FERRANTI PEGASUS COMPUTER

VOLUME 2

LOGICAL DESIGN

FERRANTI LTD.
Head Office: HOLLINWOOD, LANCASHIRE,
ENGLAND

COMPUTER DEPARTMENT

Offices and Works:

WEST GORTON WORKS
THOMAS STREET
WEST GORTON
MANCHESTER, 12

Tel: EASt 1301

London Computer Centre:

21, PORTLAND PLACE,
LONDON W.1

Tel: LANgham 9211

SUMMARY

This volume, which forms part of the handbook on the Ferranti Pegasus Computer, deals with the logical design of the basic machine. This machine is supplied with a single high-speed tape punch and two photo-electric tape readers; provision has also been made for alternative input and output units to be added if desired. Modifications to the basic design made necessary by customers' special requirements will be dealt with in appendices or extra volumes issued separately. Two chapters, on binary arithmetic and logical operations, have been included for the benefit of readers who are not already conversant with the principles of digital computers.

This volume should be read in conjunction with Volume 2a, 'Logical Design (Diagrams)' and Volume 2b, 'Timing Charts'.

First issue MAY 1956

Printed in England by Specialised Printing Services Limited, 30/34, Langham Street, London, W.1

CONTENTS

ERRATA

Page 47, para. 3, line 7
Delete oblique *after* significant

Page 51.
For para. 17 *read* para. 15

Page 57, para. 31
In the modifier column of the subroutine for 2 *read* 1

Page 73, para. 9, line 5
For paragraph 27 *read* paragraphs 30 to 32

Page 79, para. 31, line 2
For gate *y read* gate *x*

Page 100, para. 6, lines 1 and 3
For 12D2 *read* 12Q1

Page 107, para. 1, line 3
For if *read* of

Page 116, para. 4, line 8
For R00 *read* R04

Page 120, para. 23, line 8
After inhibited *add* by X77
line 9
After storage *add* or a supernumerary drum address have been specified in a 'read' order

Page 129, para. 11, line 2
For sprcoket *read* sprocket

Page 130, para. 14, line 12
For taoe *read* tape

Page 133, para. 23, line 3
Delete bracket after components

Page 137, para. 3, line 1
For fig. 14.1 *read* fig. 13.1

Page 175, para. 2, line 13
For 5 Kc/s *read* 500 c/s

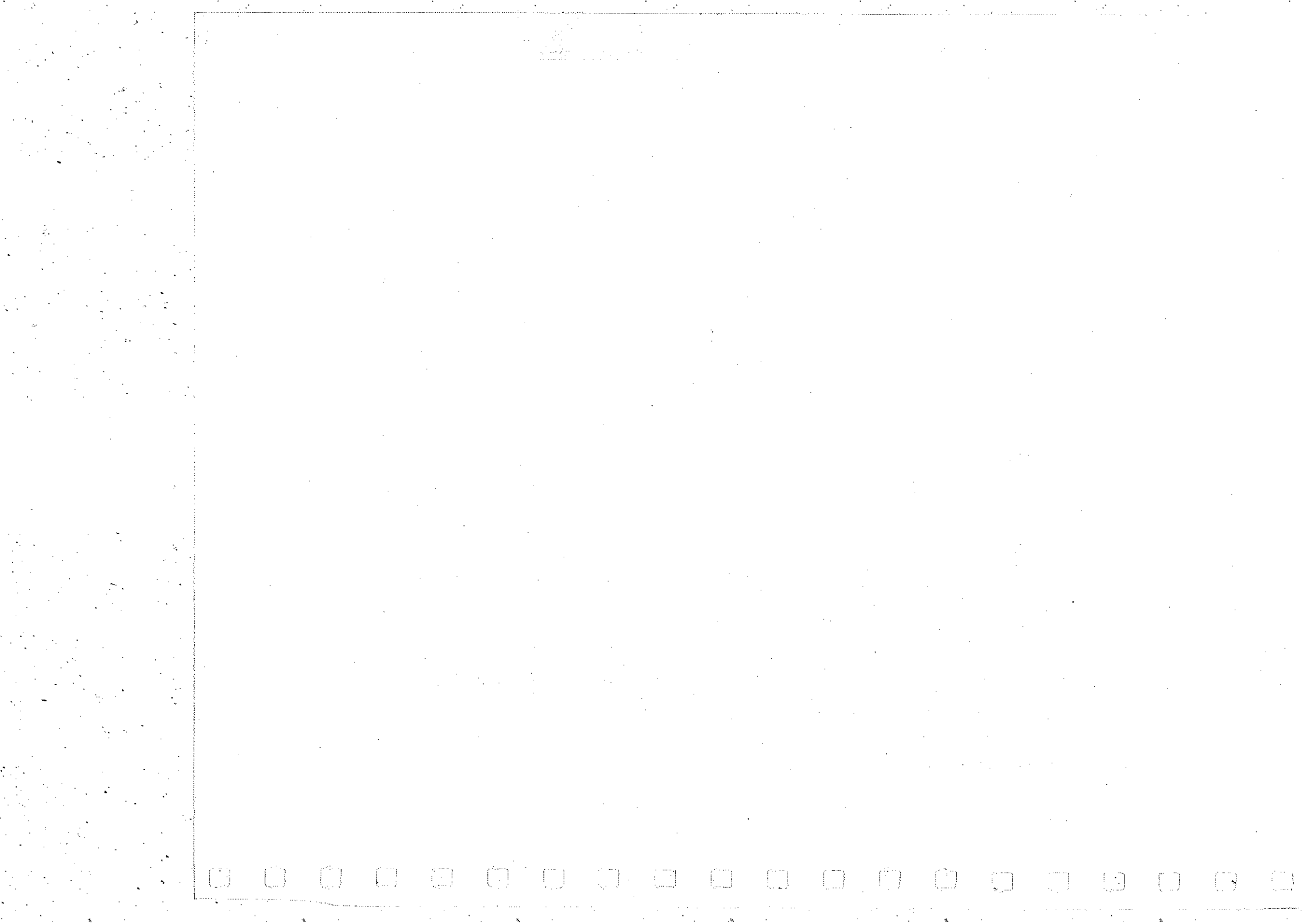
Page 176, para. 4, line 2
For continuously *read* continually

Page 177, para. 10, line 4 and Page 178, para. 14, line 1
For red *read* black

Page 181, para. 21
The function defining AE should be V3 & A & not J & not E

CHAPTER I

BINARY ARITHMETIC



CHAPTER I

BINARY ARITHMETIC

Chapters I and II of this handbook are provided as an introduction to the rest of the work. They may be omitted by readers already conversant with the principles of electronic binary computers.

SYSTEMS OF NUMERATION

Place Value

1. The numeration system used almost universally today is, to give it its full title, a place-value decimal system. It is a place-value system because the value to be ascribed to a figure depends on its position in a series: the two expressions 17 and 71 signify different numbers, although the same two figures are used. It is a decimal, or scale-of-ten, system because the value represented by each place is some power of ten. By convention, the less significant figures, i.e. those with the lower place values, are written on the right, and the more significant figures are written on the left. Thus the number 1728 in the decimal scale must be interpreted as

$$1 \times 10^3 + 7 \times 10^2 + 2 \times 10 + 8 \times 1$$

2. Many numeration systems have been devised without recourse to place-value notation. The ancient Greeks used the letters of their alphabet as numerals; the first nine characters represented the units, the next nine the tens, and the last nine the hundreds. Thus 800 was represented by ω, 40 by μ and 2 by β, so that $\omega\mu\beta$ would represent the number 842. It will be noticed that the order of the symbols ω, μ and β does not matter, whereas changing the order of the figures 8, 4 and 2 in the place-value system changes the number represented. The decimal place-value system originated among the Arabs; because of this, the figures 0 to 9 are sometimes referred to as Arabic numerals, although quite different symbols are used in Arabia today.

3. Intermediate between the Greek and Arabic systems come the Roman and Chinese systems. In Roman numerals, the order of the symbols is sometimes important, e.g. in the distinction between IV (four) and VI (six). As in the Greek system, however, different symbols are used for five, fifty and five hundred (V, L and D). In the Chinese system, the numbers one to nine have separate characters; for numbers greater than nine these characters are given values by special 'value' characters placed below them (Chinese writing is read downwards). The expanded form of the number 1728 given at the end of paragraph 1 is similar to the Chinese method of writing a number if the symbols 10^3, 10^2 etc. are thought of as 'value' characters. The chief advantage of

the place-value system of numbering is that there is no limit to the size of number that can be represented, whereas, without place values, the numbers that can be represented are limited by the number of characters available. Furthermore, the place-value system makes calculation much easier.

Radices

4. The number ten is called the 'radix' of the decimal system. Its choice was quite arbitrary; it seems to have been selected for no better reason than that the majority of human beings, with eight fingers and two thumbs, had 'at hand' a rudimentary decimal digital computer that sufficed for everyday calculations. Nevertheless, other civilisations have used other numeral radices. The Babylonians, for instance, used the scale of sixty, and vestiges of their system remain today in the relationships between seconds, minutes and hours and between the smaller divisions of angle. The method of selling and buying by the dozen or the gross, and the relationship between feet and inches, or shillings and pence, reflect the influence of a number scale with radix twelve, although the numbers 12 and 144 are now always written in the decimal notation. In the scale of twelve, the number 1728 (decimal scale), being the cube of twelve, would appear as 1000. The number 1234 regarded as being in the scale of twelve is equivalent to

$$\begin{aligned} & 1 \times 1728 + 2 \times 144 + 3 \times 12 + 4 \times 1 \\ = \; & 1728 + 288 + 36 + 4 \\ = \; & 2056 \text{ in the decimal scale.} \end{aligned}$$

5. The scale of twelve has much to recommend it as a basis of calculation. Twelve has more divisors than ten, and consequently would give rise to more short cuts to make computation easier. Sixty has more divisors still; but, to make calculations rapidly in the scale of sixty, one would have to know no fewer than fifty-eight multiplication tables, each containing sixty identities. Another disadvantage of the scale of sixty is that, even with a place-value notation, sixty symbols would be required. The number of symbols required (including the zero symbol) in any place-value numeration system is equal to the radix of the system; thus in the so-called 'Arabic' decimal system there are ten symbols including 0.

The Binary Scale

6. In digital computers numbers are represented digit by digit by different states of the computing elements. Thus in a mechanical decimal digital computer, the ten digits possible in each place could be represented by the stable positions of a ten-toothed ratchet. It is, of course, possible to construct an electronic computer to operate in the scale of ten. For instance, the ten digits could be represented by ten different voltage levels at each point in the machine. The difficulty would be in designing circuits that could, notwithstanding circuit tolerances and valve drift, distinguish with absolute certainty between the various levels. The fewer the voltage levels, therefore, the simpler the circuits that would be needed, and the lower the probability of error. In the limit, if the number of voltage levels be reduced to two, an 'all-or-nothing' relationship is established between two states at any point

in the machine, so that ambiguity cannot arise unless there is actually a component failure.

7. With a radix of two, i.e. in the binary scale, only two symbols are required in a place-value representation of a number; the symbols usually used are '0' and '1', and they are represented in the computer by two distinct voltage levels. In the conventional notation, the places represent ascending powers of two, according to their distance from the right-hand end. Thus the binary number

1 0 1 1 0 0 1 1

represents

$1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1 \times 1$

$= 128 + 32 + 16 + 2 + 1$

= 179 in the decimal scale.

Decimal numbers can be converted to binary numbers by successive divisions by two, the remainders being the required binary digits. Thus, as a check on the transformation carried out above,

	remainders
2)179	
2) 89	1
2) 44	1
2) 22	0
2) 11	0
2) 5	1
2) 2	1
2) 1	0
0	1

$\therefore$ 179 (decimal) = 1 0 1 1 0 0 1 1 (binary).

Note that the remainders must be written in order from the bottom upwards to give the conventional binary representation.

8. The use of the binary scale in digital computers has the further advantage that arithmetical operations in this scale can be expressed easily in terms of still more fundamental 'logical' operations. A logical operation can be defined as a choice of one from two or more mutually exclusive contingencies. The series of pulses (binary 'ones') or gaps (binary 'noughts') representing numbers in the machine are fed to crystal-diode gates and to other simple electronic circuits, which produce outputs according to the result of some logical operation on the inputs. The subject of logical operations is dealt with in more detail in Chapter II.

9. Numbers in a computer can be regarded as integers; but it will be more convenient here to regard all binary numbers entering into computation as fractions. The script notation for fractions in the binary scale is analogous to the notation for decimal fractions: a point is placed at the more significant end of the number, and the place values to be ascribed to the digits are successive powers of 1/2. For instance, the number 51/64 is equivalent to

$$1 \times 1/2 + 1 \times 1/4 + 0 \times 1/8 + 0 \times 1/16 + 1 \times 1/32 + 1 \times 1/64$$

and would be written as a binary fraction thus

0 . 1 1 0 0 1 1

ARITHMETICAL PROCESSES

Addition

10. As an introduction to binary addition, consider the summation of the two decimal numbers 65966 and 82935

1 0 1 1 1	carry digits
6 5 9 6 6	number *(a)*
8 2 9 3 5	number *(b)*
1 4 8 9 0 1	sum *(s)*

The two numbers are added digit by digit from the right-hand (less significant) end. Since the sum of the unit digits, 6 + 5, is greater than the radix, 10, the number 1 must be 'carried' into the next column and added in with the two digits already there. In this case, the carry digit makes the sum in the 'tens' column greater than the radix, and another 1 must be carried. It will be noticed that the carry digit cannot be greater than 1 if only two numbers are added. When three numbers are added, the carry digit can be as great as 2, and only when eleven or more numbers are added can the number carried exceed the radix, 10.

11. Binary addition is similar to decimal addition; the process can be illustrated with another example.

1 1 0 0 1	carry digits
0 . 0 1 1 1 0 1	number *(a)*
0 . 0 1 1 0 0 1	number *(b)*
0 . 1 1 0 1 1 0	sum *(s)*

As before, the digits are added column by column from the right. In the first (right-hand) column, the sum of the two digits is equal to the radix, so a 0 is entered in

the sum for that column and a 1 is carried into the next column. In the fifth column from the right, the digits in the two numbers, together with the carry digit, make up one more than the radix, so a 1 is entered and a 1 is carried. It is easily seen that, if three or more binary numbers were added together, the 'carry' would often run into more than one digit. To cater for multi-digit carries would involve quite a lot of extra equipment; it is usual, therefore, in binary computers to add numbers in pairs only. If more than two numbers are to be summed, two are added together, the third is added to the sum of those, the fourth to the resultant sum and so on; this may be done either by using a series of adder circuits or by working repetitively through the same adder.

Subtraction

12. Binary subtraction can be compared similarly with decimal subtraction. Consider, for example, the following decimal subtraction.

	-1	-1	-1	(10)		'borrow' line	
3	7	0	7	4	5	number	*(a)*
2	3	6	7	8	1	number	*(b)*
1	3	3	9	6	4	difference	*(d)*

In the first column, one from five leaves four. In the second column, eight from four would give a negative entry, so a number equal to the radix, must be 'borrowed' from the column immediately to the left. The 'borrow' digit is entered as -1; consequently, the operation in the third column from the right (7 - 1 - 7) requires a further 'borrow' operation. In this example, borrowing must be carried on up to the fifth column from the right, where, for the first time, the digit in *(a)* exceeds the digit in *(b)*.

13. It should be realised that a computer can only operate digit by digit, i.e. it cannot make an inspection several digits ahead to find a possible source for the borrowed digit. 'Borrowing' therefore entails the use of a 'carry' digit that must enter into the computation at every stage until it is cancelled by a particular combination of digits in numbers *(a)* and *(b)*. For example, consider the following subtraction.

		1	1	1	1	(2)	carry digit	
0	.	1	1	0	1	0	0	number	*(a)*
0	.	0	0	0	1	1	1	number	*(b)*
0	.	1	0	1	1	0	1	difference	*(d)*

A carry digit is generated in the first column because the digit in number *(a)* has a smaller value than the digit in number *(b)*. It is carried on through various combinations of digits in the two numbers until it is cancelled, after four more columns, as soon as the digit in number *(a)* is greater than the digit in number *(b)*. The entry in this column is a 0.

Negative Numbers

14. Consider now what will happen when number *(b)* is greater than number *(a)*, i.e. when the difference is a negative number:

. . . 1 . 1 (2)		carry digit
0 . 0 0 1 1 1	number	*(a)*
0 . 0 1 0 1 0	number	*(b)*
. . . 1 . 1 1 1 0 1	difference	*(d)*

Here there is nothing to cancel the carry digit, which must be imagined to be repeated an infinite number of times to the left of the point. However, sustained repetition of the carry digit adds nothing to the information contained in the number. If it is stipulated that all numbers are to be regarded as fractions, a single digit to the left of the point will be enough to ensure that the number is interpreted as negative. This digit is generally referred to as the 'sign' digit. The subtraction operation shown above is equivalent to

$$7/32 - 10/32 = -3/32$$

The difference, as written, is $1\ 29/32$, which is the complement with respect to 2 of the true negative difference, i.e.

$$1\ 29/32 - 2 = -3/32$$

15. A system in which negative numbers are recorded as their complements with respect to some number outside the operating range of the computer greatly simplifies the processes of addition and subtraction. The alternative system of using a sign digit as a direct indication of sign, for example by expressing the number $-n$ as $1 + n$, would often involve numerical complementation (i.e. the conversion of n to $1-n$) during a computation and would in consequence increase either the computing time or the amount of equipment required. It has been assumed in the above explanation that the numbers in the machine are to be regarded as wholly fractional, implying that negative numbers are to be expressed as their complements with respect to 2. The convention for integers is similar; for example, a 38-digit negative binary integer would be recorded as its complement with respect to 2^{40}. Two important consequences of this method of recording negative numbers are, first, that the binary number

1 . 0 0 0 0 0 0

must be interpreted as -1, there being no permissible way of representing +1; and, second, that zero is to be treated as a positive number since the sign digit is necessarily zero.

Overflow

16. With the sign convention described in the previous paragraph, all fractional numbers must lie in the range

$$-1 \leqslant n < 1.$$

It will often happen, however, that the addition or subtraction of two numbers will cause 'overflow', i.e. produce a number outside the permitted range and in the range

$$-2 \leqslant n < -1$$

or $$1 \leqslant n < 2.$$

To ensure that overflow is detected as such, and is not mistaken for a change of sign, 'carry' must not be suppressed until the third digit place to the left of the point. With this proviso, the addition of $35/64$ and $52/64$ would give

```
  0 0 . 1 0 0 0 1 1
+ 0 0 . 1 1 0 1 0 0
-------------------
  0 1 . 0 1 0 1 1 1
```

The fact that the two digits preceding the point are different is to be understood as showing that the result of the operation is the positive number $1\ 23/64$ and not the negative number $-41/64$. The negative number $-41/64$ would appear instead with two identical digits preceding the point as for instance, in the result of the following subtraction:

```
  0 0 . 0 0 1 0 1 1
- 0 0 . 1 1 0 1 0 0
-------------------
  1 1 . 0 1 0 1 1 1
```

17. Not only must the *result* of an addition or subtraction appear with two digits preceding the point; the operands entering into these operations must also have two significant digits preceding the point. Thus negative numbers that are to be added or subtracted must first be expressed as their complements with respect to 4, which simply means that their sign digits must be repeated. As an example of negative overflow consider the addition of the two negative numbers $-35/64$, represented as $(4 - 35/64)$, and $-52/64$, represented as $(4 - 52/64)$

```
  1 1 . 0 1 1 1 0 1
+ 1 1 . 0 0 1 1 0 0
-------------------
  1 0 . 1 0 1 0 0 1
```

In the sum, the two digits preceding the point are different, indicating overflow. The more significant digit of the two is 1, showing that the sum must be interpreted as a negative number, i.e. not as $2\ 41/64$, its apparent value, but as $2\ 41/64 - 4$, or $-1\ 23/64$.

Multiplication

18. Multiplication by powers of two in the binary scale, like multiplication by powers of ten in the decimal scale, is accomplished by shifting the whole digit series the corresponding number of places to the left. Similarly division by powers of two

can be accomplished by shifting to the right. It will be seen that the method of expressing negative numbers as their complements with respect to 2 allows them to be shifted to the right in the normal way, provided that the sign digit is repeated before each shift. Thus the negative number

1 . 0 1

or -3/4

becomes 1 . 1 1 1 0 1

or -3/32 after three right shifts.

Overflow in left shifts, like overflow after addition or subtraction, can be detected by comparing the digits preceding the point. Thus the number -17/64

1 . 1 0 0 0 0 1

becomes 1 1 0 . 0 0 0 1

or -1/16 after two left shifts, the non-equivalence of the digits preceding the point showing that there has been overflow. Normally the digits to the left of the point are deleted after they have been compared for overflow.

19. Multiplication by numbers other than powers of 2 can be accomplished by successive shifts and additions. Thus multiplication by ten, which is $2^3 + 2$, would require the multiplicand shifted three places to be added to the multiplicand shifted only one place. Thus 5/64 could be multiplied by ten as follows

5/64	0 0 . 0 0 0 1 0 1
5/64 × 8	0 0 . 1 0 1 0 0 0
+ 5/64 × 2	+ 0 0 . 0 0 1 0 1 0
= 5/64 × 10	0 0 . 1 1 0 0 1 0 = 50/64

It will be appreciated that the same result could be obtained by shifting the multiplicand twice, adding to the original multiplicand, and shifting once more, thus

5/64	0 0 . 0 0 0 1 0 1	
shift twice	0 0 . 0 1 0 1 0 0	5/64 × 4
add	0 0 . 0 1 1 0 0 1	5/64 × 5
shift again	0 0 . 1 1 0 0 1 0	5/64 × 10

A logical circuit could be designed to perform the necessary shifts and additions.

20. The process described in the previous paragraph for multiplication by ten is analogous to the 'long multiplication' taught in schools; but it is much simpler, since digits of the multiplier can only have the values 0 or 1, so that there is no necessity for the use of tables. Ten, in the binary scale, is

1 0 1 0

The process of multiplication by ten can therefore be written in full thus:

```
      0 0 . 0 0 0 1 0 1     multiplicand
x 1 0 1 0                   multiplier

    ---------------
      0 0 . 0 0 0 0 0 0
    0 0 0 . 0 0 1 0 1       partial
  0 0 0 0 . 0 0 0           products
0 0 0 0 0 . 1 0 1
    ---------------
      0 0 . 1 1 0 0 1 0     final product
```

As a further example consider the formation of the product of the two fractions 25/64 and 23/64. The null partial products are omitted for brevity in this example.

```
    0 . 0 1 1 0 0 1     multiplicand
x 0 . 0 1 0 1 1 1       multiplier

  ---------------------
            0 1 1 0 0 1
          0 1 1 0 0 1   partial
        0 1 1 0 0 1     products
      0 1 1 0 0 1
  ---------------------
0 . 0 0 1 0 0 0 1 1 1 1 1 1   final product
```

giving the required product 575/4096

Rounding

21. The last example shows that the product of two six-digit binary fractions is a binary fraction twelve digits in length. This result is general; the product of two full-length numbers is a double-length number, which will normally have to be treated in two parts. (The implications of double-length working are discussed in paragraph 30). Alternatively the product can be 'rounded' to a single length. The rounding process used in Pegasus is similar to the 'rounding up' process used in decimal arithmetic, in which the digits to be deleted are made to 'carry' 1 up into the rounded result if the most significant of them is a 5 or greater. Similarly, in rounding a binary number, a 1 is 'carried' up into the rounded result if the most significant of the digits to be deleted is a 1. Thus, in rounding from six figures to three,

0 . 0 1 1 0 1 1 becomes 0 . 0 1 1

and 0 . 0 1 1 1 0 0 becomes 0 . 1 0 0

22. In fact, the rounding process can be replaced by the addition of 1 at the most significant end of the portion of the number to be deleted; . this will automatically produce the 'carry' digit required for rounding. Thus

```
  0 . 0 1 1 0 1 1          0 . 0 1 1 1 0 0
+         1              +         1
-------------------      -------------------
  0 . 0 1 1 ✗ ✗ ✗          0 . 1 0 0 ø ø ø
```

Rounding may occasionally cause overflow; but overflow can never result from rounding after multiplication. If the two operands in multiplication are fractions, the product cannot be numerically greater than the smaller of them; hence rounding cannot produce a number greater than would be produced by adding 1 one place beyond the end of the smaller.

Division

23. The normal method of decimal 'long' division involves at each stage some form of inspection to determine whether the divisor will 'go into' the dividend. The only way in which a computer can make an inspection of this kind is to carry out a subtraction (or addition if divisor and dividend have opposite signs) and to test the residue for sign. To avoid the necessity for reconstituting the dividend after such an operation, a process is used in which the machine makes an addition or subtraction at each stage. Briefly, the process is one in which the numerical value of the residue is made as small as possible by repetitive additions or subtractions of the divisor, each operation being carried out with a significance one place lower than that of its predecessor. The sign of the residue is tested after each addition or subtraction to determine the type of operation for the next step. At the end of the process, the record of the amount of addition or subtraction is the required quotient.

24. Consider, for example, the division of 15/64 by 5/8

```
   divisor        dividend
0 . 1 0 1   )  0 . 0 0 1 1 1 1
             - 0 . 1 0 1
             -------------------
               1 . 1 0 0 1
             +     1 0 1
             -------------------
               1 . 1 1 1 0 1
             +       1 0 1
             -------------------
                     1 0 1
             -       1 0 1
                     -------
                     0 0 0
```

The two operands in this example have the same sign: hence, for the numerical value of the dividend to be decreased, the first operation must be subtraction. The residue

resulting from the first operation is negative, i.e. of opposite sign to the divisor, so that the second operation (carried out one place to the right) must be addition, This again gives a negative residue, prescribing addition as the third operation. The fourth operation is subtraction as the third residue is positive. Altogether the divisor has been subtracted from the dividend in the first and fourth places, and has been added in the second and third places. The net amount that has been *subtracted* is therefore

$$(1 + 0.001) - (0.1 + 0.01)$$

times the divisor. Hence the quotient is

$$\begin{aligned} & 1.001 - 0.11 \\ = \; & 0.011 \\ = \; & 3/8 \end{aligned}$$

25. In general, of course, division operations will not 'come out' exactly. The end procedure will then depend on whether we wish the final residue to have a particular sign (unrounded division) or to be as small as possible, regardless of sign (rounded division). Consider, for example, the division of $13/64$ (0 . 0 0 1 1 0 1) by $5/8$ (0 . 1 0 1). The rounded quotient is obviously $3/8$ (0 . 0 1 1) with a residue of $-2/64$ (1 . 1 1 1 1 1 0). The unrounded quotient, if the residue is to be positive, is $1/4$ (0 . 0 1 0), with a residue of $3/64$ (0 . 0 0 0 0 1 1). The first three steps of the division process are as follows:

0 . 1 0 1)	0 . 0 0 1 1 0 1	
	- 1 0 1	+ 1 . 0
	1 . 1 0 0 1	
	+ 1 0 1	- 0 . 1
	1 . 1 1 1 0 0	
	+ 1 0 1	- 0 . 0 1
	0 . 0 0 0 0 1 1	0 . 0 1
	residue	quotient

It will be seen that the residue and quotient happen to be correct at this stage for unrounded division. It would be uneconomical on equipment, however, to make the machine sense this condition and stop; it must be allowed to finish the last division operation.

0 . 1 0 1)	0 . 0 0 0 0 1 1	0 . 0 1
	- 1 0 1	+ 0 . 0 0 1
	1 . 1 1 1 1 1 0	0 . 0 1 1
	residue	quotient

26. The quotient and residue now happen to be correct for rounded division. However, rounding in general requires one further digit to be examined that will not appear in the rounded quotient, and thus requires the process to be taken one step further. A 'nought' is therefore 'brought down' to form the last digit of the residue, and the next operation - in this case, addition - is carried out

	1.1111100		0.011
+	101	-	0.0001
	0.0000001		0.0101
	residue		quotient

27. Now, owing to the method of forming it, the last digit of the quotient must always be a 'one'. It would change to a 'nought' or would remain the same according as the next operation on the *quotient* were to be subtraction or addition. In this case, as the residue is positive, the next operation on the *residue* would be subtraction, and on the *quotient* addition; so the fourth quotient digit is not going to change. If the quotient is to be rounded to three digits, therefore, its third digit must be made 1, and its fourth digit 0. The simplest way of doing this is to add 1 in the fourth place of the quotient; of course a corresponding operation must be carried out on the residue.

	0.0000001		0.0101
-	101	+	1
	1.1111100		0.011Ø
	final residue		rounded quotient

Note that operations in the last stage take place with the same significance as operations in the penultimate stage. The general rule for *rounded* division is therefore that the process must be taken two stages 'too far', the last two operations being carried out with the same significance.

28. It is convenient to make the process of unrounded division as nearly as possible the same as the process of rounded division. The division is therefore taken only one stage 'too far', and the surplus quotient digit is then removed by a straightforward subtraction, which implies, of course, an addition to the residue.

	0.0000001		0.0101
+	101	-	1
	0.0000110		0.010Ø
	final residue		unrounded quotient

As in rounded division, the last two operations are carried out with the same significance; but in unrounded division, the last operation on the residue is *always* addition, and on the quotient *always* subtraction.

29. In the division examples given, the operands have been regarded as fractions (although they might equally well have been regarded as integers, e.g. as 5 and 15 in the example of paragraph 24). If the dividend is numerically greater than the divisor, the quotient will overflow, i.e. will be greater than unity if the numbers are treated as fractions. The computation should, of course, be organised so that this condition either does not occur or is corrected if it occurs. Notice that, since it is possible for a quotient to be taken out of range by rounding, the overflow condition cannot be tested conclusively until the end of the division process.

Double-length Working

30. To avoid the accumulation of 'rounding' errors, it is sometimes necessary to work with double-length numbers. The sign of a double-length number is indicated by the sign digit of the more significant half; the less significant half is then always positive, and can be regarded as a small correction term that is to be added to the rest of the number. As the two halves of a double-length number must enter into arithmetical operations separately, some facility must be provided whereby a 'carry' operation can take place between them if the less significant half overflows or changes its sign.

31. The 'carry' operation between the two halves of a double-length number is termed 'justification'. In the case of simple positive overflow, the justification process must add 1 in the last place of the more significant half. A change of sign might produce a number such as

1 1 . 1 0 1

which is to be interpreted as

$$3\ 5/8 - 4 = 5/8 - 1$$

To keep the less significant half of the number positive (equal to + 5/8), a 'one' must be subtracted from the last place of the more significant half.

32. The term 'justification' is used in the printing trade to denote the process of adjusting to a uniform length, the lines on a printed page. It is used in a similar sense here. The process outlined in the previous paragraph can be used to form a double-length number from a single-length number that has overflowed, i.e. has exceeded the standard length. For this purpose, the out-of-range number is regarded as the less significant half of a double-length number, the more significant half being wholly zero. There are two possibilities, according as the out-of-range number to be justified is positive or negative. Thus if the number is positive,

0 . 0 0 ..... 0 0 justified with 0 1 . 1 0 ..... 0 1

becomes 0 . 0 0 ..... 0 1, 1 0 ..... 0 1

whereas, if it is negative,

0 . 0 0 ..... 0 0 justified with 1 0 . 1 0 ..... 0 1

becomes 1 . 1 1 ..... 1 0, 1 0 ..... 0 1

It will be noticed that positive overflow involves the addition of 1 in the last place of the more significant half, whereas negative overflow involves the subtraction of 1 in the penultimate place. If the numbers are regarded as wholly fractional, justification becomes equivalent to a right shift of as many places as there are digits in a single-length number.

Floating Point

33. In floating-point arithmetic, a number is treated in two parts, the argument and the exponent. The argument is simply a fraction, positive or negative; the exponent (in binary arithmetic) is a power of two by which the argument is assumed to be multiplied. Thus the argument 0 . 0 1 1 together with the exponent 7, would express the binary number:

$$0 . 0 1 1 \times 2^{7} = 1 1 0 0 0 0$$

or 48 in decimal notation. Multiplication in floating-point arithmetic requires the arguments to be multiplied together and the exponents to be added; thus

$$\begin{aligned} & 0 . 0 1 1 \times 2^{7} \times 0 . 1 0 1 \times 2^{6} \\ & = 0 . 0 1 1 \times 0 . 1 0 1 \times 2^{13} \\ & = 0 . 0 0 1 1 1 1 \times 2^{13} \end{aligned}$$

34. Floating-point addition involves a few more operations, as the two numbers must first be adjusted to have the same exponent. The usual practice is to shift the argument of the number with the smaller exponent, and to adjust this exponent correspondingly, until the two exponents are equal. The arguments can then be added directly. This process involves only right shifts, (i.e. towards the less significant end) and consequently cannot lead to overflow. Thus

$$\begin{aligned} & (0 . 0 1 1 \times 2^{7}) + (0 . 1 0 1 \times 2^{5}) \\ = & (0 . 0 1 1 \times 2^{7}) + (0 . 0 0 1 0 1 \times 2^{7}) \\ = & (0 . 0 1 1 + 0 . 0 0 1 0 1) \times 2^{7} \\ = & \quad 0 . 1 0 0 0 1 \times 2^{7} \end{aligned}$$

35. It is usual to express the arguments of floating-point numbers in a standard form. Ideally this form should ensure that overflow cannot ensue from a single addition or subtraction, which means that the argument should be in the range

$$-1/2 \leqslant n < -1/4$$

or

$$1/4 \leqslant n < 1/2$$

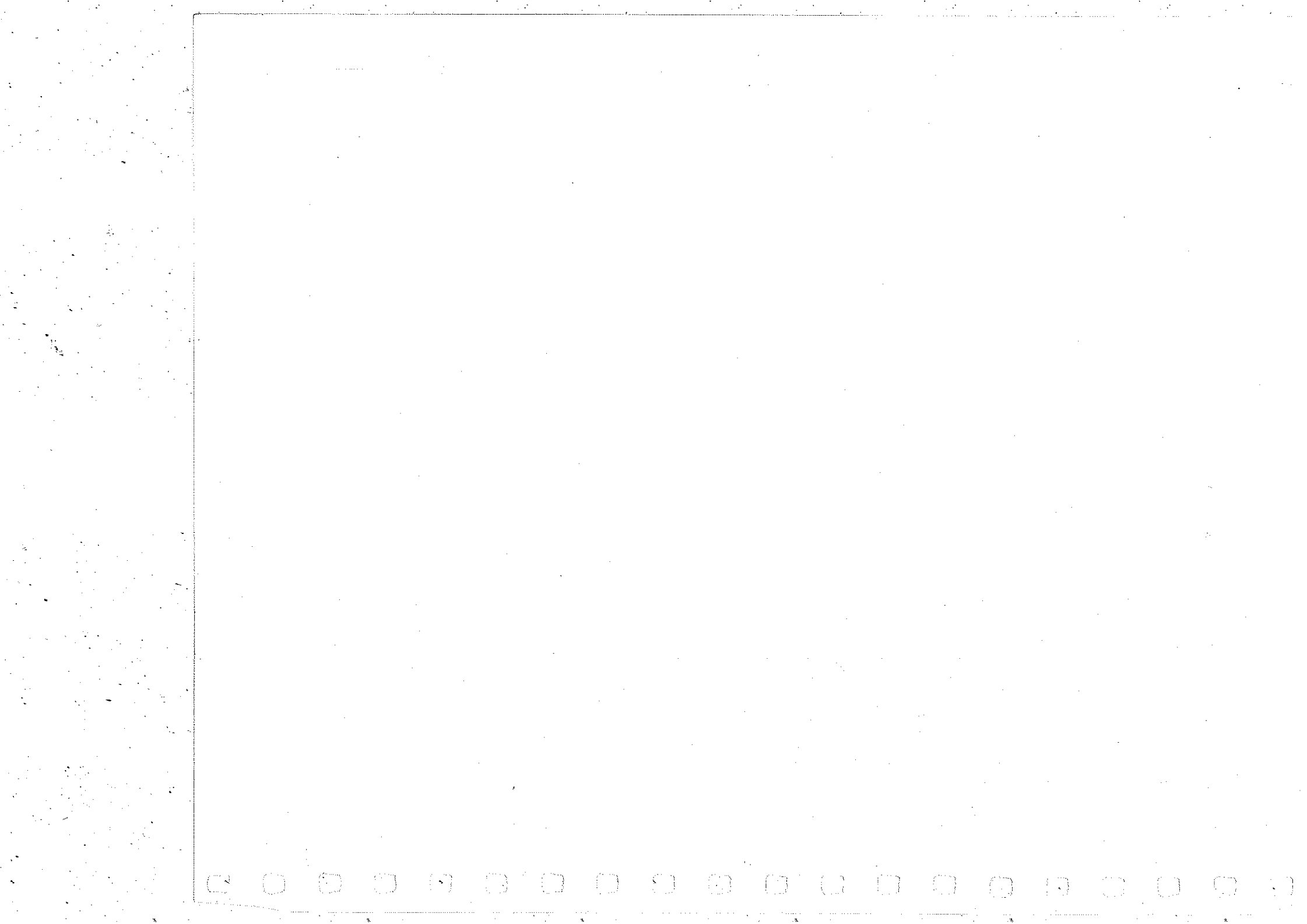
i.e. the first two digits after the point should be different. The process of putting the argument into this form is called 'normalisation'; the argument is shifted to the left the desired number of places, and the number of the shifts is subtracted from the exponent. Thus

$$\begin{aligned} & 0 . 0 0 0 0 0 1 0 1 \times 2^{6} \\ = & 0 . 0 1 0 1 \times 2^{2} \end{aligned}$$

In the case of one right shift - the maximum necessary if there has been no overflow - the exponent is increased by one.

C H A P T E R I I

LOGICAL OPERATIONS



CHAPTER II

LOGICAL OPERATIONS

Chapters I and II of this handbook are provided as an introduction to the rest of the work. They may be omitted by readers already conversant with the principles of electronic binary computers.

PRINCIPLES OF LOGIC

AND Operation

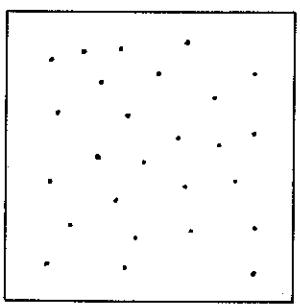
1. Suppose that it is required to select a team of computer-maintenance engineers, and that the only requirement for selection is the possession of the following two qualifications:

(a) the ability to use a soldering iron

(b) the ability to understand logic.

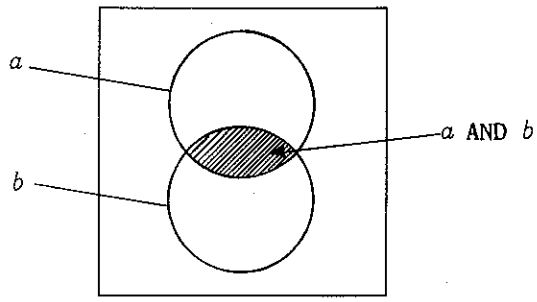
then, out of the set of all applicants for the post of computer-maintenance engineer, those would be selected who possessed qualification *(a)* *and* qualification *(b)*. A selection such as this, in which a smaller class is chosen out of a larger set according to some rule, is called a 'logical operation'. This particular logical operation is called the 'AND operation'.

2. The AND operation can be illustrated diagrammatically as follows. Suppose that all applicants for posts as computer-maintenance engineers are put together in one room like this.



Suppose now that they can be arranged in the room in such a way that one circle can be drawn around all those who possess qualification *(a)*, and another circle can be drawn

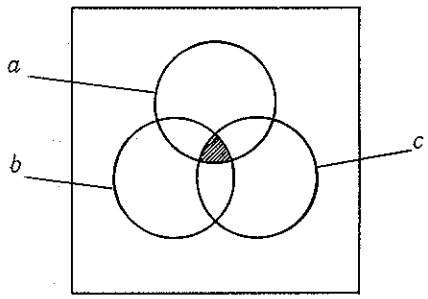
around all who possess qualification *(b)*, thus



The region of 'overlap' of the two circles, the shaded area, then includes all those who possess qualification *(a) and* qualification *(b)*. We shall use the algebraic expression

$$a \;\&\; b$$

to designate a class selected in this way. If we wish, we can apply the AND operation to more than two classes like this.



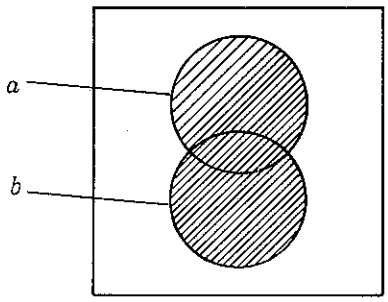
The area shaded would then correspond to the algebraic expression

$$a \;\&\; b \;\&\; c$$

OR Operation

3. Suppose, now that more computer-maintenance engineers are required than the AND operation can select. To obtain the requisite numbers, the requirements might be relaxed so that the successful candidate need have only *one* of the two qualifications - on the assumption that the other can be learnt in time; of course, those who possess *both* qualifications will be selected as well. The class of successful candidates can

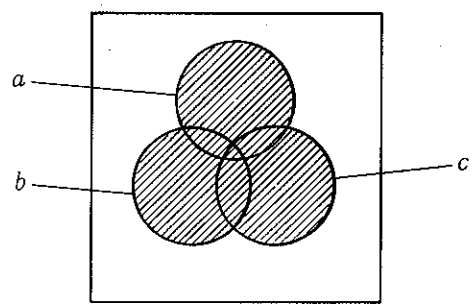
then be represented by a figure-of-eight area like this.



The logical operation employed here is called the 'OR operation', and is expressed in symbolic form by

$$a \vee b$$

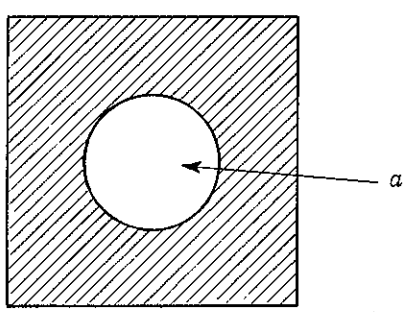
(Read 'a or b'. The symbol $\vee$ is the first letter of Latin 'vel', meaning 'or'.) The OR operation, like the AND operation, can be applied to three or more operands.



The OR operation is sometimes referred to as 'mixing'; the reason for this term will be evident when the electrical design of the packages is considered.

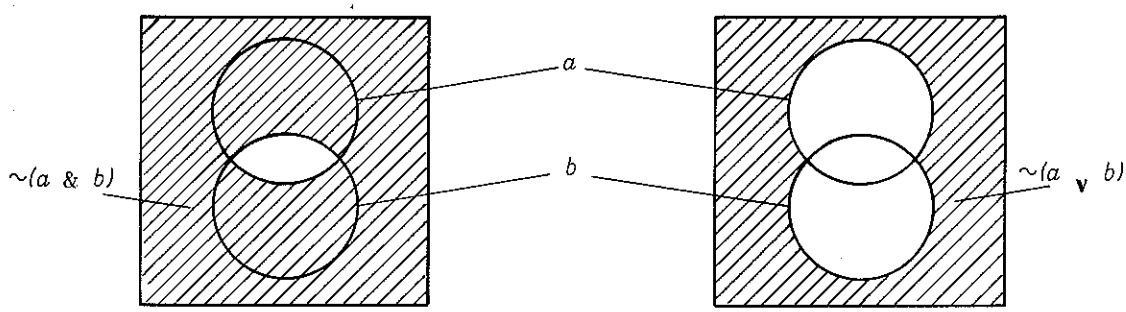
NOT Operation

4. It will be noticed that the AND and OR operations require at least two operands. To these must be added a third operation, which can be applied to one operand alone. This is the 'NOT operation', also called 'inversion'. Applied to the class of all those who can use a soldering iron, the NOT operation would produce all those who *cannot* use a soldering iron. In the following diagram, if the unshaded area represents *(a)*, the shaded area will represent NOT *(a)*.



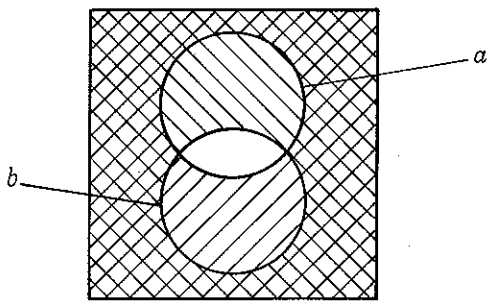
Algebraically, the NOT operation is shown by putting a bar over the operand, thus $\bar{a}$, or by putting a tilde before it, thus $\sim a$.

5. The NOT operation can, of course, be applied to the result of a previous logical operation. For instance, these two figures



show shaded areas representing $\sim(a \,\&\, b)$ and $\sim(a \vee b)$. Similarly, the AND and OR operations can be applied to the results of inversion. In the next diagram, areas representing $\sim a$ and $\sim b$ are shaded in different directions; the cross-hatched area is therefore

$$\sim a \,\&\, \sim b$$



6. It will be noticed that the cross-hatched area in the last figure is exactly the same as the shaded area representing $\sim(a \vee b)$ in the figure preceding it. This illustrates an important logical theorem, which may be stated algebraically, thus

$$\sim a \,\&\, \sim b \;=\; \sim(a \vee b)$$

It can be shown similarly that

$$\sim a \vee \sim b \;=\; \sim(a \,\&\, b)$$

In other words, the AND operation before inversion is equivalent to the OR operation after inversion and vice versa.

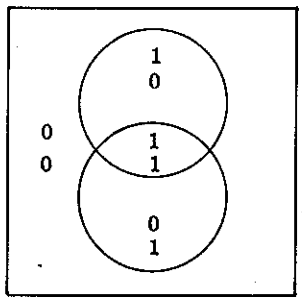
Truth Tables

7. A method has been given above whereby the consequences of logical operations may be investigated with the aid of diagrams. Another graphical aid to understanding logical processes is the 'truth table'. With the two hypothetical qualifications for computer maintenance engineers given at the beginning of this chapter, there are the following four possibilities: possession of both qualifications, possession of qualification *(a)* only, possession of qualification *(b)* only, possession of neither. These four possibilities are represented by the four *columns* in the table below, in which a 1 represents the presence, and a 0 the absence, of the qualification

qualification *(a)* 1 1 0 0

qualification *(b)* 1 0 1 0

These four columns represent the four areas into which a square is divided by two overlapping circles as shown below.



8. The truth table can now be expanded to include logical combinations of a and b as follows:

a	1 1 0 0
b	1 0 1 0
a & b	1 0 0 0
$a \vee b$	1 1 1 0
$\sim a$	0 0 1 1
$\sim b$	0 1 0 1

The combination *(a & b)*, i.e. a and b *both* present, occurs only in the first column, where a 1 must be written in the row representing *(a & b)*. The combination $(a \vee b)$, a or b or both, occurs in the first three columns, but not in the fourth. The inverses, $\sim a$ and $\sim b$, are obtained simply by substituting a 0 for a 1 and a 1 for a 0 in a and b. It will be noticed that the *(a & b)* row gives the product that would be obtained by multiplying together, column by column, the digits in the first two rows. Thus

$$1 \times 1 = 1$$

and in all other cases the product is zero. The *(a & b)* row also gives the 'carry' digit that would be produced in a binary sum, column by column, of the digits in the first two rows. The other results of binary arithmetical operations can be produced, as will be shown, by more complex combinations of logical operations.

Boolean Algebra

9. The algebra of logic is called Boolean algebra after George Boole, its inventor, a Lincoln mathematics teacher. Consider the following identity in ordinary algebra.

$$a \times (b + c) = (a \times b) + (a \times c)$$

The identity will still hold if the arithmetical operators are replaced by logical operators

$$a \,\&\, (b \vee c) = (a \,\&\, b) \vee (a \,\&\, c)$$

This can easily be checked diagrammatically, or by using a truth table. Eight columns are now necessary to exhaust the eight possible combinations of a, b and c. Thus

a	1 1 1 1 0 0 0 0
b	1 1 0 0 1 1 0 0
c	1 0 1 0 1 0 1 0
$b \vee c$	1 1 1 0 1 1 1 0
$a \,\&\, (b \vee c)$	1 1 1 0 0 0 0 0
$a \,\&\, b$	1 1 0 0 0 0 0 0
$a \,\&\, c$	1 0 1 0 0 0 0 0
$(a \,\&\, b) \vee (a \,\&\, c)$	1 1 1 0 0 0 0 0

The fifth and eighth rows of the truth table are identical, proving the truth of the proposition.

10. An important feature of Boolean algebra is that the operators & and $\vee$ have identical properties, whereas the arithmetical operators + and x have not. In mathematical terms, & and $\vee$ are distributive over each other, whereas + is not distributive over x. If, in the first identity in paragraph 9, the + and x signs are interchanged, the identity no longer holds for all values of a, b and c. If the signs are interchanged in the second identity, however, a new identity is produced

$$a \vee (b \,\&\, c) = (a \vee b) \,\&\, (a \vee c)$$

which can be checked as before.

11. Two further symbols are used in Boolean algebra. The symbol I is the totality element and would correspond to a row of ones in a truth table. In diagrammatic form, it is the square that contains all the circles representing operands. The symbol 0 is the null element; in the truth table it is a row of noughts: diagrammatically it is a circle of zero area. With these conventions the following identities emerge:

$$a \vee \sim a = \mathrm{I} \qquad a \,\&\, \sim a = 0$$

$$a \vee \mathrm{I} = \mathrm{I} \qquad a \,\&\, \mathrm{I} = a$$

to which may be added the trivial identities

$$a \vee a = a \,\&\, a = a$$

APPLICATIONS IN ARITHMETIC

Addition

12. It was shown in paragraph 8 that certain results of binary arithmetic appear in a simple truth table. The truth table for binary addition of two single-digit numbers is

number	*(a)*	1	1	0	0
number	*(b)*	1	0	1	0
sum	*(s)*	0	1	1	0
carry	*(c)*	1	0	0	0

It is important to realise that the four columns are treated individually here, the table is *not* meant to represent the addition of two four-digit binary numbers. As was stated in paragraph 8, the carry digit is given by

$$c = a \,\&\, b$$

The occurrence of the sum digit can be expressed in words by '*a* or *b*, but not both'. In algebraic symbols, therefore

'*a* or *b*' becomes $a \underset{\mathbf{v}}{} b$

'not both' becomes $\sim(a \,\&\, b)$

Hence the sum digit is given by

$$s = (a \underset{\mathbf{v}}{} b) \,\&\, \sim(a \,\&\, b)$$

This formula can be checked by building it up step by step in a truth table in the manner shown in paragraph 9.

13. The occurrence of the sum digit in the table might have been expressed alternatively by the phrase, '*a*, but not *a and b*; or *b*, but not *a and b*'. In algebraic form therefore

$$s = [a \,\&\, \sim(a \,\&\, b)] \underset{\mathbf{v}}{} [b \,\&\, \sim(a \,\&\, b)]$$

There are obviously other ways of writing the formula for the sum digit; the two given here have been chosen because of their immediate practical application.

14. The truth table given in paragraph 12 does not represent the full process of binary addition, as no account has yet been taken of the carry digit. If c' is the carry digit from a previous operation and c the carry digit to the next operation, the full truth table for addition is

c'	1	1	1	1	0	0	0	0
a	1	1	0	0	1	1	0	0
b	1	0	1	0	1	0	1	0
s	1	0	0	1	0	1	1	0
c	1	1	1	0	1	0	0	0

It will be noticed that a carry digit is generated when a 1 is present in at least two of a, b and c'. The full formula for the carry digit is therefore

$$c = (a \,\&\, b) \;\vee\; (b \,\&\, c') \;\vee\; (a \,\&\, c'),$$

the OR operations ensuring that the special case is included when a 1 is present in a, b and c together. This formula can be condensed somewhat by combining any two of the bracketed terms; thus

$$c = [a \,\&\, (b \vee c')] \;\vee\; (b \,\&\, c');$$

15. A sum digit is generated in the presence of a 1 in one or all of a, b and c'; but there is no sum digit when a 1 is present in only two rows. Several formulae for s are possible in terms of a, b and c'. One of these is given below on account of its practical application; the proof is left as an exercise for the reader

$$s = \sim[\sim(b \,\&\, c') \;\&\; (b \vee c') \,\&\, a] \;\&\; [a \vee \{\sim(b \,\&\, c') \;\&\; (b \vee c')\}]$$

Subtraction

16. The process of binary subtraction can be analysed in the same way as addition. The four possible cases where two single-digit numbers are involved are

1	1	0	0	number *(a)*
- 1	0	1	0	number *(b)*
0	1 ...111		0	difference *(d = a - b)*

These may be written in the form of a truth table:

a	1	1	0	0
b	1	0	1	0
$d = (a-b)$	0	1	1	0
c	0	0	1	0

It will be noticed that the 'difference' row is exactly the same as the 'sum' row given in the addition table (refer to paragraph 12). This is convenient, because it means that the same circuit arrangement, with minor modifications, can be used for addition or subtraction. The carry digit occurs when b is greater than a. It corresponds to the case when $\sim a$ and b are present together, i.e. it can be expressed by the formula.

$$c = \sim a \,\&\, b$$

17. The full truth table for subtraction, in which the carry digit, c', from the previous column is taken into account, is given below. The first four columns can be compiled from the example given in paragraph 13 of Chapter I; the last four columns are, of course, the same as the four in the truth table given in the preceding paragraph, with 'noughts' in the c' row.

c'	1	1	1	1	0	0	0	0
a	1	1	0	0	1	1	0	0
b	1	0	1	0	1	0	1	0
$d_{(a-b)}$	1	0	0	1	0	1	1	0
$c_{(a-b)}$	1	0	1	1	0	0	1	0

As before, row d is the same as row s in the addition table.

18. The carry digit in subtraction is what would be generated if $\sim a$ were substituted for a in the addition table. Consequently, the formula for the subtractive carry digit can be derived from that for the additive carry digit (paragraph 14) by a simple substitution, viz:

$$c_{(a-b)} = [\sim a \ \& \ (b \ \mathbf{v} \ c')] \ \mathbf{v} \ (b \ \& \ c')$$

Multiplication and Division

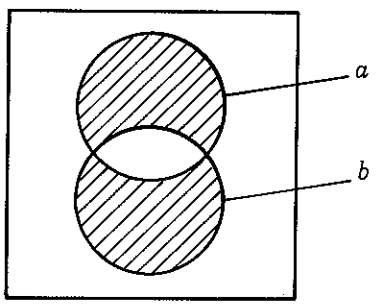
19. Multiplication consists simply of a series of additions and shifts, the partial products being given by the operation $(a \ \& \ b)$. Division, as was shown in Chapter I, can be carried out by a series of additions and subtractions, the next operation and the sign of the quotient digit being determined by the sign digit resulting from the previous operation. (See Chapter I, paragraph 23 *et seq.*)

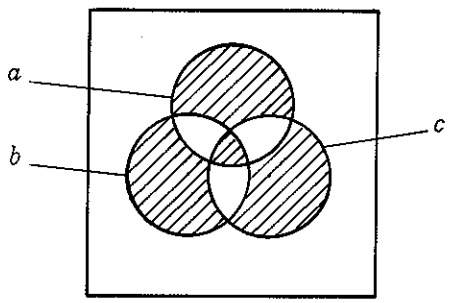
OTHER LOGICAL OPERATIONS

20. As well as the AND and OR operations and the inversion operation discussed at the beginning of this chapter, it is sometimes useful to work in terms of another operation, the NOT-EQUIVALENT operation, which is written

$$a \nleftrightarrow b$$

(read 'a not equivalent to b'). It amounts to a selection of those instances where a and b are not the same, e.g. those candidates for the post of computer maintenance engineer who have either of the two qualifications but not both. The NOT-EQUIVALENT operation on two and three operands is shown in terms of shaded areas in the diagrams. It is also referred to as the 'EXCLUSIVE-OR' operation.





21. In truth table form, the various possibilities for the NOT EQUIVALENT operation are shown below for two and three operands.

a	1 1 0 0
b	1 0 1 0
$a \nleftrightarrow b$	0 1 1 0

c	1 1 1 1 0 0 0 0
a	1 1 0 0 1 1 0 0
b	1 0 1 0 1 0 1 0
$(a \nleftrightarrow b) \nleftrightarrow c$	1 0 0 1 0 1 1 0

If reference is made to previous truth tables, it will be seen that the NOT-EQUIVALENT operation gives the sum digit in binary addition and the difference digit in binary subtraction. This fact cannot readily be used in computer work, as there is no simple electronic circuit that will perform the operation. Moreover, the NOT-EQUIVALENT operation does not obey the rules of simple algebra, in that it is non-distributive, and should consequently be avoided as far as possible. The expressions given in paragraphs 12 and 13 for the sum digit can, of course, be used to give the NOT-EQUIVALENT operation in terms of the AND, OR and NOT operations.

$$a \nleftrightarrow b = (a \vee b) \,\&\, \sim(a \,\&\, b)$$

$$a \nleftrightarrow b = [a \,\&\, \sim(a \,\&\, b)] \vee [b \,\&\, \sim(a \,\&\, b)]$$

22. An important theorem on the NOT-EQUIVALENT operation can be enunciated algebraically as follows:

$$a \nleftrightarrow (a \nleftrightarrow b) = b$$

and conversely

$$b \nleftrightarrow (a \nleftrightarrow b) = a$$

This principle can be used to exchange the locations of two numbers in a computer without the need for a third location as a temporary store. Thus the process to be used is:

(i) replace a with $(a \nleftrightarrow b)$
(ii) replace b with $b \nleftrightarrow (a \nleftrightarrow b) = a$
(iii) replace $(a \nleftrightarrow b)$ in the original location of a with $a \nleftrightarrow (a \nleftrightarrow b) = b$

A trivial identity with a useful application is

$$a \nleftrightarrow \mathrm{I} = \sim a$$

Thus a NOT-EQUIVALENT operation taken *digit-by-digit* between a binary number

$$a = 0.11011 \quad \text{say}$$

and a 'full house' of 'ones', representing I, thus

$$\mathrm{I} = 1.11111$$

would give the result.

$$1.00100$$

Now the negative of a, with the convention referred to in Chapter I of representing a negative number as its complement with respect to 2, is

$$-a = 1.00101$$

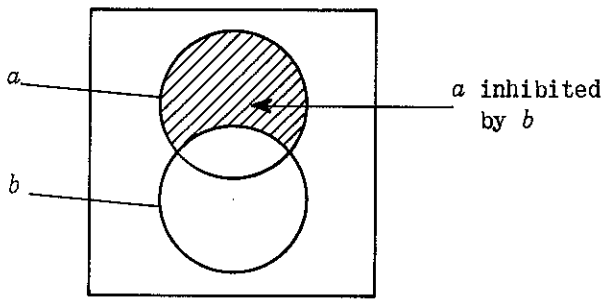
which is the same as the result of the NOT-EQUIVALENT operation, but with the least significant digit changed.

CHAPTER III

BASIC LOGICAL CIRCUITS

With a suitable correcting operation, therefore, the NOT-EQUIVALENT operation can be used for forming arithmetical complements.

23. Another important logical operation is that of 'inhibition', the suppression of one operand when the other is present; diagrammatically



This operation is obviously equivalent to

$$a \,\&\sim b$$

In general, any logical operation can be expressed in terms of others. It is interesting to note that there is one logical operation, NEITHER-NOR, from which all the others can be synthesised. For instance

$$a \text{ NN } a = \sim a$$

also $$a \text{ NN } b = \sim(a \vee b)$$

hence $$a \vee b = \sim(a \text{ NN } b)$$

$$= (a \text{ NN } b) \text{ NN } (a \text{ NN } b)$$

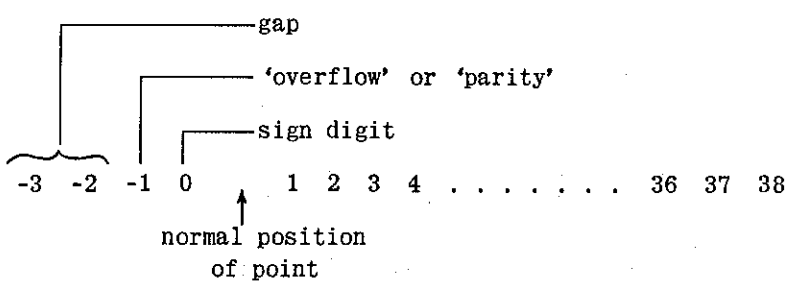
CHAPTER III

BASIC LOGICAL CIRCUITS

1. In Pegasus, a binary number is represented by a time series of voltage pulses, the presence of a pulse at a particular instant representing a 'one' in the corresponding binary place, and its absence representing a 'nought'. The pulses, which have a peak level between +2 V and +13 V and a base level of -10 V, are fed into logical elements; these are electronic circuits that perform operations analogous to the logical operations discussed in Chapter II. The logical elements can be grouped so as to perform the relatively more complex operations of binary arithmetic. It should be realised at the outset that, since arithmetical operations are most conveniently begun at the less significant ends of numbers, the pulse trains representing those numbers should be fed through the system with their least significant digits first. The digit pulses, as seen on an oscilloscope with a normal left-to-right timebase, would appear, therefore, in the reverse order to that of the corresponding digits in the conventional representation of a binary number. For this reason, the oscilloscopes in the built-in monitor in Pegasus have been designed with right-to-left timebases. The timing charts, Volume 2b, have been drawn with the time arrow running from right to left for the same reason.

2. The longest binary number that can enter into arithmetical operations as a single unit is called a 'word'. In Pegasus, the word consists of 38 significant binary digits with a 'sign' digit placed at the more significant end (and therefore appearing last in time). This means that decimal numbers up to approximately a quarter of a billion (British) can be dealt with, or an equivalent accuracy can be obtained in fractions, without the need for double-length working. The full word length corresponds to 42 binary digits. Two of these are zero, and act as a buffer between one word and the next. The digit occurring next in time after the sign digit may be 1 or 0; it is used for overflow testing (see Chapter I, paragraph 16) or for checking against storage errors by the 'parity count' method (see Chapter X).

3. For convenience, the digit places are numbered according to the power of 1/2 given in their place value when the binary point is assumed to be in the normal position, i.e. after the sign digit. Thus the sign digit is numbered 0, the 'overflow' or 'parity' digit -1, and the two gap digits -2 and -3, like this:-



Digit timing in Pegasus is referred to ϕ-pulses that run from 0 to 41. A word is said to be in 'standard timing' when digit-pulse 38 occurs simultaneously with timing pulse ϕ0; the sign digit then occurs with ϕ38 and the two gap digits with ϕ40 and ϕ41. In general, if a word is delayed on standard timing by t digit-times, digit-pulse d will occur with timing-pulse ϕ where

$$\phi = 38 + t - d$$

LOGICAL ELEMENTS

4. The logical OR operation on pulses is performed by a network of crystal diodes and resistors called an OR gate; this is arranged so that an output is obtained if a pulse is present on at least one of the input terminals. (This type of circuit is sometimes referred to as a 'threshold-1' element). The AND operation is performed by an AND gate, another network of crystal diodes and resistors, which gives an output only if a pulse is present on *all* the input terminals. There are three types of AND gate in Pegasus, having two, three or four input terminals. (These can be referred to as 'threshold-2', 'threshold-3' and 'threshold-4' elements if all the input terminals are used). It is worth noting here that a three-input AND gate of the type used in Pegasus can be used as a two-input gate, or a four-input gate as a three-input gate, by leaving one input 'floating'. The NOT operation (inversion) is performed by a thermionic-valve circuit, which inverts the digit-pulse waveform.

5. In addition to the AND and OR gates and inverters, the system requires some method of shifting a binary number one digit place to the left, i.e. of delaying the pulse train by one digit-time. Digit delays are required in forming the partial products in multiplication; also, the carry digit in addition and subtraction must be delayed to bring it into the next 'column'. The action of a digit delay is essentially that of deleting a pulse and reshaping it one digit time later; digit delays are often used, therefore, to 'clean up' pulses that have become attenuated or mis-shapen or have been slightly delayed by stray circuit time constants. It is also necessary to delay pulse trains by one word-time; a word can be stored (i.e. 'remembered') for an indefinite time by making it circulate repetitively through such a one-word delay. The type of one-word delay used in Pegasus is a magnetostrictive nickel line; pulses are propagated down a line of appropriate length as compression waves, and are amplified and reshaped after detection at the remote end. Further details of the logical elements will be found in a companion volume to this handbook.

6. Fig. 3.1 shows the symbols used for AND gates, OR gates, inverters and delays; later it will be necessary to introduce symbols to represent the combinations of these elements that make up the logical packages. Note that the presence of 'arrowheads' distinguishes an OR gate from an ordinary soldered connection in which a lead divides into two branches.

7. Fig. 3.1 also shows how an inverter, two AND gates and a nickel line can be used as a one-word store. The input word is allowed through AND gate x by a train of gating pulses, which are positive for one word-time only. It passes down the line, emerging one word-time later, and recirculates through AND gate y as long as the output of the

inverter is positive, i.e. as long as there are no pulses applied at the input of the inverter. The content of the store can be erased simply by feeding one word-length of pulses into the inverter so as to close AND gate y. Another type of 'memory' circuit is the 'staticiser', which will 'remember' a single digit only. The circuit is similar (see fig. 3.1) except that a digit delay is used instead of a nickel line. The digit to be 'remembered' is selected by a setting pulse, which is applied to AND gate x at the appropriate instant. If the digit is a 1, so that it is represented by a positive pulse, a pulse will circulate via AND gate y until it is erased by a resetting pulse fed at any instant into the inverter.

ARITHMETICAL CIRCUITS

Adders

8. In paragraph 10 of Chapter II, it was shown that the carry digit in addition is obtainable by a simple AND operation, i.e.

$$c = a \,\&\, b$$

Hence the first logical circuit in fig. 3.2 would be enough to generate the carry digit, the delay being needed to bring the digit into the next 'column' for the next step in the addition. The binary digits inserted in the diagram are not intended to be binary numbers, but are truth-table representations of conditions at various points in the circuit. Thus

a	1 1 0 0
b	1 0 1 0
$a \,\&\, b$	1 0 0 0

9. The sum digit in addition can be expressed by the formula

$$s = (a \vee b) \,\&\, \sim(a \,\&\, b)$$

This sequence of operations can be expressed in words as follows:-

(i) OR operation on a and b.
(ii) AND operation on a and b.
(iii) Invert result of step *(ii)*.
(iv) AND operation on results of steps *(i)* and *(iii)*.

The second logical circuit shown in Fig. 3.2 will perform this series of operations; as before, digits are inserted in the diagram to represent the truth-table condition at each point, thus:

a	1 1 0 0
b	1 0 1 0
$a \vee b$	1 1 1 0
$a \,\&\, b$	1 0 0 0
$\sim(a \,\&\, b)$	0 1 1 1
$s = (a \vee b) \,\&\, \sim(a \,\&\, b)$	0 1 1 0

10. The circuits just described will generate the carry and sum digits when two binary numbers are added; but they cannot perform the full process of addition, as no account has yet been taken of the carry digit, c', from the previous column. The combination of these two circuits is called a 'half adder'. It can be used in cases where a digit cannot be present both in the 'carry' and in input b. The third circuit in fig. 3.3 shows a half adder. During the first digit-time, a digit at b is added to the digit at a; during the second and successive digit times, any carry digit produced in the previous place is added to the digit at a.

11. In paragraph 13 of Chapter II, an alternative formula for the sum digit was given. This was

$$s = [a \,\&\, \sim(a \,\&\, b)] \;\vee\; [b \,\&\, \sim(a \,\&\, b)]$$

which implies the following sequence of operations:

(i) AND operation on a and b.
(ii) Invert result of step *(i)*.
(iii) AND operation on a and result of step *(ii)*.
(iv) AND operation on b and result of step *(ii)*.
(v) OR operation on results of steps *(iii)* and *(iv)*.

An alternative circuit for generating the sum digit, and a half adder built on this principle are shown on the right-hand side in fig. 3.2; as before, truth-table digits are inserted in the figure at the appropriate points. Two of the AND gates in each of the half adders are doing the same job, producing $(a \,\&\, b)$, and could, of course, be replaced with a single gate. However, small economies like this have been sacrificed in Pegasus in favour of greater economies resulting from the use of standard packages.

12. A half adder can only deal with two full-length inputs, one of which must be, after the first digit-time, the sequence of carry digits, whereas the full process of addition requires three inputs - two numbers and a carry sequence. It is possible, however, to make a full adder by combining two half adders as shown in fig. 3.3; the arrangement is shown schematically on the left. In the first half adder, number b is added to the carry sequence; in the second half, the sum of these two is added to number a to give the final output. The carry digit can arise in either half of the addition, but not in both, as can be easily verified from the truth table: the two 'carry' outputs can therefore be combined in an OR gate to give the final carry sequence.

13. The central diagram of fig. 3.3 shows the complete circuit of a full adder evolved from two half adders. One half adder is of the first type discussed: the other half adder is of the second type. The truth-table for full addition is given below, and the lines of this table are inserted in the appropriate places in the figure.

	c'	1 1 1 1 0 0 0 0
	b	1 0 1 0 1 0 1 0
↓	carry	1 0 1 0 0 0 0 0
	sum 1	0 1 0 1 1 0 1 0
	a	1 1 0 0 1 1 0 0
OR ←	carry	0 1 0 0 1 0 0 0
↓	sum 2	1 0 0 1 0 1 1 0
→	c	1 1 1 0 1 0 0 0

This is the most convenient general arrangement for the Pegasus packages; but there are certain economies that can be made without upsetting its operation. First, two 'carry' delays are unnecessary: the OR operation can take place as well before the delay as after it. Second, the AND gate numbered 6 is unnecessary. It can easily be verified - by a truth-table or otherwise - that two two-input AND gates in cascade can be replaced by one three-input AND gate. AND gate 6 may therefore be omitted; its two inputs would then be applied direct to AND gates 7, 8 and 11, which would become three-input gates. Third, no connection need be made between inverter 5 and AND gate 7. It will be recalled that the purpose of gate 3 and inverter 5 is to suppress the *sum* digit in the first half adder when b and c' are present together; it is therefore unnecessary to connect these to gate 7, which is concerned only with the carry digit. The carry digit resulting from the simultaneous presence of b and c' is produced in any case by gate 1. The final circuit of the full adder is therefore as shown in the third diagram of fig. 3.3.

Subtractors

14. In Chapter II (paragraph 16 *et seq.*) it was shown that the difference digit in subtraction is identical with the sum digit in addition, and that the two operations differ only in the carry sequence generated. It was shown, too, that the subtractive carry can be produced in the same way as the additive carry if $\sim a$ be substituted for a in its generation so that the carry digit becomes

$$\sim a \ \& \ b \quad \text{instead of} \quad a \ \& \ b$$

The short truth tables are given below for reference.

a	1 1 0 0	a	1 1 0 0
b	1 0 1 0	b	1 0 1 0
$s_{(a+b)}$	0 1 1 0	$d_{(a-b)}$	0 1 1 0
$c_{(a+b)}$	1 0 0 0	$c_{(a-b)}$	0 0 1 0

It would, of course, be possible to modify the half adder to form a half subtractor by supplying an extra inverter to form $\sim a$ from a; but this is not really necessary. The inverter already in the half adder produces $\sim(a \ \& \ b)$; an AND operation between its output and b would therefore give $\sim(a \ \& \ b) \ \& \ b$, which, by Boolean algebra, can be shown to be equivalent to

$$(\sim a \ \vee \ \sim b) \ \& \ b$$

$$= \sim a \ \& \ b \qquad (\text{since } \sim b \ \& \ b \ = \ 0)$$

Thus a half subtractor can be formed from a half adder simply by connecting the output of the inverter, instead of a, to the AND gate producing the carry digit. A half subtractor derived from the second form of half adder is shown in fig. 3.4, with 'truth-table' lines inserted as appropriate.

15. The half subtractor, like the half adder, cannot deal with two input sequences as well as a carry sequence. A full subtractor can, however, be built up from a half

adder and a half subtractor. In this context, it should be remembered that the subtractive carry digit is to be regarded as negative, i.e. it must be added to b if the difference $(a - b)$ is required and to a if the difference $(b - a)$ is required. This point is illustrated in the schematic diagram at the top of fig. 3.4. The full-subtractor circuit of fig. 3.4 is derived from this in the same way as the full-adder circuit was derived, one delay, one AND gate and one unnecessary connection being eliminated. The full truth table is given below for reference.

c'	1 1 1 1 0 0 0 0
a	1 1 0 0 1 1 0 0
b	1 0 1 0 1 0 1 0
$d_{(a-b)}$	1 0 0 1 0 1 1 0
$c_{(a-b)}$	1 0 1 1 0 0 1 0

16. The adder and subtractor circuits can be combined to form the adder-subtractor circuit shown in fig. 3.4. When the circuit is used as an adder, AND gate 16 is opened by the triggering waveform so that a is used in generating the carry sequence. If it is required to use the circuit as a subtractor, gate 17 is opened instead of gate 16, so that the output of inverter 10 is used in place of a for generating the carry sequence.

CIRCUIT SYMBOLS

17. Certain combinations of logical elements occur very frequently in logical circuits. For example, the arrangement of a digit delay preceded by an OR gate fed from two AND gates occurs in the adder-subtractor and in the staticiser. For this reason, the standard packages used in Pegasus have been designed in terms of common combinations of logical elements. Inevitably there are components that remain unused in the logical design; but the number of package types is reduced, with consequent manufacturing economies. As well as the logical elements already discussed, the use of cathode followers is demanded by considerations of output loading. Furthermore, output amplifiers are required so that the results of computer operations can be transmitted to some form of output mechanism.

18. Nine types of logical package are used in Pegasus. The symbols for the combinations of logical elements on these are given in fig. 3.5. It will be noticed that there are generally several logical combinations to a package. The basic elements that incorporate thermionic valves, *viz.* digit delays, nickel lines, inverters and cathode followers, are generally preceded by crystal gates. Digit delays may be preceded by one three-input AND gate (single delay, package type 2) or by two three-input AND gates feeding an OR gate (twin delay, package type 1). Nickel lines (package type 6), of which there is only one to a package are fed in the same way as twin delays; the output stage is actually a digit-delay circuit. The nickel-line package also contains a special inverter for use as the 'erase' element of a one-word store. Two lengths of nickel line, 35-digits and 42-digits, are used in Pegasus. Most of the logical combinations have alternative outputs, a direct output and a 'mix' output through an OR crystal (which is drawn obliquely on the symbol). OR gates are usually made up from these 'mix' outputs; otherwise the combination of OR gate and cathode-follower (package type 8) is used.

19. The number generator, package type 7, is used for producing a train of digit pulses from tape-reader outputs or handkey settings. It consists of three two-input AND gates whose outputs feed a cathode-follower through an OR gate. This cathode-follower has a 'mix' output only, as several such elements must generally be used together in practice. Output-one elements, package type 9, are used for feeding neon indicator lamps or external-conditioning relays. In the latter application two elements are used in parallel, and are preceded by two AND gates feeding an OR gate, suitable gating networks being provided on the packages. Output-two elements, package type 10, can supply more power than output-one elements, and are used for driving the output equipment.

20. All packages except those of type 6 (nickel line) carry two or more identical elements. These are numbered in logical diagrams, as shown in the last column of fig. 3.5, to correspond to the related pin numbers on the package connected, the element with the lowest pin numbers being referred to as element 1 on the package. Where an element has two or more entry gates, these are distinguished from one another with the letters *x*, *y* and *z*, the letter *x* referring to the gate with the lowest pin numbers. The actual construction and layout of the packages is the subject of a separate handbook, to which the reader is referred for details of pin numbers and component values.

21. The packages may carry up to seven test points, which are accessible from the front of the cabinet. A test point is provided at the output of each element, those elements with alternative direct and 'mix' outputs generally being provided with a test point for each. All 'mix' test points are marked 'M', with reference number of the corresponding element on the package. Direct-output test points are marked with a letter, representing the type of logical circuit, and the reference number. Cathode-follower test points are marked with a number only. The test points corresponding to elements 4 and 5 on the cathode-follower package (type 8) may be connected either to the direct or to the 'mix' output as convenient.

22. The Pegasus computing cabinet comprises three bays, each of which is large enough to accommodate ten rows of twenty packages each. There are only four rows in the first bay; these are numbered from the top downwards 10 to 13. The rows in the second and third bays are numbered 20 to 29 and 30 to 39 respectively. The packages within a row are designated by letters, which are in alphabetical order from left to right when the row of packages is viewed from the front of the cabinet. The letters used are:

C D E F G H J K L M P Q R S T U V W X Y

Letters B, I, O and N are omitted, and letters A and Z are used for the interconnectors between bays. In logical diagrams, each symbol is marked with the row number and the letter of the package carrying the corresponding components, and the reference number of the element on the package. Thus a logical symbol marked 26G1 would correspond to the first element on the fifth package in the seventh row down of bay 2.

Applications

23. To illustrate the use of the combined symbols the staticiser and one-word store of fig. 3.2 and the adder-subtractor of fig. 3.4 are redrawn in fig. 3.6. As inverters do not exist in Pegasus as separate entities, a combination of inverter and AND gate must be used for erasing the content of the store or for clearing (resetting) the staticiser. Note how the 'mix' outputs of the cathode-followers after the 'add' and 'subtract' AND gates in the adder-subtractor are used together to make an OR gate. Note too that the output is taken through a delay; delays are often used in Pegasus, as here, simply to restore the amplitude and timing of a waveform after a series of logical operations. When a negative number is produced by subtraction, an infinite series of carry digits is generated. As only one of these, the sign digit, is necessary for general purposes, or two of them for overflow detection, a 'carry suppression' waveform is employed to close the two AND gates preceding the 'carry' delay at the appropriate instant.

24. Fig. 3.6 also shows a parity-counting circuit. If the number of 'ones' in a binary number is odd, the parity is said to be odd: if the number is even, the parity is even. Parity counts applied at strategic points in a computer provide a simple method of checking for circuit faults automatically; if any one digit changes its value, the fact will be detected immediately. The circuit shown consists of a staticiser, S, an inverter (preceded by an AND gate) and a cathode-follower. The staticiser is 'set' just before the digit waveform appears at the inputs of the cathode-follower and inverter, so that a pulse circulates through the 'y' gate until the first 'one' in the digit waveform appears; this acts as an 'erase' signal and resets the staticiser, The output of the staticiser is now zero, so the AND gate before the inverter is closed. The next 'one' in the digit waveform will pass through the cathode-follower and activate the staticiser again, until it is reset again by the next 'one'. Owing to the staticiser delay, the final parity will not be indicated until after the last digit has passed. The parity count will then be positive for a waveform with even parity, and zero for a waveform with odd parity. Note the use here again of 'mix' outputs to form an OR gate.

C H A P T E R I V

STORAGE AND THE ORDER CODE

CHAPTER IV

STORAGE AND THE ORDER CODE

Readers who are not conversant with the techniques and terminology of computer engineering are advised to study the Introduction to Volume I before reading any further in this volume.

STORAGE FACILITIES

Main Store

1. The main store in Pegasus is a drum coated with ferromagnetic material and rotating at 3720 r.p.m. It carries 40 information tracks (strictly speaking, track pairs; alternate digits are written on alternate tracks as will be explained in Chapter XI) of which 32 are available for normal storage; the other 8 tracks are reserved for the permanent record (initial orders and test routines) and are capable of being 'read from' but not 'written on'. Each track carries 128 words, which are arranged, from the standpoint of the programme, in 16 'blocks' of eight words each. The magnetic drum will thus accommodate up to 512 8-word blocks, or 4096 words, in normal-storage locations, and 128 8-word blocks, or 1024 words, as a permanent record.

Computing Store

2. Transfers of information between the main store and the immediate-access computing store can be made in single words or in blocks of eight words. The computing store contains seven blocks of eight registers, most of which are of the type described in Chapter III. One block, which is treated as block 7 for transfer purposes, consists of accumulators. These are not accumulators in the strict sense: they do not all incorporate arithmetical circuits; but they are always specified or implied in arithmetical orders, and they are always used in accumulative arithmetical processes.

3. The first accumulator, accumulator 0, is imaginary; but it is treated in orders as if it were a nickel line containing one word-length of zeros, and can thus be used, for instance, for erasure. It will be seen that it is logically convenient for one of the accumulators to be permanently empty. Accumulator 1 is the source or destination for single-word main-store transfers; otherwise it is used in precisely the same way as accumulators 2 to 5. Accumulators 6 and 7 form part of the multiplier-divider. A double-length product is formed with its more significant half in accumulator 6 and its less significant half in accumulator 7: in division, the quotient is formed in accumulator 7 and the residue in accumulator 6. Accumulators 6 and 7 also contain facilities for carrying out double-length shifts; they can, of course, be used in the same way as the other accumulators if desired.

4. The standard input and output devices are treated as special registers in the computing store. Address 15 refers to the row of twenty handkeys on the programmer's panel, on which a half-length number can be set up by the operator. Orders specifying address 17 call for information to be read in or punched out, according to the function in the order, so that the holes in the tape correspond exactly with digit pulses in the machine. Address 16 has a similar significance to address 17, except that there is an automatic conversion between the character on tape, which is in a special self-checking code, and the number that it represents. There are four more special registers, which, although not nickel lines, are treated in orders as such. These, addresses 32 to 35, supply, in binary form, the constants -1, 2^{-1}, 2^{-10}, and 2^{-13}, which are used, principally, when orders are being collated by the input routine. The storage facilities in Pegasus are shown diagrammatically in fig. 4.1; addresses are given according to the convention used by programmers.

Symbols

5. In the literature on Pegasus, capital-letter symbols are used to represent storage addresses, or the digit groups in orders that correspond to these addresses, thus:

N = register address (ordinary register, special register or accumulator)

X = accumulator address

U = block in computing store

B = block in main store

S = single-word location in main store

The corresponding lower-case symbols are used to represent the contents of these addresses; these may be primed to indicate the contents of an address after an order has been obeyed. Thus

$$x' = x + n$$

means that the contents of accumulator X are added to the contents of register N, and the resulting sum is replaced in accumulator X.

$$u' = b$$

means that the contents of block B in the main store are transferred to block U in the computing store.

6. The letters P and Q, and the corresponding lower-case letters, are usually used in referring to accumulators 6 and 7 respectively. Otherwise a particular accumulator is referred to by 'X' followed by a number, and the contents of a particular accumulator by the number alone. Thus

$$s' = 1$$

indicates the transfer of the contents of accumulator 1 to address S in the main store.

As a modifier and a counter may be stored at opposite ends of the same accumulator, a distinction is made between the two parts of the word by subscript letters, thus:

$$x_m \qquad \text{and} \qquad x_c$$

THE ORDER CODE

7. The order code has been designed so that two orders can be stored in one register. The order length is 19 digits, the first order (A-order) occupying digit places 1 to 19, and the second order (B-order) occupying places 20 to 38. Digit place 0, the position of the sign digit in numbers, is occupied in order pairs by the stop-go digit. This digit is used in connection with the optional stop; if it is a 1 - the normal state of affairs - the order pair is obeyed in the usual way; if it is a 0, which can be ensured by punching the 'full-stop' character after either of the two orders on tape, the machine carries out an optional stop, i.e., it stops (unless the facility is inhibited by the panel key) after the order pair has been transferred to the order register, and before either order is obeyed. The stop-go digit is deleted before the order pair enters the order register.

8. A two-address code is used. That is to say, a typical order is one that requires the machine to extract two numbers from addresses in the store, perform an operation on them, and replace the result in one of the two addresses specified. The first seven digits of an order are referred to as N-digits, because they generally refer to an N-address (ordinary register, special register or accumulator). If an accumulator is specified as an N-address, the first four N-digits will all be zero, the last three digits giving the binary equivalent of the address. The special registers are called for by groups of N-digits giving the binary equivalents of the decimal numbers shown in fig. 4.1; it will be noticed that only the last six digits of the seven will be required. For ordinary-register addresses, the first N-digit is always 1; the next three digits give the block, and the last three the position within the block, in binary form.

9. After the seven N-digits (in the conventional representation, but before them in time) come three X-digits, which generally give, in binary form, an accumulator address. The X-digits are followed by six F-digits specifying the function, i.e. the type of operation that is to be performed. The last three digits of an order are referred to as M-digits; these give the address of a modifier, a number that must be added to the order before it is obeyed. The digit groups have been discussed here in their sequence from the more significant end of the order; this is the sequence in which the groups appear on the monitor displays, and also corresponds to the way in which the handkeys, which can be used for inserting a single order into the machine, are arranged. As was stated in the last chapter, words travel through the machine with their least significant digits first: consequently the M-digits of an order enter the order register first and are decoded first; these are followed by the F-digits, whose decoding determines the way in which the X-digits and F-digits are to be interpreted. Note, however, that, although the 'less significant' order of a pair enters the order register first, it is obeyed second.

Function Code

10. As far as possible, the functions that the machine can perform have been arranged in groups within which there is considerable logical similarity. This not only helps the programmer to remember the order code easily but also simplifies the logical design of the machine. The first three F-digits in the order specify the function group. As three binary digits can give up to eight combinations (corresponding to the binary representations of the numbers 0 to 7), eight function groups could be accommodated in the code. Seven groups are in fact used. The last three F-digits in the order denote the particular function in the group; consequently there can be up to eight functions in any one group. In programmer's notation, the functions are written as two-digit octal numbers; thus function 54 would be function 4 in group 5.

11. The simplest types of function are those that require addition, subtraction or logical operations between two numbers in the store. There are two groups of these; in the first the answer is to be replaced in an accumulator, in the second, the answer is to be replaced in a register. In these two function groups, the N-digits are to be interpreted as the address of a register (which may be an accumulator or a special register), and the X-digits are to be interpreted as the address of an accumulator.

12. The functions in group 0 - arithmetical and logical transfers from register to accumulator - are as follows:

0 0 Clear specified accumulator and transfer to it word in specified register.

0 1 Add number in register to number in accumulator. Answer in accumulator.

0 2 Clear accumulator and transfer to it negative of number in register.

0 3 Subtract number in register from number in accumulator. Answer in accumulator.

0 4 Subtract number in accumulator from number in register. Answer in accumulator.

0 5 AND operation between number in accumulator and number in register. Answer in accumulator.

0 6 NOT-EQUIVALENT operation between number in accumulator and number in register. Answer in accumulator.

0 7 Unallocated.

13. The functions in group 1 - arithmetical and logical transfers from accumulator to register - are as follows:

1 0 Clear register and transfer to it word in accumulator.

1 1 Add numbers in register and accumulator. Answer in register.

1 2 Clear register and transfer to it negative of number in accumulator.

1 3 Subtract number in accumulator from number in register. Answer in register.

1 4 Subtract number in register from number in accumulator. Answer in register.

1 5 AND operation between numbers in accumulator and register. Answer in register.

1 6 NOT-EQUIVALENT operation between numbers in accumulator and register. Answer in register.

1 7 Unallocated.

14. Function group 2 is used in multiplication and division orders. It was shown in Chapter II that the product of multiplication is generally a double-length number, and must be held in two store lines. The results of multiplication orders are automatically written with the more significant half in accumulator 6 and the less significant half in accumulator 7. The sign is indicated by the sign digit in accumulator 6, the sign digit in accumulator 7 being zero. If required, a rounded single-length product may be obtained in accumulator 6 as explained in Chapter I by adding 1 at the more significant end of accumulator 7 and allowing 'carry' between the two accumulators if necessary. As in orders containing function groups 0 and 1, the N-digits specify a register, and the X-digits specify an accumulator. The multiplication orders are:

2 0 Multiply together numbers in accumulator and register. Double-length product in accumulators 6 and 7.

2 1 Multiply together numbers in accumulator and register. Rounded single-length product in 6. Content of 7 is increased by 1 at m.s. end.

2 2 Accumulative multiplication. Multiply together numbers in accumulator and register and add double-length product to double-length number already in 6 and 7.

15. Pegasus has facilities for dealing with double-length dividends and single-length divisors. The divisor is called for by the N-digits of the order: the more significant half of the dividend (and the sign digit) is in the X-address; but the less significant half of the dividend must have been placed in accumulator 7 by a previous order. The quotient is formed in accumulator 7 and the residue (effectively shifted up 38 places) in accumulator 6. The division orders are:

2 4 Divide double-length number with m.s. half in specified accumulator and l.s. half in accumulator 7 by number in specified register. Quotient in 7: residue ($\times 2^{38}$) in 6.

2 5 Divide double-length number by single-length number as in 24. *Rounded* quotient in 7: corresponding residue ($\times 2^{38}$) in 6.

2 6 Divide *single-length* number in specified accumulator by number in specified register. Quotient in 7: residue ($\times 2^{38}$) in 6.

16. Operations on double-length numbers, which are held in two separate registers, must be done in two parts. In double-length addition and subtraction it will often happen that a digit must be carried from the less significant half to the more significant half. The machine will not do this automatically, but must be provided with a special order - a 'justify' order. The X-digits in 'justify' orders are not decoded. The N-digits specify a register (often accumulator 6), which contains the more significant half, the less significant half being in accumulator 7. 'Justify' orders contain function 23, which must be interpreted thus:

2 3 Justify. Do any 'carry' operation necessary to make a double-length number from the numbers contained in the specified register (more significant half) and accumulator 7 (less significant half).

17. No functions have been allocated to numbers in group 3; the machine will stop if a group-3 function appears in a programme. Orders containing group-4 functions are called counter orders. A counter is a number held at the less significant end of an accumulator, and is used in repetitive processes to ensure that a group of orders is obeyed the correct number of times. Unity is subtracted from the counter after each repetition, until the counter becomes zero, when the machine proceeds to the next part of the computation. Counters are also used as a convenient source of small numbers. In counter orders, the X-digits specify an accumulator address; but the N-digits must be interpreted as a binary number, the counter. The counter in an order can therefore have at most 7 binary digits unless it is extended by modification; the counter in an accumulator can be extended by additions or shifts up to 25 binary digits in length.

18. Functions in group 4 are closely similar to functions in group 0:

- 4 0 Clear accumulator specified by X-digits, and place N-digit (counter) number at less significant end.
- 4 1 Add counter number to less significant end of number already in specified accumulator.
- 4 2 Clear accumulator and place negative of counter number in it.
- 4 3 Subtract counter number from number already in accumulator.
- 4 4 Subtract number in accumulator from counter number, and place difference in accumulator.
- 4 5 AND operation between counter number and number in accumulator. Answer as a counter in accumulator.
- 4 6 NOT-EQUIVALENT operation between counter number and number in accumulator. Answer as a counter in accumulator.
- 4 7 Unallocated.

19. Multiplication or division by powers of 2 is done in the binary system simply by shifting the number up (towards the more significant end) or down (towards the less significant end) the requisite number of places, appropriate action being taken to carry out rounding if necessary or to reproduce the sign digit. Facilities are also provided for carrying out 'logical' shifts, i.e. shifts in which no provision is made for rounding or for dealing with a sign digit. The N-digits of shift orders (function group 5) are to be interpreted as a number indicating the number of places to be shifted. In single-length shift orders, the X-digits specify the accumulator containing the number to be shifted: double-length shift orders are carried out on the contents of accumulator 6 (more significant half) and 7 (less significant half), and the X-digits are not used. Double-length right shifts do not include rounding. The shift functions are:

- 5 0 Multiply the number in the specified accumulator by 2^N.
- 5 1 Divide the number in the accumulator by 2^N. Answer to be rounded.
- 5 2 Shift word in accumulator *up* N places (towards more significant end).
- 5 3 Shift word in accumulator *down* N places (towards less significant end).
- 5 4 Multiply double-length number in accumulators 6 and 7 by 2^N.
- 5 5 Divide double-length number in accumulators 6 and 7 by 2^N. Answer unrounded.

20. In the type of working known as 'floating point', numbers are expressed in two parts: the argument and the exponent. The argument indicates the digits in the number without regard to place value, and the exponent implies their place values. In Pegasus the argument is normally expressed as a fractional number with the two digits following the point not equivalent. Thus the binary number

1 1 0 0 . 1 0 1 1

is equal to

$2^5 \times 0 . 0 1 1 0 0 1 0 1 1$

and would therefore have exponent 5 and argument

0 . 0 1 1 0 0 1 0 1 1

'Normalise' orders (function 56) are provided to shift the argument of a floating-point number into this standard form.

21. The 'normalise' order can only be applied to a number whose argument is a double-length number held in accumulators 6 and 7 (or a single-length number in 6 alone with 7 empty). The X-digits of the order specify the accumulator holding the exponent, which is placed, like a counter, at the less significant end. Exponents will normally have up to nine binary digits. The N-digits in the order must be interpreted as setting an upper limit to the number of shifts permitted; this prevents the machine from continuing to shift if the argument of the number becomes zero.

5 6 Normalise. Shift argument in 6 and 7 into normal form - and subtract number of left shifts from exponent in specified accumulator. Right shifts regarded as negative.

22. Normally, Pegasus will carry out orders in the sequence in which they are stored in the registers. It may not always be necessary to carry out all the orders in the programme, or it may be necessary to repeat a group of orders several times. For example, in evaluating by successive approximations, a sequence of orders must be carried out repetitively until an approximation sufficiently close to the desired result is obtained. To enable the machine to break the normal sequence of orders, 'jump' orders (function group 6) have been provided. The 'jump' condition is made to depend on the value of a number held in the accumulator specified by the X-digits of the order. The N-digits specify the address of the register containing the order that is to be carried out next. Since orders can be held in only one of the 48 ordinary registers, six N-digits are enough to specify the register unambiguously. The seventh (most significant) N-digit is used to specify to which order of the pair the jump is to be made. Normally the machine will jump to the first order of the pair: to specify the second order, a + is written after the register address in the programme, and the corresponding character is punched on the tape. The input routine then interprets this as a 0 in the most significant place of the N-digits in the jump order.

23. The 'jump' functions are as follows:

6 0 Jump to order specified if number in specified accumulator is zero, (N.B. the order 'jump in any case' can be given by specifying accumulator 0.)

6 1 Jump if number in accumulator is not zero.

6 2 Jump if number in accumulator is zero or positive.

6 3 Jump if number in accumulator is negative.

6 4 Jump if overflow staticiser (see Chapter VII) is clear, or clear overflow staticiser.

6 5 Jump if overflow staticiser is set, and clear overflow staticiser.

6 6 Unit modify. Add 1 to modifier (most significant 13 digits) in specified accumulator, and jump if result is not exactly divisible by 8, i.e. if last 3 binary digits are not all zero.

6 7 Unit count. Subtract 1 from counter (least significant 25 digits) in specified accumulator and jump if the result is non-zero.

24. Transfers between the main store and the computing store can be done in single words or in blocks of eight words. Single words are always transferred to or from accumulator 1. In single-word transfer orders, the N-digits specify the main-store block containing the word, and the X-digits the position of the word in the block. In block-transfer orders, the N-digits again specify a block in the main store; but the X-digits specify a block of registers in the computing store. As a special facility, the machine has been designed to interpret block 7 of the computing store, in *block-transfer orders only*, as the block of accumulators. To specify without ambiguity any one of the 640 blocks in the main store would require up to ten N-digits. As there are only seven N-digits in an order, the most significant three digits in the block address must be supplied, if needed, by modification (see below). The main-store-transfer functions (group 7) are:

7 0 Transfer word specified by X-digits in main-store block specified by N-digits to accumulator 1.

7 1 Transfer contents of accumulator 1 to position specified by X-digits in main-store block specified by N-digits.

7 2 Transfer the eight words in main-store block specified by N-digits to the corresponding positions in register block specified by X-digits.

7 3 Transfer the eight words in register block specified by X-digits to main-store block specified by N-digits.

25. The other functions used in group 7 are the 'external conditioning' function, the 'interchange' function and the 'stop' function. The 'stop' function is self-explanatory.

7 7 Stop.

External-conditioning orders enable the machine to be switched rapidly between different input or output devices. Each of the seven N-digits in the order is associated with a

set of relays, which can be connected in the main control circuit of one of the input or output mechanisms. Thus the external-conditioning function is interpreted as:

7 4 Switch on the relays associated with the non-zero N-digits in the order.

The 'interchange' function enables a block of information to be exchanged between the computing store and a 'buffer' store in an input/output device such as punched-card equipment or a magnetic-tape unit. The X-digits of the order specify a block in the Pegasus computing store. The precise interpretation of the N-digits depends on the type of input/output device used. The 'interchange' function is therefore:

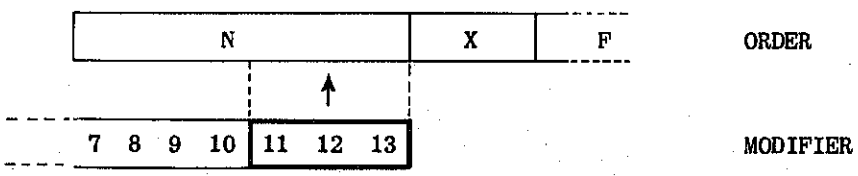
7 6 Exchange the contents of the computing-store block specified by the X-digits for the contents of a buffer storage block indicated by the N-digits in the input/output device.

The full function code is summarised in fig. 4.2.

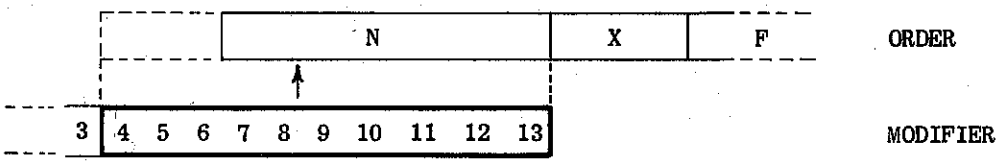
Modification

26. The last three digits of an order, the M-digits, are interpreted by the machine as the address of an accumulator, at the more significant end of which has been placed a binary number called the modifier. The modifier is 13 digits long and occupies digit places 1 to 13 of the word. (The less significant end of the same accumulator can be used for a counter, which may therefore have up to 25 binary digits). The modifier is automatically added to the order at the more significant end before the order is obeyed. The precise way in which this is done depends on the function; but the effect of modification is generally to change the addresses in the order, so that the order, when obeyed, refers to a different part of the store. The facility for modifying orders simplifies programming and speeds up the operation of the machine, as a single order can, by successive modifications, be made to perform the same operation on several numbers in different parts of the store. In unmodified orders, the M-digits will be zero: this is one of the reasons why it is convenient to have accumulator 0 permanently empty.

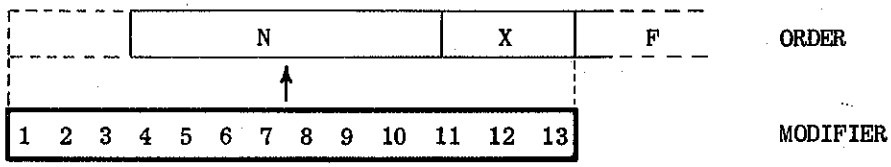
27. The digits in a modifier are usually treated by the machine in two groups, viz. the least significant three digits, and the most significant ten digits. Thus the machine may be said to interpret the modifier as an octal number, i.e. as number of eights plus a number of units, a 'block' number and a 'position' number within a block. In all arithmetical orders (function groups 0, 1 and 2) only the least significant three digits ('position' digits) of the modifier are used. These are added to the N-address (register address) in the order thus:



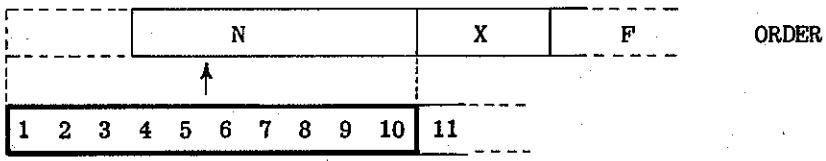
In counter, shift and jump orders (function groups 4, 5 and 6) the least significant ten digits of the modifier are added to the N-digits of the order, possibly extending the length of the order by up to three digits, thus:



28. In single-word transfer orders (functions 70 and 71), the whole modifier is used, the least significant three digits of the modifier being added to the X-digits of the order, and the rest of the modifier ('block' digits) to the N-digits of the order, again possibly extending the order length, thus:



In block-transfer orders (functions 72 and 73) the least significant three digits in the modifier are not used; the remainder (the 'block' number) are added to the N-digits (drum-block address) in the order thus:



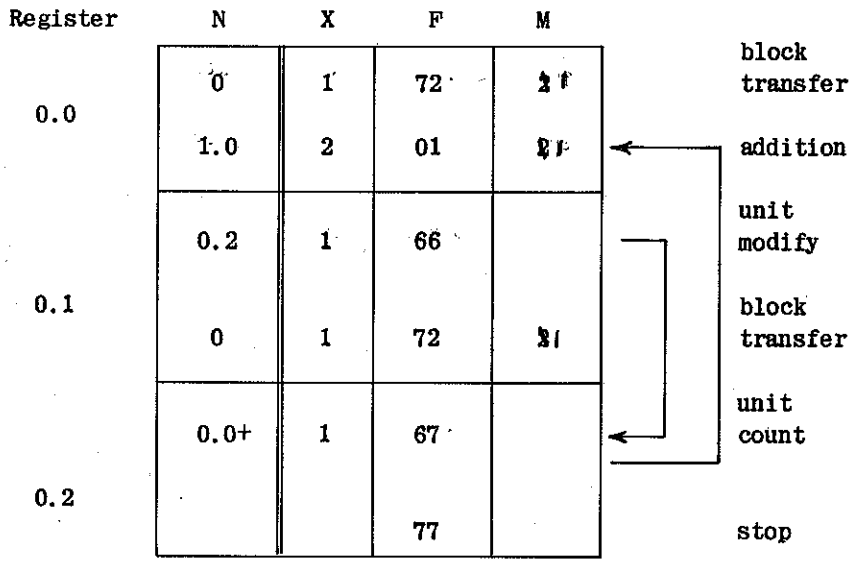
EXAMPLE

29. As an example of the use of orders and modifiers, consider the summation of twenty binary numbers stored in addresses 4002 to 4021 (block 500, word 2, to block 502, word 5) in the main store. The modifier required in this case is 4002 (block 500, position 2), which will be used in conjunction with a counter of value 20, both having been placed previously in one of the accumulators, say accumulator 1. Let the orders concerned with the summation be stored in block 0 of the computing store, and let block transfers from the main store be made to block 1 of the computing store. Finally let the summation be done into accumulator 2, which will be empty initially.

30. Two main types of order are required in this problem: block-transfer orders bringing a block of words from the main store into block 1 of the computing store, and addition orders putting the sums into accumulator 2. It is not necessary to write a separate order for every step in the problem, as modification will ensure that the interpretation of a general order is correct at any stage in the process. The 'unit modify' jump order (function 66) will ensure that the modifier is increased by 1 after

each addition, and will ensure that a new block is brought out of the main store when all the numbers in the previous block have been 'used up'. A 'unit count' jump order (function 67) will test when the process is finished and will allow control to pass to a 'stop' order or to an output routine.

31. The routine for the summation of twenty consecutively stored numbers consists of three order pairs stored and written as follows



32. The first order in the first pair is a block-transfer order, which, after modification by the block number in the modifier, will be interpreted as

500	1	72	

'Transfer the contents of block 500 in the main store to block 1 (registers 1.0 to 1.7) in the computing store'.

The second order in the first pair is an addition order. It is modified by the units (last three digits) in the modifier, and will be interpreted as

1.2	2	01	

'Add the contents of register 1.2 into accumulator 2.'

Next comes a unit-modify order. This adds 1 to the modifier in accumulator 1, making the modifier 4003, and, finding that the modifier is *not* divisible by 8, transfers

control to the first order in register 0.2, which is a unit-count order. The counter is diminished by 1, showing that the first stage of the operation is complete, and control jumps to the second order in register 0.0.

33. At this stage, the counter is 19 and the modifier is 4003 (block 500, position 3). The second order in register 0.0. is therefore interpreted as

1.3	2	01	

'Add the contents of register 1.3 into accumulator 2'.

Thus a 'loop' has been set up by the use of the unit-count and unit-modify jump orders. which ensures that the orders 0.0+, 0.1 and 0.2 are obeyed cyclically. The modifier is increased by 1 each cycle until, when the last number in block 1 of the registers has been added into the accumulator, its value is 4008, which is divisible by 8. At this point, the jump does not occur, but control passes to the block-transfer-order in 0.1+. This order, after modification by 4008 (501 eights), is interpreted as

501	1	72	

'Transfer the contents of block 501 in the main store
into block 1 of the computing store'.

34. After the block transfer, control passes to the unit-count order (15 - 1 = 14) and then jumps back to the addition order. The loop consisting of addition, unit modify and unit count is then repeated eight more times, until the modifier is once more divisible by eight (4016). Then, as before, the block-transfer order comes into operation to bring from the main store into the computing store the remainder of the numbers to be added. This time, the modifier makes the block-transfer order into

502	1	72	

The next order, the unit-count order, reduces the counter from 7 to 6, and control jumps back to the start of the loop. This time, the loop is repeated only six times, After the sixth addition, the counter is diminished by 1 to zero, showing that the process is finished. There is no jump back to the addition order; but control passes straight on to order 0.2+, which stops the machine.

CHAPTER V

TIMING AND RHYTHM

CHAPTER V

TIMING AND RHYTHM

'CLOCK'

1. The digit frequency and the word frequency (which is 1/42 of the digit frequency) in Pegasus are normally locked to the rate of rotation of the magnetic-drum store, which is servo-controlled with reference to a crystal oscillator. The timing of digit pulses is fixed by the 'clock' track on the drum, which gives a sinusoidal output of period approximately 3μ sec.; this is shaped by special packaged circuits into a square wave of unity duty factor - the 'clock' waveform. While testing, it is sometimes necessary to produce the clock waveform direct from the crystal oscillator. A switch on the engineer's panel enables the change-over to be effected. All digit pulses are 'clocked up' on entering single-digit delays or nickel lines, i.e. they undergo an AND operation with the clock waveform, to ensure that any change in their timing due to stray circuit time constants is automatically corrected at intervals in their travel.

p-PULSES

2. Word-frequency timing is referred to a sequence of p-pulses, which recur once in every 42 digit-times. These are controlled by a 'commutator', which is essentially a pulsed frequency divider counting down by a factor of 42. As an introduction to its operation, consider the scale-of-two and scale-of-three counters illustrated in fig. 5.1. Suppose that the output of the scale-of-two counter is initially zero; the input to the inverter is then zero, so that the input to the delay is positive. One digit-time later, the output of the delay will be positive and its input zero. Thus the scale-of-two counter will generate pulses and spaces alternately. In the scale-of-three counter, two delays are used, whose 'mix' outputs are returned together to the input of the inverter. A pulse travelling through the two delays will hold the inverter input positive, and consequently the input to the first delay zero, for two digit-times. There will therefore be two spaces for every digit in the output, i.e. an output pulse will appear once every three digit-times. Note that an output taken from the first delay would also give one pulse in three; but this pulse would be one digit-time earlier than the one in the output of the second delay.

3. A 42-digit commutator could obviously be made as a straightforward scale-of-42 counter using 41 delays and an inverter, with the 'mix' outputs of all the delays taken together to the inverter input. Loading considerations would, of course, require the mixing to be done in several stages. However, the same result can be achieved with less equipment. The prime factors of 42 are 7, 3 and 2. Three counters producing pulses at intervals of 7, 3 or 2 digit-times would therefore produce a pulse simultaneously once every 42 digit-times. Fig. 5.1 shows a suitable basic arrangement. The outputs of the three counters are fed to an AND gate, whose output is therefore one pulse every word-time.

Commutator

4. The commutator circuit used in Pegasus is normally locked to the rate of rotation of the drum by means of the waveform picked up from the address track. When the engineer's 'clock' switch is set to 'crystal' during testing, the commutator is free-running (except in so far as it is controlled by 'clocking-up' at the inputs to all delay elements). The complete commutator circuit is shown in fig. 5.1, with package references marked on all the logical symbols. Assume first that the 'clock' switch is set to 'crystal' so that the input to cathode-follower 12F5, and consequently to gate x of delays 12X1 and 12S2, and to gate y of delay 12S1, is permanently positive. The three counters will then cycle in the normal way, producing one pulse every 7, 3 and 2 digit-times respectively. Once every 42 digit-times, the three inputs to the AND gate preceding delay 12U3 will all be positive, so that a pulse will emerge from delay 12U3 one digit-time later. The pulses circulating in the three counters will enter either through the upper AND gates of the twin delays or through the lower gates, according as the input to inverter 12T2 is positive or negative.

5. Consider now what will happen if the lines from the three twin delays to cathode-follower 12F5 and AND gate 20P4 are broken, and are connected instead to a negative voltage supply so that gate x of 12X1 and 12S2 and gate y of 12S1 are kept permanently closed. One digit-time after coincidence has been found between the counter outputs, when the three counters are about to start a new cycle, the output of delay 12U3 will be positive; consequently, the output of inverter 12T2 will be negative. This means that the lower gates to the twin delays will be closed too, so that the three counters will have to wait one digit-time before they can begin their next cycle. Under this condition, the commutator will produce a pulse not every 42, but every 43, digit-times.

6. The so-called address 'track' like the other information 'tracks' on the drum is, in fact, a track pair, alternate digits being written on alternate tracks. The magnetisation pattern corresponds to the original pulse waveform according to a system of phase modulation, which will be explained more fully in Chapter XI (see fig. 11.6). For the present, it will be enough merely to state an important consequence of this, that a pulse is derived from the two tracks simultaneously only where a pulse was immediately followed by a gap in the original address waveform. To enable the commutator to be locked word-by-word to the drum, it is arranged that once only the digital combination '1 0' shall appear in the same position in *every* word space on the address track.

7. The waveforms picked up from the address-track pair, H8 and H9, are fed to AND gate 20P4, which will therefore give a positive output once every word-time, when the digit combination 1 0 is read off the track. The digit-time when coincidence occurs (which is the digit-time properly ascribed to the 'one', not to the 'nought') is the starting time of the commutator and thus defines $\phi 0$ for the machine. When switched on, the commutator may start at any instant; if, however, at the end of one 42-digit cycle it finds that the output of AND gate 20P4 is negative, it will wait one digit-time before beginning the next cycle, as explained in paragraph 5. Thus it will 'slip' one digit-time every word time until its instant of starting coincided with $\phi 0$. This will take, at most, 43 word-times, or about 5 milliseconds. Once it has 'locked on', the commutator will cycle with a period of 42 digit-times. Notice how signals H8 and H9 enter into an AND operation with +13 V; this is to standardise the level of the AND-gate output.

ϕ-Pulse Generator

8. A pulse can be generated at a given instant in each word by feeding an AND gate from three correctly chosen points in the commutator. If the locking pulse from the address track defines $\phi 0$, output T1 in fig. 5.1 will contain a pulse every seventh digit-time starting at $\phi 2$, and output T2 will give a pulse every third digit-time starting at $\phi 1$. Output T3 will give all the odd-numbered ϕ-pulses, and output T4 the even-numbered ϕ-pulses. There will be coincidence between T1 and T2 every 21 digit-times, *viz.* at $\phi 16$ and $\phi 37$; consequently T1, T2 and T4 will be coincident only at $\phi 16$. T1, T2 and T4 are used as inputs to a delay (12W2 in fig. 5.1) whose output is therefore $\phi 17$.

9. $\phi 17$ is used as the input to the chain of delays shown in fig. 5.2, which generates ϕ-pulses for use throughout the computer. ϕ-pulses are available at all digit-times except at $\phi 13$, $\phi 14$, $\phi 15$ and $\phi 16$. Loading considerations dictate the use of cathode-followers in all heavily loaded lines. The convention here is to represent ϕ-pulses by a number alone; those that are eventually to be 'repeated' by cathode-followers carry a subscript $_0$, and the outputs of the various cathode-followers carry subscript numbers to distinguish them one from another. The origins of repeated waveforms not shown in fig. 5.2 are given in tabular form in fig. 5.3. The computer also requires inverted ϕ-pulses; the origins of these, and the cathode-follower repeaters used, are also shown in fig. 5.3. The inverse of a ϕ-pulse is, of course, a word length of pulses with a gap in the appropriate position. Note the use sometimes of OR gates and AND gates as repeaters (two of the input crystals being used in parallel as a safety measure), and of single delays (the input being then the preceding ϕ-pulse).

10. Bay-to-bay interconnectors, where applicable, are indicated on diagrams by small circles. The interconnector reference numbers give the row, the position in the row (A or Z), and, after an oblique mark, the pin number, Z-connectors (in general) are coupled pin-for-pin to the A-connector in the adjacent bay. Some of the A-connectors in bays 1 and 2 lead direct to the desk panels.

11. Fig. 5.2 also shows how the commutator can be monitored, $\phi 17$ and $\phi 12$, from the extreme ends of the delay line, being applied to an OR gate, whose output should be two pulses in each word-time. The number at the monitor point refers to a wafer and pin on selector switch SW A, the larger of the two switches mounted on the bulkhead between bays 2 and 3. It will be noticed that the 'mix' outputs of several elements in fig. 5.2 are used. V22, equivalent to $\phi 18$, is used when an order is taken from the handkeys (see Chapter IX). V4, equivalent to $\phi 1$, is used for carry suppression in the adder-subtractor in the Mill (Chapter VII). Timing waveform T39 is the 'mix' of ϕ-pulses 8, 9, 19, 24, 26, 27, 28 and 41.

'Start' Order Pair

12. Waveform T39 is the 'start' order pair, which is fed into the order register of the machine when the 'start' switch on the programmer's panel is at START. It will be shown in Chapter IX that the machine senses the pulses in T39 as if they had been delayed by one digit time on standard timing, i.e. as if it had been presented with an order pair which, in standard timing, had positive pulses ('ones') at $\phi 7$, $\phi 8$, $\phi 18$,

$p23$, $p25$, $p26$, $p27$ and $p40$. Now p-pulses can be expressed as digit pulses in standard timing by the formula

$$p + d = 38.$$

where d is the digit position according to the convention given in paragraph 3 of Chapter III. Consequently waveform T39 is interpreted by the machine as an order pair with 'ones' in digit positions 31, 30, 20, 15, 13, 12, 11, -2; or, in descending order of significance

-2 11 12 13 15 20 30 31

13. The A-order of a pair occupies the digit places up to 19; digit-place -2 is that of the most significant 'modifier-extension' digit; places 11, 12 and 13 are occupied by the first three F-digits, and place 15 by the fifth F-digit. The four sections of the A-order of the 'start' pair are then, as binary numbers

N	1 0 0 0 0 0 0 0 0
X	0 0 0
F	1 1 1 0 1 0
M	0 0 0

and the order can be written

512	0	72	0

'Transfer the contents of block 512 in the main store to block 0 in the computing store'.

Block 512 contains the first eight words of the permanent Initial Orders.

14. The second order of the 'start' pair contains positive digits in places 20, 30 and 31. Its four sections, in binary notation are then

N	1 0 0 0 0 0 0
X	0 0 0
F	1 1 0 0 0 0
M	0 0 0

so that the order can be written

0.0	0	60	0

'Jump if the contents of accumulator 0 are zero (which is always true) to the first order in register 0.0 (i.e. to the first order of the block just transferred)'.

p-Pulse Groups

15. Groups of consecutive p-pulses are required in various parts of the machine for opening gates. The usual way of generating these is to use staticisers, which are set and reset by appropriate p-pulses. Six such staticisers are shown in fig. 5.4; notice that the first pulse in the output is delayed one digit-time on the setting pulse, and that the last pulse in the output is coincident with the resetting pulse. A group of three p-pulses can be produced in an OR gate as shown at the top centre of fig. 5.4. Long consecutive groups can be generated as shown at the top right of the figure, by means of inverted p-pulses. Fig. 5.4 also shows the generation of various timing waveforms for the multiplier-divider.

RHYTHM

16. Orders are obeyed in Pegasus in a series of beats, which last an integral number of word-times each. The basic rhythm is made up from the three beats, A, B and C, in which an order pair is obeyed. The order pair is picked up from the computing store in beat C and transferred to the order register, in which it circulates during beats A and B. The circulation gate is closed during the next C-beat so that the old pair is erased as the new order pair is being read in. Orders are not necessarily lost after they have been obeyed. It must be understood that an order pair is merely *copied* into the order register from the store; it continues to circulate in store until it is replaced by information from another part of the store as a result of a transfer order. After the order pair has entered the order register, the first order of the pair (the one at the more significant end) is obeyed during the A-beat. The duration of the A-beat depends on the order being obeyed, but is always an integral number of word-times. The second order of the pair is obeyed during the B-beat, which, like the A-beat, lasts a number of word-times depending on the order.

Three-Beat Counter

17. Fig. 5.5 shows the beat counter that controls the basic A-B-C rhythm. It consists essentially of two staticisers, 24M1 and 24M2. As will be shown, these two cannot be active together except, perhaps, in the transient condition of the machine when it is first switched on. When 24M1 is active, the machine is in the A-state, i.e. output A is a train of positive pulses; when 24M2 is active, the machine is in the B-state and output B is positive; when neither staticiser is active, the machine is in the C-state, and output C, the output of inverter 24P2, is positive. One of these three conditions must always obtain.

18. Assume initially that the circuit is in the C-state. The two input gates to 24M2 will be closed; one because A is negative the other because B is negative. However, the output of inverter 24P1 is positive, as one of the two inputs to its AND gate is zero; consequently a pulse on the beat 'trigger' line (X80) can pass through delay 24M1 via gate x. Unless the machine is in the 'start' condition, waveform X18 is positive and the pulse can circulate in 24M1 via gate y. The circuit is now in the A-state, and output ~C, which is one of the inputs to inverter 24P1, is positive. The next trigger pulse will therefore make the output of this inverter zero, which will

cut off the A-staticiser by closing gate *y*. If the terminal marked ~J is positive, however, this trigger pulse will activate staticiser 24M2, since the A-waveform will remain positive until one digit-time after the trigger. Also one digit-time after the trigger, the output of inverter 24P1 will become positive again, opening the gate *y* of delay 24M2 so that the pulse can circulate through it.

19. When staticiser 24M2 is active, and the circuit is in the B-state, one of the inputs to inverter 24P1 is again positive. The appearance of another trigger pulse will therefore make the output of 24P1 negative, which will stop the B-staticiser. Moreover, as it will prevent the A-staticiser from being triggered, the circuit will return to the condition in which neither staticiser is active - the C-state. Thus trigger pulses to this circuit will, as long as the machine is not in the 'start' condition and as long as ~J is positive, change the state of the circuit in turn from C to A, from A to B or from B to C. Input X18 from the start key is zero only when the key is moved to START. Circulation in the A and B staticisers is then inhibited, ensuring that the circuit starts in the C-state even if A or B has been set by spurious pulses during warming up. The input marked ~J is positive

(i) in all orders except jump orders
(ii) in jump orders when the condition stipulated for a jump is *not* satisfied.

This input is provided to ensure that the B-beat is omitted when the machine is required to jump from the first order in a pair (the A order) to another order pair. Waveforms A and C are taken, via 'output-one' elements to neon indicator lamps. These are used when the machine has stopped - either because of a fault or because of a stop inserted in the programme - to indicate in which part of the order pair the stop occurred (see Chapter XVII).

D-Waveform

20. The C-beat, during which the order pair is copied into the order register, lasts one word-time only. Beats A and B last at least two word-times each. In general in the first word-time, referred to as word-time AD or BD, the operands are routed out of the store into the part of the machine where the operation is to take place; in the last word-time, word-time AE or BE, the result of the operation is routed back to store. The circuit used for generating the D-waveform during the first word-time of each beat is shown in fig. 5.5. It is simply a staticiser, 24T1, which is triggered by X80, the waveform that triggers the main beat counter. Staticiser 24T1 is reset by ϕ-pulse 40; as will be shown, X80 is also a pulse at ϕ40, so the D-waveform lasts one word-time only. (Because its generator incorporates a delay, the D-waveform lasts from ϕ41 to ϕ40 in the next commutator cycle). It should be noticed that D will be positive during the whole of the C-beat, which can therefore be referred to alternatively as word-time CD.

E-Waveform

21. The E-waveform, which marks the last word-time in a beat, is obtained from staticiser 24V1. This is set and reset at ϕ40, so that word-time E, like word-time D, runs from ϕ41 to ϕ40. For arithmetical-transfer orders, counter orders and jump orders, i.e. for orders containing functions in groups, 0, 1, 4 and 6, word-time E must follow

immediately after word-time D. These functions are defined by the signals G00, G01, G04 and G06 (see Chapter VI), which are mixed in OR gate 22U6. To ensure that AE and BE follow AD and BD and cannot occur at any other time for these functions, the priming waveform for the staticiser is obtained by an AND operation between the appropriate function signal and a waveform (D & ~C), which is obtained from AND gate 24S1. There is no word-time CE; hence, although G01, G04 and G06 cannot be positive during C, it is necessary to use ~C as a component of the triggering waveform because G00, representing function group 0, can be positive at any time. The D-component of the triggering waveform is necessary, in spite of the waveform ~E fed to the lower gate of 24V1, to prevent the E-staticiser from being triggered at the same time as the D-staticiser in the special case when the machine has been stopped between AE and BD.

22. The only other orders in which word-time E must follow immediately after word-time D are 'justify' orders (function 23), the 'stop' order (function 77) and, in cases of programme error, the unallocated functions. Functions 23 and 77 are defined by waveforms G02 & G13 and G07 & G17 respectively, which undergo AND operations with D in the gates preceding delay 22V1, and provide another possible trigger for the E-staticiser. There is a special circuit in the multiplier-divider for controlling the timing of multiplication and division orders. Generally word-time E follows word-time K of the multiplier; the exception to this is function 22 (waveforms G02 and G12) in which an extra word time, L, is required between the end of K and the beginning of E. K and the inverse of (G02 & G12), combined in gate *y* of 24X1, form another possible trigger for the E-staticiser.

23. For function-22 orders, the E-staticiser must be set after word-time L in the multiplier-divider; L is therefore one of the components of V3, which primes the E-staticiser. However, L and E are positive together during the last word-time of the other multiplication and division orders; consequently gate *x* of staticiser 24V1 must be controlled by ~E to prevent the staticiser from being set a second time at the end of L after having already been set at the end of K. The other components of V3 are signals indicating that a shift order has been completed, that a drum transfer or interchange is complete or that the external-conditioning relays have been set up. It is arranged for an interval equivalent to between two and three drum revolutions to elapse between D and E of an external-conditioning order to give the relay contacts time to change over.

24. Cathode-follower repeaters for the rhythmic waveforms and their inverses are tabulated in fig. 6.9. Fig. 6.9. shows too the connections to the monitor selector switch.

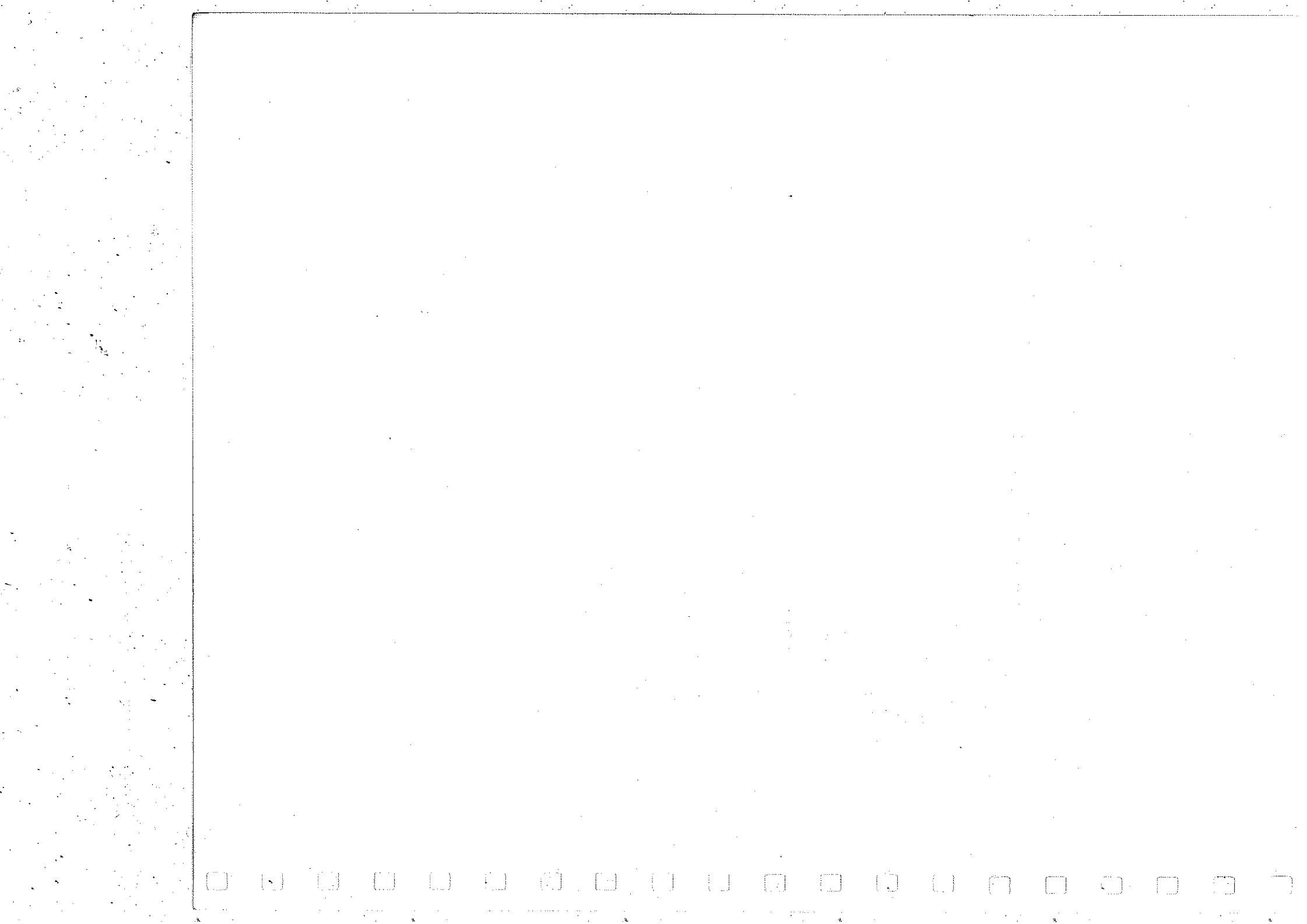
BEAT TRIGGER (X80)

25. The trigger waveform, X80, for the beat counter is generated in AND gate 24S2. Provided that the STOP signal, V24, is negative, X80 is produced by coincidence between ϕ40 and either C or the output of staticiser 24T2. This staticiser is set by E and reset by D; it will therefore cause X80 to be produced at ϕ40 of word-time E unless the STOP signal is positive, in which case until the STOP line is cleared it will

'remember' that the machine was in the E-state. The generation of X80 is inhibited by the STOP signal (V24), the effect of which in turn is inhibited by B or J. $\sim(B \vee J)$ incorporated to assist the special facility of the machine called 'single-shot working' (see Chapter IX), whereby at the depression of a switch the machine obeys a single order and then stops. The method of generating X80 ensures that the machine cannot stop in the B-state, but must continue to the end of the C-beat, to pick up the next order pair from store. Neither can the machine stop in the A-state when the A-order requires a jump; it must continue into the C-state and pick up the new order pair.

CHAPTER VI

THE ORDER REGISTER AND DECODING



CHAPTER VI

THE ORDER REGISTER AND DECODING

THE ORDER REGISTER

1. The order register is essentially a closed delay loop in which an order pair can circulate while it is being obeyed; it also provides a means of separating the digits in an order so that groups of them can be sensed simultaneously for the purposes of decoding. The order pair is fed into the order register during C. The order at the more significant end of the word is obeyed first; this happens during the A-beat, which lasts a number of word-times depending on the order. During the last word-time (AE) of the A-beat, circulation in the order register takes a shorter path, which has the effect of deleting the A-order and 'shuffling' the B-order up towards the more significant end of the word to give it the timing of the A-order. This is done to simplify the procedure for decoding. The B-order circulates normally during the B-beat; in the last word-time (BE) of this beat, it is replaced in the order register by the order number, which gives the address of the next order to be obeyed.

2. The order register is shown in fig. 6.1. It gives the following facilities:-

(i) temporary storage for the order pair being obeyed,
(ii) a means of separating the digits of an order for decoding,
(iii) a method of 'shuffling' the B-order into the position of the A-order,
(iv) a means of separating the digits of the order number for decoding,
(v) an adder for modifying orders, adding unity to the order number, or counting shifts.

3. The order adder is a full adder of the type normally used in Pegasus, and has already been described in Chapter III (fig. 3.3). Provision is made for closing the gates to the 'carry' delay (25T1) at $p0$; this is to suppress the 'carry' digit produced in shift counting. There are seven possible '*b*' inputs to the adder and four possible '*a*' inputs. Gate *x* of twin delay 25J1 is used for normal circulation. A 35-digit line (25Q) is inserted in the normal-circulation loop so that, with delays of one digit-time in the adder and after the input gate, the total circulation time along this path is 42 digit-times, or one word-time. The shorter loop used for 'shuffling' is 19 digit-times long. The digits in an order can be sensed at any one of eighteen points (I0 to I17) after the adder, or at one point (W1) before the adder. Outputs from some of these points are fed to the decoding circuits.

4. Timing Chart 3 gives the timing of the various waveforms controlling gates to the order register and the decoding circuits. For the purposes of this chapter, it is assumed that orders requiring only two word-times are being obeyed and that the controlling handswitches are in the position for normal running. The order-register

operations involved in starting and stopping the machine and in manual working are discussed in Chapter IX. The operation of the order register is further complicated by the facilities it provides for 'jump' and 'shift' orders. These are discussed fully in Chapters VIII and XVI respectively, and will be referred to here only in passing.

5. The order pair enters the order register from the N-bus, the output line common to all parts of the computing store, through gate *x* of twin delay 25J2 (refer now to fig. 6.2). At this point its timing is delayed one digit-time on standard timing, and the less significant end of the word enters first; consequently the timing of the B-order on entry is $p1$ to $p19$, and that of the A-order is $p20$ to $p38$. The input gate is controlled by two waveforms. Except in some jump orders, one of these, X15, is positive from $p1$ of CD to $p41$ of AD. (Note that D runs from $p41$ to $p40$ in the next commutator cycle.) The other control waveform, is positive from $p1$ to $p38$ of every word-time provided that the 'start' key is not at MANUAL. Thus the order pair enters delay 25J2 between $p1$ and $p38$ of CD. The stop-go digit, which appears on the N-bus at $p39$, and the 'parity' digit, which appears at $p40$, do not enter the order register; their places can thus be occupied by new digits produced by modification.

6. The gate allowing entry to the order register for normal circulation, gate *x* of 25J1, is controlled by two waveforms. One of these, U16, the inverse of CD delayed one digit-time, ensures that the circulation gate is closed during CD while the new order pair is being fed into the order register. The other control waveform on the circulation gate, X21, is simply the inverse of all the other waveforms that control gates leading to the *'b'* input of the adder; it is positive from the beginning of CD to $p19$ of AE, and again from $p0$ of BD to $p41$ of BE (see Timing Chart 3), provided that the handswitch is not set to START or MANUAL, and that a jump is not called for. Thus normal circulation can occur only during beats AD and BD and the first half of AE.

7. The 'shuffle' loop is connected to the order register through gate *y* of 25J2, which is controlled by timing pulses $p1$ to $p38$ (see fig. 5.4) and by X19, the output of staticiser 25D1. This staticiser is set at $p19$ of AE if a jump is not called for (i.e. if ~J is positive) and is reset at $p0$; consequently gate *y* of 25J2 is open from $p20$ to $p38$ in AE. The B-order, which had timing $p1$ to $p19$ on first entering the register, has timing $p20$ to $p38$ on the 'shuffle' loop; this is the same as the timing of the A-order on first entering the register.

8. Nickel-line register 25P contains the order number, which gives the address of the order-pair currently being obeyed. As all order-pairs are taken from ordinary registers, the order number must be six digits long - three to specify the block and three to specify the position in the block. Order-number control is explained fully in paragraphs 25 to 27 below. The content of the order register or of the order-number register can be studied on the engineer's monitor display or on the programmer's monitor display. The contents of these two registers make up the only information available to the programmer's display, the selection between the two being made by a switch on the programmer's panel, as shown in fig. 6.2.

DECODING

9. The first part of an order to enter the order register, and consequently the first to be decoded, is the group of M-digits. The circuit used for M-decoding is

also used for X-decoding, and will be described later. The next digits to be decoded are the F-digits. These determine the operation that the machine is to perform, and also dictate how the N-digits and X-digits are to be interpreted and how the modification is to be carried out. To determine the type of modification, the F-digits are sensed during CD (for the A-order) or during AE (for the B-order) as explained in paragraph 27.30/32. The normal decoding of the F-digits occurs during AD (A-order) or BD (B-order). Fig. 6.3 shows how the digits of an order are distributed in the order register at different times. It will be seen that the M-digits of the A-order (or of the B-order after the shuffle) enter (or re-enter) the order register from ϕ20 to ϕ22, the F-digits from ϕ23 to ϕ28, the X-digits from ϕ29 to ϕ31 and N-digits from ϕ32 to ϕ38. At ϕ41, the F-digits are distributed between outputs I11 and I16. A more detailed version of fig. 6.3, showing all possible applications of the order-register outputs appears as Timing Chart 5.

F-Decoding

10. The F-decoding circuit is shown in fig. 6.4. It comprises six staticisers, one for each F-digit; these are set at the beginning (ϕ41) of AD and BD, the setting waveform being D41 inhibited by C, and have inputs I11 to I16. They therefore 'remember' the digits present at I11 to I16 at ϕ41 - the F-digits - until they are reset at ϕ39, nearly two word-times later. The reset waveform also depends on the 'not start' waveform, X18, which is negative when the handswitch is set to START. Thus the staticisers are always cleared at the start of a programme.

11. Because of the delay of one digit-time in the staticisers, their outputs will be significant from ϕ0 pf D to ϕ39 of E. If the original F-digit was a 'one', the direct output of the corresponding staticiser will be positive: if the original digit was a 'nought', the inverted output will be positive. Thus for any given configuration of F-digits there will be only one combination of the six outputs or their inverses such that all will be positive. Thus function 53, which in binary coded form is

1 0 1 0 1 1

will give positive outputs at

F0 ~F1 F2 ~F3 F4 F5

12. The F-outputs and their inverses are combined in threes in AND gates (followed by delays to reshape the pulses) as shown in fig. 6.4. The resulting G-outputs will of course be significant one digit-time later than the F-outputs, i.e. from D1 to E40. G00 to G07 correspond to the first three F-digits, i.e. to the function group, the first digit of the function in programmer's octal notation; G10 to G17 correspond to the last three F-digits - the second digit in programmer's notation. Thus function 53 appearing in an order would make only G05 and G13 positive, all the other G-waveforms being zero. Both F-waveforms and G waveforms are used to control the operation of the machine, and the order code has been arranged so that the loading on these waveforms is reduced to a minimum. Thus G01 can be used alone to control the many operations that are common to all functions in group 1; ~F0 will control what is common to functions in the first half of the order code, or, a function group having been specified

by a G-waveform, ~F5, will control gates relating to all the even-numbered functions in the group. Waveform G03 corresponds to unallocated function group 3; it is used to set a staticiser, whose output stops the machine (see Chapter IX and fig. 9.1). Cathode-follower repeaters for the F-waveforms and G-waveforms are listed in fig. 6.9.

13. It should be noticed that, while the F-staticisers are inactive, all the inverted outputs will be positive, making G00 and G10 positive; the machine will then be in a condition for carrying out function 00 - transfer from a register to an accumulator, This will occur during stops, during CD and during the first two-digit-times in beats A and B. However, no transfer can take place, since the entries to the store lines are open for transfers only during E.

X/M Decoding

14. The M-digits in an order are decoded during CD and AE, whereas the X-digits are decoded during AD and BD; consequently, as there are the same number of X-digits as M-digits, and since the X-digits generally, and the M-digits always, refer to accumulator addresses, the same decoding circuit can conveniently be used for the two groups. This circuit is shown in fig. 6.5. As in the F-decoding, the digits are 'remembered' by staticisers; but these are fed not direct from the order register itself but from a subsidiary chain of three digit delays - an arrangement that is necessary for the M-decoding so that the digits can be sensed in advance of the adder in the order register. The X-digits are tapped off the order register at I7, where they appear from $p38$ to $p40$ (see fig. 6.3). They enter the decoding circuit through gate x of twin delay 26Y1, and at $p41$ they are distributed along the inputs to the three staticisers.

15. The setting waveform for X-decoding (D & ~C & 41) is essentially the same as that used for setting the F-staticisers (see fig. 6.4); so the outputs S8, S9 and S10 and their inverses become significant for X at the same instant as the F-outputs, at $p0$ of AD or BD. However, the X/M-staticisers are reset by $p39$; so their outputs will be significant for just less than one word-time. As in the F-decoding, a 'one' will give a positive direct output, and a 'nought' will give a positive inverted output. S8 or ~S8 corresponds to the most significant X-digit; thus, for example, X-digits 011 would give positive outputs at ~S8, S9 and S10. The X-digits generally specify an accumulator address: if the function requires that they be given some other interpretation, they are decoded elsewhere in the machine; a casual decoding is then carried out in the circuit of fig. 6.5, but the three S-outputs are not allowed to influence the operation.

16. The M-digits are sensed from the order register at W1, just before the order adder, where they appear from $p21$ to $p23$. They are gated into the decoding circuit between $p3$ and $p27$ of CD or AE and they are distributed along the staticiser inputs at $p24$ when the setting pulse arrives from AND gate 24Y3. The use of ~J ensures that the M-digits of the B-order are not decoded when this order is to be 'jumped over'. The staticisers remain active until $p39$; so their outputs are significant for less than half a word-time for M-decoding; this is long enough to allow the thirteen-digit modifier to be routed into the order register from the accumulator where it is stored. It will be noticed that the timing waveforms on the two gates preceding twin delay 26Y1 ensure that digits cannot enter the decoding circuit through the two simultaneously.

N-Decoding

17. For most orders, the operands are routed out of store during D, and the result of the operation is routed back to store during E. The N-decoding and the X-decoding must both operate during D for routing out; but only one decoding circuit need be used during E, as there is only one address for replacement. Depending on the function in the order, the replacement address may be given by the N-digits or the X-digits; consequently the decoding circuit used for replacement must be able to operate on either set of digits in the order. As the N-digits may specify any address in the computing store - ordinary register, special register or accumulator - the N-decoding circuit alone gives all the facilities necessary for decoding the replacement address, only the last three staticisers being used when replacement to an X-address (accumulator) is called for. Another function of the N-decoding circuit is to decode the order number, which appears in the order register during BE. The order number is arranged to have the same timing as the last six N-digits of an order; as it always specifies a register address, its decoding follows the same process as that of the N-digits.

18. The N-decoding, like the F-decoding is carried out in two stages, the outputs of the decoding staticisers and their inverses, which correspond to the binary form of the addresses, being combined in the second stage to give waveforms corresponding more nearly to the programmer's notation. The first of the two stages of decoding is shown in fig. 6.6. Consider first the least significant three of the seven decoding staticisers, 34K2, 34L1 and 34L2, which are concerned with the last three N-digits or order-number digits or with the X-digits during E when these specify the replacement address. For N-decoding (or order-number decoding) the order register is tapped at I3, where the last three N-digits appear from ϕ37 to ϕ39 (fig. 6.3). The digits enter the decoding circuit through gate *x* of twin delay 34F1, which is controlled by waveform X42, the inverse of X43, the output of twin delay 22V2. X43 is the action waveform specifying X-replacement decoding (see below). The other gating waveform, X41, is negative only during transfers to and from the main store. The decoding staticisers are set at ϕ40, when the three digits will be distributed along their inputs, and are reset again at ϕ40 in the next word-time. During main-store *block* transfers, a half adder comes into operation so that the replacement address is increased by unity each time a word is transferred (see Chapter XI). The X-digits are sensed at I6 in the order register, where they appear from ϕ37 to ϕ39 in the same timing as the last three N-digits, enabling the same setting and resetting waveforms to be used. The X-digits are fed to the decoding circuit through gate *y* of twin delay 34F1, which is controlled by waveform X41 (not block transfers) and by X43, the output of twin delay 22V2.

19. The order groups that require replacement to an X-address (accumulator) are, 0, 2, 4, 5 and 6. There is one exception, the 'justify' order, function 23, in which the more significant half of a double-length number is to be replaced in an address designated by the N-digits. Gate *x* of delay 22V2 is controlled by ~F2, which is positive for all functions of binary form.

$$\delta \quad \delta \quad 0 \quad \delta \quad \delta \quad \delta$$

the symbol δ representing an unspecified digit, which may be either 0 or 1. These are all the functions for which the first three binary digits represent an even number,

i.e. all the functions in groups 0, 2, 4 and 6. The effect of ~F2 is inhibited by G02 and G13 (function 23), and the gate is opened during AD and BD (~C & D) allowing the decoding staticisers to be set at the end of these word-times to give significant outputs during E. Gate *y* of 22V2 is opened by G05 (function-group 5) and ~E; so the staticisers will be set for the X-digits at $p40$ preceding E when a function in group 5 has been decoded.

20. The F-waveforms become significant at $p0$ of AD and BD, and the G-waveforms at $p1$ of the same word-times. However, ~F2 is positive when no F-decoding has taken place; it is therefore necessary to use ~C in combination with it at gate *x* of twin delay 22V2 to prevent a casual decoding of X-digits at the end of CD. Gate *x* of 22V2 will always be open at $p41$ (the first digit-time) in AD or BD because ~C and ~F2 will both be positive, the F-decoding not yet having taken place. Consequently the output of 22V2 will always be positive at $p0$ of AD or BD. This output will drop again at $p1$ if the function decoding makes ~F2 drop, and will become positive again at $p2$ if a group-5 function (G05 positive) is decoded. In the case of function 23, the output of 22V2 will be positive at $p0$ and $p1$ of AD or BD, because of ~F2, and will drop again at $p2$, one digit-time after G02 and G13 have become positive. Note that, whereas waveform D is used in conjunction with ~F2, the effect of G05 must be 'timed' by ~E. This is because the orders of groups 0, 2, 4 and 6 that require X-replacement are all two-word-time orders (replacement to accumulators 6 and 7 in multiplication and division orders is automatic), whereas orders with group-5 functions (shift orders) are generally multi-word orders, and require the output of 22V2 to remain positive until the end of the last word-time preceding E.

21. Before the arrangement shown in fig. 6.6 for decoding the first four N-digits can be properly understood, some explanation of the structure of the binary N-address will be necessary. The whole computing store can be thought of as being made up of several eight-register blocks; the accumulators form one such block (of which the first member is imaginary); the ordinary registers comprise six complete blocks, and the special registers (handkeys, input and output and constants) additional incomplete blocks. The position of a register within a block is specified by the last three N-digits, and the block itself by the first four; the N-digits of an order are thus able to specify any one of eight registers in any one of sixteen blocks. There are, of course, far fewer than 128 registers in the computing store, so those registers that do exist can be allotted addresses in the manner most convenient for programming. Addresses 1 to 7 (block 0) are the accumulators, and addresses 15, 16, 17, 32, 33, 34 and 35 the special registers. Addresses from 64 onwards are reserved for the ordinary registers, register 0.0 corresponding to address 64, register 0.1 to 65 and so on. Thus the second, third and fourth N-digits give in binary form the block number of any register in octal notation, while the first N-digit serves to distinguish an ordinary register from a special register or accumulator with the same octal block and position number, e.g. to distinguish *ordinary register* 0.4 from *accumulator* 4, or *ordinary* register 1.7 from special register 15 (17 in the octal scale).

22. The first N-digit can thus be thought of as an ordinary-register marker. It is not inserted by the programmer, but by the input routine, which interprets a 'point' after the first address digit in an order on the programme tape as indicating an ordinary register. Now the order number can only refer to an ordinary register; consequently

it is unnecessary to provide it with an ordinary-register marker digit, and only the six digits specifying the block and position need be stored in the order-number register. But, to obtain the correct address-gating waveforms, the order number must be decoded as if it were preceded by an ordinary-register marker, i.e. the first N-decoding staticiser must be artificially stimulated when the order-number is decoded. The other occasions when the first N-staticiser must be deliberately stimulated are in satisfied jump orders and in block-transfer orders. The first N-digit in a jump order indicates not that the new order pair must come from an ordinary register (which is necessarily so) but whether the jump is to be made to the first or second order in a pair (see Chapter VIII). In block-transfer orders it is the three X-digits that specify the ordinary-register block (see Chapter XI).

23. The first four N-digits are tapped off the order register at I0, where they appear from $p37$ to $p40$. The second, third and fourth digits enter the decoding circuit through gate x of twin delay 34F2, where their entry is controlled by X43 (not X-replacement) and X41 (not block transfers). The first digit enters through OR gate 34E6 and AND gate 34G1, and the four digits are distributed along the staticiser inputs at $p40$. The other possible inputs to the first staticiser are J (for jump-address decoding), X40 (for decoding block-transfer addresses) and X20 (for decoding the order number). X20 is equivalent to (BE~J), and is therefore positive at the end of BE when the order-number is decoded. The use of ~J in this application is fortuitous; waveform X20 was available (it is used to control the entry of the order number to the order register, see fig. 6.2) and suitable for the purpose.

24. The second stage of N-decoding is shown in fig. 6.7. The staticiser outputs, S1 to S7, are combined to give N-waveforms corresponding to register blocks or to positions within blocks. Waveforms N20 to N27, which correspond to positions within blocks, are derived by combining the last three staticiser outputs or their inverses in AND gates preceding delays. As four digits are required to specify a register *block*, the three inputs available on the delay elements must be extended by the use of extra AND gates. Waveforms N10 to N15, specifying ordinary-register blocks, are made up of S1 and combinations of S2, S3 and S4 and their inverses. The accumulator block is specified by N00 (~S1 & ~S2 & ~S3 & ~S4), and the special registers by N01, N02 and N04. The N-waveforms correspond to the addresses in the *octal* scale; thus register 15 (handkeys) in octal form is 17 (one eight plus seven units) and would be reached by waveforms N01 and N27; register 33 (constant 1/2) in octal form is 41 (four eights and one unit) and would require waveforms N04 and N01. The N-waveforms become significant one digit-time later than the S-waveforms, i.e. at $p0$, except for N04, which is produced by an AND gate without a delay and therefore becomes significant at $p41$.

THE ORDER NUMBER

25. The parts of the machine concerned with the order number are shown in fig. 6.2. The order number is normally held in the order-number register, 25P, a nickel-line store of the usual type, and is routed into the order register through gate y of twin delay 25J1 during BE, except when a jump is called for. Since it specifies the address of an ordinary register and is to be decoded on the N-staticisers, it must have the same timing as the last six N-digits of an order; thus it appears from $p32$ to $p37$ on

entering gate *y* of 25J1. Coincident with the least significant digit of the order number, i.e. at ϕ32, a digit is fed into the '*a*' input of the order adder; thus the value of the order number is increased by unity in the order adder before it enters the order register proper and is decoded. The order number is routed back to the order-number register along the normal-circulation loop, and enters gate *y* from ϕ31 to ϕ36 of CD (see fig. 5.4 for generation of timing waveform). The inverse of the CD-waveform closes the normal-circulation gate of the order register (25J1/*x*) and also the feed-back gate *(x)* of the order-number register, erasing the old order number as the new order number is being read in.

26. The order number must not be increased by unity after a manual (handkey) order, otherwise (since it will already have been increased during BE of the previous order pair) an order pair would be missed. Moving the control switch to MANUAL inserts a pulse into delay 25G1, where it circulates until beat C after the manual order has been obeyed. The inverted output of 25G1 then closes gate *y* of 25V2, suppressing the addition during BE. It is necessary to incorporate delay 25G1, which 'remembers' that the order in the order register is manual, because the control switch is usually moved from MANUAL to NORMAL before the order is obeyed. Manual orders are considered in detail in Chapter IX.

27. The sequence for satisfied jump orders is as follows. Waveform ~J becomes negative, making the output of AND gate 25S4, and hence of delay 25G2, negative too. The result is that gates 25J1/*y* and 25V2/*y* are closed during BE, preventing the entry of the order number to the order register and suppressing the addition; instead, the normal-circulation gate, 25J1/*x*, remains open so that the jump order can re-enter to have its N-digits decoded in E as the new order number. It is these digits that are routed back to the order-number register during CD, the most significant (positive for jumps to the A-order) being deleted by the timing waveform. When the satisfied jump occurs in the A-order of a pair, the B-beat is omitted altogether. The subject of jumps is treated in detail in Chapter VIII.

MODIFICATION

28. It will be remembered that a modifier is a number thirteen digits long, which is held at the more significant end of the accumulator specified by the M-digits of an order. The methods of modifying were explained fully in Chapter IV, and may be summarised as follows:-

(i) For orders containing functions in group 7, the modifier is delayed three digit-times before addition. For all other modifying orders, the modifier is delayed six digit-times.

(ii) For orders containing functions in groups 0, 1 and 2, only the last three digits of the modifier are used: for orders containing functions in groups 4, 5 and 6, the last ten digits are used; for orders containing function 72 or 73, the first ten are used: for orders containing function 70, 71, 74 or 75, all thirteen digits are used.

29. When the M-digits have been decoded, the staticised outputs, S8 to S10 or their inverses, are used to route the modifier from the accumulator containing it via the X/M-bus to a common output line called the M-bus. This occurs during the latter half of CD or AE (except, of course, when the B-order is to be 'jumped over'), just before the X-digits of the corresponding order enter the order register. The timing on the M-bus is delayed by three digit-times on the timing on the N-bus, and is therefore correct for the modification of orders containing functions in group 7. For modifying orders containing functions in the other groups an extra three digit delays are inserted, giving a total delay of six digit-times.

30. Fig. 6.8 shows how the modifier's entry to the order adder is controlled. Gate *y* of twin delay 25V1, through which the modifier enters when delayed six digit-times, is controlled by staticiser, 26M1, which is set $\phi 31$ if either I2 or I3 is zero at that time. At $\phi 31$, I2 and I3 are the second and third F-digits respectively (refer to fig. 6.3). The staticiser is set, therefore, by functions of binary form

$$\delta \quad 0 \quad \delta \quad \delta \quad \delta \quad \delta$$

or

or $\quad \delta \quad \delta \quad 0 \quad \delta \quad \delta \quad \delta$

The first gives function groups 0, 1, 4 and 5, and the second groups 0, 2, 4 and 6. Hence the staticiser is set at $\phi 31$, and the 'delay 6' gate is opened at $\phi 32$, for all function groups except 3 (which is unallocated) and 7. The least significant N-digit enters the order register at $\phi 32$; thus the modifier is added to the N-digits of the order. Staticiser 26M1 is reset at $\phi 34$ through inverter 24K3, allowing only three digits of the modifier to enter the adder unless I4 happens to be positive at $\phi 34$. At $\phi 34$, I4 is the first F-digit. Thus it will be positive for functions of the form

$$1 \quad \delta \quad \delta \quad \delta \quad \delta \quad \delta$$

i.e. for functions in groups 4, 5, 6 and 7. Hence the staticiser is only reset after three digit-times for orders containing functions in groups, 0, 1 and 2. For functions in groups 4, 5 and 6, it is not reset until $\phi 41$, allowing the least significant ten digits of the modifier to enter the adder.

31. For orders containing functions 70 to 73, the modifier passes direct from the M-bus into the order adder through gate *y* of 25V1. This gate is opened by various combinations of F-digits and ϕ-pulses, and is held open by a repeater, 26Q3, until $\phi 41$, which is the timing of the most significant digit on the M-bus. Delay 26Q1 and AND gate 26T2 give the combination of I0 and ~I2 at $\phi 28$, and (one digit-time later owing to the delay in 26Q1) of I0 and W1 at $\phi 29$. It can easily be verified from fig. 6.3 that I0 and I2 at $\phi 28$ at the third and fifth F-digits, and that I0 and W1 at $\phi 29$ are the first and second. Thus gate *x* of 25V1 is opened at $\phi 29$ for functions of the form

$$1 \quad 1 \quad 1 \quad \delta \quad 0 \quad \delta$$

i.e. for functions 70 and 71 (single-word transfers). The least significant modifier digit appears on the M-bus at $\phi 29$; consequently the whole modifier is routed into the adder when the order contains function 70 or 71. It is worth noting that the same will

apply for functions 74 and 75 (external conditioning and interchange), which can be modified, if desired in the same way as single-word transfers.

32. Gate x of 25V1 can be opened alternatively by the combination of I2 and I3 at $\phi 31$ and I2 and I6 at $\phi 32$; these are the first, second, third and fifth F-digits and specify functions of the form

$$1 \quad 1 \quad 1 \quad \delta \quad 1 \quad \delta$$

i.e. functions 72 and 73 (block transfers). As these functions open the gate to the adder from $\phi 32$ only, the least significant three modifier digits will not be used. Unallocated function 76 and stop function 77 will also open the gate the adder from $\phi 32$; but the modification will have no significant effect. Notice that the modification is controlled in all cases by a special decoding of the F-digits, because the addition of the modifier must have been completed *before* the normal decoding operates.

C H A P T E R V I I

THE MILL

CHAPTER VII

THE MILL

00	$x' = n$	10	$n' = x$
01	$x' = x + n$	11	$n' = n + x$
02	$x' = -n$	12	$n' = -x$
03	$x' = x - n$	13	$n' = n - x$
04	$x' = n - x$	14	$n' = x - n$
05	$x' = x \,\&\, n$	15	$n' = n \,\&\, x$
06	$x' = x \neq n$	16	$n' = n \neq x$
40	$x' = c$		
41	$x' = x + c$		
42	$x' = -c$		
43	$x' = x - c$		
44	$x' = c - x$		
45	$x' = x \,\&\, c$		
46	$x' = x \neq c$		

1. The basis of the Mill is an adder-subtractor of the type described in Chapter III; this is preceded by a gating arrangement that ensures that the operands are routed as prescribed by the function in the order. The Mill also contains a device for sensing overflow, a circuit for testing 'jump' conditions, and circulation loops of digit length 42, 41 (for right shifts) and 43 (for left shifts). The Mill is used in some way or other in most orders; but this chapter will be concerned principally with orders prescribing simple addition or subtraction operations or logical operations, i.e. with orders containing functions in groups 0, 1 and 4.

FUNCTION GROUPS 0, 1 AND 4

2. As explained in Chapter I, a negative number entering into addition or subtraction must be expressed as its complement with respect to four instead of, as during storage, with respect to two; this is necessary to enable overflow to be distinguished from a change of sign, and involves no more than a single repetition of the sign digit. At the beginning of word-times AD and BD, outputs S1 to S10 or their inverses, and the 'N' combinations of S1 to S7 or their inverses, become significant. These waveforms open gates between the selected store lines and the N-bus and X-bus, which feed the Mill. The timing on the buses is delayed by one digit time on standard timing; so the sign digit appears on them at ϕ39.

Routing

3. Fig. 7.1 shows how operands are routed into the Mill. From the buses, they pass through the sign repeaters, 22L1 and 22L2; these are staticisers that can be

active for only one digit-time, their feedback gates being open only at $\phi 40$. Hence the digits that emerge from the delays at $\phi 40$, the sign digits, can recirculate and be repeated at $\phi 41$. After the sign digit has been repeated, the timing of the operand is, of course, delayed two digit-times on standard timing.

4. The additions, subtractions and NOT-EQUIVALENT operations prescribed by functions in groups 0, 1 and 4 are carried out in the Mill adder-subtractor. The AND operations are done before the adder-subtractor; but the results are routed back to store through the adder-subtractor. The functions in the three groups of the order code have been arranged to correspond with one another, which simplifies the necessary gating circuits. Except for subtraction orders, functions 2, 3 and 4 in each group, it does not greatly matter which operand goes to which of the two adder inputs. In subtraction, the '*b*' input is the negative one, i.e. the subtractor output is $a - b$. The order code has been arranged so that function 3 of the three groups implies replacement to the source of the '*a*' input to the subtractor, whereas function 4 implies replacement to the source of the '*b*' input.

5. The first routing operation is to arrange the operands according to the replacement address, i.e. according to the function *group*; this is done in the gates preceding delays 22K1 and 22M1. The output of 22K1 is the operand from the store line to which replacement will eventually be made - an N-line for function group 1 (G01), an X-line for function group 0 or 4 (~F1 & ~F2). The other operand, the output of delay 22M1, comes from an N-line for function group 0 (G00) and from an X-line for function group 1 (G01). The other operand for function group 4 is of course the counter number, which is tapped off the order register (see below).

6. AND operations are carried out in one of the AND gates preceding delay 22E2, which is opened by waveform G15 (the sixth function in any group). The result of the operation is then routed through the adder-subtractor via its '*a*' input. The second routing operation for addition, subtraction and NOT-EQUIVALENT operations is carried out in the gates preceding delays 22E1 and 22F1. Gating waveform F3 fives any function in the second half of a group, and ~F3 gives any function in the first half; waveform F5 gives the odd-numbered functions, and ~F5 gives the even-numbered functions. Hence gate *y* of 22F1 and gate *x* of 22E1 are open for even-numbered functions in the second half of a group (4 and 6), routing the operand from the replacement address to the '*b*' (negative) input as required by function 4. Gate *y* of 22E1 is open for odd functions in the first half of a group (1 and 3), routing for the operand from the replacement address to the '*a*' (positive) input as required by function 3. Gate *x* of 22F1 is open for the whole of the first half of a function group, and is used in functions 0 and 2, when only one operand is required, as well as in functions 1 and 3.

Counters

7. The counter number, which is simply the number represented by the modifier-extended N-digits of an appropriate order, enters the Mill through delay 22R3. It must be placed at the less significant end of a word; so, since operands entering the first set of gates preceding the Mill are delayed two digit-times on standard timing, the least significant digit of the counter must appear at the input to delay 22R3 at $\phi 2$. Fig. 6.3 shows that at $\phi 2$ the least significant N-digit appears in the

order register at output I10, which is therefore the tapping point required. The counter is gated in by G04 and by U1, the output of staticiser 22M2, which is positive from $p2$ to $p11$ of D to allow the whole counter, which in a modified order may be up to ten digits in length, to enter the Mill.

Adder-subtractor

8. The adder-subtractor and the circuits generating its controlling waveforms are shown in fig. 7.2. Carry suppression is applied to prevent carry digits from being repeated after the second sign digit when a negative number is generated by addition or subtraction. The facility for carry suppression also enables the adder-subtractor to perform a NOT-EQUIVALENT operation. It will be remembered from Chapter II that the 'sum' digit in addition or the 'difference' digit in subtraction is obtained by a NOT-EQUIVALENT operation; consequently an adder-subtractor in which the carry sequence is entirely suppressed simply performs a NOT-EQUIVALENT operation.

9. The operands entering the adder-subtractor are delayed by four digit-times on standard timing. The first sign digit therefore appears at $p0$, i.e. at $p(38 + 4)$, and the repeated sign digit at $p1$. To prevent the formation of further carry digits, the gates before delay 21H1, which produces the carry sequence, must therefore be closed at $p1$. The suppression waveform is derived from inverter 21F3, which inverts the 'mix' of timing pulse $p1$ (see fig. 5.2) and the output of twin delay 21E2. The inputs to gate y of 21E2 are ~F1 and G16 inhibited by G05. ~F1 gives function groups 0, 1, 4 and 5; consequently 'carry' is suppressed for the whole time that the machine is performing functions 06, 16 and 46 - the NOT-EQUIVALENT functions.

10. There is no special facility for 'carry' suppression in counter orders, as counters are sometimes to be regarded as full-length numbers. Consequently, the programmer must take care to ensure that a modifier held in the same accumulator as a counter is not altered unintentionally when the machine performs a function in group 4. It is worth noting in this connection, however, that function 46 can never affect a modifier, and that functions 40, 42 and 45 can sometimes be useful in deleting a modifier. There is a special facility for carry suppression in unit-count 'jump' orders (function 67, see Chapter VIII), by virtue of the signals applied at gate x of twin delay 21E2.

11. The adder-subtractor operates as a subtractor when the output of twin delay 21E1, X10, which is applied to AND gate 21G3, is positive; it operates as a ~~subtractor~~ adder when X11, the inverse of X10, which is applied to AND gate 21G2, is positive. X10 is positive when ~F1 is positive (i.e. for function groups 0, 1, 4 and 5) and when either F4 or F3 is positive. F4 is positive for functions 2, 3, 6 and 7 in a group, and F3 is positive for functions 4, 5, 6 and 7. Thus the circuit is in the 'subtract' condition for functions 2, 3 and 4 (in which subtraction is prescribed), for function 6 (in which the condition of the circuit is unimportant in the function groups considered here), for function 5 (in which the 'b' input to the subtractor is empty in these function groups) and for function 7 (which is unallocated). It is in the 'add' condition for function 1 (in which addition is prescribed) and for function 0 (in which one of the two inputs is empty). It is also in the 'add' condition for the whole of function groups 2, 3, 6 and 7.

12. The output of the adder-subtractor passes through a 35-digit delay line, 21L, and a single-digit delay, 21M3, before it is returned to store via the input bus. (The 35-digit line is also used in drum transfers to economise on equipment). The total delay involved in routing from store through the Mill and back to store is one word-time; thus for function group 0, 1 and 4 the result of an operation on operands routed out during AD or BD is ready for routing back during AE or BE. The delay of one word-time is made up as follows.

Store to adder input	4 digit-times
Adder	1 digit-time
Adder output to Mill output	36 digit-times
Mill output to store	1 digit-time
	Total 42 digit-times

OVERFLOW

13. Overflow is detected by comparing the sign digit with the repeated sign digit; if these two are different (i.e. not equivalent), overflow has occurred, and a staticiser is set. The sign digits are tested, as shown in fig. 7.2, at the outputs of two delays, 21M2 and 21P2, which are connected to the Mill output point. The necessary NOT-EQUIVALENT operation is performed by inverter 21Q2 and AND gate 21R2, the output of 21R2 being positive if the output of either delay is positive, but zero if the outputs of the two delays are both positive. The input to delay 21M2 is inhibited by waveform ~G07 (not function group 7); this is to prevent a spurious sensing of two digits during drum transfers, the timing in the Mill being then rather different from what is usual. In 'shift' orders, the output of delay 21P2 is fed back again to the Mill input, and, to prevent the repeated sign digit from entering into shift operations the input gate to delay 21P2 is closed at ϕ39.

14. The output of delay 21M2 is in standard timing, with the first sign digit appearing at ϕ38. The repeated sign digit therefore appears here, and the first sign digit appears at the output of 21P2, at ϕ39, which is the instant when the staticiser must be set. This of course occurs during word-time E of orders containing functions in groups 0, 1 and 4, the sign digit having been on the N-bus or X-bus at ϕ39 of D. The staticiser is not set for logical-shift orders (functions 52 and 53) or, except in E, for 'normalise' orders (function 56), as the two digits compared will not necessarily be associated with the sign of a number. Except for these, and group-7 orders, the overflow staticiser can be set in any operations that occur in the Mill. It can also be set in division orders and in double-length left shifts (function 54), which take place in the multiplier-divider. The setting waveform in these cases is mixed into the feedback loop of the staticiser.

15. The second sign digit emerging from the Mill is deleted by a timing waveform on the input bus before the number is written back to store, and replaced by a parity digit.

If there has been no overflow, this will not matter; but, if overflow has occurred, an indication of the sign of the stored number will lie in the fact that the overflow staticiser is active. The overflow staticiser is not reset automatically. If the programmer expects overflow, he may insert a 'justify' order, which, taking account of the state of the overflow staticiser and the apparent sign of the number in store, will form a double-length number of the correct value and sign. 'Justify' orders also reset the overflow staticiser by applying an inhibiting waveform to AND gate 21R1, and hence to the feedback gate of the staticiser. Waveform X18 on AND gate 21R1 is positive except while the 'start' switch is at START. It is applied here to ensure that the machine is always started with the overflow staticiser clear.

16. If the programmer suspects that overflow might occur, he will usually insert a 'jump' order containing function 64 or 65. This will give the machine alternative courses to follow according as overflow has, or has not, occurred, and will also reset the overflow staticiser if necessary by closing the feedback gate at $\phi 2$ of E after function 64 or 65 has been decoded. If, owing to a programme error or an electrical fault, no steps are taken to reset the overflow staticiser once it has become active, the machine will stop as soon as it decodes an order to write information on to the drum. The way in which this is done is described in Chapter XI.

CHAPTER VIII

JUMPS

C H A P T E R V I I I

JUMPS

60. Jump if $x = 0$

61. Jump if $x \neq 0$

62. Jump if $x \geqslant 0$

63. Jump if $x < 0$

64. Jump if overflow stat. is clear, or clear overflow stat.

65. Jump if overflow stat. is set, and clear overflow stat.

66. $x_m' = x_m + 1$ Jump if $x_m' \not\equiv 0 \pmod 8$

67. $x_c' = x_c - 1$ Jump if $x_c' \neq 0$

1. 'Jump' orders, function group 6, are provided to enable the computer to take alternative courses of action depending on the result of a previous operation. Either control will pass in the normal way to the next order in the sequence (no jump) or control will pass to an order in an address specified by the N-digits of the jump order (jump condition satisfied). The first four jump functions (60 to 63) prescribe a transfer of control conditional on a test made on the number in the accumulator specified by the X-digits of the order. (An unconditional transfer of control can be achieved by prescribing a jump if the content of accumulator 0 is zero). Orders containing function 64 or 65 enable the machine to take appropriate action if overflow has occurred in a previous operation. The last two 'jump' functions, 66 and 67, are used in unit-modify and unit-count orders; an example of the use of these was given in Chapter IV. The counter or modifier to be used is held in the accumulator specified by the X-digits of the order. The jump condition is tested in the Mill.

MILL OPERATIONS

2. The X-digits of most jump orders specify the address of the number to be tested for a jump condition. This number is routed out of store during AD or BD. With its sign digit repeated, it enters the Mill through gate *y* of twin delay 22K2 (see fig. 8.1) if a function in group 6 (G06) is specified; the number then passes through delay 22E2 to the '*a*' input of the adder-subtractor. For functions 60 to 63, there will be no other input to the adder-subtractor, and the number will pass through unaltered. There is a total delay of five digit-times between the store lines and the adder output; consequently the sign digit (ϕ38 in store) will leave the adder at ϕ1 of E.

3. The circuits following the adder-subtractor are controlled by gating waveforms derived from the F-decoding circuits. The table below indicates which of the F-waveforms will be positive for any given function in group 6. (If a waveform is negative, its inverse will of course be positive).

Function	F0	F1	F2	F3	F4	F5
60	1	1	0	0	0	0
61	1	1	0	0	0	1
62	1	1	0	0	1	0
63	1	1	0	0	1	1
64	1	1	0	1	0	0
65	1	1	0	1	0	1
66	1	1	0	1	1	0
67	1	1	0	1	1	1

4. The jump staticiser itself, 21W1, is set at $\phi 2$ of E for any function in group 6, provided that a NOT-EQUIVALENT operation between the waveforms at the outputs of twin delays 21T1 and 21U1 gives a positive result, i.e. it will be set if the waveform at either of these outputs is positive, but not if both are positive. It is reset again at the beginning of beat C. Now 21T1 is a staticiser with a direct-output feedback loop into which the outputs of delays 21S2 and 21T2 are mixed; its output is therefore positive if there is a pulse at the output of 21S2 or 21T2, or if any pulse came through since the last occurrence of the reset pulse, $\phi 3$. (Owing to the construction of the Pegasus packages, the 'mix' output of any unit is made positive whenever the direct output is made positive even if the current comes from an external source).

Functions 60 to 63

5. Gate y of 21T2 is controlled by waveforms $\sim$F3 and $\sim$F4, and can therefore be open for function 60 or 61. If the number in the accumulator specified in the order is non-zero, the output point of 21T1 will become positive (this can happen as late as $\phi 2$ if the accumulator contains nothing but a sign digit) and will remain positive until $\phi 4$. Gate x of delay 21U1 is opened by $\sim$F5 and $\sim$F3, so that its output is positive for function 60 or 62 and is negative for function 61. The NOT-EQUIVALENT condition will be satisfied, therefore, and the jump staticiser will be set, for function 60 only if the specified accumulator is empty, and for function 61 only if the number in the specified accumulator is non-zero.

6. Gate x of 21S2 can be open for function 62 or 63 ($\sim$F3 & F4) and is opened only at $\phi 1$; consequently only the sign digit can pass through, appearing at the output

point of 21T1 at ϕ2. The output of 21U1 is positive for function 62; so the jump staticiser is set for function 62 only if the sign digit is zero, i.e. if the number in the accumulator is positive or zero. The output of 21U1 is zero for function 63, and the jump staticiser is set, therefore, only when the sign digit is a 'one'.

'Overflow' Jumps (64 and 65)

7. Nothing is routed through the Mill for orders with function 64 or 65 (overflow testing), the X-digits of these orders being zero; instead the overflow signal is fed to the right-hand gate of delay 21T1, so that the output of this delay is positive if the overflow staticiser is set. The output of delay 21U1 is positive for function 64 (G14) but not for function 65; consequently the jump staticiser will be set for function 64 if the overflow staticiser is *not* set, and for function 65 if the overflow staticiser *is* set. The overflow staticiser is automatically cleared by functions 64 and 65 by the output of inverter 21Q1.

Unit Modify (66)

8. Functions 66 and 67 require an arithmetical operation on the number before it is tested for a jump condition. The Mill adder-subtractor is in the 'add' condition for all functions of group 6; as fig. 7.2 shows, subtraction is only possible when ~F1 is positive - in function groups 0, 1, 4 and 5. For function 66 (unit modify), a single pulse is applied to gate *x* of twin delay 22G2, which feeds the adder-subtractor *'b'* input, at ϕ28, and enters the adder-subtractor at ϕ29. As there is a delay of four digit-times between store and the adder-subtractor, the digit entering the *'a'* input of the adder-subtractor at ϕ29 would have left the store line at ϕ25, and would therefore have been the digit in place 13 in the word - the least significant digit of the modifier. Hence the modifier from the specified accumulator is increased by unity in the adder-subtractor, from which its least significant digit emerges at ϕ30.

9. Gate *y* of delay 21S2 is open for function 66 from ϕ30 to ϕ32, allowing only the least significant three digits of the modifier to pass through. These three digits constitute the remainder when the modifier is divided by 8; consequently, unless the modifier is an exact multiple of 8, staticiser 21T1 will become set, and, as the output of delay 21U1 is zero for this function, the jump staticiser will be set as well.

Unit Count (67)

10. As the adder-subtractor is in the 'add' condition, the subtraction of unity required in unit-count orders, function 67, must be achieved by adding -1. The timing of the counter in store is ϕ0 to ϕ24, and its timing on the adder-subtractor *'a'* input is ϕ4 to ϕ28. A string of digit pulses, equivalent to -1, enters the *'b'* input to the adder through gate *y* of delay 22G2 during the same interval. Carry suppression is applied as shown; gate *x* of twin delay 21E2 is open at ϕ27 for function 67, causing the entries to the 'carry' delay (see fig. 7.2) to be closed at ϕ28. The last digit of the diminished counter must then emerge from the adder-subtractor at ϕ29. Gate *x* of delay 21T2 is open for function 67 from ϕ5 to ϕ29, so that staticiser 21T1 is set, at latest at ϕ30, if the value of the counter is non-zero. As the output of delay 21U1 is zero for function 67, the jump staticiser will become set only when subtracting unity has not reduced the counter to zero.

11. During E of jump orders, the number tested is, of course, routed back to store in the usual way. Fig. 10.5 shows that the Mill is connected to the input bus for functions group 6, and that the 'erase' signal is generated. In functions 60 to 65, there is no difference between the number routed out and the number replaced; in functions 66 and 67, there is a difference of unity between the new counter or modifier and the old. Timing Chart 7 refers to jump orders.

ORDER-REGISTER CONTROL

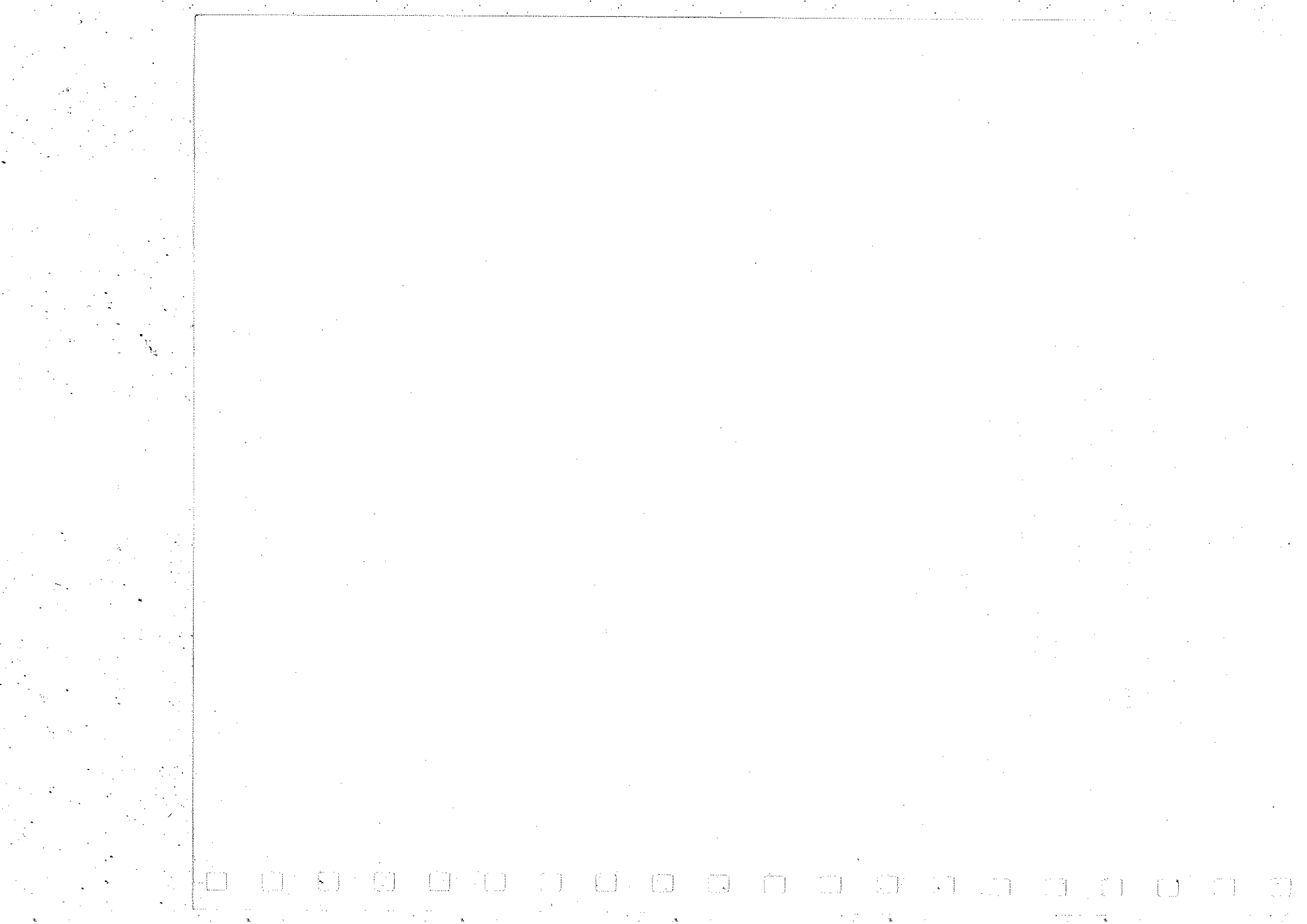
12. When a satisfied jump order is the second in an order pair the basic rhythm of the machine is not altered. The only modification to the normal sequence of events is that the digits in the order-number register are not fed into the order register; instead, the jump order makes another circulation in the order register during BE, and its N-digits are decoded instead of the order number, to give the address of the new order pair. The addition of unity in the order adder is suppressed, and the N-digits of the jump order are fed back to the order-number register to replace the old order number. (See Chapter VI, paragraph 27). When a jump is called for, waveform ~J becomes zero at ϕ3 of E. Fig. 6.2 shows how the effect of this is to close the order-number gate and open the normal-circulation gate to the order register. The waveform controlling the order-number gate also controls the *'a'* input to the order adder and inhibits the addition of unity that would be necessary for an ordinary order number. When a satisfied jump order is the first order in a pair, the procedure is similar except that the B-beat is omitted altogether owing to the control exercised by ~J on the B-staticiser and the beat trigger (see fig. 5.5). Waveform ~J also prevents the 'shuffle' circulation in the order register, which would normally occur during AE, and causes the normal-circulation gate to open instead.

13. Jumps to the B-order of a pair are denoted, in programmer's notation, by a + written after the N-address. The input routine then arranges for the most significant N-digit of the collated order to be zero. To give the B-order the correct timing in the order register after a jump order specifying it, the machine is allowed to follow the A-beat in the normal way so that the 'shuffle' occurs as usual during AE. The A-order, however, is not allowed to enter the order register; consequently, during the A-beat, the machine will obey the null order, 'Transfer the contents of accumulator 0 to accumulator 0'.

14. The order pair is gated to the order register by two waveforms, one of these, X16, is positive from ϕ1 to ϕ38 provided that the computer is not set for manual operation (see fig. 6.2). The other, X15, is generated by the circuit shown at the bottom of fig. 8.1. In normal circumstances staticiser 25K1 is set at ϕ0 of CD, and is not reset until ϕ41 (the beginning) of AD. Consequently its output is positive from ϕ1 (CD) to ϕ41 (AD), allowing the whole order pair into the order register. If, however, the output of staticiser 25K2 is positive, staticiser 25K1 will be reset at ϕ19 of CD, so that its output will be positive only from ϕ1 to ϕ19 and only the B-order, the first in time on the N-bus, can enter the order register. Staticiser 25K2 is set if J is positive, i.e. if the jump condition is satisfied, and if order-register output I9 is zero at ϕ7. Fig. 5.3 shows that I9 at ϕ7 is the most significant N-digit, which is zero only for jumps to the B-order.

TIMING CONTROL

15. The omission of the B-beat after jumps in A is achieved by using ~J to control the B-staticiser in the beat generator (fig. 5.5). If the machine is in the A-state and ~J is negative, the next beat trigger will cut off the A-staticiser without activating the B-staticiser; the machine will then revert to the C-state and the jump staticiser will be reset. A further effect of waveform ~J is shown in fig. 5.5. Normally a stop during the A-beat would inhibit the beat trigger, leaving the machine 'suspended' in the A-state until the stop is cleared. If, however, waveform ~J is negative (satisfied jump), the output of inverter 24R1 will remain positive, enabling the machine to change over to the C-state and pick up the new order pair before the stop operates; the new order pair will then circulate in the order register until the stop is cleared. This situation will often occur in single-shot working.



C H A P T E R I X

STOPS AND STARTS

CHAPTER IX

STOPS AND STARTS

1. There are two controlling handswitches on the programmer's panel; these will be referred to as the 'stop' switch and the 'start' switch. The 'stop' switch has three positions - RUN, STOP and SINGLE SHOT. When it is at STOP, the operation of the beat counter is inhibited, so that the next order cannot be obeyed but remains circulating in the order register. The movement from STOP to SINGLE SHOT is spring-loaded; the machine obeys a single order when the switch is moved to SINGLE SHOT, and then stops in a condition in which it is ready to obey the next order. The 'start' switch determines the source of the next order to be obeyed; it also has three positions - START, NORMAL and MANUAL. When it is at START, the 'start' order pair (refer to Chapter V, paragraph 12) is fed into the order register; when it is at NORMAL, orders are taken from store in the ordinary way; when it is at MANUAL, the order fed into the order register corresponds to the setting of the top row of handkeys on the programmer's panel.

'STOP' SWITCH

2. The circuit of the 'stop' switch is shown in fig. 9.1. The switch itself is made to control a system of staticisers, to avoid spurious effects due to the rise or fall of potential at the switch contacts; it is the setting pulses of these staticisers that actually initiate the switching processes, and, if a 'half' digit should be produced, it will either be deleted or else regenerated into a standard digit pulse after about five circulations through the staticiser. The setting pulses are arranged to occur as long as possible before the outputs of the staticisers are required.

3. While the switch is at STOP, staticisers 12P2 and 12M2 (fig. 9.1) are both active. The 'mix' output of 12P2 is one of the alternative constituents of waveform V24, which inhibits the beat trigger (refer to fig. 5.5); the direct output, X78, is used to inhibit the hooter drive, which operates on the internal stops. When the switch is moved to RUN, staticiser 12M1 becomes active, the first pulse emerging from it at ϕ40. Nearly one word-time later, at ϕ39, the output of inverter 12L3 becomes negative, cutting off 12P2 and 12M2 and also certain other staticisers concerned with internal stops. The result of cutting off 12M2 is that 12M1 is cut off one digit-time later. After this, while the switch is at RUN, the three staticisers associated directly with it are inactive, and waveform X70 is permanently positive.

4. When the switch is moved to the SINGLE SHOT position and released, the machine obeys a single order and then stops in a condition to obey the next order at the next movement of the switch. For SINGLE SHOT, as for RUN, staticiser 12M1 first becomes active; nearly one word-time later, the output of inverter 12L3 falls for one digit-time cutting off staticiser 12M2, which in turn cuts off staticiser 12M1. The 'trigger'

side of staticiser 12P2 remains primed, however; consequently, although this staticiser is cut off at $\phi 39$, it is set again at $\phi 0$, its output being negative only from $\phi 40$ to $\phi 0$ after the switch is operated and remaining positive thereafter. The beat trigger is produced at $\phi 40$; hence its inhibition, V24, is removed once only, allowing one order to be obeyed, after which the machine will stop again. The generation of the beat trigger is shown in fig. 5.5. It will be seen that the machine cannot stop in a B-state but must continue to the next C-state to pick up the next order pair from store; nor can it stop in an A-state when a jump is called for, but continues into the C-state to enable the new order pair to be selected.

INTERNAL STOPS

5. The output of staticiser 12P2, which inhibits the beat trigger, is mixed with seven other 'stop' signals that are produced by the machine in the event of a programme error or electrical fault. These include a stop caused by an attempt to transfer information to the drum while the overflow staticiser is active (see Chapter XI), stops caused by failure of the parity check on words routed out of the main store or computing store (see Chapter X), and a temporary stop caused by an attempt to use the input or output equipment while either is still 'busy' (see Chapter XII). The way in which the other three stops are sensed is shown in fig. 9.1; these arise from the 'stop' order (function 77), the absence of a 'go' digit in an order pair (optional stop), or the specifying of unallocated functions in orders. Normally a hooter will sound whenever an internal stop, except the 'busy' stop, occurs, and a neon lamp will come on to show the cause of the stop. The operation of the hooter on stops can be inhibited with a switch on the programmer's panel; it can also be cut off by moving the 'stop' switch to STOP. The operation of the hooter is described in Chapter XVII.

6. When the machine decodes the 'stop' function (G07 & G17), staticiser 12P2 is set unless the switch on the engineer's panel associated with it is at OFF. Staticiser 12P2 can be cleared by flicking the 'stop' switch to STOP, and then back to RUN or SINGLE SHOT. Notice that it is the movement from STOP to RUN or SINGLE SHOT that clears the staticiser, by making waveform X70 negative for one digit-time ($\phi 39$); the staticiser remains active, and the neon lamp remains alight, while the switch is at STOP, so that the cause of the stop can be checked if necessary. A '77' stop can occur in C (next order - A) or in A (next order - B).

7. For the optional stop, the stop-go digit is sensed from the N-bus, where it appears at $\phi 39$ of CD; the machine stops if this digit is zero. The inverse of the N-bus waveform is gated during CD through AND gate 25H3, and is used to control the setting of staticiser 12P1 at $\phi 39$; hence the optional stop always occurs at the end of CD, with the 'stop' order pair in the order register and ready to be obeyed. Like the stop-order staticiser, the optional-stop staticiser can be cleared by flicking the 'stop' switch to STOP, and back to RUN or SINGLE SHOT. The optional stop can be inhibited by means of a switch on the programmer's panel. The output of this switch is not used directly, but is made to control a staticiser, 23V1; consequently it can be operated while the machine is running, without injecting 'half' digits into the system. The setting of the optional-stop staticiser is inhibited also by waveform X3, which is negative while a manual (handkey) order is being fed into the order register.

The reason for this is that an order pair will be routed from store on to the N-bus at this time, and, although it will not enter the order register, its stop-go significance will be sensed. The stop preceding a manual order will usually be an optional stop; as the order number is not increased when a manual order is used, the stop would occur again if nothing were done to prevent it.

8. The unallocated functions are the whole of group 3, function 7 of groups 0 to 5, function 76 and, in installations without a buffer store, function 75. The staticiser associated with this type of stop, 12R2, is set at ϕ40 under the control of the mixed outputs of twin delays 22C1 and 22C2. The output of 22C1 is positive when any function in group 3 is decoded, or when G17 is positive without F1 and F0 being positive together, i.e. for function 7 in any group except 6 and 7. The output of 22C2 is positive for function 76, or, if connected, 75 ('interchange' function). Feedback in staticiser 12R2 can be inhibited by means of a switch on the engineer's panel; flicking the switch off and then on clears the staticiser once it has been set. The 'stop' switch has no effect on this type of stop; but the staticiser is cleared by the 'start' switch on START. Like the '77' stop, this type of stop can occur in C or A. The 'mix' output of 22C2 is also used to set E, so that unallocated functions will occupy two word-times and cause a stop between beats. Notice that the output of 22C1 will also appear at the output point of 22C2.

'START' SWITCH

Starting

9. The 'start' order pair was analysed in Chapter V, and was seen to be equivalent to an order pair transferring the first block of eight words into the computing store from the isolated part of the main store and then transferring control to the first order in the block. The 'start' switch, shown in fig. 9.2, has two poles, one of which controls the entry of the 'start' order pair to the order register. As with the 'stop' switch, all the significant terminals of the 'start' switch are connected to staticisers to prevent 'half' digits from being injected into the system. As there are no positive digits in the 'start' order pair between ϕ12 and ϕ19, a setting pulse at ϕ11 gives staticiser 25E1 plenty of time in which to settle down before its output is required.

10. The output of staticiser 25E1, X17, is used to gate the 'start' order pair into the order register. As the inverse of this waveform, X18, is used to control the circulation gate of the order register, it is only the last active word-time of the staticiser, when the switch has been returned to NORMAL, that is significant. X18 is also fed to the beat generator (fig. 5.5) to ensure that the machine starts in a C-state. It is important to notice, however, that the 'stop' switch *must* be at STOP while the 'start' order pair is being fed in. The normal-circulation gate of the order register is controlled by U16, which is negative during CD (see fig. 6.2); and CD will be the state of the machine if the 'stop' switch is at RUN while the 'start' switch is at START, since D is not controlled by X18. If the 'stop' switch is at STOP, the beat trigger, and hence D, will be inhibited, so that C alone of the rhythmic waveforms will be positive. U16 will then be positive, and normal circulation will

occur as soon as X18 becomes positive, i.e. after the 'start' switch is moved to NORMAL. When the 'stop' switch is moved to RUN, the machine will obey the 'start' order pair and proceed with the first block of Initial Orders.

11. Waveforms X17 and X18 are used to set several conditions that must obtain at the start of a programme. X17 is used to reset the external-conditioning relays. X18 is fed to several of the internal-stop staticisers and the overflow staticiser to clear them, and to the F-decoding staticisers, as the normal reset waveform for these, E39, may not have been present since the machine was switched on. (The N-decoding and X/M decoding staticisers are reset by p-pulses).

Manual Operation

12. The other pole of the 'start' switch enables a manual order set up on the hand-keys to be fed into the order register. There are twenty-two handkeys; so a complete nineteen-digit order can be manufactured, with an extra three digits at the more significant end to give the effect of modification. This is normally an A-order; but it must be presumed to be associated with a null B-order. The usual procedure for inserting a manual order into a programme is to prepare the programme with a 'stop' order pair in the required place. When the machine has stopped, and the order to be inserted has been set up on the handkeys, the handswitches are moved in succession to STOP, to MANUAL and to NORMAL; these operations will feed the manual order into the order register. Two SINGLE-SHOT operations are then performed, one to obey the manual order, and the other to obey the null B-order. The order pair that originally caused the optional stop will then be fed again into the order register during C. Returning the 'stop' switch to RUN will now enable the machine to continue with the programme. Notice that, if the inserted order specifies a satisfied jump, only one single-shot operation will be necessary.

13. Fig. 9.2 shows the circuits controlled by the MANUAL pole of the 'start' switch. The output of staticiser 25C1 is positive from $p1$, and remains positive as long as the switch is at MANUAL. The 'mix' output of this staticiser is used to prevent the addition of unity to the order number: as can be seen from fig. 6.2, the fact that a manual order is being, or has been, obeyed will be 'remembered' by repeater 25G1 until CD; consequently the order number brought into the order register during the preceding BE will be the same as the last order number brought in before the machine was stopped (unless, of course, the manual order required a jump). Thus no orders are omitted when a manual order is inserted.

14. When the output of staticiser 25C1 has been positive for one word-time, staticiser 25C2, which was set when the switch was at NORMAL, is reset through inverter 25F3; the effect of this is to close the N-bus gate to the order register. At the same time, staticiser 25D2 is set, its direct output gating the manual order into the order register through gate y of 25E2, and its 'mix' output inhibiting normal circulation to erase the previous content of the order register. As staticiser 25D2 is controlled by the output of staticiser 25C2, it can be active for only one word-time, after which the normal-circulation gate opens so that the order register can retain the manual order. The effect of manual operation on the order-register gates can be seen from fig. 6.2.

15. The manual order cannot be obeyed until the 'stop' switch is moved to RUN or SINGLE SHOT. If, however, it is intended for the manual order to be followed by the rest of the programme, or even by a different manual order, the 'start' switch must be returned to NORMAL to enable staticiser 25C2 to become active again; this must be done before the 'stop' switch is operated. It is often useful during maintenance to leave the 'start' switch at MANUAL so that, by moving the 'stop' switch to RUN, the machine can be made to obey repetitively the same manual order, and the operation of the circuits involved can be studied at leisure. In this case, the manual order will enter the order register through gate x of twin delay 25E2, where it is gated by waveform X15. Waveform X15, it will be remembered, is the waveform that controls entry from the N-bus to the order register; it is positive from ϕ1 to ϕ41 of CD, unless a jump to the second order of a pair is called for, when it is positive only from ϕ1 to ϕ19 (see fig. 6.2). The order register thus operates in the same way for manual repetition as for normal order sequences, except that the order pair is fed in from the handkeys instead of the N-bus, and that the order number remains constant.

16. The manual order, therefore, is fed into the order register either by switching to MANUAL, when staticiser 25D2 is active for one word-time, or, with the switch set at MANUAL, during CD, when X15 is positive. A manual order can be inserted in the B-position in the order register by stopping in A (77 stop or optional stop followed by one single shot). The 'start' switch can then be put to MANUAL and back to NORMAL to insert the manual order. After the manual order has been obeyed, the order pair in which the machine was stopped will enter the order register again (unless the manual order required a jump), and the A-order of the pair will be obeyed once more. In general, two single-shot operations are required when a manual order is inserted: one for the order itself, and one for the null B-order following it. Only one single-shot operation is necessary when the order is inserted in the B-position or when the order requires a jump.

17. Manual jump orders leave the beat counter in a state prescribed by the order itself, as do normal jump orders. If the switch is returned to NORMAL before the order is obeyed, the order, or order pair, prescribed in the jump, is taken into the order register to be obeyed next: if a manual order is being run repetitively, the timing controls will operate as for a jump; but the order taken into the order register will not come from store but from the handkeys. A manual jump-to-A order will be run repetitively in the normal way for jumps; but a jump-to-B order will make X15 positive from ϕ1 to ϕ19, allowing only the null B-order following the manual order to enter the order register. Under these circumstances, the jump order will be obeyed in alternate bars as follows:

```
    C   Manual order enters O.R.
  ┌─A   Jump order decoded.
  │ B   Omitted.
  └─►C   Null B-order alone enters O.R. Manual (A) order annulled by X15.
    A   Null order obeyed (as usual in jumps to B).
    B   Null B-order obeyed.
    C   Manual order enters O.R.
        e.t.c.
```

See also Timing Chart 10.

HANDKEYS

18. The handkeys are arranged in five groups, marked E, N, X, F and M, the last form groups corresponding to the four digit groups of an order, and the first group corresponding to the extra digits produced by modification. The handkeys are connected to number generators as shown in fig. 9.3, and are gated in succession for one digit-time each from $\phi 18$ to $\phi 39$. It will be seen that the manual order emerges from delay 24J3 (fig. 9.2) from $\phi 20$ to $\phi 41$ and enters the order adder from $\phi 21$ to $\phi 0$; so it has the timing of the A-order in a pair. An extra digit is inserted at $\phi 18$, the timing at this point of the most significant N-digit (i.e. the ordinary-register marker) in the B-order, the purpose of which is to enable the 'shuffle' arrangement in the order register to be tested. The manual B-order thus causes the machine to perform the operation of transferring the contents of register 0.0 through the Mill into accumulator 0, and parity checks will operate as explained in Chapter X. Notice that the extra digit will *not* appear on the monitor display of the handkey setting.

19. It should be noted that, although provision is made for setting the M-digits of a manual order, no modification can take place on the first entry of the manual order to the order register. This is because the modifier is gated on to the M-bus by CD or AE (see fig. 6.8); but, when the manual order is first fed in, the machine is in a stop condition, i.e. its rhythmic state is that of C alone or A alone. Handkeys are provided for the M-digits principally so that single orders can be fed into store using address 15 (see Chapter X); however, when a manual order is obeyed repetitively, it will be modified in the normal way during every bar but the first.

NOTE

The reason for having a bleed to -10V as well as a bleed to -150V on two contacts of the 'stop' switch is to increase the rate of fall in potential at these contacts during single-shot working. When the switch is moved from STOP to SINGLE SHOT, the input to gate y of staticiser 12M2 must fall below cut-off before the end of the word-time when staticiser 12M1 is active, otherwise 12M1 will be set again. When the switch is moved from SINGLE SHOT to STOP, the input to gate x of 12M1 must fall below cut off before $\phi 39$ of the first active word-time of 12M2, otherwise 12M1 will be set again, the beat-trigger inhibition will be removed, and another order will be obeyed.

C H A P T E R X

THE COMPUTING STORE

desired. Their storage function is described in Chapter XV. The other accumulators are not strictly accumulators in the sense that there is an arithmetical circuit necessarily associated with each. However, as every arithmetical order must specify an accumulator address, the Mill and the accumulator specified can be thought of as forming together an accumulator of the conventional type.

5. Accumulators 2, 3, 4 and 5 are shown in fig. 10.2, and accumulator 1 is shown in fig. 10.3. They are similar to the ordinary registers; but, as they are to be selected alternatively by the N-decoding circuit or the X/M-decoding circuit, they require two sets of output gates and have two output lines. The inverters used for erasure are not those on the nickel-line packages, as they have more to do than simply clear the store lines. It will be remembered that the N-decoding is always used for routing words into the store lines, even when replacement is to the X-address. The N-waveforms are combined in pairs in AND gates, the outputs of which open the input gates to the lines, and, combined with the 'erase' signal, clear the selected accumulator of its previous content by closing the feedback gate. The content of an accumulator can be routed out as a result of N-decoding, as the address-waveform AND gate (33J1 for accumulator 2) is connected to a delay gate (33P2*y*), the output of this delay leading to the N-bus. The X-address output routing is done in two stages to economise on entries to gates. The output of the last X/M-decoding staticiser, S10, or its inverse, ~S10, is used in the first stage of the routing. S10 corresponds to odd-numbered addresses and ~S10 to even-numbered addresses; thus the output of accumulator 2 is gated by ~S10 in AND gate 33J2, or that of accumulator 3 by S10 in AND gate 33M3, on to an output line common to the two accumulators. This is connected to gate *x* of twin delay 34U2, where it is gated by ~S8 and S9, which are positive for either accumulator 2 or accumulator 3. (Package references for S-waveform repeaters are given in fig. 10.6.)

6. A complication arises from the fact that accumulators can be used for holding modifiers. It may sometimes happen that a number produced in one order is required in the next order for use as a modifier; consequently a number being routed back to an accumulator during AE by the N-decoding of an A-order might be called for at the same time by the M-decoding of the B-order. The delay of one word-time involved in routing through the store line could not be tolerated in this case, so the store line is by-passed. Accumulator 3, for instance, is by-passed through AND gate 33M4, which is opened by the N-decoding waveforms (N00 & N23) in coincidence with the signal corresponding to the last digit of the M-decoding (S10). The store line can be by-passed only when the outputs of corresponding X/M-decoding staticisers and N-decoding staticisers are significant together, which can happen only in AE. The normal output gate from the store line (33M3) for accumulator 3 is closed during replacement by the waveform from inverter 33R3; consequently the old content of the store line cannot appear on the X/M-bus; though it will appear as usual on the N-bus.

Special Registers

7. The programmer's handkeys have already been described in Chapter IX (see paragraphs 18 and 19 and fig. 9.3) in connection with manual-order control. They can also be used as a source of half words (single orders, or numbers accurate to 19 binary places) that are fed into the machine under the control of orders. The handkeys represent address 15, which is equivalent to 17 in the octal scale, and therefore corresponds to address

machine if an even-parity word appears on the bus during a 'drum-read' order. This circuit also checks, whenever it is not being used for anything else, the parity of the address digits on the address track of the drum.

X/M-Bus

13. The X/M-bus is shown in fig. 10.4. The word on this bus, which will have been selected by the X/M-decoding, passes on to the X-bus through AND gate 34G3 during AD or BD (D & ~C), timing waveform 1-39 cutting off the parity digit. During CD or AE, the digits on the X/M-bus are to be treated as those of a modifier, which will be routed through three digit delays on to the M-bus as shown in fig. 6.8. The X/M-bus parity-counting circuit is similar to that described in Chapter III (paragraph 23 and fig. 3.6). except that its operation is inhibited by S8 & S9 (accumulator 6 or 7) or by ~S8 & ~S9 & ~S10 (accumulator 0). Staticiser 32K2 is set at ϕ0, one digit-time before the least significant digit from store appears on the bus. Its output at ϕ41, one digit-time after the parity digit has passed, is negative for odd parity or positive for even parity (except for the content of accumulator 0, 6 or 7, when it will be negative whatever the parity). The output of the staticiser is used to stop the machine as explained in paragraph 21. The parity is tested only during AD or BD; it is not tested during CD or AE when modifiers are being routed out of store. This is because the final parity count, which appears at ϕ41, would not stop the machine until the end of the next beat, and could therefore be misleading. The parity of 'modifier' words is tested of course during 'unit-modify' orders.

N-Bus

14. The N-bus, also shown in fig. 10.4, is fed through OR gate 34P6 from the accumulators or special registers (W19) or the ordinary registers (W20 or W21). The whole word, complete with parity digit, passes through AND gate 24Y2 to the main store, the order register, or, in some installations, a buffer store. With the parity digit deleted, it passes through AND gate 24Y1 to the multiplicand-divisor register, the Mill (through the sign repeater) or to the circuit that senses for an optional stop (fig. 9.1). The conventional type of parity-counting circuit is used, the output of staticiser 35C1 being positive for even parity and negative for odd parity. It makes a count during every word-time except E, because a test during E could not stop the machine until the end of the next beat, which might be misleading. The word appearing on the N-bus during E has no programme significance anyway.

15. As some of the words that appear on the N-bus may be transferred to the main store, their parity must be correct even if they come from one of the 'random-parity' addresses. The N-bus parity counter is not inhibited, therefore, like the X/M-bus counter, for these addresses; instead, a parity digit is injected, if necessary, through delays 35F3 and 35F2, appearing on the bus at ϕ40. Two of the inputs to 35F2 are identical with two of the inputs to the staticiser, and will both be positive for even parity. The third input, from delay 35F3, is positive at ϕ38 for any address in which the parity is not necessarily odd. Cathode-followers 33Q4 and 33Q5 represent accumulators 6 and 7; AND gate 35E2 represents accumulator 0; AND gate 35E4 represents the non-existent ordinary-register blocks 6 and 7. (The representation of the accumulators as block 7 in drum-transfer orders is transformed by the decoding circuits into

fed to the parity circuit through delay 20M1 except during E, when the Mill output is being checked, or when X34 is positive, when the machine is reading orders or numerical data from the drum. Notice that, as the beat counter is inhibited at ϕ40 through a staticiser and AND gate 20P1, and as the parity-count staticiser is set at ϕ41, address-track digits appearing on T23 at ϕ39 and ϕ40 are not counted. The seven address digits appear at T23 from ϕ6 to ϕ12.

PARITY STOPS

20. The input parity waveform from AND gate 20P1 is fed to 'stop' staticiser 12Q2 (fig. 10.6), which is set at ϕ39 in drum-read orders when waveform X34 is positive (i.e. while information is actually being read from the drum) or in other orders (for the address-track check) during all word-times except E (when the input bus may carry random-parity information from the Mill). The setting of the staticiser can be inhibited with the 'drum parity failure' switch on the engineer's panel. It is inhibited too when the 'drum-crystal' switch on the engineer's panel is at CRYSTAL; this is to prevent a 'failure' from being recorded when the drum is inoperative and the computer is controlled by the crystal clock. The output of staticiser 12Q2 is used to control an indicator lamp on the programmer's panel; it is also used to inhibit further 'read' action, in cases of drum parity failure, and to inhibit the half adder that counts through the registers in a block during block-transfer orders. Address-track parity failure is distinguished from storage parity failure by the type of order that has caused the stop.

21. The X-parity and N-parity waveforms are fed to 'stop' staticisers 34C1 and 34C2, which are set at ϕ41. The outputs of these two staticisers control the same indicator lamp; but they are distinguishable by virtue of separate connections to the 'test 2' wafer of the monitor selector switch. As information is routed on to the N-bus whenever a word is read into store, the N-bus parity should be correct during 'read' orders (in the absence of a storage fault); however, during the first 'read' order after the machine is switched on, the whole computing store will be empty, all register contents then having even parity. To prevent a parity stop due to casual routing out during 'read' orders in the 'start' routine, the setting of the stop staticisers is inhibited by X34, the 'read action' signal. Parity checks on the output buses during 'drum read' orders in any case have little relevance to the order being obeyed. Notice that ~X34 clears the staticisers if either has been set during the early part of a 'read' order before coincidence (see Chapter XI) has been found.

22. The 'mix' outputs of the three parity-stop staticisers are used, with the other internal stops (fig. 9.1) to inhibit the beat trigger (fig. 5.5). The setting of the three parity staticisers can be inhibited, or a staticiser can be cleared, after it has become set, with a switch (KS5) on the programmer's panel. Notice that this switch controls both gates of the drum-parity staticiser, 12Q2; this is necessary because 12Q2 is set at ϕ39, producing an output pulse at ϕ40 just when the stop line is sampled. The other two parity-stop staticisers are set at ϕ41.

4. The staticiser outputs, S21 to S26, and their inverses are combined in threes as shown in fig. 11.2 to give waveforms R00 to R04 and R10 to R17. The recording heads are arranged in 'columns' of eight, the heads in each column being fed from a common output stage for writing, and feeding into a common amplifier for reading. Waveforms R00 to R04 are used for selection between columns, and waveforms R10 to R17 are used for selection within a column. The columns, like the tracks, are arranged in pairs, and each of waveforms R00 to R04 will therefore actuate a pair of columns. Waveform R04 and the pair of columns controlled by it are concerned only with the permanent part of the store.

5. Waveforms R00 to R04 are fed to the read-switch controls (package type 19) and the write-switch controls (package type 14), the 'mix' outputs of the delays going to the read-switch controls. The 'mix' crystals are used simply as resistors to improve the smoothing; these, combined with the shunt capacitors at the outputs of the delays, and smoothing circuits within the control packages, ensure that the final switching waveforms contain no spurious 'spikes'. Row selection requires negative switching waveforms; consequently R10 to R17 are produced in inverters. Staticiser outputs S24, S25 and S26 are taken through the 'mix' crystals, and shunt capacitors are added, to give some smoothing apart from that provided on the control packages themselves.

Address-Track Coincidence

6. The least significant four N-digits in drum-transfer orders specify one of the sixteen blocks in a track pair. Moreover, the X-digits in single-word transfer orders (functions 70 and 71) specify a particular word out of the eight in a block. Selection within a track is achieved by waiting until the drum is in such a position that there is exact correspondence between the digits derived from the address track and the address digits in the order; a 'coincidence' signal is then generated, which, by initiating appropriate gating waveforms, causes the information to be routed to or from the drum. When there is exact correspondence, a NOT-EQUIVALENT operation between the outputs of delays 20H2 and 20E2 (see fig. 11.3) gives a null result for a period of 7 digit-times; staticiser 20F1 cannot then become active, and the coincidence signal, the inverse of the staticiser output, will be positive for rather more than one word-time.

7. First consider single-word transfers, functions 70 and 71. In these, waveform ~F4 is positive and F4 is negative; a timing pulse therefore emerges from twin delay 20J1 at $\phi 4$ and is repeated in delay 20H1 until $\phi 11$. Output I16 from the order register is thus gated into delay 20H2 from $\phi 5$ to $\phi 11$, allowing the three X-digits and the last four N-digits of the order to enter (refer to fig. 6.3). It is arranged for the digits coming out of the amplifying and shaping circuits associated with the address track to have the same timing as their counterparts at I16. The waveforms from the pair of address tracks are combined in twin delay 20E2, where they are gated by T4 (even ϕ-pulses) and T3 (odd ϕ-pulses). The output of 20E2 is also fed to the input-bus parity circuit for checking (see Chapter X, paragraph 19).

8. Gate x of the coincidence staticiser, 20F1, can be opened only from $\phi 6$ to $\phi 12$, which is the timing of the digits emerging from delay 20H2; consequently, unless there is perfect correspondence between the digits in the order and those on the address

required position will usually be sufficient; but occasionally coincidence will occur during word-time D of the order, when switching transients might be intolerably high. It is therefore arranged for there to be a delay of one word-time for writing, or three word times for reading (because signal levels are lower for reading than for writing), before the coincidence signal can become effective. If coincidence should occur during this delay period, the machine must wait for one full drum revolution before the 'write action' or 'read action' waveform becomes positive and the transfer can take place.

Action Waveforms

13. The circuit controlling the 'write action' and 'read action' signals is shown in fig. 11.4. The delay is provided by a 42-digit line, 32X, which, with twin delay 32U1, normally forms part of the multiplier-divider circuit. A pulse coincident with the setting pulse for the track-selection staticisers, which occurs at ϕ3 of word-time D when any drum-transfer function is decoded, is injected into the 42-digit line, emerging at ϕ3 of word-time D + 1. If the order contains a 'write' function (71 or 73) making F5 positive, this pulse will be able to pass immediately through gate x of 20J2: if, however, a 'read' function (70 or 72) is specified, making ~F5 positive, the pulse cannot pass through delay 20J2, but is fed back to the input of the delay line. Digit delay 32U1 in the feedback loop ensures that the pulse shifts on one digit-time every cycle, so that it eventually emerges from the line at ϕ5 of word-time D + 3, when it can pass through gate y of delay 20J2. (The waveform ~41 applied to delay 32U1 is required for multiplication and division orders; it does not affect the operation of the circuit in this application.)

14. The output of twin delay 20J2 is repeated through delay 20H3 as long as is necessary for the setting of staticiser 20F2. This staticiser is set at ϕ34 for 'write' orders, or at ϕ38 for 'read' orders, during the word-time in which coincidence occurs. It is subsequently reset at ϕ34 or ϕ38 of word-time E, when the transfer operation will have been completed. Its output, X371, the 'drum-transfer action' signal, has three functions to perform: first it is inverted and used to suppress further activity in the delay line and in repeater 20H3; second it is used, as explained in paragraph 16, in the timing circuit that determines the end of the operation; third it is used to initiate the 'read action' or 'write action' signal. It can be seen from fig. 11.4 that the 'read action' signal becomes positive at ϕ40 and the 'write action' signal at ϕ36.

15. The 'write action' signal is normally inhibited by the output of staticiser 12R1 if overflow has occurred. Staticiser 12R1 is set when the 'overflow' signal is positive in coincidence with the output of AND gate 21G1, which represents either of the two 'write' functions, 71 and 73. The output of this staticiser controls an indicator lamp on the programmer's panel and is also mixed with the other internal stops (fig. 9.1) to stop the machine at the end of the order. The staticiser can be cleared by operating the appropriate 'inhibit' switch on the engineer's panel: it is automatically cleared by waveform X18 when the 'start' switch is operated.

16. Writing can be suppressed alternatively by waveform X36, derived from inverter 24K1; this is simply the inverse of waveform X33, representing the 'write' functions,

store can be regarded as magnetic saturation for 3 microseconds in one direction followed by magnetic saturation for 3 microseconds in the opposite direction, a 'nought' being the reverse of this. The phase-modulated waveform is generated, and the phases are split for feeding the push-pull writing amplifiers, by logical elements.

21. Consider first the arrangement in fig. 11.5 for the even digits. The digit waveform is applied to gate *y* of twin delay 36E1, where it is gated by T4 (even ϕ-pulses). As the delay involved in the routing up to this point is 36 digit-times, the timing of the sign digit (or stop-go digit in orders) is ϕ32 (38 + 36 - 42 = 32), so that it is included among the even digits, the parity digit being one of the odd digits. The output of 36E1 is inverted and fed back to gate *x*, which is controlled by odd ϕ-pulses; consequently, if at any time the digit waveform is zero when T4 is positive, there will be no pulse through gate *y* of 36E1, but there will be a pulse one digit-time later through gate *x*. The result is that 'ones' at even positions in the digit waveform will cause a pulse followed by a gap at the output of 36E1, whereas 'noughts' at even positions will cause a gap followed by a pulse. The output of 36E1 is essentially the same as the desired phase-modulated waveform. It is applied via delay 36G1, where it is gated by the 'write action' signal, to one side of the push-pull writing amplifier, while its inverse is applied via 36G2 to the other side. The arrangement for the odd digits is similar, except that the Mill output is gated initially by odd ϕ-pulses. The outputs of delays 36H2 and 36H3 are applied to a second push-pull stage, which feeds the writing head for the odd track.

22. Each digit written on the track consists of a positive and a negative part, and occupies a period of two digit-times, or 6μ sec. Thus the parity digit is written (on the odd track) at ϕ35 and ϕ36, the sign digit (on the even track) at ϕ34 and ϕ35, and the least significant digit (also on the even track) at ϕ38 and ϕ39 of the previous word-time. The 'write action' signal must be positive by the time that the least significant digit of the word appears at the input to delay 36G1 or 36G2, i.e. by ϕ37. In fact, the 'write action' signal is positive from ϕ36 to ϕ35, the odd-digit outputs, 36H2 and 36H3, being gated from ϕ37 to ϕ36. Thus the transition from 'write' to 'not write' on each track comes in the middle of the 6μ sec. period occupied by the appropriate gap digit rather than adjacent to one of the extremities of the digital information.

'Read' Orders

23. The outputs of the read amplifiers (package type 18) are sampled by narrow strobe pulses to eliminate timing variations due to eccentricity of the drum. The resulting waveforms are then lengthened and fed to the input bus via AND gates 20P2 and 20P3 (refer to fig. 10.5), where they are gated with odd or even ϕ-pulses. The mixed outputs of the two AND gates are gated to the input bus with the 'read action' signal and a timing waveform, 41 to 38, which deletes any spurious pulses appearing in the intervals between words. The effect here of the 'read action' signal, which also produces the 'erase' signal for clearing the store lines, is inhibited should the parity of a word have changed during drum storage. The incorrect word is read into the computing store; the remainder of the block of information in the computing store remains unchanged. The full effect of drum parity failure is described in paragraph 30.

W.477

or a supernumerary drum address have been specified in a 'read order'.

Ordinary-Register Blocks

27. In block-transfer orders, the X-digits normally specify a block of ordinary registers. These digits are tapped off the order register at I6, and are fed through gate *y* of delay 34F2 to the second, third and fourth staticisers. As with 'jump' orders, the selection of an ordinary register is normally subsumed under the function in order, and the first N-staticiser can therefore be fed with a waveform derived from the function digits. The output of twin delay 26Y2, which is positive for drum-transfer functions, is therefore fed, like J, via OR gate 34E6 to the first staticiser.

28. The eight words involved in block-transfer orders are routed to or from the drum in eight consecutive word-times. To ensure the correct source or destination in the computing store for each word, the address within the block must be changed by unity each word-time. This is accomplished by the arrangement associated with the last three decoding staticisers. The normal input from the order register through twin delay 34F1 to these staticisers is inhibited, and the staticisers are supplied instead from the output of a half adder comprising elements 34K1, 34J1 and 34H1. The half adder becomes effective when the 'block-transfer-action' signal is positive, which, as can be seen from fig. 11.4, is simultaneous with the 'write action' or 'read action' signal, i.e. at ϕ36 for 'write' or ϕ40 for 'read'.

29. First consider 'block-read' orders. Initially the last three N-staticisers are inactive, making S5, S6 and S7 negative and ~S5, ~S6, and ~S7 positive; thus the first word read from the drum will be transferred to the first register (number 0) in the block. The 'block-transfer action' signal becomes positive at ϕ40: but this does not immediately affect the setting of the staticisers. At ϕ36 in the next word-time, a pulse enters gate *x* of twin delay 34K1; arriving at the output of delay 34H3 at ϕ40, when it causes the last staticiser to become active and, hence, S7 to become positive; thus the next word to leave the drum will be routed into position 1 in the register block. The pulse continues to circulate through delays 34H1, 34H2, and 34H3 until, in the next word-time, it enters into a half-addition operation; the result, a pulse representing the binary number two, causes S6 to become positive instead of S7 so that the next word will be routed into position 2 in the register block.

30. The procedure for 'block-write' orders is much the same, with the difference, however, that the route from the computing store to the drum, which passes through the 35-digit delay in the Mill, is nearly one word-time long. The gates between the N-bus and the Mill are opened as soon as the function decoding is complete (see fig. 11.5); during the waiting period before coincidence has been found, therefore, the word in address 0 in the block is continually being routed into the Mill. As soon as the 'write action' signal becomes positive, the word starts to pass to the drum, and, a few digit-times later, the gates from address 0 can safely be closed and those from address 1 can be opened. The 'block-transfer action' signal for 'write' orders becomes positive at ϕ36; the first digit emerges from twin delay 34K1 immediately after this, so the last N-staticiser is set at ϕ40 of the beat in which coincidence occurs.

31. The waveforms controlling the last three decoding staticisers are applied together to AND gate 34G4, whose output is positive, therefore, when the staticisers are ready to be set for address 7 in the block, showing that the process is nearly

output of delay 21P1 becomes positive; if the handkey is at ON, staticiser 12H2 will become active at $\phi 3$ of word-time D. Nearly one word-time later, at $\phi 0$, staticiser 12J1 will become active, making the inputs to the five punch selector staticisers all positive, thereby causing five holes to be punched to give the 'erase' character. Staticiser 12J1 remains active until the selector bars are locked for the first punching operation, i.e. until the 'busy' signal becomes zero: when this happens, staticiser 12J2 is set at $\phi 0$, and 12J1 is cut off simultaneously. The output of 12J2 is used to gate the first five binary digits of the block address (the three modifier-extension digits and the most significant two N-digits) from the order register to the selector staticisers. These staticisers are set at $\phi 2$ to $\phi 6$; consequently, as can be seen from fig. 6.3, the order register must be tapped at I5. As soon as the 'busy' signal becomes zero again, staticiser 12J2 is cut off, and staticiser 12H1 is set. The output of 12H1 gates the last five N-digits, from I10 in the order register, to the selector staticisers.

36. It will be seen that the first staticiser in fig. 11.8, staticiser 12H2, becomes active during word-time D even if the 'busy' signal is still positive as a result of a previous order. If the input or output equipment is in use at this time, 12H2 will not be cut off until the 'busy' line becomes clear, when staticiser 12J1 can be set. The 'mix' outputs of staticisers 12H2, 12J1 and 12J2 are fed to the coincidence circuit (fig. 11.5) to prevent the coincidence signal, and hence the 'read action' or 'write action' signals, from being generated until the selector staticisers are ready to be set for the third character, otherwise the transfer would be completed and the order erased from the order register before the address could be punched out. There is no need to inhibit the coincidence signal after staticiser 12H1 has become set, as the third character is recorded on the selector staticisers during the next five digit-times. The third character is punched during the last part of the block-transfer order and during several succeeding orders - unless, of course, one of these requires the use again of the input or output equipment.

SUPERNUMERARY ADDRESSES

37. The order structure enables drum addresses up to 8191 (1023.7) to be specified, whereas the highest address on the drum is 5119 (639.7). 'Writing' into supernumerary addresses is permissible; in fact it is often a convenience in programming; 'reading' from non-existent tracks, however, would involve the pickup of random noise, which might or might not give rise to waveforms with correct parity. Supernumerary addresses in 'read' orders are therefore treated as errors, and are made to set the 'drum-parity failure' staticiser, thereby isolating the input bus from the reading equipment and inhibiting the beat counter and half adder. The logical arrangement is shown in fig. 10.6. Supernumerary addresses are defined by S21 & (S22 $_{\vee}$ S23); if the read-action signal is positive as well, digits are injected into the feedback loop of staticiser 12Q2; thereafter the operation is the same as for drum-parity failure. Notice that there is no inhibition on the punching-out of supernumerary addresses.

Tape Character	Value	Parity	Significance		Converted Value	
			LET.	FIG.	Binary	Decimal
000.00	0	0	Figure Shift		10000	16
000.01	1	1	A	1	00001	1
000.10	2	1	B	2	00010	2
000.11	3	0	C	*	10011	19
001.00	4	1	D	4	00100	4
001.01	5	0	E	(	10101	21
001.10	6	0	F	)	10110	22
001.11	7	1	G	7	00111	7
010.00	8	1	H	8	01000	8
010.01	9	0	I	≠	11001	25
010.10	10	0	J	=	11010	26
010.11	11	1	K	-	01011	11
011.00	12	0	L	υ	11100	28
011.01	13	1	M	Line Feed	01101	13
011.10	14	1	N	Space	01110	14
011.11	15	0	O	,	11111	31
100.00	16	1	P	0	00000	0
100.01	17	0	Q	>	10001	17
100.10	18	0	R	⩾	10010	18
100.11	19	1	S	3	00011	3
101.00	20	0	T	→	10100	20
101.01	21	1	U	5	00101	5
101.10	22	1	V	6	00110	6
101.11	23	0	W	/	10111	23
110.00	24	0	X	×	11000	24
110.01	25	1	Y	9	01001	9
110.10	26	1	Z	+	01010	10
110.11	27	0	Letter Shift		11011	27
111.00	28	1	.	. ,	01100	12
111.01	29	0	?	n	11101	29
111.10	30	0	£	Car. Ret.	11110	30
111.11	31	1	Erase		01111	15

The point in this column represents the sprocket hole.

0 = even
1 = odd

TAPE CODE

4. Errors are not sensed automatically; but Input and Initial Orders will generally test the significance of what is read in, and will cause the machine to enter a loop stop if there has been an error. The characters used in programmes, and therefore read in under the control of Input, all have odd parity on tape and converted values below sixteen (except 'carriage return', for which special provision is made by the stipulation that it must always be followed by 'line feed', and by other programming rules). The alteration of a single digit in any of these characters, due to an electrical or mechanical fault, by changing the parity and hence producing a character that represents a number greater than 16, cannot therefore cause one programme character to be mistaken for another. The error shows up immediately.

5. With address-16 input, the five digits on tape, after parity conversion, are placed at the less significant end of the accumulator specified in the order. With address-16 output, the least significant five digits in the specified accumulator undergo parity conversion and the result is punched out as a tape character. When address 17 is specified in an order, there is no parity conversion. Address-17 output causes the least significant five digits in the accumulator to be punched out in their original form: with address-17 *input*, however, the digits on tape are read into digit places 9 to 13 in the accumulator, i.e. into the least significant five 'modifier' places. The principal function of address-17 input is to read warning characters from tape; the digits of these are therefore put automatically into the most convenient position for use as modifiers to select the appropriate indicator (see chapter on Input, Volume IV).

INPUT

Input Signal

6. The holes on the tape are read photo-electrically by a Ferranti high-speed tape reader, and the photocell outputs are fed into number-generator elements as shown in fig. 12.1. Each element will convert up to three constant signals into a series of three consecutive digit pulses; thus one number-generator element can be regarded as performing the converse operation to that of three staticisers. The photocell outputs are first applied to gates, where they undergo an AND operation with the +13V supply to standardise their levels. From here they pass to the number-generators, where they are gated in turn by single pulses at intervals of one digit time; thus the output of the cell reading the least significant holes on the tape passes through gate 11V2x at ϕ23 and ϕ40, the output of the next through gate 11V2y at ϕ24 and ϕ41 and so on. The outputs of the two number generators are mixed to produce a series of five binary digits from ϕ23 to ϕ27 and a similar series from ϕ40 to ϕ2. Gate x of number generator 11V5 is suppressed by a bias of -10V to prevent the appearance of spurious digit pulses.

7. The twin delay that feeds the N-bus, 11S1, is controlled on one side by timing waveform 5-29 and by a waveform specifying register 17, (N02 & N21). On the other side it is controlled by the inverse of the same timing waveform and waveform (N02 & N20), which specifies register 16. Thus the number generated with timing ϕ23 to ϕ27 will pass through delays 11U2 and 11U3, reaching the input to 11S1 between ϕ25 and ϕ29, and will arrive unmodified on the N-bus provided that register 17 (no parity check) has been specified.

'Busy' Stop

12. The 'mix' outputs of staticisers 11Q1 and 11Q2 are applied to output-one element 13Y5, which feeds the 'input busy' indicator on the programmer's panel. The direct outputs are fed, together with a similar signal from the output equipment, to OR gate 11H6, whose output is the 'busy' signal. Provided that the gate preceding delay 26X1 is open, the 'busy' signal, which is mixed with the other stop signals (see fig. 9.1), inhibits the beat trigger and hence prevents the machine from obeying the next order. The gate preceding 26X1 is opened by a combination of order-register outputs, and, as the beat trigger is produced at ϕ40, it is the values of these outputs at ϕ39 that are significant.

13. At ϕ39, order-register outputs W1 and I0 to I2 represent the first four N-digits of an order, and I9 the first F-digit, as can be seen from fig. 6.3. The gate preceding 26X1 will be open, therefore, and the beat trigger will be inhibited until the 'busy' line is clear, if an order enters the order-register in which the first F-digit is zero, i.e. an order that contains any function in the first half of the code (a possible 'read' or 'punch' order), and in which the N-address in binary form is

$$0\ 0\ 1\ 0\ \delta\ \delta\ \delta$$

The only existing addresses that can be of this form (decimal values from 16 to 23) are addresses 16 and 17; consequently the 'busy' stop can operate only on 'read' or 'punch' orders. It will be noticed that the first N-digit is sampled at W1, before the order adder; this is permissible, as the type of modification used in the first half of the function code cannot possibly change the first N-digit from 'nought' to 'one' if the third N-digit is also to be a 1 (i.e. a modifier of 7 or less cannot change a number less than 64 into a number of 80 or more).

14. The order digits are sensed direct from the order-register so that the stop can operate before word-time D of the order. They are sampled first at ϕ39 of CD (for an A-order) or AE (for a B-order); consequently the machine stops in C or A if the next order prescribes reading or punching. The order remains circulating in the order-register, and the beat trigger is inhibited at ϕ40 of every word-time, until the 'busy' line becomes clear. Under normal circumstances, the 'busy' stop will not hold up the operation of the computer for more than one two-hundredth of a second for input or one twenty-fifth of a second for output. The system makes provision, however, for the machine to stop if the input tape over-runs the sprocket-hole. In this case, the sprocket-hole signal will become positive and activate the 'busy' line and the indicator lamp through 11Q1. Staticiser 11Q2 will not, of course, become active, as it can only be set in D. This type of stop can easily be cleared by pulling the tape gently back into alignment.

OUTPUT

15. The standard output unit is a Creed & Co. Model 25 high-speed reperforator, operating at 33 characters per second. Those 'digit' signals from the computer that are positive operate selector coils, which position selector levers below the punching

are properly closed; thus fluctuations in the inverter output due to contact bounce will not affect staticiser 11J2, which cannot be cut off while 11J1 is still active.

18. It should be noted that the 'busy' signal does not prohibit the use of a function such as 11 ($n' = n + x$) to transfer information direct from input tape to output tape. The character would be read during D and set up on the punch selector staticisers during E. The reader and punch would then be turning together, the 'busy' signal being produced by both; though, of course, the reader would stop some time before the punch.

Output Signal

19. The five digits to be punched out are routed (except during block-address punching) from store through the Mill and on to the input bus, where the parity is counted. When address 17 is specified, gate y of twin delay 12D1 (fig. 12.3) is closed, and the five digits pass from the input bus, where their timing is $\phi 0$ to $\phi 4$, through delay 12E1 and gate x of 12D1 to the selector staticisers. These are set individually from $\phi 2$ to $\phi 5$ of word-time E of the order, and remain active during the extension of the beat until the 'busy' line is cleared.

20. When address 16 is specified, the first four digits (in time) on the input bus pass through gate x of 12D1 on to the staticisers. This gate is closed, however, at $\phi 5$, the most significant digit being taken from the input-bus parity circuit (fig. 10.5) through gate y of 12D1. At $\phi 5$, the parity-count waveform represents the parity of the whole five-digit group - '0' for 'odd', '1' for 'even'. Notice that the output parity-conversion operation is identical with the input operation, and is therefore its own inverse.

EXTERNAL CONDITIONING

21. External-conditioning orders, function 74, enable the machine to be switched by means of the programme between several input and output devices. Each of the N-digits in an external-conditioning order corresponds to one of the external devices; if the corresponding digit is a 'one', mechanical relays are set, bringing the device into operation. To enable the relay to become properly set before the device can be used, the order is artificially prolonged by making the machine find coincidence three times between the N-digits of the order and the digits on the address track of the drum. The drum must make two complete revolutions between the setting of the relays and the generation of the E-waveform at the end of the order. The coincidence circuit operates as for drum transfers (see fig. 11.3). For function 74, waveform ~F4 is positive; consequently correspondence will be sought, as for single-word transfers, against the last four N-digits and the three X-digits. (In this case, the X-digits will all be zero).

22. Fig. 12.4 shows the external-conditioning circuits. Staticiser 20W1 is set at $\phi 39$ of the word-time in which coincidence first occurs provided that none of the external equipment is busy, and is not reset until the function waveforms, G07 and G14, become negative at the end of word-time E. The output of this staticiser is used to set the

multiplication takes place during the next thirteen word-times, three multiplier digits being 'paralleled' on staticisers each word-time, beginning at the less significant end of the number. During this period, the circulation time in accumulator 7 is arranged to give three right shifts so that three new digits are brought on to the staticisers each word-time, the digits that have already been sensed being subsequently deleted. The product is formed initially in accumulator 6, which incorporates three full adders arranged in cascade, the digit delays at the outputs of the first two adders ensuring that the three partial products are added in the correct significance to one another and to the number already in the accumulator. The partial products are formed, of course, simply by gating the multiplicand with the appropriate staticised multiplier digit.

5. During multiplication, the circulation time in accumulator 6 is shortened to give three right shifts; thus the partial sum formed in one word-time is shifted down three places to give it the correct significance for adding to the next three partial products in the next word-time. To accommodate the product as it grows in length, the least significant three digits of the partial sum in accumulator 6, which will not be modified by subsequent additions, are shunted at the end of each word-time into accumulator 7, where they replace three multiplier digits that have already been dealt with.

6. The first of the three adders in accumulator 6 can be gated to act as a subtractor. This, as will be shown, is necessary to deal with the sign digit of the multiplier. The sign digit is staticised, and the appropriate subtraction is carried out, during the thirteenth word-time of the multiplication proper, which is referred to as word-time K. Word-time K is followed by word-time L. For functions 20 and 21, L is the last word-time in the order, and is therefore coincident with E. In orders containing function 22, the previous contents of accumulators 6 and 7 must be added to the product. This requires an extra word-time; consequently K, L and E are made consecutive. Multiplication orders are timed by a special counting circuit, which is associated with an auxiliary beat generator that produces waveforms K and L. The E-waveform generator is then triggered at the end of K or L according to the function. During word-time E, the contents of accumulators 6 and 7 are adjusted to take up standard timing; the accumulators then revert to the normal 42-digit-time circulation.

Negative Operands

7. Negative numbers are represented in store as their complements with respect to 2, and it is in this form that the multiplier enters accumulator 7. The binary number $(2 - x)$, which represents a negative multiplier $-x$, can be regarded as the sum of a sign digit, 1, and a fractional part, $(1 - x)$. The product of the multiplicand, n, and the fractional part of the multiplier is therefore $(n - nx)$, whereas the product of the multiplicand and the sign digit is simply n. The *difference* between these two products is therefore $-nx$, which is the desired result. The operation of subtraction will generate the correct sign digit, giving the result $(2 - nx)$ when the multiplicand is positive and the multiplier is negative. If the multiplier, x, is positive, of course, it will have a zero sign digit, and no subtraction will take place.

8. Negative multiplicands can best be considered by regarding the sign digit as the representative of an infinite series of 'carry' digits formed by a previous subtraction operation. While the number is in store, a single 'carry' digit suffices to indicate

Accumulative Multiplication

12. The addition of two double-length numbers implied by function 22 requires two word-times. It is possible, however, to carry out the first half of the addition simultaneously with the addition of the first three partial products. This is done by placing q, the original content of accumulator 7, in accumulator 6 during word-time D, so that it is added on at the less significant end of the product during D + 1. The original content of accumulator 6, p, is meanwhile shunted into the Mill, where it circulates during the whole of the multiplication time. The addition of p to the more significant half of the product takes place in the Mill so that overflow can be detected. During word-time L, which is separate from E for function 22, the more significant half of the product, complete with repeated sign digit, is routed from accumulator 6 into the Mill, where the required addition takes place; the sum is routed back to accumulator 6 during word-time E.

Overflow

13. Overflow can occur in orders containing function 20 or 21 in the special case where -1 (within range) is being multiplied by itself to give +1 (out of range). To test for overflow, the content of accumulator 6 is transferred during word-time L/E to the Mill and is tested in the normal way; this transference does not, of course, affect what is in accumulator 6. Notice that rounding in multiplication can never cause overflow (see Chapter I, paragraph 22).

CONTROLS

Timing

14. The timing of multiplication orders is controlled by the results of a test made on the content of nickel-line register 32X, shown in fig. 13.2. This register is used, too, for division orders, and also to give a delay in drum-transfer orders (see Chapter XI). In multiplication orders, the register is fed from staticiser 32U2, which is set at $p29$ of word-time D (M3 is positive for all multiplication orders during D only). It is reset at $p41$; so its output is a sequence of 12 digit pulses from $p30$ of D to $p41$ of D + 1. These enter the nickel line, emerging one word-time later with the same timing, and circulate through the line via gate y of delay 32U1 during multiplication; the effect of circulation is to delay the pulse group by one digit-time each word-time, and the effect of ~41 on 32U1y is to delete the least significant pulse in the group each cycle.

15. The K-staticiser is set as soon as the output of AND gate 31U4 is positive at $p40$. This will be at the end of word-time D + 12, when the pulse group in the nickel line will have been reduced to a single pulse emerging at $p41$, the output of the line at $p40$ being zero making the output of inverter 32Q2 positive. It will be noticed that the output of the inverter is positive also at $p40$ of word-times D, L, E and K; however, the waveforms on AND gate 31U4 prevent the emergence of a pulse during word-times D, L and E, and waveform ~K on the staticiser prevents its being set at the end of word-time K. Waveform X51 on AND gate 31U4 is positive only during multiplication or division orders (see fig. 13.3); so the output of the AND gate cannot become positive during any other type of order.

16. The L-staticiser is set at $\phi 40$ of word-time K and remains active for one word-time. It will be seen from fig. 5.5. that the E-staticiser is set at the same time during multiplication orders except for orders containing function 22 (G02 & G12). In this case the E-staticiser is set at the end of L. Note that it is necessary to apply ~E to the setting gate of the E-staticiser; if this were not done, the E-staticiser would be set twice in multiplication and division orders (except for function 22), first at the end of K and again at the end of L/E.

Gating

17. Fig. 13.3 shows how the gating waveforms that control the multiplication process are derived. A combination of ~F0, F1 and ~F2 with ~(F4 & F5) in AND gate 21G4 gives waveform X50, which is positive for functions 20, 21, 22, 24, 25 and 26, i.e. for all multiplication and division functions. X50 is combined with ~F3 (first half of function group) at gate *y* of staticiser 31D1, which is reset at L40; its output, M1, is therefore positive for multiplication orders from D1 (the F-waveforms are not significant until D0) to L40. M1 is then combined with various timing and rhythmic waveforms to give the control waveforms peculiar to multiplication.

CIRCUIT DETAILS

18. The multiplicand register and the parts of accumulator 6 concerned with multiplication are shown in fig. 13.4, the adder-subtractor and adders being shown in detail in fig. 13.5. The parts of accumulator 7 concerned with multiplication are shown in fig. 13.6. Two of the timing charts refer to multiplication, and it is recommended that these be studied in conjunction with the remainder of this chapter.

Function 20

19. Consider first the operation of the multiplier for function 20, 'multiply n by x to give a double-length product.' During word-time D, waveform X58 becomes negative, clearing the multiplicand register and the two accumulators of their previous contents. It becomes negative at $\phi 1$, one digit-time after the F-waveforms become significant; consequently it must be applied at points in the accumulators where the timing is delayed one digit-time on the standard - before delay 30S2 in accumulator 6, and before delay 31L1 in accumulator 7. As waveform X58 is actually positive at D41 and D0, it must be combined at the feedback gate of the multiplicand register with timing waveform 1 to 40; otherwise positive digits that might enter, or be generated in, the multiplicand register at $\phi 41$ and $\phi 0$ would never be erased, and would appear at the output with the timing of repeated-sign digits.

20. During D the multiplicand is routed into its register and the multiplier is routed into accumulator 7. The multiplier enters through delay 31Q3, where it is gated by waveform M3 from $\phi 1$ to $\phi 40$ of D. It comes from the X-bus via the sign repeater; as the repeated sign appears at $\phi 41$ (see fig. 7.1), it does not enter accumulator 7, the sign repeater being used here simply as an extra delay to give the correct timing.

21. At the end of word-time D, waveform M4 becomes positive, and waveform M5 becomes zero. These two waveforms are applied to two gates preceding twin delay 32R1 in accumulator 7, and serve to reduce the length of the loop from 42 digits to 39 digits,

giving the desired three right shifts. The multiplier appeared at the output of the sign repeater from $p2$ to $p40$ of D, and appears, therefore, at gate y of twin delay 32R1 from $p40$ of D to $p36$ of D + 1. M4 and the timing waveform (41 - 37) on this gate become positive together at $p41$ (the beginning) of D + 1; consequently, the least significant multiplier digit ($d38$) is deleted and the rest of the word is retained. Digits $d36$, $d37$ and $d38$ are distributed along the inputs to the three staticisers at $p0$ of D + 1. The staticisers are set at $p0$ by waveform M7 (see fig. 13.3); so their outputs, M8, M9 and M10, are significant from $p1$ to $p40$, $p41$ and $p0$ respectively.

22. M8, M9 and M10 are used to gate the multiplicand into the 'b' input of the adder-subtractor and the two adders in accumulator 6. (It will be seen from fig. 13.3 that the adder-subtractor will add during all word-times except K). The timing of the multiplicand at the entries to the adder gates is $p1$ to $p0$, sign digits appearing from $p39$ to $p0$. As M8 is zero at $p40$, only two sign digits enter the first adder; moreover, any 'carry' in the adder beyond the second sign digit is prevented by the suppression waveform ($p1$ to $p40$) applied to the carry delay. The multiplicand entering the second adder contains three sign digits, and that entering the third adder contains four. The delay of one digit-time in the first two adders ensures that the three partial products are added in echelon, the first partial sum appearing at the output of delay line 30D with timing $p37$ of D + 1 to $p36$ of D + 2 and with the most significant digit indicating the sign.

23. The partial sum circulates in accumulator 6 via AND gate 31M4; but the least significant three digits are prevented by the timing waveform from passing through this gate, and are shunted instead through delay 32S2 to accumulator 7, appearing at gate y of delay 31P1 (fig. 13.6) from $p39$ of D + 1 to $p41$ of D + 2. The sign digit of the first partial sum in accumulator 6 appears at the output of delay 30Q2 at $p38$ of D + 2, and is then repeated three times, so that it will 'line up' with the sign digits of the partial products formed in the next word-time. The sign repeater (delay 30Q2) and the entry gate to accumulator 7 for the last three product digits (31P1y) are both controlled by waveform X61. This is positive from $p2$ of word-time D + 1 to $p0$ of word-time L, except at $p1$ of each word-time. The 'gap' at $p1$ is a fortuitous result of the way in which X61 is produced; the waveform produced by delay 31Q1 (see fig. 13.3.) is designed for use in division, but is suitable as a source of (~D & ~L) in this application.

24. The multiplier digits remaining in accumulator 7 ($d37$ to $d0$) appear at the output of delay 31P1 from $p1$ to $p38$ of D + 1 and are followed from $p40$ of D + 1 to $p0$ of D + 2 by the last three product digits. At $p0$ of D + 2, multiplier digits $d35$ to $d33$ are distributed along the staticiser inputs, and, one digit-time later, outputs M8, M9 and M10 become significant again. During word-time D + 2, the timing waveform, $p41$ to $p37$, on delay 32R1 (accumulator 7) allows multiplier digits $d34$ to $d0$ to pass through, followed by one gap digit and the three product digits just shunted from accumulator 6. What happens in accumulator 7 during D + 2 is similar to what happens during D + 1; this process, and that occurring in accumulator 6, is repeated ten times more, from D + 3 to D + 12.

25. During word-time D + 12, the most significant five multiplier digits, $d4$ to $d0$, appear at the output of delay 31P1 from $p1$ to $p5$, the rest of the digits having been deleted during preceding circulations. These five digits are followed by a gap digit ($p6$) and then by 36 product digits that have been shunted from accumulator 6, the last

two (most significant) of these appearing at the output of 31P1 at $p41$ and $p0$ of word-time K. The last three multiplier digits ($d2$ to $d0$) are staticised at $p0$ of word-time K, the sign digit ($d0$) appearing at M8 from $p1$ to $p40$. During word-time K, the adder-subtractor is set to subtract (see fig. 13.3), and the final partial sum appears at the output of delay line 30D from $p37$ of K to $p36$ of L/E. During $p39$ and $p40$ of K, two more product digits are shunted into accumulator 7; at $p41$, however, waveform ~L is negative, so gate y of delay 31P1 is closed. At $p41$ of L/E, therefore, accumulator 7 contains 38 product digits; these appear at the output of 31P1 from $p4$ (K) to $p41$ (L/E). and are preceded by the two remaining multiplier digits ($d1$ and $d0$) at $p1$ and $p2$ of K.

26. Waveform M4 remains positive during word-time L/E, giving three more right shifts in accumulator 7 to put the less significant half of the product into standard timing. The remaining two multiplier digits reach gate y of delay 32R1 at $p38$ and $p39$ of K, and are therefore deleted; the product digits reach the output of delay 32R1 from $p0$ to $p37$ of word-time L/E, when they are in standard timing. M4 becomes zero, and M5 becomes positive again, at $p41$ of the first word-time following the multiplication order, as can be seen from fig. 13.3.

27. The digits of the final partial sum appear at the output of delay line 30D in accumulator 6 from $p37$ (K) to $p36$ (L/E); two of these are shunted into accumulator 7 via delay 32S2, the remainder being the more significant half of the final product. The most significant two digits emerging from the delay line represent the sign, and should be identical except in the special case of overflow ($-1 \times -1 = +1$). The 'multiply' (right-shift) circulation in accumulator 6 is inhibited at AND gate 31M4 as soon as the L-waveform becomes positive, and the normal circulation gate, 31S2y, is opened by waveform X59 as soon as the E-waveform becomes positive, i.e. at the same time as the L-waveform for function 20. The timing waveform on this gate (41 - 37) ensures that the repeated sign digit is deleted; the word emerging from delay 31S2 then appears from $p0$ to $p38$, which is standard timing.

28. The equipment in the Mill is used to test for overflow during word-time L/E. The normal-circulation loop in accumulator 6 is tapped between delays 32S2 and 31S2, and the more significant half of the product is led off to the Mill through delay 32S1 (the timing waveform ~39 gating this delay is required in left shifts). The parts of the Mill concerned with multiplication orders are shown in fig. 13.7. The word enters from accumulator 6 through twin delay 32V1 and gate y of twin delay 32V2, which is opened by waveform M4. Sign repetition in 32V1, which is necessary for function 22, is not operative during L/E. The Mill tests for overflow in the normal way (see Chapter VII). Transfer to the Mill does not, of course, affect the content of accumulator 6, and further circulation in the Mill is inhibited by the waveform (~D & ~E) on delay 21P3. Notice, however, that the overflow staticiser is set, if it is to be set, at D39 of the *next* order for function 20 (or 21).

Function 21

29. The process is practically the same for function 21 as for function 20, the only difference being the insertion of a rounding digit into accumulator 6 during word-time D. This digit is fed into the multiplication loop through gate y of twin delay 31P2

at $p38$ of D (M3) for function 21 (M3 & G11), and enters the adder-subtractor at $p40$. During D + 1 it makes one circulation of 39 digit-times in the accumulator, entering the adder-subtractor again at $p37$, when it will have the correct timing.

Function 22

30. Function 22 requires the product to be added to the double-length number $[pq]$ contained in accumulators 6 and 7 at the beginning of the order. The more significant half, p, which is added on in word-time L (separate from E), is stored in the Mill while the multiplication is taking place. It is fed out of accumulator 6 during D through delay 32S1 (fig. 13.4). (Waveform ~39 on this delay is effective only in shift orders.) The sign of p is repeated in twin delay 32V1 (fig. 13.7) in preparation for the addition, and the word then passes through gate x (G02 & G12) of twin delay 32V2 and through two more delays to the 'b' input of the Mill adder-subtractor. Its timing at this point is $p4$ of D to $p1$ of D + 1, which is normal for Mill operands. It circulates in the Mill through gate x of twin delay 21D2 and the adder-subtractor 'a' input.

31. The less significant half, q, of $[pq]$ is fed from accumulator 7 into accumulator 6 during D, and is added in with the first three partial products. Accumulator 7 is tapped for this purpose at the output of twin delay 31P1, where q appears from $p1$ to $p38$ (neglecting any sign digit). The word is fed into accumulator 6 through gate x of twin delay 31P2 during D (M3), and enters the adder-subtractor from $p3$ to $p40$ (again neglecting sign digits). During D + 1 it circulates with three right shifts, entering the adder-subtractor again from $p0$ to $p37$, when it will have the desired timing. Note that there is an automatic 'justification' of $[pq]$ in function 22 as long as q is in range, i.e. if q carries a positive digit in the sign position, q will be added on to the partial product, and hence to the final product, as a negative number. The sign of q is repeated three times during D + 1 in the same way as the sign of the partial sum. The sign repeater is of course inoperative during D, as waveform X61 is then negative.

32. Multiplication proceeds in the same way for function 22 as for functions 20 and 21 until the end of word-time K. In the next word-time for function 22, waveform L is positive but E is still zero; consequently, the number in accumulator 6, which is the more significant half of the product - possibly modified by 'carry' from the addition of q to the less significant half - can neither circulate normally through delay 31S2 (X59 is zero) nor with three right shifts through AND gate 31M4 (~L is zero). Instead, it passes out to the Mill through 32S1. (It will be remembered that it already has a repeated sign.) It enters the 'b' input to the adder-subtractor at the same time as p enters the 'a' input from the circulation loop. It will be seen from fig. 7.2 that the adder-subtractor in the Mill is set to add for any function in group 2; if overflow occurs during this operation, it will be sensed in the usual way. The resulting sum, p', is routed back to accumulator 6 from the normal Mill output point through delay 31V3 (fig. 13.4) during word-time E. The timing waveform (41 - 37) on delay 31V3 ensures that the second sign digit is deleted. Further circulation in the Mill is inhibited by the gating waveform on delay 21P3.

CHAPTER XIV

DIVISION

CHAPTER XIV

DIVISION

$$\left.\begin{array}{l}24.\\25.\end{array}\right\}\quad \frac{x + 2^{-38}q}{n} = q' + \frac{2^{-38}p'}{n} \qquad \begin{cases} q' & \text{unrounded} \\ q' & \text{rounded} \end{cases}$$

$$26.\quad \frac{x}{n} = q' + \frac{2^{-38}p'}{n} \qquad q' \quad \text{rounded}$$

BASIC PRINCIPLE

1. The division orders in Pegasus enable a double-length number to be divided by a single-length number to give a single-length quotient, unrounded (function 24) or rounded (function 25), and a single-length residue; or a single-length number to be divided by another single-length number (function 26) to give a single-length rounded quotient and a single-length residue. The divisor is specified by the N-address of the order, and is fed during word-time D of the order into the divisor register - the register that is used for the multiplicand in functions 20 to 22. The X-digits of the order give the address of the more significant half of the dividend, which is fed into accumulator 6 during D, the less significant half, q, of the dividend having been placed in accumulator 7 by a previous order. For function 26, accumulator 7 is cleared during D. The quotient, q', is formed in accumulator 7, and gradually replaces dividend digits, which are shunted one-by-one into accumulator 6 as the division proceeds. The number, p , remaining in accumulator 6 at the end of the operation is the residue, effectively shifted up 38 places. For rounded division (function 25 or 26) its range is given by

$$-1/2 \leqslant p'/n < 1/2$$

i.e. its numerical value is as small as possible. For unrounded division (function 24) it has the same sign as the divisor, and its range is given by

$$0 \leqslant p'/n < 1$$

Because the unrounded residue has the same sign as the divisor, the division can be continued with a subsequent order to give a *double*-length quotient, the less significant half of which will necessarily be positive, i.e. to give a double-length *justified* quotient.

2. The principle of the division process was explained in Chapter I. The number in accumulator 6 is to be made numerically as small as possible by repetitively adding or subtracting the divisor, each addition or subtraction being carried out with a significance one place lower than that of the previous addition or subtraction. At the end of the process, the total amount that has been subtracted from the dividend is the

nearest multiple of the divisor that can be 'contained' in it. The quotient is therefore simply a record of the number of subtractions that have taken place. Subtracting the divisor digits from the number in accumulator 6 implies that the divisor is 'contained' in the dividend with the corresponding significance; a 'one' must then be *added* to the quotient in accumulator 7 in the corresponding place. Similarly an addition in accumulator 6 requires a 'one' to be subtracted from the quotient.

3. The numbers entering into arithmetical operations are presumed to be in the range $-1 \leqslant n < 1$. If the divisor is numerically smaller than the dividend, therefore, or if the two are equal, the quotient will be out of range. It is arranged for this condition to be sensed, and for the overflow staticiser in the Mill to be set if an 'out-of-range' quotient is formed. As the dividend can exceed the divisor by as little as 2^{-76}, the range of the quotient cannot be known for certain until the division process is completed; moreover, an out-of-range quotient can be produced by rounding. The overflow staticiser is set, therefore, during word-time E of the order.

4. If the divisor and dividend have the same sign, the former must be *subtracted* from the latter to give the minimal residue after the first step; if the signs are different, the operation must be addition. The sign of the divisor must be compared with that of the residue in accumulator 6 after each operation so that the adder-subtractor control can be set for the next operation. The quotient is formed digit by digit in accumulator 7 by a half adder-subtractor - a full adder-subtractor is not necessary, as only one digit is inserted each word-time. This must be set for addition when the full adder-subtractor in accumulator 6 is set for subtraction, and for subtraction when the full adder-subtractor is set for addition.

5. The operands are routed into the accumulators during word-time D, their signs are compared, and, during word-time D + 1, the first addition or subtraction takes place in accumulator 6. At the same time, there is an addition or subtraction of a 'one' in the 'sign' position of the quotient in accumulator 7. Circulation in the two accumulators during D + 1 requires 43 digit-times, giving one left shift, so that the next addition or subtraction can take place with a significance one place lower. One digit is then shunted from accumulator 7 into the least significant place in accumulator 6. Additions or subtractions continue in this way until D + 39, when all the dividend digits will have been 'used up', and a quotient 39 digits long (including the sign) will have been formed in accumulator 7.

Terminating Procedure

6. At every stage from D + 1 to D + 39, the least significant quotient digit is necessarily a 'one', as it is formed by adding or subtracting a 'one' in a place originally occupied by a 'zero'. The value of the last quotient digit is therefore undetermined in word-time D + 39, and the division must be carried one stage further (D + 40) before its true value can appear. In D + 40 the quotient is correct for an unrounded result; but it carries a surplus 'one' at its less significant end. In unrounded division, this digit can be removed simply by subtraction, which is done in word-time K.

7. In rounded division, the true value of the surplus digit must be known, as it may influence the value of the rounded quotient; consequently, the sign digits of the

divisor and residue must be compared in the usual way to determine whether the *next* operation would have been addition or subtraction. If subtraction, the surplus digit would become a 'nought'; the last digit of the quotient proper would then be unaltered by rounding: if addition, the surplus digit would have remained a 'one'; rounding would then have required an extra 'one' to be added in the last place of the quotient proper. Rounding therefore requires an extra addition or subtraction, made according to the usual rules, but carried out still with the significance of the surplus 39th quotient digit, i.e. without the usual one-place shift; the surplus digit will then always be reduced to zero, and 'carry' from it will effect the rounding.

8. The residue in accumulator 6 should be the difference between the dividend and an exact multiple of the divisor. If the quotient is to be a true record of the amount that has been subtracted from the dividend, the extra operations in accumulator 7 during word-times D + 40 and K must be accompanied by exactly inverse operations in accumulator 6; thus there must be no left shift in accumulator 6 during word-time K. Word-time E (which coincides with L in division) enables the timing of the words in the two accumulators to be brought back to standard; this entails one right shift in accumulator 6, but circulation in accumulator 7 remains normal. It will be seen that division requires, in all, 43 word-times.

Zero Residues

9. An attempt to divide by zero is detected by its effect on the overflow staticiser (paragraph 3); but division into zero, a legitimate operation, requires special provisions. The quotient in such a case is obviously zero, and must be built up in accumulator 7 according to one or other of the following series:

$$0 = -1 + 1/2 + 1/4 + 1/8 + \ldots\ldots\ldots\ldots\ldots + 2^{-37} + 2^{-38} + 2^{-38}$$

$$0 = +1 - 1/2 - 1/4 - 1/8 - \ldots\ldots\ldots\ldots\ldots - 2^{-37} - 2^{-38} - 2^{-38}$$

In rounded division, it will not matter which of the two methods is used; the quotient is bound to be correct. In unrounded division, however, the last operation (word-time K) in accumulator 7 is always subtraction (see paragraph 6) and the second series is therefore the one to be applied. To ensure this, the first operation (word-time D + 1) must be made addition in accumulator 7 and, consequently, subtraction in accumulator 6. Now, in the Pegasus system, zero is treated as a positive number - sign digit zero; hence, with a negative divisor, the normal sign sensing would prescribe addition in accumulator 6 and subtraction in accumulator 7 during D + 1. A special circuit is provided, therefore, to test whether the dividend is zero, and to force the machine to subtract in accumulator 6 and to add in accumulator 7 when a zero dividend is found. Zero dividends will occur rarely and could be covered simply by suppressing the whole division operation. However a similar situation will obtain when the divisor divides exactly into the dividend to leave a zero residue at some stage in the process. The net result of further operations must then be to leave unaltered the quotient and residue evolved so far.

Overflow Sensing

10. If the dividend is numerically greater than the divisor (quotient out of range), the first operation in accumulator 6, whether addition or subtraction, will leave

unaltered the sign of the number in this accumulator. Thus for out-of-range division, the first two operations prescribed will be of the same type. To test this, an extra 'one' is placed in accumulator 7 during word-time D in a position one place more significant than the position that will be occupied by the sign digit of the quotient, i,e. in such a position that it will represent the number 2. After two similar operations in accumulator 7 (out-of-range quotient), this accumulator will contain

$$\text{either} \quad 2 + 1 + 1/2 = 3\ 1/2 = 11.1$$

$$\text{or} \quad 2 - 1 - 1/2 = 1/2 = 00.1$$

After two non-similar operations (in-range quotient), accumulator 7 will contain

$$\text{either} \quad 2 - 1 + 1/2 = 1\ 1/2 = 01.1$$

$$\text{or} \quad 2 + 1 - 1/2 \quad 2\ 1/2 \quad 10.1$$

11. Thus the first two quotient digits in accumulator 7 at the end of word-time D + 2 are the same for an out-of-range quotient, and different for an in-range quotient. The only way in which these two digits can become modified subsequently is by rounding in word-time K, when it is possible for an in-range quotient to be taken out of range, of for an out-of-range quotient to be brought back into range, by adding or subtracting 2^{-38}. The first two quotient digits are therefore tested for equivalence during word-time E, and, if equivalence is found, the overflow staticiser in the Mill is set. The extra test digit is gated out during E after the test. It should be noted that the sign digit remaining in accumulator 7 will give the sign of the quotient correctly for any in-range division operation.

PROCESS DETAILS

12. The division process is illustrated schematically in fig. 14.1. During word-time D, the divisor is fed from the specified N-address into the divisor register - this is the register that was used for the multiplicand; it is controlled for division by the same waveforms that control it for multiplication. Also during word-time D, the overflow-test digit is inserted through the half adder-subtractor into accumulator 7. As well as enabling overflow to be sensed easily, this digit serves to prevent the generation of a series of 'carry' digits by the first true quotient digit, should this be subtracted, there being no carry suppression on the half adder-subtractor. The more significant half of the dividend is fed from the X-bus into accumulator 6 during D, and, for functions 24 and 25, but not function 26, undergoes a justification operation in the second adder, the sign of q, the less significant half of the dividend, having been set on a special staticiser in accumulator 7 at the end of E of the previous order. For function 26, accumulator 7 is cleared during D.

13. The division proper starts in word-time D + 1, and continues until D + 40, i.e. until the process has gone one place beyond the least significant digit of the dividend. Circulation in the two registers gives one left shift per word-time during this interval, and one digit is shunted from accumulator 7 into accumulator 6 at the beginning of each word-time from D + 1 to K. It will be seen from fig. 14.1 that the shunted digit

appears in accumulator 6 one word-time before it is used; consequently the last dividend digit appears in accumulator 6 in D + 38. This is followed in D + 39 by a gap digit (required for the extra addition or subtraction in D + 40); and in D + 40 and K by the test digit and the sign digit of the quotient, which are subsequently deleted. The circulation time in the two registers is normal during K, as required by the algorithm. The quotient is now in standard timing; but one right shift is needed in accumulator 6 during E to bring the timing of the residue back to standard. The quotient is tested for overflow during E, and then the test digit is deleted.

14. The fortieth residue, p_{40}, at the end of D + 40, is in the range

$$-2^{-39}n \leqslant p_{40} < 2^{-39}n.$$

During word-time K in unrounded division, $2^{-39}n$ is added to this residue to give a final residue in the range

$$0 \leqslant p' \leqslant 2^{-38}n$$

This operation implies overflow. Overflow in accumulator 6 can occur in other word-times in the special case where a zero residue is being divided by -1; here the operation required is subtraction (see paragraph 9), making the next residue +1, which is out of range. In view of these facts, the operands entering accumulator 6 must carry at least two sign digits. In fact, to effect some economy of equipment, three sign digits are used (it will be remembered that multiplication requires a triple sign repetition at the output of the multiplicand-divisor register); but carry suppression in the adder-subtractor makes the most significant digit of each residue meaningless. This digit is removed during circulation in the accumulator.

CONTROLS

Timing

15. The timing of division orders, like the timing of multiplication orders, is controlled by a test made on the content of register 32X (refer to fig. 13.2). During word-time D of division orders, a series of 39 pulses enters gate x of twin delay 32U1 from $p1$ (when Q1 becomes positive) to $p39$. These emerge from the nickel line from $p2$ to $p40$ of D + 1, and, circulating through gate y of 32U1, emerge from the line again from $p3$ (D + 2) to $p41$ (D + 3). During the next circulation and subsequently, the series of pulses is shifted up one place in significance, and the most significant pulse of the group is deleted, each word-time. At $p40$ of D + 40, no pulse appears at the output of the line; consequently the K-staticiser is set. As with multiplication orders containing function 20 or 21, the E-staticiser (fig. 5.5) and the L-staticiser are set together at $p40$ of K.

Gating

16. The generation of the main gating-control waveforms for division is shown in fig. 14.2, with the exception of the adder-subtractor controls, Q4 and Q5, which appear in fig. 14.5. These waveforms are produced in the first instance by combinations of the appropriate F-waveforms. Waveforms X51 and X58 are common to multiplication and

division. Waveforms Q1 and Q2 (division only) are positive from $p1$ of D - one digit-time after the F-waveforms become significant; combined with timing and rhythmic waveforms, they perform the gating operations peculiar to division. Waveform X59 controls normal circulation in accumulator 6, and waveform X60 controls normal circulation in accumulator 7.

CIRCUIT DETAILS

17. Fig. 14.3 shows the components of accumulator 6, and fig. 14.4 the components of accumulator 7, concerned with division. Two of the timing charts refer to division, and it is recommended that these be studied in conjunction with the rest of this chapter. The adder-subtractor and the second adder in accumulator 6 are used, (refer to fig. 13.5 for the complete circuits of these), and accumulator 7 incorporates a half adder-subtractor for building up the quotient digit-by-digit. The sign digit of the word in accumulator 7 will have been 'remembered' on staticiser 32T1 (fig. 14.4) from $p40$ of word-time E of the previous order. This digit is therefore available during word-time D of any division order so that the dividend can be justified for sign. The staticised sign digit is fed into one input of the second adder in accumulator 6 during the whole time that x, the more significant half of the dividend, is being fed into the other input. The result is that -2^{-38} is added to x when q is negative; the double-length number $[xq]$ is then correctly $x + 2^{-38}q$, whether q is positive or negative, provided that it is in range. Notice that this operation is suppressed for function 26 (F4 positive).

18. Normal circulation in accumulator 6 is inhibited by waveform X59 on gate y of twin delay 31S2, which becomes negative at the end of D. The previous word is erased by waveform X58 on twin delay 30S2. The normal-circulation gate in accumulator 7, gate x of twin delay 31L1, is closed by both X58 and X60 from $p1$ of D, and the left-shift gate, gate x of twin delay 31K1 is opened by Q2 at the same instant - except for function 26 (F4 positive), when both circulation loops are broken during D and the accumulator is cleared.

19. The test digit is inserted into accumulator 7 during D through the half adder-subtractor, which is set to add (see paragraph 27), and emerges from nickel line 31y at $p37$ - three digit-times after the sign digit of q, and two digit-times before the least significant digit of q. After emerging from the nickel line, the contents of accumulator 7 pass through three delays and through gate x of twin delay 32R1. This gate is controlled by M5, which can be negative only in multiplication, and by ~39, which is used for deleting the digits of q one-by-one, starting with the sign digit in word-time D + 2.

20. The sign of the number in accumulator 6 is sampled at Y12 after the second adder (fig. 14.3), and is compared with the sign digit of the divisor (as explained in paragraph 24) to determine the setting of the adder-subtractor and the half adder-subtractor. During D, it is the third sign digit that is sampled; during succeeding word-times, owing to the change in loop length, it is the second. The digit of q following the sign is transferred from accumulator 7 (from the output of delay 31L2) at $p40$ of D and enters accumulator 6 through gate y of delay 31T1 at $p41$ (the beginning) of D + 1, one word-time before it is required. The gating waveform here is Q2, which

is positive from $p1$ of D to $p40$ of D + 40. The justified x reaches the output of twin delay 31S1, in accumulator 6, via the left-shift loop, from $p0$ to $p40$ (D + 1); it is therefore preceded immediately by the single digit transferred from accumulator 7. This digit enters the adder-subtractor at $p1$, and x enters from $p2$ (D + 1) to $p0$ (D + 2), in step with the divisor.

21. The divisor enters the adder-subtractor in accumulator 6 through gate y of 30P1, the timing waveform on this gate ensuring that it carries only three sign digits. The residue proper of the first operation emerges from the adder-subtractor from $p3$ (D + 1) to $p0$ (D + 2); it is preceded by the last digit transferred from accumulator 7, and is followed by a meaningless 'third sign' digit, which is subsequently deleted at $p39$ before delay 32S1 in the left-shift loop. The processes occurring in D + 1 are repeated from D + 2 to D + 40, an extra digit being transferred from accumulator 7 each word-time. In D + 2 and subsequently, this digit appears immediately after (in time) the final sign of the previous residue (see the timing chart). It is because of this that 'carry' in the adder-subtractor must be suppressed for two digit-times (see fig. 13.5). Suppression at $p41$ is for 'carry' from the second (true) sign digit of the divisor, and 'carry' from the spurious 'third sign' digit must be suppressed at $p0$ to prevent modification of the dividend digit just transferred from accumulator 7.

22. The least significant dividend digit enters accumulator 6 at $p41$ of D + 38. This is followed in D + 39 by a gap digit - necessary for the extra operation in D + 40 - and in D + 40 by the test digit of the quotient. This test digit is removed from accumulator 6 at $p41$ of K by the change-over from left-shifting to normal circulation; it appears at the input to twin delay 31S2 at $p40$, before waveform K is positive, and at the input to 31S1 at $p41$, after waveform Q2 has become negative. At $p41$ of K, the sign digit of the quotient is transferred into accumulator 6. This digit actually remains in accumulator 6 for two full word-times; it is gated out during the right shift in word-time E by the timing waveform on gate x of 31T1, although it appears at the input of 31S2 at $p40$, and at the input of 31S1 at $p41$ of the next beat.

23. The sign digit of q, the word originally in accumulator 7, is deleted at $p39$ of D + 2 (except, of course, for function 26, when the whole word is deleted in D); its least significant digit is deleted at $p39$ of D + 40, to leave only the quotient, with its test digit and with a surplus 'one' at its less significant end. During K, circulation in accumulator 7 is normal, and the half addition or subtraction reduces the fractional part of the quotient to the normal length. At $p37$ of E, the test digit and the sign digit of the quotient, which are at the outputs of delays 32P2 and 32P3 respectively, are tested for equivalence. If they are both positive, or if they are both zero, the quotient is out of range; the output of AND gate 32M2 becomes positive, and the overflow staticiser in the Mill is set (see fig. 7.2). As for Mill overflow, the overflow-indicator lamp lights, and the machine stops if a subsequent order attempts to transfer information to the main store. If it is not known which of the two operands in a division order is the greater, the usual practice is to follow the division order with a 'jump' order that will clear the overflow staticiser and take appropriate action if overflow has occurred.

Adder-Subtractor Control

24. The generation of waveforms Q4 and Q5, which control the adder-subtractor in accumulator 6 and the half adder-subtractor in accumulator 7, for division is illustrated

in fig. 14.5. The full adder-subtractor must be set to add, and the half adder-subtractor to subtract, when the signs of the divisor and the residue are not equivalent. Under this condition, staticiser 30X2 is set, its output being positive from $\phi 2$ to $\phi 0$. The sign of the divisor is sensed off the N-bus (after the sign repeater, where it has the desired timing), and is 'remembered' on staticiser 30X1 until the end of the division order, i.e. until D of the next beat. The sign digit of the residue appears at Y12, the output of the second adder in accumulator 6, at $\phi 1$; and it is fed, with the staticised sign of n, into a NOT-EQUIVALENT circuit. Provided that the output of staticiser 30S1 is positive, the output of AND gate 30W4 will be positive at $\phi 1$ of any word-time in a division order if the signs of divisor and residue are different. In unrounded division (function 24) the full adder-subtractor must add, and the half adder-subtractor must subtract, in word-time K, regardless of the normal sign sensing. This condition is satisfied by applying waveforms G14 and K to one gate of twin delay 31W1.

25. The full adder-subtractor must subtract, and the half adder-subtractor must add, initially if the dividend is zero or if the residue falls to zero during the division process. The two parts of the dividend or residue are tested simultaneously, and a positive digit in either sets staticiser 30S1, the output of which controls the add-subtract staticiser through AND gate 30W4. The testing process is rather complex, and can best be followed by reference to the timing chart. The less significant part of the residue in accumulator 7 is applied to gate x of staticiser 30S1. As this part is gradually being reduced in length and is being replaced in accumulator 7 by the quotient, it is gated by the digit-pulse group used for timing, which is being reduced at the same rate and will therefore mask out the quotient digits in accumulator 7. All the residue digits are sensed except the most significant, which will have been transferred to accumulator 6 two word-times previously and will have entered into an addition or subtraction operation, and the sign digit, which will already have been dealt with by justification. Staticiser 30S1 will therefore be set, unless the digits sensed from accumulator 7 are all zero, by $\phi 40$ of D or by $\phi 41$ of D + 2 to D + 39, and will be reset at $\phi 2$ of the following word-time. Note that the single dividend digit remaining in accumulator 7 during D + 40 is not sensed.

26. The more significant part of the dividend or residue is sensed at the output of the second adder in accumulator 6, and is mixed into the feedback loop of staticiser 30S1. The feedback gate is open except at $\phi 2$ of D and at $\phi 2$ and $\phi 3$ of any other word-time. As can be seen from the timing chart, this enables all the digits in accumulator 6 to be sensed, except for the meaningless 'third sign' digit and the digit last transferred from accumulator 7 (which will still form part of the group in accumulator 7, and will be sensed as such). The second sign digit in accumulator 6 reaches the feedback gate at $\phi 1$ and emerges from 30S1 at $\phi 2$, too late to affect the add-subtract control. This digit can be different from the digit preceding it in time only in word-time L/E, during which there is no addition or subtraction, or immediately after overflow has been induced in accumulator 6 by subtracting -1 from 0, when it will be zero anyway. Consequently, unless the part of the residue in accumulator 6 is zero, the output of 30S1 will be positive at the latest by $\phi 1$.

27. The condition of the add-subtract staticiser, 30X2, during word-time L/E is immaterial, as the full adder-subtractor and the half adder-subtractor have an 'a' input

only at this time. Its condition during D is important, however, as it is then that the test digit must be added into accumulator 7. If accumulator 6 was empty during the previous word-time, staticiser 30S1 will not be set, and AND gate 30W4 will be closed, preventing the setting of staticiser 30X2 and hence putting the half adder-subtractor in the desired 'add' condition. Otherwise the condition of 30X2 must depend on the NOT-EQUIVALENT operation in elements 30R6 and 30Y2. Now Y12, the output of 30E1 in accumulator 6, is zero at D1, as also is the output of staticiser 30X1, which has ~D on its feedback gate; consequently the NOT-EQUIVALENT operation gives a null result, leaving 30X2 inactive, and putting the half adder-subtractor into the 'add' condition during D as required.

CHAPTER XV

ADDRESSES 6 AND 7 AND JUSTIFICATION

CHAPTER XV

ADDRESSES 6 AND 7 AND JUSTIFICATION

23. $[nq]' = n + 2^{-38}q$

STORAGE

1. Accumulators 6 and 7 in the multiplier-divider can be used, like accumulators 1 to 5, for the storage of numbers, counters or modifiers. They can be gated to the input bus, the N-bus and the X-bus or M-bus in the usual way; but, because of their special application, the operation of the parity-check and parity-insertion circuits is modified when these addresses are specified (see Chapter X).

2. Fig. 15.1 shows the elements of accumulator 6 concerned with normal storage. The accumulator can be cleared, at gate *y* of twin delay 33W1, by the 'erase' waveform in combination with address waveforms N00 and N26. The input word is applied to gate *x* of the same element. The word in the accumulator is fed out to the N-bus through gate *x* of twin delay 33W2, gate *y* of which is used for the output of accumulator 7. Waveform T33 on these gates is necessary to prevent a spurious parity failure during a multiplication or division order, should accumulator 6 or 7 be specified as the N-address. The N-decoding is significant during the whole of the order, and the unusual timing in the accumulator might thus cause a 'one' to appear on the bus at $\phi 40$, i.e. in the 'parity' position, when the parity-injection circuit had prescribed a 'nought'. Accumulator 6 is gated to the X/M-bus through AND gate 33V2 or AND gate 33V3, the latter being used when a modifier is required immediately after it has been routed back to store (see Chapter X). The outputs of these two AND gates are mixed with the outputs of two similar AND gates in accumulator 7, and are applied, via twin delay 34U1, to the X/M-bus.

3. The elements of accumulator 7 concerned with normal storage are shown in fig. 15.2, which also shows elements concerned with justification. Accumulator 7 can be cleared at AND gate 33Y4 by the 'erase' waveform in combination with address waveforms N00 and N27. The input word is gated in through AND gate 33Y3. The gating arrangement for routing out is similar to that for accumulator 6, AND gate 33Y2 being used for modifiers required immediately after replacement. It should be noted that, if accumulator 6 or 7 is specified by the address digits in a multiplication or division order, the operand will be routed out to the relevant bus and erased in the accumulator by waveform X58, on 31L*i*x even if it is to be replaced in its original location.

JUSTIFICATION

4. In multiplication and division orders, accumulators 6 and 7 can generally be regarded from a programmer's standpoint as one double-length accumulator. In accumulative multiplication (function 22) the sum is automatically 'justified', i.e. any 'overflow'

digit produced by addition in the less significant half is automatically carried up into the more significant half. In division operations with a double-length dividend (function 24 or 25) there is automatic justification for the sign of the less significant half provided that this is within range. Double-length addition operations, however, are carried out in two parts in the Mill, where there is no provision for automatic justification. To permit 'carry' between the two halves after addition, the 'justify' function, function 23, has been provided. The less significant half of the number to be justified must have been placed in accumulator 7 by a previous order; but the more significant half can be in any address in the computing store. The X-digits of 'justify' orders are not used. Justification takes two word-times.

5. 'Justify' orders can also be used to form double-length in-range numbers from single-length numbers that have overflowed. The number to be justified is regarded as the less significant half of a double-length number, the more significant half of which is zero before justification, the N-digits of the order specifying any convenient storage location that is known to be empty. If numbers are being regarded as fractions, the final result is then equivalent to a right shift of 38 places.

6. Justification was discussed fully in Chapter I, paragraphs 31 and 32. Owing to the system of treating negative numbers as their complements with respect to 2, there is no necessity for complementation in justification; moreover, the less significant half of a justified double-length number will always be positive regardless of the sign of the more significant half. Thus there are only four possibilities, according as the less significant half has or has not overflowed, and according as it is positive or negative before justification. The operations on the more significant half of the number in the four cases are tabulated below. The third column shows the value of the so-called 'sign' digit of the less-significant half, q; this is the digit immediately to the left of the point, and will have been changed in cases of overflow.

overflow	sign of q	'sign' digit of q	operation on n
no	+	0	none
yes	+	1	add 2^{-38}
no	-	1	subtract 2^{-38}
yes	-	0	subtract 2^{-37}

7. The repeated sign digit necessary in Mill operations is deleted on routing the number back to store (timing waveform 41-37 on the input bus) though it may, of course, be replaced by a parity digit. Overflow will have been recorded on the overflow staticiser in the Mill; the output of this staticiser, together with the remaining 'sign' digit of q, gives all the information needed to determine the justification operation. The 'sign' digit of q is recorded from ϕ40 of word-time E preceding the 'justify' order until word-time E of the 'justify' order itself on staticiser 32T1 (fig. 15.2), the output of which, Y10, is significant from D41 to E41. One word-time after it has been staticised, this digit is deleted by waveform X29 on delay 31V1. Waveform X29 is also used to reset the overflow staticiser in 'justify' orders; it is negative only at ϕ40 of D in a 'justify' order.

Mill Operations

8. The actual justification operation is carried out in the Mill. The adder-subtractor here is set for addition during group-2 orders (see fig. 7.2), so that subtraction must be carried out by the equivalent operation of adding the complement with respect to 2. The Mill routing for justification is shown in fig. 15.3. The more significant half of the number is routed in from the N-bus through the sign repeater (justification can cause overflow) and through twin delays 22K2 and 22E2, appearing at the '*a*' input to the adder from D4 to E1 - the normal timing for addition or subtraction.

9. The other operand is generated in twin delay 32R2 and inverter 32Q1 (fig. 15.3) by comparing the output of the overflow staticiser with the digit staticised on 32T1 in accumulator 7. The operation of this circuit can best be understood by reference to the timing chart. As the output of twin delay 32R2 passes through twin delays 32W1 and 22H2, it is 'trimmed' to the length of a word with a repeated sign digit by waveforms G13 and D delayed one digit time and by waveform ~2. When there is no overflow, the output of twin delay 32R2 is simply waveform Y10 (from the sign-digit staticiser in accumulator 7) delayed one digit-time; so the '*b*' input to the adder is zero if *q* is positive, or -2^{-38} if *q* is negative. When *q* is positive and overflow has occurred, the output of the staticiser in accumulator 7 is again positive; the effect of inverter 32Q1 is to ensure that the output of 32R2 during D is a single pulse at $\phi 2$; so the '*b*' input to the adder is then equivalent to $+2^{-38}$. When overflow has occurred and *q* is negative, the staticiser in accumulator 7 is inactive. The output of 32R2 is then equivalent to the output of the overflow staticiser delayed one digit time, but with a pulse deleted at $\phi 2$ by ~1 on 32R2; in this case, the least significant digit entering the '*b*' adder input will be zero, making the whole '*b*' input equivalent to -2^{-37}.

10. The overflow staticiser in the Mill (fig. 7.2) is cleared automatically when waveform X29, applied via AND gate 21R1 to its feedback gate, becomes zero at D40 of a 'justify' order. It may, of course, become set again at E39 in the usual way if justification has caused overflow at the more significant end of the double-length number.

CHAPTER XVI

SHIFTS

CHAPTER XVI

SHIFTS

50	$x' = 2^N x$
51	$x' = 2^{-N} x$
52	shift x up N places
53	shift x down N places
54	$[pq]' = 2^N [pq]$
55	$[pq]' \quad 2^{-N} [pq]$
56	Normalise $[pq]$

1. Single-length shifts (functions 50 to 53) can be applied to a word in any accumulator. The accumulator address is specified by the X-digits of the order, while the N-digits are to be interpreted as a binary number specifying the number of shifts. Functions 50 and 51 give true arithmetical shifts, i.e. multiplication or division by powers of 2, in which facilities are included for reproducing the sign digit in the correct place and for overflow sensing. Function 51, single-length right shift, gives a rounded result. Functions 52 and 53 give simple positional shifts; there is no rounding, and no attempt is made to delete or reproduce the digit in the 'sign' position or to sense for overflow.

2. Functions 54 and 55 give double-length arithmetical shifts, and are applied to the contents of the multiplier-divider accumulators (6 and 7). With functions 54 and 55, as with single-length shifts, the N-digits in the order specify the number of shifts; the X-digits, however, are unused. The sign digit is reproduced in the correct position, and overflow is sensed; but double-length right shifts, function 55, are unrounded. 'Normalise' orders, function 56, are provided for convenience in floating-point working. They are applicable to a double-length number, the argument of which must have been placed in accumulators 6 and 7. The X-digits of a 'normalise' order specify the accumulator holding the exponent of the number, while the N-digits place an upper limit on the number of shifts permitted (and thus prevent the shift from continuing indefinitely if the argument is zero).

SINGLE-LENGTH SHIFTS

3. Single-length shifts are carried out in the Mill, at a rate of one place each word-time. The word to be shifted is routed into the Mill during word-time D, and circulates round a 43-digit loop for left shifts, or a 41-digit loop for right shifts, for a number of word-times equal to the desired number of shifts. The E-staticiser is then set, and the E-waveform opens the appropriate gates for routing back to store.

As the number of shifts is specified by the N-digits of the order, the shift timing is carried out in the order register, where -1 is added (the order adder having no facilities for subtraction) to the N-digit number each word-time. When the N-digits have all become zero, the addition of -1 in the order adder ceases, and the E-staticiser is set.

Timing

4. The timing control circuit is shown in fig. 16.1, and its operation is illustrated in the timing charts. The X-digits and F-digits of the order are decoded in the normal way at the beginning of word-time D to produce the requisite address waveform, waveform G05, and another waveform designating the particular function in the group. The order then circulates normally in the order register, the N-digits re-entering the order adder during word-time D from $p33$ to $p39$, followed, possibly, by three extended N-digits ($p40$ to $p0$) produced by modification. At the same time, a series of ten pulses representing -1 is fed into the other input of the order adder. 'Carry' is suppressed at $p0$; so the number represented by the N-digits (possibly extended by modification) is reduced by unity in the order adder. The N-digits are sensed at I9 in the order register, where the least significant appears at $p1$ and the most significant (modifier extension) at $p10$ (see fig. 6.3).

5. The digits at I9 are inverted, and the resulting waveform is applied to the feedback gate of staticiser 25Y2 (fig. 16.1). This staticiser is set at $p0$; so its output is positive at $p1$, simultaneous with the appearance of the least significant N-digit at I9. As long as there are N-digits left in the order register, staticiser 25Y2 will be cut off before $p11$. When the last shift is in progress, and the N-digits in the order register have all been reduced to zero, the output of staticiser 25Y2 remains positive until $p11$, enabling staticiser 25Y1 to become active. If 25Y1 is active, its output is positive from $p12$ to $p41$. This output sets the E-staticiser for functions 50 to 55, and is used (in special circumstances) to set the K-staticiser for function 56. The inverted output of 25Y1, is used to inhibit further operations in the order adder; thus the N-digits in the order register remain zero until the end of the beat, when the order is erased. Staticiser 25Y1 is set again during E (and during K for function 56); the inverse of its output is therefore positive at all times except from $p12$ to $p41$ of E and the word-time immediately preceding it, and of the word-time immediately preceding K in 'normalise' orders. This inverted output is also used to control the setting of the overflow staticiser in double-length shift orders.

Mill Operations

6. Fig. 16.2 shows the elements of the Mill that are used in shift orders. For single-length shifts, the operand is fed in from the X-bus through the sign repeater and through twin delays 22K2 and 22E2, appearing at the '*a*' input to the adder-subtractor, with repeated sign, from $p4$(D) to $p1$(D + 1), i.e. with the usual timing. It re-enters the adder-subtractor, again through the '*a*' input, from twin delay 21D1, having travelled round the left-shift loop (~F5) or the right-shift loop (F5) according to the function in the order. The '*b*' input to the adder-subtractor is zero in shifts except only for functions 51 and 56. As can be seen from fig. 7.2, the adder-subtractor will add in function 51 and subtract in function 56. 'Carry' is suppressed at $p1$.

7. In arithmetical left shifts, function 50, the only special requirement is the deletion of the most significant digit during each cycle, so that the word will not extend beyond the normal length for the Mill (i.e. with one repeated sign). This is achieved by waveform ~0 on delay 21C1 in the left-shift loop. Overflow in left shifts, should the shift go too far, is detected in the normal way by a NOT-EQUIVALENT operation between the sign and repeated-sign digits, the result of the operation being made as usual to control the overflow staticiser, which is set at ϕ39. The word is tested for overflow during every word-time of the order.

8. In arithmetical right shifts, the sign digit must be repeated during each cycle. This is accomplished by repeater delay 21C3, feeding the right-shift loop, which produces a pulse at ϕ0 whenever a pulse appears on the loop at ϕ41. Provision is also made for rounding in right shifts. At ϕ3 of D, a pulse emerges from twin delay 22G1, and thus enters the *'b'* input of the adder-subtractor one digit-time before the least significant digit of the word to be shifted enters the *'a'* input. In the next word-time, after one right shift, this pulse will have moved its timing to ϕ2 when a second pulse, entering the *'b'* input through the other side of twin delay 22G1 is added to it. It will be seen that 'carry' from this operation will perform any rounding necessary after one shift. If the least significant digit of the original word was zero, the 'carry' digit will remain in position until another digit is added to it after the next shift. In some cases, the rounding operation will produce 'carry' for some distance up the word, leaving a series of zeros; subsequent rounding operations will then simply substitute spurious 'ones' for zeros. To prevent these spurious digits from interfering with the more significant end of the word, they are deleted one-by-one at ϕ41, ~41 being applied to delay 21P3 in the shift loop. The word emerging from the Mill at the end of the shift operation will often have one spurious digit at its less significant end; but this will be gated out by the timing waveform, 41 to 37, on the input bus.

9. There are two special requirements in logical shifts: first, it must be made impossible for the overflow staticiser to be set; second, any repetition of the digit in the 'sign' position, which in right shifts would be carried on by repeater 21C3, must be avoided. Setting the overflow staticiser is inhibited by X6, the output of inverter 21X1, which is negative only for logical-shift functions, 52 and 53 (G05 & F4 & ~F3). The digit in the 'sign' position will already have been repeated between the X-bus and the Mill. The repeated digit is deleted at the input to delay 21P2, where it appears at ϕ39. It will be seen that the gate preceding 21P2 can be closed only at ϕ39, and only then if a logical-shift function is specified.

DOUBLE-LENGTH SHIFTS

10. Double-length shifts are carried out in accumulators 6 and 7, into which the number must have been placed by a previous order, the more significant half being in accumulator 6. The two halves of the number are shifted independently, one place per word-time, and a digit is shunted from one accumulator to the other each word-time. Fig. 16.3 shows how the control waveforms for double-length shifts are generated. They are produced in the first instance by coincidence between waveform G05 and another G-waveform representing a particular function in the group. The control waveforms are all modified by ~D and ~E, allowing normal circulation in the accumulators during these

two word-times. The special requirements of the 'normalise' function (56) are described in paragraph 15 *et seq.*

Right Shifts

11. Fig. 16.4 shows the elements of accumulator 6 that enter into double-length shift operations. At the end of word-time D, waveform X59 becomes negative, and prevents the normal circulation through gate *y* of twin delay 31S2. In right shifts, the output of delay 32S2 is routed through delay 32S1 and gate *y* of 30S2. The sign digit is repeated in twin delay 30Q2, waveform X61 on gate *y* of this delay being positive at ϕ39(~D & ~E) in double-length right-shift orders. (30Q2 is also used as a sign repeater in multiplication). The word (including the repeated sign digit) emerges from the 35-digit line, 30D, from ϕ38 to ϕ35; the least significant digit is then shunted into accumulator 7. This digit is subsequently deleted in accumulator 6 by waveform ~41 on gate *y* of 30S2. During E, circulation reverts to normal, gate *y* pf 30S2 is closed, and gate *y* of 31S2 is opened. The last digit shunted into accumulator 7 appears at the input to 31S2 in accumulator 6 at ϕ40, before the gate is open, and is therefore deleted in accumulator 6. Thereafter the word circulates normally through gate *x* of 30S2; waveform X58 on this gate is always positive in shift orders.

12. Fig. 16.5 shows the elements of accumulator 7 that are used in shift orders. Normal circulation is inhibited, except during D and E, by waveform X60 on gate *x* of twin delay 31L1, waveform X58 remaining positive for shift functions. The right-shift loop passes through gate *y* of 31L1, which is controlled by the right-shift waveform, X54, and by timing waveform 1-37 to ensure the deletion of the least significant digit before each shift, double-length right shifts being unrounded. The single digit from accumulator 6 enters through gate *x* of twin delay 31L2, where it appears at ϕ38, one digit-time after the most significant digit of *q* enters 31L1.

Left Shifts

13. The left-shift loop in accumulator 7 passes through twin delays 31P1, 31K1 (where the main control waveform, X57, is applied) and 31L2. The most significant digit of *q* is tapped off accumulator 7 at the output of twin delay 31K1, where it appears at ϕ39; this digit is deleted at gate *y* of 31L2 by waveform X62, which is negative from ϕ39 to ϕ41 in left-shift orders (fig. 16.3). The digit shunted from accumulator 7 in left shifts enters accumulator 6 through gate *y* of twin delay 31H1 (see fig. 16.4). The left-shift loop in accumulator 6 incorporates delays 32S1, 31S1, 33W1 and 30S2, the main control waveform, X56, being applied to gate *y* of 31S1. The most significant digit in accumulator 6 appears at this gate at ϕ38, and is therefore deleted by the timing waveform, 41 to 37.

14. Overflow in double-length left shifts is indicated on the overflow staticiser in the Mill. The sensing device is shown in fig. 16.1; it is simply a NOT-EQUIVALENT circuit, operative only for group-5 orders, which compares the most significant two digits in accumulator 6 when they appear at outputs Y17 and Y18, on either side of twin delay 30S2 (fig. 16.4). There is no digit in the 'repeated-sign' position in accumulator 6; consequently the overflow test must be made one word-time earlier than usual, i.e. it must be understood to show not that overflow *has* occurred but that overflow *will*

occur in the next word-time. The test is made at $\phi 39$ of every word-time except E and the word-time preceding it (the last 'shifting' word-time), when it is suppressed by waveform X28 at AND gate 32M4 (see fig. 16.1). The output of this AND gate is mixed into the feedback loop of the overflow staticiser (fig. 7.2). Notice that the overflow signal is significant for accumulator-6 overflow one digit-time later than for Mill overflow.

NORMALISATION

Argument

15. The standard range for the argument in floating-point working is:-

$$-1/2 \leqslant [pq] < -1/4$$

or $$1/4 \leqslant [pq] < 1/2$$

This means that the sign digit and the digit immediately to the right of (less significant than) it are equivalent, and that the second digit after the point is complementary to these two. Overflow cannot then occur as a result of a single addition or subtraction; moreover, if there has been no overflow, no more than one right shift will be needed to bring the argument back to the standard range. The system followed in normalising in Pegasus is to shift the argument to the left until it is just outside the standard range, i.e. until it is in the range

$$-1 \leqslant [pq] < -1/2$$

$$1/2 \leqslant [pq] < 1$$

and then to perform one right shift to normalise. If the argument is already just outside the standard range, only the right shift is performed. If the range of the argument is already correct, the normalise order performs one left shift followed by one right shift.

16. The most significant two digits in accumulator 6 are tested for non-equivalence at the end of every word-time, including D, as they are in double-length left shifts. As soon as the NOT-EQUIVALENT operation gives a positive result, showing that the left shift has gone one step too far, the K-staticiser is set, and the machine performs one right shift. The E-staticiser is set, as usual, at the end of K. Circulation in accumulators 6 and 7 is normal during E. The shifts for function 56 are gated by the same control waveforms as for functions 54 and 55. These waveforms are produced, as shown in fig. 16.3, by coincidence between waveforms G05, ~E, G16, and either K or ~K, according to the sense required of the shift. The most significant two digits in accumulator 6 are tested at Y17 and Y18 in the same way as for double-length left shifts; but the overflow staticiser is not set; instead the 'mix' output of twin delay 32K1 (fig. 16.1), gated with G16, is applied to the K-staticiser as shown in fig. 13.2. As the K-staticiser is set at $\phi 40$, Y17 and Y18 are actually sensed at $\phi 39$.

17. The N-digits in a 'normalise' order are interpreted by the machine as a number setting an upper limit on the number of left shifts. Some upper limit must be set, otherwise the normalise order would continue indefinitely if the machine encountered a zero operand; moreover, the limit enables certain operations to be speeded up by neglecting small numbers. The N-digit number is reduced by unity in the order register each word-time, in the same way as for the other functions of group 5. If non-equivalence between the first two digits in accumulator 6 has not been reached by the time that all the N-digits are reduced to zero, staticiser 25Y1 is set (fig. 16.2) and, from it, the K-staticiser. Note that the normal setting of E is inhibited for function 56 at AND gate 22Y2. The inverted output of 25Y1 prevents further operations in the order adder; so the N-digits of the order remain zero in the order register until the end of the beat. The normalise order always gives one right shift during K; consequently, the net result, if N is reduced to zero, is N-1 left shifts. The argument is then in the range

$$-1/4 \leqslant [pq]' < 1/4$$

~E is used on AND gate 31U3 to prevent K from being set again after E in limit-terminated normalise orders, G16 being positive at $p40$, and the output of 25Y1 being positive also if the N-digits have been reduced to zero.

Exponent

19. The X-digits of a 'normalise' order specify the address of the exponent of the operand. The number in the X-address must be regarded as an integer (or alternatively as a fraction equal to 2^{-38} times the exponent). It must be reduced by the net number of left shifts carried out by the order. This operation is carried out in the Mill. What happens, in fact, is that the exponent is first increased by 2 in word-time D; then in each succeeding word-time, including K but not E, it is reduced by unity. As the argument is actually right-shifted in K, the value of the exponent is consequently reduced by an amount equal to the net number of left shifts. Fig. 16.2 shows the Mill circuit. The exponent enters from the X-bus sign repeater via twin delays 22K2 and 22E2, its least significant digit appearing at the *'a'* input to the adder-subtractor at $p4$ of D, and its repeated sign at $p1$ of (D + 1) (or of K in the case of one right shift). The exponent circulates through the normal circulation loop and gate *y* of 21D2 until word-time E, when it is copied back into store. During E it is erased from the Mill by the gating waveform on delay 21P3 in the circulation loop.

20. In normalise orders, function 56, the Mill adder-subtractor is set to subtract (see fig. 7.2). The operation during D is therefore the subtraction of -2, i.e. of a series of 39 positive digits (including a repeated sign digit). These are produced in staticiser 22H1, and appear at the adder-subtractor *'b'* input from $p5$ of D to $p1$ of (D + 1). During each subsequent word-time a single pulse, equivalent to +1 enters the *'b'* input from delay 22H2 at $p4$. (A pulse actually enters at $p4$ of E, but it is ineffective as there is no *'a'* input to the adder-subtractor at this time; it simply follows the operand through the Mill and is deleted eventually by the gating waveform on the input bus.)

21. As the exponent is unlikely to require more than about nine binary digits, it can be placed, quite conveniently, in the 'counter' position of an accumulator already

holding a modifier. Operations on the exponent will not spoil the modifier provided that:

(i) there are enough gaps between the modifier and exponent to allow the exponent to increase numerically without overflowing into the modifier;

(ii) there is a buffer digit between the modifier and exponent, so that the exponent can change its sign without projecting a series of 'carry' digits into the modifier.

22. Overflow of the word containing the exponent is sensed in the usual way and recorded on the overflow staticiser in the Mill. The staticiser setting is inhibited until E by applying X411, the output of inverter 21X2, to the AND gate preceding the overflow staticiser (fig. 7.2). This is done so that a possible temporary overflow caused by the addition of 2 to the exponent in D, which may be cancelled by a subsequent subtraction, shall not be recorded. Notice that operations on the nine-digit exponent cannot of themselves cause overflow, which can only result from a programme error when an exponent is allowed to overflow into a modifier in the same accumulator. Even if the word overflows before E, the overflow will still be apparent during E, when the test is made.

C H A P T E R X V I I

MONITORING FACILITIES

CHAPTER XVII

MONITORING FACILITIES

1. The programmer or maintenance engineer can acquire information on the internal condition of the machine from four sources: the tape punch, the hooter, the neon indicator lamps and the oscilloscope displays. The machine will automatically punch out block-transfer addresses if so desired (see Chapter XI); otherwise the punch is used under the control of warning characters, which are treated in Volume 4.

HOOTER

2. The hooter is a loudspeaker that is mounted under the left-hand wing of the control desk. It is fed as shown in fig. 17.1 with the 'hoot' waveform from the address track or a 'noise' waveform derived from a flying probe in bay 2. This probe can be plugged into any package test socket in bay 2 so that the hooter can be made to give out continuously a sound corresponding to one of the main control waveforms, the volume being controlled by a potentiometer in bay 2. The 'hoot' waveform is a series of eight digit pulses appearing in the address waveform (T23) from ϕ32 to ϕ39 in even-numbered blocks only. This is gated in from ϕ32 to ϕ41 to eliminate the other information on the address track, and is also gated with V15, which is the 'mix' of all the internal stop waveforms except the temporary 'busy' stop. The operation of the hooter can be inhibited with switch KS35 on the programmer's panel or by the 'stop' switch when at STOP or SINGLE-SHOT. The output of AND gate 13W4 is rectified and filtered to select a 'hoot' signal at half 'block' frequency - approximately 500 c/s. This is fed to output-one element 13X3 while the 'noise' signal is fed to output-one element 13X4; the mixed outputs of these two are then further amplified by two output-one elements in parallel to provide power for the hooter.

INDICATOR LAMPS

3. The neon indicator lamps concerned with the logical operation of the machine are mounted either on the programmer's panel or above the monitor. They are all fed from output-one elements in the machine. Six of the lamps on the programmer's panel can operate only when the machine has stopped, and serve to indicate the type of stop that has occurred. They are listed below, together with references to the figures showing the relevant logical units.

optional stop (stop-go digit)	fig. 9.1
stop order (function 77)	fig. 9.1
unallocated functions	fig. 9.1
computing-store parity failure	fig. 10.6
drum parity failure	fig. 10.6
write with overflow	fig. 11.4

Two more 'stop' indicators are concerned with the 'busy' stop: one is lighted when the tape reader is busy; the other when the tape punch is busy. These two indicators will be flashing continuously during input and output routines; but will give a steady indication of a stop if one of the devices becomes faulty. The logical elements concerned with the input and output tape drives are shown in figs. 12.2 and 12.3.

4. There are three more neon indicators on the programmer's panel. One of these indicates overflow (see fig. 7.2). It will be flashing continuously during a normal run; but it will give a steady indication during a stop due to an attempt to write on to the drum after overflow, or after a single-shot or manual operation when overflow has occurred. The other two neon lamps indicate, after an internal stop or a single-shot operation, whether the next order to be obeyed is the A-order or B-order of a pair. The machine will, in fact, stop either in a C-state (before an A-order), or in an A-state (before a B-order); consequently the two neon lamps are fed with the C-waveform or the A-waveform from the beat generator (see fig. 5.5).

5. Seven lamps above the monitor panel indicate which of seven possible input or output devices were selected by the last external-conditioning order. These lamps are fed from the output elements that drive the external-conditioning relays (see fig. 12.4). Six more lamps on the monitor panel indicate the track selected in drum transfer orders. These are fed, through output-one elements, with the S-outputs from the first stage of the partial decoding (see fig. 11.1). The track is therefore indicated as a binary number, the lighted lamps representing 'ones' and the unlighted lamps 'noughts'. The first of the six lamps will be lighted only during transfers from the isolated part of the store.

MONITOR

Programmer's Display

6. Two oscilloscope tubes are provided. The right-hand tube, the programmer's display, gives a representation either of the contents of the order register or of the number in the order-number register, the choice being made by means of a switch on the programmer's panel. The order pair before the 'shuffle' is shown in two parts, the A-order appearing above the B-order. The oscillogram, whether of order number or order pair shows the 'ones' as vertical lines, and the 'noughts' as dots. By operating a 'scale' switch mounted above the oscilloscope, the programmer can cause the displayed digits to be spaced out into groups, which correspond from left to right to the E-digits, (modifier extension of the order in control), the N-digits, the X-digits, the F-digits and the M-digits. The digits are displayed in descending order of significance, i.e. the timebase runs from right to left.

7. A stationary display is possible only if the series of digits appearing at the monitor point during the scan period is repeated at regular intervals. This will be so in general

(i) if the machine is stopped,
(ii) if the machine is obeying a manual order repetitively,
(iii) if the machine is obeying repetitively a closed loop of instructions.

8. The timebase is triggered from the logical circuits in a way that can be selected by the operator with the aid of a 'trigger' switch on the monitor panel and a set of seven 'trigger' handkeys on the programmer's panel. With the 'trigger' switch in the position marked REPETITIVE, the timebase is triggered every two word-times. This is the fastest possible, owing to the necessity for a flyback interval between the appearance of the two halves of the order pair on the tube screen; the A-order, which occurs after the B-order in time, being displayed first on the tube. The next five positions of the 'trigger' switch enable the condition of the machine to be studied in any word-time of a simple five-word bar in which the order number coincides with a number, n, set up on the last six of the triggering handkeys.

9. The position on the 'trigger' switch marked MULTI-BEAT enables the monitor timebase to be triggered in any word-time of a bar. A special counting circuit starts to operate after word-time CD of the bar; the timebase is triggered in the next word-time after coincidence has been found between the output of the counting circuit and a binary number set up on the triggering handkeys. Thus the content of the order register can be studied at word-time D + 6, say, of a shift order run manually by setting the triggering handkeys to 'six', i.e. to

0 0 0 0 1 1 0

and by turning the 'trigger' switch to MULTI-BEAT. The position on the 'trigger' switch marked DRUM ADDRESS enables the timebase to be triggered when the seven-digit address on the drum address-track coincides with the handkey setting. To enable the timebase to be triggered twice, four times, or eight times per drum revolution, the first three handkeys have an 'up' position, in which the corresponding digit is inhibited from the standpoint of coincidence-seeking. The position marked EXTERNAL on the trigger switch enables the monitor to be triggered from any suitable waveform generated within the machine a triggering probe being provided, which can be plugged into a test socket on one of the packages.

Engineer's Display

10. The left-hand oscilloscope is referred to as the engineer's display, although several of the facilities associated with it may be made use of by the programmer. Those switch positions on the monitor panel that are likely to be of the greatest use to programmers are engraved in white; the rest of the engraving is in black. The timebase of the engineer's tube is triggered in synchronism with the timebase of the programmer's tube, according to the setting of the 'trigger' switch and the handkeys; but a coarse and a fine timebase control give continuous variability from 100μ sec. to 30m. sec. for the timebase of the engineer's display. With the coarse control set at HALF WORD, a twin trace can be obtained similar to that on the programmer's tube.

11. The information displayed on the engineer's tube is determined by the setting of the 'Y-plate' switch, which is mounted to the left of the tube face, and by two 'computing store' switches, whose control knobs are mounted below it. The switch connections are shown in fig. 17.2. The first position of the 'Y-plate' switch, marked EARTH, gives a zero level for calibration purposes. With the switch in one of the next three positions, the tube will display digital information, mainly from the computing

store, as selected by the two 'computing store' switches. The three positions of the 'Y-plate' switch referred to are marked WAVEFORM, CLOCKED DIGITS, and NORMAL. With the switch at WAVEFORM, the raw digit-pulse waveform is displayed. With the switch at CLOCKED DIGITS, the digit waveform is 'clocked up' before it is displayed, with, generally, the elimination of spurious spikes. The NORMAL position of the switch enables the engineer's tube to display a pattern similar to the one on the programmer's tube, a series of vertical lines and dots representing, respectively, binary 'ones' and 'noughts'. These can be spaced on the display according to the digit groups of orders by operating a 'scale' switch mounted above the tube.

12. The 'Y-plate' switch has a position marked METER SWITCH, which enables the information displayed to be selected with the meter switch on the engineer's panel. The other six positions of the 'Y-plate' switch are associated with flying probes that can be plugged into the test sockets of the packages. The switch pins are connected to probe sockets, which are mounted, two to each bay, in the trimming in the front portion of the cabinet. The left-hand sockets, the A-sockets, are associated with power sockets intended for use with special probes carrying cathode-follower units.

13. The output of any ordinary register in the computing store can be displayed on the engineer's tube by setting the upper 'computing store' switch, switch B, to give the required block, and the lower 'computing store' switch, switch A, to give the position of the register in the block. The accumulators are selected by the lower switch alone, provided that the upper switch is set to one of the 'block' positions 0 to 5. The lower switch can also be used to select, in the same way as the accumulators, information from any one of the following sources:

(i) the drum track last specified in a drum-transfer order,
(ii) the output of the 'manual' handkey on the programmer's panel,
(iii) the output of the tape reader,
(iv) the Mill,
(v) the order-number register,
(vi) the order register.

14. An alternative set of annotations, engraved in ~~red~~ black on the lower 'computing store' switch, becomes relevant when the upper switch is turned to TEST 1. By this means, any one of the following waveforms can be displayed on the engineer's tube:

(i) the 'clock' waveform from the clock amplifier or crystal oscillator (switch position CLOCK 0),
(ii) the two 'clock' outputs (switch positions CLOCK 1 and 2),
(iii) the seven 'reset' waveforms,
(iv) the content of the drum address track or any part thereof,
(v) pulses $\phi 12$ and $\phi 17$ from the extreme ends of the commutator delay line,
(vi) the waveform on the N-bus,
(vii) the waveform on the M-bus,
(viii) the waveform on the X-bus.

15. The position marked TEST 2 on the upper 'computing store' switch, enables the lower switch to make a selection between the waveforms set out below:

Switch Position	Waveform
X2	A
X3	B
X4	C
X5	D
X6	E
X7	N-parity (output of 'stop' staticiser)
N0	X-parity (output of 'stop' staticiser)
N1	Accumulator 6 (nickel-line output)
N2	Accumulator 7 (nickel-line output)
N3	Multiplicand-divisor register
N4	Multiplier timing register
N5	K
N6	L
N7	Input bus

The 'computing store' switches are mounted in bay 3, and the information selected by them is routed through a cathode-follower unit mounted above them. Plug-type links on the cathode-follower unit and in the oscilloscope Y-plate lead enable the selected information to be displayed alternatively on an external oscilloscope.

MONITOR LOGIC

Timebase Control

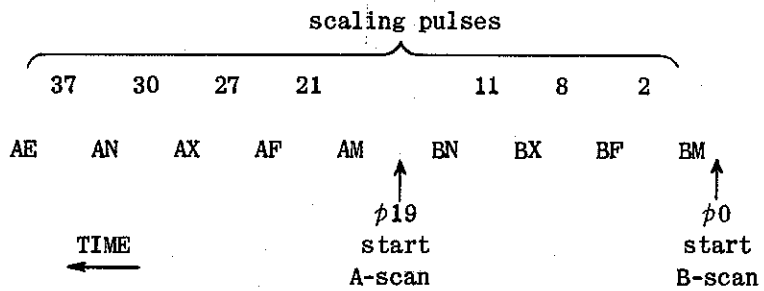
16. The monitor is controlled by an auxiliary beat generator giving two outputs and their inverses, which, suitably combined and gated, are used to produce triggering waveforms to start the scan, to stop the scan on the programmer's tube (and on the engineer's tube when the timebase switch is set to HALF WORD) and to shift the scan up and down for twin-trace displays. This beat generator can be triggered, according to the setting of the control switches, from a 'coincidence' circuit that compares the number set up on handkeys with the output of the order number register, with the output of the drum-address circuits or with the output of the special multi-beat counting circuit. Only the logical circuits will be described here. For a description of the electronic circuits, the reader is referred to Volume 3 of the handbook.

17. Fig. 17.3 shows the circuit generating the monitor control waveforms. It incorporates a beat generator containing two staticisers, 13D1 and 13D2; this is triggered by waveform V25, which is positive at $\phi 41$ following coincidence between the handkey setting and the order number, the drum address or the output of the multi-beat counter. Staticiser 13D1 is set first, staticiser 13D2 being set at the beginning ($\phi 41$) of the next word-time. Staticiser 13D1 is cut off again at the next appearance of V25, and 13D2 is cut off one word-time later. When the trigger switch is set to REPETITIVE,

V25 occurs every word-time; the connections to inverters 13E2 and 13E3, however, ensure that 13D1 cannot be cut off unless 13D2 is active, and cannot be set unless 13D2 is inactive; consequently the two staticisers must be active for at least two word-times each. The staticiser outputs, X110 and X111, and their inverses, X105 and X106, are such, therefore, that X110 will 'overlap' X106, and X111 will 'overlap' X105, for just one word-time in each monitor cycle, these word-times running from $p0$ to $p41$.

18. Fig. 17.3 shows how the beat waveforms are combined with p-pulses or with V25 to produce trigger pulses for the monitor circuits. The operation can best be understood by reference to the timing chart. The scan on the right-hand tube is started at $p19$ for the A-order or $p0$ for the B-order, and is stopped at $p0$ for the A-order or $p19$ for the B-order. The trace is shifted down at the end of the A-scan and up again shortly before the start of the next A-scan. The scan on the left-hand tube is controlled in the same way when the coarse timebase control is at HALF WORD (Z178 positive). With the coarse control in any other of the operative positions, the trace is kept permanently in the 'down' condition, and the timebase is triggered one digit-time after the appearance of V25, or at $p13$ or $p27$ when the trigger switch is in one of the special drum-trigger positions. Notice that, except for half-word timebases, there is one sweep per word-time for repetitive triggering.

19. Fig. 17.4 also shows how triggering pulses are generated for speeding the trace to give 'scaled' displays in which the digit groups in orders are separated for easy recognition. The diagram below shows the scaling pulses in relation to an order pair in standard timing in the order register during beats C and A. The B-order, of course, does not appear modified at the monitor point until BD, when it has been shuffled into the timing of the A-order.



Scaling is brought into operation with a switch mounted above the corresponding tube.

Order-Number Triggering

20. When the 'trigger' switch is in one of the five positions marked CD, AD, AE, BD, BE, the monitor beat trigger, V25, is produced at $p41$ (the beginning) of the specified word-time whenever the order number coincides with the handkey setting. The circuit arrangement is shown in fig. 17.4. The order number is tapped off the order register at I0, where it appears from $p34$ to $p39$ of BE, (or of AE in a satisfied 'jump' order). Coincidence is indicated by the output of staticiser 13K1; this is fed with a pulse from twin delay 12D2 at $p35$ of every BE (or AE if a jump is specified), which will continue to circulate through either gate x or gate y according as the corresponding

order-number and handkey digits are both 'ones' or both 'noughts'. After coincidence has been found, the output of 13K1 will remain positive until the next BE (or AE & J), circulation being through gate *y* as the outputs of 13M2 and 13M1 will both be zero during this period.

21. The particular word in the bar is selected by gating the coincidence signal with the output of delay 12E2, 13F1 or 13F2, which will be positive at $p41$ (the beginning) of the word-time selected by the switch. This gating waveform is generated by coincidence between X80 (the D-staticiser trigger) or V3 (the E-staticiser trigger) and suitable rhythmic waveforms as follows:

CD	is defined by	X80 & (B_vJ)
AD	"	X80 & C
AE	"	V3 & A & ~J & ~E
BD	"	X80 & A & ~J
BE	"	V3 & (B_vJ) & ~E

Notice that ~E must be used in conjunction with V3, which is positive both before, and during, E in several functions. The use of J in defining BE and ~J in defining AE (see fig. 17.4) is a fortuitous result of equipment economy.

Drum-Address and Multi-Beat Triggering

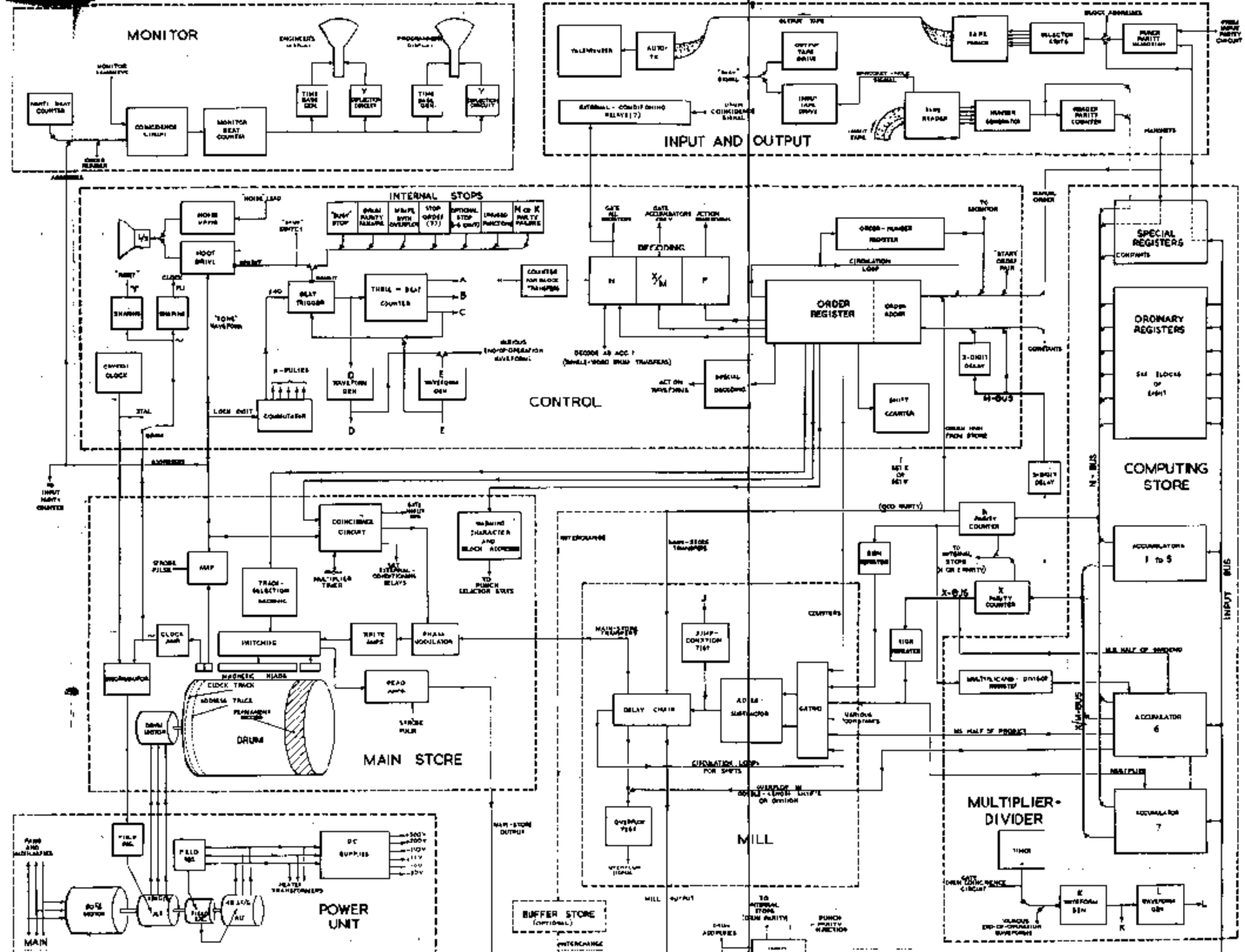
22. Because the timing of the order number at I0 and that of the drum address at T23 are different, two sets of number generators must be used for the handkeys. The arrangement for drum-address triggering is shown in fig. 17.5. The address digits, which appear at T23 from $p6$ to $p12$ are gated into the coincidence circuit through gate *x* of 13M1, where some or all of the first three digits will be deleted if the corresponding handkeys are in the 'up' position. Coincidence is sought as for order-number triggering; a pulse is injected into staticiser 13K1 at $p41$ and circulates through gate *y* until $p7$ when the first digit is tested for coincidence. When coincidence has been found for the tested portion of the address, the output of 13K1 remains positive until $p7$ of the next word-time, when coincidence must fail. The coincidence signal is gated with $p41$ to produce V25.

23. The multi-beat counter, consisting of elements 11X, 11T1 and 11Y1, is essentially a half adder arranged so that the content of nickel line 11X is increased by unity each word-time. The cycle starts in C, when the previous content of 11X is erased at gate *y* and replaced by a single digit injected through gate *x* at $p6$. The output of AND gate 11R3 is zero during C; consequently a handkey setting of zero will give coincidence during C followed by an oscilloscope scan in AD. The output of the counter is such that a handkey setting of *n* gives coincidence during C + *n*, and consequently an oscilloscope scan during AD + *n*. When *n* is equal to the number of word-times in a bar, the scan will occur during CD of the next bar.

Special Drum Trigger

24. There are two positions on the trigger switch that enable the engineer's display to be triggered at $p13$ or $p27$ after drum-address coincidence has been found. The purpose of this is to enable waveforms from the drum to be examined on as large a scale as possible with the timebase coarse control at 100μ sec. and the fine control suitably adjusted. The triggers at $p13$ and $p27$, together with the normal trigger at $p0$, enable a word to be broken up into 'thirds' on the display.

25. Staticiser 10Y1 (fig. 17.5) is set at $p40$ after coincidence has been found, and remains active for one word-time. Its output enables the 'start scan' pulse to be produced at $p13$ or $p27$ according to the setting of the 'trigger' switch (see fig. 17.3). It will be seen that the special triggers cannot be produced while the coarse timebase control is at HALF WORD, the two scans then being started at $p0$ and $p19$ for all operative positions of the trigger switch except EXTERNAL. The scans on the programmer's display are triggered in the normal way at $p0$ when the special triggers are controlling the engineer's display.


MONITOR
INPUT AND OUTPUT
INTERNAL STOPS
CONTROL
SPECIAL REGISTERS
ORDINARY REGISTERS
COMPUTING STORE
MAIN STORE
DRUM
MILL
MULTIPLIER-DIVIDER
POWER UNIT
BUFFER STORE
ORDER REGISTER
DECODING
INPUT BUS
MAIN