A SIGNED BINARY MULTIPLICATION TECHNIQUE

By ANDREW D. BOOTH

(*Birkbeck College Electronic Computer Project, 21 Torrington Square, London, W.C.1*)

[Received 1 August 1950]

SUMMARY

A technique is described whereby binary numbers of either sign may be multiplied together by a uniform process which is independent of any foreknowledge of the signs of these numbers.

IN the design of automatic computing machines it is necessary to have available some means of multiplying together two numbers whose signs are not necessarily positive. This, of course, is completely trivial when the process is to be performed by a human operator since a large number of processes exist. However, few of these seem to be suitable for mechanization with the types of circuit currently available on account of the complexity of the discrimination required for their execution, and the problem is to find a procedure which can be engineered with the minimum of equipment. Several ways of accomplishing this have been used to date, all more or less unsatisfactory, for example:

(1) the machine may use numbers in the form (sign) (absolute value of number), in which case, although multiplication (and division) are particularly simple, the much more frequent operation of subtraction needs special circuitry;

(2) negative numbers may be represented in complementary form mod 2^p

when it is necessary either first to convert them to positive form, multiply, and then to correct the resulting product to its signed value by means of a special sub-routine; or to apply appropriate corrections to the product, obtained in the usual way, by neglecting the fact that the numbers may be non-positive. The nature of these corrections is seen from the following discussion.

Assume that the machine in question (this is the case in the machines under construction in this and many other laboratories) deals with negative numbers by taking their complements mod 2, then:

$$+m \equiv m$$

$$-m \equiv 2-m.$$

Whence if two numbers m and r are to be multiplied, the machine will generate the following results:

$$+m\times+r = +mr, \tag{a}$$

$$-m\times+r = 2r-mr, \tag{b}$$

$$+m\times-r = 2m-mr, \tag{c}$$

$$-m\times-r = 4-2m-2r+mr. \tag{d}$$

Thus, in order to correct (b), (c), and (d), it is necessary to apply the following process:

(1) if m is negative, subtract $2r$ from the product obtained in the normal manner;

(2) if r is negative, subtract $2m$ from the product obtained in the normal manner;

the application of *both* of these corrections also gives correct results in case (d), since if m and r are negative, subtraction is in effect addition and since operations are all mod 2 the added 4 is in any case ignored by the machine.

It is evident that the application of this correction process involves examination, by the machine, of the signs of both m and r and this, in turn, requires for the efficient engineering of the sequence the storage of the signs of m and r in auxiliary circuits (**1**).

Such correction operations as envisaged above are highly undesirable, and it is natural to inquire whether any process exists whereby multiplication can be performed in a uniform manner without the necessity of any special devices to examine the signs of the interacting numbers. That this was a reasonable quest was rendered probable by the development, by Burks, Goldstein, and von Neumann (**2**), of the so-called 'non-restoring' process for the division of signed binary numbers, and a somewhat complicated process for multiplication had in fact been suggested previously by Rey and Spencer (**3**).

An extremely simple process (both mathematically and technically) has now been evolved and forms the subject of this note; it was suggested by the standard 'shortcutting' method of multiplication used on desk machines operating in decimal scale. Thus, to multiply by 057737 (i.e. +57737) on a desk calculating machine using the shortcutting method, the operator effectively multiplies by $1\,\bar{4}\,\bar{2}\,\bar{3}\,4\,\bar{3}$. Similarly to multiply by the number 977563 (i.e. -22437) the operator effectively multiplies by $1\,0\,\bar{2}\,\bar{2}\,\bar{4}\,\bar{4}\,3$. The corresponding process for binary numbers gives for multiplication by the positive numbers 0111011, multiplication by $1\,0\,0\,\bar{1}\,1\,0\,\bar{1}$. The process is precisely this, provided the multiplication starts with the least significant digit, and may be described as follows:

To multiply two numbers m and r together, examine the nth digit (m_n) of m,

(1) If $m_n = 0$, $m_{n+1} = 0$, multiply the existing sum of partial products by 2^{-1}, i.e. shift one place to the right.

(2) If $m_n = 0$, $m_{n+1} = 1$, add r into the existing sum of partial products and multiply by 2^{-1}, i.e. shift one place to the right.

(3) If $m_n = 1$, $m_{n+1} = 0$, subtract r from existing sum of partial products and multiply by 2^{-1}, i.e. shift one place to right.

(4) If $m_n = 1$, $m_{n+1} = 1$, multiply the sum of partial products by 2^{-1}, i.e. shift one place to the right.

(5) Do not multiply by 2^{-1} at m_0 in the above processes.

Note that if m is given to n digits, at the start of the process it is assumed that $m_{n+1} = 0$.

In proving this process it will be assumed that operations are (mod 2). Thus

$$m \equiv m_0 . m_1 m_2 ... m_n$$
$$\equiv m_0 . 2^0 + 2^{-1}m_1 + 2^{-2}m_2 ... + 2^{-n}m_n ... + 2^{-N}m_N \quad (m_n = 0, 1).$$

Now consider the multiplication of r by the number $0.0...01_n 00...0$, i.e. by 2^{-n}. It is seen that the process (1)–(5) gives, for the product:

$$-r \times 2^{-n} + r \times 2^{-n+1},$$

i.e.
$$r \times 2^{-n}(2-1) = 2^{-n}r,$$

which is the correct result whatever the sign of r since the multiplication by 2^{-n} is achieved by successive right-shift operations.

Thus, up to m_0, the following results will be obtained in the complete multiplication by m:

$$(+m) \times r \text{ is correct for all signs of } r,$$

$$(-m) \times r = (1-m)r = r - mr \text{ for all signs of } r.$$

At stage m_0, however, the quantity $m_0 . r$ $(m_0 = 0, 1)$ is subtracted from these results, giving:

$$(+m) \times r - 0 \times r = +mr \text{ for all signs of } r,$$

$$(-m) \times r - 1 \times r = (1-m)r - r = -mr \text{ for all signs of } r.$$

Whence the process outlined gives the correct product for $m \times r$ (mod 2) whatever the signs of m and r.

In conclusion an example of the application of the procedure to each possible combination of sign will render the process clear.

(1)		$m = 0{\cdot}101\ (+\frac{5}{8})$	$r = 0{\cdot}110\ (+\frac{3}{4})$	
	$m_3 = 1\ (m_4 = 0)$	subtract r		1·010
		shift right		1·101,0
	$m_2 = 0,\ m_3 = 1$	add r	1·1010	
			0·110	
			0·0110	
		shift right		0·001,10
	$m_1 = 1,\ m_2 = 0$	subtract r	0·00110	
			1·010	
			1·01110	
		shift right		1·101,110
	$m_0 = 0,\ m_1 = 1$	add r	1·101,110	
			0·110	
			0·011,110 (mod 2)	
		no shift		0·011,110 $(= +\frac{15}{32})$
(2)		$m = 1{\cdot}011\ (-\frac{5}{8})$	$r = 0{\cdot}110\ (+\frac{3}{4})$	
	$m_3 = 1\ (m_4 = 0)$	subtract r	1·010	
		shift right		1·101,0
	$m_2 = 1,\ m_3 = 1$	shift right		1·110,10
	$m_1 = 0,\ m_2 = 1$	add r	1·110,10	
			0·110	
			0·100,10	
		shift right		0·010,010
	$m_0 = 1,\ m_1 = 0$	subtract r	0·010,010	
			1·010	
			1·100,010	
		no shift		1·100,010 $(= -\frac{15}{32})$
(3)		$m = 0{\cdot}101\ (+\frac{5}{8})$	$r = 1{\cdot}010\ (-\frac{3}{4})$	
	$m_3 = 1\ (m_4 = 0)$	subtract r	0·110	
		shift right		0·011,0
	$m_2 = 0,\ m_3 = 1$	add r	0·011,0	
			1·010	
			1·101,0	
		shift right		1·110,10
	$m_1 = 1,\ m_2 = 0$	subtract r	1·110,10	
			0·110	
			0·100,10	
		shift right		0·010,010
	$m_0 = 0,\ m_1 = 1$	add r	0·010,010	
			1·010	
			1·100,010	
		no shift		1·100,010 $(= -\frac{15}{32})$

(4) $m = 1{\cdot}011\ (-\tfrac{5}{8})$ $r = 1{\cdot}010\ (-\tfrac{3}{4})$

$m_3 = 1\ (m_4 = 0)$	subtract r	0·110	
	shift right		0·011,0
$m_2 = 1,\ m_3 = 1$	shift right		0·001,10
$m_1 = 0,\ m_2 = 1$	add r	0·001,10	
		1·010	
		1·011,10	
	shift right		1·101,110
$m_0 = 1,\ m_1 = 0$	subtract r	1·101,110	
		0·110	
		0·011,110	
	no shift		0·011,110 $(+\tfrac{15}{32})$

REFERENCES

1. A. D. Booth and K. H. V. Britten, *General Considerations in the Design of an Electronic Computer* (Princeton, 1947).
2. A. Burks, H. Goldstein, and J. von Neumann, *Logical Design of an Electronic Computing Instrument* (Princeton, 1946).
3. T. J. Rey and R. E. Spencer, *Sign Correction in Modulus Convention*, Cambridge Conference Report, 1950.