Solution of Simultaneous Linear Equations using a Magnetic-Tape Store

By D. W. Barron and H. P. F. Swinnerton-Dyer

This paper discusses the special problems involved in solving a set of simultaneous linear equations when magnetic tape is being used for auxiliary storage. An account is given of a subroutine written to carry out this process on EDSAC 2. A description of EDSAC 2 and its auxiliary magnetic-tape equipment is given as an Appendix.

1. Introduction

The solution of a set of simultaneous linear equations on a digital computer is a standard problem. Matrix division and inversion can be regarded as special cases, in which the equations have several right-hand sides. It often happens that many of the coefficients vanish, and that an iterative process is most convenient; but it is necessary also to have a process which is generally applicable, and of these the most used is the method of Gaussian elimination (see Section 2).

The first library subroutine ever written for EDSAC 2 was for this process; it is 50 orders long and will solve 31 equations (as many as can be fitted in the main store) in about four seconds. Recently we have found the need to solve sets of equations larger than this, and for these, auxiliary magnetic-tape storage has to be used. The problems of access involved lead to some complications of technique and modifications of the usual procedure; the purpose of our paper is to discuss these.

We are concerned with the solution of equations as a subroutine, not as a complete program. If we merely need to solve a given set of equations it is trivial to write the input and output orders needed; but our experience suggests that the solution of simultaneous equations usually arises as part of a larger task.

There is a fundamental distinction, according as the equations are stored by rows or by columns. In Section 3 we discuss the technique used with row storage, and give a detailed description of the EDSAC 2 subroutine we have written to carry out the process. In Section 4 we give a theoretical discussion of the method used with column storage. It appears that, other things being equal, column storage leads to a faster and more accurate solution, at the cost of some additional effort in the programming; we have as yet no practical experience of this, since we only realized it after programming the method involving row storage.

The process we use involves a method of interchange different from the usual one. We have therefore given in Section 2 a brief discussion of Gaussian elimination and the precautions needed when using it. In view of the possibility of working in fixed point, we have also said something about scaling. We have mentioned also a method of estimating the error, which serves, at the same time, as a check against mistakes made by the machine.

The description of the subroutine assumes some knowledge of EDSAC 2. Unfortunately, there is no easily accessible account in print; we have therefore included a brief description of EDSAC 2, and in particular of the auxiliary magnetic-tape equipment, as an Appendix. This may be of interest also to readers who prefer to skip the body of the paper.

2. Gaussian Elimination

2.1. *Pivotal Reduction*

We consider the solution of a set of n equations with one right-hand side, which we write in the usual form

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}.$$

The first part of the process is to reduce the equations to triangular form by operations on rows, the basic operation being the subtracting from one row of a multiple of another. The kth stage consists of eliminating x_k from equations $k+1, \ldots, n$ by subtracting appropriate multiples of the kth equation. Thus, if we use undashed letters to denote the values of elements before, and dashed letters the values after, this stage has been carried out, we have

$$a'_{ij} = a_{ij} - a_{ik}a_{kj}/a_{kk},$$
$$b'_i = b_i - a_{ik}b_k/a_{kk},$$

for $i = k+1, \ldots, n$ and $j = k, \ldots, n$. This process is the *pivotal reduction*; we shall call the kth equation the *pivotal equation* and the element a_{kk} the *pivot*.

If we perform the operation above for $k = 1, 2, \ldots, n-1$, the resulting equations will be in triangular form, and the x_i can now be obtained, in reverse order, by the back substitution

$$x_i = \left\{ b_i - \sum_{j=i+1}^{n} a_{ij}x_j \right\} \Big/ a_{ii} \qquad (i = n, n-1, \ldots, 1).$$

The process described above is unsatisfactory, since large errors will be introduced if any multiplier is large —that is, if a pivot is unnecessarily small; and it will break down entirely if a pivot vanishes. The errors may arise in two ways.*

* Normally the error arising from (ii) is entirely masked by that from (i), and in an error analysis of the elimination process above, (ii) would not produce additional avoidable error. However, we are concerned to find a similar process which is satisfactory: now (ii) must be mentioned because it would be a source of avoidable error if we only cured the troubles caused by (i).

(i) In the elimination the kth equation has to be multiplied by a large factor before being subtracted from subsequent ones; thus significant figures are lost in the later equations.

(ii) In the back substitution $b_k - \Sigma a_{kj}x_j$ is a small number formed as the difference of much larger ones. Thus it is not known to the full precision available in the machine (nor usually is a_{kk}); this produces an error in x_k and consequently in the $x_i (i < k)$ calculated from it.

Normally, both these difficulties are avoided by interchanging rows at each stage, so that the largest relevant element in the kth column comes into the pivot position; the multiplying factor a_{ik}/a_{kk} is then always less than one. With magnetic-tape storage, however, such a search and interchange may present difficulties—it certainly could not be combined with the speeded-up process of Section 3. Fortunately we can settle for a little less: it is enough to ensure in the elimination

(i) that no equation is multiplied by a factor greater than unity before being subtracted from another, and

(ii) that the element a'_{kk} in the pivotal position at the end of the kth stage is the largest of those available.

2.2 *Scaling*

If the solution of the equations is to be done in fixed point, it is necessary to scale the coefficients and right-hand sides. Since one may have to shift right during the calculations, it is natural to do the scaling according to the way the coefficients are stored, scaling by columns if they are stored in columns and by rows if they are stored in rows. Scaling by columns is equivalent to multiplying the variables by constants; thus it will have little effect on the process of solution, and is obviously always legitimate. In scaling by rows there is one preliminary danger: it may happen that the coefficients in some column are all small, simply because the corresponding variable is large. If no precautions are taken, row scaling will mean that these numbers, and others derived from them, contain a needlessly large relative error. Despite the extra inconvenience, in the case of row storage it is therefore wise to scale initially by columns.

Once this has been done, subsequent row scaling presents no special difficulties. If an equation with all coefficients small appears in the course of the calculations, it is in fact advantageous to shift it left as far as possible, even though this may lead to the use of it as a pivotal equation. The errors which this seems to introduce are inherent in the equations (which are ill-conditioned); on the other hand, we do diminish the effect of subsequent round-off errors.

2.3 *Checks*

The classical method of checking is to form residuals by substituting the solution obtained back into the original equations. This is not wholly satisfactory, for two reasons. First, we may have obtained a solution of a near set of equations (which is as much as we are entitled to in practice) even though the residuals are not small. Second, even if the residuals are small we cannot deduce that the answers obtained contain a large number of significant figures.

A better check—both for mistakes of calculation and as a measure of the cumulative effects of round-off error—is to work with an extra right-hand side which is the sum of all the left-hand and right-hand sides. The solution for this right-hand side should be

$$1 + \text{sum of all the other solutions.}$$

The inaccuracy in this is a measure of the error in the other solutions. (This is essentially the check used in hand calculation; but then, since mistakes are more important than round-off errors, it is applied to each step.) If some columns are very small or very large, suitable multiples of them should be taken.

3. Technique Using Row Storage

3.1 *Method*

We now describe the technique used for the solution of n equations with m right-hand sides when the equations, including their right-hand sides, are stored in rows in successive blocks of the magnetic tape—that is in n blocks of $m + n$ words each.

The process is best understood by considering first a simplified version, as follows.

(i) Read equation 1 to the main store.

(ii) Read equation 2 to the main store, and interchange with equation 1 if $|a_{11}| < |a_{21}|$.

(iii) Eliminate x_1 from equation 2 by subtracting from it a_{21}/a_{11} times equation 1.

(iv) Rewrite processed equation 2 in its own block.

(v) Repeat steps (ii), (iii), and (iv) for equations $3, \ldots, n$.

(vi) Return to block 1 on tape and write back the eventual equation 1 (after all the interchanges).

We have now eliminated x_1 from equations 2 to n, by means of equation 1 and can repeat the process for x_2 in equations 2 to n (for which the tape is correctly positioned), and so on. Although there is no search for the largest pivot, this process is satisfactory since it satisfies the conditions laid down at the end of Section 2.1. When the equations have been triangulated, the back substitution proceeds in the usual way, reading down successively equations n, $n-1, \ldots, 1$ and forming the values of the variables in the main store.

This process requires n sweeps through the tape (each sweep one block shorter than the previous one) and is therefore very slow. It is possible to arrange that the same arithmetic processes are fitted into fewer sweeps of the tape, by eliminating several variables in one sweep. Essentially this depends on the fact that we can eliminate x_2 from equation 3 as soon we have eliminated x_1 from equations 2 and 3, and so on. Thus we may produce a revised version as follows.

(i) Read equation 1.
(ii) Read equation 2, exchange with equation 1 if desirable, and eliminate x_1 from equation 2.
(iii) Read equation 3, exchange with equation 1 if desirable and then eliminate x_1, exchange result with equation 2 if desirable and eliminate x_2.
(iv) Repeat with equations 4, 5, . . . so long as there is room in the main store.
(v) Suppose the last equation that can be read in this way is equation r. Rewrite it, after processing, in its own block; then read equation $r + 1$, eliminate $x_1, x_2, \ldots, x_{r-1}$, with interchange if necessary, and rewrite it in its block. Then do the same for equations $r + 2, \ldots, n$.
(vi) Return to the head of the tape and copy the processed equations $1, 2, \ldots, r - 1$ into their own blocks.

We are now (after one sweep) in the same situation as after $r - 1$ sweeps of the previous technique; and we can continue the triangulation in the same way, starting with equation r. There is, however, a further refinement. To get the maximum speed in this process, we must pack as many equations as possible into the main store, and it is inexpedient to waste space on the trivial zeros which appear in the front of any equation from which variables have been eliminated. Thus, whenever a variable is eliminated from an equation, that equation is moved up one place in the store, and only the non-trivial part of an equation is written back on the magnetic tape. If an equation has to be written back on to the tape as soon as it has been processed [stage (v) above], then the tape is moved back into position during the elimination; even so, if less than 10 variables are being eliminated, the arithmetic is finished before the tape is in position.

After the triangulation is finished, the back substitution is easy. The only possible difficulty is that we may have so many right-hand sides that not all the values of all the variables can be held at once in the store. In this case we deal with as many right-hand sides at once as we can. It is natural to write up the values of the variables in the positions within the blocks previously occupied by the right-hand sides; in the last sweep through of the back substitution, the triangulated equations can be deleted, leaving the solutions as blocks of m words on the tape.

3.2 *Organization of the EDSAC 2 Program*

From the remarks above it is clear that speed of solution depends on having as much of the main store as possible for working space. To this end the entire contents of the main store, as they stand at the start of the process, are written on the magnetic tape, and only the orders for triangulation or back substitution brought back into the main store. These orders are placed at the beginning of the store, so that the working space is a continuous block.

The subroutine is in four main sections: these are concerned with initial preparation, control of magnetic tape, triangulation, and back substitution. There are two entry points, A and B. The user first enters at A, specifying m and n by the contents of the modifier registers, and the tape number by a program parameter. After checks for obvious inconsistencies (no right-hand sides, too many equations, non-existent tape, etc.) the subroutine plants those variable orders which depend on m and n, moves the selected tape back to its beginning and re-marks it with blocks as follows:

one block of 70 words for the triangulation and magnetic-tape control subroutines,
one block of 1,019 words into which the contents of the main store will be dumped (for engineering reasons there are some registers whose contents cannot be dumped),
n blocks of $n + m$ words for the equations, and
one block of 44 words for the back-substitution program.

The magnetic-tape control, triangulation and back-substitution programs are then written in the appropriate blocks, after which control returns to the master routine.

The user now writes his equations and right-hand sides in blocks 1 to n using the magnetic-tape control subroutine, which at this moment is set for blocks of $m + n$ words. When the equations are written, the user re-enters the subroutine at entry B, whereupon the sequence of events is as follows:

(i) The contents of the entire store (less five words) are dumped on tape.
(ii) The triangulation and magnetic-tape control sections are read from the tape into their standard position.
(iii) Triangulation is carried out.
(iv) The back-substitution section is read in place of the triangulation and obeyed, and the answers are written back on the tape.
(v) The former contents of the entire store are read back from the tape, the last few orders for this being placed in those registers whose content was not dumped in (i) and which will not therefore be overwritten.
(vi) The magnetic-tape control subroutine is set to read blocks of m words, and control is returned to the main program.

If the sum-check fails in (v) the machine will stop, but the answers obtained will probably be correct. The reader will notice that the various subroutines are so placed on the tape that when any of them is wanted it is in the next block to be read; thus no time is wasted in unnecessary traversing.

There is an entry A′ which may be used instead of A, whose effect is to have the solutions finally written up as m blocks of n words. Using this involves some changes in the description, but no differences of principle.

It will be seen that neither of the checks mentioned in Section 2.3 is built into the program. To take a second

copy of the equations involves two magnetic tapes, or a great deal of wasted time during the writing up of the equations. The extra column for a sum check, also, is more easily formed when the equations are written up in the main program, and its interpretation varies so much that it is best left to the individual user.

We have worked throughout in floating point. It may be that fixed-point work, with scaling as in Section 2.2, would be more accurate, but the extra labour of programming would be considerable, and the scaling by columns would involve spending time on an extra sweep through the tape.

The number of equations that the program will solve is limited by the need to hold at least two equations simultaneously in the main store during the triangulation. The bound imposed by this is $m + n \leqslant 474$.

In addition, this number of equations would fill one standard (1,800 ft) reel of magnetic tape, and the fifty-odd hours taken in the calculation might be thought excessive.

3.3 *Speed*

The only satisfactory way to time a program of this nature is with a stop-watch. We find, for example, that to solve 100 equations with 1 right-hand side takes nearly 7 minutes. However, it is interesting to compare this with a theoretical estimate. We have about 950 registers of working space, and so the number of sweeps through the tape should be about

$$(\tfrac{1}{2}n^2 + mn)/(950 - m - n),$$

plus an allowance for wastage because we can only read complete blocks. In one sweep each block is covered four times (read—go back—write—go back at end). There are n blocks of $m + n$ words, with a factor of $\frac{4}{3}$ because of the gaps left for safety between blocks, and the tape is traversed at 1,250 words per second. Assuming that the time is dominated by magnetic-tape movements, and that the back substitution takes a further $mn/950$ traverses, this suggests a time of about

$$2n^2(n + m)(n + 4m)\mu\text{sec}.$$

In fact this is about half the truth; the discrepancy is probably in the time taken to stop and reverse the magnetic tape after most blocks. However, the functional form of the answer should be correct, and suggests that the actual time is about $4n^2(n + m)(n + 4m)\mu$sec.

4. Technique Using Column Storage

We now consider how the calculation should be carried out if the equations are stored by columns on the tape. As before, we start by giving a simplified procedure in which only one variable is eliminated in each sweep of the magnetic tape. Since all the coefficients in a column are available together, we can now do the interchanges in the classical way; thus the sequence is as follows.

(i) Read column 1 into the main store, interchange the largest element with the first, keeping a record of the interchange, form the multipliers a_{i1}/a_{11} and write the revised column 1 back on the tape.

(ii) Read column 2 into the main store, make the appropriate interchange, form a new column 2 by

$$a'_{i2} = a_{i2} - \frac{a_{i1}}{a_{11}}a_{i2} \quad (i > 1)$$

and write the new column back on the magnetic tape.

(iii) Repeat (ii) for columns 3, 4, . . ., $m + n$.

These operations eliminate x_1 from all equations except the first; and the rest of the triangulation goes on in a similar way. In contrast with the case of Section 3, the blocks remain of constant length: thus when eliminating x_k, we read down a block of n words and start processing at the kth word of the block.

As in the case of row storage, time can be saved by eliminating several variables at once. In this case, when eliminating x_k, we can write the kth column on the tape as soon as it has been processed; only the n–k multipliers deduced from it need be held in the main store, together with a note of the interchange involved. Thus only $(\frac{1}{2}n^2)/(950 - n)$ sweeps through the tape should be needed in the triangulation; this represents a large saving in time (as against the method using row storage) unless m is small.

A further advantage of column storage appears if the arithmetic is done with fixed-point numbers—which is possible even if the equations are initially given in floating point. This gives greater accuracy, because fixed-point numbers are held to greater precision in the machine. The interchanges involved in processing a column during one sweep can all be done before the arithmetic is started. This arithmetic is equivalent to the formation of a scalar product and can be done double-length, with a further drastic reduction of the round-off error. The initial conversion from floating- to fixed-point numbers can be done during the first normal sweep through the tape, and needs only a few extra orders; it does, however, increase the block lengths from n to $n + 1$, since the scaling factor has to be retained.

We have not yet had occasion to program this process. It appears to be more efficient than that using row storage, but the gain in efficiency is at the cost of much greater complication in the organization of the program. This comes from the red-tape difficulties: the counts are more elaborate and do not fit together so tidily. The extra trouble of conversion to fixed point (in a machine having both facilities) is slight, and is certainly worth while.

Mr. Wilkinson has kindly informed us of a program on ACE which uses a similar method. There each column is processed completely the first time it is read into the main store: thus scalar products can be accumulated and round-off error is minimized. This process requires n scans of the matrix, which is not serious with drum storage but would be very slow on magnetic tape.

Acknowledgements

We wish to thank Dr. M. V. Wilkes, F.R.S., Director of the University Mathematical Laboratory, for permission to use EDSAC 2, and J. H. Wilkinson for much helpful criticism. D.W.B. is indebted to the Department of Scientific and Industrial Research for a maintenance grant covering the period during which this work was carried out.

Appendix: EDSAC 2

1. General Description

EDSAC 2 is a parallel machine working in the binary system. The high-speed store is composed of ferrite cores and has a capacity of 1,024 words of 41 digits each. This is called the *free store* since its contents can be freely changed by the programmer, in contrast to the *reserved store*, also composed of ferrite cores, which contains permanently wired-in subroutines, together with the working space they need (see Renwick, 1957). A single-address order code is used, with two orders per word. There are two *modifier registers* (index registers, or B-registers), and the *sequence-control register* (or instruction counter) can also be treated as a modifier so that orders with relative addresses can be written in a form which is independent of their position in the store. The arithmetic unit provides full fixed-point and floating-point facilities, with provision for conversion between the two systems. The floating-point representation of a number uses 32 digits for the numerical part and 8 digits for the exponent, giving a range of approximately 10^{39} to 10^{-39}, with a precision of 9 significant decimal digits. Negative numbers are stored as true complements, and zero is the same in both representations. Approximate operating times are at present as follows:

Floating add/subtract	120–200 μsec
Floating multiply	270–520 μsec
Floating divide	670 μsec
Fixed multiply	290–540 μsec
Fixed divide	540 μsec
Shift n places	36 + 6n μsec
Others	20–50 μsec

Input and output are by means of five-track paper tape; there are two input and two output channels, selected by the program. Paper tape is read photo-electrically at speeds up to 1,000 characters per second (with the ability to stop on a single character) and output is at 30 characters per second. Auxiliary storage is on magnetic tape, using two Decca twin-tape units.

2. Special Features

The control system of EDSAC 2 is based on a large switching matrix containing 1,024 ferrite cores each 8 mm in diameter (see Wilkes, Renwick, and Wheeler, 1958). Wires threaded through these cores control the various gates in the machine; it is thus possible to execute the various elementary transfers, which together constitute an order, by switching a series of cores. The way in which wires are threaded through the matrix determines the sequence in which the cores in the matrix are switched, and also the gates to be opened. At appropriate places the selection of the next core to be switched depends on conditions elsewhere in the machine, thus making alternative paths possible; for example, in a multiplication the digits of the multiplier are inspected one by one and determine whether or not the shifted multiplicand is to be added to the partial result in the accumulator. The use of this control-matrix system makes it possible to include in the order code relatively complicated orders for the transfer of data to and from the magnetic tape.

There are some 80 orders in the order code including full arithmetic facilities in both fixed and floating point. After every floating-point operation the number in the accumulator is "standardized" so that the numerical part is between $\frac{1}{2}$ and 1 in modulus, the exponent being adjusted accordingly. Particular attention has been paid to securing efficient round-off, and small integers are treated exactly. Thus the result of multiplying 2 by 3 in floating-point form is 6, and not 5·9999. . . . Orders are provided both for direct and accumulative multiplication; in the latter case the product can be subtracted from the number in the accumulator if required. Exchange, and add-to-store orders are also provided. A cyclic left-shift order is available which facilitates table-look-up operations.

With only two modifiers it is important to make sure that they can be used efficiently. It is possible to set a number in the modifiers positively or negatively, and to add to or subtract from a modifier; the operand in these orders may be a given number or may be the content of a specified half-register in the store. Orders are also provided which combine an increase or decrease in the modifier content with a conditional jump on the resulting value, and straightforward conditional jumps, exchange, add-and-store, and store orders are available. The content of a modifier register may be stored as the function part of an order, so that "instruction arithmetic" can be carried out conveniently and quickly in the modifiers. There are two special orders which enable a specific quantity, or the content of a specified store half-register, to be used to modify the next order. They serve to provide additional modifier registers, and permit both modifiers to operate on a single order, or an order which affects the modifiers to be itself modified, neither of which operations is otherwise possible. They are invaluable in the "red-tape" sections of a program.

There is a comprehensive input conversion routine which is accommodated in the reserved store. This

provides a floating-address facility whereby orders and numbers may be labelled and referred to in the program by their labels; such references may precede the assigning of a value to a label. The method used for processing the floating addresses is similar to that described by Wilkes, Wheeler, and Gill (1957). Addresses can be modified by having parameters added to or subtracted from them; numbers can be written with decimal and/or binary scaling factors, thus $3 \cdot 14159_{10}80_2 - 200$. All the necessary conversion of orders and numbers is carried out during input; the input tape is read at almost full speed, the necessary work of conversion occupying the time between reading of successive rows of holes. A full description of the programming system will be found in Ref. 4.

3. The Magnetic-tape System

The magnetic-tape system is non-autonomous, in the sense that responsibility for the transfer of data is shared between the program and the control unit of the machine. The read and write orders cause the transfer of a block of data of specified length, together with the computation of a check sum, so that the share of responsibility borne by the control unit is relatively large. The tape is $\frac{1}{2}$ in. wide and carries seven recording channels; of these, five are used for data, one carries synchronizing pulses, and the seventh is a marking channel by which the tape is divided into *blocks*. The non-return-to-zero system of recording is used, and a single (7 channel) head is used for reading and writing, the necessary switching being carried out electronically. The tape moves at 100 inches per second and the recording density is approximately 100 digits per inch on each (data) channel, so that the transfer time is of order 0·8 msec per word. The tape runs out of contact with the head, the gap being about 0·0005 in. wide.

The recording of signals in the block-marking channel is controlled by machine instructions, so that the program can divide the tape into blocks of individually chosen lengths to suit the problem in hand. The blocks are numbered serially, a "label" being written in the data channels immediately following the block mark. The label also contains an indication of the capacity of the block. Once a tape has been marked and labelled, data can be written in a block without disturbing the contents of other blocks, and once so written can be read any number of times, so that the system is a truly addressable, eraseable store. It differs in this respect from other magnetic-tape systems in which data are recorded continuously along the whole length of the tape and must be read into the high-speed store and re-recorded on another tape if alterations are to be made. It is an advantage of the non-return-to-zero system of recording that new information can be written over old information with a smooth join between new and old.

The orders provided for reading and writing effect the transfer of a block of data; it is the responsibility of the program to ensure that the selected tape is in the correct position, moving at full speed, with the circuits set to read or write as appropriate. A block check-sum, computed modulo $2^{40} - 1$, is written automatically at the end of each block, and, on reading, the difference between this and the sum of the data transferred is placed in the accumulator so that the program can take appropriate action if this should be non-zero. The usual action is to repeat the reading operation up to a maximum of twenty times, and then to give some indication to the operator if the transfer is still incorrect. It is possible to check writing by reversing the tape and then reading the information just written, but this is time consuming and was not thought worth while in the program described in Section 3.2.

During magnetic transfers, the arithmetic unit and control unit of the machine are fully occupied, so that computing cannot proceed in parallel. However, an automatic interrupt facility is provided which enables a tape to be positioned on a selected block whilst computing is in progress. If the tape is moving, but reading or writing is not taking place, each time a block mark passes the reading head the tape-drive is stopped and a special flip-flop is set; this flip-flop is examined at the end of each order and, if it is found to be set, the normal control sequence is interrupted. The number in the address part of a half-register set aside for this purpose in the reserved store is increased by one; if it is not equal to 2047 the tape is restarted, otherwise the function part of the half-register is set to zero and the tape is not restarted. In either case the flip-flop is reset and the machine carries on with the execution of the next order. All this takes very little time, so that, if the count is not equal to 2047, the tape is restarted before it has suffered any appreciable deceleration. In order to make use of this facility the program must first start the tape and read the next block label; the tape must then be restarted forwards or backwards as appropriate and a suitable constant placed in the reserved store. At any later stage the program can find out whether the tape is positioned correctly by inspecting the counter in the reserved store; in fact no harm is done if a reading or writing operation is initiated before a positioning operation has been completed.

References

1. Renwick, W. (1957). "A Magnetic-core Matrix Store with Direct Selection using a Magnetic-core Switch Matrix," *Proc. I.E.E.*, Vol. 104, Part B, Supplement No. 7, p. 436.
2. Wilkes, M. V., Renwick, W., and Wheeler, D. J. (1958). "The Design of the Control Unit of an Electronic Digital Computer," *Proc. I.E.E.*, Vol. 105, Part B, No. 20, p. 121.
3. Wilkes, M. V., Wheeler, D. J., and Gill, S. (1957). *The Preparation of Programs for an Electronic Digital Computer*, Addison-Wesley Press, New York (Second Edition).
4. *Programming for EDSAC 2*, University Mathematical Laboratory, Cambridge (Second Edition, December 1959).