THE ENGLISH ELECTRIC COMPANY LIMITED

DEPT. COMPUTER DEVELOPMENT
KIDSGROVE
F.S.D. or Order No.

REPORT No. K/GD.y.99.
A.C.E.D. Ref. No:
Deptl. Ref.
Date 17.6.63.

Copies	TO:-
1	Mr. J. Boothroyd, Chief Instructor, Kidsgrove.
1	Mr. T.H. Elliott, Chief Product Design Engineer, Kidsgrove.
1	Mr. C. Robinson, Computing Services Manager, Kidsgrove
1	Mr. A.C.D. Haley, Chief Development Engineer, Kidsgrove.
1	Mr. G.D. Royle, Group Engineer, F.E.S., Kidsgrove.
1	Mr. R.H. Allmark, Computer Development, Kidsgrove.
12	KDF 9 Engineers
35	Instruction Department, Kidsgrove.
1	Mr. A.B. Barnes, London.
1	Chief Engineer's File

THE KDF 9 MULTIPLIER-DIVIDER UNIT
PART A
C.2

Report by
J.R. ELLISON

Summary

This report is a description of the KDF 9 Multiplier-Divider Unit. The report comprises two parts:

PART A is an introduction to multiplication and division in a digital computer and describes the logical building blocks which are required to effect these operations in KDF 9.

PART B is an engineering description of the Multiplier-Divider Unit of KDF 9 with respect to the hardware, its timing and certain aspects of its logical organisation. Special circuits which are used mainly in the Multiplier-Divider Unit are also described.

Contents

1.0. General
2.0. Glossary of Abbreviations
3.0. An Introduction to Multiplication and Division
4.0. Basic Multiplication
5.0. Basic Division

1.0. GENERAL

The KDF 9 Multiplier-Divider (M.D.) Unit occupies one cubicle, and may be regarded as an extension to the Arithmetic Unit (A.U.), which is described in Report K/GD.y.77. The M.D. Unit is provided with operands by the A.U. from the Nesting Store (N.S.), and a multiplication or division is started by Arithmetic Control (A.C.), which then waits until the operation is complete. In this respect the M.D. Unit is an autonomous unit with its own timing and control section which restarts A.C. when it has completed the desired operation, at the same time placing the result(s) of that operation back into the N.S.

Multiplication and division operations are controlled by A.C. sequences. A.C. is described in Report K/GD.u.331. There are seven sequences involving multiplication, namely :-

X	XD	XF	XDF
X + F	$X \pm n$	X + Cq	

and six sequences involving division:-

$\div$ R	$\div$ D $\div$ I	$\div$ F	$\div$ DF

The sequences will be referred to in this report but are not described in detail. The sequences perform the major portion of the tasks which do not involve multiplication or division proper, namely exponent manipulation in floating point working, rounding off of results and overflow detection. Many of these functions are carried out whilst the multiplication or division is in operation, although terminal operations upon results, e.g. rounding off, are necessarily carried out after the results have been placed back in the A.U.

The M.D. Unit modifies its actions slightly in some sequences, notably floating point multiplications and divisions and single or double-length fixed point divisions. These modifications will be described in detail in this report.

PART A of this report is a description of the logical building blocks which are required to effect multiplication and division with particular emphasis on the KDF 9 M.D. Unit.

PART B is an explanation of the timing of M.D. control and certain aspects of its logical organisation, together with a description of the special circuits used mainly in the M.D. Unit.

Abbreviations are used extensively in the report. After the first use of any word which may be used again an abbreviation will be given in brackets. Thereafter the abbreviation will be used unless this is considered to be confusing. A list of abbreviations follows on the next page:-

2.0. GLOSSARY OF ABBREVATIONS

Abbreviation	Name	Notes
A.C.	Arithmetic Control	
A.R.	Assimilated Remainder	
A.U.	Arithmetic Unit	or Mill
B.1.	Register in A.U.	48 Bits
B2	Register in A.U.	48 bits
B.P.	Binary Point	between D0 and D1
C	Stored Carry	
C1	Stored Carry Register 1	48 bits
C1a	Stored Carry Register 1a	4 bits
C2	Stored Carry Register 2	48 bits
Dn	Digit n of a Register	$0 \leq n \leq 47$
Dd	Dividend	
Dr	Divisor	
G.D.	Guard Digit	
l.s.	least (or less) significant	
M	Multiplier (operand)	
M1	Register used to store M and P or -Q	48 bits
M2	Register used to store M and P or -Q	48 bits
M3	Register used to store Dd (l.s.) or +Q	48 bits
M4	Register used to store Dd (l.s.) or +Q	48 bits
M5	Register used to store digits of P	4 bits
M.D.	Multiplier-Divider Unit or Register used to store Multiplicand and Divisor	48 bits
m.s.	most (or more) significant	
N.S.	Nesting Store	
N.S.L.1	Next Step Logic 1	associated with PS1 & C1
N.S.L.2	Next Step Logic 2	associated with PS1a & C1a
P	Product	
P.P.	Partial Product	
P.R.	Partial Remainder	
P.S.	Partial Sum	
P.S.1	Partial Sum Register 1	48 bit
P.S.1.a	Partial Sum Register 1a	4 bit
PS2	Partial Sum Register 2	48 bit
Q	Quotient	
+Q	Positive part of Q	
-Q	Negative part of Q	
T.E.	Truncation Error	
U.P.R.	Unshifted Partial Remainder	= PR ÷ 2
W1	Register in A.U.	48 bits
W2	Register in A.U.	48 bits

3.0. AN INTRODUCTION TO MULTIPLICATION AND DIVISION

The following sections of the report show how parallel binary multiplication and division may be carried out in a digital computer, with the emphasis placed on the methods used in KDF 9. The multiplier and divider logics in KDF 9 are intimately associated, but the common areas have been arranged to optimise multiplication rather than division when speed of operation is involved, since multiplication is regarded as the more important machine instruction.

The M.D. Unit represents a compromise between speed of operation and the amount of equipment required. Thus more simple methods involve less equipment at the cost of speed, whilst more sophisticated methods, of which there are several, were regarded as being too costly in proportion to the extra operating speed obtainable.

4.0. BASIC MULTIPLICATION

Any multiplication method may be regarded as the formation of a Product (P) which is directly related to the two numbers which are multiplied (operands) namely the Multiplier (M) and the Multiplicand (MD). The process of multiplication may be regarded as one involving a series of additions and shifts.

The normal pencil and paper method of decimal multiplication affords a good starting point for the explanation. It consists of producing several intermediate results called Partial Products (P.P.'s), which as a final step are added together to form the Product (P). The P.P's are formed by inspecting the digits of M one-at-a-time, and multiplying M.D. by the value of the M digits, taking account of the power of 10 to which the P.P. must be raised, by shifting.

Example 1 546 x 279 decimal

MD	546	
M	279	
PP1	4914	$= 9 \times 10^{0}_{1} \times 546$
PP2	3822	$= 7 \times 10^{1}_{2} \times 546$
PP3	1092	$= 2 \times 10^{2} \times 546$
P	152334	

The process is more easily mechanised if the P.P's are added as they are formed, since if the basic process shown above was adopted a large number of P.P's would have to be stored and finally added together, if M was large.

A more practical process in terms of electronic circuitry would be :-

1 Make M.D. available in a register and M in another register

2 Inspect the least significant (L.S.) digit of M, and add M.D. into a third register PP (which initially contains zero), the number of times specified by the L.S. digit of M.

3 Shift M and P.P. one place right (L.S.), losing the digit of M just inspected. Store the L.S. digit of P.P. in the M.S. digit position of M which was left empty when M was shifted right.

4 Finish when all digits of M have been inspected.

A further decimal example is given using the method. M is placed in brackets in order to distinguish P.P. and M digits.

Example 2 144 x 203 decimal

Initially we have in the registers:-

P.P. = 0 0 0 M = 2 0 3

M.D. = 1 4 4

M DIGIT INSPECTED

Steps

1	Add M.D. x 3 to P.P.	∴ P.P. = 4 3 2	M = (2 0 3)
2	Shift P.P. & M right	0 4 3	2(2 0)
3	Add M.D. x 0 to P.P.	0 4 3	2(2 0)
4	Shift P.P. & M right	0 0 4	3 2(2)
5	Add M.D. x 2 to P.P.	2 9 2	3 2(2)
6	Shift P.P. & M right	0 2 9	2 3 2
7	Finish		

Result P = 2 9 2 3 2

The example illustrates how it is possible to store P.P. digits in the M register as a shift takes place. M therefore is a double-purpose register storing both M and P.P. digits, and thus effectively saving a register.

It is also easy to devise a multiplication method whereby the digits of M are inspected starting at the M.S. end, in which case P.P. and M are shifted left after each step. In this case however, the addition of M.D. at any step will involve a double-length addition, which in turn involves more equipment and takes longer. Also the digit of P.P. which spills after the shift cannot be stored in the M register, since untreated digits of M will be disturbed by the subsequent addition unless this addition is made variable-length. An additional register if therefore required so that a double-length P.P. may be stored. For this reason and because of adder complication the method may be rejected.

The method of Example 2 may be used equally well for the multiplication of binary numbers, and in one respect becomes more simple. Since the digits of M can only be 0 or 1 in binary either M.D. is not added to P.P. (M.D. x 0) or M.D. is added to P.P. (M.D. x 1), whereas in the decimal case digits of M may be 0 to 9 inclusive and M.D. may have to be

added up to 9 times at each step. In this respect Example 2 would be more complicated if M had been 999, and several small sub-multiplications of M.D. would have had to be carried out.

The following example shows a 6-bit binary multiplication.

Example 3 0 0 1 0 1 1 x 0 0 1 1 0 1 binary.

Initially we have in the registers :-

P.P. 0 0 0 0 0 0 M 0 0 1 0 1 1

M.D. 0 0 1 1 0 1

Steps

1	Add M.D. to P.P.	P.P. = 0 0 1 1 0 1	M = (0 0 1 0 1 1)
2	Shift P.P. & M right	0 0 0 1 1 0	1(0 0 1 0 1)
3	Add M.D. to P.P.	0 1 0 0 1 1	1(0 0 1 0 1)
4	Shift P.P. & M.	0 0 1 0 0 1	1 1(0 0 1 0)
5	Do not add M.D.	0 0 1 0 0 1	1 1(0 0 1 0)
6	Shift P.P. & M.	0 0 0 1 0 0	1 1 1(0 0 1)
7	Add M.D.	0 1 0 0 0 1	1 1 1(0 0 1)
8	Shift P.P & M.	0 0 1 0 0 0	1 1 1 1(0 0)
9	Do not add M.D.	0 0 1 0 0 0	1 1 1 1(0 0)
10	Shift P.P.& M.	0 0 0 1 0 0	0 1 1 1 1(0)
11	Do not add M.D.	0 0 0 1 0 0	0 1 1 1 1(0)
12	Shift P.P. & M.	0 0 0 0 1 0	0 0 1 1 1 1
13	Finish		

Result P 0 0 0 0 1 0 0 0 1 1 1 1

In decimal this represents
+ 11 x + 13 = + ~~141~~ 143.

Significant points worth noting at this stage are :-

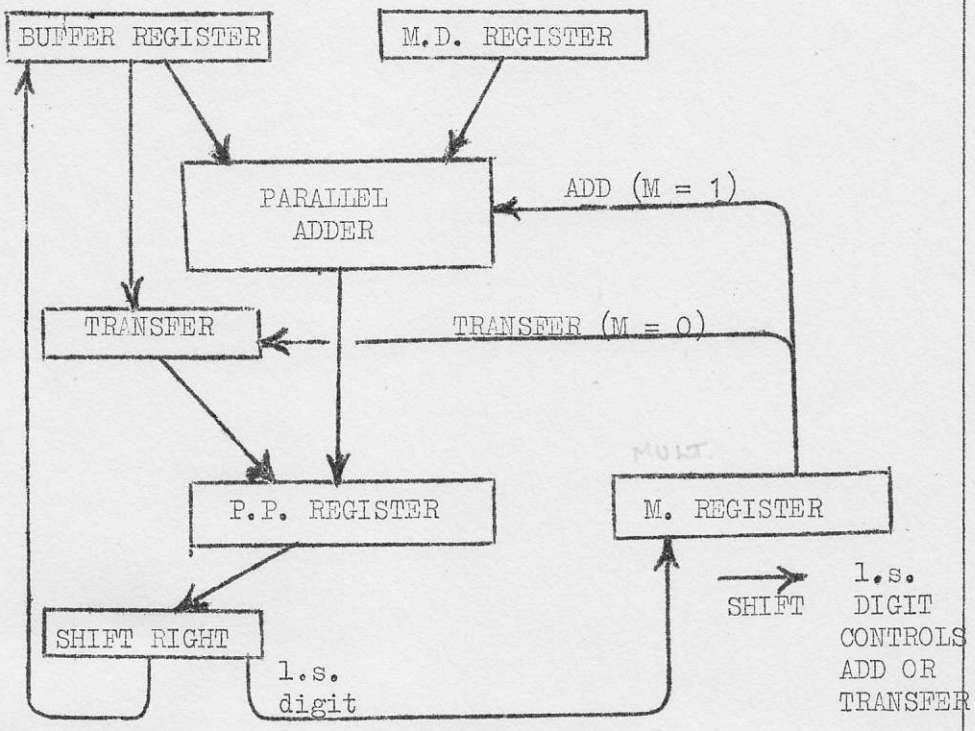
a) A single-length i.e. one word length adder, is required to add M.D. to P.P.

b) Digits of P.P. which are stored in place of M after each shift remain unaltered by subsequent operations and become digits of P.

c) The maximum size of P is restricted to two words, namely the P.P. and M. registers. It will be shown later in the report that this is adequate for a true representation of P, except in one special case when an overflow is indicated.

d) The binary operands, if negative, are in two's complement notation. It will be shown in Sect. 4.4 that sign correction is required to obtain a correctly signed P, if M is negative.

A simple block diagram of the hardware required for the mechanisation of Example 3 is shown :-

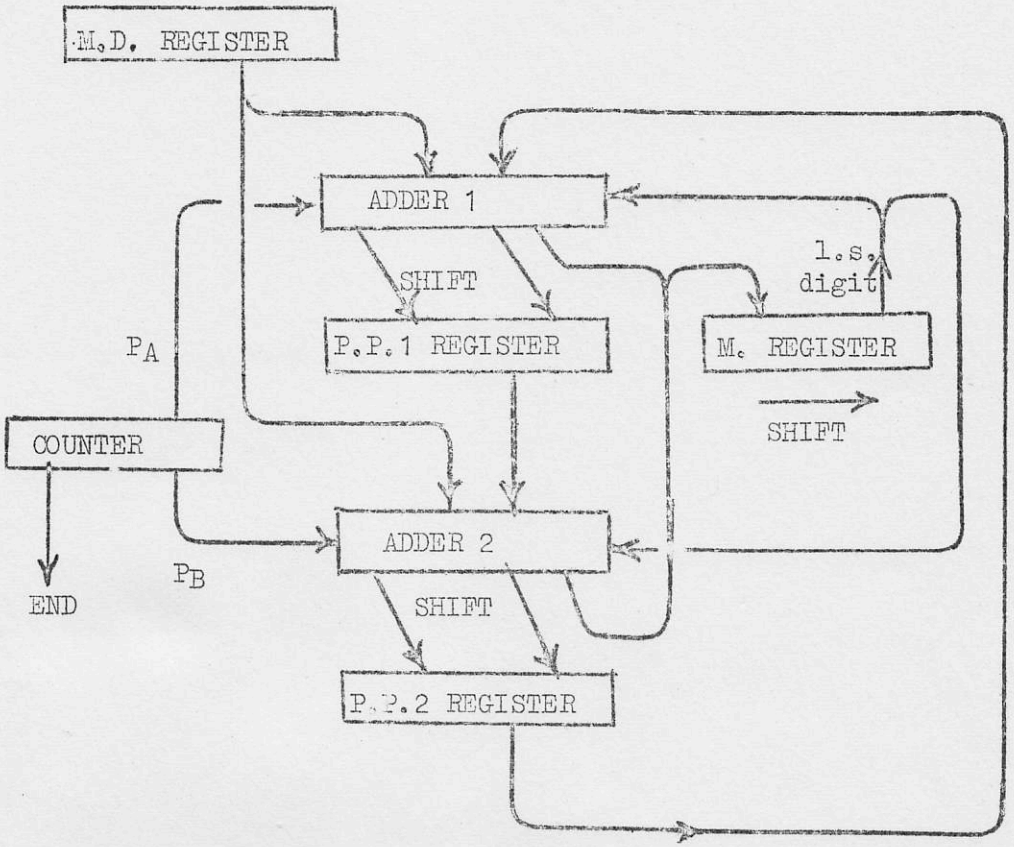


All the registers and the adder are one word long (48 bits in KDF 9). The m.s. bit is at the left of each register. The buffer register is required in order to hold a copy of P.P. shifted right one place, since a current P.P. cannot be used to generate the next P.P. if they are both required to be stored in the same register. The adder only adds M.D. to BUFFER if the current L.S. digit of M is a one, if it is zero BUFFER is merely transferred to P.P. at add time. Thus if the current M digit is zero P.P. is effectively shifted right one place via BUFFER without involving M.D.

4.1. Add and Shift Combined

The requirement for a buffer register is eliminated by combining the add and shift operations and by the introduction of a second adder. The second adder is used to speed up the overall multiplication by replacing the shift right into BUFFER by an addition - plus-shift operation.

The BUFFER now becomes a second P.P. register and all operations become useful ones, as shown in the following block diagram:-



The outputs of an adder are connected one place to the right into the P.P. registers to produce the shift. P.P.'s are produced alternatively by alternate use of the adders which are controlled by two hypothetical pulses PA and PB which are assumed to be interlaced. The digits of M are examined in the same way as before, and when all the digits of M have been examined multiplication is complete. The mechanism which shifts M one place right at each step and stores digits of P in the M register is more fully explained in Section 4.5. It is assumed in the diagram that if the current M digit is 0 the appropriate adder adds zero to the P.P. input, and if the current M digit is 1 the adder adds M.D. to P.P. The counter is used to determine how many digits of M have been dealt with, and to terminate multiplication at the right time.

4.2. The Adders and Speed of Operation

It will be obvious at this stage that multiplication is a fairly lengthy operation when compared with the time required to add two numbers, or shift a number one place, both of which may be regarded as two of the fastest operations possible in KDF 9. It can be seen that 48 such operations are required in the multiplication. Efforts have been made in the M.D. Unit to reduce the addition time by the use of a redundant form of number representation, which will be described in Section 4.3. The resulting increase in speed is only made possible because of the repeated nature of the additions in the multiplication, redundant notation cannot be applied for example in the A.U. adder, without altering the whole logical structure of the A.U.

The time required for the parallel addition of two numbers which are stored in parallel form in registers, depends upon two basic factors :-

a) The time required to set or reset the storage element which is used to store a sum digit. In KDF 9 the register storage element is a transistor flip-flop whose set or reset time depends upon the circuit elements. Whatever the final form of the adder, the time required to store the sum during the addition is therefore fixed by the flip-flop design and can only be improved by the use of faster flip-flops.

b) The time required to assimilate "carries" arising in less significant stages of the addition, and which may affect sum digits far removed from the source of the carries. In the KDF 9 A.U. Adder, the carries are propagated by special circuits which allow a carry to be propagated from the L.S. to the M.S. digit of the sum in a total time of approximately 0.15 μSec. This time, together with the set/reset time of the sum flip-flop which is 0.2 μSec, means that we must allow at least 0.5 μSec for each addition in the scheme outlined in Section 4.1., if the multiplier is to run synchronously with the basic KDF 9 clock pulses. The multiplication time is now 48 x 0.5 = 24 μSec for a 48 bit M. The use of a redundant form of number notation eliminates the need for carry propagation during multiplication, and hence the time for each addition is now limited only by the flip-flop speed. Allowing 0.2 μSec for the flip-flop to Switch we may allow 0.25 μSec for each addition, making the total time for synchronous multiplication 48 x 0.25 = 12 μSec.

4.3. Redundant Number Notation

In a decimal addition such as 5 5 5 5 5 5 + 5 5 5 5 5 5, each sum digit except the L.S. one, is affected by a carry from a stage of less significance. The carries arising in this addition are illustrated by writing them down under the sum :-

Example 4

```
          5 5 5 5 5 5
    +     5 5 5 5 5 5
SUM     1 1 1 1 1 1 0
CARRIES   1 1 1 1 1 1
```

In normal addition, as shown, the carries are automatically assimilated into the sum at each step.

In redundant notation the sum is represented by two numbers called a Pseudo Sum (P.S.) and Carries (C). In redundant notation the example becomes :-

Example 5

```
          5 5 5 5 5 5
        + 5 5 5 5 5 5
P.S.    0 0 0 0 0 0 0
C.      1 1 1 1 1 1
```

Carry digits arising in any stage are not added to P.S. but are stored. A P.S. digit is derived only from the two operand digits in the corresponding position, the effect of carries from lower stages is ignored. Similarly a C digit is only derived from the two operand digits in the next L.S. position. Thus the formation of P.S. and C are completely parallel operations, all digit positions being treated at the same time, whereas the formation of a normal sum may be regarded as a serial operation because carries must propagate from the L.S. to the M.S. end of the sum. Thus P.S. and C representation is basically a faster operation.

The true sum may be obtained by adding P.S. and C by the normal method which propagates carries.

In most arithmetic operations the fact that the sum consists of two parts P.S. and C would be a disadvantage and would conplicate the arithmetic unit and the associated store. In multiplication however the elimination of the carry propagation time is a decided advantage, because the repeated additions during multiplication may be carried out more quickly in redundant form although the final result will also be in redundant form. Thus to obtain the normal form of result at the end of multiplication a final carry propagate addition must be carried out. In order to save equipment, and because the A.U. adder is standing idle at the end of multiplication, this final addition to form P is carried out using the A.U. adder.

The boolean equations for the addition of binary numbers in redundant notation are the same as for normal addition, the carry being stored instead of propagated. Thus:-

P.S. digit = $c \not\equiv x \not\equiv y$
C digit = $c(x + y) + x\,y$

Where x and y are the corresponding operand digits and C is the current carry digit.

These expressions are fully derived in Appendix I, with particular reference to the multiplication operation.

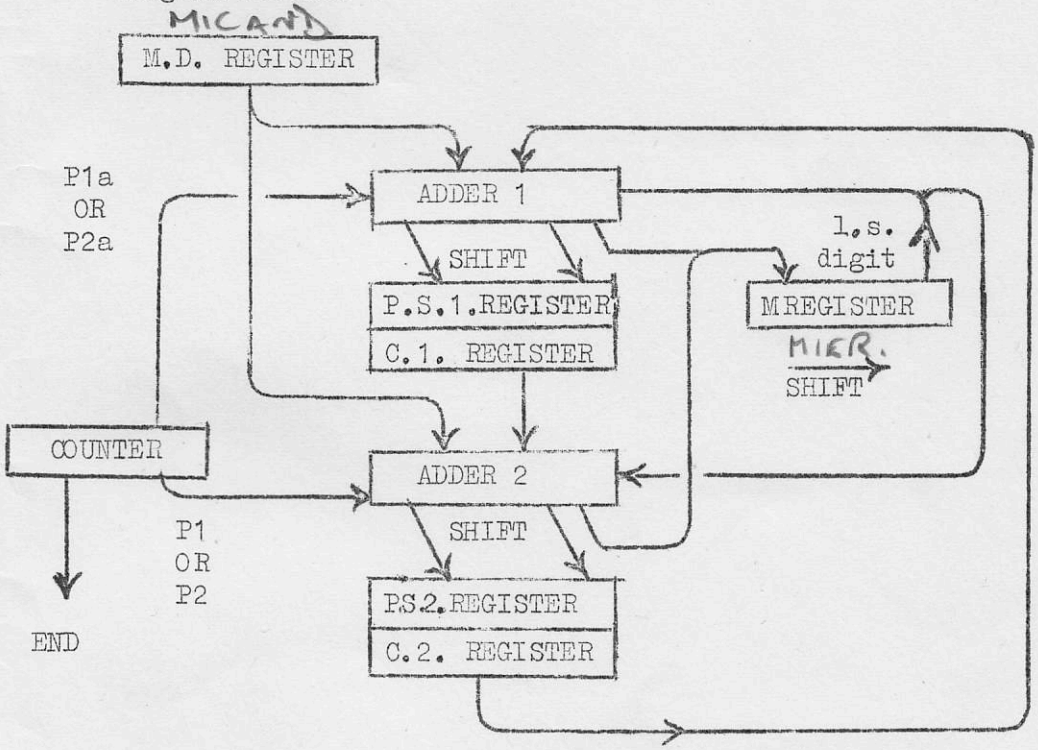
The formation of P.S. and C may also be carried out if one of the operands is in redundant form. The binary addition 13 + 25 + 7 + 6 may be accomplished in redundant notation as follows, by applying the equations given above for P.S. and C :-

Example 6

	0 0 0 1 1 0 1	= 13
Initial C	0 0 0 0 0 0 0	= 0
+	0 0 1 1 0 0 1	= 25
First P.S.	0 0 1 0 1 0 0	
First C	0 0 1 0 0 1 0	
+	0 0 0 0 1 1 1	= 7
Second P.S.	0 0 0 0 0 0 1	
Second C	0 1 0 1 1 0 0	
+	0 0 0 0 1 1 0	= 6
Third P.S.	0 1 0 1 0 1 1	
Third C	0 0 0 1 0 0 0	
True sum	0 1 1 0 0 1 1	= 51

Note that since the initial operands are in non-redundant form, the initial C is zero. The final true sum is obtained by adding the final P.S. and C in the normal way.

The scheme of Sect. 4.1 is now modified. Each P.P. register becomes two registers P.S. and C. The block diagram becomes :-



The adders produce P.S. and C. digits. An addition of M.D., P.S. and C, takes place if the L.S. digit of M is a 1. An addition of P.S. and C only, takes place if the L.S. digit of M is 0. Both operations have a built in shift to the right of one place by connecting the adder outputs to the next L.S. digit of P.S. and C. The addition of P.S. to C. when the L.S. digit of M is 0, has the effect of maintaining the C registers as single-length numbers, i.e. digits previously stored in C cannot spill off the L.S. end of the C register if during a shift right the previous P.S. and C are assimilated. This is an important effect see Sect. 4.6. Note that since the C digits of an addition normally refer to the next M.S. position, the shift right of one place means that the C digit of any one bit position remains in that position, whereas the P.S. digit is shifted right one place and hence the L.S. digit of P.S. spills and is stored in the M.S. bit of the M register.

It has already been indicated in Sect 4.2 that 0.25 μSec is allowed for each addition. The additions are accomplished by means of four basic machine pulses P1, P1a, P2 and P2a. There is a set of these pulses each microsecond. Each pulse occurs once per microsecond and the pulses are interlaced so that their relative spacing is 0.25 μSec. Thus P1 and P2 are used to drive one adder and P1a and P2a are used to drive the second adder. This is shown on the block diagram. Each adder is therefore used every 0.5 μSec.

A further example of binary multiplication (6 bit) is now given to illustrate the use of the P.S. and C registers :-

Example 7 0 0 1 0 1 1 x 0 0 1 1 0 1 (11 x 13)

Initially :-

MD = 0 0 1 0 1 1 M = 0 0 1 1 0 1

P.S.1 0 0 0 0 0 0

C1 0 0 0 0 0 0

Operations

Add M.D.,	PS1 & C1	0 0 1 0 1 1	M =(0 0 1 1 0 1)
	PS2	0 0 0 1 0 1	1(0 0 1 1 0)
	C2	0 0 0 0 0 0	
Add PS2 & C2			
	PS1	0 0 0 0 1 0	1 1(0 0 1 1)
	C1	0 0 0 0 0 0	
Add M.D.		0 0 1 0 1 1	
	PS2	0 0 0 1 0 0	1 1 1(0 0 1)
	C2	0 0 0 0 1 0	
Add M.D.		0 0 1 0 1 1	
	PS1	0 0 0 1 1 0	1 1 1 1(0 0)
	C1	0 0 0 0 1 0	
Add PS1 & C1			
	PS2	0 0 0 0 1 0	0 1 1 1 1(0)
	C2	0 0 0 0 1 0	
Add PS2 & C2			
	PS1	0 0 0 0 0 0	0 0 1 1 1 1
	C1	0 0 0 0 1 0	

True Product = 0 0 0 0 1 0 0 0 1 1 1 1

The true Product P is obtained by adding P.S.1 and C.1. to obtain the M.S. half, the L.S. half being obtained from register M. In KDF 9 P.S.1 and C.1 are added in the A.U. adder, this is more fully explained in Section 4.6. The example given may be directly applied to 48 bit numbers, in which case the final P will be 95 bits in length. The result still appears in P.S.1, C.1 and M.

4.4. Sign Correction

It has been shown that the multiplication procedure outlined in Section 4.3 produces the correct P if the operands are positive. We must now consider the effects of having one or both operands negative.

It will be seen that the correct P will be produced if M.D. is negative and M is positive provided that the shift after each operation is an arithmetic one, i.e. the sign is copied. Multiplication involves the repetitive addition of M.D. and hence the final P must be negative in this case.

If M is negative however, the simple process does not produce the correct result whatever the sign of M.D. For example the binary number 1 1 1 1 1 0 = -2 would be treated as +62 since all M digits which are ones produce additions of M.D. to the Partial Product. The procedure may be suitably modified to produce the correctly signed and valued result for negative M, by a consideration of the properties of a two's complement binary number.

Any two's complement binary number N may be represented by the equation :-

$$N = \sum_{r=n-1}^{r=1} A_r \, 2^{n-1-r} - A_0 \, 2^{n-1}$$

where N is a number with n digits numbered o to n-1, digit o being the most significant and A_r is the binary value of the rth digit of N.
where $0 < r \leq n-1$

In other words N may be considered positive up to, but excluding the sign digit, its sign and value being determined by subtracting 2^{n-1} if the sign (M.S.) digit is a one. For example :-

6 bits	1 1 1 0 1 0	= -6	= +26	-2^5
5 bits	1 1 0 0 0	= -8	= +8	-2^4
4 bits	1 0 1 1	= -5	= +3	-2^3

If this method of representation is applied to the treatment of M digits during multiplication the multiplication procedure becomes :-

1 Make MD available in a register and M in another register.

2 Clear the PS1 and C1 registers

3 Inspect the L.S. digit of M; if it is 0 add P.S. to C; if it is 1 add M.D., P.S. and C.

Shift the result and the contents of M one place right during the addition.

4 Have the L.S. 47 digits of M been inspected?

no ↓ yes

5 Inspect the L.S. digit of M; if it is 0 add P.S. to C; if it is 1 (M negative) subtract M.D. from P.S. and C. Shift the result and contents of M one place right during the operation

↓

6 FINISH

Thus the operation before finish becomes a subtraction of M.D. if the current L.S. digit of M is 1. This corresponds to a negative M since at this stage the sign digit D0 of M has been moved into the L.S. digit position.

Since a subtraction of M.D. from P.S.2 and C.2. involves the redundant representation of borrows rather than carries, it is best accomplished by turning the subtraction into an effective addition by inverting M.D. and adding, the representation of borrow digits being impossible, implying the storage of -1 in any digit position of the carry register. If a subtraction is carried out by adding M.D. (inverse of M.D.) the L.S. digit of the result must be modified to account for the loss of a 1 when M.D. is inverted since :-

$$\overline{M.D.} = -M.D. - 1$$

A simple example is shown to illustrate the point.

Example 8

```
  0 1 1 1
- 0 1 1 0   M.D.
  0 0 0 1
```

is equivalent to :-

```
  0 1 1 1   ____
+ 1 0 0 1   M.D.
+       1   ONE INSERTED
  0 0 0 1
```

Spill digit not stored.

The logical conditions for subtraction during multiplication are derived in Appendix 2, where it is shown that a simple change of rules can be made in order to insert the extra 1 in the L.S. digit position.

An example of 6 bit multiplication involving a negative

M is given :-

Example 9 1 1 0 1 0 1 x 1 1 0 0 1 1 (-11x-13)

Initially

	M.D. 1 1 0 1 0 1	M 1 1 0 0 1 1
PS1	0 0 0 0 0 0	
C1	0 0 0 0 0 0	
Add M.D., P.S.1 & C.1.	1 1 0 1 0 1	
P.S.2	1 1 1 0 1 0 →	1(1 1 0 0 1)
C.2	0 0 0 0 0 0	
Add M.D., P.S.2 & C.2.	1 1 0 1 0 1	
P.S.1	0 0 0 1 1 1	1 1(1 1 0 0)
C.1	1 1 0 0 0 0	
Add P.S.1 & C.1		
P.S.2.	1 1 1 0 1 1	1 1 1(1 1 0)
C.2.	0 0 0 0 0 0	
Add P.S.2 & C.2		
P.S.1.	1 1 1 1 0 1	1 1 1 1(1 1)
C.1	0 0 0 0 0 0	
Add M.D., P.S.1 & C.1	1 1 0 1 0 1	
P.S.2.	0 0 0 1 0 0	0 1 1 1 1(1)
C.2.	1 1 0 1 0 1	
Add M.D., P.S.2 & C.2.	0 0 1 0 1 0	
Insert+ 1	1	
P.S.1	1 1 1 1 0 1	0 0 1 1 1 1
C.1.	1 0 1	
	0 0 0 0 1 0	

True Product 0 0 0 0 1 0 0 0 1 1 1 1

4.5. The M registers

Although this section properly belongs to PART B of this report, it is considered convenient at this stage to show how the digits of M may be examined one-at-a-time, and how digits of P can be stored in place of M, by shifting M to the right.

The system used in KDF 9 to carry this out is shown on FIG SHEET 1. The system uses the basic machine pulses P1, P1a, P2 and P2a which were briefly introduced in Section 4.3 Each pulse is 0.25 μSec long and occurs every 1 μSec. The pulses are interlaced as shown on FIG SHEET 1.

Three flip-flop registers M1 (48 bit), M2 (48 bit) and M5 (4 bit) are used to carry out the shifting. In order to show the transfer and shift paths more clearly the registers are drawn five times on FIG SHEET 1. Register M5 is used to store the digits of P which are shifted down from the adders after each shift or add/shift operation. Digits D44 to D47 of M1 are examined at the appropriate pulse time in order to determine the next operation. It will be seen that this is equivalent to examining only the L.S. digit of M on the simplified shifting scheme described in previous sections of the report. A flow diagram of the events which take place between the M registers is shown on FIG SHEET 2.

The multiplication is started at P2 time by the

appropriate A.C. sequence. Following an initial set-up microsecond which clears PS1, C1 and the counter (see STEP 2 in multiplication procedure Section 4.4), the multiplication lasts for 12 μSec before P is transferred back into the A.U.

It will be seen that the four L.S. digits of M1 are examined in turn to decide the next operation. This separation is possible because of the four basic machine pulses. The timing problem is eased because of the separation, since if only the L.S. digit was examined, M would have to be shifted right every 0.25 μSec, which is too short a time for the flip-flops.

The fairly complicated system of interlaced transfers and shifts has therefore been evolved. Let us examine how one digit D47 of M1 is replaced after examination. It will be seen that D47 is examined at P2 time. At the next P1 time D43 of M1 is shifted to D47 of M2. At P1a time D47 of M2 is transferred to D47 of M1, so that at the next P2 time, D47 of M1 contains the digit previously in D43 of M1. By this time also, D46, D45 and D44 of M1 will have been examined and will be in the process of being replaced by D42, D41 and D40 of M1 respectively. The shifts and transfers are therefore taking place after a digit examination, but are complete before the next examination. In a similar way to the replacement of D47 by D43, D43 is replaced by D39 and so on down the register, every 4th bit being shifted right 4 places.

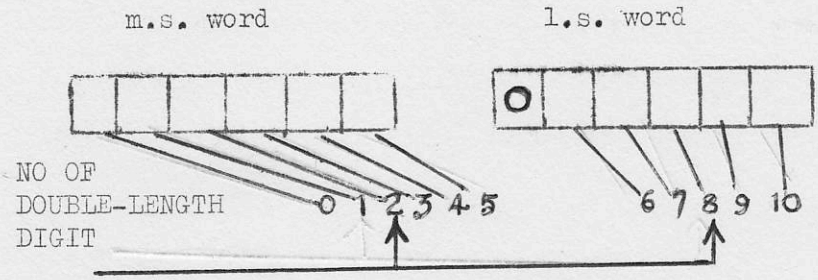
At the first examination of D45 of M1 at the first P1 time, M1 is shifted right into M2, whereas M2 is <u>not</u> transferred to M1, since this would destroy D44 of M1 with the contents of D44 of M2 before D44 has been examined. This transfer is shown on FIG SHEET 1 as FIRST P1 transfer.

By a similar mechanism digits of P which are spilling from the adders as a result of shifting right will be progressively shifted along the M1, M2 registers after they have been stored initially in M5. The replacement of M by P in this way leads to a complication. After 12 μSec all the digits of M have been examined, and the actual multiplication is complete. If the shifts and transfers between the M registers are also stopped at this time, which they are, it means that they are not complete since they lag on the examinations of M1 digits. In order to save time at the end of multiplication, the outstanding shifts and transfers are not carried out, and P, whose digits have all been formed, is recovered from M2 and M5. The outputs of M2 and M5 are selected so that P is assembled in the correct order. The final relationship between M2, M5 and P is shown on FIG SHEET 3.

4.6. <u>Product Representation and Assimilation</u>

The result P of the multiplication, is restricted to a double-length representation, since it must be stored ultimately in the two registers at the top of the N.S., namely W1 and W2. A double-length number in KDF 9 has 95 significant bits. The M.S. bit D0 of the M.S.

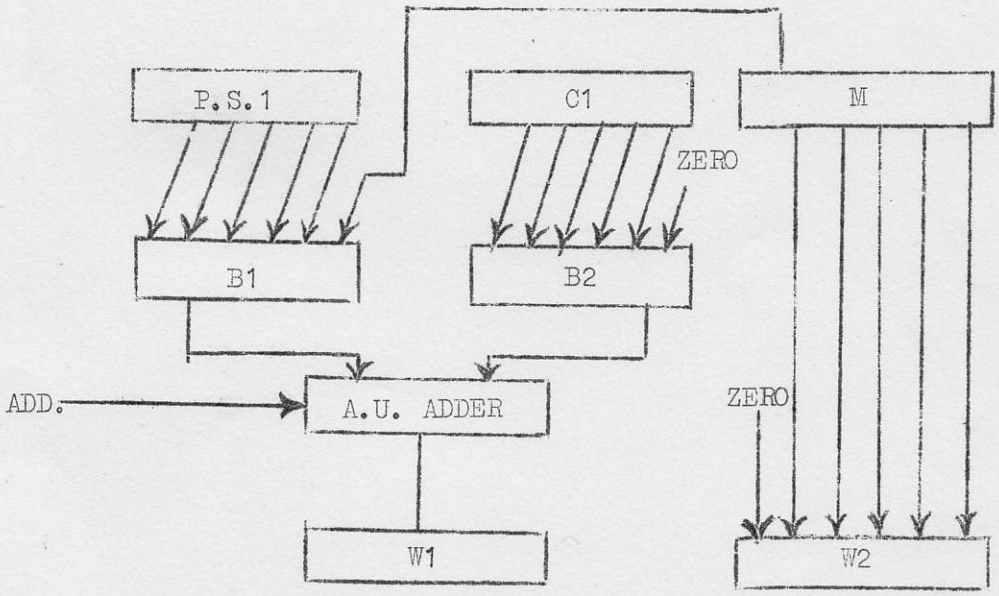
half is the sign digit. The M.S. bit of the L.S. half has no significance and is maintained zero. D1 of the L.S. half and D47 of the M.S. half are assumed to be adjacent bits. Assuming 6 bit numbers for convenience, the representation of a double-length number is :-



The multiplier produces three words of result which are arranged to produce a true double-length P as follows :-

The contents of registers P.S.1 and C.1, which must oe added to produce the M.S. half of P, are placed into registers B1 and B2 respectively, shifted left one place. Registers B1 and B2 are those in the A.U. which are associated with the A.U.Adder. D1 of M5 is placed in D47 of B1, D47 of B2 is set to zero.

An addition of B1 to B2 is initiated by the A.C. sequence. The result, which is the M.S. half of P is placed in W1 Digits of M5 and M2 are placed in the D1 to D47 positions of W2 as explained in Section 4.5 and shown on FIG SHEET 3, D0 of W2 is set to zero. W.2 now contains the L.S. half of P. The block diagram shows the action diagrammatically :-



The transfer of information between P.S.1, C.1, M5/M2 and B1, B2, W2 is controlled by the multiplier control and counting circuits. The addition of B1 and B2 is controlled by the A.C. sequence.

Registers PS1 and C1, which contain a redundant representation of the M.S. half of P, have therefore

to be added in order to form the true M.S. half of P. It can be seen that it is important to maintain the C registers single-length during multiplication, in order that two additions, with a possible carry from L.S. to M.S. halves, are not required to form P from the two sets of redundant numbers which would accrue. This would be time-wasting and would involve more register storage and shifting facilities in the M.D. Unit. The C registers are maintained single-length by adding P.S. to C. when the current M digit is 0, instead of merely shifting P.S. and C one place right.

Throughout the explanation of multiplication, P has been considered as being the result of the multiplication of two integers. Since in the division operation it is usual to treat the operands as binary fractions (explained in Section 5.1), the principle may be extended to multiplication. This in no way alters the basic multiplication procedure already explained. It merely provides a more consistant approach between multiplication and division.

A binary fraction is considered to have its Binary Point (B.P.) between D0 and D1. Thus the operands in multiplication become :-

$$M = D0 \cdot D1, D2, D3, \text{ etc. } D47,$$
$$MD = D0 \cdot D1, D2, D3, \text{ etc. } D47,$$

And the Product P becomes :-

$$P = D0.D1, D2, D3, \text{ etc. } D47, D48, D49, - \text{ etc. } D94$$

It will now be shown, using a fractional notation, that in one case, P cannot be correctly represented. The argument is equally true if P is regarded as being an integer.

4.7. Product Overflow

The maximum and minimum values of P are limited by the double-length representation, to the range :-

$$-1 \leq P \leq 1 - 2^{-94}$$

i.e. the maximum positive value of P is the binary number :-

0 . 1 1 1 1 etc 1 1 1 etc 1 1 1

$2^0 \; 2^{-1}$ $\qquad\qquad\qquad\qquad$ 2^{-94}

and the minimum (most negative) value is the number :-

1 . 0 0 0 0 etc 0 0 0 etc 0 0 0

The maximum value of P will be obtained either :-

a) By multiplying the two maximum positive operands

or

b) By multiplying the two most negative operands. The minimum value of P will be obtained :-

By multiplying the maximum positive by the most negative operand.

se a) The maximum positive single-length operand is $1 - 2^{-47}$
Hence $P = (1 - 2^{-47})^2$
$= 1 - 2^{-46} + 2^{-94}$
and P will be within range
since $P \leq 1 - 2^{-94}$ is satisfied.

se b) The most negative single-length operand is -1
Hence $P = (-1)^2 = +1$
and P will be out of range
since $P \leq 1 - 2^{-94}$ is violated.

Thus this is a failing case. It is the only failing case, since if one of the operands is made slightly more positive, P will be within range. For example if we multiply

$$P = \frac{-1 \times (-1 + 2^{-47})}{1 - 2^{-47}}$$
and $P \leq 1 - 2^{-94}$ is satisfied.

ase c) Maximum positive operand multiplied by most negative operand :-

$P = (1 - 2^{-47}) \times -1$
$= -1 + 2^{-47}$
and $-1 \leq P$ is satisfied.

The failing case -1 x -1, is detected by the A.C. sequence during the addition of PS1 and C1, and the overflow indicator is set. The result of multiplication in the failing case, after transfer to B1 and B2 will be of the form:-

B1 0 . 1 1 1 ---etc --- 1 1 0
B2 0 . 0 0 0 ---etc --- 0 1 0

and the A.U. adder detects that the addition of two positive numbers gives a negative result, thus setting overflow.

Floating Point Multiplications

The mechanism of floating point multiplication has been fully described in Report K/GD.y.83 sections 1.1.3, 1.3.2, 2.2.3 and 3.2 and will not be repeated in this report.

The multiplier behaves in the same way as for fixed point numbers except that :-

a) Digits D1 to D8 of M.D. are copies of D.O. and replace the characteristic of M.D.

b) Digits DO to D7 of M are zero, digit D8 becomes the sign of the fractional part of M.

c) Only 40 digits of M are examined, and the active multiplication time, ignoring the set-up microsecond and terminal addition operation, becomes

The preparation of inputs to the multiplier and the treatment of results are carried out by the appropriate A.C. sequence.

4.9. KDF 9 Multiplier Block Diagram

The block diagram of the multiplier is shown on FIG SHEET 4. It incorporates all the salient points already introduced and serves as a basis for the engineering description in PART B of this report.

5.0. BASIC DIVISION

For any two integers a and b, there exists another pair of integers q and r such that :-

$$a = b.q + r$$

where $0 \leq r < b$

The process of determining q and r, given a and b is known as division. The integers a and b, which are the operands, are usually called the Dividend (D.d) and the Divisor (D.r.) respectively. The results q and r are called the Quotient (Q) and the Remainder (R) respectively. Division is carried out most simply by reducing Dd (in modulus) repeatedly by Dr, until the remaining portion of Dd is so small that a further reduction would change its sign. The number of operations performed gives Q; Q is positive if Dd and Dr have the same sign, Q is negative if Dd and Dr have different signs.

If the signs of Dd and Dr are the same, the process consists of a series of subtractions of Dr.

For example :-

+11 ÷ + 3

+ 11	1st sub
- + 3	
+ 8	2nd sub
- + 3	
+ 5	3rd sub
- + 3	
+ 2	

Q = + 3 R = +2

-11 ÷ -3

- 11	1st sub
- - 3	
- 8	2nd sub
- - 3	
- 5	3rd sub
- - 3	
- 2	

Q = + 3 R = -2

If the signs of Dd and Dr are not the same, the process consists of a series of additions of Dr.

For example:-

+ 11 ÷ -3

+ 11	1st add
+ - 3	
+ 8	2nd add
+ - 3	
+ 5	3rd add
+ - 3	
+ 2	

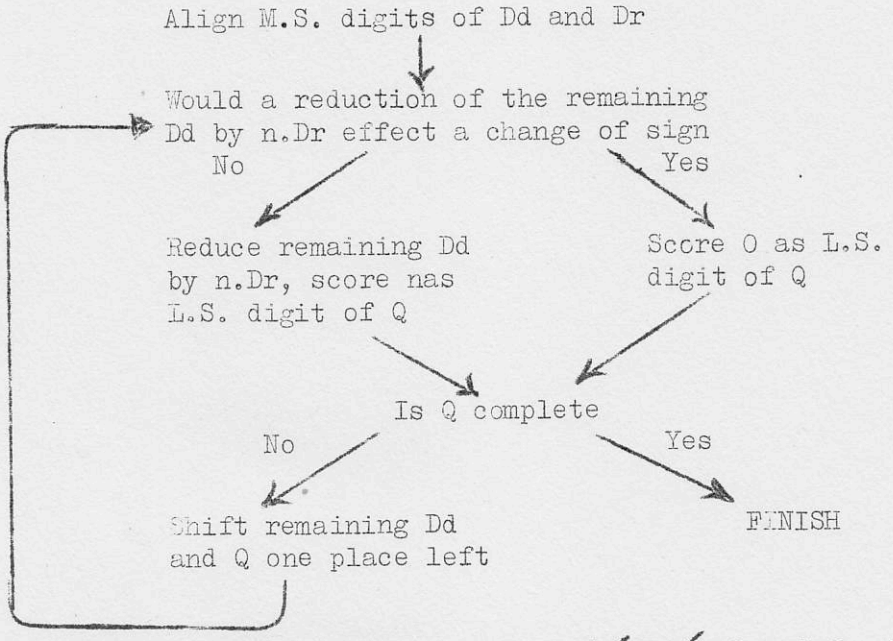
Q = - 3 R = +2

-11 ÷ +3

- 11	1st add
+ + 3	
- 8	2nd add
+ + 3	
- 5	3rd add
+ + 3	
- 2	

Q = - 3 R = - 2

The process becomes very long if Dd >> Dr, since Dr must must be subtracted (added) Q times. It may be shortened by shifting Dd with respect to Dr, and shifting Q as it is formed. This is the normal pencil and paper long division progress, which may be mechanised as follows:-



Note for decimal division $1 \leq n \leq 9$ and for binary division n = 1.

An example of decimal division follows :-

+ 42964 ÷ 39

	4 2 9 6 4	Q = 0 0 0 0
-	3 9	
	3 9 6 4	Q = 0 0 0 1
-	3 9	
	6 4	Q = 0 0 1 1
-	3 9	
	6 4	Q = 0 1 1 0
-	3 9	
	2 5	Q = 1 1 0 1

And Q = 1101 R = 25

The example has been selected so that n = 1 in all successful subtractions.

Binary division is somewhat simplified by the fact that digits of Q can only be 0 or 1. If the above procedure is followed, multiples of Dr will never be required in the reduction operations. For example :-

0 1 1 1 1 0 ÷ 0 0 0 1 0 0 (30 ÷ 4)

Dd =	0 1 1 1 1 0	M.S. digits aligned.
Dr =	0 1 0 0 0 0	
Subtract	0 0 1 1 1 0	Q = 0 0 0 0 0 1
Dr/2	0 0 1 0 0 0	
	0 0 0 1 1 0	Q = 0 0 0 0 1 1
Dr/4	0 0 0 1 0 0	
	0 0 0 0 1 0	Q = 0 0 0 1 1 1

Q = 0 0 0 1 1 1 R = 0 0 0 0 1 0

5.1. Mechanisation of the Division Process

It has been shown that division may be accomplished by a simple recursive process, which is basically the long division process of pencil and paper. Up to this point only division of integers has been considered. Using pencil and paper it is an easy task to keep track of the radix point, when numbers with integral and fractional parts are divided. This is not so easy in the computer, since there is no simple method of storing and keeping track of, the Binary Point (B.P.) The position of the B.P. will vary with the relative values of the operands.

In order to standardise the position of the B.P. in the result Q, it is convenient and usual in a binary computer using two's complement notation, to specify the position of the B.P. in Q *1 The B.P. is assumed to be between D0 and D1 of Q, where D0 is the M.S (sign) digit. In this case the range of Q will be :-

$$-1 \leq Q < +1$$

i.e. 1 . 0 0 0 - - - etc - - - 0 0 0 = - 1
to 0 . 1 1 1 - - - etc 1 1 1 = just < + 1

In order to preserve the position of the B.P. in Q, the initial restriction /Dd/<</Dr/ is imposed so that Q is always fractional *2. This in turn means that the relative magnitudes of /Dd/and/Dr/ must be preserved. The initial operation to align the M.S. digits of Dd and Dr, which is carried out before a pencil and paper division, must not be carried out in the computer, since the latter has no means of remembering the relative shift, and the resulting Q cannot be guaranteed fractional. Keeping these restrictions in mind, there are two basic methods of carrying out division in the computer :-

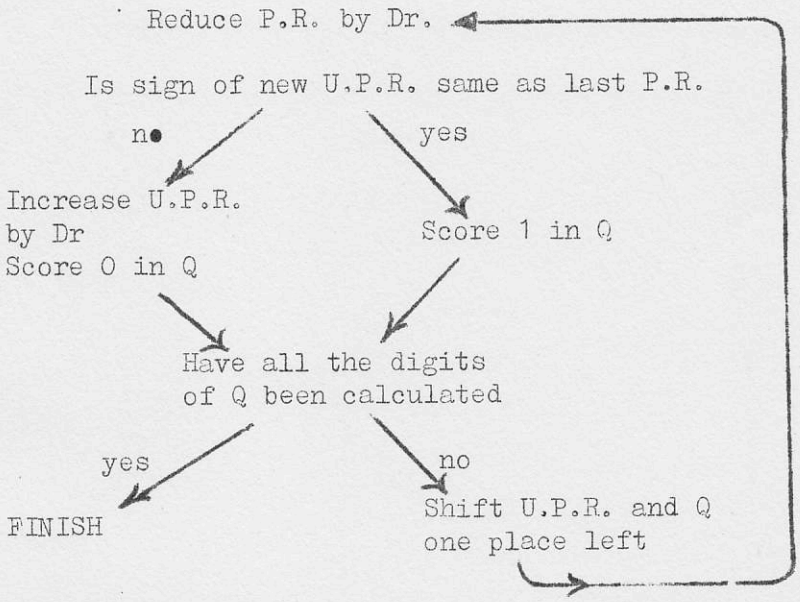
a) Restoring Division

Division using the restoring method involves carrying out trial reductions of the Partial Remainder (P.R.) by Dr. The P.R. is considered to be the remaining portion of Dd after reduction operations have taken place; it is the result of a reduction operation shifted left one place. It is also convenient to introduce the Unshifted Partial Remainder (U.P.R.) at this time. If the sign of U.P.R. changes after a trial reduction, a 0 is scored in Q and the reduction is cancelled by carrying out the opposite operation before proceeding. Thus a subtraction

*1 An alternative approach is given by the concept of "Integral Places". See the KDF 9 Programmer's Manual for a complete explanation.

*2 The condition /Dd/<</Dr/ is normally a programmer as well as a hardware restriction. The KDF 9 instruction ÷ I is an exception in which the A.C. sequence takes care of the relative magnitudes of /Dd/and/Dr by scaling Dd.

of Dr which makes the signs of the previous P.R. and the new U.P.R. different, i.e. is non-successful, is reversed by carrying out an addition of Dr to the new U.P.R. before proceeding. The flow diagram of binary restoring division becomes :-



The reduction operation implies a subtraction if the signs of U.P.R. and Dr are the same, or an addition if they are different. Since it is impossible to inspect the sign of U.P.R. and at the same time possibly alter it by the next operation, it is more convenient to inspect P.R. in order to determine the next step. But U.P.R. is shifted left to form P.R. and unless an extra storage position is provided at the M.S. end of P.R., the sign digit will be lost. The extra storage position is known as the sign Guard Digit (G.D.), and hence the next operation is determined by examining the G.D. and the sign digit of Dr. In Section 5.3 the relative positions of U.P.R. and P.R., and the registers which are used to store them are more fully explained. The first operation in division is a reduction of Dd by Dr and in this respect Dd may be considered to be the first P.R. The sign of Dd is initially copied into the G.D. position of the P.R. register so that the first operation may be treated in the same way as the rest.

Restoring division is not an efficient way of calculating Q, because non-successful reductions and the consequent correction operations, are time-wasting. A procedure known as non-restoring division allows every reduction to contribute to the formation of Q.

b) Non-Restoring Division

In this method, non-successful reductions of P.R. are allowed to take place and are corrected by the next or later operations. Suppose at some stage the G.D. of P.R. and sign of Dr are the same and a subtraction of Dr is carried out. A 1 is scored in Q before Q is shifted, and U.P.R. is shifted left to form P.R. The G.D. of P.R.

is now examined once more and if it is the same as the sign of Dr, the previous operation was valid. If on the otherhand it is not the same, an addition of Dr to P.R. is carried out as the next step and a 1 is subtracted from Q.

For example, if at some stage during division (six bits only), P.R. and Dr are represented by :-

P.R. = 0 0 0 0 0 1 1
Dr = 0 0 0 1 0 0

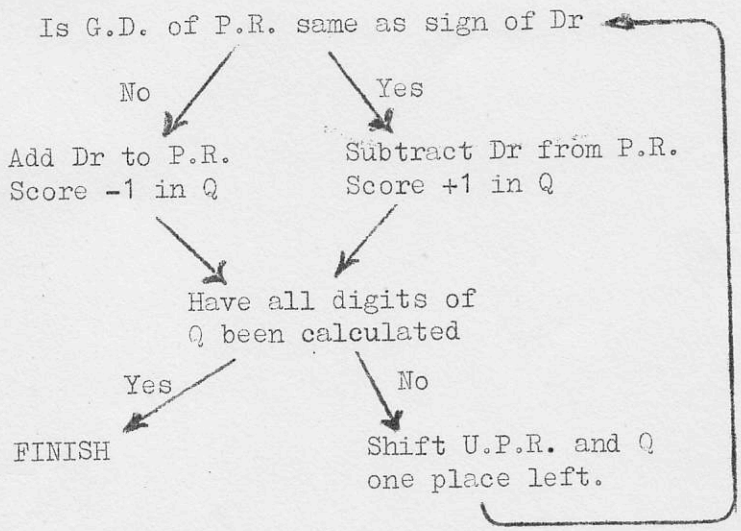
(The G.D. is underlined)

Then restoring division would ultimately score 0 in Q since the trial subtraction of Dr would change the sign of U.P.R. and would therefore be corrected by an addition.

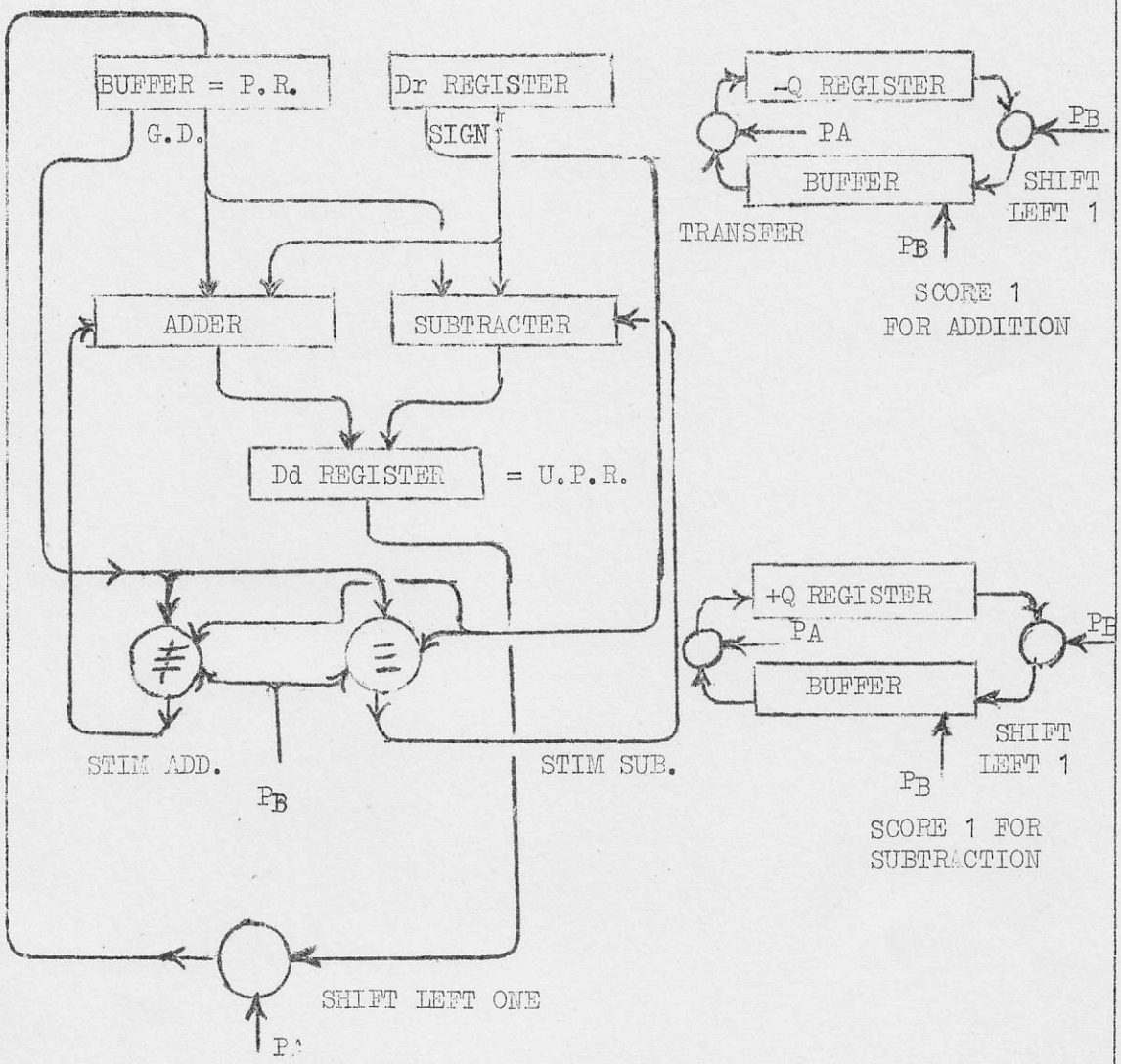
In non-restoring division however, a subtraction of Dr from P.R., scoring +1 in Q, followed by an addition of Dr to P.R. scoring -1 in Q would be carried out as follows :-

P.R.	= 0 0 0 0 0 1 1	
Dr	= 0 0 0 1 0 0	
Subtract	1 1 1 1 1 1	Q = 0 0 0 0 0 +1
Shift	1 1 1 1 1 1 0	Q = 0 0 0 0 +1 0
Dr	0 0 0 1 0 0	
Add	0 0 0 0 1 0	Q = 0 0 0 0 +1 -1
Resultant Q		= 0 0 0 0 0 1

When the negative and positive parts of Q are assimilated the result is the same as for the restoring method. The flow diagram of non-restoring division is :-



A simple block diagram of a non-restoring divider which forms a basis for the method used in KDF 9 is shown :-



In the block diagram operations are carried out at one of two pulse times. These pulses have been called PA and PB and are assumed to be interlaced.

Digits of Q are stored in registers +Q and -Q. When a subtraction takes place a 1 is scored in +Q, when an addition takes place a 1 is scored in -Q. Q is split in this way in order that a full length accumulator will not be required to add or subtract 1's. -1's could be scored as 0's if a single register was used, and a final recovery process would be required to determine Q. It is more convenient however to score -1's in a separate -Q register, since it will be shown in following sections of the report that 0's as well as +1's and -1's must be scored.

At the end of division the contents of -Q must be subtracted from +Q to obtain Q. This operation is carried out in the A.U. adder and is similar to the formation of the product after multiplication.

Both +Q and -Q are shifted left after each division step. This is accomplished with the aid of buffer registers. The Dd also has a buffer associated with it, for the same purpose.

Because much of the equipment used for division is also used in multiplication, and since the equipment is optimised for multiplication, the divider is further complicated. The major complication is that of the

redundant representation of sum and carry, after an add or subtract operation has taken place.

5.2. Redundant Number Representation in the Divider

It was shown in Section 4.3 how a redundant number representation is used in order to increase the speed of multiplication. Because of cost and space considerations, the KDF 9 divider uses a great deal of the equipment used in multiplication. In particular the circuits used to add/subtract/shift MD, PS2 and C2 (c.f. section 4.3), are also used in division to add or subtract Dr from P.R., or to shift P.R. This introduces two basic complications into division :-

a) The add/subtract/shift operation produces results which are shifted right one place, whereas division requires a left shift of one place after each step. The solution to this problem is described in Section 5.3.

b) The add/subtract/shift operation operates upon and produces separate sum/carry redundant representations and thus the P.R. is in this form.

This could be regarded as a major drawback, since at any stage of division P.R. will consist of two parts PS and C, and it has been shown that in order to determine the next step during division, the GD of P.R. and the sign of Dr must be compared. But the GD of P.R. cannot be accurately determined without assimilating its P.S. and C components in a carry-propagating adder. For example at some intermediate step we may have :-

P.S. = 1 1 1 - - - etc - - - 1 1 0) P.R.
C = 0 0 0 - - - etc - - - 0 1 0)

In which case :-

P.S. appears negative
C appears positive

whereas P.R. is in fact zero.

How are we to determine the correct value and G.D. of P.R. without carrying out a full-scale carry-propagating addition? The answer is, that we do not attempt to obtain the absolute value of P.R. and hence its G.D., we make a calculated guess by modifying the basic division process.

It is shown in Appendix 3 that the correct value of Q may be calculated if only the M.S. 4 digits of the P.S. and C components of U.P.R. are examined in order to determine the operational procedures to be carried out during division. It is also shown that the method leads to a 3 choice system of division whereby +1, -1 or 0 may be scored in Q at any step. The formation of Q from the +Q and -Q components is explained in Sections 5.5 and 5.6.

5.3. Determination of the Next Step

Because the multiplier add/subtract/shift mechanism between PS2, C2 and PS1, C1 is also used during division, the results of any division step are shifted right one place, before they are placed in PS1 and C1. Since it is necessary to shift the results left one place in order to produce P.R., the following steps are taken :-

a) A division operation is carried out on PS2 and C2 and places the results in PS1 and C1 shifted right one place.

b) PS1 and C1 are now shifted left two places into PS2 and C2 respectively.

c) P.R. is now contained in PS2 and C2, and consists of the results of the last operation effectively shifted left one place.

d) The next operation is decided by inspecting the 4 M.S. digits of the component parts of U.P.R., which are now contained in D1 to D4 of PS1 and D1 to D4 of C1 together with the sign digit of Dr, as explained in Appendix 3.

e) The next operation is carried out on PS2 and C2 as in step a).

The next operation is determined by a circuit called the Next Step Logic N.S.L., which assimilates D1 to D4 of PS1 and C1 and initiates an addition or a subtraction of Dr to or from PS2 and C2, or a shift (i.e. an addition) of PS2 and C2 only.

The N.S.L. generates the add/subtract/shift signal in accordance with the rules outlined in Appendix 3, which are restated in terms of PS1 and C1.

a) If the result of adding D1 to D4 of PS1 and D1 to D4 of C1 is 0000 or 1111, shift PS2 and C2 (note shift implies adding PS2 and C2 without involving Dr).

b) If the result of adding D1 to D4 of PS1 and D1 to D4 of C1 is not 0000 or 1111 compare the sign of the result with the sign of Dr (D0).

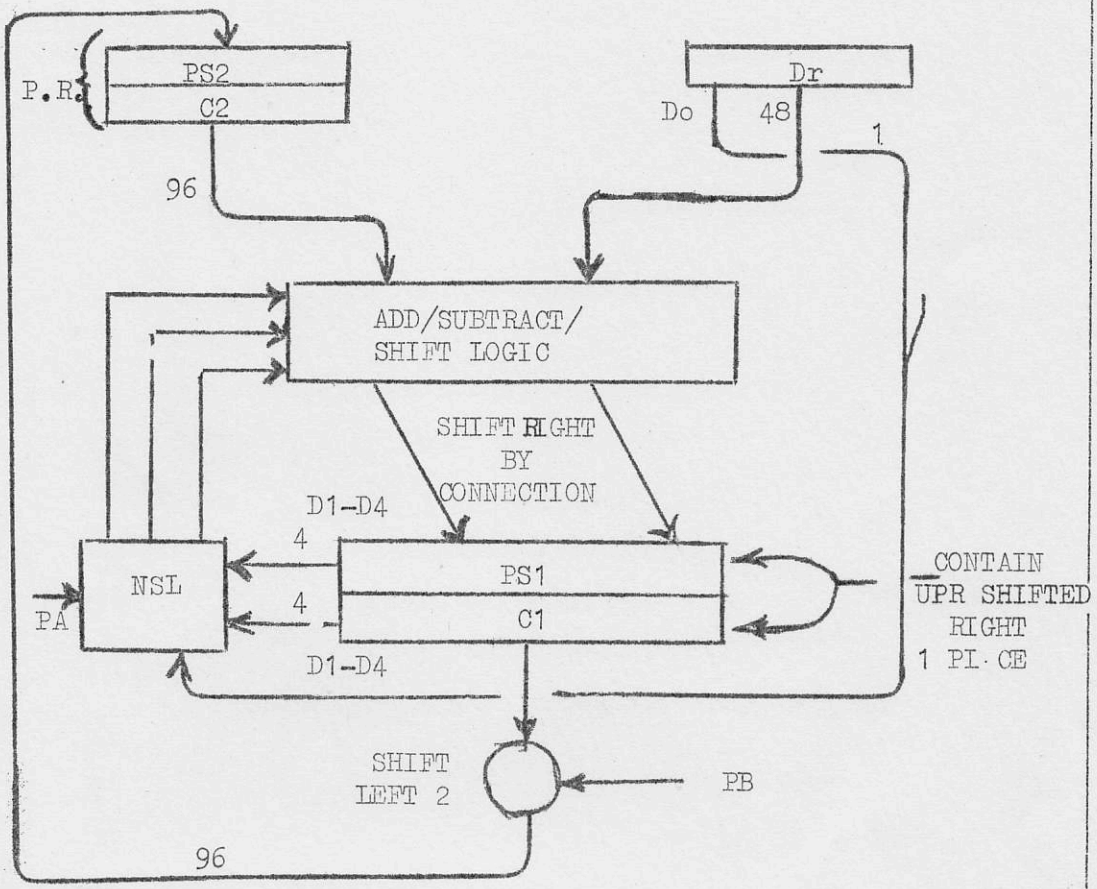
c) If the signs are the same subtract Dr from PS2 and C2.

d) If the signs are different add Dr to PS2 and C2.

e) Score 0 in Q for shift
Score +1 in Q for subtract
Score -1 in Q for add.

The N.S.L. is explained in Appendix 4. It consists of an arrangement of N.O.R. gates (diode OR gates plus D.C. inverters) and is shown in logic form on FIG SHEET 6.

A block diagram of the signal flow in the PS1, C1, PS2 and C2 area is shown :-



effective total shift is 1 place left

Numbers on interconnections indicate the number of bits involved.

Note that in the multiplier the Multiplicand (M.D.) register is connected to the add/subtract/shift logic. In division the M.D. register contains the Dr and hence this register serves a dual purpose, as do the PS1, C1, PS2 and C2 registers.

Since in the simplified diagram shown, PS1 and C1 are providing inputs to the NSL and at the same time are required to accept the results of an operation, the D1 to D4 positions of PS1 and C1 must be duplicated and used alternately to drive two sets of NSL, which are also used alternately. The duplicated portions of PS1 and C1 are known as PS1a and C1a, each of which is a 4 bit register. The two sets of N.S.L. are known as N.S.L.1 and N.S.L.2 and are connected to PS1, C1 and PS1a, C1a respectively.

In division the results of an operation are shifted right one place and placed in PS1 and C1. The operation is similar to that in multiplication, and the digit which spills off the L.S. end of PS1 is stored in register M5. When PS1 is shifted left two places into PS2, the digit stored in M5 is recovered and placed in

D46 of PS2. There is no spill from C1 and no intermediate storage is required. Because of the left shift of two places from C1 to C2, it will be seen that digits D46 and D47 of C2 are always zero after they are initially cleared.

5.4. Initial Operations

As in multiplication the first microsecond of division is reserved for the initial operations of transferring Dd and Dr into the divider and setting the cycle counter to zero.

Since Dd is in non redundant form when transferred to the divider, its C component is set to zero. The initial operations become :-

a) Transfer Dr to register M.D.

b) Transfer Dd to register PS1 shifted right two places, copy the sign digit of Dd into D1 of PS1.

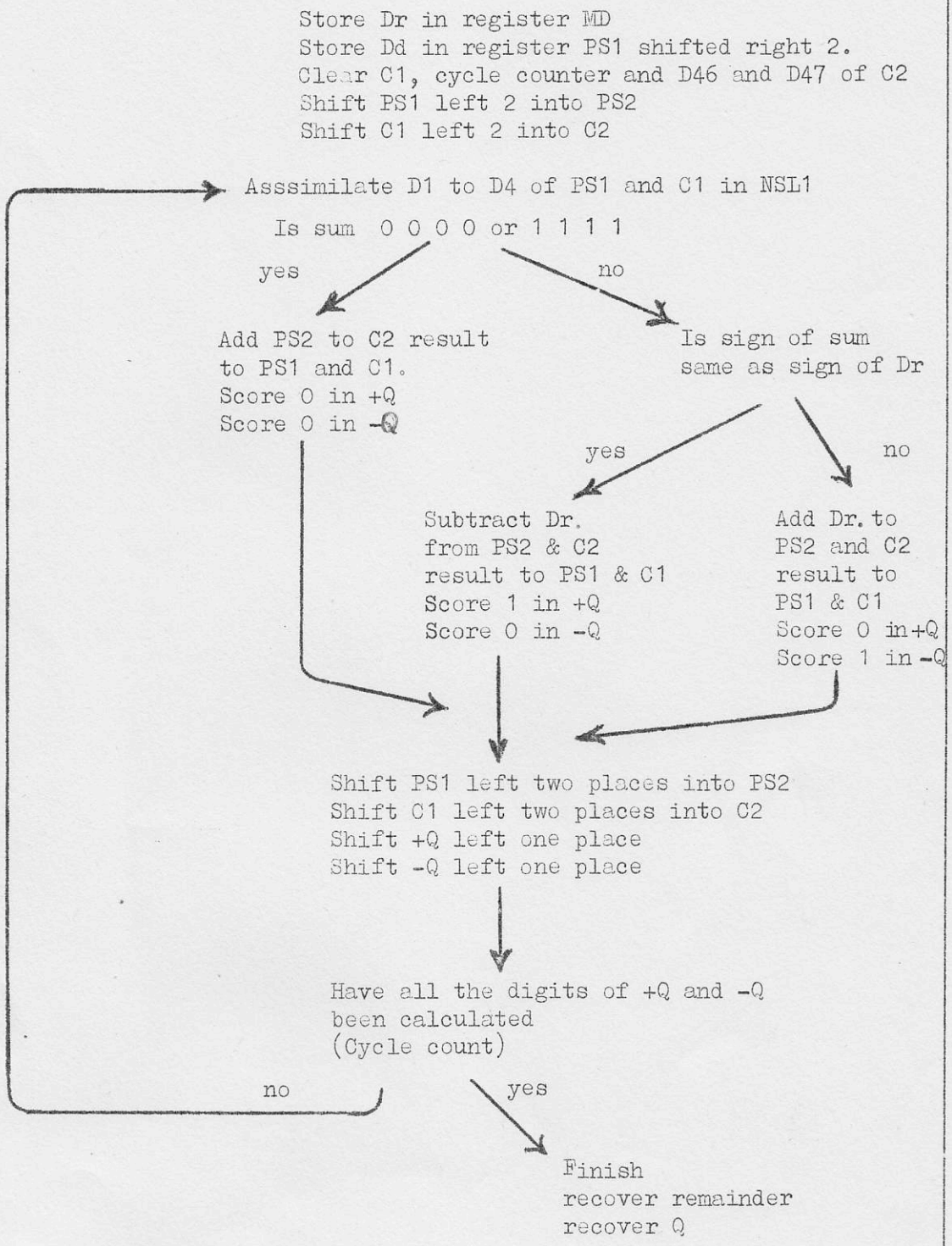
c) Clear C1 and D46 and D47 of C2.

d) Shift PS1, C1 into PS2, C2 respectively, to the left two places. Thus PS2 now contains Dd and C2 is zero.

e) Clear cycle counter.

f) Start division proper by inspecting D1 to D4 of PS1 and C1 via NSL1 to decide the first step, which will be an operation on PS2 and C2. Thus Dd which is contained in PS2 is treated as the first P.R.

The basic flow diagram of division now becomes :-



After 48 add, subtract or shift operations Q will be complete, but will be still in +Q, -Q form. The formation of +Q and -Q is not as simple as has been indicated, it is described in Section 5.5. The Remainder R is in redundant form and is contained in PS1 and C1, its recovery is explained in Section 5.6.

5.5. The formation of +Q and -Q

It has been shown that if 1 is scored in +Q for a subtraction during division, and 1 in -Q for an addition, then the actual Q may be obtained by subtracting the contents of -Q from +Q at the end of division. Note also that the technique of partial assimilation described in Appendix 3 also involves the scoring of 0 in both +Q and -Q if the current operation is a 'shift'.

If this was the case +Q and -Q could be (six bits only):-

+Q	0 1 1 0 1 0
-Q	0 0 1 0 0 1
and Q	0 1 0 0 0 1

It is more useful and just as easy to invert -Q as it is formed, so that at the end of division $\overline{-Q}$ is added to +Q to form Q. This is similar to the effective subtraction of M.D. during multiplication. The subtraction of -Q is replaced by adding $\overline{-Q}$ +1, by a consideration of the equality :-

$$\overline{-Q} = -(-Q) - 1$$

The +1 is added by forcing a carry during the addition of +Q and $\overline{-Q}$, which takes place in the A.U. adder. Thus the example would become :-

+Q	0 1 1 0 1 0
$\overline{-Q}$	1 1 0 1 1 0
+	1
Q	0 1 0 0 0 1

Spill not stored

The new rules for the formation of +Q and $\overline{-Q}$ become:-

a) If Next Step is 'shift' score 0 in +Q and 1 in $\overline{-Q}$.

b) If Next Step is 'subtract' score 1 in + Q and 1 in $\overline{-Q}$.

c) If Next Step is 'add' score 0 in +Q and 0 in $\overline{-Q}$.

Note that if +Q and $\overline{-Q}$ are added at the end of division without inserting a carry then Q-1 will be formed. This is an important effect which is used in division sequences to form the correct value of Q.

+Q and $\overline{-Q}$ are not named as such in the divider. They are formed in registers M1 and M3 respectively during division, and are recovered from these positions at the end of division. The formation of Q digits is shown on the divider schematic FIG SHEET 7. M1 and M3 have buffers associated with them which are M2 and M4 respectively. The buffers are used to shift +Q and $\overline{-Q}$ digits to the left during division. Thus M1 and M2 are also used in both multiplication and division.

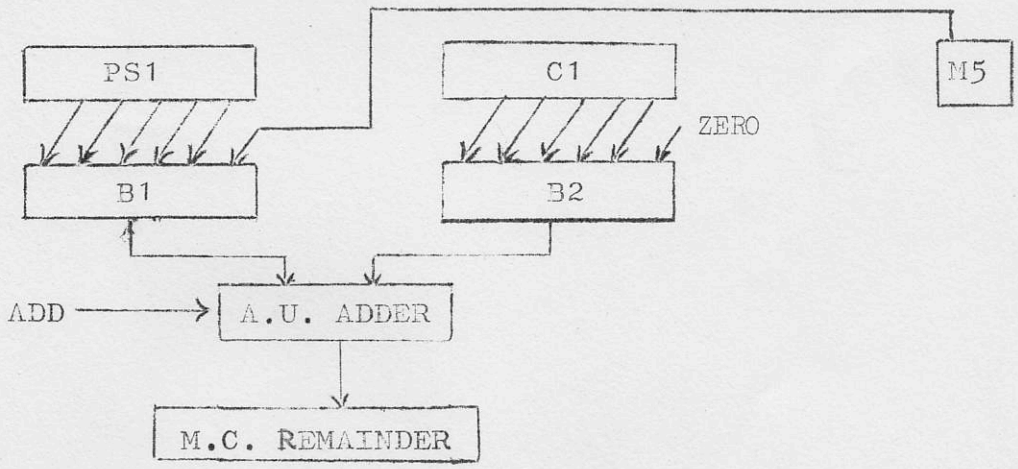
5.6 Recovery of R and Q

After the last step of division the machine remainder is contained in PS1 and C1 in redundant form. Because a division operation shifts the results right one place, the components of the machine remainder must be shifted left one place as they are transferred into the A.U. adder. It was shown in sect. 4.6 that the paths from PS1 and C1 into B1 and B2 in the A.U. also shift the component parts of the Product left one place. Thus, if these paths are also used to transfer the machine remainder after division, the digits of the component parts will appear in the correct position.

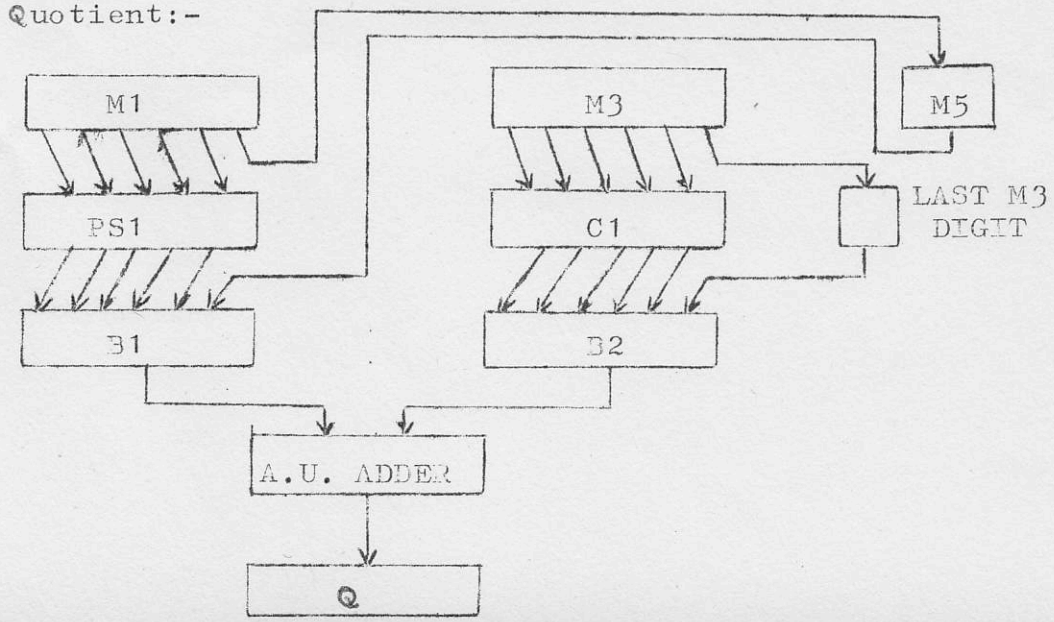
Also the component parts of Q are contained in M3 and M1 at the end of division. Since there are only two paths from the MD Unit into the A.U., from PS1 to B1 and C1 to B2, both of which shift left one place, M3 and M1 are initially placed in C1 and PS1 respectively, both shifted right one place. PS1 and C1 are now placed in B1 and B2 shifted left, and an A.U. addition is initiated

Diagrammatically:-

Remainder:-



Quotient:-



Note that the L.S. digit of the P.S. component of R is contained in $M5_1$ at the end of division, and is placed in B147 when the transfer to the A.U. takes place. This is similar to the transfer of P after multiplication. Also the L.S. digit of M1 is placed in $M5_1$ during the transfer of M1 to PS1, and recovered from $M5_1$ when the transfer to A.U. makes place.

The L.S. digit of M3 is placed in a register stage "LAST M3 DIGIT" during transfer of M3 to C1, and "LAST M3 DIGIT" is transferred to B247 when the transfers to A.U. takes place.

The "LAST M3 DIGIT" is also connected to B247 during the transfer of P after multiplication and R after division. In these cases a transfer of 0 to B247 is ensured by making sure that "LAST M3 DIGIT" is reset ot 0 at the start of every multiplication or division.

The transfer of R to the A.U. is controlled by the divider. The transfer of Q to PS1 and C1 is also controlled by the divider, but the transfer of Q from PS1 and C1 to B1 and B2 is controlled by the A.C. division sequence(s).

5.7 Divider Timing and Speed

Division basically consists of a series of add/subtract/shift operations, each of which is followed by a repositioning of P.R. and a next step decision. Thus each operation during division consists of two steps. It was shown in section 4.2 that each operation during multiplication takes 0.25 usec, and hence 48 operations require a basic 12 usec. In division however, because of the intermediate operation, add/subtract/operations can only take place at the rate of one every 0.5 usec. Thus the basic time for the formation of 48 digits of Q is 48 x 0.5 = 24 usecs.

5.8 Divider block diagram

It is now convenient to refer to the divider block diagram FIG SHEET 7. The registers and common equipment have been shown in the same relative positions on the multiplier and divider block diagrams, in order to show the correspondence.

5.9 Double-length Dividends

Certain division A.C. sequences, ÷D and ÷DF, require that Dd be double-length. It has been shown that Dd is treated as the first PR and is placed in PS2. Normally when a new PR is formed by shifting a result (UPR) left one place, zeros are introduced into the l.s. digit of PS2. If Dd is double length however, digits of the l.s. half of Dd are introduced into the l.s. digit of PS2 as the shift takes place. The l.s. half of Dd is stored initially in register M3, which is also used to store digits of $\overline{Q}$. As M3 is shifted left one place into M4, the current $\overline{Q}$ digit is introduced into the l.s. bit of M4, and the m.s. digit of the l.s. half of Dd is placed in PS247. Thus Dd digits are replaced by $\overline{Q}$ digits and registers M3 and M4 serve a dual purpose.

If Dd is initially single-length M3 is set to zero by the A.C. sequence, so that zeros are introduced into PS247. The sequence for ÷ I also uses the double length Dd facility. This will be more fully explained in part B of this report.

5.10 The Functions of PS1a and C1a

Registers PS1a and C1a, which were briefly introduced in Section 5.3, duplicate the D1 to D4 positions of PS1 and C1. Digits D1 to D4 of the results of any add/subtract/shift operations are placed on alternate occasions in PS1/C1 or PS1a/C1a, in order that the outputs of the NS logic will not change whilst the operation which they currently specify is being carried out. Thus there are two sets of NSL. NSL1 has inputs from D1 to D4 of PS1 and C1, NSL2 from D1 to D4 of PS1a and C1a. The NSL's are used alternately, so that a result produced via NSL1 has its D1 to D4 digits placed in PS1a and C1a, and a result produced via NSL2 has its D1 to D4 digits placed in PS1 and C1. The operation of selecting the destination of the D1 to D4 digits, is automatic and is determined by the P-pulse time at which the operation is carried out. Since the add/subtract/shift operations in multiplication are identical in form to those in division (but are selected by an examination of M digits), PS1a and C1a are also used alternately in multiplication, although the outputs of NSL1 and NSL2 are ignored. This is shown on the multiplier block diagram FIG SHEET 4.

5.11 Division example

An example of division using the KDF 9 method now follows. The Dd is double-length, the l.s. half being stored initially in the $\overline{Q}$ register M3. Six bit numbers and registers are used for the example and thus six operations are required to form Q compared with 48 for the 48 bit, KDF 9 word. Note that the double-length Dd has eleven significant digits, Dc of the l.s. half being ignored.

DIVISION EXAMPLE.

Dr 0.10000 = 2^{-1}

Dd 0.01010 10101 = 341 x 2^{-10}

INITIALLY: $M5_1$ $M4_0$

```
MD   0 1 0 0 0 0
PS1  0 0 0 0 1 0 1 0
C1   0 0 0 0 0 0          NSL1 +ve.
PS2  0 0 1 0 1 0          SUBTRACT
C2   0 0 0 0 0 0
MD   1 0 1 1 1 1
PS1  1 1 0 0 1 0 0        last M3 digit.
C1   0 0 1 0 1 1          NSL2 (-ve)
PS2  0 0 1 0 0 1          ADD
C2   1 0 1 1 0 0
MD   0 1 0 0 0 0
PS1  1 1 1 0 1 0 1
C1   0 0 1 0 0 0          NSL1 (+ve)
PS2  1 0 1 0 1 0          SUBTRACT
C2   1 0 0 0 0 0
MD   1 0 1 1 1 1
PS1  1 1 0 0 1 0 0
C1   1 0 1 0 1 1          NSL2 (-ve)
PS2  0 0 1 0 0 1          ADD
C2   1 0 1 1 0 0
MD   0 1 0 0 0 0
PS1  1 1 1 0 1 0 1
C1   0 0 1 0 0 0          NSL1 (+ve)
PS2  1 0 1 0 1 0          SUBTRACT
C2   1 0 0 0 0 0
MD   1 0 1 1 1 1
PS1  1 1 0 0 1 0 0
C1   1 0 1 0 1 1          NSL2 (-ve)
PS2  0 0 1 0 0 1          ADD
C2   1 0 1 1 0 0
MD   0 1 0 0 0 0
PS1  1 1 1 0 1 0 1
C1   0 0 1 0 0 0          copy of sign digits
```

M3	x10101	−Q
M1	xxxxxx	+Q
M3	101011	−Q
M1	xxxxx1	+Q
M3	010110	−Q
M1	xxxx10	+Q
M3	101101	−Q
M1	xxx101	+Q
M3	011010	−Q
M1	xx1010	+Q
M3	110101	−Q
M1	x10101	+Q
M3	101010	−Q
M1	101010	+Q

R = 0 0 0 0 1 0 1 = PS1 + C1

R = 0.00101

Q = 0.10101 = M3 + M1 + 1

R is the same sign as Dr, and Q is formed by adding M3 and M1 with a carry of 1 introduced.

5.12 Relationship between R and Dr.

It is desirable in the ÷ R and ÷ I sequences, that at the end of division, R and Dr have the same sign and:-

$$0 \leqslant R < Dr$$

The KDF 9 divider may produce a remainder of either sign, which may or may not be the same sign as Dr.

If at the end of division the signs of R and Dr are the same, the two halves of Q are added with the introduction of a carry to form Q, as indicated in Section 5.5.

If the signs of R and Dr are different, an addition of R and Dr is carried out to form a remainder which is less than Dr, and has the same sign. This is always true since:-

$$|R| < |Dr|$$

where R is effectively the last U.P.R. (see Appendix 3).

Besides making the signs of R and Dr the same, this operation also effectively nullifies the last reduction operation of the division, and hence Q-1 is formed by adding the two halves of Q in the A.U. with the introduction of a carry.

Unless the remainder is specifically required (÷R ÷I), it is rejected.

The division instructions ÷F and ÷DF ignore the remainder altogether and do not correct Q. In this respect the Q can have an error of 1 digit at the l.s. end of Q, for floating division only.

The sequences ÷F and ÷DF bring into play a "Remainder Inhibit" flip-flop, which is normally cleared if the remainder is required (e.g. ÷R). The Remainder Inhibit flip-flop prevents the PS1 B1, C1 B2 transfer of R into the A.U.

5.13 Extra Step Division

Two division sequences namely ÷ and ÷D require a rounded Q. Rounding is acheived by effectively determining the 49th Q digit. The 49th digit is not actually produced, but after 48 digits of Q have been determined an extra step of the division procedure is carried out, without altering Q. This step is controlled by an extra step flip-flop, which is set at the start of division and reset by all division sequences other than ÷ and ÷D. before the 48th step. At the end of ÷ and ÷D however, an extra step is carried out which is determined by the NSL, and which forms a new PR but does not change Q.

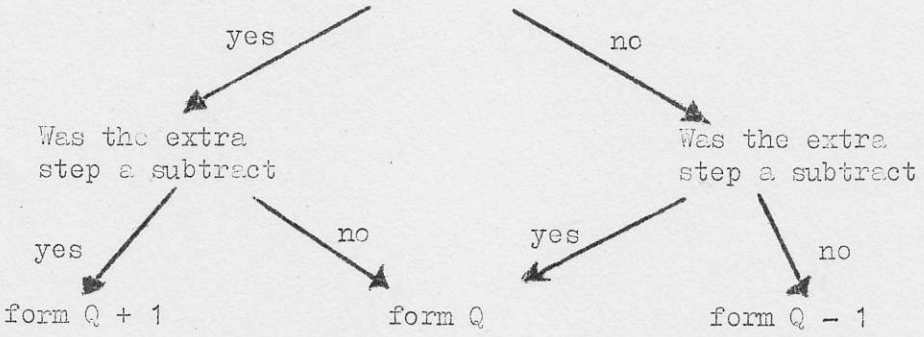
By an examination of:-

a) The signs of R and Dr

b) The nature of the extra step it is possible to determine how Q shall be rounded.

The rules are:-

Are the signs of Dr and the R produced after the extra step the same?



Q - 1 is formed by adding the component parts of Q with a carry introduced
Q is formed by adding with a carry
Q + 1 is formed by adding with a carry to form Q and then adding a further 1.

These operations are carried out by the + and +D sequences.

The operation of extra step in the divider will be explained in PART B of the report.

5.14 Floating Division

For an explanation of the mechanisation of floating point division and exponent determination etc, reference is again made to Report K/G.D.y.83 Sections 1.1.4, 1.3.3, 2.2.4 and 3.3.

The divider behaves in the same way as for fixed point numbers, and in fact the numbers received by the divider are in fixed point form.

The division, unlike multiplication is not shortened for floating point operation.

References

There are several books which explain basic binary multiplication and division very fully. The following are recommended.

1. "Digital Computer and Control Engineering" by J. S. LEDLEY McGraw Hill.
2. "Arithmetic Operations in Digital Computers" by R. K. RICHARDS VAN Nostrand.
3. "Digital Computer Design Fundamentals" by YAOHAN CHU McGraw Hill.

R.H. Allmark.
KDF 9 Development.

J.R.Ellison.
KDF 9 Development.

Enclosures: Appendices 1 - 4.
Figure Sheets 1 - 7.

Appendix 1 Addition

The truth table for the binary addition of PS1, C1 and M.D. digits giving PS2 and C2 is:-

PS1	C1	M.D.	PS2	C2
0	0	0	0	0
1	0	0	1	0
0	0	1	1	0
1	0	1	0	1
0	1	0	1	0
1	1	0	0	1
0	1	1	0	1
1	1	1	1	1

The following Boolean equations may be derived from the table:-

$$PS2 = C1 \neq MD \neq PS1$$
$$C2 = C1\ (MD \neq PS1) + MD.PS1$$

If the digit positions of the registers are numbered 0. 1. 2..... n.....47 from the m.s. end, then allowing for the shift right of one place of the PS and C results:-

$$PS2n = C1_{n-1} \neq M.D_{n-1} \neq PS1_{n-1}$$
$$C2n = C1n\ (MDn \neq PS1n) + MDn.PS1n \quad \text{and}$$
$$\overline{PS2n} = \overline{C1_{n-1}} \neq MD_{n-1} \neq PS1_{n-1}$$
$$\overline{C2n} = \overline{C1n}\ (MDn \neq PS1n) + \overline{MDn}.\overline{PS1n}$$

Similarly when PS2, C2 and MD are added the following equations are obtained:-

$$PS1n = C2_{n-1} \neq MD_{n-1} \neq PS2_{n-1}$$
$$C1n = C2n\ (MDn \neq PS2n) + MDn.PS2n \quad \text{and}$$
$$\overline{PS1n} = \overline{C2_{n-1}} \neq MD_{n-1} \neq PS2_{n-1}$$
$$\overline{C1n} = \overline{C2n}\ (MDn \neq PS2n) + \overline{MDn}.\overline{PS2n}$$

Note that digit 0 of a new PS is the same as digit 1 because of the arithmetic shift right.

If the current M digit is 0, MD is not added. The truth table for the addition of PS1 and C1 giving PS2 and C2 becomes:-

PS1	C1	PS2	C2
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1

giving the equations:-

~~$PS2n = C1n.PS1n$~~. $C2n = C1n.PS1n$

$\overline{PS2n} = \overline{C1_{n-1}} \neq PS1_{n-1}$ $\quad PS2n = C1n-1 \neq PS1n-1$

$\overline{C2n} = \overline{C1n} + \overline{PS1n}$

and:-

$$PS1n = C2_{n-1} \neq PS2_{n-1}$$
$$C1n = C2n.PS2n$$
$$\overline{PS1n} = \overline{C2_{n-1}} \neq PS2_{n-1}$$
$$\overline{C1n} = \overline{C2n} + \overline{PS2n}$$

Appendix 2 Subtraction

A subtraction of MD from PS2 and C2 takes place when the m.s. digit of M is a 1 during Multiplication, i.e. M is negative. It is accomplished by inverting M.D. and adding MD in all digit positions except the l.s. The following truth table refers to the 47 m.s. positions:-

PS2	C2	MD	PS1	C1
0	●	0	1	0
1	)	0	0	1
0	0	1	0	0
1	0	1	1	●
0	1	0	0	1
1	1	0	1	1
0	1	1	1	0
1	1	1	0	1

The following Boolean expressions are derived, allowing for the shift right of one place of the PS and C results:-

$$PS1n = \overline{C2n-1} \neq MDn-1 \neq PS2n-1$$

$$C1n = C2n\ (\overline{MDn \neq PS2n}) + \overline{MDn}.PS2n$$

$$\overline{PS1n} = C2n-1 \neq MDn-1 \neq PS2n-1$$

$$\overline{C1n} = \overline{C2n}\ (\overline{MDn \neq PS2n}) + MDn.\overline{PS2n}$$

Digit 0 of PS1 is the same as digit 1 because of the arithmetic shift right.

The truth table for the l.s. digit of the subtraction is different because the method of subtracting MD is to invert MD and add a further 1 into the l.s. digit of the subtraction. The truth table becomes:-

PS2	C2	MD	PLUS 1	PS1	C1
0	0	0	1	0	1
1	0	0	1	1	1
0	0	1	1	1	0
1	0	1	1	0	1
0	1	0	1	1	1
1	1	0	1	0	2
0	1	1	1	0	1
1	1	1	1	1	1

It can be seen that in one case four 1's must be added. This is shown as a carry of 2, i.e. a carry into the next but one m.s. position. It will be shown that this is an impossible case in practice.

In the failing case the l.s. digit of M.D. is 0, those of PS2 and C2 are both 1. The only way that the l.s. digit of C2 can be 1 is that at least two of the previous aperands must have had 1's in their l.s. digit positions, whether the previous operation was an addition of M.D. to PS and C. (Note the previous operation could not have been a subtraction in multiplication).

Now the l.s. digit of MD is 0 as stated.

Therefore the l.s. digit of both PS and C must have been 1's before the previous operation.

It is also true that the C registers must always have had 1 as their l.s. digit.

This is impossible since C1 is initially set to all 0's at the start of multiplication. The failing case is therefore impossible.

The following Boolean expressions may be derived for the l.s. digit of the subtraction by eliminating the impossible case from the truth table:-

$$PS1n = C2n-1 \neq MDn-1 \neq PS2n-1$$

$$C1n = C2n + PS2n + \overline{MDn}$$

$$\overline{PS1n} = \overline{C2n-1} \neq MDn-1 \neq PS2n-1$$

$$\overline{C1n} = \overline{C2n}.\overline{PS2n}.MDn.$$

Appendix 3 Divider Next Step Determination

This appendix shows that it is possible to determine the nature of the next operation during division, by adding together only the top few digits of PS and C, which together form a pseudo-sum/stored-carry representation of the current unshifted Partial Remainder (U.P.R), shifted right one place. It is also shown that the division algorithm becomes a three-choice method.

Initial Conditions

Initially:-

$|Dd| < |Dr|$ Condition 1

in order that Q be fractional.

Also if Dr is normalised to D1 before starting division:-

$-1 \leq Dr < -\frac{1}{2}$ $(Dr < 0)$ condition 2

$+\frac{1}{2} \leq Dr < +1$ $(Dr > 0)$ condition 3

Intermediate Condition

At any stage of division the relationship:-

$|PR| < 2|Dr|$ Condition 4

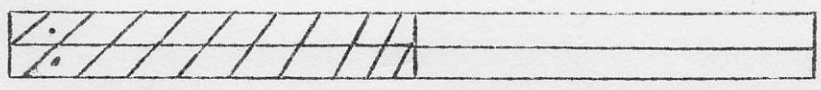
must be satisfied, in order that only one reduction of P.R. by Dr is allowed at any step, since it is only possible to score ± 1 (or 0) in Q. It will be proved that condition 4 is always satisfied, providing that condition 1 is satisfied.

Partial Assimilation

It is more convenient and less confusing in the analysis which follows, to consider the Unshifted Partial Remainder (U.P.R), which is the result of any intermediate division operation before any shifting of that result takes place.

U.P.R consists of two parts PS and C, both of which are 48 bits long. The actual sign of U.P.R must be known in order to determine the next division step. Let us assume that a partial assimilation of PS and C will suffice. Suppose only the n most significant digits of PS and C are added (i.e. the digits 0 to n-1 inclusive):-

digit no. 0 1 2 3 n-1 n n+1 46 47



The result would be U.P.R if n = 48, but if n < 48:-

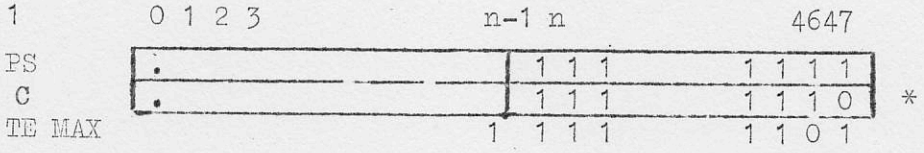
U.P.R = AR + T_E

where AR is the assimilated portion of U.P.R, i.e. sum of digits 0 to n-1,

and T_E is the Truncation Error and is the sum of the digits of PS and C which are not assimilated.

If A.R is the portion of U.P.R which is used to determine the next step, TE may be regarded as an error introduced by the incomplete nature of the assimilation.

The largest error TE MAX is obtained for the following case:-



The worst case is:-

$$TE\ MAX = 2^{-(n-2)} - 2^{-46} - 2^{-47}$$

TE can also be zero but never negative

thus $0 \leqslant TE \leqslant 2^{-(n-2)} - 2^{-46} - 2^{-47}$ condition 5.

Having established the maximum value of TE, we must now determine a value for n, and the steps to be taken during division based on the value of A.R.

Case where the next operation must be a shift

If U.P.R, and hence P.R., is zero at any stage, the next operation must be a shift of P.R. one place left, since if Dr is subtracted or added (depending upon relative signs), and the new U.P.R is shifted left one place:-

$$|\text{New P.R}| = 2|\text{Dr}|$$

and condition 4 is violated.

It is vital that when U.P.R = 0 then AR, which is the sum of digits 0 to n-1 of PS and C, shall indicate a shift of P.R. one place left. (Note shift in this context means adding just PS and C).

There are only two cases of A.R. when U.P.R is possibly zero:-

a) When the digits of A.R. are all zero and TE = 0:-

0 1 2 3 n-1 n 4647

AR = 0.000 0

TE = 000 00

b) When the digits of A.R. are all one's and $TE = 2^{-(n-1)}$:-

0 1 2 3 n-1 n 4647

AR = 1.111 1

TE = 100 00

Although there are only two cases of AR when U.P.R is zero, there are many combinations of values of PS and C which will produce these values of A.R.

Thus if A.R is all zeros or all ones, a shift of P.R. is indicated as the next step. But suppose AR indicated a shift and U.P.R is not zero. An error will be introduced by inspecting only AR, because TE may not be zero or $2^{-(n-1)}$. What is the effect of just shifting PR in this case? We must ensure that after shifting:-

$$|\text{New P.R.}| < 2|\text{Dr MIN}|$$

where Dr MIN is the minimum value of Dr.

* FOOTNOTE D47 the C component of U.P.R is always zero in division because a carry always refers to the next m.s. digit.

The maximum value of the new PR will be obtained if when U.P.R is inspected A.R. is all zeros and TE is a maximum.* Since if we just shift:-

new PR = 2 x old PR

and old PR = 2 x old UPR

then new PR = 4 x old UPR

and $4|\text{old UPR MAX}| < 2|\text{Dr MIN}|$

Hence $4|\text{TE MAX}| < 2|\text{Dr MIN}|$.

Now $|\text{TE MAX}| = 2^{-(n-2)}-2^{-46}-2^{-47}$

and $|\text{Dr MIN}| = 2^{-1}$ $\quad (\text{Dr} = -\frac{1}{2})$

Hence $4(2^{-(n-2)}-2^{-46}-2^{-47}) < 2(2^{-1})$

This is satisfied if:-

$2 \times 2^{-(n-2)} = 2^{-1}$

and $1-n+2 = -1$

$n = 4$

Thus at least 4 digits of the PS and C components of U.P.R must be assimilated to form AR, if the PR produced by a shift operation is to satisfy condition 4. Since a reduction of PR by Dr does not take place, 0 must be stored in Q.

The range of TE now becomes:-

$0 \leq \text{TE} \leq 2^{-2}-2^{-46}-2^{-47}$ condition 6.

Cases where Dr must be subtracted or added

In normal non-restoring division, the signs of U.P.R and Dr are compared in order to determine the nature of the next operation. It will now be shown that the result of a partial a similation of the PS and C components of U.P.R, i.e. AR, will give a true indication of the nature of the next division operation.

It is required to show that if AR is positive U.P.R is positive, and if AR is negative U.P.R is negative, except when AR is 0000 or 1111, when, as already shown, a shift is indicated.

It has been shown that TE is always positive.

* FOOTNOTE When r is all ones it may be seen that the maximum value of the new P.R. is 2/AR/, since TE is always positive and AR is negative in this case;

i.e. $2|\text{AR}| = 2|-2^{-(n-1)}| = 2^{-(n-2)}$

and hence $2|\text{AR}| < 2|\text{TE MAX}|$.

Hence $2|\text{TE MAX}|$ when A.R. is all zeros, yields the maximum value of the new P.R.

Hence if AR is positive, U.P.R must be positive since:-

$$UPR = AR + TE^*$$

If AR is negative we must prove that UPR is negative. This is true if:-

$$|TE| < |AR|$$

Since UPR = AR + TE

and AR is negative

and TE is always positive.

The worst case occurs when $|AR|$ is a minimum value of AR, when AR is negative, is given by the pattern:-

AR = 1.110 (1.111 means "shift")

Thus AR MIN NEGATIVE = -2^{-2}

and $|AR| = 2^{-2}$

But $|TE\ max| = 2^{-2}{}_{-2}{}^{-46}{}_{-2}{}^{-47}$ (condition 6)

$\therefore\ |TE| < |AR|$ for AR negative

and in all cases a negative AR gives a true indication of the sign of U.P.R.

Thus if AR is not 0000 or 1111, its sign is the same as the sign of U.P.R. The sign of AR may be used in conjunction with the sign of Dr to determine the next division step.

The rules for the next step become:-

a) If AR is 0000 or 1111 shift old PR left one place to form new PR (add PS to C), and score 0 in Q.

b) If AR is not 0000 or 1111 and its sign is the same as that of Dr, subtract Dr from PR, shift the new PR left and score +1 in Q.

c) If AR is not 0000 or 1111 and its sign is different to that of Dr, add Dr to PR, shift the new PR left and score -1 in Q.

$|PR| < 2 \times |Dr|$ is always true

It has been shown that if the top 4 digits of U.P.R are determined and are 0000 or 1111, a shift takes place, and

$$|\text{New PR}| < 2|Dr|$$

It can also be shown that if Dr is subtracted or added, the relationship is still true.

* FOOTNOTE In order that this be true U.P.R must be $< +1$, since if U.P.R $\geq +1$ the two's complement representation will make U.P.R appear negative, because a spill will occur into Do. But, since it is always true that:-

$$|UPR| < |Dr|$$

and $|Dr\ MAX| = 1$

then $|UPR| < +1$

and hence UPR MAX POSITIVE $< +1$

also since UPR = AR + TE

then AR + TE $< +1$

and a case such as:-

AR = 0.111

TE = 0.001

UPR = 1.000 is impossible.

If AR (and hence UPR) and Dr have the same sign, Dr is subtracted from PR and the result is shifted left (multiplied by 2) to form the new PR.
In this case:-

$$|PR - Dr| = |PR| - |Dr| \quad \text{(same signs)}$$

If AR and Dr have opposite signs, Dr is added to PR and the result is shifted.
In this case:-

$$|PR + Dr| = |PR| - |Dr| \quad \text{(opposite signs)}$$

Hence in both cases:-

$$|\text{New PR}| = 2\,||PR| - |Dr||$$

Now $|PR| \neq 0$ or else a shift would have been indicated, and assuming:-

$$|PR| < 2|Dr|$$

before the add or subtract then:-

$$|\text{New PR}| < 2|Dr|$$

which satisfies condition 4.

Thus if the PR after the first division step is $< 2|Dr|$, all successive P.R.'s will be within range.

Now initially, from condition 1, if:-

$$|Dd| < |Dr|$$

and Dd is the first PR, then it is certainly true that:-

$$|\text{First PR}| < 2|Dr|$$

and the PR after the first operation will be within range.

In order that $|PR| < 2|Dr|$, it is necessary that Dr is normalised before the actual division, so that the partial assimilation which is carried out will indicate the next step correctly. Dr is normalised by the A.C. divide sequence(s), and in order to maintain the correct relationship Dd is shifted left by the same number of places. Normalisation involves shifting Dr to the left until D0 and D1 are different, so that:-

$-1 \leq Dr < -\frac{1}{2}$ Dr negative

$+\frac{1}{2} \leq Dr < +1$ Dr positive

Appendix 4

Divider Next Step Logic

In order to determine the next division step it is necessary to assimilate the 4 m.s. digits of the PS and C components of U.P.R. The result of this assimilation is A.R. as described in appendix 3. The circuit which forms AR is called the Next Step Logic NSL. The NSL consists of a tree of NOR circuits (diode OR gates plus DC gate inverters), which has 3 outputs, one of which determines the next step according to the rules:-

a) Shift PR one place left if AR is 0000 or 1111.

b) Subtract Dr from PR and shift one place left if AR is not 0000 or 1111 and the sign of AR is the same as the sign of Dr.

c) Add Dr to PR and shift one place left if AR is not 0000 or 1111 and the sign of AR is different to the sign of Dr.

The Karnaugh map of the required conditions is shown on FIG SHEET 5. It must be noted that the PS and C components which are assimilated, lie in D1 to D4 of PS1 and D1 to D4 of C1. This arises because the result of an operation during division upon PS2 and C2 is shifted right one place into PS1 and C1, and thus the sign or m.s. digits of the PS and C components lie in the D1 position of the registers. This is explained in Section 5.3 of the main report.

The following Boolean equations, representing the appropriate next step conditions, may be derived:-

$$\begin{aligned}\text{SHIFT} = {} & (PS1_1 \neq C1_1)(PS1_2 \neq C1_2)(PS1_3 \neq C1_3)(PS1_4 + C1_4) \\ & + (PS1_1 \neq C1_1)(PS1_2 \neq C1_2).PS1_3.C1_3.\overline{PS1_4}.\overline{C1_4} \\ & + (PS1_1 \neq C1_1).PS1_2.C1_2.\overline{PS1_3}.\overline{C1_3}.\overline{PS1_4}.\overline{C1_4} \\ & + (PS1_1 = C1_1).\overline{PS1_2}.\overline{C1_2}.\overline{PS1_3}.\overline{C1_3}.\overline{PS1_4}.\overline{C1_4}\end{aligned}$$

$$\text{ADD} = \overline{\text{SHIFT}}\left\{\left[C\ IN(PS1_1 = C1_1)+\overline{C\ IN}(PS1_1 \neq C1_1)\right] \neq MDo\right\}$$

$$\text{SUBTRACT} = \overline{\text{SHIFT}}\left\{\left[\overline{C\ IN}(PS1_1 = C1_1)+C\ IN(PS1_1 \neq C1_1)\right] = MDo\right\}$$

where MDo is the sign digit of Dr, Dr being stored in register MD.

Where C IN is a term defining the presence of a carry condition into the m.s. digit position of the addition.

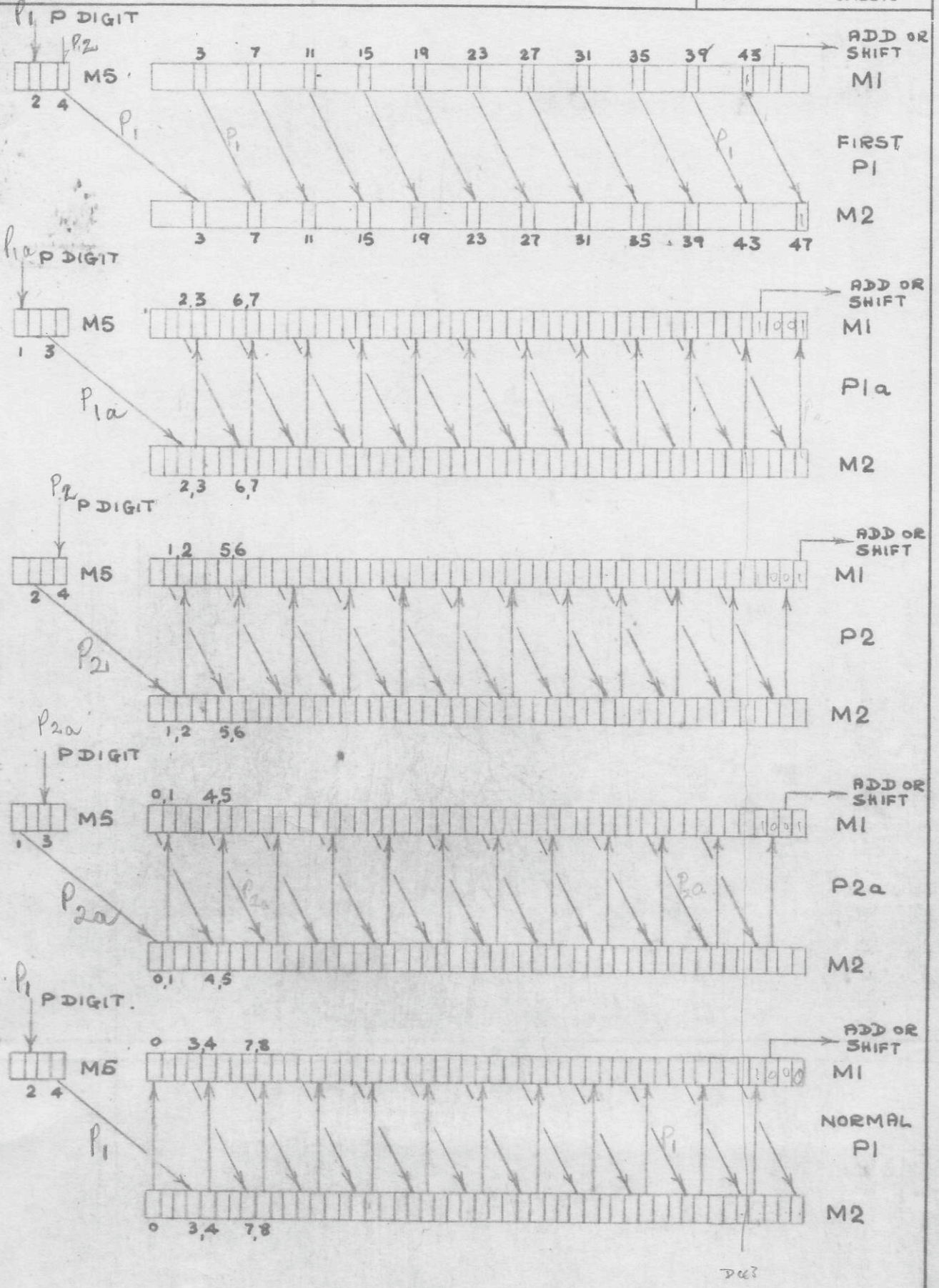
$$C\ IN = \left[PS1_4.C1_4(PS1_3+C1_3)+PS1_3.C1_3\right]\left[PS1_2+C1_2\right]+PS1_2.C1_2$$

$$\overline{C\ IN} = \left[\overline{PS1_4}.\overline{C1_4}(\overline{PS1_3}+\overline{C1_3})+\overline{PS1_3}.\overline{C1_3}\right]\left[\overline{PS1_2}+\overline{C1_2}\right]+\overline{PS1_2}.\overline{C1_2}$$

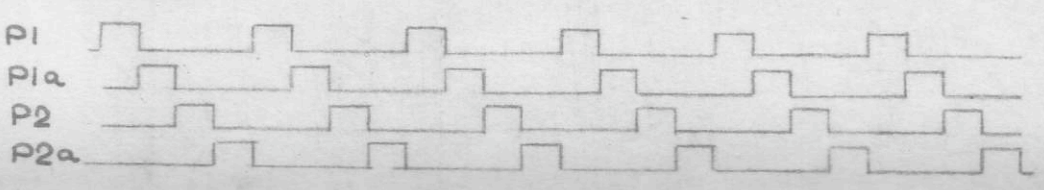
The logical arrangement of gates which forms the NSL is shown on FIG SHEET 6, together with the logical conditions. This arrangement ensures that there are a maximum of 3 D.C. gate circuits in series.

REGISTERS, AND STORAGE OF P DIGITS.

OF SHEETS

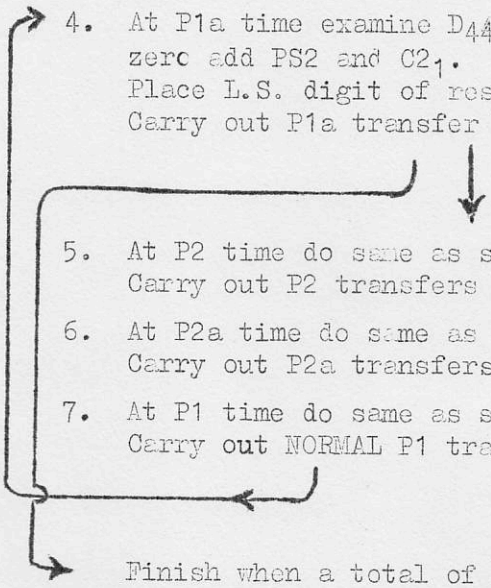


PULSE TIMING



A ORIGINAL ISSUE

1. At P2 time examine D47 of M1, if "one" add PS1, C1 and MD, if zero add PS1 and C1.
   Place L.S. digit of result in $M5_4$ (P digit)
2. At P2a time examine D46 of M1, if "one" add PS2, C2 and MD, if zero add PS2 and C2.
   Place L.S. digit of result in $M5_3$.
3. At P1 time examine D45 of M1, if "one" add PS1, C1 and MD, if zero add PS1 and C1.
   Place L.S. digit of result in $M5_2$
   Carry out FIRST P1 transfer between M1, M5 and M2 (see FIG SHEET 1)
4. At P1a time examine D_{44} of M1, if "one" add PS2, C2 and MD if zero add PS2 and $C2_1$.
   Place L.S. digit of result in $M5_1$.
   Carry out P1a transfer between M1, M2 and M5.



5. At P2 time do same as step 1 and
   Carry out P2 transfers between M1, M2 and M5.
6. At P2a time do same as step 2 and
   Carry out P2a transfers between M1, M2 and M5.
7. At P1 time do same as step 3 except
   Carry out NORMAL P1 transfer between M1, M2 and M5.

Finish when a total of 48 steps have been completed.

FIGURE SHEET 2.

MULTIPLIER L.S. HALF PRODUCT FORMATION

SHEET NO.:- 3
OF SHEETS

DIGIT OF L.S. HALF PRODUCT	TRANSFERRED FROM:	TRANSFERRED TO:
0	M5 1	B1 47
1	M5 2	W2 1
2	M5 3	W2 2
3	M5 4	W2 3
4	M2 0	W2 4
5	M2 1	W2 5
6	M2 6	W2 6
7	M2 7	W2 7
8	M2 4	W2 8
9	M2 5	W2 9
10	M2 10	W2 10
11	M2 11	W2 11
12	M2 8	W2 12
13	M2 9	W2 13
14	M2 14	W2 14
15	M2 15	W2 15
16	M2 12	W2 16
17	M2 13	W2 17
18	M2 18	W2 18
19	M2 19	W2 19
20	M2 16	W2 20
21	M2 17	W2 21
22	M2 22	W2 22
23	M2 23	W2 23
24	M2 20	W2 24
25	M2 21	W2 25
26	M2 26	W2 26
27	M2 27	W2 27
28	M2 24	W2 28
29	M2 25	W2 29
30	M2 30	W2 30
31	M2 31	W2 31
32	M2 28	W2 32
33	M2 29	W2 33
34	M2 34	W2 34
35	M2 35	W2 35
36	M2 32	W2 36
37	M2 33	W2 37
38	M2 38	W2 38
39	M2 39	W2 39
40	M2 36	W2 40
41	M2 37	W2 41
42	M2 42	W2 42
43	M2 43	W2 43
44	M2 40	W2 44
45	M2 41	W2 45
46	M2 46	W2 46
47	M2 47	W2 47

DIVIDER N.S. LOGIC KARNAUGH MAP

MD0	psi1	psi2	psi3	psi4														
					0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	CI_1
					0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	CI_2
					0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	CI_3
					0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	CI_4
0	0	0	0	0	T	S	S	S	S	S	S	S	A	A	T	A	A	A	A	A	
0	0	0	0	1	S	S	S	S	S	A	S	S	A	A	T	T	A	A	A	A	
0	0	0	1	1	S	S	S	S	A	A	A	S	T	T	S	S	A	A	A	A	
0	0	0	1	0	S	S	S	S	A	A	S	S	A	T	S	T	A	A	A	A	
0	0	1	1	0	S	S	A	A	A	A	A	A	S	S	S	S	T	S	T	A	
0	0	1	1	1	S	A	A	A	A	A	A	A	S	S	S	S	S	S	T	T	
0	0	1	0	1	S	S	A	S	A	A	A	A	S	S	S	S	T	T	A	A	
0	0	1	0	0	S	S	S	S	A	A	A	A	T	S	S	S	A	T	A	A	
0	1	1	0	0	A	A	T	A	S	S	S	T	A	A	A	A	S	S	S	S	
0	1	1	0	1	A	A	T	T	S	S	S	S	A	A	A	A	S	A	S	S	
0	1	1	1	1	T	T	S	S	S	S	S	S	A	A	A	A	A	A	A	S	
0	1	1	1	0	A	T	S	T	S	S	S	S	A	A	A	A	A	A	S	S	
0	1	0	1	0	A	A	A	A	T	S	T	A	S	S	A	A	S	S	S	S	
0	1	0	1	1	A	A	A	A	S	S	T	T	S	A	A	A	S	S	S	S	
0	1	0	0	1	A	A	A	A	T	T	A	A	S	S	A	S	S	S	S	S	
0	1	0	0	0	A	A	A	A	A	T	A	A	S	S	S	S	S	S	S	T	
1	1	0	0	0	S	S	S	S	S	T	S	S	A	A	A	A	A	A	A	T	
1	1	0	0	1	S	S	S	S	T	T	S	S	A	A	S	A	A	A	A	A	
1	1	0	1	1	S	S	S	S	A	A	T	T	A	S	S	S	A	A	A	A	
1	1	0	1	0	S	S	S	S	T	A	T	S	A	A	S	S	A	A	A	A	
1	1	1	1	0	S	T	A	T	A	A	A	A	S	S	S	S	S	S	A	A	
1	1	1	1	1	T	T	A	A	A	A	A	A	S	S	S	S	S	S	S	A	
1	1	1	0	1	S	S	T	T	A	A	A	A	S	S	S	S	A	S	A	A	
1	1	1	0	0	S	S	T	S	A	A	A	T	S	S	S	S	A	A	A	A	
1	0	1	0	0	A	A	A	A	S	S	S	S	T	A	A	A	S	T	S	S	
1	0	1	0	1	A	A	S	A	S	S	S	S	A	A	A	A	A	T	T	S	S
1	0	1	1	1	A	S	S	S	S	S	S	S	A	A	A	A	A	A	T	T	
1	0	1	1	0	A	A	S	S	S	S	S	S	A	A	A	A	T	A	T	S	
1	0	0	1	0	A	A	A	A	S	S	A	A	S	T	A	T	S	S	S	S	
1	0	0	1	1	A	A	A	A	S	S	S	A	T	T	A	A	S	S	S	S	
1	0	0	0	1	A	A	A	A	A	S	A	A	S	S	T	T	S	S	S	S	
1	0	0	0	0	T	A	A	A	A	A	A	A	S	S	T	S	S	S	S	S	

$\overline{MD_0}$

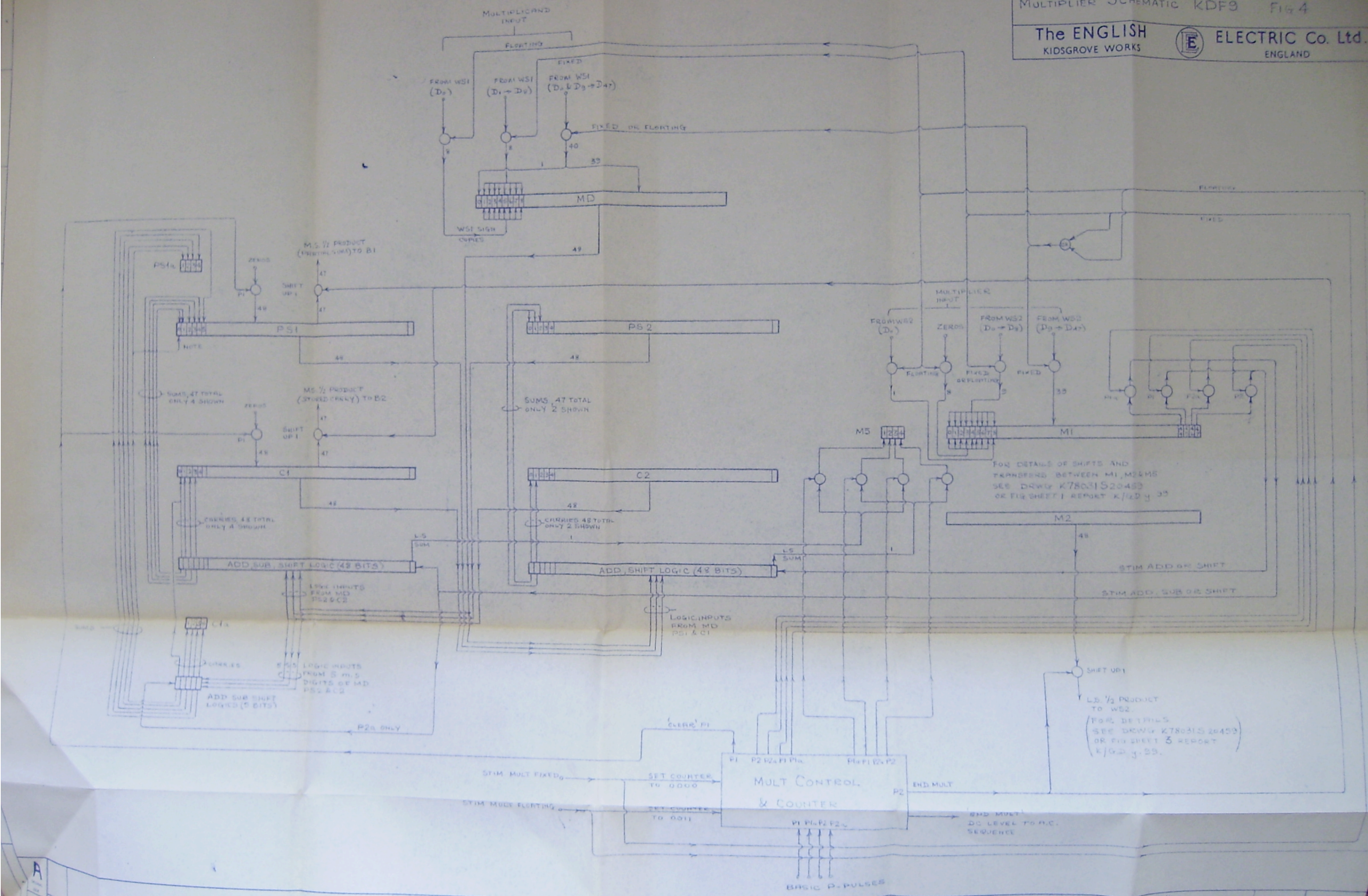
MD_0

S = SUBTRACT
A = ADD
T = SHIFT

THIRD ANGLE PROJ...

MULTIPLIER SCHEMATIC KDF9 Fig 4

The ENGLISH ELECTRIC Co. Ltd.
KIDSGROVE WORKS ENGLAND



CUSTOMER	ORDER NO.	DRAWN	J.R.E.
		TRACED	
		CHECKED	
		APPROVED	