz podkreślonym w tablicy parom bitów dzielnika.

b \ a_i	100,xx	101,0x	101,10	101,11	110,00	110,01	110,1x	111,0x	111,1x	000,0x	000,10	000,11	001,00	001,01	001,1x	01x,xx
0,1<u>00</u>	-	-	-3	-3	-3	-3	-2	-1	0	1	1	2	2	3	3	3
0,1<u>01</u>	-	-3	-3	-3	-3	-2	-2	-1	0	1	1	$\frac{1}{2}$	2	2	3	3
0,1<u>1</u>x	-3	-3	-3	-2	-2	-2	-1	-1	$\frac{0}{0}$	$\frac{0}{1}$	1	1	$\frac{1}{2}$	2	2	3

Bity oznaczone w tablicy przez x nie mają wpływu na wybór cyfr ilorazu.

W wyniku dzielenia wykonywanego zgodnie z opisanym obecnie algorytmem otrzymuje się dwuskładnikową resztę z zakresu

$$-b \leq a_n \leq b \qquad /37/$$

Jeśli na resztę końcową narzucony jest inny warunek, dzielenie wymaga wykonania jednego lub kilku dodatkowych kroków korekcyjnych.

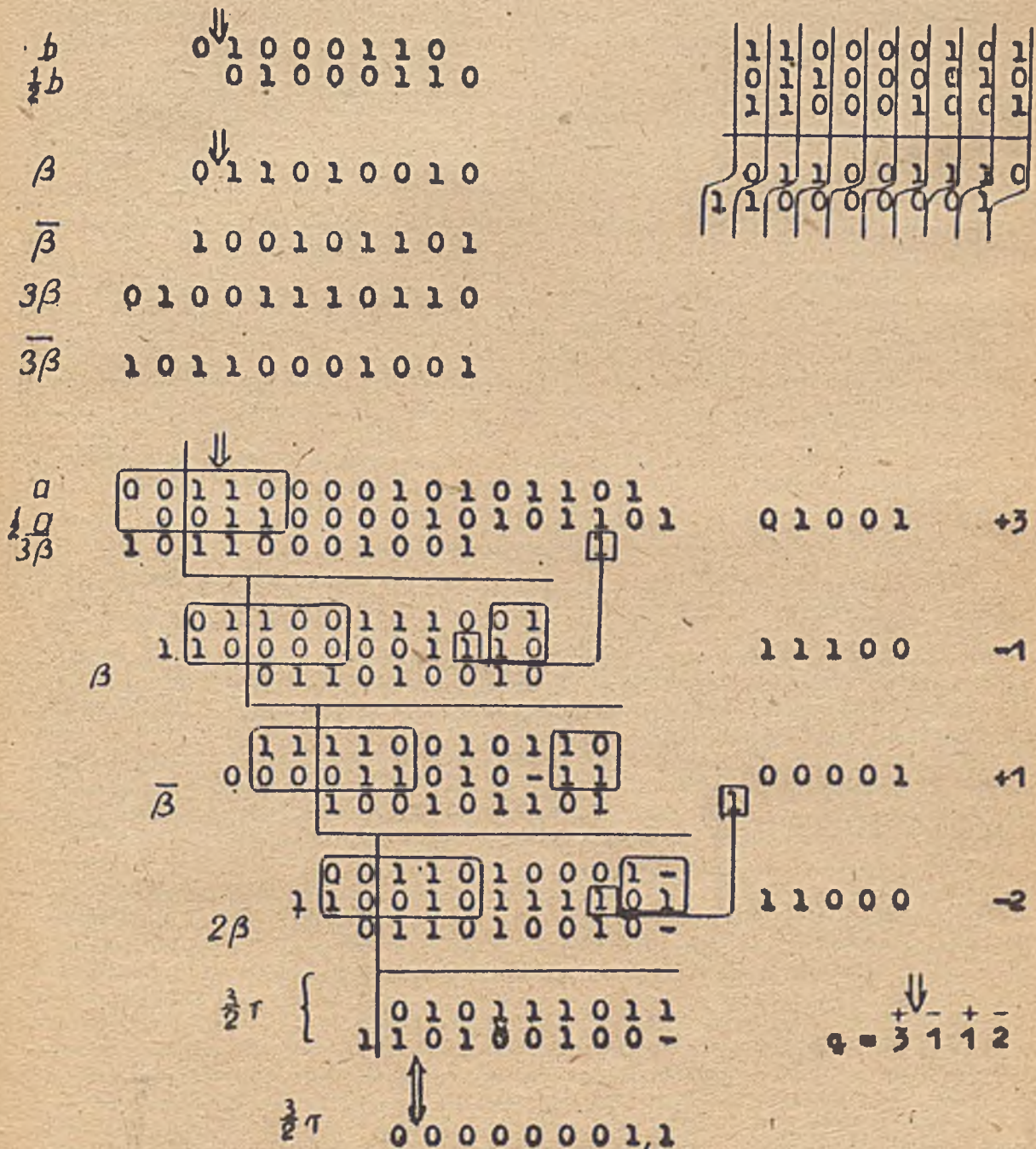

b
½b
β
β̄
3β
3β̄
a
½a
3β
β
β̄
2β
½τ
½τ
Q 1 0 0 1
+3
1 1 1 0 0
-1
0 0 0 0 1
+1
1 1 0 0 0
-2
q = 3 1 1 2
0 0 0 0 0 0 0 1,1

Рис. 4

Badania bitów dzielnika można uniknąć zawężając jego zakres, w przypadku dodatniego dzielnika, do

$$\frac{3}{4} \leqslant b < 1$$

czyli do przypadku, gdy trójka najbardziej znaczących bitów ma postać 0,11, co zostało już opisane szczegółowo w rozdz. 7.

Wyznaczenie kolejnej cyfry ilorazu odbywa się wówczas wyłącznie na podstawie bitów wziętych z pięciu pozycji binarnych składników dzielnej lub składników kolejnej reszty chwilowej. W podanej niżej tablicy, otrzymanej na podstawie dolnego wiersza poprzedniej tablicy, przyporządkowano cyfry ilorazu liczbom pięciobitowym, otrzymanym w wyniku dodawania modulo 2^5 dwóch piątek bitów, wziętych ze składników dzielnej lub kolejnej reszty i traktowanych jako liczby pięciobitowe.

100,xx	101,0x	101,10	101,11	110,0x	110,1x	111,0x	111,1x	000,xx	001,xx	01x,xx
-3	-3	-3	-2	-2	-1	-1	0	1	2	3

Opisany algorytm dzielenia o cyfrach ilorazu od -3 do +3 ilustruje podany na rys. 4 przykład liczbowy.

Strzałkami podwójnymi oznaczono na rys. 4 podobnie jak na rys. 2 i 3 miejsca przecinków binarnych, zgodnie z ustalonymi zakresami dzielnej i dzielnika. Można jednak pominąć przecinki i rozpatrywać ten przykład ja[illegible]przykład dzielenia liczb całkowitych, a mianowicie dzielenia całkowitej dodatniej dzielnej a = +12461 przez dzielnik b=+70. Z uwagi na to, że trójka najbardziej znaczących bitów tego dzielnika /01000110/ ma być doprowadzona do postaci 011, dzielenie poprzedzone jest wyznaczeniem $\frac{1}{2}$ -krotności dzielnika, oznaczonej na rys. 4 przez β, gdzie $\beta = \frac{1}{2}b$. W wyniku dzielenia otrzymuje się iloraz q=+178 oraz $\frac{1}{2}$ -krotność reszty $\frac{1}{2}r = +1,5$, co odpowiada wartości reszty r=+1. Występujące na rys. 4 oznaczenia graficzne, takie jak linie i strzałki, zostały opisane już w rozdz. 5 w odniesieniu do rys. 2. W górnym prawym rogu rys. 4 pokazano sposób redukowania trzech liczb do dwóch, dla redukcji z pierwszego kroku dzielenia.

9. PRZEKSZTAŁCENIE ZAPISU ILORAZU W BINARNY ZAPIS NIEREDUNDANCYJNY

Wykonanie dzielenia według opisanych w poprzednich rozdziałach algorytmów prowadzi do otrzymania ilorazu w zapisie o cyfrach -1,0,+1 lub 0,1,2 i podstawie rozwinięcia 2, albo o cyfrach -3,-2,-1,0,+1,+2,+3 i podstawie rozwinięcia 4.

Przekształcenie takich zapisów redundancyjnych w zapis binarny nieredundancyjny można sprowadzić, jak już wspomniano w rozdz. 2, do dodawania binarnego liczb w zapisie nieredundancyjnym. Czas takiego dodawania, obejmującego proces propagacji przeniesień, powoduje oczywiście odpowiednie zwiększenie czasu wykonania całego dzielenia.

Przekształcenie zapisu ilorazu w binarny zapis nieredundancyjny może być również wykonane na bieżąco, w trakcie otrzymywania cyfr ilorazu, począwszy od cyfry najbardziej znaczącej, na przykład w sposób opisany przez D.E. Atkinsa w [4]. Otrzymanie ilorazu w nieredundancyjnej postaci binarnej jest wówczas możliwe w zasadzie bez wydłużania czasu dzielenia.

Poniżej omówiono oba sposoby przekształceń.

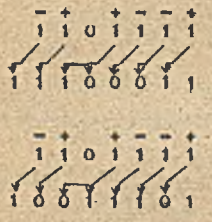
Załóżmy, że każda z cyfr -1,0,+1 oraz cyfr 0,1,2 ilorazu jest zakodowana w postaci pary bitów na tej samej pozycji binarnej, a każda z cyfr -3,-2,-1,0,+1,+2,+3 ilorazu jest zakodowana w postaci dwóch par bitów na dwóch kolejnych pozycjach binarnych, w ten sposób, że każdemu bitowi na określonej pozycji binarnej przyporządkowana jest określona z wag +1,-1. Założenie takie jest w zasadzie równoznaczne z przyjęciem, że iloraz zapisany jest w postaci dwóch liczb dodatnich w binarnym zapisie nieredundancyjnym i jest równy sumie tych liczb w przypadku cyfr 0,1,2 oraz różnicy tych liczb w przypadku cyfr -1,0,+1 i -3,-2,-1,0,+1,+2,+3. Otrzymanie ilorazu w binarnym zapisie nieredundancyjnym wymaga wówczas wykonania dodawania tych liczb w przypadku cyfr 0,1,2 oraz ich odejmowania w przypadku cyfr -1,0,+1 i -3,-2,-1,0,+1,+2,+3.

Omówimy obecnie przekształcenie zapisu ilorazu w binarny zapis nieredundancyjny na bieżąco, w trakcie otrzymywania cyfr ilorazu, począwszy od cyfry najbardziej znaczącej, najpierw dla cyfr ilorazu -1,0,+1, następnie dla cyfr 0,1,2 i cyfr -3,-2,-1,0,+1,+2,+3.

Przekształcenie zapisu ilorazu o cyfrach -1,0,+1 w zapis binarny uzupełnieniowy odbywa się następująco. Znak najbardziej znaczącej jedynki jest znakiem ilorazu. Najbardziej znaczącej jedynce dodatniej odpowiada jedynka na tej samej pozycji binarnej i ciąg zer na wszystkich bardziej znaczących pozycjach, a najbardziej znaczącej jedynce ujemnej odpowiada jedynka na tej samej pozycji binarnej i ciąg jedynek na wszystkich bardziej znaczących pozycjach. Z kolei bierze się pod uwagę następną jedynkę w ciągu. Jeśli jest ona dodatnia wszystkie bity bardziej znaczące od niej są już bitami ilorazu, a jeśli ujemna wszystkie zera na pozycjach bardziej znaczących, aż do najbliższej jedynki, jeśli takie zera istnieją, neguje się wraz z tą najbliższą jedynką. Takie postępowanie powtarza się kolejno dla wszystkich jedynek dodatnich i ujemnych, aż do otrzymania wszystkich bitów.

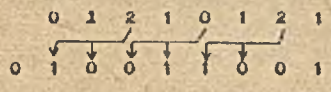
W przypadku wymaganego zapisu binarnego w postaci znak-moduł najbardziej znaczącej jedynce dodatniej odpowiada jedynka na tej samej pozycji binarnej i ciąg zer na wszystkich bardziej znaczących pozycjach, a najbardziej znaczącej jedynce ujemnej odpowiada jedynka na tej samej pozycji binarnej, jedynka na pozycji znakowej i zera na pozycjach binarnych między tymi jedynkami, jeśli nie są to jedynki na pozycjach sąsiednich. Z kolei bierze się pod uwagę następne kolejne jedynki z ciągu. W przypadku gdy najbardziej znacząca, już analizowana jedynka była dodatnia, wyznaczanie dalszych bitów jest identyczne jak w zapisie uzupełnieniowym. Natomiast, gdy najbardziej znacząca, już analizowana jedynka była ujemna, postępuje się dokładnie odwrotnie, to znaczy tak, jak w zapisie uzupełnieniowym przy zmienionych znakach wszystkich dalszych jedynek.

Poniżej podano przykład liczbowy /por. rys. 2/ przekształcenia zapisu ilorazu o cyfrach -1,0,+1, najpierw na zapis binarny uzupełnieniowy, a następnie na zapis binarny w postaci znak-moduł. Strzałkami pokazano, które jedynki ilorazu w zapisie redundancyjnym mają wpływ na poszczególne bity ilorazu w wymaganej postaci binarnej.



Przekształcenie zapisu ilorazu o cyfrach 0,1,2 odbywa się następująco. Jeśli kolejną cyfrą ilorazu jest zero, oznacza to, że otrzymane wcześniej bity na wszystkich bardziej znaczących pozycjach binarnych od tej cyfry są już ostatecznie ustalone. Jeśli kolejną otrzymaną cyfrą ilorazu jest dwójka, pozostawia się chwilowo zero na jej pozycji binarnej oraz neguje się na bardziej znaczących pozycjach binarnych ciąg kolejnych jedynek aż do najbliższego zera włącznie z tym zerem, traktując zanegowane bity jako ostatecznie ustalone. Jeśli kolejną otrzymaną cyfrą ilorazu jest jedynka, pozostawia się ją chwilowo jako kolejny bit ilorazu, aż do otrzymania na pozycjach mniej znaczących cyfry 0 lub 2. Wyznaczanie kolejnych cyfr w opisany wyżej sposób rozpoczyna się od chwilowego przyjęcia bitu zerowego na najbardziej znaczącej pozycji binarnej.

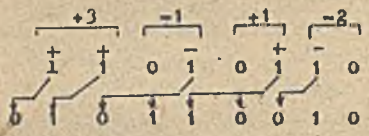
Poniżej podano przykład liczbowy /por. rys. 3/ przekształcenia zapisu ilorazu o cyfrach 0,1,2 na zapis binarny nieredundancyjny



Przekształcenie zapisu ilorazu o cyfrach od $-3,-2,-1,0,+1,+2,+3$ odbywa się następująco. Ciągowi cyfr z zakresu od -3 do $+3$ reprezentującemu iloraz przyporządkowuje się najpierw ciąg cyfr z zakresu od -1 do $+1$ w ten sposób, że kolejnym cyfrom $\bar{3},\bar{2},\bar{1},0,1,2,3$ odpowiadają pary $\underline{\bar{1}}\,\underline{\bar{1}}$, $\underline{\bar{1}}\,0$, $0\,\bar{1}$, $0\,0$, $0\,1$, $1\,0$, $1\,1$. Otrzymany tak ciąg cyfr $-1,0,+1$ przekształca się w sposób już wcześniej opisany.

Poniżej podano przykład liczbowy /por. rys. 4/ przekształcenia ilorazu o cyfrach

-3, -2, -1, 0, $+1$, $+2$, $+3$.



10. ZAKOŃCZENIE

Omówione w niniejszej pracy algorytmy dzielenia prowadzą do otrzymania ilorazu o podstawie rozwinięcia 2 i cyfrach -1,0,+1 lub 0,1,2 oraz o podstawie 4 i cyfrach -3,-2,-1,0,+1,+2,+3.

Przyjęcie podstawy 4 zmniejsza w stosunku do podstawy 2 około dwukrotnie liczbę kroków dzielenia koniecznych do wyznaczenia ilorazu o założonej dokładności, ale wymaga znacznie bardziej złożonej struktury układu dzielenia. Przyjęcie wyższej podstawy rozwinięcia ilorazu, na przykład $2^3 = 8$ lub większej, zmniejszyłoby w stosunku do podstawy 2 trzykrotnie lub więcej liczbę kroków dzielenia, ale doprowadziłoby do wielokrotnego zwiększenia liczby elementów składowych układu dzielenia.

Przyjęcie założenia, że jednemu krokowi dzielenia odpowiada wyznaczenie jednej cyfry ilorazu powoduje konieczność dysponowania różnymi wielokrotnościami dzielnika w liczbie odpowiadającej ilości cyfr ilorazu. Zmianie znaku cyfry odpowiada przy tym zmiana znaku odpowiedniej wielokrotności dzielnika. Jest ona łatwa do otrzymania przez zanegowanie bitów tej wielokrotności z uwzględnieniem korekcyjnej jedynki. Przyjęcie symetrycznego względem zera zakresu cyfr ilorazu umożliwia zatem, w ogólnym przypadku, konieczność pamiętania zmniejszonej liczby wielokrotności dzielnika i dysponowania do tego celu odpowiednio zmniejszą liczbą rejestrów. Nie wymagają przy tym oczywiście pamiętania w oddzielnych rejestrach wielokrotności dzielnika odpowiadające całkowitym potęgom dwójki ze względu na możliwość ich otrzymania przez przesunięcie i ewentualne zanegowanie bitów dzielnika.

Minimalnymi redundancyjnymi liczbami różnych cyfr dla podstaw rozwinięcia 2 i 4 są odpowiednio liczby 3 i 5.

Omówione w pracy algorytmy dzielenia przy podstawie 2 i trzech różnych cyfrach, mianowicie -1,0,+1, oraz 0,1,2 są na tyle proste, że jest właściwie sprawą oczywistą niecelowość rozważania większej liczby cyfr.

Zupełnie odwrotna sytuacja jest przy ilorazie o podstawie rozwinięcia 4. Analizowane przez autora, analogiczne do omówionych w pracy, algorytmy dzielenia o pięciu różnych cyfrach ilorazu wymagały do wyznaczania cyfr ilorazu badania tak dużych grup bitów kolejnych reszt, że realizacja odpowiednich układów dekodujących byłaby obecnie technicznie możliwa tylko przy wydłużeniu czasu poszczególnych kroków dzielenia, co oczywiście mija się z celem. Również stosowanie sześciu różnych cyfr, zatem cyfr z zakresu niesymetrycznego względem zera, nie ma żadnych zalet w porównaniu do siedmiu cyfr symetrycznego względem zera zakresu od -3 do +3. Z tych powodów oraz dla uniknięcia dalszego zwiększenia i tak dużej objętości niniejszej pracy, omówiono w niej jedynie algorytm dzielenia z ilorazem o podstawie rozwinięcia 4 i najkorzystniejszym przy podstawie 4 zakresem cyfr od -3 do +3.

Wyznaczenie trzykrotnej wartości dzielnika, odpowiadającej cyfrze 3 wymaga jednak wykonania dodatkowego dodawania /patrz rozdz.8/ wydłużającego czas trwania całego dzielenia. Może nasunąć się pytanie jakie konsekwencje pociągnęłaby za sobą próba "ominięcia" tego dodawania, czasochłonnego ze względu na propagację przeniesień. Zamiast je wykonywać można oczywiście w przypadkach cyfr -3 i +3 wprowadzić dwa kroki redukcji równoległej w miejsce jednego. Konsekwencją tego byłyby takie wady jak znaczne wydłużenie czasu dzielenia, zmienny czas dzielenia i większa niejednorodność poszczególnych kroków. Innym rozwiązaniem byłoby operowanie, w przypadku cyfr od -3 do +3, odpowiednią wielokrotnością dzielnika w postaci dwuskładnikowej, co prowadziłoby w odpowiednich krokach dzielenia do redukcji równoległej większej niż 3 liczby składników. Skomplikowałoby to również bardzo znacznie algorytm dzielenia. Można to uzasadnić następująco: załóżmy, że redukcja równoległa wielu składników do mniejszej liczby składników z zachowaniem ich sumy odbywa się ze względu na wymaganą dużą szybkość redukcji w jednej warstwie niezależnie działających układów, które nazwiemy koderami. Najprostszą strukturę całej warstwy koderów uzyskuje się przy koderach, których wejścia obejmują najmniejszą liczbę pozycji binarnych redukowanych składników, a więc tylko jedną pozycję binarną. Łatwo pokazać,

że maksymalnymi liczbami bitów, które mogą być redukowane przez takie kodery do liczby bitów 1,2,3,...,n są odpowiednio liczby bitów 1,3,7,...,2^n-1. Wynika stąd wniosek, że najprostszym przypadkiem równoległej "jednowarstwowej" redukcji jest redukcja z 3 do 2 składników, a następnie z 4 do 3, z 5 do 3, z 6 do 3, z 7 do 3, z 8 do 4 itd. składników. Przyjęcie "jednowarstwowej" redukcji i dwuskładnikowego dzielnika prowadzi zatem, jak łatwo zauważyć, do trójskładnikowych kolejnych reszt i redukcji z 5 do 3 składników. Stopień złożoności odpowiedniego pojedynczego kodera redukującego z 5 do 3 bitów w porównaniu do jednopozycyjnego sumatora redukującego z 3 do 2 bitów można przy tym z grubsza oszacować co najmniej jako $2^5:2^3=4$.

Na zakończenie warto poruszyć następujące zagadnienie, nie omawiane w niniejszej pracy. W każdym z omówionych algorytmów dzielenia, po redukcji z 3 do 2 składników, następuje analizowanie grupy bitów wziętych z dwóch składników kolejnej reszty.
Załóżmy na razie, że w skład tej analizy wchodzi omówione w pracy dodawanie modulo 2^3, 2^4 lub 2^5 dwóch liczb trzy, cztero, lub pięciobitowych, zależnie od zastosowanego algorytmu. Otrzymaną w wyniku trzy, cztero lub pięciobitową liczbę można traktować jako jednoskładnikową bardziej znaczącą część dwuskładnikowej reszty. Przy kolejnej redukcji trzech liczb bardziej znacząca ich część, wprawdzie bardzo krótka, jest wtedy jednak już dwuskładnikowa. Powstaje pytanie, czy i jak można przekształcić bardziej znaczącą część tych trzech redukowanych liczb/zamiast omówionej redukcji równoległej/, by w wyniku tego przekształcenia otrzymać od razu potrzebną do wyznaczenia cyfry ilorazu bardziej znaczącą część kolejnej reszty w postaci jednej liczby trzy, cztero lub pięciobitowej. Inaczej mówiąc powstaje pytanie, jak rozwiązać bardzo szybkie dodawanie trzech liczb o niewielkiej liczbie bitów, z których jedna liczba ma ponadto kilka zer /znaną liczbę zer/ na najbardziej znaczących pozycjach binarnych. Właściwa odpowiedź na to pytanie zależy w dużym stopniu od rodzaju bazy elementowej, a odpowiedź na nie powinien chyba dać w konkretnym przypadku konstruktor odpowiedniego układu dzielenia.

LITERATURA

[1] FLORES I.: Arytmetyka maszyn cyfrowych, WNT, Warszawa 1970

[2] ROBERTSON J.E.: A new class of digital division methods, IRE Trans. on Electronic Computers, Sept. 1958, t. EC-7, s. 218-222

[3] TOCHER T.D.: Techniques of multiplication and division for automatique binary computers, Quarterly Journal of Mechanics and Applied Mathematics 1958 t. 2 cz. 3, s. 364-384

[4] ATKINS D.E.: Higher-radix division using estimates of the divisor and partial remainders, IEEE Trans. on Computers, Oct. 1968 t.C-17, s.925-934

[5] MAJERSKI S.: Układ cyfrowy cyklicznego sumowmożenia, Pat. nr 104832, zgł. 10.05.76

[6] MAJERSKI S.: Układ cyfrowy do obliczania iloczynu skalarnego wektorów o składowych binarnych, Pat. nr 107341 zgł. 01.07.76

[7] MAJERSKI S., MAJERSKI W.: Urządzenie cyfrowe do obliczania wartości złożonych wyrażeń arytmetycznych, Pat. nr 106470, zgł. 01.02.77

[8] MAJERSKI S.: Układ cyfrowy mnożenia binarnego liczby przez sumę dwóch liczb, Pat.nr 109901, zgł. 21.12.76

[9] MAJERSKI S., MAJERSKI W.: Układ cyfrowy do obliczania pierwiastka kwadratowego w zapisie binarnym, Pat. nr 109471, zgł. 29.7.77

[10] MAJERSKI S., MAJERSKI W.: Układ cyfrowy dzielenia binarnego, Pat. zgł. 25.06.79 /P-216598/

[11] MAJERSKI S., KUBERA W.: Cyfrowy układ binarny do obliczania wartości funkcji wykładniczej, Pat. zgł. 31.12.79 /P-221060/

[12] MAJERSKI S., KUBERA W.: Cyfrowy układ binarny do logarytmowania, Pat. zgł. 02.01.80 /P-221168/

[13] WITKOWSKI A.: SRT division with a normalized divisor, Prace naukowo-badawcze Instytutu Maszyn Matematycznych, Warszawa 1980, s. 1-16

ANDRZEJ WITKOWSKI

DZIELENIE SRT ZE ZNORMALIZOWANYM DZIELNIKIEM

Streszczenie

W artykule przedstawiono metodę wyboru cyfry ilorazu dla dzielenia SRT. W prezentowanej metodzie dzielnik D jest sprowadzony do zakresu: $1-1/r \leq |D| < 1$. W artykule pokazano, że precyzja badania dzielnika i reszty częściowej potrzebna do określenia cyfry ilorazu zależy od zakresu zmienności dzielnika. Następnie znajduje się optymalny zakres zmienności dzielnika/zakres, dla którego precyzja badania dzielnikami reszty częściowej jest najmniejszy/. Zakres ten określony jest nierównością: $1-1/r \leq |D| < 1$. Dla takiego zakresu zmienności dzielnika można zaprojektować urządzenie wyznaczające cyfrę ilorazu dla dowolnej podstawy dzielenia r. Tak zaprojektowane urządzenie omówiono w niniejszej pracy.

W artykule przedstawiono dalej sposób sprowadzenia dzielnika do optymalnego zakresu: $1-1/r \leq |D| < 1$. Sposób ten polega na mnożeniu dzielnika przez czynniki a_i. Czynniki wyznacza się na podstawie kilku najstarszych bitów dzielnika. W pracy wyznaczono precyzję badania dzielnika potrzebną do obliczenia czynników a_i.

Prezentowana metoda pozwala realizować bardzo szybkie układy dzielenia.

ДЕЛЕНИЕ SRT СО СТАНДАРТНЫМ ДЕЛИТЕЛЕМ

Резюме

В статье представлен метод выбора цифры частного для деления SRT . В представленном методе делитель D сводится к виду: $1-1/r \leq |D| < 1$. В статье показывается, что точность исследования делителя и частичного остатка, необходимая для определения цифры частного, зависит от вида делителя. Затем находится оптимальный вид делителя /вид для которого точность исследования деления и частичного остатка является минимальной/. Этот вид определяется неравенством: $1-1/r \leq |D| < 1$. Для такого вида делителя можно спроектировать устройство, определяющее цифру частного для любого основания деления. Так спроектированное устройство рассматривается в настоящей работе.

В статье дальше представлен способ сведения делителя к оптимальному виду: $1-1/r \leq |D| < 1$. Этот способ заключается в умножении делителя на сомножитель a_i. Сомножители определяются на основе нескольких самых старших битов делителя. В работе определяется точность исследования для вычисления сомножителей a_i.

Предоставленный метод позволяет реализовать очень быстрые системы деления.

STANISŁAW MAJERSKI

THE BINARY NUMBER DIVISION WITH THE PARTIAL REMAINDERS AND THE QUOTIENT IN REDUNDANT NOTATIONS

Summary

The algorithms of the binary division designated for high-speed parallel digital circuits are presented. The algorithms eliminate the subtractions and additions with carry propagation, which appear in the classical method of the nonrestoring division. These operations are replaced by the faster three-summand carry-save additions, for which the name of the parallel reduction of three summands to two summands, is introduced, for easier distinction from the common additions. As a result of the parallel reduction, in the successive division steps, the two-summand partial remainders are obtained. Then, the successive digits of a quotient in a redundant notation are determined on the basis of some bits taken from these remainders. This quotient notation may be currently transformed into the common binary notation. For the three described algorithms the redundant quotient notations with the digits -1,0,+1, the digits 0,1,2, and the digits from -3 to +3 have been assumed. In the third of these algorithms, two quotient bits correspond to one division step and to one quotient digit.

ДЕЛЕНИЕ ДВОИЧНЫХ ЧИСЕЛ С ИЗБЫТОЧ... ЗАПИСЬЮ ОЧЕРЕДНЫХ ОСТАТКОВ И ЧАСТНОГО

Резюме

Предметом настоящей работы являются алгорифмы двоичного деления предназначенные для быстродействующих параллельных цифровых схем. В этих алгорифмах исключены операции вычитания и сложения с распространением переноса, которые характерны для классического метода деления без восстановления остатка. Эта операция заменена менее продолжительной операцией сложения трёх слагаемых с запоминанием переноса, которое с целю отличить от обыкновенного сложения названа параллельным сокращением числа слагаемых с трёх до двух. На очередных шагах деления получаются в результате параллельного сокращения очередные частичные остатки в виде двух слагаемых. На основе нескольких разрядов таких остатков определяется очередные цифры частного в избыточной записи. Такая запись по ходу процесса деления может быть преобразована в обыкновенную двоичную запись. В трёх рассматриваемых алгорифмах принято соответственно избыточные записи частного с цифрами -1,0,+1, с цифрами 0,1,2 и цифрами от -3 до +3. В третим алгорифме одному шагу деления и одной цифре частного соответствуют два двоичных разряда.