Higher-Radix Division Using Estimates of the Divisor and Partial Remainders

DANIEL E. ATKINS, STUDENT MEMBER, IEEE

***Abstract*—The nature of a class of division techniques which permit the selection of quotient digits in digital division by the inspection of truncated versions of the divisor and partial remainder is reviewed in detail. Two types of mechanisms, or so-called model divisions, for the selection of quotient digits are introduced. For both types of techniques, analytic tools are suggested for determining the number of bits which must be inspected as a function of the radix and form of representation of quotient digits. The analysis accounts for the representation of the partial remainder in a redundant form such as the one produced by an adder–subtractor which eliminates carry–borrow propagation.**

***Index Terms*—Arithmetic unit, binary division, computer arithmetic, division.**

Introduction

THIS PAPER reviews the nature of a class of division techniques especially suited for implementation in an electronic digital computer. The techniques permit the selection of quotient digits by the inspection of truncated versions of the divisor and partial remainder. The radix-two member of this class, the so-called SRT division, has been known for some time. An example of this radix-two case was described by Nadler [1] as early as 1965. Whether or not the Nadler division is equivalent to the SRT is obscured by the fact that it is discussed in conjunction with a stored carry adder–accumulator. The SRT division was given that name by Freiman [2] because it was discovered independently at about the same time by D. Sweeney of IBM, J. E. Robertson of the University of Illinois [3], and T. D. Tocher, then of Imperial College, London [4]. The extension of the technique to higher radices as reported here is based primarily upon work by J. E. Robertson [5]. References [8]–[11] are provided for those wishing to read further concerning division techniques.

The purpose of this paper is to first review and illuminate the theory of this division technique, and then to develop analytic expressions for determining the number of bits of divisor and partial remainder which must be inspected for a given radix and a given form of representation of quotient digits.

It will also be demonstrated that the required precision is related to the type of selection mechanisms used to generate quotient digits.

Manuscript received March 4, 1968; revised July 5, 1968. This work was supported in part by the Atomic Energy Commission under Contract USAEC AT(11-1)-1018.

The author is with the Department of Computer Science, University of Illinois, Urbana, Ill. 61801.

The Recursive Relationship

Digital division as implemented in an electronic computer consists of preliminary operations, e.g., normalization, a recursive process; and a terminal operation, e.g., changing the form of the remainder. Although preliminary and terminal operations vary from machine to machine, they generally consume much less of the execution time than the recursive operations. For restoring, nonrestoring, and the schemes to be described in this paper, this recursive relationship is defined by

$$p_{j+1} = rp_j - q_{j+1}d, \tag{1}$$

where the symbols are defined as follows:

j = the recursive index = 0, 1, $\cdots$, m − 1
p_j = the partial remainder used in the jth cycle
p_0 = the dividend
p_m = the remainder
q_j = the jth quotient digit in which the quotient is of the form

$$q_0 \vartriangle q_1 q_2 \cdots q_m$$

$\uparrow$ radix point

m = the number of digits, radix r, in the quotient
d = the divisor
r = the radix.

Although not germane to the *theory* of division, it is interesting to note in passing that this relation points to possibilities for accelerating the execution of division. Verbally, the equation says that each partial remainder must be multiplied by the radix rp_j, i.e., shifted left one digital position, and that the selected quotient digit must then be multiplied by the divisor $q_{j+1}d$, and subtracted from this shifted partial remainder. The division process will thus be accelerated if the shift and/or the subtraction time is decreased. In practice, all values of $q_{j+1}d$ are stored in registers or are readily available via shift gates from the register containing the divisor. The rapid formation of $q_{j+1}d$ thus reduces to minimizing the necessity for forming awkward multiples requiring an addition, and to accelerating the selection of $q_{j+1}d$ at the divisor input to the adder–subtractor.

Secondly, note that the recursive index j is implicitly an inverse function of the radix. When actually implemented on a machine, digits of a radix higher than two are represented by two or more binary bits. A string of l binary digits (bits) is equivalent to $l/2$ radix-four digits.

In general, for l bits of radix two, there corresponds $m=l/\log_2 r$ digits of radix r, where for practical cases, $r=2^n$ and $n=$integer>0. Thus to produce a quotient of given precision, the number of iterations required and concomitantly, the execution time, are decreased as the radix is increased.

Representation of Quotient Digits

As noted above, the use of a higher radix reduces the number of cycles required to perform a division of given precision. The implementation of such a scheme may be costly, however, and costlier still if quotient digits are represented as they are in manual methods or machine restoring division. In these cases, quotient digits have the values 0, 1, 2, $\cdots$, $r-1$. A radix-four restoring division therefore requires that multiples of 1, 2, and 3 times the divisor be available for subtraction from the partial remainder. The 1 time multiple is readily available, of course; the 2 times multiple is formed by merely shifting left one binary position; the 3 times multiple, however, requires extra time and/or hardware. It may be formed by a tripler circuit or by addition of 1 time and 2 times the divisor which is then stored in an auxiliary register.

With higher-radix SRT division the problem of forming divisor multiples is mitigated by using both plus and minus quotient digit values. The quotient digits are of the form $-n$, $-(n-1)$, $\cdots$, -1, 0, 1, $\cdots$, n, where n is an integer such that $1/2(r-1)\leq n\leq r-1$. Within this range the actual choice of n for a given r is largely a function of design details.

The necessity for the range restriction is as follows: at least r unique digits are required to represent a number, radix r. In the representation introduced above, there are $2n+1$ unique digits, and thus the requirement $2n+1\geq r$. On the other hand, for radix r, the maximum value of a quotient digit n should not be greater than the value of the maximum digit representable, thus $n\leq r-1$. Combining these two inequalities yields the restriction stated above.

With plus and minus quotient digits, a higher-radix division may be implemented with fewer awkward multiples of the divisor. Now the quotient digits for a radix-four division are -2, -1, 0, $+1$, $+2$. All the necessary multiples of the divisor may be formed by shifting and complementation and require no auxiliary registers.

The second, but probably more significant, consequence of this representation of quotient digits is that it introduces redundancy into the representation of the quotient. If $2n$ is greater than $r-1$, then there are more symbols available to represent a number than is actually necessary. Therefore, some numerical values may be represented in more than one form. For example, with $r=4$, $n=2$, and with $^-$ representing negation, the number 6 could be represented as 12, or $2\bar{2}$. As explained in later sections, this redundancy permits less precision in comparing the divisor and partial remainder in selecting a quotient digit. This statement seems intuitively correct, since without redundancy each quotient digit may be represented in only one way and thus must be selected precisely. With redundancy, the quotient digit, and thus the comparison of divisor and partial remainder, need not be precise. However, this nonunique representation does complicate the division in that the redundant form must eventually be converted to a conventional representation.

Range Restrictions

With the quotient representation now defined, consider the derivation of range restrictions on the partial remainders. Recall from the manual execution of a division that in determining whether a quotient digit is correct or not, one is essentially applying the restriction that $0\leq p_{j+1}<d$, where p_{j+1} is the result of the subtraction of q_{j+1} times the divisor from the jth partial remainder. If p_{j+1} is not within this range then q_{j+1} is changed until it is. For nonrestoring division, negative partial remainders and negative quotient digits are allowable, and thus the range restriction is $|p_{j+1}|\leq|d|$. It seems reasonable, therefore, to hypothesize other division techniques for which $|p_{j+1}|\leq k|d|$, and which utilize the quotient digit representation introduced in the last section. The upper limit on k is 1.

We now adopt the hypothesis that even though we may be working with a radix greater than two, the divisor is in a binary normalized form, i.e., restricted to the range $1/2\leq d<1$. In this case, as we shall find, the lower limit of k is 1/2.

First reconsider the recursive relationship (1). After p_{j+1} is formed on the jth cycle, it is multiplied by the radix r (shifted left); j is increased by one and becomes rp_j of the present cycle. Since $|p_{j+1}|\leq kd$, it follows that p_j must obey the same restrictions, i.e.,

$$r|p_j| \leq rk|d|. \tag{2}$$

Substituting (1) into (2) yields

$$-kd \leq rp_j - q_{j+1} \leq kd. \tag{3}$$

Equation (1) is now normalized with respect to the divisor and is rewritten, letting $z_j=p_j/d$ and $z_{j+1}=p_{j+1}/d$:

$$z_{j+1} = rz_j - q_{j+1}. \tag{4}$$

Equation (4) may be interpreted graphically as a plot of z_{j+1} versus rz_j with the quotient digit q_{j+1} as a parameter. Such a representation shall be called a "z–z plot." Recall that the quotient digits assume values $-n$, $-(n-1)$, $\cdots$, -1, 0, $+1$, $\cdots$, n. Fig. 1 is such a graph. To facilitate discussion, each plot corresponding to a different quotient digit is called a "q line."

The goal of this section is to demonstrate that a correct division procedure exists which incorporates the above range restrictions and quotient representation.

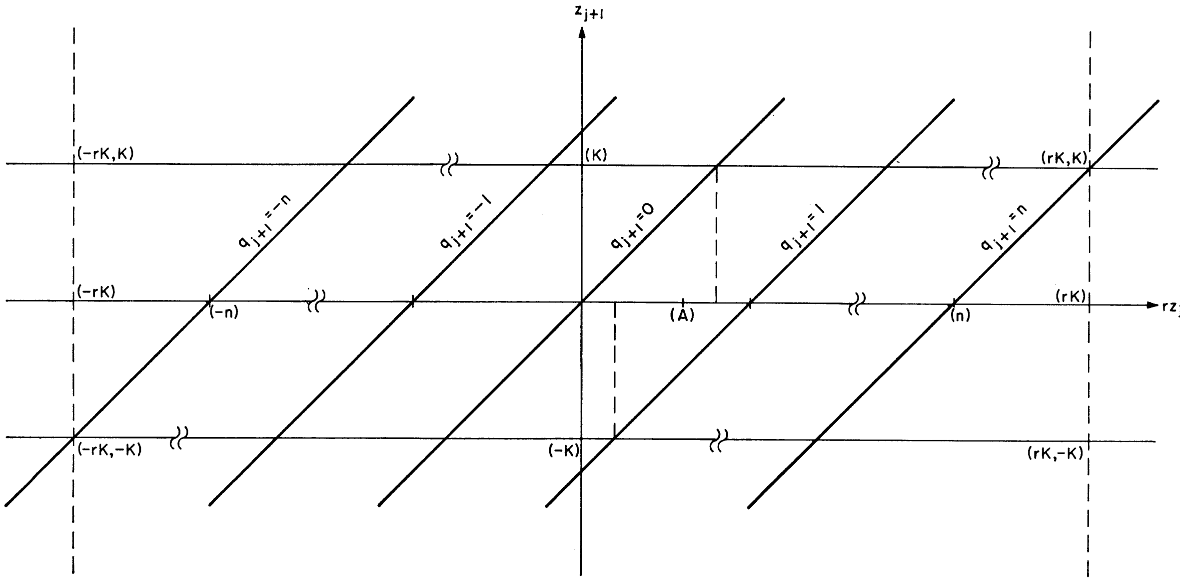

Fig. 1. z–z plot of division procedure.

This existence is substantiated if for each value of rz_j in the allowed range there corresponds a quotient digit and a z_{j+1}, also in their allowed ranges. In terms of Fig. 1, this means that for any point on the rz_j axis such that $-rk \leq rz_j \leq rk$, one must be able to move on a line segment normal to the rz_j axis and intersect a q line at a point corresponding to a z_{j+1} within the range $-k \leq z_{j+1} \leq k$. This allowed range is enclosed between the lines $z_{j+1} = k$ and $z_{j+1} = -k$ in Fig. 1.

To satisfy the foregoing requirements, the maximum value of rz_j, i.e., rk, must occur at the intersection of $z_{j+1} = k$ and the q line, $z_{j+1} = rz_j - n$. Similarly, the minimum value must occur at the intersection of $z_{j+1} = -k$ and the q line, $z_{j+1} = -rz_j + n$. These bounds on rz_j are indicated by the dashed vertical lines of Fig. 1.

Fig. 1 now points to the value of k in terms of r and n. At the upper-right vertex of the bounding rectangle, $z_{j+1} = k = rz_j - n$. But since $rz_j = rk$,

$$k = \frac{n}{r - 1} \cdot \tag{5}$$

The division is now characterized by tangible parameters, namely the radix and the maximum value of quotient digits. Combining (5) with the restriction on n, $(r-1)/2 \leq n \leq r-1$, verifies the statement at the beginning of this section, $1/2 \leq k \leq 1$.

Redundancy in the Quotient Representation

We have established that the quotient digit representation introduces redundancy into the quotient. This fact is also manifested in Fig. 1 in the regions on the rz_j axis for which either one of two q lines may be legitimately selected. For example, at point A one may move vertically upward to the $q_{j+1} = 0$ line or downward to the $q_{j+1} = +1$ line. In either case the quotient digit is correct. Fig. 2, a specific case of Fig. 1, testifies to the fact that this freedom of choice is not merely the result of an inaccurately drawn graph. Here $r = 4$, $n = 2$. The vertical dashed lines define the overlap regions.

The production of a redundant quotient requires extra hardware and perhaps time to convert it to a conventional binary representation acceptable by other sections of the machine and by programmers. This conversion is discussed at greater length in subsequent sections of this paper. With no redundancy, the divisor and the shifted partial remainder must be compared (usually by subtraction) to the full precision defined for the machine. With redundancy, the designer is at liberty to inspect fewer bits of the divisor and shifted partial remainder than define full precision. Handling fewer bits may save time and hardware. In Fig. 2, for example, a correct quotient digit is selected knowing $rz_j = rp_j/d$ to a precision only great enough to contain it within an overlap region. Exactly what precision is required for a given value of r and n is the subject of the next section.

In terms of z–z plots such as Figs. 1 and 2, the redundancy is proportional to the width of the overlap regions. The width of this region in terms of n and r is found as follows. Consider two adjacent lines of Fig. 1, i.e., $z_{j+1} = rz_j - i$ and $z_{j+1}' = rz_j - (i-1)$. The overlap, Δrz_j, is the difference between rz_j for $z_{j+1} = n/(r-1)$ and rz_j for $z_{j+1}' = -n/(r-1)$. Solving for this difference yields $\Delta rz_j = 2n/(r-1) + 1$. The ratio $n/(r-1)$, is, therefore, a measure of redundancy.

As redundancy (width of overlap region) is *increased*, the required precision of inspection of divisor and partial remainder, and thus hopefully the execution time, is *decreased*. Therefore, it appears that for a given r, n should be as large as possible, i.e., n should equal $r-1$. Such a choice may not be practical, however, since $n = h$ requires the ability to form h multiples of the divisor. The choice of n is therefore bound up in the usual tradeoff between time and hardware.

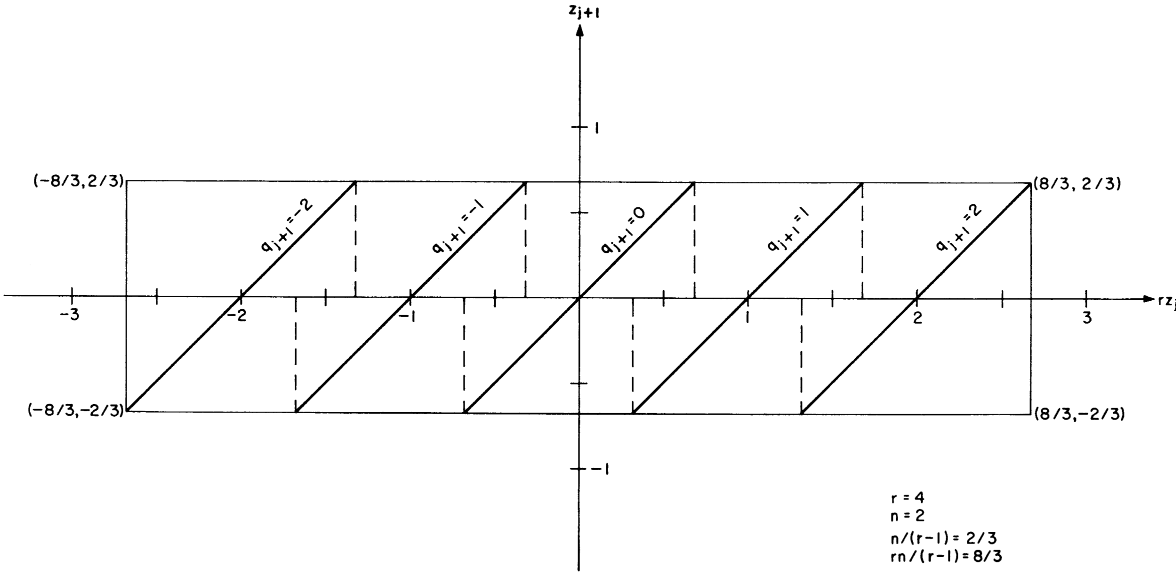

Fig. 2. z–z plot with $r=4$, $n=2$.

The P–D Plot

Now consider another graphical representation of the division procedure. This construction, suggested by C. V. Freiman of IBM [2], is useful in further describing higher-radix SRT division and in computing the required precision of inspection of the divisor and shifted partial remainder. The basis for the plot is recursive relationship (1), together with the range restriction

$$|p_j + 1| \leq \frac{n}{r-1} d.$$

The figure is thus essentially a plot of partial remainder versus divisor values and therefore in this paper shall be referred to as the "P–D plot."

Solving the recursive relationship for rp_j yields

$$rp_j = p_{j+1} + q_{j+1}d. \tag{6}$$

For a fixed quotient digit, the upper limit of rp_j as a function of the divisor d occurs when p_{j+1} is maximum, i.e., when

$$p_{j+1} = \frac{n}{r-1} d,$$

and thus

$$rp_{j\,\max} = \left(\frac{n}{r-1} + q_{j+1}\right)d. \tag{7}$$

Likewise, the lower limit occurs with

$$p_{j+1} = \frac{-n}{r-1} d,$$

and thus

$$rp_{j\,\min} = \left(\frac{-n}{r-1} + q_{j+1}\right)d. \tag{8}$$

These linear equations may be plotted as functions of d with q_{j+1} as a parameter ranging from $-n$ to $+n$ in steps of 1. The area between $rp_{j\,\max}$ and $rp_{j\,\min}$ for a given $q_{j+1}=i$ will be denoted as the "$q(i)$ area."

The division procedure is now determined. A given value of divisor d and the jth shifted partial remainder will specify a point in a $q(i)$ area. The digit i will be the value of the next quotient digit q_{j+1} which in turn is used in forming the next partial remainder. In this representation, the redundancy is manifested as overlapping of the $q(i)$ regions, i.e., some pairs of d and rp_j will specify a point for which either $q_{j+1}=i$ or $q_{j+1}=i-1$ is a valid choice.

Fig. 3 is an example of a P–D plot for a division with $r=4$, $n=2$. The equations for the lines plotted, 2′, 2, etc., are given in Table I. The region for which $q_{j+1}=2$ is a valid choice, the $q(2)$ area, is between lines 2′ and 2; the $q(1)$ area is between lines 1′ and 1, and so forth. Note the overlap between $q(i)$ areas; for example, the region between lines 1′ and 2 in which either the choice $q_{j+1}=1$ or $q_{j+1}=2$ is correct. Note further that the figure is symmetric about both axes.

On the right half of Fig. 3 (the same may be done on the left), "steps" have been drawn within the overlap of the $q(i)$ regions. The width of a "tread" (constant rp_j, d varying) defines a "divisor interval," the value of rp_j for each tread defines a comparison constant, and the distance between comparison constants defines a "partial remainder interval." Phrased in this terminology, division consists of locating a given divisor value within the appropriate divisor interval, locating the shifted partial remainder within the appropriate interval (using comparison constants), and selecting a value of q_{j+1} enclosed by the intersection of the boundaries of these intervals. Since a divisor and partial remainder must be located only to within an "interval," they need not be inspected

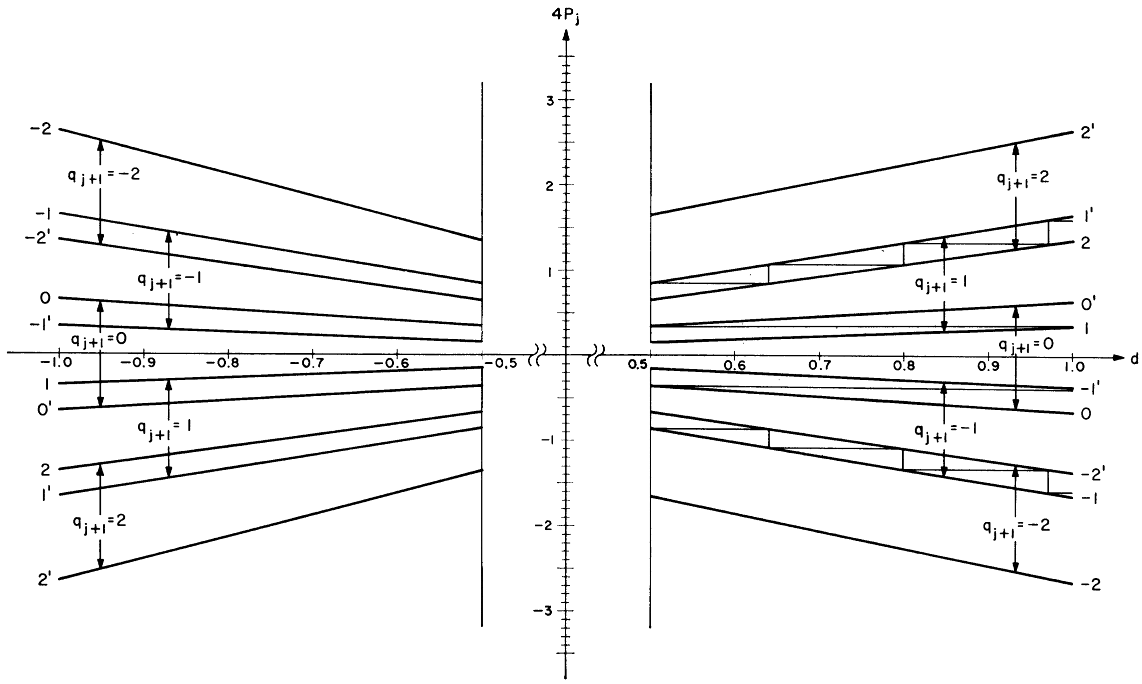

Fig. 3. P–D plot with $r=4$, $n=2$.

TABLE I

EQUATIONS DEFINING THE REGIONS OF FIG. 3

$$rp_j = \pm \frac{n}{r-1} d + q_{j+1} d$$

$$r = 4$$

Designation in Fig. 3	q_{j+1}	p_{j+1}	rp_j
2′	2	2/3 d	8/3 d
2	2	−2/3 d	4/3 d
1′	1	2/3 d	5/3 d
1	1	−2/3 d	1/3 d
0′	0	2/3 d	2/3 d
0	0	−2/3 d	−2/3 d
$\bar{1}'$	$\bar{1}$	2/3 d	−1/3 d
$\bar{1}$	1	−2/3 d	−5/3 d
$\bar{2}'$	$\bar{2}$	2/3 d	−4/3 d
$\bar{2}$	$\bar{2}$	−2/3 d	−8/3 d

to full precision in selecting a correct quotient digit. Here is where the redundancy pays dividends.

Techniques for selecting divisor intervals and comparison constants are detailed in the next two sections. At this point, however, we shall make several general observations. First, note that the comparison constants are compared with the high-order N_p bits of the shifted partial remainder and, similarly, that the end points of the divisor intervals are compared with the N_d high-order bits of the divisor. The comparison constants and end point of the divisor intervals should therefore be numbers which are representable with N_p and N_d bits, respectively. The choices illustrated in Fig. 3 which maximized the width of the divisor intervals do not meet this requirement.

In Fig. 4, however, more practical choices are shown. The dashed lines represent the theoretical choices used

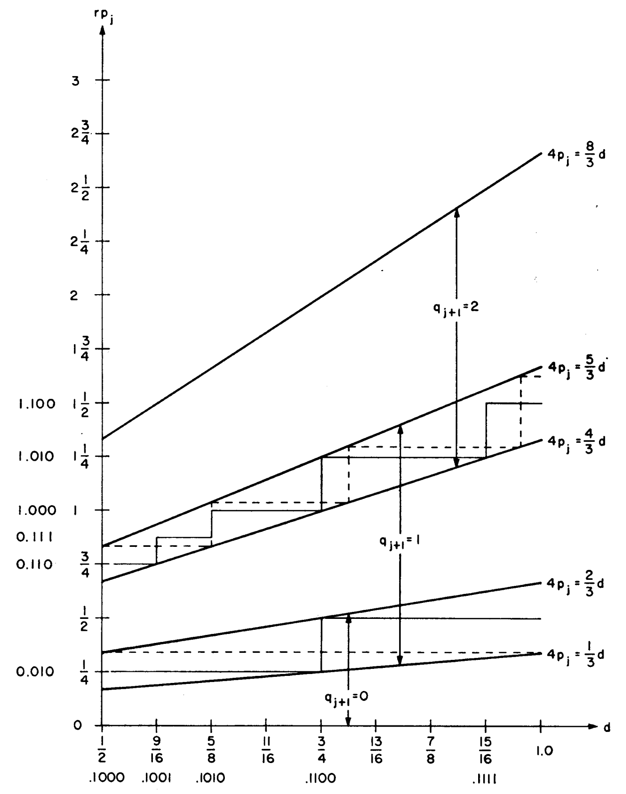

Fig. 4. Divisor intervals and comparison constants with $r=4$, $n=2$.

in Fig. 3. Now, although the number of steps has been increased, the boundaries fall at points easily representable in binary notation. Note that inspection of four bits plus sign of the partial remainder and divisor is sufficient to locate the correct choice of quotient digit.

The second observation is that the choice of divisor intervals and comparison constants is bound up with the

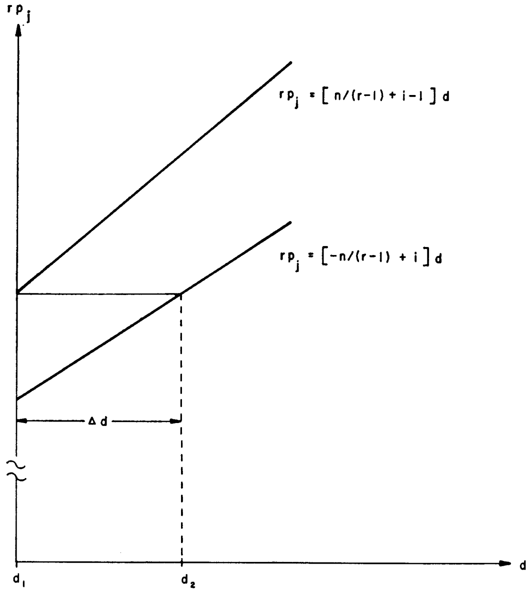

Fig. 5. Detail of a P–D plot overlap region.

required precision of inspection of the partial remainder and divisor; if, for example, the divisor intervals widths are increased, the required precision of divisor inspection (number of bits) may be decreased. Furthermore, the maximum precision of inspection of the divisor is determined by the divisor interval of smallest width. By inspection of Fig. 4, the reader might guess where this step is, but we shall now locate it analytically. The result of this derivation will be useful in the next sections.

The length of a divisor interval is limited by the boundaries of the overlap region. The maximum precision of inspection is required where the divisor interval is minimum. To determine where this minimum divisor interval occurs, consider the detail of the overlap of the $q(i)$ and $q(i-1)$ regions shown in Fig. 5.

For a given value of rp_j, the maximum width of a divisor interval is

$$\Delta d = d_2 - d_1 = \frac{rp_j}{\dfrac{-n}{r-1} + i} - \frac{rp_j}{\dfrac{n}{r-1} + i - 1} \tag{9}$$

$$= rp_j R \frac{2n - R}{R^2 i^2 - R^2 i + nR - n^2}$$

where $R = (r-1)$.

The interval Δd is minimum when i is maximum and rp_j is minimum. The maximum value of i is n; the minimum value of rp_j for $q_{j+1} = n$ will occur when the upper bound of the overlap region intersects $d = 1/2$, i.e., when $d_1 = 1/2$. The precision of required inspection of divisor is thus determined by the divisor interval closest to $d = 1/2$ and between $q_{j+1} = n$ and $q_{j+1} = n-1$.

The Cost of Quotient Digit Selection

To this point we have established that an important feature of the division is the ability to select quotient digits from truncated versions of the divisor and shifted partial remainder. We now turn to the more specific question of what precision is required in these approximations, i.e., how many bits of the divisor and shifted partial remainder must be inspected to guarantee correct quotient digit selection. In a sense, this required precision is the "cost of quotient digit selection."

The cost will be shown to be a function of the choice of radix, and to a certain extent, of the method of selecting the quotient digits. Robertson [5] has suggested that the mechanism for selection of quotient digits may be viewed as a "limited precision model" of the full precision division. This concept is exemplified in the following example.

A radix-256 division would require 8 quotient bits per shift of partial remainder. To generate these 8 bits, 12 bits of the partial remainder and 13 bits of the divisor are presented to a division mechanism which need be only elaborate enough to produce 8 bits of quotient from a 12-bit dividend and a 13-bit divisor. The results of this limited precision division (8 bits) are returned to the full precision mechanism as part of the full precision quotient, and are used in forming the next full precision partial remainder. Note that the number defining full precision may be changed in discrete steps by changing the number of "calls" to the model division. Furthermore, the model division scheme may be quite different from that of the full precision division.

For purposes of computing costs of quotient selection, we shall consider two classes of model division procedures. The first will be those involving the use of an auxiliary arithmetic unit and employing addition and/or subtraction in forming the quotient digits. Examples of schemes in this class include a radix-four division performed in the exponent arithmetic unit, or the procedure suggested by Wallace [6] which is logically equivalent to forming the approximate reciprocal of the divisor and multiplying by the partial remainder. This class will be referred to as the "arithmetic models."

The second class consists of those methods which are the logical equivalent of a table look-up. This technique may be viewed as the direct implementation of a P–D plot, i.e., decoding the divisor interval and the partial remainder interval, and producing the quotient digit indicated by their intersection. This class will be referred to as the "table look-up models."

Before considering these two models in further detail, let us state more precisely the conditions which must be obtained in the choice of model division and precision of inspection. Let

m = the number of bits to the right of the radix point of divisor and dividend

$r\hat{p}_j$ = the truncated version of the shifted partial remainder

ϵ = the number of bits to the right of the radix point in $r\hat{p}_j$

$\Delta p = \pm(2^{-\epsilon} - 2^{-m}) \approx \pm 2^{-\epsilon}$, the uncertainty in rp_j

$\hat{d}$ = the truncated version of the divisor

δ = the number of bits to the right of radix point in $\hat{d}$

$\Delta d = \pm(2^{-\delta} - 2^{-m}) \approx \pm 2^{-\delta}$, the uncertainty in $\hat{d}$.

The following "cost criterion" summarizes the requirements on the quotient selection mechanism, Δd and Δp.

Cost criterion: Given the approximations $r\hat{p}_j \pm \Delta p$ and $\hat{d} \pm \Delta d$, the integer result of $r\hat{p}_j/\hat{d} = i$ performed in the model must be such that on the appropriate P–D plot, the rectangle defined by $(\hat{d} \pm \Delta d,\ rp_j \pm \Delta p)$ is entirely within the $q(i)$ region.

Cost Determination for an Arithmetic Model

We first consider the determination of the cost for a division using an arithmetic model. In this case, $r\hat{p}_j$ and $\hat{d}$ are presented to a limited precision arithmetic unit and the division carried out to produce a rounded integer quotient. If the bit position to the right of the radix point in the model is 1, the integer portion is increased by one and truncated; otherwise the result is merely truncated. This rounding is necessary if the cost criterion is to hold for an arithmetic model.

Equation (9) indicated that maximum precision is required in the overlap of the $q(n)$ and $q(n-1)$ regions in the vicinity of $d = 1/2$. The precision determined here will be sufficient for any other region of the P–D plot. Fig. 6 is a detail of this region.

Two additional factors must now be considered: a redundantly represented partial remainder, and a negative divisor. A division scheme which meshes well with multiplication must cope with redundantly represented partial remainders. One consequence of the representation is that the truncation error (Δp) attributable to considering only a few higher-order bits of the partial remainder may be either positive or negative. When a negative (2's complement) divisor is permitted, truncation error may also be negative.

In the divisor interval $1/2 \pm \Delta d$, the dividing line between the selection of $q = n$ and $q = n-1$ is $r\hat{p}_j = 1/2(n-1/2)$, since $r\hat{p}_j/\hat{d} = 2 \times 1/2(n-1/2) = n-1/2$ which must be rounded to n. For the cost criterion to hold, the rectangle $(1/2 \pm \Delta d,\ 1/2(n-1/2) \pm \Delta p)$ must not extend below the bottom of the overlap region defined by $rp_j = (n-2/3)d$. Such a rectangle is indicated by the dashed lines in Fig. 6. Since this rectangle is not unique, there is some available tradeoff between Δp and Δd. To achieve more quantitative results, we now limit the analysis to the special but useful case in which the radix is of the form $r = 2^{2k}$, where k is a positive (nonzero) integer.

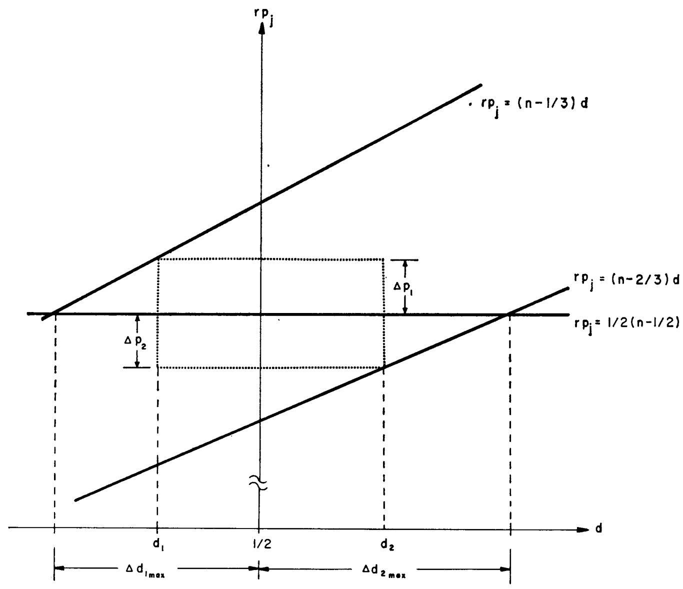

Fig. 6. Cost calculation from P–D plot.

A division with $r = 2^{2k}$ may be implemented with a cascade of k adder–subtractors, with multiples of 1 time and 2 times the divisor available to the first stage of the cascade, multiples of 4 times and 8 times to the second, and so forth through multiples of $2^{(2k-2)}$ times and $2^{(2k-1)}$ times available to the kth stage. In this case, n, the largest multiple of the divisor which may be formed, is the sum of the largest multiples which may be formed at each stage in the cascade, i.e., $n = 2 + 8 \cdots 2^{(2k-1)}$. Furthermore, the sum of this geometric series is $n/(r-1) = 2/3$. Thus we shall consider the case $r = 2^{2k}$, $n = 2/3(r-1)$.

For practical implementation, the rectangular region defined horizontally by Δp will be symmetric about $d = 1/2$ and $rp_j = 1/2(n-1/2)$. Referring to Fig. 6, note that Δd must be smaller than the smaller of $\Delta d_{1\,\max}$ and $\Delta d_{2\,\max}$. The following demonstrates that $\Delta d_2 < \Delta d_{1\,\max}$.

$$\Delta d_{2\,\max} = 1/2\left(\frac{n-1/2}{n-2/3} - 1\right)$$
$$\Delta d_{1\,\max} = 1/2\left(\frac{-n-1/2}{n-1/3} + 1\right) \tag{10}$$

$$\Delta d_{1\,\max} - \Delta d_{2\,\max} = 1 - \frac{n^2-n+1/4}{n^2-n+2/9}, \tag{11}$$

since

$$\frac{n^2-n+1/4}{n^2-n+2/9} > 1,$$

$$\Delta d_{1\,\max} - \Delta d_{2\,\max} < 0 \tag{12}$$

$$\Delta d_{1\,\max} < \Delta d_{2\,\max}.$$

Thus, choosing $\Delta d \leq \Delta d_{1\,\max}$ will insure that the rectangle will fit horizontally.

Similarly

$$\begin{aligned} \Delta p_1 &= (n - 1/3)d_1 - 1/2(n - 1/2) \\ \Delta p_2 &= -(n - 2/3)d_2 + 1/2(n - 1/2) \end{aligned} \tag{13}$$

$$\begin{aligned} \Delta p_1 - \Delta p_2 &= (n - 1/3)d_1 + (n - 2/3)d_2 \\ &\quad - (n - 1/2). \end{aligned} \tag{14}$$

Let

$$\begin{aligned} d_1 &= 1/2 - \Delta d \\ d_2 &= 1/2 + \Delta d. \end{aligned} \tag{15}$$

Substituting (15) into (14) yields

$$\Delta p_1 - \Delta p_2 = \frac{-\Delta d}{3} \leq 0$$

and thus $\Delta p_1 \leq \Delta p_2$.

As implied earlier, if we are certain that $r\hat{p}_j = 1/2(n-1/2)$ will produce the quotient selection $q_{j+1} = n$, then $\Delta p \leq \Delta p_2$ will be sufficient. If we cannot guarantee this, then $\Delta p \leq \Delta p_1$ must hold.

We shall adopt the latter, more cautious approach. If we selected the former, then the $(n-1/3)$ term in (21) would be replaced by $(n-2/3)$. The results in Table II, however, will be the same.

Recalling that $\Delta d = 2^{-\delta}$, we want

$$2^{-\delta} \leq \Delta d_{1\,\max}, \tag{16}$$

which from (10) becomes

$$2^{-\delta} \leq 1/2\left(\frac{n - 1/2}{n - 1/3} - 1\right) \tag{17}$$

where

$$n = 2/3(2^{2k} - 1).$$

Let

$I(x) = x$ if x is an integer
$\quad = $ next larger integer if x is not an integer.

The minimum value of δ is therefore

$$\delta_{\min} = -I\left(\log_2\left(1/2\left(1 - \frac{n - 1/2}{n - 1/3}\right)\right)\right). \tag{18}$$

Possible values of δ are thus

$$\delta = \delta_{\min}, \delta_{\min} + 1, \cdots, m. \tag{19}$$

Similarly, since $\Delta p = 2^{-\epsilon}$, combining the fact that $\Delta p_1 \leq \Delta p_2$ with (13) yields

$$2^{-\epsilon} \leq 1/12 - 2^{-\delta}(n - 1/3) \tag{20}$$

and thus

$$\epsilon = -I(\log_2(1/12 - 2^{-\delta}(n - 1/3))) \tag{21}$$

where δ is defined by (19).

TABLE II
COSTS FOR ARITHMETIC MODEL

k	r	n	δ	ϵ	N_d	N_p
1	4	2	$\delta_{\min}=$ 5	5	5	7
			6	5	6	7
			7	4	7	6
			8	4	8	6
			·	·	·	·
			·	·	·	·
			·	·	·	·
			m	4	m	6
2	17	10	$\delta_{\min}=$ 7	7	7	11
			8	5	8	9
			9	4	9	8
			10	4	10	8
			·	·	·	·
			·	·	·	·
			·	·	·	·
			m	4	m	8
3	64	42	$\delta_{\min}=$ 9	9	9	15
			10	5	10	11
			11	4	11	10
			12	4	12	10
			·	·	·	·
			·	·	·	·
			·	·	·	·
			m	4	m	10
4	256	170	$\delta_{\min}=$ 11	11	11	19
			12	5	12	13
			13	4	13	12
			14	4	14	12
			·	·	·	·
			·	·	·	·
			·	·	·	·
			m	4	m	12

Now let

N_d = number of bits of $\hat{d} = \delta$
N_p = number of bits of $r\hat{p}_j = \epsilon + 2k$.

Note also that the sign of d and rp_j must be known to model. Table II summarizes the results of (19) and (21) for $k = 1, 2, 3, 4$. Note that ϵ approaches a lower limit of 4 when the 1/12 term in (21) becomes dominant.

Thus it appears there are three feasible cases for which the cost of inspection is as follows.

Case 1:

$$\begin{aligned} N_p &= 4k + 3 \\ N_d &= 2k + 3. \end{aligned}$$

Case 2:

$$\begin{aligned} N_p &= 2k + 5 \\ N_d &= 2k + 4. \end{aligned}$$

Case 3:

$$\begin{aligned} N_p &= 2k + 4 \\ N_d &= 2k + 5. \end{aligned}$$

Case 3 would probably be the most practical case to implement since N_p is minimum. N_p bits of the redundantly represented partial remainder must be converted into conventional form before each model division. Since this assimilation is essentially a serial process, the assimilation time is directly proportional to N_p.

Cost Determination for a Table Look-Up Model

This class of models is a logical implementation of the P–D diagram. In its most brute force form, this model may be viewed as a grid or matrix with vertical lines which are the outputs of decoders applied to $\hat{d}$ and with horizontal lines which are the outputs of the decoders applied to $r\hat{p}_j$. At each intersection of the lines is an AND gate with one input connected to the vertical line, the other to the horizontal line. Each point of intersection corresponds to a quotient digit value i, and thus the output of each AND gate is connected to the input of the appropriate OR gate, the true output of which is $q_{j+1}=i$.

The overlap regions are divided by steps, as discussed earlier, such that the cost criterion will hold in all intervals. To determine the required N_p and N_d in this case, we again consider the worst-case region of the P–D plot where $d=1/2$ and between $q(n)$ and $q(n-1)$ as shown in Fig. 6.

Again, if we choose the dividing line between $q_{j+1}=n$ and $q_{j+1}=n-1$ to be at $1/2(n-1/2)$, then the calculations from the arithmetic model also hold for the table look-up case with $r=2^{2k}$. Recall, however, that we generally wish to minimize N_p since this will reduce the assimilation time in forming $r\hat{p}_j$ in each cycle. We can accomplish this by selecting the comparison constants, the dividing line between choice of quotient digit values, as close to the top of an overlap region as possible.

In the arithmetic models, the comparison constants are implicit in the model; thus, for example, we had no choice but to use $1/2(n-1/2)$ in the cost calculations. In the present case, however, we may select any value which is within the overlap region and an integer multiple of $2^{-\epsilon}$.

The value of $1/2(n-1/2)$ is always an exact binary number, specifically, a number with a fractional part of 3/4. The distance from $1/2(n-1/2)$ to the upper limit of the overlap region along $d=1/2$ is $1/2(n-1/3)-1/2(n-1/2)=1/12$. This means that the largest comparison constant we may choose in this region without increasing ϵ to be greater than 4 is $1/2(n-1/2)+1/16$. If we design the logic such that $r\hat{p}_j=1/2(n-1/2)+1/16$ and $\hat{d}=1/2$ selects $q_{j+1}=n$, then Δd and Δp cost calculations are as follows:

$$2^{-\delta} \leq \Delta d_{\max}$$

$$2^{-\delta} \leq 7/48 \cdot \frac{1}{n-2/3}$$

$$2^{-\epsilon} \leq 7/48 - 2^{-\delta}(n-2/3).$$

In the manner outlined in the last section we obtain Table III and the three cases.

Case 1:

$$N_p = 2k+4$$
$$N_d = 2k+3.$$

TABLE III
Costs for Table Look-Up Models

k	n	δ	ϵ	N_d	N_p
1	2	$\delta_{\min}=$ 4	4	4	6
		5	4	4	6
		6	3	4	6
		7	3	3	5
		.	.	.	.
		.	.	.	.
		.	.	.	.
		m	3	m	5
2	10	$\delta_{\min}=$ 7	4	7	8
		8	4	8	8
		9	3	9	7
		.	.	.	.
		.	.	.	.
		.	.	.	.
		m	3	m	7
3	42	$\delta_{\min}=$ 9	4	9	10
		10	4	10	10
		11	3	11	9
		.	.	.	.
		.	.	.	.
		.	.	.	.
		m	3	m	9
4	170	$\delta_{\min}=$ 11	4	11	12
		12	4	12	12
		13	3	13	11
		.	.	.	.
		.	.	.	.
		.	.	.	.
		m	3	m	11

Case 2:

$$N_p = 2k+4$$
$$N_d = 2k+4.$$

Case 3:

$$N_p = 2k+3$$
$$N_d = 2k+5.$$

The first entry $N_d=4$, $N_p=6$ is not included in the above linear equations, but this is the most practical case for $k=1$, radix four. By comparison with the results in Table II note that for a given k, a case may be found for which a table look-up model requires fewer bits of comparison than the corresponding arithmetic model.

Quotient Conversion

The quotient developed by higher-radix SRT division will, in general, include negative digits and eventually must be converted to a conventional binary form. This conversion time and hardware is the greater part of the price paid for the accrued advantages of redundancy.

First consider a specific case: conversion of a result produced by a nonrestoring division. Here quotient representation is the same as that discussed earlier except that 0 is *not* an allowable digit. This conversion may be performed sequentially as the quotient digits are generated, and thus requires no additional terminal operations. The digit q_{j+1} is unchanged if it is positive; otherwise, it is replaced by $r+q_{j+1}$, and the adjacent higher-order digit q_j, decreased by 1. Note that since 0 is not a permissible digit, there is no requirement for a borrow propagation in decreasing q_j by 1. The hardware required is of the order of a 2-digit subtractor.

It is not generally possible, however, to perform SRT division not allowing $q=0$. Nonrestoring division may be viewed as SRT division with $n=r-1$. For this case, the $q(0)$ region of a P–D plot is completely overlapped by the $q(1)$ and $q(-1)$ regions. The quotient digit value $q=0$ may, therefore, be eliminated and the conversion consequently simplified. For cases of division with $n<r-1$, the $q(0)$ region is not subsumed by other regions, and thus $q=0$ must be allowed if the division is to be completely defined.

With the possibility of $q=0$, the conversion is complicated, for now the difference q_j-1 may require a borrow from q_{j-1}. Furthermore, this borrow must propagate to the left until it encounters a nonzero digit. This potential for borrow propagation requires that the equivalent of a full precision subtractor be available to the quotient register if conversion is to occur as the quotient digits are generated.

Alternately, the full precision quotient may be generated and stored in the redundant form and then converted during an extra terminal step. A high-speed arithmetic unit frequently employs a redundant representation of the partial product during multiplication, e.g., carry–save adders, which also require a terminal conversion. One possibility, then, is to share the hardware for conversion of both products and quotients.

An Implementation

Higher-radix SRT division is being used in the high-speed arithmetic unit of the Illinois Pattern Recognition Computer—Illiac III. This unit employs a cascade of 4 adder–subtractors. At each stage in the cascade, a radix-four, table look-up model division selects a radix-four quotient digit which is used to select the addition or subtraction of a multiple of the divisor from the partial remainder at the next stage. Thus in one pass of the partial remainder through the cascade, 8 bits of the quotient are formed.

Readers interested in the details of this unit are invited to contact the author. Implementation is also discussed in [7].

Acknowledgment

The author wishes to acknowledge the helpful discussions with Prof. J. E. Robertson, Prof. S. R. Ray, and R. T. Borovec concerning material in this paper.

References

[1] M. Nadler, "A high speed electronic arithmetic unit for automatic computing machines," *Acta Tech.* (Prague), no. 6, pp. 464–478, 1956.
[2] C. V. Freiman, "Statistical analysis of certain binary division algorithms," *Proc. IRE*, vol. 49, pp. 91–103, January 1961.
[3] J. E. Robertson, "A new class of digital division methods," *IRE Trans. Electronic Computers*, vol. EC-7, pp. 218–222, September 1958.
[4] T. D. Tocher, "Techniques of multiplication and division for automatic binary computers," *Quarter. J. Mech. App. Math.*, vol. 2, pt. 3, pp. 364–384, 1958.
[5] J. E. Robertson, "Methods of selection of quotient digits during digital division," Dept. of Computer Science, University of Illinois, Urbana, Ill., File 663, 1965.
[6] C. S. Wallace, "Suggested design for a very fast multiplier," Dept. of Computer Science, University of Illinois, Urbana, Ill., Rept. 133, pp. 8–9, 1963.
[7] D. E. Atkins, "The theory and implementation of SRT division," Dept. of Computer Science, University of Illinois, Urbana, Ill., Rept. 230.
[8] O. L. MacSorley, "High-speed arithmetic in binary computers," *Proc. IRE*, vol. 49, pp. 67–91, January 1961.
[9] J. B. Wilson and R. S. Ledley, "An algorithm for rapid binary division," *IRE Trans. Electronic Computers*, vol. EC-10, pp. 662–670, December 1961.
[10] G. Metze, "A class of binary divisions yielding minimally represented quotients," *IRE Trans. Electronic Computers*, vol. EC-11 pp. 761–764, December 1962.
[11] I. Flores, *The Logic of Computer Arithmetic.* Englewood Cliffs, N. J.: Prentice-Hall, 1963.