Chapter 11

The Pilot ACE[1]

J. H. Wilkinson

Introduction

A machine which was almost identical with the Pilot ACE was first designed by the staff of the Mathematics Division at the suggestion of Dr. H. D. Huskey during his stay at the National Physical Laboratory in 1947. It was based on an earlier design by Dr. A. M. Turing and its principal object was to provide experience in the construction of equipment of this type. It was not intended that it would be used on an extensive programme of computation, but it was hoped that it would give practical experience in the production of subroutines which would serve as a useful guide to the design of a full scale machine. An attempt to build the Pilot Model, during Dr. Huskey's stay, was unsuccessful, but a year later after the formation of an Electronics Section at the NPL a combined team consisting of this section and four members of the Mathematics Division started on the construction of a Pilot Model, the design of which was taken over almost unchanged from the earlier version. The machine first worked, in the sense that it carried out automatically a simple sequence of operations, in May 1950 and by the end of that year it had reached the stage at which a successful Press Demonstration was held. The successful application of the machine to the solution of a number of problems made it apparent that, in spite of its obvious shortcomings, it was capable of being converted into a powerful computer comparable with any then in existence and much faster than most. Accordingly a small programme of modifications was embarked upon early in 1951, but the machine was not functioning satisfactorily again until November of that year. After a month of continuous operation it was transferred from the Electronics Section to Mathematics Division where it has since been in use on a 13-hour day. During its first year of full scale operation it achieved a 65% serviceability figure based on a very strict criterion. Its performance during its second year has so far been considerably better than this.

[1]*Automatic Digital Computation, National Physical Laboratory, Teddington, England*, pp. 5–14, March, 1953.

General description

The Pilot ACE is a serial machine using mercury delay line storage and working at a pulse repetition rate of 1 megacycle/sec. Its high speed store consists of 11 long delay lines each of which stores 32 words of 32 binary digits each, with a corresponding circulation period of 1024 microseconds, 5 short lines storing one word each with a circulation period of 32 microseconds and two delay lines storing two words each. It was inevitable that in the design of a machine originally intended for experimental purposes, overriding consideration should be given to the minimization of equipment rather than to making the machine logically satisfying as a whole. This is reflected to a certain extent in the code adopted for the machine and in its arithmetic facilities, which are in general fairly rudimentary. The design of the machine was also decisively influenced by the attempt to overcome the loss of speed due to the high access time of the long storage units. The machine in fact uses what is usually known as a system of "optimum coding."

Code of Pilot ACE

The Pilot ACE may be said to have a "three-address code" though this form of classification is not particularly appropriate. Each instruction calls for the transfer of information from one of 32 "sources" to one of 32 "destinations" and selects which of eight long delay lines will provide the next instruction. This third address is necessary because consecutive instructions do not occupy consecutive positions but are placed in such relative positions that, in so far as is possible, each instruction emerges during the minor cycle in which the current instruction is completed. An unusual feature of the instructions is that the transfers they describe may last for any number of consecutive minor cycles from one to thirty-two. The instruction word contains three other main elements which are known as the wait number, the timing number and the characteristic which together determine when the transfer starts, when it stops and which instruction in the selected instruction source is the next to be obeyed. The structure of the instruction word is as follows:

Next instruction source	Digits 2–4
Source	Digits 5–9
Destination	Digits 10–14
Characteristic	Digits 15–16
Wait number	Digits 17–21
Timing number	Digits 25–29
Go digit	Digit 32

The remaining digits are spare.

Coding of a problem takes place in two parts, in the first of which only the source, the destination and the period of transfer are specified, the last being a function of the characteristic, wait number and timing number. In the second part, the detailed coding, the other elements are added.

The sources and destinations

Simplest among the sources and destinations are those associated with the short delay lines. The six one-word delay lines are each given numbers and these for reasons associated with the history of the machine are 11, 15, 16, 20, 26 and 27. They are usually referred to as Temporary Stores or TS's because they are used to store temporarily those numbers which are being operated upon most frequently at each stage of a computation. In general TSn has associated with it a source, source n, and a destination, destination n. An instruction of the type

15–16

in the preliminary stage of the coding represents the transfer of a copy of the contents of TS15 via source 15 to TS16 via the destination 16. After it has taken place both stores contain the number originally in TS15. The period of the transfer is not mentioned in the coding because a transfer of more than one minor cycle is irrelevant. Most transfers are for one minor cycle and hence the period of transfer is not specified unless it is greater than one minor cycle. Associated with the TS's are a number of functional sources and destinations. TS16 for instance has two other destinations 17 and 18 associated with it, in addition to destination 16. Any number transferred to destination 17 is added to the contents of TS16 while any number transferred to destination 18 is subtracted from the contents of TS16. TS16 may be said to have some of the functions associated with the accumulator on an orthodox machine. The period of transfer to destinations 17 and 18 is very important. Thus

15–17 (n minor cycles)

has the effect of adding the contents of TS15, n times to the contents of TS16. This prolonged transfer is used in this way to give small multiples (up to 32) of numbers. Similarly, we may have

15–18 (n mc)

The instruction

16–17 (n mc)

is of special significance because it has the effect of adding the content of TS16 to itself for each minor cycle of the transfer, that is it gives multiplication by 2^n or a left shift of n binary places.

TS26 has associated with it a number of functional sources. Source 17 gives the ones complement of the number in TS26, Source 18, the contents divided by 2, and Source 19, the contents multiplied by 2. The instruction

18–26 (n mc)

thus has the effect of dividing the contents of TS26 by 2^n, that is a right shift of n places. Similarly

19–26 (n mc)

gives a left shift of n places.

There are two functional sources which give composite functions of the numbers in TS26 and TS27. These are Source 21 which gives the number

TS26 & TS27

and Source 22 which gives the number

TS26 $\not\equiv$ TS27

There are a number of sources which give constant numbers which are of frequent use in computation. These are Source 23 which gives the number which has a zero everywhere except in the 17th position, usually known as P17, Source 24 which gives P32, Source 25 which gives P1, Source 28 which gives zero and Source 29 which gives a number consisting of 32 consecutive ones. These sources are valuable because they provide numbers with an access time of one minor cycle and are thus almost as useful as several extra TS's.

The use of a number of TS's with the arithmetic facilities distributed among them makes it possible to take advantage of the placing of instructions in appropriate positions in the long storage units so that they emerge as required. The coding of a trivial example will illustrate the uses of the TS's and their associated sources. It is required to build up the successive natural numbers, their squares and their cubes simultaneously. It is natural to store the values in TS's and we may suppose TS15 contains n, TS20, n^2 and TS26, n^3.

	Instruction	*Description*	
1.	28–15	zero to TS15 *i.e.* 0	These 3 instructions set the
2.	28–20	zero to TS20 *i.e.* 0^2	initial values
3.	28–26	zero to TS26 *i.e.* 0^3	
	—		
4.	26–16	TS16 contains n^3	
5.	20–17 (3mc)	TS16 contains $n^3 + 3n^2$	
6.	15–17 (3mc)	TS16 contains $n^3 + 3n^2 + 3n$	
7.	25–17	TS16 contains $n^3 + 3n^2 + 3n + 1$	
8.	16–26	TS26 contains $(n + 1)^3$	
9.	20–16	TS16 contains n^2	
10.	15–17 (2mc)	TS16 contains $n^2 + 2n$	
11.	25–17	TS16 contains $n^2 + 2n + 1$	
12.	16–20	TS20 contains $(n + 1)^2$	
13.	15–16	TS16 contains n	
14.	25–17	TS16 contains (n + 1)	
15.	16–15	TS15 contains (n + 1)	Next instruction (4)

The instructions (1) to (3) set the initial conditions. The instruction (4) — (15) have the effect of changing the contents of 15, 20, 26 from n, n^2, n^3 to (n + 1), $(n + 1)^2$, $(n + 1)^3$. As remarked earlier, each instruction selects the next instruction and here instruction (15) selects instruction (4) as the next instruction. In the preliminary coding this is usually denoted by using an arrow; it must be catered for in the detailed coding by the correct choice of the timing number, as will be shown below.

The branching of a programme is achieved by the use of two destinations, destination 24 and destination 25. If a transfer is made from any source to destination 24 then the next instruction is one or other of two according as the number transferred is positive or negative. Similarly if a transfer is made to destination 25 then the next instruction is one or other of two according as the number transferred is zero or non-zero. In the preliminary coding the bifurcation is denoted by the use of arrows, thus:

$$\begin{tikzcd} & \text{15–24} \arrow[dl] \arrow[dr] & \\ +\text{ve} & & -\text{ve} \end{tikzcd}$$

In the detailed coding the effect is that if the number transferred to destination 24 is negative then the timing number is increased by 1. Similarly for destination 25; the two possible next instructions are consecutive in the store.

The two double word stores are numbered DS12 and DS14. DS12 has only source 12 and destination 12 associated with it, but DS14 has, in addition to source 14 and destination 14, a number of functional sources and destinations. Source 13 gives the contents of DS14 divided by 2, while transfers to destination 13 have the effect of adding the numbers transferred to DS14. In specifying transfers from, and to, the double length stores, the time of the transfer must be specified, *i.e.* whether it takes place in an even or an odd minor cycle or both. Thus the transfer

12–14 (odd minor cycle) usually written
12–14 (o)

represents the transfer of the word in the odd positions of DS12 to the odd position in DS14 while

12–14 (2 minor cycles)

represents the transfer of both words in 12 to the corresponding positions in 14. The operation

13–14 (2n)

gives us a method of shifting the contents of TS14 n places to the right while

14–13 (2n)

produces a shift of n places to the left.

The machine is not equipped with a fully automatic multiplier. To multiply two numbers, a and b, together, a must be sent to TS20, b to DS14 odd, zero to DS14 even and a transfer (source irrelevant) made to destination 19. The product is then produced in DS14 in 2 milliseconds, but a and b are treated as positive numbers. Corrections must be made to the answer if a and b are signed numbers. To make multiplication fast, it has been made possible to perform other operations while multiplication is proceeding. Thus the corrections necessary if a and b are signed numbers may be built up in TS16 during multiplication, and signed multiplication takes only a little over two millisecs. It is, of course, therefore, a subroutine but a very fast one. The amount of equipment associated with the multiplier is very small. The main part of the store consists of the long storage units known as DL1, DL2, . . . , DL11. Each of these has a source and a destination with the same number as the DL number. The words in each DL are numbered 0 to 31 and the nth word in DLM is usually denoted by DLM_n. Transfers to and from long lines in the preliminary coding are denoted thus:

8_n- 16	(transfer nth word of DL8 to TS16)
8_{m-n}-17	(add all the words from 8_m to 8_n *i.e.* n − m + 1 consecutive words of $DL8^m$ to TS16)

Detailed coding

In the second stage of the coding the true instruction words are derived from the preliminary coding. This is a fairly automatic process and recent experience has shown that it can be carried out satisfactorily by quite junior staff. The timing of each instruction is given relative to the position of that instruction in the store. This is an incidental feature of the code which arose from the attempts to minimize equipment. It would be dropped in any future machine in favour of an absolute timing system. If an instruction occupies position m in a DL and has a wait number W and timing number T then the transfer always begins in minor cycle (m + W + 2) and the next instruction is always in minor cycle (m + T + 2) of the selected next instruction source. The period of transfer depends on the value of the characteristic. If the characteristic is zero then the transfer lasts for the whole period from (m + W + 2) to (m + T + 2), that is (T − W + 1) minor cycles. If the characteristic is one, then the transfer is for one minor cycle, that is minor cycle (m + W + 2). If the characteristic is three then the transfer is for two minor cycles (m + W + 2) and (m + W + 3). The characteristic value, two, is not used. The characteristic value zero gives a prolonged transfer which is peculiar to the Pilot ACE. The characteristics 1 and 3 are analogous to the facility on EDSAC whereby full length or ½-length words may be transferred. On the Pilot ACE we transfer single or double length words. This facility is invaluable for double length, floating and complex arithmetic. In the above definitions the numbers (m + W + 2) etc. are to be interpreted modulo 32. In general, timing and wait numbers are simpler than they appear from the definitions because they are very frequently both zero, corresponding to a transfer for one minor cycle. The detailed coding of the problem given earlier will illustrate the procedure. All the instructions are in DL1 so that the next instruction source is always one. The key to the headings in the following table is:

m.c.	Minor cycle position of instructions in DL1
N.I.S.	Next instruction source
S	Source
D	Destination
C	Characteristic
W	Wait number
T	Timing number

The last column gives the position of the next instruction in DL1; it is given by (m + T + 2). The first 4 instructions occupy minor cycles, 0, 2 and 4, 6 and each takes two minor cycles, and gives a transfer for one minor cycle only. The next instruction occupies minor cycle number 8 and it requires a transfer lasting 3 minor cycles. The simplest and fastest way of getting this is to have W = 0 and T = 2 giving a transfer of (2 − 0 + 1) minor cycles. The next instruction is in position (8 + 2 + 2), that is minor cycle 12, and so on. When we reach the instruction in minor cycle 31, viz. 25–17, a transfer for one minor cycle is required. The simplest way is to have W = 0 T = 0 and this makes the next instruction occupy position (31 + 0 + 2) i.e. position 33 which is position 1. If position 1 had been already occupied, a value of T could have been chosen in order to land in an unoccupied position. In order to ensure that a transfer of one minor cycle only took place, the characteristic could have been made 1. It should be appreciated that the choice of C, W and T is far from unique. Whenever possible T = 0 and W = 0 are chosen because this gives the highest speed of operation besides being simplest. The instruction occupying position 1 is of special interest because this is the last instruction of the cycle needed to build up a square and cube and it must select as its next instruction the first of the cycle, which is, in position number 6. This is achieved by making T = 3 (giving the next instruction in m.c. 1 + 3 + 2 = 6). This incidentally gives a transfer lasting four minor cycles but since it is a transfer from one TS to another and no functional source or destination is in use, the prolonged transfer produces no harmful effect. If a prolonged transfer had to be avoided then the characteristic could be taken as 1. It is seldom necessary to use any characteristic other than zero for transfers to and from TS's but when transfers are made to and from DL's, characteristic values of 1 or 3 are almost universal. All 12 instructions which comprise the repeated cycle of the computation take a total time of one major cycle exactly (32 minor cycles) the last instruction of the cycle having been specially designed to get back to the beginning of the cycle. This is in contrast to the position in a machine not using optimum coding, where 12 major cycles would be necessary quite apart from the fact that the multiplications by factors of 3 and 2, each of which uses one instruction, would normally need more than one instruction if a prolonged transfer were not available. Figure 1 gives a simplified diagram of the machine. The sequence of events in obeying the instruction

N	S		D	C	W	T
2	16	–	2C	0	8	10

occupying $DL1_2$ for example is as follows. Starting from the time when the last instruction was completed, the instruction from

Minor cycle position of instructions in DL1	*Next instruction source*	*Source*	*Destination*	*Charac-teristic*	*Wait no.*	*Timing no.*	*Minor cycle position of next instruction*
0	1	28	15	0	0	0	(2)
1	1	16	15	0	0	3	(6)
2	1	28	20	0	0	0	(4)
3							
4	1	28	16	0	0	0	(6)
5							
6	1	26	16	0	0	0	(8)
7							
8	1	20	17	0	0	2	(12)
9							
10							
11							
12	1	15	17	0	0	2	(16)
13							
14							
15							
16	1	25	17	0	0	0	(18)
17							
18	1	16	26	0	0	0	(20)
19							
20	1	20	16	0	0	0	(22)
21							
22	1	15	17	0	0	1	(25)
23							
24							
25	1	25	17	0	0	0	(27)
26							
27	1	16	20	0	0	0	(29)
28							
29	1	15	16	0	0	0	(31)
30							
31	1	25	17	0	0	0	(1)

$DL1_2$ will have passed into the special TS marked TS COUNT during minor cycle number 2. By the end of minor cycle number 3, S switch number 16 will be over and also N switch number 2. The contents of TS16 will be passing into HIGHWAY and those of DL2 into INSTRUCTION HIGHWAY. At the beginning of minor cycle number 12 (*i.e.* 2 + 8 + 2), D switch number 20 will go over, and TS20 will stop recirculating and the number on the HIGHWAY will pass into TS20. The transfer will continue until minor cycle 14 (*i.e.* 2 + 10 + 2) when the D switch number 20 will switch back. At the beginning of minor cycle 14, the switch X on COUNT will go over and the number on INSTRUCTION HIGHWAY during this minor cycle, $DL2_{14}$, will pass into COUNT. At the end of minor cycle 14, the X switch will close again and $DL2_{14}$ will be trapped in COUNT. The cycle of events is now complete. COUNT is associated with a counter and it is this counter which determines from the wait, timing, and characteristic numbers of the trapped instruction, when the D and X switches go over and back.

Input and output

The only part of the instruction word not described is the GO digit. If the GO digit is a one, the instruction is carried out at high speed, but if it is a zero the machine stops and does not proceed until a manual switch is operated. The GO digit is omitted in strategic instructions when a programme is being tested. It also

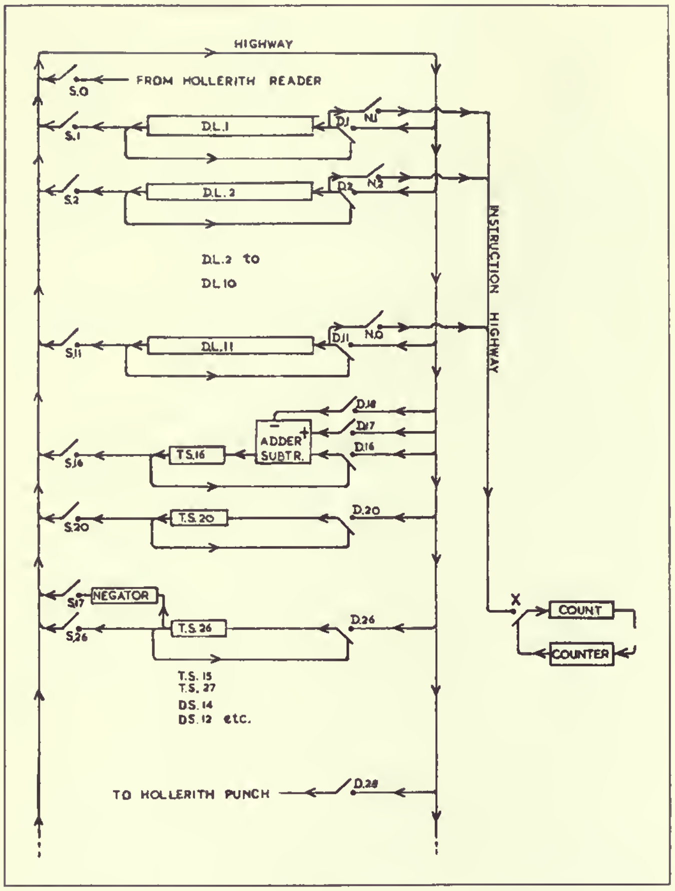

Fig. 1. Simplified diagram showing some sources, destinations, and next-instruction sources.

serves a further purpose in synchronising the input and output facilities with the high speed computer. Input on the machine is by means of Hollerith punched cards. When cards are passed through the reader the numbers on the card may be read row by row as each passes under a set of 32 reading brushes. When a row of a card is under the reading brushes, the number punched on that row, regarded as a number of 32 binary digits, is available on source 0. In order to make certain that reading takes place when a row is in position and not between rows, transfers from source 0, have the GO digit omitted and it is arranged that the Hollerith reader has the same effect as operating the manual switch each time a row comes into position. The passage of a card through the reader is called for by a transfer from any source to destination 31. No transfer of information from the card takes place unless the appropriate instruction using source 0 is obeyed during the passage of the card. Output on the machine is also provided by a Hollerith punch. The passage of a card through the punch is called for by a transfer from any source to destination 30. While a card is passing through the punch a 32 digit number may be punched on each row by a transfer to destination 28. Again synchronisation is ensured by omitting the GO digit in instructions calling for a transfer to destination 28, and arranging that the Hollerith punch effectively operates the manual switch as each row comes into position. The reader feeds cards at the rate of 200 cards per minute and the punch, at the rate of 100 cards per minute. The speed of input for binary digits is $200 \times 32 \times 12$ per minute or 1280 per second. The output speed is 640 digits per second. Data may be fed in and out in decimal, but it then requires conversion subroutines. The computation involved in the conversion is done between the rows of the card and up to 30 decimal digits per card may be translated. This speed of conversion is only possible because of the use of optimum coding. The facility for carrying out computation between rows of cards is used extensively particularly in linear algebra when matrices exceeding the storage capacity of the machine are involved. The matrices are stored on cards in binary form with one number on each of the 12 rows of each card, all the computation being done either between rows when reading or when punching. Times comparable with those possible with the matrices stored in the memory are often achieved in this way, when the computation uses a high percentage of the available time between rows. Up to 80% of this time may be safely used.

Initial input

The initial input of instructions is achieved by choosing destination 0 in a special manner. When a transfer is made to destination 0, then the instruction transferred becomes the next to be obeyed and the next instruction source is ignored. Source 0 has already been chosen specially since it is provided from a row of a card. The instruction consisting of zeros has the effect of injecting the instruction punched on a row of a card into the machine as the next to be obeyed. The machine is started by clearing the store and starting the Hollerith reader which contains cards punched with appropriate instructions. Destination 0 is also used when an instruction is built up in an arithmetic unit ready to be obeyed.

Miscellaneous sources and destinations

Destination 29 controls a buzzer. If a non-zero number is transferred to destination 29 the buzzer sounds.

Source 30 is used to indicate when the last row of a card is in position in the reader or punch. This source gives a non-zero number only when a last row is in position. The operation of the arithmetic facilities on DS14 may be modified by a transfer to destination 23. If a transfer with an odd characteristic is made from any source to destination 23 then, from then on, DS14 behaves as though it were two single length accumulators in series. This means that carries are suppressed at the end of each of the single words. This condition persists until a transfer is made to destination 23 using an even characteristic, when DS14 behaves as an accumulator for double length numbers with their least significant parts in even minor cycles and more significant parts in odd minor cycles.

The operation TS20 is modified by transfers to destination 21. If a transfer with an odd characteristic is made to destination 21 then TS20 ceases to have an independent existence and from then on is fed continuously from DL10. Source 20 then gives the contents of DL10 one minor cycle later than from source 10. TS20 reverts to its former condition when a transfer with an even characteristic is made to destination 21. The facility is used to move the 32 words in DL10 round one position so that the word in minor cycle n is available in minor cycle (n + 1).

Assessment of optimum coding

A detailed assessment of the value of optimum coding is by no means simple. Roughly speaking, subroutines are on an average about 4 or 5 times as fast as on an orthodox machine using the same pulse repetition rate. In main tables a somewhat lower factor is usually achieved. The factor of 4 or 5 would be exceeded if less of the advantage given by optimum coding were used to overcome disadvantages due to the rudimentary nature of the arithmetic facilities on Pilot ACE. Even so, the bald statement of the average ratio of speeds does not do full justice to the value of optimum coding on the Pilot ACE. Its value springs as much from the fact that it has made possible the programmes in which computing is done between the rows of cards and also the high output speed of decimal numbers. The binary decimal conversion routines for punching out several decimal numbers simultaneously on a card and also decimal-binary conversion routines for reading several numbers, achieve a ratio of something like 14 to 1, and on a machine which is being used extensively for scientific computation on a commercial basis this is of immense importance.

Future programme

Engineered versions of the Pilot Model are now under construction by the English Electric Company. These machines will be similar to the Pilot Model but will have a little more high-speed store, an automatic divider, two quadruple length stores and a subtractive input on the double length accumulator besides several minor modifications including a rationalization of the numbering of the stores! In addition a magnetic drum intermediate store with the equivalent of 32DL's storage capacity will be added. A full scale machine will probably soon be under development employing a 4 address code. Typical instructions will be of the form

$$A \pm B \quad C$$

and will select the next source of instruction. This code is more economical in instruction storage space and since all single word stores will then become complete accumulators with all facilities except multiplication on them, it will be possible to take much fuller advantage of optimum coding.

Sources, destination and next instruction sources

Sources	*Destinations*	*Next instr. sources*
0. Input	0. INSTRUCTION	0. DL11
1. DL1	1. DL1	1. DL1
2. DL2	2. DL2	2. DL2
3. DL3	3. DL3	3. DL3
4. DL4	4. DL4	4. DL4
5. DL5	5. DL5	5. DL5
6. DL6	6. DL6	6. DL6
7. DL7	7. DL7	7. DL7
8. DL8	8. DL8	
9. DL9	9. DL9	
10. DL10	10. DL10	
11. DL11	11. DL11	
12. DS12	12. DS12	
13. DS14 + 2	13. DS14 add	
14. DS14	14. DS14	
15. TS15	15. TS15	
16. TS16	16. TS16	
17. TS26	17. TS16 add	
18. TS26 ÷ 2	18. TS16 subtract	
19. TS26 × 2	19.† MULTIPLY	
20. TS20	20. TS20	
21. TS26 & TS27	21. Modifies Source 20	
22. TS26 ≢ TS27	22. —	
23. P17	23. Modifies Source 13, Destination 13	
24. P32	24. DISCRIMINATE on sign	
25. P1	25. DISCRIMINATE on zero	
26. TS26	26. TS26	
27. TS27	27. TS27	
28. Zero	28. Output	
29. Ones	29. BUZZER	
30. Last row of card	30.† PUNCH	
31. —	31.† READ	

† Independent of source used.

References

WilkJ53; TuriS59