Alan Turing's Manual for the Ferranti Mk. I

Alan M. Turing
Transcribed by Robert S. Thau

February 13, 2000

Transcriber's preface

This is a transcription of a very interesting document from the early history of computer science — Alan Turing's manual for an early computer built by the English firm Ferranti as a commercialized version of the prototype computer which had been developed by the Manchester University staff. Turing had moved to Manchester from the National Physical Laboratory (where he had been, for a time, architect of the ACE computer project and responsible for the rather unusual architecture of the ACE). At the time he wrote this manual, he was in charge of what we would now call system programming for the machine, including the development of common tools and the establishment of conventions for library routines.

This was the first of at least three manuals for the machine, and was apparently written before the machine was fully installed and operating (the library input routines, for instance, are described in the future tense, and the description of auxiliary paper-tape-handling hardware is frankly speculative). It was quickly superseded by a second manual for the machine, incorporating some of the material from this one (the introductory material was more or less intact), and adding a description of a second set of conventions for shuffling routines from the disk into electronic storage ("Scheme B"; Turing's original routines were described in the second manual as "Scheme A"), and a couple of interpretive routines. The table of contents of the second manual, and the complete first chapter, are on line at `http://www.computer50.org/mark1/`; the direct link is `http://www.computer50.org/kgill/mark1/mark1book.html`.

A note on the machine Turing was describing

The nomenclature of the early Manchester machines is somewhat confused, and so deserves brief elaboration here. The first computer built at Manchester was the "baby machine" of Williams and Kilburn. The original "baby machine" had a tiny amount of memory, and just barely enough circuitry to run small programs for, e.g., greatest common divisor. In this initial configuration, it was more a test harness for computer components, most notably its cathode-ray-tube memory, than a useful computer in its own right. However, over time, it was elaborated to the point that it could be, and was, used for useful work, via the addition of more memory tubes, more instructions, a multiplier, a magnetic drum, and the first known use of an index register (though a peculiar one to modern ways of thinking, as its contents could modify opcode as well as address bits). The final state of the prototype hardware is documented in an appendix of Turing's manual.

Once the prototype was operating, the English firm Ferranti was hired to produce a productized, commercialized version of the machine. This commercialized version was similar in overall design to the final state of the prototype hardware, but different in detail; it had a different instruction set, more index registers (B-lines), and added a few other facilities (hardware random number generator, arithmetic on the contents of B-lines, etc.). Ferranti referred to this machine as the "Mark I", since it was the first computer which they had produced. However, Turing refers to it here as the "Mark II" (or the "Ferranti machine"), since it was the second computer which he had worked with at Manchester.

Much more information on the two Mark Is in their various incarnations may be found on line at Manchester University's web site on their own history, which also has numerous photographs of the original machinery and a recent reconstruction of the first "baby machine" (the URL, once again, is `http://www.computer50.org/mark1/`). The reader is also referred to Martin Campbell-Kelly's description of the programming environments at Manchester throughout the life of the Mark I, in *Annals of the History of Computing*, vol. 2, pp. 130-168; this contains much interesting information on "Scheme B" and R. A. Brooker's historically significant Autocode system.

A few words on the historical context

As is obvious, this manual describes programming in absolutely raw machine code, making no concessions to the human frailty of the user. Attitudes on this point varied from group to group in early computing; this sort of "machine centrism" was not universal. In particular, David Wheeler in the EDSAC group at Cambridge had managed to cram what we'd now call a crude, but useful assembler into the forty instructions' worth of toggle switches comprising what we'd now call the EDSAC's bootstrap ROM. Turing went straight the other way — even when taking input from paper tape, his input routines were designed to take that input in a format which looked as much as possible like the physical image of data in a storage tube. (See p. 31).

Indeed, Turing's absolutism on this point (e.g., requiring users to memorize the obscure 32-symbol teleprinter code used pervasively in his manual, and persistently using little-endian numeric notations even in the text of his manual) was something of a puzzle to many of his colleagues; the memoir of EDSAC project leader Maurice Wilkes includes an anecdote of Turing doing little-endian arithmetic on a chalkboard during a conference talk, to the general befuddlement of the other researchers in attendance.

However, there was a method to this seeming madness. As one might infer from Turing's detailed description of the machine's console, operating procedures, and so forth, and his recommendation of their use as the most effective way of debugging a routine (indeed, going so far as to suggest selectively write-protecting the drum against errant writes by physically removing valves, i.e. vacuum tubes), Turing found direct hands-on contact with the machine to be of great value, and thought that programmers should seek it if possible. Obviously, this requires a detailed understanding of the internal hardware representations of just about everything. This is in marked contrast to the attitude present in the EDSAC group, and most other places, at the time, in which physical contact with the machine was generally restricted to a small number of trained staff. This attitude wasn't always appreciated by the programmers themselves, as Steven Levy's book *Hackers* documents in its discussion of computers and programmers at MIT at few years later.

A particular curiosity is the description of the routine ACTION and its companions in connection with the "formal mode of operation" on pp. 48-50. This may well be the first written description of anything resembling an operating system, though its purpose is described mainly as providing an audit trail for what was done in the run of a problem on a machine. In this connection, it is also interesting to note the "false cue" facility, which allowed a half-track of the magnetic drum to serve as a directory for routines elsewhere on the drum. In fact, Turing does refer to tracks devoted to this purpose as "directories", though the entries in those directories are keyed by number, and not by the English names of the routines (as they might be in the directory of a modern library or file system); in that sense they are less akin to the directories of a modern file system than, say, the tables of offsets in some modern shared library formats.

Another point of interest is the several references to formal proofs of a program's validity. Turing did publish a complete example of a program proof in the proceedings of a 1949 conference on automatic computing machines at Cambridge — the very paper whose presentation Wilkes described above. Unfortunately, the proof seems to have had little influence on further development, in part, perhaps, because of the confusing nature of the presentation. In his discussion of the paper, Wilkes states

> It would enhance Turing's reputation if I could state that he did in some way anticipate the contribution that [theoretician R. W.] Floyd made many years later. This I cannot do, although Turing did use the word "assertion" and he did point out the separate need to show that the execution of the program would terminate. What was missing was the concept of the loop invariant; if he had had the concept, he would have had no difficulty in giving the value of the invariant ...

This assertion deserves perhaps a little comment. While Turing's paper does not cite the invariants of the two loops in his routine as being of particular importance, they are present, as the preconditions of the loop heads, amid a large table of similar assertions about other portions of the routine. (If not, it could hardly be a valid proof!)

While admittedly viewing the situation from a considerably farther remove than Dr. Wilkes, I think it might be more accurate to say that Turing was missing the notion of structured programming — that there is a set of primitive control constructs (if-then-else, while loops, and subroutine invocations) which can be used to synthesize any program. Given this idea, it is straightforward to come up with Floyd-style inference rules for each of the control constructs, which make it more straightforward to construct proofs whose structure mirrors the structure of the program (as opposed to the ad hoc stew of assertions in Turing's paper). This is consistent with Turing's repeated lament (in section 19 and in the conference paper) of the lack of an adequate *notation* for program proofs, which follows, in my view, from the lack of a suitable notation for programs. On the other hand, it would regrettably be entirely in character for Turing to be aware of the general usefulness of loop invariants to the construction of proofs, but to regard it as an incidental detail not worth taking time to point out in his brief presentation.

Of course, this is all further obscured by the purely notational confusion experienced by Wilkes, and doubly so by subsequent readers of the conference proceedings, who had to contend with a paper on verification of a factorial routine from which all factorial signs had been omitted. Corrected versions may be found in the volume 14 of the Babbage Institute Reprint Series for the History of Computing, which reprints the conference proceedings, or in the reprint of the paper itself, with commentary, in *Annals of the History of Computing*, vol. 6, pp. 139-143.

One other point deserves mention. The magnetic drum hardware of the Ferranti machine exchanged data with the machine's main memory by transferring a tube full of data at a time, in units which Turing refers to as "pages". Turing also describes purely software conventions for allowing routines on one page to call routines stored on another, which does not reside in main memory at the time. Succeeding hardware and software systems at Manchester had similar conventions, for data as well as code, which were increasingly automated (particularly in Brooker's Autocode system), culminating in the Manchester Atlas machine which, as is well known, was the first to support demand paging in the modern sense.

While Turing was not responsible for the design of the magnetic storage hardware he described, he may well have chosen the terms in which he described it; the term "page" is used in much the same way, as part of the same metaphor of "books" and "pages", in a lecture which Turing gave at the London Mathematical Society in 1947, well before he took the job at Manchester, in which he noted, after a somewhat fanciful digression concerning memories based on physical, paper books, that

> If we are to have a really fast machine, then, we must have our information, or at any rate a part of it, in a more accessible form than can be obtained with books. It seems that this can only be done at the expense of compactness and economy, e.g., by cutting the pages out of the books and putting each one into a separate reading mechanism. Some of the methods of storage which are being developed at the present time are not

unlike this.

(This lecture has been reprinted in volume 10 of the Babbage Institute's reprint series, which also contains Turing's original report on his ACE design). It is therefore possible that Turing, widely considered to be of most significance to computer science as a theoretician, has made at least one lasting contribution to the day-to-day vocabulary of practical systems programming.

Notes on the transcription

For the most part, I have tried to let the document speak for itself; however, I have added some notes on points where Turing's text was less than completely clear. In particular, I have added lists of relevant instructions to Turing's description of various features of the machine; Turing's original intent seems to have been for the reader to refer to the single long list in the appendices. I have also added a few brief notes on how examples in the manual reflect Turing's other interests and activities, and corrected some obvious typographical errors in the listings, and so forth; these are noted as they occur. All my additions to the text are placed in [square brackets]; the rest is Turing's original text.

I'd like to thank Brian Napper of the University of Manchester for his assistance in proofreading the document; he found a number of typos, mostly introduced by myself, which had escaped my attention. The fault for any remaining errors, of course, is my own.

Lastly, a few notes on this draft. I've tried to note and correct typos where I've found them, but I have not yet checked some of the more complicated math; as noted in the body of the text, the Tchebysheff polynomials and the formulae in the description of the Riemann hypothesis are particularly suspect.

Contents

[Author's] Preface

This handbook is written mainly for the benefit of those who will actually do programming for the Mark II machine. It is not intended to be an introduction to computing machinery, nor is it intended to explain the engineering techniques used in the machine. Although it is believed that the handbook is complete in itself, and eventually could be understood by anyone with a mathematical training, it is recommended that the reading be supplemented by some closer contact with the machine itself.

At the time of writing, comparatively little programming has been done with the Mark II machine. It is hoped to issue supplements consisting of particular routines from time to time. The conventions and suggestions about programming that are given are largely based on experience gained from the Mark I machine.

1 General Remarks on Electronic Computers

Electronic computers are intended to carry out any definite rule of thumb process which could have been done by a human operator working in a disciplined but unintelligent manner. The electronic computer should however obtain its results very much more quickly. The human computer with whom we are comparing it may be imagined as supplied with various computing aids. He should have a desk machine, paper to write his results on, and more paper on which is written a detailed account of how the calculation is to be carried out. These aids have their analogues in the electronic computer. The desk machine is transformed into the computing circuits, and the paper becomes the "information store" or more briefly the "store", whether it is paper used for results or paper carrying instructions. There is also a part of the machine called the control which corresponds to the computer himself. If his possible behaviour were very accurately represented this would have to be a formidable complicated circuit. However we really only require him to be able to obey the written instructions and these can be made so explicit that the control can be quite simple. There remain two more components of the electronic computer. These are the input and output mechanisms, by which information is to be transformed from outside into the store or conversely. If the analogy of the human computer is to be maintained these parts would correspond to his ears and voice, by means of which he communicates with his employer.

2 Scales of notation

The information stored on paper by the human computer will mostly consist of sequences of digits drawn from 0, 1, ..., 9. There may also be other symbols such as decimal points, spaces etc. and there may be occasional remarks in English, Greek letters etc. There may in fact be anything from 10 to 100 different symbols used, and there is no particular need to decide in advance how many different symbols will be concerned. With an electronic computer however such a decision has to be made; the number of symbols chosen is ruled very largely by engineering considerations, and with the vast majority of machines the number is two. Machines (e.g. ENIAC) have however been made with 10 different symbols. The number for the Ferranti machine is two, and the symbols used are 0 and 1.

It is not difficult to see that information expressed with one set of symbols can be translated into information expressed with another set by some suitable conventions, e.g. to convert a sequence of decimal digits into sequences of 0's and 1's we could replace 0 by 0000, 1 by 1000, 2 by 0100, 3 by 1100, 4 by 0010, 5 by 1010, 6 by 0110, 7 by 1110, 8 by 0001, and 9 by 1001. Alternatively one could assume that the sequence of decimal digits represented an integer according to the ordinary Arabic convention. This same integer could also be represented in the scale of two and would then appear as a sequence of 0's and 1's. There is an infinity of alternative possible conventions. However we are not obliged to choose any one of them. The possibility of this translation process was only mentioned to show that there need be no loss of generality involved in using only two symbols.

Although we shall not need these translation conventions we shall often wish to interpret a sequence of 0's and 1's as meaning some integer. The most natural convention to choose is that by which the value of a 1 in the rth position from the right hand is 2^{r-1}, so that 25 is represented by 10011 instead of 11001. These facts may be described by saying that the machine uses 'the scale of two with the most significant digits at the right hand end'.

Although the scale of two is appropriate for use within an electronic computer it is not so suitable for work on paper, and it is not possible to avoid paper work altogether. Without attempting to explain the reasons at this stage let us accept that there are are occasions when it is desirable to write down on paper the sequence of symbols stored in some part of the machine. Suppose for instance that the sequence was

10001110111010001001100011100101010101101100100110

The copying of such sequences is slow and very liable to inaccuracy. It is very difficult to 'keep one's place'. It is therefore advisable to represent such a sequence on paper in a different form not subject to these difficulties. The method chosen is to divide the sequence into blocks of five

10001 11011 10100 01001 10001 11001 01010 10110 11001 00110

and then to replace each block by a single symbol, according to the table below. The above sequence then becomes `Z"SLZWRFWN`.

0	00000	/	11	11010	J	22	01101	P
1	10000	E	12	00110	N	23	11101	Q
2	01000	@	13	10110	F	24	00011	O
3	11000	A	14	01110	C	25	10011	B
4	00100	:	15	11110	K	26	01011	G
5	10100	S	16	00001	T	27	11011	"
6	01100	I	17	10001	Z	28	00111	M
7	11100	U	18	01001	L	29	10111	X
8	00010	$\frac{1}{4}$	19	11001	W	30	01111	V
9	10010	D	20	00101	H	31	11111	£
10	01010	R	21	10101	Y			

These symbols are essentially the teleprinter code, except that the combinations `00000`, `01000`, `00010`, `11011`, `11111`, which in true teleprint are represented by

no effect, line feed, space, carriage return, figure shift, letter shift

respectively have here been given the representations /, @, :, $\frac{1}{4}$, ", £. These symbols have been chosen so as to enable the upper case of a typewriter to be used throughout. In manuscript or with other typewriters we permit the synonyms % for /, $\frac{1}{2}$ for $\frac{1}{4}$, $

for £. With certain kinds of teleprint apparatus it may also be necessary to permit the synonyms `2` for `@`, `4` for `:`, `8` for $\frac{1}{4}$, `5` for `"`, `0` for £. These six combinations will be known as 'stunts'.

The user is strongly recommended to learn the above table. A number of aids to computation in the scale of 32 are given on Figs. A, B, C & D [in the appendices]. These include addition and multiplication tables, special tables to assist in multiplication by powers of two, powers of 10 in the scale of 32, and aids to decimal-teleprint conversion. In principle it is possible to do without these aids for the machine itself can do all the conversion processes required. In practice it frequently happens that some single number is required in the scale of 32, and it is found less trouble to do the conversion by hand than to use the machine. To convert a decimal number less than 1 to scale of 32 multiply by 1024 subtracting and recording the integral part at each stage. This can be done very quickly with a Brunsviga [mechanical calculator] with transfer. The integral parts obtained may be broken up into two teleprint characters with the aid of the table on Fig. B.

3 The Forms of Storage Used

The information store in the Mark II machine consists of the magnetic store and the electronic store. The information in the magnetic store is of considerable volume viz. 655360 binary digits: in other words it corresponds to paper on which is written 655360 digits each of which might be either 0 or 1. But this information is not particularly readily available. It is (to maintain the analogy) as if it were written in a book. In order to find any required piece of information it is necessary to open the book at the appropriate page. The electronic store has a considerable smaller capacity viz. 20480 digits but this information is much more readily available and is to be compared rather to a number of sheets of paper exposed to the light on a table, so that any particular word or symbol becomes visible as soon as the eye focuses on it.

The information in the magnetic store consists of magnetised areas of nickel on the cylindrical surface of a rotating wheel. Each digit stored is represented by one magnetised area. These 655360 areas are arranged in 256 tracks of 2560 digits each. The centres of the areas forming one track lie in a plane perpendicular to the axis of the wheel (and therefore on a circle). The 256 planes are equidistant; the 2560 digit areas on one circle are not however equidistant. The information in one track is further subdivided into two equal parts, which may be described as the left page and the right page if we continue to follow the simile of the book.

The information in the electronic store consists of 20480 digits stored in 16 'tubes' of 1280 digits each. [In fact, all sixteen tubes were not expected to be available; see p. 45.] A tube thus contains the same amount of information as a half-track or page of magnetic store. It may also be described as an 'electronic page'. Its geometrical arrangement is however very different. A tube of information is divided into 64 'lines' of 20 digits each. These digits are stored as charges on the inner surface of the front of a cathode ray tube, the digits of one line forming a straight horizontal segment. Another line is stored in the continuation of this segment. The other lines are in parallel segments, the whole forming a rectangular array. (See photograph [omitted from this typescript]). The lines may be numbered consecutively through the 16 tubes. They could be numbered 0, 1, ..., 1023. However, this numbering is seldom used. One prefers to use the labelling obtained by transforming to the scale of 32 i.e. to teleprinter code, thus the lines are known as `//`, `E/`, . . . , `££`. In their geometrical arrangement on the tubes they are as below

Tube 0		*Tube 1*			*Tube 15*	
`//`	`/E`	`/@`	`/A`		`/V`	`/£`
`E/`	`EE`	`E@`	`EA`		`EV`	`E£`
`@/`	`@E`	`@@`	`@A`	. . . . . .	`@V`	`@£`
⋮	⋮	⋮	⋮		⋮	⋮
`£/`	`£E`	`£@`	`£A`		`£V`	`££`

Each tube may naturally be described as formed of two 'columns'. Thus the second character describing a line gives the column in which it is to be found. It might appear at first sight as if the arabic labelling of the lines would be simpler, but this is not so. The names used above for the lines have to be used in the instructions (Fig. E below) and in every other automatic connection. It is therefore desirable that the programmer should use them also.

Associated with each line is a 'line pair' or 'long line' consisting of that line together with the next line, e.g. the lines `R/` and `J/` together form a line pair and so do `BH` and `GH`. The programmer is recommended to use the former in preference to the latter. (As yet of course he is in no position to use either.) The two long lines referred to have the names `R/` and `BH`. There is an exception to the rule that a line pair always consists of the line named together with the next. A line pair never consists of parts from different tubes. When the line named is the last of a tube the pair consists of the last of the tube followed by the so called 'sixty-fifth line' of the tube. On account of this and other reasons one is recommended to use even-numbered line pairs in preference to odd-numbered ones, especially where consecutive sequences of lines are involved.

Although the information on the magnetic wheel is arranged geometrically so differently from that in the electronic store it may be found convenient to imagine it as if it were similarly arranged. There is very little to interfere with the illusion. The only convenient method for making the content of a track visible ('opening the book') is to copy the track onto a pair of tubes, a process which effectively conceals the true arrangement of the digits of the track. This way of thinking permits us to divide up the magnetic information also into lines or line-pairs. There is however a good deal of ambiguity about the naming of these lines, for a page of magnetic information could be copied onto any one of the 16 electronic tubes. We make no attempt to remove this ambiguity and accept that there are 16 alternative names for a line stored on the wheel, e.g. the same line could also be known as track 14 (left) `I/` or track 14 (left)

IT. There is however a presumption that if it is known by the latter name then in its principal application track 14 (left) will be copied onto tube 8 [3 in the original].

On each tube there is an additional 'identification' line, sometimes known as the 'sixty-fifth line'. This line does not share in the normal activities of store-lines. Its properties are described on p. 25. There is also a 'sixty-fifth line' in each half magnetic track. Its properties are described on p. 28.

4 Description of the Reduced Machine

We shall describe the machine by first explaining the behaviour of another machine obtained by omitting from the Mark II machine a number of its parts and facilities. This machine will be called the 'reduced machine'. The full Mark II machine can then be described in terms of a number of modifications of the reduced machine. Programmes made up for the reduced machine can actually be run on the full machine. For the benefit of those who know the structure of the whole machine we may say that the reduced machine is obtained by omitting the wheel, the B tube and the multiplier, the input and output, and using only a forty digit accumulator and a selection of functions.

The state of the reduced machine may be described by

- The content of the electronic store, i.e. for each of the 1024 lines //,...,££, its content as 20 binary digits or four teleprint characters. (The 'sixty-fifth lines' are not included.)
- The content of the accumulator. This is eight teleprint characters.
- The content of the control, or instruction number (I.N.). This is two teleprint characters.[1]

We are not however interested in the state of the machine at every moment from a programming point of view. We shall be content to know its state at isolated moments such that we can reasonably say that the machine has carried out one 'step' between two consecutive such moments. Fortunately the construction of the machine admits of our choosing such moments satisfactorily. They are the moments where the so called 'prepulses' or 'completion signals' occur. The state of the machine at one prepulse is completely determined by its state at the previous one.[2]

There is thus a function Φ such that if Σ_0, Σ_1,...,Σ_r,... are the consecutive states of the machine (at prepulses) then $\Sigma_{r+1} = \Phi(\Sigma_r)$ for each r. It remains to describe

[1] The control tube also holds the 'present instruction' (P.I.) also known as the 'actual instruction', but the behaviour of the machine can be more easily explained if we do not assume that this forms part of the 'state'. Also the I.N. line can be seen to contain four characters, but two are irrelevant.

[2] This description is slightly inaccurate owing to the fact that the obeying of certain instructions lasts until after the next prepulse. But this makes no practical difference for the programmer, and when manual prepulses are used (p. 35) the effect does not occur.

the function Φ. It is usual to describe it in terms of 'obeying an instruction'. The original state Σ of the machine determines an instruction $\mathbf{I}(\Sigma)$, and this instruction gets 'obeyed', i.e. the final state $\Phi(\Sigma)$ or Σ' is determined by $\mathbf{I}(\Sigma)$ and Σ. This is simply a way of saying that $\Phi(\Sigma)$ can be written in the form $\Psi(\mathbf{I}(\Sigma), \Sigma)$. We may describe $\Psi(\mathbf{I}, \Sigma)$ as 'the result of obeying the instruction $\mathbf{I}$ when the machine is in state Σ'. The step from writing $\Phi(\Sigma)$ to writing $\Psi(\mathbf{I}(\Sigma), \Sigma)$ is not by itself a very helpful one, for any function Φ could be expressed in this form (e.g. even if $\mathbf{I}(\Sigma)$ always had the same value for every Σ). But there are restrictions on the form of Ψ which do make this step helpful. The instruction consists of two parts, the line name and the function symbol. The restriction on Ψ may now be stated as follows. $\Psi(\mathbf{I}, \Sigma)$ does not differ from Σ in any part of the electronic store except in the line-pair named in $\mathbf{I}$. Further, this is the only part of the store whose content is relevant to any changes which do take place.

The function $\Psi(\mathbf{I}, \Sigma)$ must be described by giving its form for the various function symbols case by case. The instruction $\mathbf{I}(\Sigma)$ is also known as the 'content of the P.I. line'. $\mathbf{I}(\Sigma)$ is obtained as follows. Add one to the content of control (I.N.). This gives the name of the line whose content is $\mathbf{I}(\Sigma)$. The line-pair name is contained in the first two (teleprint-) characters of $\mathbf{I}(\Sigma)$, and the function symbol in the last two. The function symbol consists of one of the characters `/` or `T` followed by a second character. This permits us 64 function symbols but (in the reduced machine) we shall assume that only the nine listed below ever occur. In the equations which we give $\mathbf{S}$ represents the content of the named line interpreted as an integer (or more strictly as a residue mod 2^{40}), likewise $\mathbf{A}$ represents the content of the accumulator, and $\mathbf{C}$ that of the I.N. line. This last is reckoned modulo 2^{10}. Dashed letters refer to the contents of these after the instruction has been obeyed. The equations $\mathbf{S}' = \mathbf{S}$, $\mathbf{A}' = \mathbf{A}$, $\mathbf{C}' = \mathbf{C} + 1$ are to be understood wherever $\mathbf{S}'$, $\mathbf{A}'$, $\mathbf{C}'$ are respectively not mentioned in the equations.

We shall frequently use the word 'line' for line-pair in cases where it is evident that a long line is meant. We shall use 'short line' when we wish to emphasise that a long line is not meant.

Function symbol	*Equations*
`/H`	$\mathbf{C}' = \mathbf{C} + 1 \pmod{2^{10}}$ if $2^{40} < \mathbf{A} \leq 2^{39}$
	$\mathbf{C}' = \mathbf{S} \pmod{2^{10}}$ otherwise
`/P`	$\mathbf{C}' = \mathbf{S}$
`/S`	$\mathbf{S}' = \mathbf{A}$
`T/`	$\mathbf{A}' = \mathbf{S}$
`T:`	$\mathbf{A}' = 0$
`TI`	$\mathbf{A}' = \mathbf{A} + \mathbf{S}$
`TN`	$\mathbf{A}' = \mathbf{A} - \mathbf{S}$
`TF`	$\mathbf{A}' = -\mathbf{S}$
`TK`	$\mathbf{A}' = 2\mathbf{S}$
`T£`	(no effect)

5 Examples of programmes on the reduced machine.

1) As a first example we will take the case of a programme for the calculation of a product by repeated addition. The numbers to be multiplied are held in lines `/C` and `@C` and the product eventually appears in `:C`. The line `/C` gets altered in the process. The instructions to effect this multiplication are in tube 0, as shown below. Lines `DS` and `JS` also have special contents as shown.

MULREP listing

[A typographical error has been corrected — the instruction in line `S/` was given in the original as `/CT/`, which disagrees with the check sheets given further below, and also simply doesn't work in context. Similar errors are endemic to the machine-code listings and commentary further on, and will be corrected without further comment. There is, however, yet another error which is duplicated on the check sheets, which I note below in square braces as usual.]

```
                // /CT/          It is assumed that we have
                E/ DSTI          the following fixed contents:
                @/ D//H
                A/ R//P               DS ££££
MULREP          :/ /C/S               RS ££££
                S/ :CT/               JS ////
                I/ @CTI
                U/ :C/S
                ¼/ JS/P          [should be DS/P?]
                D/ A/
                R/ @/
```

The names of the lines have been shown on the left of their contents.

In order that a routine may be of some use it is necessary that a statement of what it will do should be written down in an unambiguous form. For the above routine this might read

> MULREP. The routine is entered at `//` and left at `A/`. Its effects are described by the equation $[\texttt{:C}]' = [\texttt{/C}][\texttt{@C}] + [\texttt{:C}]$

Here MULREP is just a code name for the routine. The above description of the properties of MULREP are to be understood to mean "suppose that at some moment the machine contains the instructions of MULREP (i.e. lines `//` to `R/` as above also `DS`, `JS`) and that control contains *££*. The contents of the long lines at this moment will be described by writing their names in square brackets. Then the machine will afterwards eventually contain `@/` in control. We denote the contents of lines at the first such moment by their names in square brackets with dashes. The equation

$[\texttt{:C}]' = [\texttt{/C}][\texttt{@C}] + [\texttt{:C}]$ then holds, and also $[\nu]' = [\nu]$ unless ν is $\texttt{:C}$, $\texttt{/C}$, or one of the lines $\texttt{GK}$, $\texttt{MK}$, $\texttt{VK}$ or on page 0 or 1."

We may make a number of remarks about this statement and the routine.

(a) The contents of control which are mentioned are less by 1 than the values given for the entry and exit points of the routine. The reason for this should become clear to the reader if he refers to the method of obtaining **I** from **C**. The first step was to add 1 to the content of control.

(b) We use the 'dash notation' as we did with single instructions. In either case undashed expressions describe the state before the operation, dashed expressions values after.

(c) The equation $[\texttt{:C}]' = [\texttt{/C}][\texttt{@C}] + [\texttt{:C}]$ should read rather
$[\texttt{:C}]' = [\texttt{/C}][\texttt{@C}] + [\texttt{:C}] \pmod{2^{40}}$.

(d) In squares where the character is irrelevant we leave blanks.

(e) We have in this case arranged that after the operation is over the machine goes into a 'loop' in which its states are repeated. This is called a 'loop stop'.

(f) The conditions concerning $\texttt{GK}$, $\texttt{MK}$, $\texttt{VK}$, and pages 0, 1 are regular conventions. None of these are actually altered in this case, but it is convenient to be able to alter pages 0, 1 (which contain the instructions in normal cases) and the other three lines, without being obliged to mention the fact in the official report on the routine.

We sometimes also use the notation $[\nu]_s$ to mean the content of the short line ν.

It is of course important that some efforts be made to verify the correctness of the assertions that are made about a routine. There are essentially two types of method available, the theoretical and the experimental. In the extreme form of the theoretical method a watertight mathematical proof is provided for the assertion. In the extreme form of the experimental method the routine is tried out on the machine with a variety of initial conditions and is pronounced fit if the assertions hold in each case. Both methods have their weaknesses. This whole question is taken up in greater detail on pp. 68-70. For the present let us just mention one form of verification which is described as 'check sheets'. It is quite easy to follow out in detail, on paper, the behaviour of the machine for a few steps, or even a few hundred steps. If it were necessary to record the complete state of the machine after each instruction has been obeyed it would be a heavy undertaking. However it is quite sufficient to record the content of the accumulator and control, with the last instruction obeyed and any alterations that may be made in any lines of the store. This has been done on an accompanying sheet for the routine MULREP, taking the case of a multiplicand 27 and a multiplier 2. It is necessary to take a very small multiplier if the process is to come to an end in an reasonable time. It will of course be recognized that the process in question is not a practically suitable one for multiplication, and is only given as an example.

Check sheets

```
                                               / C   @///////
              MULREP                           @ C   "///////
                                               : C   @///////
     / /   / C T /    @///////
     E /   D S T I    E///////
     @ /   D / / H    E///////
     : /   / C / S    E///////         / C   E///////
     S /   : C T /    ////////
     I /   @ C T I    "///////
     U /   : C / S    "///////         : C   "///////
     ¼ /   J S / P    "///////
     / /   / C T /    E///////
     D /   D S T I    ////////
     @ /   D / / H    ////////
     : /   / C / S    ////////         / C   ////////
     S /   : C T /    "///////
     I /   @ C T I    PE//////
     ¼ /   J S / P    PE//////         : C   PE//////
     / /   / C T /    ////////
     E /   D S T I    ££££££££
     @ /   D / / H    ££££££££
     A /   R / / P
     A /   R / / P
```

[Note a bug: The line at `JS` contains `////`, i.e. zero, so the instruction `JS/P` should transfer control to `E/`, not `//` as shown here; this follows both from Turing's description and the behavior of the other two branches, `D//H` and `R//P` in the example.]

In the making of check sheets and such matters detail of procedure is of great importance. The work should be done on paper with quarter inch squares on which vertical lines are ruled in ink or printed. (However if forms are printed they should be such as suit the full machine.) All other writing should be in pencil. Each line refers to one state of the machine. In the first column is found the instruction number, and in the second the corresponding instruction. In the third is the content of the accumulator. In the fourth the line named in the instruction is repeated if that instruction results in the changing of that line, and in the fifth is the new content of that (long) line. These of course do not describe the state of the machine completely, but if one wishes to know the content of an other line one may glance up the fourth column until one finds the name of the line in question. If the recommendation to use only such long lines as have even numbers has been respected this glance will be sufficient. If however the recommendation has been ignored it will be necessary to take note also of the occurrence of the two lines whose numbers differ by one from that of the line in question. In the content of control, the instruction and the name of the altered line one complete quarter inch square is allowed for each teleprint character. For the contents of the accumulator and the store lines however two teleprint characters should be written in each square. This does not cause excessive crowding, and this economy of space becomes important with the full machine. The convention in check sheets about the content of control needs some explanation. If each line is regarded as corresponding to a prepulse the contents of the accumulator and of the store lines agree, and the instruction column gives the corresponding content for the P.I. line of the control tube. The first column however is not the content of control but the name of the line from which the instruction was taken. This will be a distinction without a difference in the majority of cases. The true value of **C** for each line is to be obtained by adding 1 to the value in the first column of the immediately previous line of the check sheets [unless the previous line was a taken branch].

Certain abbreviations may be permitted to reduce the amount of writing involved. When the content of the accumulator is unaltered it is sufficient to leave the entry in the accumulator column blank. When this content consists of the same character repeated eight times one may write the character once and large. When the content of control is increasing steadily by 1 one may leave it blank.

It will be noticed that blocks of consecutive lines may be copied direct from the routine to the check sheets.

As a second example we give the routine SUMPGA. All abbreviations are applied in its check sheets. The routine also has been set out in the standard manner. The first character of the line name is given in a narrow column between the two columns representing the store, and the second character, which is common to all the lines of the column, is given at the side of the column at its head. The manner of entering and leaving the routine has been satisfactorily described in the 'official account' given with the routine. The manners chosen are quite appropriate for the reduced machine but not for the full machine. The methods used with the full machine are described on p. 42.

[SUMPGA listing]

/	V E T /	/		E
	M E T I	E		
	: / / S	@		
	/ C T /	A		
		:		
		S		
	M E T /	I		
	G E T N	U		
	M E / S	¼		
	D S / H	D		
		R		
		J		
		N		
		F		
		C		
		K		
		T		
		Z		
		L		
		W		
		H		
		Y		
		P		
		Q		
		O		
		B		
		G	@ / / /	
		"	/ / / /	
		M	V E / /	
		X	/ / / /	
		V	/ ¼ T I	
		£	/ C / S	

SUMPGA
Enter at `//` leave `R/`
$[\texttt{/C}]' = [\texttt{/C}] + \Sigma_{r=0}^{31}[\texttt{/}\tfrac{1}{4} + 2r](mod\ 2^{40})$

Method. Note. When about to obey `VET/` with $2n$ in `M/` we have
$[\texttt{/C}]' = [\texttt{/C}] + \Sigma_{r=n+1}^{31}[\texttt{/}\tfrac{1}{4} + 2r](mod\ 2^{40})$

It is assumed that [`DS`] = ££££££££

[Further note on operation: note that `://S` at `/@` stores a long line, and therefore writes instructions into both short lines `:/` and `S/`]

[SUMPGA check sheet]

```
                                        / ¼   E   /
                                        @ ¼   @   /
                                        V D      /   T
                                        / C      /
/ /   V E T /   /¼TI/C/S
      M E T I   VDTI/C/S
      : / / S                           : /   VDTI/C/S
      / C T /      /
: /   V D T I      /   T
S /   / C / S                           / C      /   T
      M E T /   VE  /
      G E T N   ME  /
      M E / S                           M E   ME  /
      D S / H
/ /   V E T /   /¼TI/C/S
      M E T I   MDTI/C/S
      : / / S                           : /   MDTI/C/S
      / C T /      /   T
: /   M D T I      /   O
S /   / C / S                           / C      /   O
      M E T /   ME  /
      G E T N   GE  /
      M E / S                           M E   GE  /
      D S / H
/ /
      .
      .
      .
      M E / S                           M E   £
      D S / H
R /
```

Exercise

[In the original, given before the listing and check sheet of SUMPGA; I presume this was an error on the part of the typist.] Provide a routine, with check sheets and official account, for the purpose of putting a number into 'standard form', by multiplying it by a power of 2. A number X is in standard form if $2^{39} \leq X < 2^{40}$.

6 The multiplier and the double length accumulator

We shall now begin to describe the various respects in which the full machine differs from the reduced machine. Most of these differences are essentially independent. One may satisfactorily learn the effect of each difference as if added to the reduced machine in the absence of the others, and having learnt these effects will be in a position to manage the whole machine.

We begin by considering the effect of adding a multiplier to the machine. If we regard a standard number as consisting of forty binary digits then the product of two such numbers will occupy eighty digits. For this reason it is necessary to have an eighty digit accumulator. This decision in itself requires us to adopt a more sophisticated attitude to our numbers and our rows of digits. In the reduced machine it was almost possible to regard the long lines as representing integers, but it was necessary to admit that they were really to be reckoned modulo 2^{40}. Long lines with a '1' in the most significant place could be regarded as representing negative numbers, provided that it is accepted that positive numbers greater than $2^{39}-1$ cannot be represented. With the double length accumulator similar considerations apply with greater complexity. Numbers held in the store must be reckoned modulo 2^{40} as before, but numbers in the accumulator must be reckoned modulo 2^{80}.

In order to be able to express these matters clearly it is necessary to have notations which draw the essential distinctions, though these distinctions may appear pedantic, and though in a majority of applications the notation is not needed in all its detail. We distinguish therefore between 'rows of digits' and numbers. I do not think that either of these expressions needs much elaboration. By numbers I shall mean real numbers. The content of any part of the machine will be a row of digits or an assembly of such rows, and not a number (with the exception of the multiplicand). Additions and multiplications are however performed on numbers and not rows of digits. However, in order that the processes of the machine may be described in terms of these operations it is necessary to be able to relate rows of digits to numbers, and vice-versa.

It would be sufficient in theory to be able to connect one number with each row, in such a way that all rows got different numbers. In practice four possible conventions present themselves particularly forcibly. We assume that our row of digits is of length R and the rth digit is ϵ_{r-1}.

- *The plus-convention:* The associated number is $\Sigma_{r=0}^{R-1}\epsilon_r 2^r$
- *The plus-or-minus convention:* The associated number is $\Sigma_{r=0}^{R-1}\epsilon_r 2^r - \epsilon_{R-1}2^R$
- *The fractional plus-convention:* The associated number is $2^{-R}\Sigma_{r=0}^{R-1}\epsilon_r 2^r$
- *The fractional plus-or-minus convention:* The associated number is $2^{-R}(\Sigma_{r=0}^{R-1}\epsilon_r 2^r - \epsilon_{R-1}2^R)$

We shall use all of these conventions, both in connection with the store lines, and with the accumulator. To convert a row into a number according to one of these conventions one writes respectively $+$, $\pm$, $f+$, or $f\pm$ as a suffix after the content of the row.

As regards the converse process, that of defining rows of digits in terms of numbers it will suffice to be able to take any sequence of consecutive digits from the binary expansion of a number. Accordingly we say that for any real number α, and integers m, n for which $n \geq m$, $\{\alpha\}_m^n$ is the row of digits forming the coefficients of the mth to the nth powers of two in the binary expansion of α. If possible this expansion is to be terminating.

A number of statements are made below in illustration of these conventions.

1. $\texttt{//G}_+ = 13 \times 2^{11}$
   $\texttt{//G}_\pm = -3 \times 2^{11}$
   $\texttt{//G}_{f+} = \frac{13}{16}$
   $\texttt{//G}_{f\pm} = \frac{-3}{16}$

2. $|(\{\alpha\}_{-20}^{-1})_+ - \alpha| < 2^{-20}$ provided $0 < \alpha < 1$
   $(\{1\}_{-20}^{-1})_+ = 0$

3. $|\{[\texttt{/C}]_+[\texttt{@C}]_+\}_{40+}^{79}[\texttt{:C}]_+ - \{[\texttt{/C}]_+[\texttt{:C}]_+\}_{40+}^{79}[\texttt{@C}]_+| < 2^{40}$

A further convention which may be used in this connection is the use of the symbol Θ. This is somewhat analogous to the use of O, o in analysis. One uses $O(f(x))$ to mean 'some function $\phi(x)$ such that there exists a positive K satisfying $|\phi(x)| < Kf(x)$ for all sufficiently large x'. There is also an understanding that the functions ϕ and constants K may be different at every appearance of O. The use of Θ is similar. $\Theta(x)$ means simply 'some quantity α satisfying $|x| < \alpha$' and again may be different on every appearance. Thus for instance from $\Theta(1) = \frac{1}{2}\Theta(1)$ one cannot conclude $\Theta(1) = 0$ for the two appearances of $\Theta(1)$ might have the values $\frac{1}{4}$ and $\frac{1}{2}$. Using this convention 2) above could be written

$$\{\alpha\}_{-20+}^{-1} = \alpha + \Theta(2^{-20}) \text{ provided } 0 < \alpha < 1$$

In terms of these conventions we may explain the properties of the multiplier as follows. To describe the state of the Mark II machine we give the states of the stores and control as in the reduced machine, but the accumulator is an eighty digit one. We also have to describe the state of the 'multiplicand'. This is considered to be an integer which may take any value from -2^{39} to $2^{39}-1$. In this respect the multiplicand is exceptional. The contents of all other parts of the machine are considered as rows of digits. Having explained this a number of further function symbols should become intelligible, viz. those marked m in the list below.

[The appendices to the manual do contain a list of instructions (in fact, two; one as part of a quick-reference sheet), but oddly, neither is marked as described. From those lists, the function symbols directly relevant to the multiplier are as follows, where **D** denotes the value of the multiplier:

Function symbol	*Equations*
`/C`	$\mathbf{D}' = \mathbf{S}_+$
`/K`	$\mathbf{D}' = \mathbf{S}_\pm$
`/¼`	$\mathbf{A}' = \{\mathbf{A}_+ - \mathbf{DS}_+\}_0^{79}$
`/D`	$\mathbf{A}' = \{\mathbf{A}_+ - \mathbf{DS}_\pm\}_0^{79}$
`/N`	$\mathbf{A}' = \{\mathbf{A}_+ + \mathbf{DS}_+\}_0^{79}$
`/F`	$\mathbf{A}' = \{\mathbf{A}_+ + \mathbf{DS}_\pm\}_0^{79}$

There are several additional function symbols whose properties can only be explained (or must be redefined) with reference to the 80-bit accumulator of the real machine, as opposed to the 40-bit accumulator of Turing's "reduced machine". If we define $\mathbf{L} = \{\mathbf{A}\}_0^{39}$ and $\mathbf{M} = \{\mathbf{A}\}_{40}^{80}$, then these are as follows:

Function symbol	*Equations*
`/E`	$\mathbf{S}' = \mathbf{M}$
`/A`	$\mathbf{S}' = \mathbf{M}$, $\mathbf{A}' = \{\mathbf{L}_+\}_0^{79}$
`/S`	$\mathbf{S}' = \mathbf{L}$
`/I`	$\mathbf{L}' = \mathbf{M}$, $\mathbf{M}' = \mathbf{L}$
`/U`	$\mathbf{S}' = \mathbf{L}$, $\mathbf{A}' = \{\mathbf{M}_+\}_0^{79}$
`T/`	$\mathbf{A}' = \{\mathbf{S}_+\}_0^{79}$
`TA`	$\mathbf{S}' = \mathbf{L}$, $\mathbf{A}' = \{0\}_0^{79}$
`T:`	$\mathbf{A}' = \{0\}_0^{79}$
`TI`	$\mathbf{A}' = \{\mathbf{A}_+ + \mathbf{S}_+\}_0^{79}$
`T¼`	$\mathbf{A}' = \{\mathbf{S}_\pm\}_0^{79}$
`TN`	$\mathbf{A}' = \{\mathbf{A}_\pm - \mathbf{S}_\pm\}_0^{79}$
`TF`	$\mathbf{A}' = \{-\mathbf{S}_\pm\}_0^{79}$
`TC`	$\mathbf{A}' = \{\mathbf{A}_\pm + \mathbf{S}_\pm\}_0^{79}$
`TK`	$\mathbf{A}' = \{2\mathbf{S}_\pm\}_0^{79}$

Further supplementary tables of function codes will be added after subsequent sections, except where the relevant instructions are specifically described in the original text.]

We may allow certain abbreviations as admissible, viz.

(a) Where it is clear that a row of digits is meant, and the number N of these digits is known one may write U for $\{U\}_0^N$, where U is an expression representing a real number, e.g., the equation for `/F` may be abbreviated from $\mathbf{A}' = \{\mathbf{A}_+ + \mathbf{DS}_\pm\}_0^{79}$ to $\mathbf{A}' = \mathbf{A}_+ + \mathbf{DS}_\pm$.

(b) When it is evident that a real number is meant, and it is irrelevant whether a suffix $+$ or $\pm$ is used (or irrelevant whether $f+$ or $f\pm$ is used) one may omit the suffix (or use f only) e.g. the equation for `/F` may be abbreviated further to $\mathbf{A}' = \mathbf{A} + \mathbf{DS}_\pm$.

[Turing makes little use of the notation introduced here in the rest of the manual, except briefly towards the end when discussing systematic errors in programs.]

7 The logical operations

There are a number of operations which can be applied to rows of digits and which have no particular relation to the interpretation of those rows as numbers. The only ones which we use are what we call the 'logical operations', on account of a connection with the calculus of propositions in logic. In order to bring out this connection one should identify 1 with 'truth' and 0 with 'falsity'. The symbol '$\vee$' may be read as 'or', '$\wedge$' may be read as 'and', '$\not\equiv$' as 'not equivalent to' [and $\sim$ as 'not']. These symbols satisfy the equations

$$\begin{array}{llll} 0 \vee 0 = 0 & 0 \wedge 0 = 0 & 0 \not\equiv 0 = 0 & \sim 0 = 1 \\ 0 \vee 1 = 1 & 0 \wedge 1 = 0 & 0 \not\equiv 1 = 1 & \sim 1 = 0 \\ 1 \vee 0 = 1 & 1 \wedge 0 = 0 & 1 \not\equiv 0 = 1 & \\ 1 \vee 1 = 1 & 1 \wedge 1 = 1 & 1 \not\equiv 1 = 0 & \end{array}$$

However we shall apply these symbols to rows of digits rather than to single digits. The value for each digit is then to be calculated by means of the above table from the corresponding digits of the arguments, e.g.,

$$(01110) \vee (10010) = (11110)$$

The main application of these operations occurs where several different pieces of information are packed into one line. For this purpose the operation '$\wedge$' is of the most use. It can be used for breaking a line up into its various significant parts. The operation '$\vee$' may also be used for combining parts together, but this can usually also be done with the aid of addition. The operation '$\sim$' can be obtained by using '$\not\equiv$' in combination with a row of 1's. [The following function codes are used to call for these operations:

Function symbol	*Equations*
`TD`	$\mathbf{A}' = \mathbf{A} \vee \{\mathbf{S}_\pm\}_0^{79}$
`TR`	$\mathbf{A}' = \mathbf{A} \wedge \{\mathbf{S}_\pm\}_0^{79}$
`TJ`	$\mathbf{A}' = \mathbf{A} \not\equiv \{\mathbf{S}_\pm\}_0^{79}$
`TS`	$\mathbf{S}' = \mathbf{L} \vee \mathbf{S}$, $\mathbf{A}' = 0$
`TE`	$\mathbf{L}' = \mathbf{S}' = \mathbf{L} \vee \mathbf{S}$, $\mathbf{M}' = \mathbf{M}$

where, as before, we define $\mathbf{L} = \{\mathbf{A}\}_0^{39}$ and $\mathbf{M} = \{\mathbf{A}\}_{40}^{80}$. Note that the `TS` and `TE` operations are deprecated as "disadvantageous" on p. 50, where the programmer is enjoined against their use.]

Example of use of '$\wedge$'

The name of a certain line is contained in the first two characters of `/`$\frac{1}{4}$. It is required to transfer this line to `/C`.

/	/ ¼ T /	/	£ £ / /	E
	/ E T R	E	/ / / /	
	@ E T D	@	/ / T /	
	: / / S	A	/ C / U	
		:		
		S		
		I		
		U		
		¼		
		D		
		R		
		J		

[Note once again that `://S` at `A/` writes a `/C/U` instruction from `AE` into `S/`, in addition to writing a `T/` instruction into `:/`, since it is a long-line store.]

Example for reader

Provide a sequence of instructions which will produce a line consisting of the first 10 digits of [`/C`] together with the last line of [`@C`].

8 The B-tube.

It will have been noticed that a number of the examples already given involve the formation of an instruction in the accumulator by addition, the copying of this instruction into the list of instructions, and the subsequent obeying of it. This is a rather clumsy process, and appears particularly so when it is desired to add the same number to several 'skeleton instructions' to obtain actual instructions. The 'B-tube' is primarily a device to enable this to be done more readily. It provides for eight different quantities which may be added to any instruction. One such quantity normally *has* to be added to each instruction. This could become a nuisance, but we avoid any such difficulty by adopting the convention that normally one of those lines (called **B**0) is to be zero.

We must therefore extend still further the quantities which go to make up the state of the machine and include the eight four-character lines of the B-tube. These are called **B**0, ..., **B**7. We also have to modify considerably the description of the manner in which the 'instruction $\mathbf{I}(\Sigma)$ next to be obeyed' is to be determined from the state Σ of the machine. We first carry out the process described for the reduced machine, viz. add 1 to control and take the short line whose name is the resulting quantity. This however is not the instruction itself but what we may call the 'presumptive instruction'. This presumptive instruction we will call **J**. It is a row of 20 digits. This presumptive instruction is to be divided into four parts:

- Address part, digits 0-9
- B-line part, digits 10, 11, 12
- Spare digit, 13
- Function number part, digits 14-19

We must next look at the function number part of the presumptive instruction and decide whether it is 'B-normal' or 'B-exceptional'. The B-exceptional cases are those where digits 14, 18, 19 are all 1, i.e., where the third character is one of `T`, `Z`, ..., `£` and the fourth one of `O`, `B`, ..., `£`. The next step is to form the 'actual instruction'. In the B-exceptional cases this is identical with the presumptive instruction, but in the B-normal cases it is obtained by adding the content of one of the B lines to the presumptive instruction. The number of the B line concerned is given in the B-line part of the presumptive instruction.

It will be seen on looking at the list of equations that quantities $\mathbf{B}$, $\mathbf{B}'$ appear in certain of them. These are the (before and after) contents of the B line whose number is given in the B line whose number is given in the B line part of the actual instruction. Likewise, $\mathbf{S}$, $\mathbf{S}'$ are the store lines whose addresses are to be found in the actual instruction and the equations applied are to be determined according to the function number part of the actual instruction.

In addition to the B-lines there is yet another quantity to be included in the state of the machine. This is the one-digit row $\mathbf{Q}$, known as the 'B sign flip flop'. Its properties are entirely covered by the equations. It is particularly useful in counting processes.

The B-tube may be considered as having essentially three functions. The primary purpose is the modification of presumptive instructions to give actual instructions. This was the only purpose in the Mark I machine. Since counting operations often go hand in hand with altering the B additives, the Q facility was added, and also the facility for subtracting (`TL` and `TG`) from the B lines as well as setting them (`TT` and `TO`). By adding the functions `TZ` and `TB` it was made possible to use the B tube as a 'shunting station' for 20 digit lines. The complexities of the exceptional instructions were made necessary by the fact that when one is setting a B line one does not usually wish the instruction involved to be modified by what is already in the B line in question. However when using the B line as a shunting station, particularly when transferring from a B line to an 'indeterminate' position it is necessary that the instruction involved should bring two different B-lines into action, one containing information as to the destination of the content of the other. These cases can be covered by the 'B-normal' variants `TT`, `TZ`, `TL` of the more frequently used `TO`, `TB`, `TG` (see p. 57).

[Once again, here is a list of relevant instructions from the full table:

Function symbol	*Equations*
`/T`	$\mathbf{C}' = \mathbf{S}_+$ if $\mathbf{Q} = 1$
	$\mathbf{C}' = \mathbf{C}_+ + 1$ otherwise
`TT`	$\mathbf{B}' = \{\mathbf{S}_+\}_0^{19}, \mathbf{Q}' = \sigma(\mathbf{B}')$
`TO`	as `TT`, but B-exceptional
`TZ`	$\{\mathbf{S}'_+\}_0^{19} = \mathbf{B}$, $\{\mathbf{S}'_+\}_{20}^{39} = \{\mathbf{S}_+\}_{20}^{39}$, $\mathbf{Q}' = \sigma(\mathbf{B}')$
`TB`	as `TZ`, but B-exceptional
`TL`	$\mathbf{B}' = \{\mathbf{B}_+ - \mathbf{S}_+\}_0^{19}, \mathbf{Q}' = \sigma(\mathbf{B}')$
`TW`	as `TL`
`TG`	as `TL`, but B-exceptional

where $\sigma(X)$ is the most significant bit of X. Note also that the "official" dummy instruction, `T£`, is B-exceptional, so that it does not change the state of the machine regardless of the contents of the B tube; it is the only documented B-exceptional instruction aside from those listed above.]

As an example of the use of the B-tube we have again programmed the process of adding up the lines in page 4 [3 in the original]. We shall make a number of remarks about this routine.

a) It will be seen that the use of the B tube for the modification of instructions fits very well with its use counting repetitions of an operation. The two are combined in this routine.

b) On the check sheets we put both the presumptive instruction and the instruction proper in one line in the second column when they differ. There are the alternatives of using two lines, and of having an extra column. The use of an extra line is not very practical as one is liable to copy a long sequence of instructions onto check sheets without noticing that some are only presumptive. If only one line is used this can be rectified with an india-rubber. The extra column would not be sufficiently often used to justify it.

c) As a space economy measure one can often combine an 'addressless instruction' (i.e. one of the instructions `//`, `/L`, `/W`, `/G`, `/V`, `T:`, `T£` for which the store line is irrelevant) with a 'control transfer number', i.e. the first ten digits of a store line referred to in a 'control transfer instruction' (`/H`, `/P`, `/Q`, `/O`, or `/M`). This occurs for instance in the case of the short line `/@`, which contains `E@T:`. The whole line represents an instruction clearing the accumulator. The appearance of `E@` at the beginning of the instruction rather than e.g. `//` or `::` is only relevant to the line as an instruction in that line `E@` becomes brightened momentarily during the obeying of this instruction.

SUMPGB

@	E @ T :	/	V E / /	A
	/ A Q O	E	@ / / /	
	/ $\frac{1}{4}$ Q C	@		
	E A Q G	A		
	/ @ / T	:		
	/ C / S	S	Cue	

Entered at `/@`, left at `S@`
$[\texttt{/C}]' = \{\Sigma_{r=0}^{31}[\texttt{/}\tfrac{1}{4} + 2r]\}_0^{39}$
[Further note: a `Q`α function symbol is a `T`α with **B**7 selected in the B-line portion; see the teleprinter code table on p. 3. So, for example, `QC` is a `TC` operation ($\mathbf{A}' = \mathbf{A}+\mathbf{S}$), except that **B**7 is added to the instruction before it is obeyed.]

				/¼	E//////
				@¼	@//////
				⋮	⋮
				MD	//////E/
				VD	//////@/
/@	E@T:	/	/	VD	//////@/
	/AQO			B7	V E / /
	/¼QC VDTC	///////@/	/		//////@/
	EAQB			B7	M E / /
	/@/T				E @ T :
@@	/¼QC MDTC	///////A/	/		//////E/
	EAQB			B7	G E / /
	/@/T				
@@	/¼QC GDTC	//////TA/	/		/////T//
	EAQB			B7	O E / /
	⋮			⋮	
	EAQB	£££££££A/	/	B7	V £ £ £
	/@/T				E @ T :
	/C/S			/C	££££££A/

Exercise

Make up a routine, with an official account, for the purpose of copying column ¼ onto column D.

9 Miscellaneous special functions

9.1 Dummy stops

It is possible by setting certain switches to arrange that the machine 'stops' when certain instructions are reached. This is effected by arranging that these instructions have no completion signals. The instructions in question are `/L` and `/G`. Each of these can be independently made either a 'time wasting instruction' or a stop.

Dummy stops are very useful in testing routines. The programmer is recommended to insert them at points where major operations may be considered complete. [Note that a `/L` dummy stop is included in the "routine changing sequence", which brings subroutines into electronic storage from the drum; see p. 42.]

9.2 The hooter

When an instruction with function symbol `/V` is obeyed an impulse is applied to the diaphragm of a loudspeaker. By doing this repeatedly and rhythmically a steady note, rich in harmonies, can be produced. This is used to enable the operator to be called to attend to the machine in some way. The simplest case is where the whole of a job is completed and it is required to clear the electronic stores and start something different. All that is then required is to repeat a cycle of instructions including a hoot, e.g.

```
FS    NS/V
CS    FS/P
```

In this case every second instruction will put a pulse into the speaker. These pulses will occur at intervals of 8 beats i.e. 1.92 ms giving a frequency of 521 cycles (about middle C). Or one could use the loop of three instructions

```
O@      /V
G@    P@/V
M@    G@/P
```

which gives a slightly louder hoot a fifth lower in frequency. Single pulses applied to the loudspeaker are distinctly audible as something between a tap, a click, and a thump. This fact can be turned to good account. By putting hoot instructions into programmes at suitable points one is enabled to 'listen in' to the progress of the routine. Some indication of what is going on is given by the rhythm of the clicks that are heard.

9.3 The hand switches

One can set up a row of twenty digits on twenty switches. This row '**H**' can affect the behaviour of the machine through instructions with function symbol `//` or `/Z`. The former of these will be discussed under magnetic transfers. The latter is used for putting small pieces of information into the machine by hand. [The equations for `/Z` are $\{\mathbf{S}'\}_0^{19} = \mathbf{H}$, $\{\mathbf{S}'\}_{20}^{39} = \{\mathbf{S}\}_{20}^{39}$.] Suppose for example that we have a routine for calculating some function of a four character line, and suppose that the calculation takes five minutes. It would then be reasonable to put the arguments in through the switches. This would be particularly so if the arguments used depend partly on the judgment of the experimenter and partly on the values recently obtained, e.g. if one were trying to find a zero of the function, but one was not wishing to repeat the process often enough to mechanise it fully. Again if one were playing chess against the machine this would be the natural way of registering one's moves. [These are allusions to two of Turing's own projects — his very early work on machine chess, and his project to investigate the zeros of the Riemann zeta function.] (See also the 'formal mode', pp. 48-50).

9.4 The position of the most significant digit

In the calculation of logarithms, reciprocals and square roots it is desirable to be able to 'standardize' numbers i.e. to express them in the form $2^n\alpha$ where $\frac{1}{2} \leq \alpha < 1$. This is made possible at high speed by the use of instructions with the function symbol `/@`.

[This instruction also has cryptographic applications which Turing does not mention, but which he would have known very well from his work at Bletchley Park, which was then still very secret. Cryptography is also a significant application of the `/R` function symbol, which counts the number of one bits in a word; Turing refers to this as the "sideways adder" in his quick-reference summary, but does not refer to the instruction at all in the main text of the manual. The relevant equations are:

Function symbol	*Equations*
`/R`	$\mathbf{A}' = \{\mathbf{A}_+ + 2^{40}t(\mathbf{S})\}_0^{79}$
`/@`	$\mathbf{A}' = \{\mathbf{A}_+ + 2^{40}\mu(\mathbf{S})\}_0^{79}$

where $\mu(\mathbf{S})$ is defined as follows:

If $\mathbf{S}_+ \neq 0$, then $2^{\mu(\mathbf{S})} \leq \mathbf{S}_+ < 2^{\mu(\mathbf{S})+1}$
If $\mathbf{S}_+ = 0$, then $\mu(\mathbf{S}) = 0$

and $t(\mathbf{S})$ is the number of one bits in $\mathbf{S}$.]

9.5 The random numbers generator

In the Mark II machine the principle that the state of the machine at one completion signal determines the state at the next is abandoned if instructions with function symbol `/W` are used. The behaviour of the machine is then to be described as a 'stochastic' (i.e. chance-controlled) process and is suitable for calculations concerned with stochastic processes. The instruction `/W` actually puts random digits into the twenty least significant digits of the accumulator.

The following problem is suitable for the use of this facility.

A man in New York starts walking from a street intersection, and at each street intersection decides in which direction to walk by twice tossing a coin (each of the four directions is chosen equally often). It is required to find the probability that before walking twenty blocks he will have succeeded in returning to his starting point. For this purpose New York is to be assumed to be an infinite rectangular lattice of streets and avenues.

9.6 The clock

The quantity $\mathbf{Z}$ is a twenty digit line which is intended to indicate the time. If $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are the contents of $\mathbf{Z}$ at two times t_1 and t_2, measured in seconds, then

$$(1-\sigma(\mathbf{Z}_1))(1-\sigma(\mathbf{Z}_2))\left|\left\{\frac{t_1-t_2}{5}-(\mathbf{Z}_{1+}-\mathbf{Z}_{2+})\right\}_{0\pm}^{14}\right| < \frac{t_1-t_2}{100}+1$$

This is a way of saying that the first fifteen digits of $\mathbf{Z}$ measure the time, the unit being five seconds, to an accuracy of 5%, but that when the most significant digit is 1 the value given cannot be relied on. This only occurs for periods of less than a quarter second at intervals of five seconds.

[Instructions with function symbol `/Y` set the low 20 digits of $\mathbf{S}'$ to $\mathbf{Z}$.]

9.7 The sixty-fifth lines

We mentioned on p. 6 that an extra identification line was added to each magnetic page and that there is a corresponding line on each tube. The only significant effects of these lines are

(a) They may be seen on the monitor tube with the other lines,

(b) they are copied from magnetic to electronic store in reading transfers and checked in read-like checking operations (see. p. 28),

(c) they can be copied into the accumulator by instructions `T@`. This is an irregular function, not corresponding to the normal rules. It is intended that it be used with addresses whose first six digits are all 0, i.e. which are names of lines each of which is the first of some page. Under these circumstances the effect of the instruction can be written $\mathbf{A}' = \{\alpha_+\}_0^{79}$ where α is the content of the sixty-fifth line of the page in question. With other addresses the same equation holds, but α must then be interpreted as the content of the short line preceding the line referred to in the address. For example the effect of `FET@` is $\mathbf{A}' = \{[\mathtt{NE}]_{s+}\}_0^{79}$. It may be observed that this is the only function connecting a short line with the accumulator.

[(d) In loads of odd line-pairs at the end of a page; see p. 5.]

But see also p. 50.

[Relative branches]

[Three branch introductions have been introduced so far — the unconditional `/P`, the **A**-conditional `/H` and the **B**-conditional `/T`. These are all absolute branches, setting $\mathbf{C}' = \mathbf{S}$ if the branch is taken. They have relative variants, `/Q`, `/M` and `/O` respectively, which are identical in their effects except that if the branch is taken, we have $\mathbf{C}' = \mathbf{S} + \mathbf{C} + 1$.]

The time occupied by various operations

The machine is synchronised by an oscillator with a frequency of 100 kc/s. One cycle (occupying 10 μs) of this oscillator may be called a 'digit period'. These digit periods determine the most fundamental rhythm of the machine, but there is another almost equally important rhythm, in which time is divided into 'beats' of 24 digit periods. Four of these periods form what is called the 'blackout period' of the beat. The remaining twenty are used for operations on twenty digit rows, each digit being dealt with in one digit period. The 'prepulses' or 'completion signals' occur in the blackout periods, and are separated by integral numbers of beats, i.e. by multiples of 240 μs. As has been mentioned (p. 6) some instructions overlap their successors, but for programming purposes one may take it that the time consumed in an instruction is that dividing the prepulse initiating it with that initiating its successor. The numbers of beats for each instruction are shown on p.E [in the appendices]. The times for the magnetic instructions are mentioned under the heading of the magnetic wheel.

10 The magnetic wheel

The organisation of the magnetic storage into tracks and pages has already been described. To recapitulate, there are 256 tracks each consisting of a left hand page and a right hand page. Each page consists of 1280 digits which although actually arranged on a circle may be more conveniently thought of as arranged in a similar manner to the 1280 digits of a tube.

In order to be able to make use of the information stored in the wheel, arrangements are made to enable one to transfer either a complete trackful or a complete page from the magnetic to the electronic store. The process by which this is done is described as a 'reading transfer'. Likewise in order to store information previously held in the electronic store arrangements are made for transfers in the opposite direction, from the electronic store to a track. These are called 'writing transfers'. In addition there are 'check transfers' by means of which the content of a part of the magnetic store is compared with a part of the electronic store.

These transfers cannot be carried out by instructions satisfying the essential conventions of the 'reduced machine', for it is no longer true that in such an operation at most one line of the electronic store is altered. Instead of referring to such a line it is necessary to specify the track concerned and also the tube or tubes. For these reasons the magnetic transfers are controlled on a different principle from the previously described operations. The transfer is described by a special 'magnetic' instruction, which is not an instruction as previously understood. When a magnetic transfer occurs an ordinary instruction is also required. However this instruction merely specifies that a magnetic transfer is to be made and states where the magnetic instruction describing the details of that transfer is to be found. Suppose for example that we encounter the instruction `VE/:`. Referring to the list of meanings of the function symbols we see that `/:` means '**S** as magnetic instruction'. Hence `VE/:` means 'The content of line `VE` is to be obeyed as a magnetic instruction'. Now suppose further that in the line `VE` there is `F/EC`. Now `F/EC` interpreted as a magnetic instruction (as described in detail below) says 'Transfer the left half of track 13 to electronic page 7'. The same would happen if (e.g.) `AA//` or `YE//` were obeyed when `F/EC` was set up on the hand switches (**H**).

We have still to describe the detailed coding of the magnetic transfers. It is thought to be unnecessary to make arrangements to transfer the two halves of a track to *any* two tubes. Instead the tubes are partnered, each odd numbered tube having the next lower (even numbered) tube as its partner. The two halves of a track, if both are transferred, can only be transferred to partnered tubes. This partnering of tubes is somewhat similar in effect to the partnering of pages in tracks, and we may speak of an even numbered tube as the left member of its partnership and the odd one as the right member.

The magnetic instruction is best thought of as consisting of four parts:

a) A part describing a track

b) A part describing a tube

c) A part which with the aid of a) and b) describes one or two magnetic pages and relates them to an equal number of tubes. In the case that there are two the pages or tubes are partnered.

d) A part describing the manner of transfer, i.e. whether reading, writing or checking.

e) Special functions digit

c), d) and e) taken together may be described as the 'function' part of the magnetic instruction. The coding for d) may be simply described

00 Reading
10 Checking (read-like)
01 Writing
11 Checking (write-like)

These four must be explained in greater detail.

00 Reading simply means transferring from the magnetic to the electronic store. The 'sixty-fifth line' of any half-track involved is transferred as well as the main body of sixty-four lines.

10 Checking means comparing the corresponding parts of the electronic and magnetic stores. The check may be said to 'fail' if there is any discrepancy. When it fails $\mathbf{C}' = \mathbf{C} + 1$, but $\mathbf{C}' = \mathbf{C} + 3$ if it succeeds. There is also a neon which changes from bright to dark or vice-versa whenever a check fails. In the case under consideration (read-like checking, 10) the sixty-fifth line is checked as well as the rest. It is in this respect that there is a similarity with reading.

01 Writing means transferring from the electronics to the magnetics without transferring the sixty-fifth lines.

11 This write-like checking does not involve the sixty-fifth lines.

With all magnetic instructions except the successful checks we have $\mathbf{C}' = \mathbf{C} + 1$.

[Two digits are involved in coding for c). One digit specifies the number of pages involved;] 0 means that one page is transferred, and 1 that both pages are transferred. The other digit decides whether the tubes and pages are paired in the 'natural' or 'reversed' order. The natural pairing is to take the left page with the named tube and the right with the partner, and in the reversed pairing the left page goes with the partner and the right with the named tube. When only one page or tube is involved the tube concerned is the named tube. Taking the two digits together the effects are

00 Left page (half track) with named tube

10 Right page with named tube

01 Left page with named tube and right page with its partner

11 Right page with named tube and left with its partner

The magnetic functions can of course also be used for operations which are not magnetic transfers at all. All the functions of this kind have 1 for the 'special functions' digit whereas the functions already described have 0. The special functions are described in the next section under input and output.

Since there are only 16 tubes and 256 tracks one only requires 12 digits for a) and b) and 5 for c), d) and e), i.e. 17 in all. The short line assigned for the magnetic instruction has 20 digits, so that there are 3 spares. It is convenient to place these spares in such a way that the various different parts do not overlap teleprinter characters thus

- 1st and 2nd characters describe track
- 3rd character describes function

- 4th character for the named tube.

Take for instance the magnetic instruction `NEIC`. The combination `NE` determines the track as `NE`, i.e. 44, the character `C` determines the tube as `C`: this means the tube whose first line is `/C`, more often known as tube 7. The function character `I` or 01100 is to be broken into 01 stating that the left page (of track 44) is to be taken with the named tube (i.e. 7) and the right page (of 44) with its partner (i.e. 8), 10 meaning check (read), and 0 stating that it is not a special function. [Note that by the rules previously stated, the partner of tube 7 would be tube 6, not tube 8.]

It is hoped that when the details above have been read through once or twice the diagram Fig. G will then suffice to enable one to remember them.

11 The input and output mechanisms

Information can be fed into the Mark II machine from teleprint tapes and out of it onto other teleprint tapes and onto a printer. The input mechanism provides a further departure from the principles of the 'reduced machine', which was capable of taking only a finite number of states, although this number was quite large, and each state was completely determined by the last one. We now have to regard the state as determined by the last state in combination with the character which is momentarily in the reading head.

The apparatus required consists of a tape reader together with a punch, and a printer. The tape in the input only affects the behaviour of the machine when 'special magnetic function `O`' occurs. The character on the tape is then superimposed ('or' combination) on the five most significant digits of the accumulator, and the tape moves forward so that the next character is in the reading head. The latter process takes a certain time, (about 5 ms) and arrangements are therefore made to prevent another reading process from occurring until the next character is in position. There is no interference with the other instructions, and if another such reading instruction is given prematurely it is not 'lost', but merely held up. This process is described somewhat incompletely below in the equation $\mathbf{A}' = \mathbf{A} \vee 2^{75}\mathbf{T}$, where $\mathbf{T}$ represents the content of the reading head.

In order to facilitate the changing of tapes a switch is provided which inhibits this special function.

The other special magnetic functions are concerned with output. Of these `T` is most essential. Its effect is to punch the character given in the five most significant digits of the accumulator. This character will also be printed by the printer unless it is in the figures-shift position or the printer is switched off. If it is in the figures-shift position the corresponding figure is obtained. These figures are given in the table below

/	E	@	A	:	S	I	U	¼	D	P	M	F
0	1	2	3	4	5	6	7	8	9	¼	-	.

No guarantee can be given concerning what will happen when other characters are printed on figure shift. There may be only a smudge.

A normal teleprinter responds differently to the stunt characters `/`, `@`, `:`, `¼`, `"`, `£` producing respectively no effect, line feed, space, carriage return, figure shift, letter shift. The printer associated with the machine prints these characters, but arrangements are also made to do the stunt operations. These are provided by other special magnetic functions as shown below. No effect is produced on the punch in these cases.

When any one of the special magnetic functions is being obeyed, any other special magnetic function, other than `B`, is held up until the first is complete. Ordinary instructions, and ordinary magnetic instructions and the special function `B` may be obeyed during such a period.

The special function `B`, if obeyed within 10 ms of the special function `T` superimposes the character set up for punching on digits 35 to 39 of the accumulator. Its effect at a later time cannot be guaranteed.

		Time in beats
`T`	$\mathbf{P}' = \{\mathbf{A}_+\}_{75}^{79}$ (and possibly print $\{\mathbf{A}_+\}_{75}^{79}$)	about 62; 38 if printer off
`Z`	Space	about 62; 4 if printer off
`L`	Carriage return	
`W`	Line feed	
`H`	Figure shift	
`Y`	Letter shift	
`O`	$\mathbf{A}' = \mathbf{A} \vee \{2^{75}\mathbf{T}_+\}_0^{79}$, Input tape moves.	about 28.
`B`	$\mathbf{A}' = \mathbf{A} \vee \{2^{75}\mathbf{P}_+\}_0^{79}$	5
`£`	no effect	4

[**P** in these equations evidently represents the punch buffer.] Writing operations [to magnetic storage?] take 258 beats and reading and checking each take 183 beats. It is important to observe that no provisions for carriage return are made other than `L` above, consequently even if there is no intention of ‘page printing’ results it is still necessary to provide carriage returns to prevent the margin stop being reached.

The ‘or’ ($\vee$) connection is particularly suitable for these processes. It is very easy to provide the apparatus to do it, and also, if the operation is applied several times the same effect occurs as if it were applied only once.

Connected with the printer there is also a keyboard with which one can ‘break in’ to print additional remarks.

[Examples]

The magnetic coding has now been completely described and we therefore choose this point to include some examples and exercises.

Examples of magnetic instructions

`SSFG`	means	Track 165L [i.e., 165 left] to be checked with page 13 (i.e., columns `G`, `"`) in a write-like manner i.e. ignoring sixty-fifth lines.
`@/RS`	means	Pages 2, 3 (columns `:`, `S`, `I`, `U`) to be written (not reversed) on track 2, i.e. columns `:`, `S` on 2L; `I`, `U` on 2R.
`COWS` and `LAWN`	both mean	Line feed.

Exercises

1. Interpret `ABCD` as a magnetic instruction.
2. Make up a magnetic instruction to mean 'read track 19L [i.e., 19 (left)] onto page 4'.
3. Make a routine to compare electronic pages 3 and 4 and to give a loop stop with hoot if they differ.

[Library input routines]

The general user of the machine will be more concerned with the properties of the input and output routines which are available than these actual properties of the machine itself. The output routines have not yet been decided on but we shall describe here the arrangements intended for the input routine.

The material on the tapes used for input is divided into 'meaningful sequences and 'intermediate rubbish'. Each meaningful sequence begins with a special character called a 'warning character'. These warning characters are chosen from those which are not very frequent in English and do not include `/` or `£`. The purpose of the former restriction is to enable one to include English words or sentences in the rubbish. The character `/` must not be a warning because one wishes to leave blank spaces at the beginning of the tapes, and to start the tape at some point in this blank space without particular care. If `£` is not a warning character it is possible to remove meaningful sequences by punching additional holes in any warning characters which occur in them. The characters which it is at present intended should be warnings are `J`, `K`, `Z`, `Q`, `"`, `X`. The length of the meaningful sequence and its treatment depends on which warning character is used.

Warning character J

The length of the sequence is 11 characters. No action is taken unless the last character has 0 for its last digit. In this case two lines of columns `½` and `D` are altered, namely those whose address has its first character given as the sixth character of the sequence. The second third fourth and fifth characters form a line which is copied into the appropriate line of column $\frac{1}{4}$ and the seventh eighth ninth and tenth go into column `D`. For example the effect of the sequence `JABCDUFGHIJ` is to put `ABCD` into `U`$\frac{1}{4}$ and `FGHI` into `UD`. This apparently somewhat involved process is designed to make the hand punching of material as simple as possible. It is only necessary to imagine a page of material flanked with a column of `J`'s on either side and then punch each row straight across. The purpose of the second `J` is to facilitate the correction of errors. If it is noticed that a mistake has been made before the second `J` has been punched the line may be nullified by replacing the last `J` by one of the characters `T`, `Z`, ..., `£`.

Warning character K

This permits the writing of material into any successive sequence of lines. The length of the meaningful sequence is determined by the fourth character of the sequence; if the value of this character (+ convention) is n the length of the sequence is $4n + 4$. The last $4n$ characters are to be regarded as divided into consecutive groups of four, which are to be written into consecutive lines. The address of the first of these lines is given by the second and third characters of the sequence. Thus for example the effect of the sequence `KZSAVKTAVST/E:TC` is to put `VKTA` into `ZS`, `VST/` into `LS`, and `E:TC` into `WS`. It need hardly be mentioned that extravagant effects are to be anticipated if this facility is used to write into lines which contain the input routine itself, i.e. into columns `/`, `E`, `@` or `A`. It is permitted however to write into any of the other columns including `C` and `K`.

Warning character Z

One character sequence, i.e. there is only the warning character itself. The instruction contained in `CS` is obeyed. The effect of this under normal circumstances is to 'enter the routine changing sequence' and is described on p. 42.

Warning character Q

The second character (plus convention) gives the length of the sequence, reduced by two. The effect is to punch all the characters of the sequence, including the first two. Its main purpose is to enable titles of input tapes to be recorded in the output.

Warning character "

This is used for the input of numbers in decimal form. It continues from the warning character up to and including the first occurrence of one of the three characters `P`, `M`, `£` after the warning character and the two immediately following it. In other words its length is the shortest consistent with being at least of length four and ending with `P`, `M` or `£`. It may be divided as follows

> Warning character (`"`), Two characters of address, Content characters, End character (`P`, `M` or `£`).

If the end character is £ there is no effect, just as if the whole sequence were intermediate rubbish. If it is P or M, and if the content characters are all chosen from /, E, @, A, :, S, I, U, ¼, D the effect is to alter the (long) line described by the address part. If the content is given by characters $\delta_1, \delta_2, \ldots, \delta_n$ the new value of the line is $\{\pm \sum_{r=1}^{n} 10^{n-r}\{\delta_r\}_+\}_0^{39}$ the plus or minus sign being taken according as the final character is P(lus) or M(inus). If however it should happen that there is a character not included in these ten, it is to be considered that the tape has been incorrectly punched, and the machine stops with a continuous hoot (middle C).

Warning character X

Five character sequence. The last four characters form an instruction which is obeyed shortly after reading the last character of the sequence. Before doing so however the accumulator is filled from the long lines HK and PK, the former filling the least significant half. After the instruction has been obeyed it is emptied back into those lines. They therefore in effect act like an accumulator and are collectively called the 'pseudo accumulator'.

[Input routine timing]

The speed of the input routine is mainly limited by the speed of the input process itself, i.e. special magnetic function T. With decimal input ["binary" in the ms.] the speed is about half this.

Tape handling equipment

It is essential that some further equipment be provided for the handling of tapes independently of the computer. The first essential is a punch controlled from a keyboard. Almost equally important are means for copying tapes and means for converting material on tapes into a typewritten form. These are standard teletype apparatus. The provisions mentioned below are not intended to be final but merely represent our intentions at the time of writing.

It is intended that there shall be two keyboard perforators. These will have 32 keys with the characters /,E,...,£ engraved, punching the corresponding combinations when depressed. At the same time the teleprint signal corresponding to the character in question is transmitted along a teleprint line. There are also keys engraved with [0,]1,2,3,4,5,6,7,8,9,¼,-,. whose effects are respectively duplicates of /,E,@,A,:,S,I,U,¼,D,P,M,F. There will also be a reperforator which accepts signals from a teleprint line and punches the corresponding combinations of holes. There will be two tape readers which accept an input of tape and provide an electrical output of teleprint signals. These units may be coupled in various ways. For instance one may connect both a reader and a keyboard perforator to a reperforator. This enables one to copy a tape with interpositions of new material, and possibly with omissions. In another arrangement the tape from the reperforator may be threaded through the reader, which is electrically connected to the reperforator. Under these circumstances a given sequence of characters can be repeated indefinitely often on a tape.

There will also be a mechanism for the printing out of the contents of tapes. In the letter shift position this will give the standard characters used in this handbook; on figure shift the equivalents

`0`	`1`	`2`	`3`	`4`	`5`	`6`	`7`	`8`	`9`	$\frac{1}{4}$	`-`	`.`
`/`	`E`	`@`	`A`	`:`	`S`	`I`	`U`	$\frac{1}{4}$	`D`	`P`	`M`	`F`

will apply. The changes from figure to letter shift and back are manually controlled.

12 The console

In order to make the account of the machine complete it is necessary to describe the means by which the behaviour of the machine can be controlled manually. In practice these are best learnt in connection with the machine itself, but an attempt to describe them completely here will nevertheless be made.

In principle two controls are sufficient. We have already mentioned the input mechanism: this becomes a method of controlling the machine when a means of changing tapes if provided. If to this is added a key which clears everything (i.e. sets all the electronic stores to zero, including accumulator etc.) we can do all that we require, at any rate if the input programme has been stored in an appropriate part of the magnetic store. [The first instruction executed will then be `////`, which will cause the magnetic instruction set up in **H** to be obeyed; this can then load a bootstrap routine into page 0, as discussed below in detail.] In principle also it is sufficient for the machine to signal to the operator with the output mechanism, or the hooter. We regard these facilities as standard, but there are a number of others which we regard as providing variants of the standard machine.

We have mentioned the dummy stops. For each of the instructions `/L` and `/G` there is a switch deciding whether that instruction shall be a 'waste time' or a stop. When the machine is in the standard condition these are both time wasters. We have also mentioned the hand switches **H**. Besides these there are a number of other switches and keys; ('switches' are stable in either position, whereas 'keys' must be kept pressed if they are to remain in the 'active' position). We list them below together with those which have been mentioned before.

Switch for the dummy stop `/L`
Switch for the dummy stop `/G`
Hand switches **H**

MAN/AUTO switch Manual instruction switches **MI**0, **MI**1, . . . , **MI**19	Selection of instructions
Completion signals on-off switch **CS** Key for single completion signals **KCS** Slow completion signals key	Control of completion signals

Key accumulator clear **KAC** Key **B** clear **KBC** Key control clear **KCC** Key multiplicand clear **KMC** Key everything clear **KEC**	Clearing keys
Digit keys **P**0,...,**P**39 Write-erase key Key line clear [**KLC**]	Manual writing facilities
Input controlling switch Printer on-off switch Switches for connecting input to printer	Input and output control
Write-power switch Other writing suppression methods	Writing suppression

The purpose of the clearing keys, other than **KLC**, are self explanatory. When the MAN-AUTO switch is in the AUTO position the instructions are selected and obeyed in the manner already explained in detail. When this switch is in the MAN position a different arrangement applies. In this case the actual instruction is the combination set up in the manual instruction switches. The presumptive instruction is only of interest when the function symbol is `TO`, `TB` or `TG`. For this purpose it may be taken that the presumptive instruction was *££££*. Thus it is not possible by manual instructions to make any other use of the **B** tube than transfers and additions involving **B**7. [One gathers this applies even to 'B-exceptional' operations such as `TT` for which the presumptive and actual instructions are normally identical.] The effect of obeying an instruction in the MAN condition is determined by the equations on p.E [in the appendices], with one modification, viz. that everywhere where $\mathbf{C}+1$ appears $\mathbf{C}$ should be read instead. This applies to the 'normally understood' equation $\mathbf{C}' = \mathbf{C}+1$, and permits the interpolation of manual instructions amongst automatic ones. It also applies to the relative transfers `/Q`, `/O`, `/M` and results in the transfer, if applicable, being delivered to the line immediately previous to that which would have been reached on the automatic system. The user is recommended to avoid such manual instructions.

The switches and keys for the control of completion signals in effect decide when the instruction, determined by the rules of the AUTO or MAN operation, whichever is applicable, shall be obeyed. If the completion signals switch is in the on position the completion signals occur at the times described on p.E [in the appendices], i.e. the machine works 'full speed ahead'. If a dummy stop which has been switched on is encountered the completion signals cease. Completion signals may also be given with the single completion signals key. These completion signals are not interrupted by stops. They may be given when the completion signals switch is on. It is therefore possible to restart after a stop by giving a single completion signal. The chief application of the single completion signals key is in the verification of the correctness of routines by the comparison of check sheets with the actual behaviour of the machine. It is also in this connection that the dummy stops find their use. They make it possible to hurry through parts which are known to be correct. Another aid to this checking process is the 'slow completion signals key', which provides completion signals at a rate of 50 per second.

The digit keys **P**0-**P**19 provide means for altering individual line pairs of the store. They only operate in the MAN condition with completion signals on. The line of the store which can then be affected is that whose address is contained in the first half of the instruction. It follows from what has already been said that under the conditions in question the instruction on the manual instruction switches will be being repeatedly obeyed. It is advisable therefore to choose some innocuous value for the function part of this instruction. (Any value which does not mention $\mathbf{S}'$ in its equations and does not involve a magnetic transfer is satisfactory. If the normal condition of the switches is considered to be `////` probably `/T` is the most convenient). Under these circumstances the effect of operating key $\mathbf{P}n$ is equivalent to an instruction with the equation $\mathbf{S}' = \mathbf{S} \vee \{2^n\}_0^{19}$. For example if we have set up `GF/T` on the switches and depress **P**1,**P**6, and the short line `GF` contains `MICE`, we shall afterwards find that `GF` contains `VICE`. At the same time the content of control may have been altered by the obeying of `GF/T`. It is assumed for this purpose that the write-erase key is not depressed, so that it is in the write position. If this key is maintained depressed during the operation of the digit key $\mathbf{P}n$ the effect is given by the equation $\mathbf{S}' = \mathbf{S} \wedge \sim \{2^n\}_0^{19}$ i.e. instead of altering digits 0 to 1 the reverse applies. Operation of **KLC** clears the line, whatever the position of the write-erase key. It will be observed that the same effect is obtained by setting the function symbol momentarily e.g. to `TA`.

Exercise. Assuming the digit keys **P**0-**P**19 to be unserviceable devise a procedure to replace them by the use of **H** and other controls.

Note. Since **H** consists of switches, whilst **P**0-**P**19 are press-button keys the latter are somewhat more convenient to use.

The changing of the position of a switch should preferably be done at a time when its position is irrelevant to the working of the machine. All switches and associated equipment are designed with the intention of avoiding transient effects, i.e. if it is irrelevant which stationary position the switch is in, it is equally irrelevant whether it is being flicked back and forth. Also if it should happen that a change is made during a time when its position is relevant, the effect produced at each individual instruction during that period will be either what would be expected in the on position or what would be expected in the off position, and not any third alternative. It may be taken that the period between being definitely on and definitely off does not exceed 30 ms. No statement is made concerning what will happen when two or more switches are simultaneously neither fully on nor fully off. Likewise the operation of a key is preferably done at a time during which the intermediate state of that part of the machine immediately affected by the operation is irrelevant to the behaviour of the rest of the machine. For instance the effect of clearing the accumulator with completion signals on is not always very easily predicted. It is best to do such operations with completion signals off. If however it is necessary to do them with completion signals on the following assumptions apply to keys other than **KEC**. In every individual digit period it may be assumed that the key is either definitely operated or definitely not operated. The transition period between operation and non-operation may be assumed not to exceed 30 ms. This assumption cannot of course be applied without more knowledge of the detailed circuits and timing than has been given here. It may be said however that it is adequate to justify the use of **KAC** or **KBC** for signalling to the machine with completion signals on. This is done in a few routines. The key **KEC** is in a somewhat different category since it effectively operates a number of separate keys. The use of **KEC** with completion signals on is considered on pp. 40-41.

The effect of undesired magnetic writing transfers can be disastrous. In many problems no such transfers need be made. During the solutions of these problems it is usual to suppress all writing transfers by switching off the 'write power'. In most other problems there are certain tracks on which one does not wish to write, e.g. the tracks on which the routines are written. In these cases one wishes to be able to suppress the writing on these. For this purpose one may suppress the writing on a block of sixteen tracks by the removal of a valve [i.e., vacuum tube] from the writing circuits. A 'block' consists of a set of tracks whose numbers are of form $16n + m$ where $0 \le m < 16$ and n remains fixed throughout the block. One may also suppress writing on a single track by removal of a valve. The removal of valves is normally not admissible, but is licensed in these cases.

The facilities additional to the output and the hooter by which the state of the machine may be observed comprise a number of monitor tubes, which make visible the contents of the various electronic stores and some neons. The contents of the accumulator, B-tube, control and multiplicand are displayed on different tubes and the content of the store is displayed on two further tubes. Which pages are displayed on these tubes is determined by switches. Any pair can be chosen. The neons comprise

- **Q** digit
- Sign of **D**
- Stop indicator
- Check indicator
- [Test indicator]

The **Q** neon is bright if **Q** = 1, the **D** neon if **D** is negative. The 'stop' neon is bright if the last instruction obeyed was a dummy stop, regardless of whether it was switched on or not. The test indicator shows bright if the last instruction was one of the four 'conditional transfer' or test instructions, viz. `/T`, `/H`, `/O`, or `/M` and moreover one for which the test 'came out negative' i.e. in which **Q** or the most significant digit of **A** (whichever is relevant, i.e. is explicitly mentioned in the equations) is 1. The check neon shows the parity of the number of failed checks as described on p. 28.

A number of special points must be noted. The quantity given on the multiplicand line, interpreted according to the plus convention, is the modulus of the content of the multiplicand. The sign of the content of the multiplicand must be found from the neon. On the control tube one finds not only the quantity **C**, (which we have also called the instruction number I.N., and which, after adding 1, describes the line in which the next instruction to be obeyed, is to be found), but also the last instruction obeyed. This is the true instruction, not the 'presumptive instruction', i.e. the content of the B-tube has been taken into account. This does not of course apply to the case when the MAN-AUTO switch is set at MAN. In this case one may think of the control tube as showing the instruction which would have been obeyed in the AUTO condition. There is no means of monitoring the presumptive instruction, other than by looking at the appropriate line of the store *before* obeying the instruction.

The monitor tubes can of course be used to observe the general progress of a computation, but this usually proceeds too fast for observation of detail. They are most useful for observing the machine in a stationary condition. This occurs with

1. Completion signals switched off
2. Dummy stops
3. Loop stops

With completion signals switched off one can observe successive states of the machine by operating the single completion signal key. If there is any doubt as to the correctness of the programme it is useful to compare this sequence of states with those given on the 'check sheets'. Provided that the machine can be spared for the purpose this is the quickest way of finding errors in the programme. In making check sheets it is often convenient to indicate repetitions of a process by dots, rather than write out all the detail. When these dots represent a very large number of steps of the machine it may be inconvenient to go through these steps with the single step facility. It is then best to put on the automatic completion signals, but it is difficult to stop them just at the right moment. If however dummy stops are included in the programme at appropriate points they can be used to stop the machine where required. A special form of this is mentioned below (p. 44).

13 Starting the machine

Our explanations and examples have up to now assumed the machine somehow to have reached certain particular states, and the reader has been expected to restrain temporarily his curiosity as to how these states were reached. In this section we attempt to satisfy this curiosity. This does not involve making any new statements about the construction of the machine; it is an application of the properties of the machine already (and almost completely) described.

The essential process to describe is that of ‘manually writing given information into a track’. This is done in two parts, of which the first is the writing of the information into a page or two partnered pages of the electronic store and the second consists in transferring this information to the track in question. The writing into the electronic store is done a line at a time. The procedure is as follows

- CS off
- Set function part of **MI** e.g. to `T£` or `T/`.
- MAN-AUTO switch to MAN
- Operate **KEC**
- CS on
- Set store line part of **MI** to the address of the first line to be written in, e.g. if the content required is `E//T` operate **P**0, **P**19.
- Set the store line part of **MI** to the address of the second line-pair to be written in, e.g. to `E/`.
- Operate the keys for the content of this second line.
- Repeat with the remaining lines required.

This procedure can be varied when errors are made. The most satisfactory arrangement in the opinion of the writer is to operate **KLC** whenever an error is made and then rewrite the line. Alternatively the ‘erase’ key may be used.

To copy from the electronic store to the track the procedure is

- C.S. off
- **MI** to `////`
- **H** to a magnetic instruction describing the transfer required.
- Operate **KCS**

It will be appreciated that there are a number of variations of procedure possible, and that the procedure above makes no allowance for checking, although checking should in fact be done. An important variation is the possibility of using **H** in place of **P**0,. . . **P**19.

The procedure outlined above should be non-recurrent. It is intended that as soon as the input programme, or even the bare bones of the input programme, have been put into the magnetic store, this method will no longer be used. There will be an interim period during which a variety of different methods may be used, because the programmes for the final arrangement have not yet been put in, but these are of no interest. The final arrangements will also not be described, for they have not yet been decided. [But see the discussion of, e.g., the WRITE routine on p. 65.]

It must not of course be imagined that the 'initial conditions' of the various routines which we have given as examples would normally be achieved by such a 'writing-in' process. They will normally be achieved during the action of some other routine which uses the given routine as subroutine. Moreover when making up these (sub-) routines one does not normally have any particular means in mind by which the initial conditions are to be achieved, apart from the general conventions governing the change of routine. They are simply made up in the (normally justified) belief that these initial conditions will arise in some useful connection.

In connection with starting procedures the use of **KEC** should also be mentioned. We will suppose that the machine is out of control, i.e. that we do not know what it is doing, either because the machine itself has made some unknown error, or because the operator has forgotten what he has done, or has done something whose consequences he does not choose to consider. We assume the switches to be in the positions for normal operation, i.e.

- C.S. on
- Dummy stops off
- Auto condition

We will also assume **H** set to `////`. Now consider the effect of operating **KEC**. As mentioned in the description of the console this clears the store, and the special stores **B** and **C**. It has other effects which do not concern us so much here. As the key is allowed to return to its normal condition the connections causing the clearing effect will be broken in some unknown order. There may possibly be bounce on the contacts causing the clearing effect to return temporarily. We shall show that this complicated behaviour of the key need not be known in any detail, and that the machine as a whole 'comes under control' very shortly after the key has returned to its normal position. We make only the following assumptions.

i) The period of bounce does not exceed 156 ms.

ii) During the period of bounce the effect of altering a line of store or control or B-tube is the alteration as described for normal operation combined with the possible replacement of some digits 1 by 0.

   For this purpose the processes

- Adding 1 to control (I.N.)
- Selecting the instruction from the store and copying onto control tube (P.I.)
- Obeying the instruction

must be treated as separate.

iii) Digits 10 to 19 of lines `//` to `U/` of track 0L [i.e., 0 (left)] contain `//`.

It will be seen that if at any prepulse-time $\mathbf{C}_+ \leq 6$ and iii) still holds, then $\mathbf{C}_+ \leq \mathbf{C} + 1$ and iii) holds at the next prepulse. It follows that $\mathbf{C}_+ \geq 6$ cannot hold until the fifth prepulse after the beginning of the bounce period. It will be seen that all the intervening instructions are magnetic transfers occupying more than 30 ms and that therefore when $\mathbf{C}_+ = 6$ for the first time, the bounce period is over, and iii) still holds. When the next instruction has been obeyed the machine will be in the same condition as if it had proceeded in the normal manner from the condition with control and store clear. There is no externally distinguishable behaviour of the machine in the two cases, for in neither case is there any hooting, printing or punching before the point at which the common behaviour starts.

It will be seen that the **KEC** method of starting is valid also if some other magnetic transfer is set up on **H**, provided that the routine brought down to [electronic] p. 0 satisfies iii). Such routines may be termed 'self-starter' routines. They include routines brought down to p. 1 and starting at `/@`, though there is a delay of about two seconds before the routine is entered in this case.

[At least on the basis of Turing's descriptions of console functions, it would appear that if **KEC** were only operated with completion signals *off*, then we could dispense with the redundant initial magnetic transfers at the start of 'self-starter' routines, and for that matter a lot of this argumentation . . .]

14 Conventions

The machine has a very great flexibility. Although this has obvious advantages, it has also certain disadvantages which can become serious unless precautions are taken. It is for instance possible to alter the whole content of the electronic and magnetic stores merely by putting an appropriate tape on the input. Although we may be often glad of this fact it increases the possible damage which can be caused by mistakes. The remedy for this kind of difficulty lies in the introduction of *conventions*. These are in effect decisions to restrict the freedom or flexibility of the machine in various ways. It is hoped that the loss of flexibility will be fully compensated for by the advantages of the resulting reduction of the uncertainty of the state of the machine. The conventions are mostly not to be regarded as absolute commands or prohibitions, but rather as normal procedure, any deviation from which must be noted in the descriptions of the routines in which they occur.

Since these conventions do not form any part of the physical machine the user has the alternative of ignoring them altogether. Likewise he can if he wishes ignore altogether the whole of what is said about actual programmes and (at his peril) make his own from scratch, if he considers it advisable. The conventions should not be regarded as pure tyranny, but to know that they have been obeyed in the programmes one is using is a great comfort. Moreover they enable one to reduce considerably the lengths of official accounts of routines, since they allow a great deal to be taken for granted.

The permanent information PERM. The routine changing sequence

There are certain expressions which it is desirable to keep constantly available in the electronic store. The amount of material which should be so kept is a matter of opinion. At present it is proposed to keep only the powers of two and the 'routine changing sequence'. The details of this material are shown on the accompanying diagram. The forty powers of two are obtained without using more than forty-one short lines, although each power occupies a long line. However the addresses of those are not in linear sequence. This means that if we wish to use some particular power in an instruction we can do so, but if for instance we want to find $2^{[/\frac{1}{4}]+}$, a somewhat lengthy process involving a test instruction is required. Sometimes this has to be done, e.g. in the routines for the logarithm and the reciprocal. In these cases the forty one lines of PERM are combined with a further forty lines which are put into the electronic store simultaneously with the routines which require them.

[The listing of PERM is actually at the end of the manuscript; it is as follows:

:		S	C		K
/ / / /	/	/ / / /		/	
E / / /	E	/ / / @		E	
/ / / /	@	/ / / /		@	
@ / / /	A	/ / / :		A	
/ / / /	:	/ / / /		:	
: / / /	S	/ / / ¼		S	
/ / / /	I	/ / / /		I	
¼ / / /	U	/ / / T		U	£ £ / /
/ / / /	¼	/ / / /		¼	H : / C
T / / /	D	£ £ £ £		D	V S / F
/ / / /	R	£ £ £ £		R	L : / J
/ E / /	J	/ / / /		J	M K / A
/ / / /	N	F S / V		N	X K / :
/ @ / /	F	K S / P		F	£ S T /
/ / / /	C	V K T A		C	U K T R
/ : / /	K	J S / L		K	M K / S
/ / / /	T	V S T F		T	M K Q O
/ ¼ / /	Z	C K / H		Z	/ / Q T
/ / / /	L	£ S / :		L	£ S Q B
/ T / /	W	/ E T /		W	F S / P
/ / / /	H	/ A T N		H	Psuedo Accumulator
/ / E /	Y	M K / S		Y	
/ / / /	P	Y : / C		P	
/ / @ /	Q	M K / N		Q	
/ / / /	O	V S T J		O	
/ / : /	B	R S T R		B	
/ / / /	G	Y : T N		G	Special Working Space
/ / ¼ /	"	K S / H		"	
/ / / /	M	V K T /		M	
/ / T /	X	V S / P		X	
/ / / /	V	Cue		V	
/ / / E	£			£	

The routine changing sequence is entered by the instruction `NS/P`. The "special working space" is scratch space which any routine may overwrite at any time.]

The purpose of the routine changing sequence is to enable routines to be changed without having a lot of preparatory bother in the routine which is being left, but fairly quickly and certainly. A certain amount of preparatory bother is unavoidable. When leaving any routine it is necessary to specify the new routine which is being entered. We do it with a long line which is called the 'cue' for the new routine. When leaving a master routine for a subroutine it is also necessary to specify what is to be done when the operations of the subroutine are over. This is done with another forty digit line, called the 'link'. When leaving the subroutine the 'link' will become a cue, and will normally lead back to the master routine.

The preparation to leave a routine will always include putting the cue into `VS` where the routine changing sequence will find it and deal with it. This setting of the cue may be done just before leaving the routine, though it can be done at any previous time, provided no subroutine intervenes. With routines which have no subroutines the cue is often planted immediately after entering the routine itself. If a link is required it should be put into the least significant half of the accumulator immediately before leaving the routine. On entering the subroutine the link may be planted in `VS` at once, or if there are lesser subroutines, may be temporarily stored elsewhere.

The cue itself is required to specify the new routine which is being entered. Two different systems are used according to whether the routine in question is or is not 'of fixed abode', i.e., whether it is always kept in the same track or not. The cue itself determines which type it belongs to, and the routine changing sequence recognises the type and acts accordingly. The two types are called 'true cues' and 'false cues'. True cues end with 0 and are used with routines of fixed abode. False cues end with 1 and are used with routines of no fixed abode. The cues will be described here from the point of view of their treatment by the routine changing sequence. The same information is given again on p. 62 from a rather different point of view. [That discussion in fact describes their format, which includes a magnetic instruction to load the routine off the magnetic wheel (or for false cues, a pointer to the magnetic page containing one, and a location on that page), a crude checksum, and the address of the first instruction once the new routine has been loaded into the electronic store; the reader may wish to turn ahead for details before coming back here to see the details of how a cue is followed.]

The first step after entering the routine changing sequence is to deposit the link temporarily in `VK`, leaving the accumulator clear and free for other purposes. This is the instruction `VKTA`. The next instruction is `JS/L` [orig. `FS/L`] which is a dummy stop. This will be encountered whenever routines are changed, and enables one, when testing out routines by hand, to speed through subroutines known to be sound. The next two instructions `VSTF` and `CK/H` discriminate between the true and false cues. Let us take first the case of a true cue. The next instruction is `£S/:`, and obeys the magnetic instruction part of the cue. The next five instructions form the quantity $\{1025([\texttt{/E}]-[\texttt{/A}])\}_0^{19}$ whose value depends on the preceding magnetic transfer and, if the wrong transfer is made, is likely to have the wrong value. The correct value is contained in digits 10-19 of the cue. The next four instructions are concerned with verifying whether the value obtained agrees with the value given in the cue. If the value is wrong a middle C hoot occurs. If the value is right the routine is entered by the instruction `VS/P`, using the control transfer number part of the cue.

In the case of a false cue a new magnetic instruction is constructed and obeyed, resulting in [the] left half of the track mentioned in digits 30-38 of the cue being transferred to [electronic] p.0. The line of this track named in digits 20-29 is then transferred to the short line `£S`. The routine changing sequence is then reentered. But now there is a new line in `£S`, not ending in 1, so that the content of `VS` is now treated as a true cue. The part of the routine changing sequence which deals with false cues is in [electronic] p.7.

Connected with PERM we have the conventions

i) That it is assumed, unless otherwise specified, that PERM is present both at the beginning and at the end of a routine, i.e. that the contents of lines `/:` to `XS` are as shown in Fig. F.

ii) That the retreat from a routine is fully provided for by leaving the cue for the retreat as a link in the accumulator (**L**).

iii) That to describe the method of entering a routine it is sufficient to specify the cue, in the manner described below (p. 62).

iv) The lines `/E` and `/A` should not be left empty in a routine, and should not be used as working space in a routine.

Restricted use of electronic stores. Normal duties of pages.

It must now be explained that it is very doubtful whether the sixteen pages of electronic store will ever be available. In any case it is advisable not to use more of the store than is really necessary, on account of the difficulties of servicing. However it is futile to practice excessive space economy with a routine which is normally to be used in connection with others which practice less economy. It is necessary therefore to have fairly definite understandings about how much may be used. The following may be assumed

v) Five pages of electronic store will always be available. In other words it is not recommended that any programming be done for a machine with less store.

vi) No 'library' programming should be done on the assumption that more than eight pages will ever be available.

The decision as to which of the sixteen tubes should be made functional is at our disposal. It seems preferable that these should be chosen amongst the first eight, although it might be difficult to give any very convincing reasons for doing so. Of these any five may be taken, but it is desirable that a particular five be chosen and not altered. This is essential if library routines are used. The selection which has been chosen is 0,1,2,4,7, i.e. columns `/`,`E`,`@`,`A`,`:`,`S`,½,`D`,`C`,`K`. This selection may appear somewhat arbitrary. It may appear less so after it has been explained how these pages will normally be used.

Pages 0 and 1 (columns `/`,`E`,`@`,`A`) are customarily used to contain routines, i.e. instructions together with auxiliary fixed numbers etc. The magnetic transfers applied in the routine changing sequence will normally be to one or both of these pages.

Page 2 (columns `:`,`S`) contains most of PERM

Page 4 (columns $\frac{1}{4}$,D) is used as systematic working space, i.e. whenever it is required to store tables or other material systematically arranged it is recommended that this page be used.

Page 7 (columns C,K) is used as unsystematic working space, i.e. its content is largely unrelated long lines. It also includes a part of PERM viz. the part of the routine changing sequence concerned with false cues.

The uses of the other three pages if ever available is anticipated as follows

Page 3 (columns I,U) may be used either as a further part of PERM or as a more unsystematic or systematic working space or any combination of these. The lines used for the former applications should preferably be the earlier ones so that the systematic working space may consist of consecutive lines.

Page 5 (columns R,J) will be available as additional systematic working space.

Page 6 (columns N,F) will be available for systematic or unsystematic working space, the latter preferably being restricted to the later lines.

These decisions and suggestions have been influenced by the following considerations.

(a) The choice of the five pages must be convenient when there are only five available, but must also be convenient when there are six, seven or eight.

(b) The routines must be restricted to relatively few pages so as not to interfere with other forms of storage.

(c) Similarly used pages should preferably be partnered. This applies particularly to systematic working space and to the space used for routines.

(d) Systematic working space should if possible consist of consecutive lines.

(e) The powers of two must be consecutive with the space used for routines.

(f) The first few instructions after operating **KEC** are taken from column /.

These suggestions are supported by the following conventions

vii) That in all library routines the instructions must be kept in columns /,E,@,A.

viii) All alterations of lines on pages 3-7 must be mentioned in the account of the properties of the routine, with the exception of long lines GK,MK,VK.

It is intended that lines GK,MK,VK be used as special short form working space i.e. to contain quantities which are no longer of interest once the routine is finished. Lines MK and VK are in any case used in the routine changing sequence. It will be seen that the presence of these three lines and the part of PERM on page 7 makes this page useless for systematic working space. If further short term working space is required one may use those lines of pages 0,1 which contain instructions which will not be obeyed again before they are wiped out by a magnetic transfer. It should hardly ever be necessary to use any other part of the unsystematic working space as special working space [i.e., scratch space].

The normal uses of the pages are set out briefly again on Fig. G.

B-tube conventions

The conventions concerning the use of the B-tube are

ix) At the end of a routine **B**0=//// unless otherwise stated. Indeed any alteration of **B**0 requires special mention.

x) Uses of the B tube resulting in the function number part of an actual instruction differing from that of the corresponding presumptive instruction, need special attention.

xi) When a choice of B lines is available the higher number is to be preferred.

xii) Alterations of B lines other than **B**6, **B**7 must be mentioned.

It may be remarked that **B**7 is used in the routine changing sequence.

Conventions regarding the use of magnetic storage

There are a number of different ways of using magnetic storage which are worth distinguishing.

a) Tracks containing fixed information, namely a number of different routines of rather general application and PERM.

b) Tracks used to contain other routines, which are changed from time to time, but not as part of a computation.

c) Tracks used for working space.

d) Tracks used for systematic working space.

e) Tracks used for special working space.

It is not yet necessary to make many decisions yet about which tracks should be used for which purposes. We may perhaps decide

xiii) Tracks 0-15 are for permanent routines. The writing will normally be suppressed for this block of tracks.

xiv) Tracks 16-31 are for less permanent routines as b) above. Writing is suppressed during computations.

xv) Tracks 32 to 63 are working space. Track 32 is special working space, i.e. its content may be altered in a subroutine without mention. Unsystematic working space should preferably be in the earlier tracks of these two blocks.

The formal mode of operation

There are a number of modes or styles in which the machine may be used, and each mode has its conventions restricting the operations considered admissible. The engineers for instance will consider the removal of a valve or the connection of two points temporarily with crocodile clips to be admissible, but would frown on certain uses of a hatchet. The removal of valves and all alterations of connections are certainly not permitted to the programmers and other users, and they have additional taboos of their own. There are in fact a number of modes of operation which might be distinguished, but only the *formal mode* will be mentioned here. This mode has rather stringent and definite conventions. The advantage of working in the formal mode is that the output recorded by the printer gives a complete description of what was done in any computation. A scrutiny of this record, together with certain other documents should tell one all that one wishes to know. In particular this record shows all the arbitrary choices made by the man in control of the machine, so that there is no question of trying to remember what was done at certain critical points.

The conventions involved in the formal mode are of two kinds, restrictions on the programming and punching of tapes, and restrictions on the actions of the man in control of the machine.

Restrictions on the actions of the man in control

(a) The only operations permitted are the changing of input tapes and of the hand switches **H**, operation of **KEC** and **KAC**, and tearing off of the punched and printed outputs. These operations may only be carried out as permitted in (b) to (g).

(b) These actions may only be performed when certain printed signals are given by the machine. The details of this are described in the official account of the routine ACTION. For instance when a tape is to be changed `+ + CHANGE::........--` is printed and a hoot occurs. The full stops represent a 'descriptive word' of eight characters giving the controller some indication of how to choose the next tape to put in.

(c) Only specially marked 'titled tapes' may be used.

The programmer issues instructions both to the machine and to the controller. The latter may be as complicated as the programmer considers advisable, but must of course be consistent with (a), (b) and (c). The complete programme is restricted by the further conditions:-

(d) The actions which the programmer permits the controller at any stage must be determinable from the previous printed output.

This ensures that the programmer, looking at the printed output afterwards, will be able to verify that his instructions were obeyed, always assuming that the machine made no mistake and that (a),(b) and (c) were not disregarded. The routine ACTION is constructed so as to make a printed record of every action of the controller, assuming he has obeyed (a),(b) and (c).

(e) Instructions involving **H** are only used as follows. Instruction `//` is only used in the first few lines of the self-starter routine INITIAL, kept in track 0L [i.e., track 0 (left)]. Instruction `/:` is only used in the routine ACTION already referred to.

(f) Special magnetic instructions, concerning input or output, are only obeyed in specially constructed input and output routines, more specifically in INPUT, OUTPUTA, ACTION and WRITE.

(g) The 'titled tapes' provided for use under (c) must all begin with a titling sequence, i.e. a meaningful sequence beginning with `Q`, as described under the input routine. Precautions must be taken that no two titled tapes can have the same title i.e. the same initial meaningful sequence.

(h) Precautions must be taken that when a hoot enjoins the tearing off of a tape from the output, this tape shall, if titled, obey the restrictions mentioned under (g).

(i) If output sequences can be of indefinite length there is a danger that it may not be possible to determine the meaning of the output unambiguously. It must however be arranged that this *can* be done.

In general the programmer cannot be held responsible for the effects that may arise when his instructions are disregarded, any more than for those which are the result of errors in the machine. He will of course be well advised to have checks for both. The efficiency of his precautions will mainly affect his own work. To avoid it affecting the work of others it is advisable that title begin e.g. with the initials of the programmer, or with something identifying the job.

If the output were only on punched tape it would be quite difficult to ensure that (i) was obeyed, since any combination of characters might appear as part of the genuine output. For instance one might wish to have the machine calculate π to a certain number of places, the number possibly being determined by the machine, and then to do something else. One might have the machine indicate the end of the digits of π by writing `DIGITSOFPIENDHERE`. But there is no reason why this particular combination of digits should not be part of the binary expansion of π, indeed it almost certainly does appear somewhere, so that the appearance of this sequence of digits could not be taken as conclusive evidence that the machine had completed as much of the computation of π as it was proposing to do. Of course it is vastly more probable that this is the case than that these characters were obtained as part of the binary form of π. The objections to making such assumptions are mainly aesthetic. It is satisfactory to be in a position to regard the behaviour of the machine as constituting the proof of some mathematical statement. It is less repugnant to those with pure mathematical training to be obliged to assume that the machine has made no mistake, than to have to admit that even if no mistake was made the required result does not quite follow. I personally, for instance, would prefer to have to admit a probability 10^{-20} that the machine had made a mistake, than to have to assume that a thousand consecutive zeros occur in the first million digits of π.

The condition (i) can fortunately easily be met if the printer is available. It is for instance easy to take precautions that `--` can never occur except when printed in ACTION. This justifies one in assuming that the warning signals given in ACTION actually are produced by that routine and are not some counterfeits produced in another way.

Replacability conventions

It is essential for the possibility of programming at all that the properties of the machine should hardly ever be changed. There are however certain features of the machine which can reasonably be described as 'disadvantageous'. It is desirable to leave open the possibility that these features might at some time be removed. This suggests the convention

xvi) No disadvantageous feature of the machine should be used in a library routine.

This requires further definition. It is understood in connection with each such feature that it is known what modification improving the machine is contemplated. The routines must work whether such a modification has been introduced or not. The features at present recognised as disadvantageous are mentioned below.

a) Certain functions are not considered of particular value and rated as 'foul'. These are `/B`, `/"`, `/X`, `/£`, `TE`, `TS`, `TU`, `TH`, `TY`, `TP`, `TQ`, `TM`, `TX`, `TV`. The modifications envisaged are the changing of these to other operations, so that the convention amounts to the avoidance of their use. [Of these, the only ones with documented effects in the appendices are `TE`, `TS`, and `TU` which perform a logical 'or' from the accumulator into storage (`TE` leaving the result in the accumulator as well).]

b) The exceptional nature of the line pairs at the end of a page (see p. 5) is disadvantageous. The modification would consist in bringing these into line with the other line-pairs. The use of such line-pairs is to be avoided.

c) The sixty-fifth line facility is reckoned as disadvantageous for a slightly different reason, viz. that it is an inconvenience in the engineering, and it might at some time be abolished for engineering reasons. It certainly should not be used for any other purpose than checking.

d) Provided that input and output are always done through the official programmes the possibility of introducing improved input and output mechanisms remains open.

We must also recognize the possibility of altering routines, whilst leaving their essential properties, except for speed, unaltered. It is desirable that such changes should not affect the validity of the programmers which use them as subroutines. This suggests

xvii) No programme should depend on properties of its subroutines which are not mentioned in the official accounts of these subroutines. They should also not depend on the time of the subroutine.

The time restriction is not one which one is tempted to violate, and indeed one which could only be violated by use of the clock. Statements about the time of the routine itself will of course depend on the time of the subroutine, and are excepted, but it is desirable to state the former as a function of the latter.

15 Programming Principles

Programming is a skill best acquired by practice and example rather than from books. The remarks given here are therefore quite inadequate. [The word "inadequate" is very faint in the ms., and may have been deliberately whited out.]

If it is desired to give a definition of programming, one might say that it is an activity by which a digital computer is made to do a man's will, by expressing this will suitably on punched tapes, or whatever other input medium is accepted by the machine. This is normally achieved by working up from relatively simple requirements to more complex ones. Thus for instance if it is desired to do a Fourier analysis on the machine it would be as well to arrange first that one can calculate cosines on it. For each process that one wishes the machine to be able to carry out one constructs a 'routine' for the process. This consists mainly of a set of instructions, which are obeyed by the machine in the process. It is not usual however to reckon all the instructions obeyed in the process as part of the routine. Many of them will belong to other routines, previously constructed and carrying out simpler processes. Thus for instance the Fourier analysis process would involve obeying instructions in the routine for forming cosines as well as ones in the analysis routine proper. In a case like this the cosines routine is described as a 'subroutine' of the analysis routine. The subroutines of any routine may themselves have subroutines. This is like the case of the bigger and lesser fleas. I am not sure of the exact meaning the poet attached to the phrase 'and so ad infinitum', but am inclined to think that he meant there was no limit that one could assign to the length of a parasitic chain of fleas, rather than that he believed in infinitely long chains. This certainly is the case with subroutines. One always eventually comes down to a routine without subroutines.

[The previous paragraph would have to be amended somewhat in the light of modern practice. Currently, digital Fourier analysis generally uses FFT algorithms which do not involve any explicit evaluations of cosines (or other trigonometric functions); evidently Turing was not familiar with these approaches, though the basic idea has been traced back to the 19th century. Likewise, his final statement that "One always eventually comes down to a routine without subroutines" needs some modification in the case of recursive routines, or sets of mutually recursive routines — one always eventually comes down to a routine which does not *invoke* any subroutines (at least if the computation ever terminates!) but that is a somewhat different thing.]

What is normally required of a routine is that a certain function of the state of the machine shall be calculated and stored in a given place, the majority of the content of the store being unaffected by the process, and the routine not being dependent on this part having any particular content. It is usual also for the other part of the store to be divided into a part which is altered in the process but not greatly restricted as to its original content, and a part which is unaltered in its content throughout, and such that the correct working of the routine depends on this part having a particular content. The former can be described as 'the working space for the routine' and the latter as 'the space occupied by the routine'.

Applying this to the Mark II machine, the routines usually 'occupy' various tracks of the magnetic store. The working space includes all the electronic store, with the exception of PERM, which can equally well be reckoned as working space which is never really altered, or as a part common to all routines. The first two pages of the electronic store are also somewhat exceptional, if normal conventions are used, as they are only altered when one is copying new routines from the magnetic store onto them.

These definitions do not really help the beginner. Something more specific is needed. I describe below the principal steps which I use in programming, in the hope they will be of some small assistance.

(i) Make a plan

This rather baffling piece of advice is often offered in identical words to the beginner in chess. Likewise the writer of a short story is advised to 'think of a plot' or an inventor to 'have an idea'. These things are not the kind that we try to make rules about. In this case however some assistance can be given, by describing the decisions that go to make up the plan.

a) If it is a genuine numerical computation that is involved (rather than e.g. the solution of a puzzle) one must decide what mathematical formulae are to be used. For example if one were calculating the Bessel function $J_0(x)$ one would have, amongst others, the alternatives of using the power series in x, various other power series with other origins, interpolation from a table, various definite integrals, integration of the differential equation by small arcs, and asymptotic formulae. It may be necessary to give some small consideration to a number of the alternative methods.

b) Some idea should be formed as to the supply and demand of the economic factors involved. A balance must always be struck between the following incompatible desires

- To carry the process through as fast as possible
- To use as little storage space as possible
- To finish the programming as quickly as possible
- To achieve the maximum possible accuracy

We may express this by saying that machine time, storage space, programmers' time and inaccuracy of results all cost something. The plan should take this into account to some extent, though a true optimum cannot be achieved except by chance, since programmers' time is involved, so that a determination of the optimum would defeat its own ends. The 'state of the market' for these economic factors will vary greatly from problem to problem. For instance there will be an enormous proportion of problems (40% perhaps) where there is no question of using the whole storage capacity of the machine, so that space is almost free. With other types of problem one could easily use ten million digits of storage and still not be satisfied. The space shortage applies mainly to working space rather than to the space occupied by the routines. Since these usually have to be written down by someone this in itself has a limiting effect. [The statement that instructions are not likely to cause a space shortage is also somewhat at odds with current practice.] Speed will usually be a factor worth consideration, though there are many 'fiddling' jobs where it is almost irrelevant. For instance the calculation of tabular values for functions which are to be stored in the machine and later used for interpolation, would usually be in this class. Programmers' time will usually be the main factor in special jobs, but is relatively unimportant in fundamental routines which are used in most jobs. Accuracy may compete with machine time e.g. over such questions as the number of terms to be taken in a series, and with space over the question as to whether 20 or 40 digits of a number should be stored.

c) The available storage space must be apportioned to various duties. This will apply both to magnetic and electronic storage. The magnetic storage will probably be mainly either working space or unused. It should be possible to estimate the space occupied by instructions to within say two tracks, for a large part will probably be previously constructed programmes, occupying a known number of tracks. The quantities to be held in the working space should if possible be arranged in packets which it is convenient to use all at once, and which can be packed into a track of a half-track or quarter-track. For instance when multiplying matrices it might be convenient to partition the matrices into four rowed or eight rowed square matrices and keep each either in a track or a quarter-track. The apportionment of the electronic store is partly ruled by the conventions we have introduced, but there is still a good deal of freedom, e.g. if eight [electronic] pages are available then pages 4, 5, 6 can be used for systematic working space and may be used for various different purposes that require systematic working space.

The beginner will do well to ask for advice concerning plans. Bad plans lead to programmes being thrown away, wasting valuable programmers' time.

d) If questions of time are at all critical the 'plan' should include a little detailed programming, i.e. the writing down of a few instructions. It should be fairly evident which operations are likely to consume most of the time, and often these will consist of a small number of instructions repeated again and again. In these cases the few instructions in question should be written down so as to give an estimate of the time, and help decide whether the plans as a whole is satisfactory. Very often the 'omission of counting method' [i.e., loop unrolling; see p. 58] should be applied

e) If one cannot think of any way, good or bad, for doing a job, it is a good thing to try and think how one would do it oneself with pencil and paper. If one can think of such a method it can usually be translated into a method which could be applied to the machine.

(ii) Break the problem down

This in effect means to decide which parts of the problem should be made into definite subroutines. The purpose of this is partly to make the problem easier to think about, the remaining work consisting of a number of 'limited objective' problems. Another reason for breaking down is to facilitate the solution of other problems by the provision of useful subroutines. For instance if the problem on hand were the calculation of Bessel functions and it had already been decided to use the differential equation, it might be wise to make and use a subroutine for the solution of linear second order differential equations. This subroutine would in general be useful in connection with other subroutines which calculate the coefficients of the equation.

(iii) Do the programming of the new subroutines

It is better to do the programming of the subroutines before that of the main routine, because there will be a number of details which will only be known after the subroutine has been made, e.g. scale factors applied to the results, number of pages occupied by the subroutine, etc. It also frequently happens in the making of the subroutine that some relatively small change in its proposed properties is desirable. Changes of these details may put the main routine seriously out if it were made first. There is a danger that this procedure may result in one's 'not seeing the wood for the trees', but this should not happen if the original plan was well thought out. The programming of each subroutine can itself be divided into parts.

a) As with programming a whole problem a plan is needed for a subroutine. A convenient aid in this is the 'block schematic diagram'. This consists of a number of operations described in English (or any private notation that the programmer prefers) and joined by arrows. Two arrows may leave a point where a test occurs, or more if a variable control transfer number is used. Notes may also be made showing what is tested, or how many times a loop is to be traversed (see p. 56).

b) The operations appearing as blocks in a) may be replaced by actual instructions. It is usually not worth while at first to write down more than the last two characters of the (presumptive) instruction, i.e. the B line and function parts. These are quite enough to remind one of what was the purpose of the instruction.

c) One may then write the instructions into a page, deciding at the same time what are to be the addresses involved. Some of the finer points of this are described in the 'hints' in the next section.

d) When the programme is complete, check sheets must be made. This process has already been described (p. 9). It is often advisable to start making check sheets long before the program is complete; one should in fact begin them as soon as one feels that one has got into a muddle. It is often possible to work out most of the programme on the check sheets and afterwards transfer back onto the page or pages of instructions.

(iv) Programme the main routine

This follows principles similar to (iii).

Of course these remarks merely represent one possible way of doing programming. Individuals will no doubt vary as to the methods they prefer.

16 Programming hints

This section consists of a number of detached and rather trivial tricks and precepts, which are nevertheless considered worth while having in writing.

Manoevring space

It is seldom that one writes down a page of instructions for the first time without having forgotten a few vital instructions. It is therefore considered desirable to aim, not at pages which are chock-full, but say at ones which are about five-eighths full. The extra space is best left *between* sequences of consecutive instructions, so that once sequence may be extended without interfering with another. The spaces can also be filled with numbers if desired, when the discovery of mistakes calls for retrenchment. In this connection see also *sandwiching*. Another useful way of using reserve space is to put in a number of dummy stops or of time wasting instructions, or hoots. The latter provide 'rhythm clicks' which are very informative concerning the progress of the routine.

Do programming directly in teleprint code

It is never too soon to learn the meanings of the 64 functions [i.e., the opcodes]. The way to do so is to start programming in teleprint code straight away. Keep a list of the meanings always at hand, and refer to it as much as you wish: you will find that after a week very few references are necessary. You will not yet know all the codes, but you will know a 'working selection'. Likewise you will eventually get to know the teleprint equivalents (p. 3), but this is likely to be slower, chiefly because it is less essential to know them. Although the lines are given names which are in teleprint code, and which also correspond to numbers, for many applications particularly the ones which do not use 'systematic storage space', it is not necessary to know anything concerning the relation of these labellings, or even to have very much to do with the numbers at all. The names of the lines are just used as labels. Later it will be desirable to know the teleprint equivalents of the single characters by heart, but it is never necessary to know the equivalents for pairs of characters.

Counting procedure

One of the commonest operations is a sequence of instructions to be repeated a given number of times. Sometimes the sequence is in two successive parts of which the first is to be omitted on the first round

a) Case of omission. The counting process may be done in the B tube e.g. in **B**7. The repetitive part should preferably be programmed as follows:

(Reduce content of B line by 1)
First part of sequence
→ Second part of sequence
(Reduce content of B line by 1)
Test (/T)

The arrow from the left shows how the loop is entered, by a control transfer (e.g. `/P`). The B line must first have been set to an appropriate value. The value chosen depends on the number of times the operation is to be repeated, and also on where the reduction of the B line is done. It is intended that only one of the bracketed lines must be struck out. In whichever place it is put the number of times the reduction occurs will be greater by one than the number set in the B line, if (as probably) the latter is less than 2^{19}.

b) Case of no omission. The control transfer may be omitted thus

Set B line
Reduce B line
Sequence
Test

The rule about the number of repetitions still applies.

It is frequently desirable to subtract something other than 1. This may be because the number of repetitions is the result of a computation, and is given with some factor applied, e.g. a power of two. Alternatively it may be desired to save a line by setting the B line with some quantity already available, e.g. to count 12 one might set `GC/J` into the B tube, and subtract `///E`, the former being supposed an instruction which is used in any case, the latter available in PERM.

Discrimination by control transfer

When two cases have to be given quite different treatment, involving different sequences of instructions, it is natural to choose the relevant sequences by a test instruction (i.e. conditional transfer `/T`, `/H`, `/M`, or `/O`). When there is a large number the best method is to manufacture a control transfer number which will lead to the appropriate sequence. A good example of this is in the input programme where the six warning characters have to be given six different treatments, and the remaining twenty-six are given a seventh. This is dealt with as follows. If α is the character in question α`///` is set to **B**6, and the instruction `//IP` given. The lines `//` to `£/` contain the seven control transfer numbers appropriate to the thirty two possible values of α and the sequence required is immediately entered.

The B-tube as shunting station

When the B tube is used for the transfer of 20 digits from one short line to another, and both locations are fixed and known to the programmer, there is no difficulty. The difficulty arises when one of the short lines is in an indeterminate position, i.e. one computed by the machine, for it is usual to keep such indeterminate positions in a B-line, and the instructions involved will therefore presumably have to refer to two B-lines, one being the shunting station, the other describing the indeterminate position. To be more specific suppose that a quantity $\alpha\beta$ can easily be determined and that it is desired to set $[\alpha\beta + \texttt{GI}]'_s = [\texttt{OK}]_s$. How should this be done? The instruction `OKQO` will effect $\mathbf{B}7' = [\texttt{OK}]_s$. We need to follow this with $[\alpha\beta + \texttt{GI}]'_s = \mathbf{B}7$, which can be achieved either by the instruction `GIPZ` when $\mathbf{B}6 = \alpha\beta\texttt{E/}$ or by `GIIZ` when $\mathbf{B}6 = \alpha\beta\texttt{Z/}$. There is a school of thought which maintains that a difference between presumptive and actual instructions in respect of the function number is to be deplored, and that the former instruction is therefore to be preferred to the latter. The instructions TT, TZ, TL were introduced in deference to this view; they do not improve speed or reduce space, but they may save a little programmer's time. When doing the reverse process, e.g. setting $[\mathtt{OK}]'_s = [\alpha\beta + \mathtt{GI}]_s$ the situation is usually rather simpler. One can set $\mathbf{B}7' = \alpha\beta\mathtt{//}$ and follow this with GIQT and OKQB.

Note: If $[\alpha\beta]_+ + [\gamma\delta]_+ > 2^{10}$, $[\alpha\beta + \gamma\delta]'_s = \mathbf{B}7$ is achieved with $\mathbf{B}6 = \alpha\beta\mathtt{//}$ and presumptive instruction $\gamma\delta\mathtt{PZ}$.

Omission of counting

If the operation to be repeated contains rather few instructions e.g. three, and is crucial for the speed of the whole process it may be best to omit the instructions concerned with counting and to repeat the instructions concerned with the process in question the requisite number of times. Sometimes the number of repetitions may be the result of calculation, but even then the omission of counting method may still be applied, the number of repetitions being controlled through a control transfer entering the sequence of repeated operations at the appropriate point.

Alternative entry

It is often necessary to have a number of routines differing in certain minor particulars. One would like to use essentially the same instructions for all of them. The most convenient method seems to be to use one assembly of instructions, with various points of entry. The cues of these routines will then differ only in their first ten digits.

Changing sign in the accumulator

The instruction DSTJ has the effect $\mathbf{A}' = \{1 - \mathbf{A}_\pm\}_0^{79}$ which for most purposes is as good as $\mathbf{A}' = \{-\mathbf{A}_\pm\}_0^{79}$ which can only be achieved in two instructions, or more if 80 digits are required.

Twenty-digit numbers

It is often sufficient to specify tabular numbers to twenty digits only. One might for instance wish to have values of $2^{38}\log n$ for n from 1 to 32 with an error of not more than 2^{20}. This can be achieved by putting e.g. $[\mathtt{/}\tfrac{1}{4} + n]_s = \{\log n\}_{-19}^{1}$. Then $[\mathtt{/}\tfrac{1}{4} + n - 1]_+ = 2^{38}\log n + \Theta(2^{20})$.

Clearing the accumulator

The beginner is liable either to leave things in the accumulator to get mixed up with the next calculation or else to put in accumulator clearing instructions which could easily have been avoided. In fact it is very seldom necessary to give a special instruction for clearing the accumulator, if the points below are held in mind.

(a) Instruction `TA` clears the accumulator as well as transferring **L** [i.e., the lower half of the accumulator] to store. If both halves of the accumulator are required to be stored one can use `/U` twice and the accumulator will be cleared.

(b) If an expression of form $a + bc$ is required and the accumulator is not clear the term a should be put into the accumulator first. This applies if the final value required will be in **L**.

(c) When doing multiplications with results taken from **M** [i.e., the upper half of the accumulator] it is not necessary to clear the whole of the accumulator in advance, but only **M**. The maximum error will be 1 in either case: the mean square error will be one third with clearing but only one sixth without. If the results are taken out with `/A` then **M** remains clear for another multiplication.

Electronic space economy measures

We have explained that the economising of instructions in order to reduce the space occupied in the magnetic store is seldom worth while. There are however occasions when it is worth while to economise them to save space in the electronic store. This is nearly always in order to get instructions either into one page or into two pages. To do so makes the routine tidier, and usually has time-economy effects. For instance a one-page routine may be combined with another one-page (master) routine which uses it again and again without losing time over magnetic transfers. In these cases the two routines are able to be in the electronic store together, one on one page and one on the other. This effect can also be achieved by the disagreeable device of borrowing part of the systematic working space (if available) for part of the master routine. If a routine occupies more than two pages then (unless the same objectionable shift is relied on) it must involve magnetic transfers, and consequent loss of time. To some extent then the considerations mentioned under ‘manoevring space’ may be overruled, though they generally apply for routines of three or more pages. Some possible economy measures are described below.

Of course an economy measure which simply reduces the number of instructions in a straight sequence will normally be a time economy as well. We have mentioned two or three devices for keeping the number of instructions down, but this will mostly be learnt by experience.

Duplication of use of lines

The chief space economy measure available other than reducing the number of instructions is the use of a line for more than one purpose. One or two forms of this have already been mentioned. It is usual for instance to use addressless instructions

(p. 20) also as control transfer numbers. Another case was mentioned under the head of counting. No attempt can be made to list all such devices, but there are a number associated with control transfer numbers. These are sufficiently numerous that it should nearly always be possible to avoid using any lines specially for control transfer numbers. To make a point of doing so in cases where it is not strictly necessary is however strongly to be discouraged, as liable to lead to a most wasteful use of programmers' time. The methods already known to be available are mentioned below.

Sandwiching

If a (short) line is sandwiched between two sequences of instructions, the instruction which uses that short line can also be used as a control transfer number for the beginning of the later sequence of instructions. A long line can only be used in this way if it ends with a pair of characters representing a harmless instruction. Lines ending with `T£`, `Z£`, . . . , `££` are almost the only suitable ones.

Positioning of dummy stops

If a dummy stop or other addressless instruction be placed immediately before a 'junction', (i.e. an instruction to which a control transfer is made, but which is not the beginning of a straight sequence) then this addressless instruction may be used as a control transfer number in the usual way, and the control transfer instruction may also be used as a control transfer number for the transfer to the junction. This position will not however always be suitable from other points of view.

Relative control transfers

Relative transfers (e.g. `/Q`) are more troublesome to use than absolute ones, but provide a second string. The necessary relative transfer number may sometimes be found in PERM, or if part of the routine itself is being used as special working space, it may be found in the routine itself.

Inaccurate numbers

If a line pair is required to be known to more than twenty but less than thirty binary digits, the first two characters of the line pair used may be changed to a control transfer number. Likewise if a number is required to less than ten digits a control transfer number may also be concealed in it.

Changeling instructions

This is another space economy measure but not concerned with control transfer numbers. Suppose that the same sequence of instructions is to be used twice in different connections, but with one instruction changed. It may be worth while to use the same actual sequence of instructions. The troublesome instruction may be altered with a B line e.g. if the two effects required are `IKT¼` and `IKTF` they may be achieved with the instruction `IEZ¼`, **B**1 having as content either `/C//` or `/C/S`.

Wholesale reciprocals

A few devices which are almost purely computational, having little connection with the machine, may also be mentioned. One of these is a method of obtaining reciprocals if a number are required at once. The simplest case is where two are required, e.g. a^{-1} and b^{-1}. One can get them with only one use of the reciprocals routine, using it only to obtain $(ab)^{-1}$ and multiplying by b or by a to obtain a^{-1} or b^{-1}. Any number of reciprocals may be obtained in a similar way with only one use of the reciprocals routine, but all the numbers to be inverted must be known before any reciprocal can be obtained. Best accuracy is retained if the numbers are brought to 'standard form' i.e. to between $\frac{1}{2}$ and 1 before the process is applied.

Tchebysheff polynomials

The number of terms to be taken in a power series may often be reduced by the use of Tchebysheff polynomials. The Tchebysheff polynomial $T(x)$ of degree n is given by

$$\begin{aligned} T_0(\cos\theta) &= 1 \\ T_n(\cos\theta) &= 2^{1-n}\cos n\theta \quad (n \geq 0) \end{aligned}$$

It has the property that the coefficient of x^n is 1 and the other coefficients are chosen so that the maximum modulus of the polynomial over the interval (-1,1) shall be minimised. The resulting minimum-maximum has the value 2^{1-n}. The application of these polynomials to the calculation of power series is that one may replace a term ax^n $(-1 \leq x \leq 1)$ by $a(x^n - T_n(x))$ which is of degree $n-1$, with an error of at most $2^{1-n}a$, as compared with an error of a, which would occur if the term were merely dropped. In the case that the relevant range is $0 \leq x \leq 1$ one replaces ax^n by $a(x^n - T_n(\frac{x+1}{2}))$, with an error at most $2^{1-2n}a$. [The ms. contains a crossed-out note that the $0 \leq x \leq 1$ condition arises "by a substitution when odd or even functions are to be calculated".]

The coefficients of the polynomials may be calculated by using the recurrence relations [as documented in the ms., not yet checked]

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1}(x) &= xT_n(x) - \frac{1}{4}T_{n-1}(x) \quad (n > 1) \end{aligned}$$

and if we put $V_n(x) = T_n(\frac{x+1}{2})$ its coefficients may be calculated from

$$\begin{aligned} V_0(x) &= 1 \\ V_1(x) &= \frac{x+1}{2} \\ V_{n+1}(x) &= \frac{x+1}{2} V_n(x) - \frac{1}{4} V_{n-1}(x) \quad (n > 1) \end{aligned}$$

A number of the polynomials are given below.

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_2(x) &= \frac{1}{2}(2x-1) \\ T_3(x) &= \frac{1}{4}(4x^3 - 3x) \\ T_4(x) &= \frac{1}{8}(8x^4 - 8x^2 + 1) \\ T_5(x) &= \frac{1}{16}(16x^5 - 20x^3 + 5x) \\ T_6(x) &= \frac{1}{32}(32x^6 - 8x^4 + 18x^2 - 1) \\ T_7(x) &= \frac{1}{64}(64x^7 - 112x^5 + 56x^3 - 7x) \\ T_8(x) &= \frac{1}{128}(128x^8 - 256x^6 + 160x^4 - 32x^2 + 1) \\ T_9(x) &= \frac{1}{256}(256x^9 - 576x^7 + 432x^5 - 120x^2 + 9x) \end{aligned}$$

17 The official account of a routine

Here we mainly assemble what has been said about the official account of the routine elsewhere in the handbook. The account should include

(i) The name of the routine. The length of this must not exceed 25 characters, and should aim at about eight. It should preferably be pronounceable, and give some indication of the purpose of the routine.

(ii) A short description of the purpose of the routine in general terms in English.

(iii) The cue, or rather the 'skeleton cue' and the principal lines `/E` and `/A`. The cue consists of forty digits made up as follows.

(a) First ten digits. This is the control transfer number i.e. 1 less than the name of the line in which the routine starts.

(b) Digits 10-19. The 'check characters'. They are given by the value of $\{1025([\texttt{/E}]_+ - [\texttt{/A}]_+)\}_{10}^{19}$ when the routine is first entered. For two page routines this is determined by the 'principal lines' `/E` and `/A` of the routine itself, and is independent of the context in which it is used. This will also be the case for one-page routines with false cue occupying page 1. In those cases digits 10-19 of the cue can be given in the official account. In other cases, viz. page 0 routines and page 1 routines with true cue, these digits must be left blank. They have to be filled in differently in different applications. The information required for the filling in consists of the principal lines (`/E` and `/A`) of the routine in question and the routine from which it is reached.

    Sometimes a routine will have a 'variable' or 'undetermined' subroutine, i.e. one which is not decided until after the routine itself has been completed. This occurs for instance when an undetermined function is involved, e.g. in calculating $\int_0^1 f(x)dx$. In such cases the function is determined by a subroutine, and the subroutine is given through its cue.

    These variable subroutines will be subject to numerous restrictions imposed by the master routine, e.g. certain store lines, used by the master routine must not be used by the subroutine, and the results must be given in a specified form. The account of the master routine must also specify its own principal lines at the time of entering the subroutine, in order that the cue of the subroutine may be determinable. These principal lines will be presumed to be the same as applied on entering the master routine unless otherwise specified.

(c) Digits 20-39. These determine the magnetic transfer which will bring the routine to pages 0,1. In the case that digit 39 is 0 we have what is called a true cue, and those twenty digits are simply the magnetic instruction which has to be obeyed. If however digit 39 is 1 we have a false cue, and these twenty digits merely tell us where the magnetic instruction of the true cue is to be found. The last ten digits, with the final 1 replaced by 0, are the number of the track in which it lies. It will be in the right half of the track and the remaining digits give the line in which the required magnetic instruction will appear if the half track in question is transferred to p. 0.

If one is concerned with the false cue of a one page routine using page 1, the check number can be given without qualifying phrase, by using one's knowledge of the content of the appropriate line of a cue-bearing half track. One may take it that $[\texttt{/A}]_+ =$ track number. But more often these routines will come to p. 0 or will bring their partner pages with them.

It is always recommended that single page routines be given an alternative cue which results in both halves of its track being read. This is the easier alternative to use in cases where the routine is not used in conjunction with another one-page routine.

Routines will often have a number of cues corresponding to alternative entries.

(iv) The effects of the routine must be described accurately, so far as they are known or considered of interest. This will often consist of equations or inequalities relating the states of the machine immediately before entering and immediately after leaving the routine. It must state conditions of validity, accuracy of results etc., and should make some statement of the time taken. Those lines of the store which are altered must be mentioned, if not one of `GK`,`MK`,`VK` or on p0 or p1. It is important at this point to remember to include the lines altered in subroutines. B lines other than **B7** which are altered must be mentioned.

(v) Some account of the method should be attempted if this is not obvious. Unusual tricks (e.g. unusual uses of the B tube) should be pointed out.

(vi) The subroutines used should be listed. These need only be the subroutines directly entered from the routine. If it is required to know the indirectly entered subroutines this information must be obtained indirectly.

(vii) The names of the tapes required in order to put the routine into force should be listed. It will usually be the case that a job uses many subroutines in several different ways, so that the list of tapes required will have many duplicates. There will also probably be many whose content is known to be already stored within the machine. To save space therefore we give only the titles of the tapes in the official account of the routine. Additional data about the tapes must be found elsewhere.

(viii) It may happen that the routine uses ACTION as a subroutine. In this case the controller is required to take certain action depending on the printed output. The considerations governing the choice of action must be explained elsewhere, preferably in the account of one of the routines. It will usually be the case that the 'descriptive word' which describes the type of action is handed on from routine to routine until it is eventually printed by OUTPUTA under orders from ACTION. The explanation will usually be given in the routine from which the word originates.

Official accounts of a number of routines are given with this handbook. [But regrettably, not in the copy available to the transcriber; the official accounts of ACTION and OUTPUTA would be of great interest. The closest thing available is the set of somewhat informal accounts of routines used with the pilot machine in the appendix.]

18 Tapes

Tapes may be classified in a number of ways.

1. They may be titled or not titled, i.e. they may or may not start with a `Q`-sequence. Untitled tapes are somewhat to be discouraged, and the convention that untitled tapes may be destroyed is under consideration.

2. They may be 'input', 'output', or 'other purpose' tapes. If the printer is being used there is little point in output tapes except where they are to be used later for input, either as they stand, or after combining with others in a copying process.

3. Input tapes will nearly always be used in connection with the regular input routine INPUT. There may however be a few 'irregular' input tapes. These are likely to be chiefly useful where single characters are significant standing by themselves.

4. The most important tapes using the routine INPUT are 'writing tapes' and 'job-steering tapes'.

Writing tapes

These tapes are the standard form of storage for routines and other large blocks of information. One tape is made for each half track, and takes the form

- `Q` (title)
- `KAK@///`$\frac{1}{4}$ (magnetic half cue) or `KAK@//E`$\frac{1}{4}$ (magnetic half cue)
- `KPK` (check sum)
- `Y`
- (punching proper)
- `Z`

The title does not require much explanation. It must be preceded by the character giving its length. The magnetic half cue is similar to the second half of the cue of a routine. It determines the track in which the material on the tape is to be stored in exactly the same way as the second half of a cue of a routine determines the track from which the routine is to be read. In the case of a routine more is given viz. the whole magnetic instruction required for that reading. In the case of a tape the remainder of the magnetic instruction is given in the earlier part of the 'destination sequence'. This is `KAK@///`$\frac{1}{4}$ for tapes destined for a left half track, `KAK@//E`$\frac{1}{4}$ for ones destined for right half tracks. In the case that the tape is used for a routine the magnetic half of the cue will normally be identical with the second half of the cue of the routine, though this will not apply when the routine is stored in more than one track.

The punching proper results in transferring certain information to page 4, the 'systematic working space'. It will normally be punched with the '`J`' meaningful sequences.

The check sum is the sum of the long lines of page 4 after the punching proper has passed through.

The writing tapes are intended for use with the routine WRITE. If this routine is entered when the beginning of a writing tape is in the input, the title will appear in the output and the information in the punching proper will be written in the appropriate half-track, after a number of precautions have been taken, including a check on the track-selection mechanism, verification that the correct check sum is formed, and a check that no digits were altered in the writing process. For further information on this routine its official account must be consulted.

Writing tapes should normally be produced automatically from information within the machine. The routine for this purpose will shortly be available, but details are not yet known.

It is intended that a typewritten list be kept of all tapes with their titles, destination sequences and check sums, i.e. essentially all of the tape except the punching proper.

Job-steering tapes

It is convenient to use INPUT, in combination with an appropriate tape, as a master routine for a job. In so far as the master routine is non-repetitive this is no slower than putting a master routine away in the magnetic store and then entering it. The advantages of this method, when applicable, are

1. The state of progress visible to controller through position of tape.
2. No necessity to assign magnetic or electronic storage
3. Subroutines can be entered with less formality by using meaningful sequences `Y` and `Z`.
4. Writing operations are avoided.

Even when a certain amount of repetition is involved it may be less trouble to have repetition e.g. by copying on the tape rather than to attempt to produce a routine in which the repetition is achieved by passing through the same instructions again and again. An example of this is described in the appendix (p. 77).

Exercise. Suppose that `££ABF/EZ` is the cue of a routine whose effect is given by $f([\texttt{/C}]) = [\texttt{:C}]'$, and does not alter any long lines, except the conventional ones and `:C`, produce a job steering tape for printing out the values of f for six given arguments. Use OUTPUTA.

Directories

The twenty digit rows that are put into a cue-bearing track to assist the routine changing sequence in turning false cues into true cues may be called ‘tape addresses’. For every job it is necessary to assign an address to each tape involved. The list of these addresses or directory has to be put into the cue bearing tracks (which should be very few) before the job can begin. This process is certainly something of an imposition. It is hoped that it will be possible to have a small number of standard directories of which one will be suitable for almost any job. Those routines which are special to the job may be given true cues and will not need directory entries.

19 Checking procedures

A variety of causes may lead to wrong results. The chief of them are

- Machine breakdown, i.e. systematic error
- Intermittent machine error
- Wrong programmes on paper
- Wrong programmes in magnetic store, i.e. disagreeing with paper
- Finger trouble, i.e. erroneous use of controls

Measures concerning machine breakdown

Machine breakdown usually shows itself very clearly. In any case the measures against intermittent faults are also adequate for detecting most breakdowns.

Measures against intermittent error

This is one of the most serious problems about the machine, and certainly the most exasperating. Several kinds of action are taken about these errors.

(a) Before using the machine a daily sequence of test routines is run through. These are designed to try out every facility of the machine, though not in every combination: they test the storing power of individual lines of the electronic store and the magnetic transfers. All errors are counted. If each test routine will run for five minutes (say), without error the machine is pronounced serviceable.

(b) In any calculation, identities should be looked for and verified in the computation, e.g. if both the sine and the cosine of an angle have been obtained their squares should be added, and the machine stopped if the result is too far from unity.

(c) It is as well to try to divide a calculation up into parts which take sufficiently little time that the probability of an error having been made is relatively small, e.g. less than 0.2. If the calculation is then repeated and the same answer obtained it is rather strong evidence that the machine made no mistake on either occasion.

(d) It may be impracticable to take the course outlined in (c). It is then probably necessary to divide the working space of the machine into three equal parts A, B and C, say, either of which would be sufficient for the job if the machine did not make errors. The computation is all done in A, and is divided into 'bursts'. Each burst represents an attempt to get through a part of the calculation without error, and may be regarded as the computation of a function $f(A)$. One proceeds as follows

- Copy A into B and C
- Compute $f(A)$, possibly wrongly, leaving result in A

- Interchange A and B
- Again compute $f(A)$, leaving result in A
- Compare A and B. If there is agreement proceed to the next burst. If not copy C back to A and again compute f. If it still disagrees with B, copy C back to A [and B] and go back to the beginning of the work on this burst.

When the bursts are done in this way the working space is reduced three to one, and for many jobs this will be a fatal objection. Usually however it will be possible to apply the burst method more or less locally, i.e. to divide the problem up into parts which alter only a small part of the store and apply the bursts method to these parts.

(e) Certain particular programmed checks may be mentioned here. There is a check on punching in the input programme. If characters not representing numerals are punched for numerals the machine will stop. In the routine changing sequence there is an arrangement for verifying that the right new routine has been entered.

The effect of this last precaution, and of many other programmed checks is not so much to let one know that something has gone wrong, as to let one know it sufficiently early to avoid disastrous consequences (e.g. writing transfers), and to attempt some sort of diagnosis of the trouble. So serious an error as entering a wrong routine is almost certain eventually to have very noticeable effects.

Measures against wrong programmes

We have already described the process of making check sheets. If the initial conditions are carefully chosen nearly all the errors of the programme will be shown up when making the check sheets, if this is done conscientiously. One should aim at forgetting, whilst making the check sheets, both the purpose and the method of the routine: this tends to prevent one from writing down what one thinks an instruction should do, rather than what it really does. No doubt a colleague would be able to adopt the open mind better than the programmer himself, but he would be in danger of having such an open mind that he accepts everything that happens without question and goes on very far beyond the appearance of programme errors. A possible arrangement is for the programmer to make up check sheets and copy every tenth line (say) to blank sheets which can be passed to a colleague, for him to fill in the intermediate lines. The chief reasons for errors not being noticed at the check stage are

(a) Preconceived ideas

(b) Insufficiently many cases examined

(c) Making alterations in the routine in the middle of the check sheet work without verifying that the earlier parts of the check sheets are unaffected. Likewise altering instructions on check sheets without also altering the routine.

(d) Incorrect magnetic transfers.

(e) Incorrect control transfers, particularly 'failure to subtract 1' and confusion of the control transfer number with its own address.

(f) Failure to observe all the details of subroutines. None of the instructions of a subroutine should be shown on the check sheets. The subroutine should be treated something like a single instruction, whose properties are given in the official account of the routine. A special danger is that one may not notice the lines used in the subroutine. Every such line should be noted on the check sheets in column 4.

In order to reduce the danger (b) one should at least arrange that every instruction of the programme appears somewhere on the check sheets for some case. One should also make sure that all the points of difficulty that were considered in the construction of the programmer are represented.

There *are* sometimes points which are liable to get through the check sheets, and correspond to exceptional cases which were not considered by the programmer. An interesting example of this occurred with the obsolete machine in connection with linear interpolation. On that machine multiplication was very much slower than other operations, and the time involved was essentially proportional to the number of digits 1 in the multiplier. For this reason it was decided to approximate $\alpha_{f+}x+(1-\alpha_{f+})y$ by $(\alpha\wedge\texttt{//////}\boldsymbol{\pounds\pounds})_{f+}x+(\{-(\alpha\wedge\texttt{//////}\boldsymbol{\pounds\pounds})_{\pm}\}_0^{79})_{f+}y$. This is a reasonable approximation for all values of α_{f+} except those less than 2^{-10}, for which it is 0. The chance of this being detected by check sheets with random numbers is of course 2^{-10}. It was actually detected because $\alpha_{f+} = 0$ occurred in one of the cases investigated. A somewhat similar danger is that of 'overrunning the capacity of a line' e.g. of applying instruction `/J` where $\mathbf{A}_+ + 2^{40}\mathbf{S}_+ > 2^{80}$ so that $\mathbf{A}'_+ \neq \mathbf{A}_+ + 2^{40}\mathbf{S}_+$. This danger can only be eliminated by a detailed investigation of inequalities connected with the content of store lines at various stages in the computation: this can be made to be an investigation of accuracy also, and can in fact be made to constitute a formal proof of the validity of the routine for its officially stated properties. The accurate notations introduced on pp. 14-16 are a considerable aid in doing this, but these need to be supplemented in various ways. It is necessary for instance to have a notation for the content of the accumulator. The writer uses the notation $(\alpha\beta)$ for 'the content of $\mathbf{A}$ immediately following the instruction stored in $\alpha\beta$. Likewise $(\alpha\beta|)$ and $(|\alpha\beta)$ represent the contents of $\mathbf{L}$ and $\mathbf{M}$ at this time. But the use of such a notation implies that the investigation is restricted to a 'straight sequence' of instructions without repetitions, and the notation for store line contents implies that no line is altered more than once. No entirely satisfactory system has yet been worked out, and the subject must therefore be left. It may be added however than the understanding of the theory of a routine may be greatly aided by providing at the time of construction one or two statements concerning the state of the machine at well chosen points. This was done with SUMPGA (p. 12).

[Turing's last statement here is remarkably prescient — the theory of formal program verification which has subsequently developed is largely concerned with tracing through the consequences of such things as preconditions, postconditions, and loop invariants, which are in fact statements concerning the state of the machine at well chosen points in the program.]

Measures against routines wrong in magnetic tracks

Routines may get wrong in the magnetic tracks either because the tapes have been wrongly punched, or because of a machine error or finger trouble in connection with writing transfers. There is little that can be done about the tapes but to take care in the punching and the checking of the printed form of the punched tape. These errors are not a great anxiety, for they are likely to be eliminated at the same time as genuine programme errors and are usually smaller in number and more easily found. Once a correct tape has been made it can be relied on. The contents of the tracks can be satisfactorily checked by means of the ROLCAL routine which adds up and records the contents of each of the tracks. The sums may then be compared with the anticipated values, previously obtained on some occasion when all was well.

Measures against finger trouble

We have already described 'the formal mode', which should be a complete insurance against finger trouble. It is recommended that the formal mode be applied when the results are anything more than experimental.

20 Brief reminders

Very little of what has been said in this handbook needs always to be remembered. The things which are vital are

- The equations corresponding to the various functions
- Meanings of magnetic instructions
- The content of PERM
- The normal use of pages
- Teleprinter equivalents
- Essentials of the input routine
- Some knowledge of conventions

An attempt has been made to condense all this information on two sides of a sheet of paper. Naturally in such a form it needs much interpretation, and all the difficult parts are unrepresented. The equations are given in the most abbreviated form, and sometimes rather ambiguously. The input programme is represented by a number of examples which should be enough to remind the reader who has been through the section on this subject. See Fig.G [which is two pages in the manuscript, but three in this transcription].

Appendix — The Pilot Machine (Manchester Computer Mark I)

It is proposed to describe here the experimental machine built at Manchester University during the period 1947-1950, usually then known as the 'existing machine' but here to be described as the 'pilot machine'. The description to be given here will apply to the most recent form of the machine. The development of the machine was essentially in the order used in describing the Mark II machine. There was first a baby machine, something like the 'reduced machine', to which were added first a larger variety of functions including logical operations and a multiplier with the necessary accompanying 80 digit accumulator. A two line B tube was added at the same time. In a further stage the magnetic wheel and the input and output were provided.

Arrangement of storage on pilot machine: The electronic storage consisted of four tubes, each containing 32 lines of 40 digits. The thirty-two lines appeared under one another, i.e. there was no arrangement in more than one column. The names given to these lines were `//`,`@/`,`:/`,. . . ,`VU` i.e. the first 128 non-negative even integers. Those lines which had the same second character were said to form a paragraph, thus a paragraph was either the upper or the lower half of a tube. These lines could be divided into halves for the purposes of using them as instructions. The left half of a line if used as an instruction was given the same name as the whole line, but the right half was given a name one greater, e.g. the left half of line `VU` would form the instruction `VU` and the right half the instruction `£U`.

The B-tube had only the two lines **B**0 and **B**1. These were 40 digit lines.

The selection of instructions was on a principle very analogous to that in the Ferranti machine although differing in many details. The control tube contained two forty digit lines, the 'instruction number' **C** being on one line, and the present instruction (actual) forming half of the other line or P.I. line. If the content of control is observed at a prepulse the actual instruction next to be obeyed is obtained as follows:

- Add 2 to control
- Obtain as presumptive instruction line the (40 digit) line named in control, the first digit of control being read as 0 for this purpose.
- Take the first twenty digits of the B line whose number is given in the digit 0 of this presumptive instruction line and the last twenty digits of the B line whose number is given in digit 20, add this combination to the presumptive instruction line, and we have the actual instruction line. The actual instruction is the left or right half of this line according as the I.N. line is even or odd.

In the actual instruction the first digit is irrelevant. If it is replaced by 0 the first eight digits represent a store line, which, apart from magnetic transfers, is the only one which will be altered in the obeying of the instruction and is the only one whose content is relevant to that instruction. Its content is represented by $\mathbf{S}$, $\mathbf{S}'$ in the equations for the instructions as in the Mark II machine. The ninth to the fifteenth digits of the instruction were irrelevant, and the last five formed a function symbol. The equations for the thirty-two functions are given on p. 74. The equations presumed to hold are as with the Mark II machine except that $\mathbf{C}' = \mathbf{C} + 2$ replaced $\mathbf{C}' = \mathbf{C} + 1$. The accumulator and multiplier are as with the Mark II machine, except for the times and the actual arrangements of the digits.

Computer Function Characters

0	`/`	Magnetic transfer	
1	`E`		(time waster)
2	`@`	$\mathbf{B}0' = \mathbf{S}$	
3	`A`	$\mathbf{B}1' = \mathbf{S}$	
4	`:`	$\mathbf{D}' = \mathbf{S}_\pm$	
5	`S`	$\mathbf{A}' = \mathbf{A} + \mathbf{DS}_\pm$	
6	`I`	$\mathbf{D}' = \mathbf{S}_+$	
7	`U`	$\mathbf{A}' = \mathbf{A} + \mathbf{DS}_+$	
8	$\frac{1}{4}$		(time waster)
9	`D`	$\mathbf{S}' = \mathbf{L}$	
10	`R`	$\mathbf{C}' = \mathbf{C} + 3 - sgn(\mathbf{A}_\pm + \frac{1}{2})$	(test)
11	`J`	$\mathbf{S}' = \mathbf{M}$	
12	`N`		Dummy stop
13	`F`	$\mathbf{P}' = 1$	" "
14	`C`	$\mathbf{P}' = 0$	" "
15	`K`	Hooter operates	
16	`T`	$\mathbf{A}' = \mathbf{A} \vee \mathbf{S}_\pm$	
17	`Z`	$\mathbf{L}' = \mathbf{M}$, $\mathbf{M}' = \mathbf{L}$	
18	`L`	$\mathbf{C}' = \mathbf{S}$	
19	`W`	$\mathbf{C}' = \mathbf{C} + \mathbf{S}$	
20	`H`	$\mathbf{A}' = \mathbf{S}_\pm$	
21	`Y`	$\mathbf{A}' = -\mathbf{S}_\pm$	
22	`P`	$\mathbf{A}' = 2\mathbf{S}_\pm$	
23	`Q`	$\mathbf{A}' = 0$	
24	`O`	$\mathbf{A}' = \mathbf{A} + \mathbf{S}_+$	
25	`B`	$\mathbf{A}' = \mathbf{M}$, $\mathbf{S}' = \mathbf{L}$	
26	`G`	$\mathbf{A}' = \mathbf{A} + 2^{40}\mathbf{S}_+$	
27	`"`	$\mathbf{A}' = \mathbf{L}$, $\mathbf{S}' = \mathbf{M}$	
28	`M`	$\mathbf{A}' = \mathbf{A} + \mathbf{S}_\pm$	
29	`X`	$\mathbf{A}' = \mathbf{A} - \mathbf{S}_\pm$	
30	`V`	$\mathbf{A}' = \mathbf{A} \not\equiv \mathbf{S}_\pm$	
31	`£`	$\mathbf{A}' = \mathbf{A} \wedge \mathbf{S}_\pm$	

When nothing is stated to the contrary the equations $\mathbf{A}' = \mathbf{A}$, $\mathbf{C}' = \mathbf{C} + 2$, $\mathbf{S}' = \mathbf{S}$, $\mathbf{P}' = \mathbf{P}$ hold. $\mathbf{A} = \mathbf{L} + 2^{40}\mathbf{M}$, $0 \leq \mathbf{L} < 2^{40}$, $0 \leq \mathbf{M} < 2^{40}$ always.

These equations will mostly be self-explanatory, after the explanations that have been given in connection with the Ferranti machine. Only `N`,`C`,`F`,`K` need explanation. These were dummy stops and the combinations `N`,`C`,`F`,`K`; `C`,`K`; `F`,`K`; none; of active dummy stops could be chosen with the aid of three switches. Functions `C` and `F` also did duty in connection with the output. This is explained by the equations for $\mathbf{P}$ which is a single digit store. States 0 and 1 of this store are to be interpreted as mark and space respectively and signals are sent out along a teleprint line to a punch accordingly.

It will be seen that there are no means of using the content of B-lines except to modify instructions, also that when altering B-lines the line concerned is chosen by the function symbol, not by the B digit. It will also be observed that the test instruction (`R`) is not combined with a transfer. It is usually necessary therefore to follow it with a control transfer.

Magnetic instructions on the pilot machine

The magnetic instructions were forty digit lines. The first three characters were irrelevant, the fourth and fifth gave the track, the sixth the pair of tubes, the seventh the function symbol and the digits of the eighth were used separately for special functions.

As regards the track number, provision was made in coding for 64 tracks, although there were never more than 30 operative, and not more than 15 except at the very end of the machine's life. The coding of the track number was the obvious one i.e. with magnetic instruction $\mathbf{I}$ the track number was $\{\mathbf{I}_+\}_{15+}^{20}$. Since there were only two partnered pairs of tubes viz. tubes 0,1 and 2,3 there were only two possibilities for the tube-pair character viz. `/` for 0,1 and `E` for 2,3. The functions were on a system very reminiscent of the Mark II machine viz.

Digit 30	Read 0, write 1
Digit 31	One page 0, two 1
Digit 32	Scan 0, Action 1
Digit 33	Normal 0, Reverse 1
Digit 34	Normal 0, Check 1

Digit 30 requires no further explanation, nor does 31. In connection with digit 32 it must be explained that the content of the track is divided into two halves, called the scan and action halves, functionally very similar to the left and right halves of the Mark II machine. This selects which half track is to be concerned when a magnetic transfer using only one half track occurs. Normal (digit 33) means scan with left, action with right. If digit 34 is 1 we get checking whichever value digit 30 has. When the check succeeds the equation $\mathbf{C}' = \mathbf{C} + 6$ applies but when it fails $\mathbf{C}' = \mathbf{C} + 2$. There was also a four digit binary counter, indicated with neons which counted the number of failed checks modulo 16.

The special functions will be described under the input mechanism.

Times on the pilot machine

The digit periods on the Mark I machine were of the same length (10 μs) as on the Mark II, but each beat consisted of 45 digit periods. The majority of instructions occupied 4 beats i.e. 1.8 ms. The exceptions were magnetic transfers, taking about 60 ms, active dummy stops (indefinite), and multiplications. The time occupied by a multiplication was dependent on the number of digits 1 in the multiplier, and if this number was m the time was $2m + 7 + (-1)^m$ beats. The average time for all multipliers was thus about 47 beats or 12 ordinary instructions. For multipliers which were powers of 2 the time was equal to that taken by two ordinary instructions. The time in input was 150 ms per character, and the same in output.

Input and Output on pilot machine

We have already explained how output is controlled through the digit **P**. It may be as well however to explain the operation of a teleprinter, or rather the nature of the signals which are transmitted down teleprinter lines. Ideally these signals change instantaneously from one voltage called mark to another called space. The mark signal also does duty as a 'stop' signal and the space as a 'start'. When the line is quiescent, i.e. when no meaningful signal is being transmitted it is permanently at mark, and this mark is interpreted as 'stop'. If this mark gives way to space the succeeding 150 ms of signal are expected to conform to one of 32 patterns corresponding to which of the 32 teleprint characters are being transmitted in that period. The first 20 ms should be space, representing start, and the last 30 ms should be mark, representing stop. The intervening period is divided into five periods of 20 ms each. These periods are associated in order with the five digits of the character to be transmitted. A digit 1 is represented by mark and 0 by space. A certain tolerance is of course permitted on this ideal waveform. In particular a tolerance of 5% plus or minus is allowed on speeds of transmission.

The output routine enabled one to punch the content of any line. It operated simply by giving instruction to `F` and `C` at appropriate moments, in accordance with the above account.

An input routine might have been used accepting the same type of signals, but such a system would not have been very flexible. It was arranged instead that the input should be along six lines. When a tape was being read each character was transmitted only for a period of 20 ms, and these periods were at intervals of 150 ms. This effect was obtained by a simple modification of a tape reader (transmitter-distributor), the periods of transmission coinciding with the periods when the unmodified machine would have transmitted the first digit of a character. The sixth line always transmitted 1 during these periods, and was used to indicate these periods. All lines transmitted 0 at other times. The effect of these signals was to modify the content of the accumulator. The five digits of the character transmitted were connected to **P**35-**P**39 and also to **P**75-**P**79. The sixth line was connected to **P**0 and **P**40. All these were 'or' connections (see p. 30). Thus if the accumulator held `////////ABCDEFGH` just before a transmission period, and no instructions tending to alter the accumulator were applied and `U` were transmitted, it would contain `E//////UABCDEFGQ` from very soon after the beginning of the transmission period. [**P** in this description appears to be a persistent typo for **A**.]

It was found convenient to be able to start and stop the transmitter distributor at the discretion of the machine. A magnetic instruction with 1 for digit 36 would start it, and if this was absent a digit 1 in position 35 would stop it. Any magnetic transfer described by the remainder of the magnetic instruction would also have to be obeyed. A switch on the transmitter distributor itself enabled it to be switched off entirely. There was also on it a key producing the same effect as digit 36 of a magnetic instruction. A key corresponding to digit 35 would also have been a convenience.

The input and output apparatus produced considerable interference, and for this and reasons of space was housed in a room at some distance (on another floor) from the remainder of the equipment. An intercommunication system (not entirely distortionless, and much interfered with, both in the electrical and acoustic parts of the path) connected the two rooms, but it was found desirable to make it possible to control the machine from either room. The controls on input already mentioned, together with a key controlling **KEC**, were sufficient for controlling from the input-output room, provided the switches on the console were in standard position. The intercommunication system would then be mainly used to request that the switches be so set, or that some alteration be made concerning dummy stops. When controlling the machine from the main room, the facilities for stopping and starting the transmitter distributor automatically were most useful. This mode of operation was used most when verifying the correctness of routines with check sheets. A tape would be punched which with the aid of the input routine set the machine into the state from which its motion was to be investigated, and at which a dummy stop occurred, and the T.D. would be automatically stopped. From this tape another was produced bearing repetitions of the first, by the technique mentioned on p. 33. This tape would then be placed in the T.D. which would be switched on but stopped and the investigator would proceed to the machine room. By operating **KEC** it would then be possible for him to set the machine into the condition in which he was interested. This would involve the first copy of the original tape passing through the T.D, which would then stop. After pursuing his researches for some time the investigator could start again by once more operating **KEC** and consuming a further section of the tape. This Utopian state of affairs was only too often interrupted by the machine going out of control and consuming the whole tape.

As subsidiary tape handling equipment there was a hand punch (with considerable tendency to replace digits 1 by 0), and a printer mechanism. The latter was used to print out the contents of tapes. It could be set to any margin width and gave automatic carriage return and line feed at the right margin. This printer had figure shift and letter shift positions. In the letter shift position the alphabetic characters were represented by the capital letters, but the stunts were represented `/` for `%`, `2` for `@`, `4` for `:`, `8` for $\frac{1}{4}$, `5` for `"`, and `Ø` for *£*. In the figure shift position `Q` was represented by 1, W by 2, E by 3, R by 4, T by 5, Y by 6, U by 7, I by 8, O by 9, and P by 0. The other twenty two characters were represented by somewhat colourless characters, by no means standard, and only eleven of them different. An additional tape-handling facility was provided by the possibility of disconnecting the input and output from the machine and connecting them together so as to provide a means for copying tapes. For this purpose the T.D. produced normal teleprint signals rather than the special six-line signals used in input. Opportunities for releasing the equipment for copying purposes were frequently available.

Programming on the pilot machine

We shall describe here only those programmes which were actually used a considerable number of times and were not eventually superseded. These probably represent less than a quarter of those which were actually made. The others were either superseded by better routines, or assumed a form of machine which went out of date before being made, or required greater reliability in the machine than was forthcoming, or formed parts of schemes which never came into force for lack of time, or were just rather useless, or even not right.

Normal duties of tubes. PERM. Routine changing sequence.

As with the Mark II machine there were conventions about the normal uses of the four pages available. Pages 0 and 1 were for instructions, page 2 for systematic working space, and page 3 was divided between permanent information, unsystematic working space and the routine changing sequence. Paragraph I, i.e. lines /I to VI were unsystematic working space, with GI, MI, VI as special working space, and /U and OU was used for PERM, which contained a number of powers of 2, and also RYRYRYRY, E///E///, ££££££££, and ////////. Line VU was used for the cue, consisting of a control transfer number and a magnetic instruction packed into one line. The routine changing sequence was contained in the last halves of the lives OU,GU,MU whose contents were

```
OU      ::::::N
GU      QU::VU//
MU      V£::VU/L
```

The method of changing routines was to put the cue into VU, the link into the least significant half of the accumulator, and obey GU:L. This control transfer is followed by the dummy stop N, the magnetic instruction VU// and the control transfer VU/L. This is similar to the Mark II machine, but the checking of the transfer is omitted. Moreover false cues were never used.

[Standard routines for the pilot machine]

INPUT

The principle of meaningful sequences and warning characters was used as with the input routine of the Mark II machine. There were three warning characters `K`,`Q`, and `J` with respective lengths of meaningful sequence 11,11,10. With `K` and `Q` the second and third characters determined the line to which the remaining characters were to be sent. With `K` these two characters gave the address direct but with `Q` the middle digit of the third character was first replaced by 1. This made it impossible to interfere with the input programme itself when using warning character `Q`. With `J` the second character was sufficient to describe the address as this was restricted to be on [electronic] p.2, the page of systematic working space. For this purpose the 32 lines `/:`,`@:`,...,`VS` were to be called `/`,`E`,...,*£*. The content destined for the address so determined came from the remaining eight characters. There was also a scheme for obeying instructions from tapes. After doing any transfer as described above the accumulator would be filled from `HA` (l.s. [half]) and `PA` (m.s.) and then the two halves of line `G@` obeyed as instructions, the left half first. The accumulator was then emptied back into `HA` and `PA`, and `G@` set to contain two time wasting instructions. The consequence of this arrangement was that one could obey two instructions by writing them on a tape preceded by `KG@`. For this purpose `HA` and `PA` form a pseudo accumulator. This facility was mainly used for leaving the input routine, i.e. for the equivalent of warning character `Z` of the Mark II input routine.

OUT, OUTPG, OUTB

The fundamental output programme was OUT which enabled one to punch the whole or part of an eight character line. OUTPG enabled one to punch out the content of a whole page in the form of an input tape which could be used for putting the material back on another occasion. OUTB could be used for punching out the contents of any number of consecutive lines taken from almost anywhere in the store except pages 0,1.

Mathematical functions LOGSLOW, SINAPP, EXAPP, RECIP, RECROOT

The routines for mathematical functions were mainly chosen for simplicity of programme and generality of application [rather] than for speed.

LOGSLOW A slow routine for logarithms, applicable for range 1 to $2^{40}-1$, giving accuracy 2^{-36} taking time about 1.2 secs. This routine depended on a process of repeated doubling and squaring, doubling being applicable for values less than 2^{39}, and squaring (and dividing by 2^{40}) for larger values. A faster routine (about 0.2 secs) was made but never used. It used a table, to be obtained by the use of the slow routine.

SINAPP A slow routine for the sine, taking about half a second and accuracy about 2^{-36}. The angle was given in revolutions. The method depended on reducing to the cosine of a first quadrant angle, and using the duplication formula eight times, and the power series up tot he fourth power for an angle not greater than $\frac{\pi}{512}$.

EXAPP A routine for the exponentials of negative numbers without restriction but with poor accuracy. The time was about $\frac{1}{4}$ sec. This routine was very short occupying only a third of a page. The page was shared with SINAPP.

RECIP This was a relatively fast routine for the reciprocal, taking about $\frac{1}{4}$ sec. The method was to standardize the argument to a value between 2^{39} and 2^{40}. In the next step the five most significant digits were looked up in a sixteen entry table which gave an approximate reciprocal. By multiplying by this approximate reciprocal and using the formula $a^{-1} = b(ab)^{-1}$ the problem is reduced to that of finding the reciprocal of $1 - x$ where $|x| < \frac{1}{32}$. For this purpose the formula $(1 + x)(1 + x^2)(1 + x^4)$ was used as an approximate reciprocal of $1 - x$.

RECROOT This was a relatively slow method for the reciprocal square root. It depends on the case $p = 2$ of the recurrence relation

$$u_{n+1} = u_n(1 - \frac{1}{p} a u_n^p)$$

for $a^{\frac{-1}{p}}$. It was necessary for the argument to be within a relatively restricted range, and for u_0 to satisfy $\frac{1}{2} < au_0^2 < 2$. It was often found more convenient to use the logarithm and exponential, although much slower.

DBTEMP and BDTEMP

These were routines for conversion from binary to decimal and back, decimal numbers being according to the code

```
Q W E R T Y U I O P
1 2 3 4 5 6 7 8 9 0
```

Testing routines

A number of routines were also produced for the purpose of testing various features of the machine. These included special routines for testing the multiplier, the magnetic reading transfers and the holding properties of the stores. There were others for testing all the functions. Routines constructed for other purposes were often also used for testing.

Problems tackled

Very little time was available for full scale problems, and only three were attempted. These were

(i) Testing of Mersenne numbers $2^p - 1$ for primality by Lucas test (see e.g. Hardy and Wright p.223). By this means the primality of these Mersenne numbers, already known to be prime from other calculations, were checked. The values of p greater than 257 and less than 354 were also tested and the corresponding Mersenne numbers were found to be composite.

(ii) The Riemann hypothesis was investigated for the range $63 < (\frac{\tau}{2\pi})^{\frac{1}{2}} < 64$. This chiefly involved calculating the expression

$$Z(\tau) = 2\sum_{n=1}^{m} n^{\frac{-1}{2}} \cos 2\pi(\tau \log n - K(\tau)) + \frac{(-)^{m+1}}{(m+\frac{1}{2})^{\frac{1}{2}}} h(\zeta)$$

where

$$\begin{aligned} m &= [\tau^{\frac{1}{2}}] \\ \zeta &= \tau^{\frac{1}{2}} - m \\ k(\tau) &= \frac{1}{2}\tau(\log \tau - 1) - \frac{1}{16} \\ h(\zeta) &= \frac{\cos 2\pi(\zeta^2 - \zeta - \frac{1}{16})}{\cos 2\pi\zeta} \end{aligned}$$

for about a thousand values of τ. [Note that this is my best rendition of Turing's handwritten math, assuming that his typewritten t and handwritten τ in the ms. are meant to be equivalent, but I expect that there are errors here.] For this purpose the values of $\log n$ and $n^{-\frac{1}{2}}$ were taken from a table, and $\log \tau$ was obtained with the aid of LOGSLOW. The cosines were obtained by linear interpolation in a table with interval $\frac{\pi}{128}$. The time for each term of the series was about 160 ms.

(iii) A problem concerned with ray tracing through a lens system. The lens surfaces were spheres with collinear centres. The rays traced were mostly skew, i.e. did not meet the axis or line of the centers.

Reliability of the pilot machine

Judged from the point of view of the programmer, the least reliable part of the machine appeared to be the magnetic writing facilities. It is not known whether the writing was more often done wrong than the reading or less. The effect of incorrect writing were however so much more disastrous than any other mistake which could be made by the machine, that automatic writing was practically never done. The majority of computation was done with the writing power switched off. It is hoped that this difficulty will be met in the Mark II machine, partly by using the arrangements for switching off the write power from blocks of tracks or from single tracks, partly by greater reliability, and partly by making greater use of the input. The very much greater speed of input should facilitate this.

Other serious sources of error were the failure of storage tubes and the multiplier. Technical improvement has however been made in respect of both of these.

Figures

[The manuscript contains a number of figures at the end. Some of these are conversion and arithmetic tables whose details are of no conceivable historical interest; I just briefly describe them on this page. Those containing reference information not available elsewhere are given in on succeeding pages.]

Fig. A — Powers of 10

[This figure is a table giving powers of 10 in teleprint code, with appropriate scale factors. The last lines are:

$2^{-30}10^{20}$	`PSVGZQP@`
$2^{105}10^{-20}$	DLN¼¼BBJ

All powers of ten between these two are given.]

Fig. B — Binary-Decimal Conversion Table

[Conversion table for integers from 0 to 1023, with fractional equivalents alongside for use in interpolation.]

Fig. C — Multiplication table

Fig. D — Addition table

[These are two tables for the standard operations in base-32 arithmetic, much like grade-school addition and multiplication tables for decimal arithmetic except with 32 columns and rows.]

Fig. H — Multiplication by powers of 2

[A table to aid in computing the results of what we'd now call a shift operation, for readers who had not yet memorized the complete list of binary equivalents for teleprint code.]

Fig. E — Table of function symbols

Function symbol	*Equations*	*Time in beats*
`//`	$\mathbf{H}$ as mag.instr. $\mathbf{C}' = \{\mathbf{C}_+ + 1\}_0^9$ or $\mathbf{C}' = \{\mathbf{C}_+ + 3\}_0^9$	Various
`/E`	$\mathbf{S}' = \mathbf{M}$	5
`/@`	$\mathbf{A}' = \{\mathbf{A}_+ + 2^{40}\mu(\mathbf{S})\}_0^{79}$	24
`/A`	$\mathbf{S}' = \mathbf{M}$, $\mathbf{A}' = \{\mathbf{L}_+\}_0^{79}$	5
`/:`	$\{\mathbf{S}_+\}_0^{19}$ as mag.instr. $\mathbf{C}'$ as for `//`	Various
`/S`	$\mathbf{S}' = \mathbf{L}$	5
`/I`	$\mathbf{L}' = \mathbf{M}$, $\mathbf{M}' = \mathbf{L}$, $(\mathbf{A}' = \mathbf{A}')$	5
`/U`	$\mathbf{S}' = \mathbf{L}$, $\mathbf{A}' = \{\mathbf{M}_+\}_0^{79}$	5
`/¼`	$\mathbf{A}' = \{\mathbf{A}_+ - \mathbf{D}\mathbf{S}_+\}_0^{79}$	9
`/D`	$\mathbf{A}' = \{\mathbf{A}_+ - \mathbf{D}\mathbf{S}_\pm\}_0^{79}$	9
`/R`	$\mathbf{A}' = \{\mathbf{A}_+ + 2^{40}t(\mathbf{S})\}_0^{79}$	5 or 7
`/J`	$\mathbf{A}' = \{\mathbf{A}_+ + 2^{40}\mathbf{S}_+\}_0^{79}$	5
`/N`	$\mathbf{A}' = \{\mathbf{A}_+ + \mathbf{D}\mathbf{S}_+\}_0^{79}$	9
`/F`	$\mathbf{A}' = \{\mathbf{A}_+ + \mathbf{D}\mathbf{S}_\pm\}_0^{79}$	9
`/C`	$\mathbf{D}' = \mathbf{S}_+$	5
`/K`	$\mathbf{D}' = \mathbf{S}_\pm$	5
`/T`	$\mathbf{C}' = \{\mathbf{S}_+ + \mathbf{Q}(\mathbf{C}_+ + 1 - \mathbf{S}_+)\}_0^9$	4
`/Z`	$\{\mathbf{S}'_+\}_0^{19} = \mathbf{H}$, $\{\mathbf{S}'_+\}_{20}^{39} = \{\mathbf{S}_+\}_{20}^{39}$	4
`/L`	Dummy stop	4
`/W`	$\{\mathbf{A}'_+\}_0^{19} = \mathbf{R}$, $\{\mathbf{A}'_+\}_{20}^{79} = \{\mathbf{A}_+\}_{20}^{79}$	24
`/H`	$\mathbf{C}' = \{(\mathbf{C}_+ + 1)\sigma(\mathbf{A}) + (1 - \sigma(\mathbf{A}))\mathbf{S}_+\}_0^9$	4
`/Y`	$\{\mathbf{S}'_+\}_0^{19} = \mathbf{Z}$, $\{\mathbf{S}'_+\}_{20}^{39} = \{\mathbf{S}_+\}_{20}^{39}$	4
`/P`	$\mathbf{C}' = \{\mathbf{S}_+\}_0^9$	4
`/Q`	$\mathbf{C}' = \{\mathbf{C}_+ + 1 + \mathbf{S}_+\}_0^9$	4
`/O`	$\mathbf{C}' = \{\mathbf{C}_+ + 1 + \mathbf{S}_+(1 - \mathbf{Q})\}_0^9$	4
`/B`		4
`/G`	Dummy stop	4
`/"`		4
`/M`	$\mathbf{C}' = \{\mathbf{C}_+ + 1 + \mathbf{S}_+(1 - \sigma(\mathbf{A}))\}_0^9$	4
`/X`		4
`/V`	Hoot	4
`/£`		4

Function symbol	*B-exc*	*Equations*	*Time in beats*
`T/`		$\mathbf{A}' = \{\mathbf{S}_+\}_0^{79}$	5
`TE`		$\mathbf{L}' = \mathbf{S}' = \mathbf{S} \vee \mathbf{L}$, $\mathbf{M}' = \mathbf{M}$, $(\mathbf{A}' = \mathbf{A}')$	5
`T@`		see p. 25	5
`TA`		$\mathbf{S}' = \mathbf{L}$, $\mathbf{A}' = \{0\}_0^{79}$	5
`T:`		$\mathbf{A}' = \{0\}_0^{79}$	5
`TS`		$\mathbf{S}' = \mathbf{L} \vee \mathbf{S}$, $\mathbf{A}' = \{0\}_0^{79}$	5
`TI`		$\mathbf{A}' = \{\mathbf{A}_+ + \mathbf{S}_+\}_0^{79}$	5
`TU`		as `TS`	5
`T¼`		$\mathbf{A}' = \{\mathbf{S}_\pm\}_0^{79}$	5
`TD`		$\mathbf{A}' = \mathbf{A}' \vee \{\mathbf{S}_\pm\}_0^{79}$	5
`TR`		$\mathbf{A}' = \mathbf{A}' \wedge \{\mathbf{S}_\pm\}_0^{79}$	5
`TJ`		$\mathbf{A}' = \mathbf{A}' \not\equiv \{\mathbf{S}_\pm\}_0^{79}$	5
`TN`		$\mathbf{A}' = \{\mathbf{A}_\pm - \mathbf{S}_\pm\}_0^{79}$	5
`TF`		$\mathbf{A}' = \{-\mathbf{S}_\pm\}_0^{79}$	5
`TC`		$\mathbf{A}' = \{\mathbf{A}_\pm + \mathbf{S}_\pm\}_0^{79}$	5
`TK`		$\mathbf{A}' = \{2\mathbf{S}_\pm\}_0^{79}$	5
`TT`		$\mathbf{B}' = \{\mathbf{S}_+\}_0^{19}$, $\mathbf{Q}' = \sigma(\mathbf{B}')$	4
`TZ`		$\{\mathbf{S}'_+\}_0^{19} = \mathbf{B}$, $\{\mathbf{S}'_+\}_{20}^{39} = \{\mathbf{S}_+\}_{20}^{39}$, $\mathbf{Q}' = \sigma(\mathbf{B}')$	4
`TL`		$\mathbf{B}' = \{\mathbf{B}_+ - \mathbf{S}_+\}_0^{19}$, $\mathbf{Q}' = \sigma(\mathbf{B}')$	4
`TW`		as `TL`	4
`TH`			4
`TY`			4
`TP`			4
`TQ`			4
`TO`	*	as `TT`	4
`TB`	*	as `TZ`	4
`TG`	*	as `TL`	4
`T"`	*	[as `TL`]	4
`TM`	*		4
`TX`	*		4
`TV`	*		4
`T£`	*	Official dummy	4

No B line is added to instructions marked "B-exc".
An abbreviated version of this figure is presented in Fig.G.

Definitions	*Normal Equations*
If $\mathbf{S}_+ \neq 0$ then $2^{\mu(\mathbf{S})} \leq \mathbf{S}_+ < 2^{\mu(\mathbf{S})+1}$	$\mathbf{A}' = \mathbf{A}$ (80 digit), $\mathbf{C}' = \{\mathbf{C}_+ + 1\}_0^9$
if $\mathbf{S}_+ = 0$ then $\mu(\mathbf{S}) = 0$	$\mathbf{Q}' = \mathbf{Q}$ (1 digit), $\mathbf{S}' = \mathbf{S}$ (40 digits)
$t(\mathbf{S})$ = number of 1's in $\mathbf{S}$	$\mathbf{B}' = \mathbf{B}$ (20 digits), $\mathbf{D}' = \mathbf{D}$ (number)
$\mathbf{R}$= random digits	[$\mathbf{H}$= hand switches]
$\mathbf{Z}$= clock	[$\mathbf{L} = \{\mathbf{A}\}_0^{39}$, $\mathbf{M} = \{\mathbf{A}\}_{40}^{79}$]

Fig. F — PERM and the Routine Changing Sequence

:		S	C		K
/ / / /	/	/ / / /		/	
E / / /	E	/ / / @		E	
/ / / /	@	/ / / /		@	
@ / / /	A	/ / / :		A	
/ / / /	:	/ / / /		:	
: / / /	S	/ / / ¼		S	
/ / / /	I	/ / / /		I	
¼ / / /	U	/ / / T		U	£ £ / /
/ / / /	¼	/ / / /		¼	H : / C
T / / /	D	£ £ £ £		D	V S / F
/ / / /	R	£ £ £ £		R	L : / J
/ E / /	J	/ / / /		J	M K / A
/ / / /	N	F S / V		N	X K / :
/ @ / /	F	K S / P		F	£ S T /
/ / / /	C	V K T A		C	U K T R
/ : / /	K	J S / L		K	M K / S
/ / / /	T	V S T F		T	M K Q O
/ ¼ / /	Z	C K / H		Z	/ / Q T
/ / / /	L	£ S / :		L	£ S Q B
/ T / /	W	/ E T /		W	F S / P
/ / / /	H	/ A T N		H	Psuedo
/ / E /	Y	M K / S		Y	Accum-
/ / / /	P	Y : / C		P	ulator
/ / @ /	Q	M K / N		Q	
/ / / /	O	V S T J		O	
/ / : /	B	R S T R		B	
/ / / /	G	Y : T N		G	
/ / ¼ /	"	K S / H		"	Special
/ / / /	M	V K T /		M	Working
/ / T /	X	V S / P		X	Space
/ / / /	V			V	
/ / / E	£	Cue		£	

The routine changing sequence is entered by the instruction `NS/P`.

Fig. G — Summary

[In the manuscript, this is a set of tightly packed and somewhat interleaved tables and lists on two sides of a single sheet of paper, intended as a quick reference. I can't duplicate the formatting in TEX, but the content follows.]

```
   PERM

/ / / /   /   / / / /
E / / /   E   / / / @
/ / / /   @   / / / /
@ / / /   A   / / / :
/ / / /   :   / / / /
: / / /   S   / / / ¼
/ / / /   I   / / / /
¼ / / /   U   / / / /
/ / / /   ¼   / / / /
T / / /   D   £ £ £ £
/ / / /   R   £ £ £ £
/ E / /   J   / / / /
/ / / /   N   F S / V
/ @ / /   F   K S / P
/ / / /   C   V K T A
/ : / /   K   J S / L
/ / / /   T
/ ¼ / /   Z
/ / / /   L
/ T / /   W
/ / / /   H
/ / E /   Y
/ / / /   P
/ / @ /   Q
/ / / /   O
/ / : /   B
/ / / /   G
/ / ¼ /   "
/ / / /   M
/ / T /   X
/ / / /   V   —— -
/ / / E   £   |Cue
```

Normal uses of pages

Column	*Page*	*Use*
/,E	0	Instructions
@,A	1	"
:,S	2	PERM
I,U	3	Reserve for PERM or syst. or unsyst. W.S.
¼,D	4	Syst. W.S.
R,J	5	Reserve for Syst. W.S.
N,F	6	Reserve syst. or unsyst. W.S.
C,K	7	Unsyst. W.S.; GK-VK spec.

Magnetic transfers

Digit	*Function*
0-9	Track
10	Reverse
11	Two pages
12	Check
13	Write
14	Special
15	Spare
16-19	Tube

Special mag. funcs.

Character	*Function*
T	Print and Punch
Z	Space
L	Carriage Return
W	Line Feed
H	Figure
Y	Letters
O	Input
B	Check Output
!	Official Dummy

NS/P enters routine changing sequence
KS/P enters hoot
Check numbers are $\{1025([\texttt{/E}]_+ - [\texttt{/A}]_+)\}_{10}^{19}$

Fig. G (Summary), cont.

Input routine examples

`KGA@PQRSTUVW`	gives $[\texttt{GA}]'_s = \texttt{PQRS}$, $[\texttt{"A}]'_s = \texttt{TUVW}$
`JPQRSCTUVWJ`	gives $[\texttt{C}\frac{1}{4}]'_s = \texttt{PQRS}$, $[\texttt{CD}]'_s = \texttt{TUVW}$
`JPQRSCTUVWH`	gives no effect
`XGAT/`	gives $[\texttt{HK}]_+ + 2^{40}[\texttt{PK}]_+ = \{[\texttt{GA}]_+\}_0^{79}$
`"/`$\frac{1}{4}$`I@SP`	gives $[\texttt{/}\frac{1}{4}] = \{625\}_0^{39}$
`"ACG@£`	gives no effect
`"ACG@P`	gives hoot stop
`Z`	enters routine changing sequence Link is [`HK`]
`Y`	enters routine changing sequence Link returns to INPUT
`Q`$\frac{1}{4}$`PIFFLE`	Carriage return, line feed, punches PIFFLE

ENGLISH routine

Letters. Letter shift printing except that `@` means Figures

Figures. Figure shift printing except that `Y` means Letters. `Z,L,W,£` as magnetic special functions

Conventions

(i) Use of PERM
(ii) Link in **L**.
(iii) Entry by cue.
(iv) `/E` and `/A` must be filled in a routine and must not be used as working space.
(v) 5 pages currently available.
(vi) Not more than 8 pages available.
(vii) Instructions in `/`, `E`, `@`, `A`.
(viii) Mention altered lines except on pages 0 and 1 and lines `GK`, `MK` and `VK`
(ix) Leave **B**0 clear.
(x) Mention unusual uses of B lines.
(xi) Use higher numbered B lines first.
(xii) Mention changes of B lines other than **B**7.
(xiii) Tracks 0-15 for permanent routines.
(xiv) Tracks 16-31 for less permanent routines.
(xv) Tracks 32-63 working space — 32 special working space.
(xvi) Disadvantageous features not used.
(xvii) Use only official properties of routines.

Fig. G (Summary), cont.

//	$\mathbf{H}$ as mag. instr.	T/	$A' = \mathbf{S}_+$		
/E	$\mathbf{S}' = \mathbf{M}$	TE	FOUL		
/@	$\mathbf{A}' = \mathbf{A} + 2^{40}\mu(\mathbf{S})$	T@	65th line		
/A	$\mathbf{S}' = \mathbf{M}, \mathbf{A}' = \mathbf{L}_+$	TA	$\mathbf{S}' = \mathbf{L}$, $\mathbf{A}' = 0$		
/:	$\mathbf{S}$as mag. instr.	T:	$\mathbf{A}' = 0$		
/S	$\mathbf{S}' = \mathbf{L}$	TS	FOUL		
/I	$\mathbf{L}' = \mathbf{M}$, $\mathbf{M}' = \mathbf{L}$	TI	$\mathbf{A}' = \mathbf{A} + \mathbf{S}_+$		
/U	$\mathbf{S}' = \mathbf{L}$, $\mathbf{A}' = \mathbf{M}_+$	TU	FOUL		
/¼	$\mathbf{A}' = \mathbf{A} - \mathbf{D}\mathbf{S}_+$	T¼	$\mathbf{A}' = \mathbf{S}_\pm$		
/D	$\mathbf{A}' = \mathbf{A} - \mathbf{D}\mathbf{S}_\pm$	TD	$\mathbf{A}' = \mathbf{A} + \mathbf{S}_+$		
/R	Sideways adder	TD	$\mathbf{A}' = \mathbf{A} \wedge \mathbf{S}_\pm$		
/J	$\mathbf{A}' = \mathbf{A} + 2^{40}\mathbf{S}_+$	TJ	$\mathbf{A}' = \mathbf{A} \not\equiv \mathbf{S}_\pm$		
/N	$\mathbf{A}' = \mathbf{A} + \mathbf{D}\mathbf{S}_+$	TN	$\mathbf{A}' = \mathbf{A} - \mathbf{S}_\pm$		
/F	$\mathbf{A}' = \mathbf{A} + \mathbf{D}\mathbf{S}_\pm$	TF	$\mathbf{A}' = -\mathbf{S}_\pm$		
/C	$\mathbf{D}' = \mathbf{S}_+$	TC	$\mathbf{A}' = \mathbf{A}_+ + \mathbf{S}_\pm$		
/F	$\mathbf{D}' = \mathbf{S}_\pm$	TK	$\mathbf{A}' = 2\mathbf{S}_\pm$		
/T	B-cond. abs. Xfer	TT	$\mathbf{B}' = \mathbf{S}$		
/Z	$\mathbf{S}' = \mathbf{H}$ (20 digits)	TZ	$\mathbf{S}' = \mathbf{B}$	20 digit	
/L	(Dummy stop)	TL	$\mathbf{B}' = \mathbf{B} - \mathbf{S}$	$\mathbf{Q}' = \sigma(\mathbf{B}')$	
/W	$\{\mathbf{A}'\}_0^{19} = \mathbf{R}$	TW	$\mathbf{B}' = \mathbf{B} - \mathbf{S}$		
/H	A-cond. abs. Xfer	TH			
/Y	$\mathbf{S}' = \mathbf{Z}$ (20 digits)	TY			
/P	Uncond. abs. Xfer	TP			
/Q	Uncond. rel. Xfer	TQ			
/O	B-cond. rel. Xfer	TO	$\mathbf{B}' = \mathbf{S}$		
/B		TB	$\mathbf{S}' = \mathbf{B}$		20 digit
/G	Dummy stop	TG	$\mathbf{B}' = \mathbf{B} - \mathbf{S}$		$\mathbf{Q}' = \sigma(\mathbf{B}')$
/"		T"	$\mathbf{B}' = \mathbf{B} - \mathbf{S}$		
/M	B-cond. rel. Xfer	TM			
/X		TX		No B-addition	
/V	Hoot	TV			
/£		T£	Dummy		