THE DESIGN PHILOSOPHY OF A SMALL ELECTRONIC AUTOMATIC DIGITAL COMPUTER

Thesis Presented for

the Degree of Doctor of Philosophy

By

Paul A.V. Thomas, B.Sc. (Eng.)

June, 1961

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

CONTENTS

Page No.

1. Introduction

The Thesis being submitted is an account of the development of a small electronic digital computer which began during the summer of 1956, after the Author had attended the Convention on Digital Computer Techniques (April 1956) held under the auspices of the Institution of Electrical Engineers. At this meeting it was suggested, during one of the many discussions, that one of the problems was to instil an understanding of digital computers into the engineering world, starting right back at the students in the Universities and Technical Colleges.

With these thoughts in mind, the Author approached the late Professor B. Hague with the idea of developing simple analogue and digital computers, primarily for teaching students both how to use them and also, in the case of students of electrical engineering, their basic design. The idea was received enthusiastically and a Grant was made available to develop the two machines.

Though the work was begun in 1956, when a rough scheme was drafted in order to obtain a Grant, the main part of the work was not started until two years later, the intervening period having been devoted to the development of the analogue computer. This, in fact, proved to be a fortunate decision, as it enabled the Author to take advantage of rapidly falling prices of transistors during the two years instead of using thermionic valves, as originally intended.

/The Thesis

The Thesis concerns itself with the development of the Digital Computer only, in particular with the special problem of providing a flexible machine as comparable as possible with modern design and techniques and yet keeping the cost to a low value. In order to do this, in the final design, considerable time was devoted to the overall logical design and to analytical and experimental investigation of some of the basic units in order to achieve a maximum operating speed with essentially low frequency components.

The Author wishes to acknowledge the support and encouragement given to him by the late Professor B. Hague and to thank Dr. A.J. Small for permission to use the facilities in the Electrical Engineering Laboratories. He also wishes to thank Dr. D.C. Gilles and his staff in the Computing Laboratory for their help, and Messrs. Barr & Stroud Ltd. for their help, in particular with regard to the construction of the final machine.

2. History of Automatic Digital Computers

2.1 The Mechanical Era

The earliest known form of digital calculator was the Abacus which dates back several thousands of years, though still in use in Japan today and, it is said, was introduced into Europe about a thousand years ago by Gerber, who later became Pope Sylvester II[B4 ★]. This was followed considerably later (about 1630) by "Napier's Bones" as they were called, which were a system of numbered rods used as an aid to computation devised by John Napier of Napierian Logarithm fame[I4]. The first mechanised digital calculator was not built for a further 12 years, when Blaise Pascal produced his calculating machine, which incorporated ten position wheels for each decade and, more important, included automatic carry from one decade to the next[I4]. Finally, in 1673, Leibnitz produced a machine in which multiplication was automatically carried out by the process of "continued addition".[B4]

However, none of these machines was automatic and, though there was considerable development of these machines in the form of desk calculators, as they have come to be called, there was no further

/development

★ References to individual articles are given by a number whereas references to a book are given by a book reference letter followed by the relevant page number; thus reference B4 refers to page 4 of book reference B (see bibliography).

development towards an automatic machine until the 19th century, when Charles Babbage, a Cambridge mathematician, laid down the principles of the modern automatic calculating machines when he designed, but unfortunately never built, his "Analytical Engine". It seems only fair, therefore, to devote a few lines to this genius, without whom the modern computers might not exist.

2.1.1 Charles Babbage and his Difference Engine

Charles Babbage was born in Devonshire in 1792 and went up to Cambridge in 1810 to study mathematics. However, he was so much ahead of his time that he eventually refused to take the Mathematics Tripos, though this did not prevent him from eventually becoming the Lucasian Professor of Mathematics[B7] (1828 - 1839). In the intervening years he, together with George Peacock and John Herschel, founded the Analytical Society and it was in the Society Rooms that Babbage, in 1812, first produced his ideas for a computing machine.

At the time, the French Government were producing new tables of logarithms, the bulk of the labour force, numbering about 80, carrying out the actual calculations of interpolation involving the operations of addition and subtraction. Babbage proposed to replace these human computers by a machine, the Difference Engine, for which a very able account is given in the "Edinburgh Review" of July 1834. In 1822 he constructed and demonstrated a small working model, which would tabulate to 8 decimals, a function whose second differences were

/constant

constant[W4] and produced figures at the rate of 44 a minute[BB7]. This was received with such enthusiasm that the British Government agreed to give financial support to the construction of a larger machine, the Difference Engine; this machine was to work to an accuracy of 20 decimals and sixth-order differences and also to print out automatically the results to avoid errors of transcription. The machine was never completed, due mainly to the lack of precision equipment to manufacture component parts and the work was stopped in 1833, the Government finally abandoning the project in 1842.

The most important problem that Babbage solved in this design was a method of simultaneous operation of all "carries", thereby speeding up the propagation of "carry over" in addition from one end of a twenty-digit number to the other, changing a whole series of nines to zeros,- a slow process[B10]. The modern equivalent to this is in use today in all parallel computers.

In 1833, while the construction of the Difference Engine was suspended, Babbage conceived the idea of his Analytical Engine which, it has been suggested, was an offspring of the Difference Engine. Lady Lovelace disputes the fact, however, in her translation Note A[B345;1 *], which is important in that it has been suggested that it was the inspiration of the Analytical Engine that

/prevented

---

* The semi-colon divides a book page reference from an article reference.

prevented the completion of the Difference Engine, whereas the incompletion appears to have been due mainly to lack of support. His new ideas must undoubtedly have contributed to this decision not to resurrect the Difference Engine, as his remaining finances were used in an attempt to build the Analytical Engine. Unfortunately, he was only able to complete a small part of the machine, though thousands of detailed drawings were made, from which the Engine could have been constructed[B10]. His son, Major General H.P. Babbage, did in fact construct part of the arithmetic unit and, in 1910, he printed out a table of multiples of π to 20 decimals[BB10].

2.1.2 Babbage's Analytical Engine

As Babbage's Analytical Engine would have been the first completely automatic digital computer, as it would now be called, it seems fitting to give some description of the Engine before entering the second era.

Babbage wrote little about the Engine for, as he said himself, he had little time for writing as he was more concerned with the development. We have, therefore, to rely mainly on other sources for the details,- in particular, Lady Lovelace's translation of Menabrea's paper[1] and Major General H.P. Babbage's collection of papers[3].

Babbage envisaged the four essential units of an automatic computer, namely:

(1) A Store for numbers, which was to have consisted of 1000 columns of 50 counting wheels, each having 10 positions, so that 1000 numbers of a 50-decimal length could be stored. It is interesting to note that an adequate store capacity is today considered to be about this size[BB9, 148].

(2) An Arithmetic Unit, or Mill as Babbage called it, which had the following estimated operation times[W10]: addition or subtraction, 1 second; multiplication (50 decimals by 50 decimals) or division (100 decimals by 50 decimals), 1 minute.

(3) A Control Unit for causing the required operations to be carried out in the correct sequence. The Control Unit was based on the mechanism of the Jacquard-loom[2], which is still used for the production of fabrics having complicated patterns. Cards suitably punched were used to control hooks, which in turn carried a number of warp threads and determined whether or not these crossed the woof threads. These control cards were strung together with string so as to form a continuous chain and, just prior to the passing of the shuttle, horizontal rods associated with the above-mentioned control hooks were moved or not, according to whether or not there was a hole punched in the particular card. There may be 400 or more rods and corresponding hooks in a normal Jacquard-loom.

/Babbage

Babbage intended using two Jacquard-loom mechanisms, one for "operation" cards and one for controlling the "storage transfer" cards, these corresponding to what are now called "functions"* and "addresses" respectively. This, in fact, enables a saving of instruction data, as the same "operation" card could be used several times in succession[1,B352], a practice not actually used in modern machines. About 20 years after Babbage's death, Dr. Hollerith of the American Bureau of the Census, despairing of reducing and tabulating the data for which he was responsible, solved his problems by inventing and introducing methods of recording data by punching holes in cards and the method has been fully exploited ever since[B23].

(4) Input and Output would be also by means of punch cards but, in addition, Babbage envisaged input by means of manual setting of wheels of the store or mill registers; and for output, direct printing to avoid errors of transcription - a point that seems to have worried him greatly. A further very important feature that Babbage had foreseen was that which is now generally called a "conditional jump" in the programme of instructions, in particular the case of a function whose values pass from a positive value through zero. If this was expected, the machine

/changed

---

* See the Glossary, Appendix I.

changed to a new set of cards and proceeded with further computation; if it was unexpected, the machine would stop and ring a bell for attention. Babbage further saw the use of this facility for counting when the same portion of a programme was to be repeated for a number of times dependent on input data[1,B357]; in fact, quite a considerable account of programming technique for the Engine is given in Lady Lovelace's translation of Menabrea's paper and her added notes already referred to earlier.

Enough has been said to see that Babbage's ideas were truly in advance of the times and, had he been less ambitious in the size, he would almost certainly have produced the first automatic computer, but he seems to have spent too much of his time and capital on too many drawings of a too ambitious machine[B10].

In 1909, P.E. Ludgate proposed the construction of an automatic calculating machine, but it appears that nothing was ever built[W15]. It was nearly 70 years after Babbage's death (1871) that work again began on an automatic computer and the next era was entered.

2.2 The Electro-mechanical Era

In 1937, Howard Aiken of Harvard University began work on an "Automatic Sequence Controlled Calculator", now generally referred to as the Harvard Mark I[B173, BB10, W16, H74, E183; 4], which was completed in 1944 with the assistance of the International

/Business

Business Machines Corporation and which used the principles established by Babbage a hundred years before. In fact, it was not as ambitious as the Analytical Engine in that the store consisted of 72 "adding" registers or accumulators and 60 "constant" registers which could not be altered by the computer, as compared to the 1000 registers planned in the Analytical Engine; also the number length was less than half. The storage was on 10-position wheels with automatic tens-transmission as in the Engine, use being made of relays and electro-magnets. Orders were carried out consecutively under the control of a 24-hole wide punched paper tape in a similar manner to Babbage's cards, the instruction being of two-address form (number source and destination).

The addition time was about 1/3 second and multiplication and division times were about 6 seconds and 11.4 seconds respectively. Initially, no conditional jump order was provided unlike Babbage's intention, though at a later date this was incorporated into the computer. Although limited in its use, it is on record as the first automatic computer to be built.

Whilst this work was in progress, the Bell Telephone Laboratories began in 1938 the development of relay computing devices[BB10, W133, H80, E187]. The first of a series of machines was completed in 1940 and, though designed for fire control

/applications,

applications, these culminated in 1944 with a general purpose machine (Mark V), using 9000 relays and 50 teletype units. The number length was 7 decimals and sign, each digit being in the bi-quinary system which is easily checked; thus we see the introduction of binary rather than decimal representation of numbers in the machine. There were 29 storage registers, but the most important fact was that this was the first computer to use floating point notation,- that is, the numbers were stored in the form $a.10^b$.

The Computation Laboratory of Harvard University also built a relay computer (Harvard Mark II)[W141, E185], which was completed in 1948 and which is primarily mentioned because of its completely different design philosophy from that of the Bell Computer Mark V, the two main differences being:

Harvard Mark II: synchronous; no self-checking facilities.

Bell Mark V: asynchronous; considerable self-checking facilities.

The number length of 10 decimals in floating point was the same as in the Bell Computer Mark VI, based on Mark V, which also had a major advance of providing stored subroutines, which were semi-permanently wired and could be called for by tape instructions[W140].

Before leaving the electromagnetic relay computers, mention should be made of the Automatic Relay Calculator (A.R.C.) constructed at Birkbeck College, University of London, under the

/direction

direction of A.D. Booth[B170, BB11]. This calculator is of particular interest in that it appears to be the first machine to use magnetic drum storage (of 256 numbers) and also to store numbers in true binary form (21 digit length); at a later date the magnetic drum was replaced by an extremely compact electro-mechanical store for 50 numbers and a pluggable sequence unit for 300 instructions.

2.3 The Electronic Era

Whilst the relay machines were being constructed, thoughts were turning towards faster operating elements than mechanical devices, in particular electronic valves.

The first all-electronic computer to be built was the Electronic Numerical Integrator and Calculator (ENIAC)[E144; 5,6], designed by J.P. Echert and J.W. Mauchly at the Moore School of Engineering, Philadelphia. The machine was completed in 1946 and was, in many ways, an electronic copy of the Harvard Mark I[B174, H82], only working at a very much greater speed (several hundred times faster). Ten decimal length numbers were stored in decimal form on rings of 10 binarys - an electronic equivalent of the ten-position wheels of the Mark I; storage was provided for 20 numbers, each store being an accumulator with tens-transmission using an anticipatory-carry system similar to Babbage; hand switches were also provided for 300 numbers. Due to the small store, the programme was set up

/on a

on a plug-board and switches - a tedious operation - as the use of a paper tape or cards (as used in Mark I) would have been too slow. A "master programmer" was also incorporated, upon which sequences of operations which were often used or repeated could be set up and selected as required (similar to the endless loops of tape on the Mark I) and equivalent to what would now be called a "Subroutine"; in addition, a conditional jump facility was also provided (unlike the original Mark I). The complete equipment contained about 18000 valves, 1500 relays and consumed 100-150kW of power. It was a tour de force of electronic engineering, the designers achieving their end by straightforward means without consideration of the large amount of equipment required[BB14, W20]. It did, however, lead the way to using high-speed electronic elements in automatic computers.

Whilst the ENIAC was under construction, J. von Neumann and his co-workers at the Moore School devoted considerable attention to the problem of the optimum form of a calculating machine[BB15], the results of which first saw the light of day in a now famous report drafted by von Neumann in 1945[B176, W31; 7]. This report set in motion the construction of three large machines, which were considerably more versatile than the ENIAC though considerably smaller in physical size. These three machines, which all made use of mercury sonic delay line storage were:[BB16]

(a) The Electronic Delayed Storage Automatic Computer (EDSAC I) completed at Cambridge University in 1949 under the direction of M.V. Wilkes.

(b) The Automatic Computing Engine (ACE) Pilot Model completed at the National Physical Laboratory in 1950 under the direction of Womersley, Turing and Colebrook, and

(c) The Electronic Discrete Variable Automatic Computer (EDVAC) completed at the Moore School (about 1954) under the direction of von Neumann.

However, prior to the appearance of the EDSAC I, two other forms of storage were being successfully tried. At Birkbeck College, University of London, A.D. Booth was constructing an all-electronic computer using drum storage, the Simple Electronic Computer (SEC), which was completed about 1948 and which led to a series of All-Purpose Electronic Computers (APEC) which were constructed on similar lines[B170, BB19]. An important feature of these computers was the use of a two-address code enabling Optimum Programming[BB152; 8] to take place. This is most important on a magnetic-drum type store machine, as it reduces the waiting time inherent in a cyclic storage device.

The other form of storage being tried at this time was the Cathode Ray Tube Electrostatic Store used in a small prototype

/machine

machine built at Manchester University[B118; 9] under the direction of F.C. Williams and completed in June 1948. The advantage of such a storage system was its immediate access compared to the drum or mercury delay line; the limitation, however, was in size- one Cathode Ray Tube store being capable of holding only 1000 bits of information. It was realised that a full-scale machine would require about 100 tubes and this idea was quickly forgotten[B120].

The alternative, which was adopted in the second machine, completed in 1949, was to incorporate two level storage, as propounded by von Neumann, namely,-a small immediate access Cathode Ray Tube store and a large backing store in the form of a magnetic drum,- the latter being synchronised to the former by means of a servomechanism[10,11]. A further important addition in this machine was the introduction of the "B-tube" used for instruction modification - two lines being available[B122].

Later in the same year, a larger machine was constructed in which the transfers between the Cathode Ray Tube store and drum store were controlled by means of the programme rather than being a manual operation[B123]. This culminated in the Mark I and Mark I* computers, which were developed jointly[12] with Messrs. Ferranti Ltd.

In the meantime, the first EDSAC machine was completed, both single and double length working being possible, as with the second 1949 Manchester machine[W44]. The advantage of this

/is

is that storage is of single-length form which conveniently accommodates one instruction, the double-length number being normally used for arithmetic which would be wasteful for instruction purposes (15 - 20 digits normally being adequate).

In 1951, semiconductor diodes were introduced in the Bureau of Standards Eastern Automatic Computer (SEAC) but, apart from this, the machine is similar to the ACE[B175, BB17].

1952 saw the use of Dekatron cold cathode tubes being incorporated in a parallel decimal machine built at Harwell - though relatively speaking a slow machine, it claimed a high degree of reliability and was commonly run unattended overnight and over week-ends[B140; 13]; it should perhaps be pointed out, however, that the actual number of operations in this time would compare with only an hour or so on a fast machine.

The following year saw the advent of the first transistor computer at Manchester University[14], which was a prototype incorporating 92 point-contact transistors and a number of thermionic valves where output power was required. All the storage was carried out on a magnetic drum, auxiliary stores being provided by regenerative tracks on the drum surface. A full scale computer, based on the prototype, was completed two years later in 1955, and a commercial model was completed the following year, the MV950[15].

/Also

Also in 1955, the United Kingdom Atomic Energy Research Establishment at Harwell produced a fully transistorized computer CADET[16], in which about 1/7th of the transistors used were junction types and, in America, the IBM608[17] calculator appeared, in which only junction transistors were used. At this time also, the American Whirlwind I computer had a magnetic core store added to it[18], thus introducing this new form of high speed storage.

Concurrently with their semiconductor computer projects, Manchester University produced its Mark II[19] computer, which was similar in many respects to the Mark I but which included floating-point arithmetic, giving a great improvement in speed of operation when carrying out floating-point arithmetic. This machine was completed in 1954 and a production model, the Mercury[20], constructed by Messrs. Ferranti Ltd. in 1955 was improved by incorporating a magnetic core store[21] in place of the Cathode Ray Tube store of the earlier machine.

About the same time (1954), as the larger scientific Mark II machine was being built at Manchester, the commercial world was taking note of automatic computers, which resulted in a series of plug-programme accountancy machines using binary coded decimal or sterling form rather than pure binary. These machines tended to be slower than the scientific machines, due to the large amount of input and output operations and relatively

/little

little arithmetic. Apart from Lyons Electronic Office LEO I[22], which was, in fact, a modified EDSACI, one of the first accountancy machines appears to have been produced by the British Tabulating Machine Co. Ltd. in 1954[23]. This was followed in 1955 by the transistorized IMB608[17] and, in 1956, there appeared the HEC4E[24] and the Vickers-Armstrongs (Engineers) Ltd. Programme-Controlled Computer[25].

This same year (1956) saw the advent of several new ideas. Firstly, Messrs. Ferranti Ltd. introduced a printed circuit plug-in package system into their Pegasus computer[26], as did Metropolitan-Vickers Electrical Co. Ltd. in their MV950 already referred to earlier. The plug-in package system was not new, having been first used in computers in 1947[26], but the introduction of printed circuits did much to support the system which had had much criticism previously on the grounds of the large number of plugs and sockets of questionable reliability.

The Pegasus computer also saw the introduction of the solid delay-line storage system, which makes use of the magnetostrictive properties of nickel[27,28], this being in addition to a magnetic drum backing store.

About this time, Messrs. Decca Radar Ltd. produced their C1 computer[29], the major part of which is composed of magnetic cores and rectifiers only,- the main store again being a magnetic drum.

/In

In America, the Bell Telephone Laboratories Inc. developed a new computer system, using surface barrier junction transistors in directly-coupled logic (DCTL), resulting in economy of components. The result was the Tradic Leprechaun Computer[30], which was a small special purpose computer having great flexibility and which was primarily developed for testing the DCTL system.

The Japanese ETL Mark III[31] is worthy of note in that the main store consisted of optical glass acoustic delay lines and all numbers were represented by a sign digit and modulus rather than the complement for negative numbers.

This, then, was the situation towards the end of 1956, when it was proposed to develop an automatic digital computer in the Electrical Engineering Department of Glasgow University, primarily as a teaching device for demonstrating computing principles and computer engineering practice.

3. Basic Computer Units

From the point of view of computer engineering practice, it is obvious that one wishes to use up-to-date methods rather than those which are obsolete. This, in turn, rules out any method other than electronic, as all modern automatic calculators are electronic in order to obtain high speed of operation.

Considering then the Electronic era, we see first the electronic equivalent of an electromechanical machine, the ENIAC, the store of which was really quite impractical. The magnetic drum and mercury delay line stores were far more practical, but both suffer from their relatively slow access time unless optimum programming techniques are used, which puts more strain on the programmer. This problem was overcome by the use of the Williams' Tube store, which suffered only in its physical size and limited life, and led to the introduction of a Two-level Storage machine built at Manchester University. The fast store (CRT) was then finally replaced by magnetic core storage, which has the advantage of apparently unlimited life compared to the Williams' tube and also was physically much smaller.

At the same time as the storage system was undergoing these changes, the thermionic valve was being replaced by semiconductor devices; firstly, by diodes, as in the SEAC

/computer

computer and eventually by point contact and, more recently, by junction transistors. The main reasons for this development were: (a) reduced power consumption; (b) small physical size and (c) apparent long life.

It is now proposed to consider the merits of the various types of unit individually.

3.1 Storage Elements

The storage of numbers and instructions in a computer falls into two classes:

(a) Main storage, in which all the instructions, data and results are located, and

(b) Temporary storage, in which the various mathematical, logical and control functions are carried out. These storage units, commonly referred to as Registers, are normally of single-word length, that is, they hold one complete number (data or result) or instruction though, as we will see later, double- and half-word length* storage registers, as they are usually called, are also used. Similarly, a few temporary stores may be of single-digit length only.

/We

---

* Double- and half-length may often be only approximate.

We will consider these two types of storage independently, as the requirements are rather different.

3.2 Main Storage

It was seen in the previous Chapter that, about 1956, the main store of computers was either -

(a) the ultrasonic delay line,

(b) the magnetic drum,

(c) the magnetic core,

or else a combination of (a) and (b) or (b) and (c), in which case the magnetic drum (b) was used as a backing-up store of a two-level storage system.

3.2.1 The Ultrasonic Delay Line Store

The ultrasonic delay line store, using mercury as the conducting medium, is one of the earlier storage devices and the method of operation has been fully described elsewhere[32,33]. The main disadvantages of the system are -

(i) temperature sensitivity,

(ii) physical size and weight, and

(iii) necessity of maintaining the supplies if the stored information is not to be lost. Further, if the temperature is not maintained constant, contamination of the crystal-mercury contact can cause trouble.

For the particular machine in view, which was likely to be used intermittently, condition (iii) ruled out delay line storage as a possible main store.

3.2.2 The Magnetic Drum Store

The magnetic drum store has been incorporated in most modern computers, either as the only store or as a backing store in a two-level storage system. There has been, therefore, considerable literature on the subject, dealing with the different methods of recording on the drum surface and any further description of this type of store would seem to be unnecessary [BB122;34-38]. It appears to have all the necessary requirements of a main store and has only one serious fault,- the relatively long access time that may occur if optimum programming is not used; although this loss of time is not likely to be serious in a computer primarily designed for teaching purposes in which speed of operation is not so much a criterion as reliability and permanence of storage.

3.2.3 The Magnetic Core Store

This form of storage is much more recent than the two previous methods and has the advantages of small volume, reliability, permanence and immediate access. The method of storage depends on the fact that two possible values of remanent flux are available in a magnetic core, according to whether the magnetic field has been set in a positive or negative sense [20,39,40].

From the foregoing remarks, the most promising form of main storage appeared to be one of the two magnetic systems, with a possible preference for the core on the grounds of immediate

/access

access without the inconvenience of optimum programming. The relative cost of the two systems was then investigated,- an important point,- particularly in a small computer.

3.2.4 Magnetic Storage Cost

In 1956, several firms known to be active in the computer field were approached on the subject of computer storage components,- particularly as to cost. It appeared, however, that, in this country at that time, there were only two main suppliers of magnetic storage equipment,- Messrs. Ferranti Ltd., who manufactured magnetic drums, and Messrs. Mullard Ltd., who manufactured cores suitable for digital storage purposes.

The quotations received are given in graphical form in Fig. 1, and it can be seen that the price for the drum storage was essentially a constant figure, almost independent of storage capacity, except for the Canadian models which were designed for approximately 20,000 bits. In the case of the magnetic cores, the cost is essentially proportional to the capacity, as might be expected, until certain sizes when the cost per bit is reduced, due to a quantity price reduction, as is common practice in industry.

Fig. 1 does not give the complete picture, however, in that the external location selector system is not included. However, this is likely to be similar in cost for both methods and therefore represents the same additional cost to all graphs. It

/appeared

appeared, therefore, that the cost for a core store would be less than that for a drum store if the size was less than 40,000 - 50,000 bits, after which the Ferranti 1009 drum store would become cheaper at about £1,250. As it was desired that the entire computer was to cost about £2000, this figure was obviously too high, and it was decided that the store would therefore be of the magnetic core type, providing no further advantage could be found for the drum to justify its increased cost. On this count, it should be mentioned that whereas reading information from the drum is non-destructive, in the case of the cores the information is normally destroyed and has to be re-written if (as is general) it is required again. Fortunately, this is not a serious matter, as it only requires that every "reading cycle" is followed by a "writing cycle", which has to be available in any case for writing in new information. Non-destructive systems of read-out have been described[41-43] but, in general, they tend to be complicated.

3.3 Temporary Storage

The main requirements of the temporary stores are as for the main store, with certain additional facilities., viz., it is convenient to have an indication of the content of these stores for the particular purpose, and it must be possible to be able to shift the digit pattern left or right in at least one such store besides carrying out a range of arithmetic, logical and control test functions. There are several temporary storage

/methods

methods available, the first three of which are simply small forms or portions of the above mentioned main storage systems.

3.3.1 Main Store Types

(a) Ultrasonic Delay Lines

These are commonly used where the main store is of the same form, though in this case temperature control is not normally necessary due to the short length - a few inches. However, there is little to justify using the Delay Lines when the main store is not of the same form.

(b) Magnetic Drum

In a few magnetic drum machines, some tracks have been used regeneratively as the temporary registers[14,16,44] but, as seen in Section 3.2.4, the magnetic drum has been discarded on the grounds of cost, and hence this form of temporary register is not applicable.

(c) Magnetic Cores

Magnetic Core registers have been used in several computers and have been found to be extremely reliable, due to the nature of the magnetic characteristics. In similar manner, they have also been used in counter registers as opposed to shifting registers. The coupling between the cores in the earlier stages was by means of diodes[BB112; 29,45-47], but these were later replaced by transistors[48-50] which, giving power amplification, reduced the magnitude of the shifting-pulse power.

/The

The main advantages of the system are the low power consumption,- there being no power absorbed except during the shifting period,- and physical size. Disadvantages of magnetic core registers are:-

(i) the inability to observe the contents of the register without shifting or using the complicated non-destructive read-out systems referred to in Section 3.2.4, and

(ii) the speed of operation which is at present limited to about 100 kc/s,- though this is not a serious matter in the present case.

3.3.2 Bistable Unit

The bistable unit, or flip-flop, is an electronic circuit which, as its name implies, has two stable states, which can therefore conveniently represent the two possible binary conventions "0" and "1". Such bistable units exist in almost every electronic computer, either in word registers or single digit stores, and very high speeds of operation can be achieved.

The first reference to such a circuit seems to be due to Eccles and Jordan[51] in 1919; this was a very simple circuit incorporating two triode valves. Since then, far more elaborate circuits have been developed[E13], mainly in an attempt to increase the speed of operation and stability of the circuits. A typical circuit, using pentode valves and operating at a speed of 1 Mc/s, was used in the Manchester Mark II computer[19,52], though this is not

/the

the ultimate limit, which has since been increased to 10 Mc/s[53].

The alternative to the valve binary element is the use of transistors. In the earlier stages, point-contact type transistors were used which, due to their characteristic, required only one transistor per stage. Manchester University used these at a pulse repetition of 125 kc/s[14], but their reliability was poor. The junction transistors proved to be far more reliable than the point-contact type but, at the time, the speed of operation was limited to about 50 kc/s[16]; however, this was not a serious matter for the presently conceived machine. Since 1956, high speed transistors have been developed, giving operation speeds of 10 Mc/s and higher[54].

At the time (1956) therefore, it seemed, from the foregoing statements, that the ideal system of word storage for the particular computer would be shifting registers using valve or junction transistor bistable units and it was necessary, therefore, to consider these two in more detail.

The great advantages of the transistor binary compared to the valve binary are:-

(a) low power consumption - no heaters and generally lower power levels;

(b) apparent reliability - no ageing due to cathodic deterioration giving low emission;

/(c)

(c) low impedance circuits due to their inherent properties, which assists stability, and

(d) small volume.

Of these points, only (b) was in doubt, due to the long existence and hence experience of valves and relatively short existence of transistors; although there is still no apparent reason why a transistor should alter its characteristic over a long period of time in a normal environment.

A fifth point was investigated, however,- namely cost, which is important if there are to be a large number of binary units in the computer. It was found that at the time, whereas a suitable double triode valve cost about 18/-, a single junction transistor cost about 25/-, thus giving a price-ratio for the active elements of a binary unit of about 3:1. As the cost of the computer was to be kept to a minimum, it was decided to use valve binary units.

A final point which aided this decision was that the transistors were still undergoing development. There was a possibility, therefore, of the selected transistors, if used, becoming obsolete, whereas the probable valve types considered were well established and unlikely to become obsolete.

/3.4 Control Gates

3.4 Control Gates

Here the choice of components was more restricted, there being the possibility of (a) magnetic cores, or (b) electronic devices; both thermionic and semiconductor types being considered.

As regards magnetic gates, these are more suited to an entirely magnetic system of temporary storage (as in Section 3.3), though they have an advantage compared to the electronic gates discussed below where a number of inputs to the gate is required, this being achieved by the use of additional windings on the core[55].

The most common form of electronic gate is the diode gate, for which point contact and junction semiconductor diodes have been used to a considerable extent, due to their small size and power consumption compared to the vacuum diode with its additional heater power. Where the gate has to drive a number of loads, common practice has been to follow the diode gate by a triode valve or transistor, rather than the alternative possibility of triode gates which tend to be more costly.

Considering the relative cost of the electronic gates, there are four possibilities, assuming one follows common practice:

(a) Vacuum diodes and triodes.

(b) Semiconductor diodes and vacuum triodes.

(c) Vacuum diodes and transistors.

/(d) Semiconductors

(d) Semiconductor diodes and transistors.

As was seen in Section 3.3.2, the cost of the transistor was much greater than the vacuum triode (in 1956) so that this tended to remove the possibilities (c) and (d). Upon investigation of the relative prices of diodes, it was found that suitable semiconductor diodes were priced at 4/- each compared to the vacuum double diode at 9/-. In addition, the semiconductor diode provides a saving of heater power and physical space; thus possibility (b) seemed to be the most promising, and (c) the least.

Based on the foregoing remarks, a provisional computer design was considered in order to provide an estimate of cost for a small digital computer, whose primary use was for teaching purposes.

4. Valve Computer

It is desirable that the computer should fulfil the following conditions:

1. Simplicity of use
2. Reliability
3. Ease of demonstrating its operation
4. Flexibility
5. Low cost
6. Small size for ease of movement.

As a result of the considerations in Section 3, a provisional design was drawn up for a fixed point binary computer, in which the main storage was to be magnetic cores (Section 3.2) and all electronic elements would be thermionic valves and semi-conductor diodes (Sections 3.3 and 3.4).

4.1 General Requirements

The computer consists basically of five sections:

(i) Store

(ii) Arithmetic Unit

(iii) Control Unit

(iv) Timing Unit

(v) Power Supplies.

In addition, there is the peripheral equipment for transferring

/information

information to and from the computer.

In considering the design of the computer, certain costs are (a) essentially constant or (b) mainly dependent on the word length, whilst (c) the store is dependent on the number of words (storage locations).

4.2 Word Length

The items (b) whose cost is dependent on the word length are the temporary storage registers, whose cost can be expressed as

$R = a.w$ where a is a constant being the cost of a single digit store and w is the number of digits in a word.

On these grounds, therefore, it is desirable to keep the word length to a minimum. On the other hand, the word length is also dependent on the computing accuracy required in the case of numbers and the accommodation of all the necessary information in the case of instructions.

4.2.1 Number Form

From a purely demonstration point of view, the accuracy of computation is not very important but, from the secondary point of view of usefulness, it was decided that an absolute minimum accuracy of 4 decimal digits could be tolerated, equivalent to 14 binary digits (16,383), in addition to which must

/be

be added 1 binary digit, to indicate a positive or negative number, giving a total minimum word length of 15 binary digits.

As regards the number representation, two methods are available, namely:-

(a) (sign) and (absolute value);

(b) true complement modulus 2 for negative values, assuming that the modulus of all numbers is less than unity.

The first method has the advantage of being in the (human) conventional form, but it is not so convenient for the computer, which would have to carry out sign tests before each arithmetical operation, resulting in a slowing down of the machine. Thus,

$$+(A) + (-)(B)$$

would have to be interpreted by the machine as

$$+(A) - (+)(B)$$

that is, the addition operation becomes one of subtraction.

One machine (the ETL III) at least has been built[31] using this convention, but in general, the alternative (b) has been used. In this convention, a negative number n is represented by (C - n) where C is a constant which must be greater than the largest number that can be stored in a register which, in the present case, is just less than unity. In this case, the next largest number (in the binary scale) is $2^1 = 2$ and, therefore, C = 2 and a negative number n will be represented by 2 - n; the largest

/negative

negative number permitted will be -1, which is represented by $2 - 1 = 1.0000....$ So that any number x will lie in the range $1 > x \geq -1$.

It will be seen, therefore, that with this system the most significant digit (to the left of the binary point) is always zero for a positive number and unity for a negative number. With this system of sign convention, addition and subtraction can be carried out directly without any sign testing. In the case of multiplication and division, however, the (sign) (modulus) convention has the advantage over the complement system but, as these operations are generally slower and far less frequent, this does not give rise to a preference for the former representation, which was therefore discarded.

4.2.2 Instruction Form

The general form of a computer instruction word consists of:

(i) Function Digits

(ii) Address Digits

(iii) Modifier Digits

(iv) Miscellaneous Digits.

(i) The function digits are used to inform the computer of the operation to be carried out, i.e., add, subtract, etc., and each such function is given a number; so if the number of functions

/required

required is F, then the corresponding numbers of binary digits f in the order will be such that $2^f \geqslant F$.

(ii) The address digits give the location of all instructions and numbers in the store and the number of digits a will be such that $2^a \geqslant A$, where A is the required number of store locations or addresses. The number of addresses in any one instruction varies from one to four, according to the particular computer, so that the total address digits $a' = k.a$, where k is the number of addresses. In multi-address computers, the addresses may designate (a) the storage location of a number or numbers to be operated on; (b) the location of the result of the operation and (c) in a non-sequential computer, the location of the next instruction.

(iii) The modifier digits b for order modification to be provided, are such that $2^b \geqslant B$, where B is the number of B-modifier registers. If an instruction is to be left unmodified, then it is common to regard $B_0 = 0$ as a non-existent B-register,- that is, a register containing zero, this being the easiest method of dealing with non-modified and modified instructions.

(iv) The miscellaneous digits, if existing, may be used for special purposes, such as optional stopping of the computer, etc. Thus, if (iv) is excluded, the total word length of an instruction

$$t = f + ka + b \quad \text{digits}$$

and this must be considered in the light of reasonable values of

/F, A, k

F, A, k and B.

In the simple computer envisaged, the number of functions appeared to lie within the range of 9 to 16, less than this figure would be rather limited, though not impossible, and a greater number would seem unnecessarily ambitious for the purpose. Thus, the value of f was made equal to 4.

In the case of the number of storage locations, Booth has suggested that a fast store of 512-1024 words was a good practical size, together with a larger backing store[BB148]. For the main purpose envisaged, it seemed that to construct a store having a minimum of 256 locations would be acceptable, thus making $a \geqslant 8$.

So far as (iii) was concerned, one B-register was considered adequate in addition to $B_0 = 0$, giving $b = 1$. It was realised that the uses of the B-register would be limited by lack of numbers, but the cost of registers is relatively high so that different modifiers would be stored in the main store and transfer instructions would be available between the two.

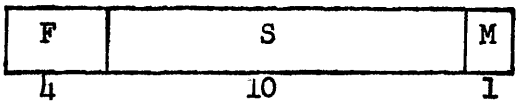
There did not appear to be any need for miscellaneous digits (iv), so that the minimum value of t would be

$$t = 4 + 8 + 1 = 13 \quad (\text{if } k = 1).$$

This is less than the number required by the number word length (15) which would, therefore, appear to be a satisfactory figure

/allowing

allowing the additional digits to be kept for doubling the size of the main store, number of functions or modifier registers. If k were greater than 1, then the minimum value of $t = 4 + 16 + 1 = 21$ would be far in excess of 15, so that it is convenient to use a Single-Address Instruction. Thus it was decided to use the instruction form:



where F denotes the function digits

S the storage location or address

M the modifier - 0 = non modified

1 = modified

It is interesting to note that the EDSAC I word length was 17, with the possibility of coupling two words together for additional accuracy of numbers[W44].

4.3 Size of Main Store

The cost of the main store is dependent on both the number of locations and word length, so that its cost is mainly of the form:

$S \doteq b.W.w.$ where b is the cost for each bit of information stored,

W is the number of words to be stored,

w is the number of digits in each word as before.

/in

in addition to which, there is the cost of the address selector system.

The value of w has already been fixed in the previous Section as equal to 15, and W was suggested as being 256 or 512, giving a resultant number of bits W.w = 3840 or 7680 respectively.

Considering now the cost of such a store with reference to Fig. 1, it is seen that the corresponding points on the graph are A and B, the latter being about 40% higher in cost; at point C, a further doubling in the size of store produces a cost of about $1\frac{1}{2}$ times A (for 4 times the size). An intermediate point between B and C can be found for which the cost lies about half-way between A and B. As this is not a standard size of matrix, the bit dimensions of which are usually 16 or 32, the resultant cost might well be higher. Also, the cost of the address selector system becomes relatively expensive if it is not being fully utilized, as in the non-standard case when W is not equal to an exact power of 2.

It was decided, therefore, to provide storage space for 256 words of 15 binary-digit length.

4.4 The Arithmetic Unit

The arithmetic unit was to provide automatic addition, subtraction, multiplication and division with both

/positive

positive and negative fixed point fractional numbers, giving the correctly signed result in all cases.

The possible modes of operation are serial and parallel, the main advantage of the latter being speed but with much greater complexity. As speed was not important in this particular computer, it was decided to operate in the serial mode, thereby reducing the cost of the equipment. For example, for the addition of two numbers of length w, only one adding unit is required in the serial mode, whereas in the parallel mode w adding units would be necessary.

4.5 Operation Timing

The sequence of operations to be carried out in the computer can be divided into four sections:

(1) Select the storage location of the instruction to be carried out - the present instruction (P.I.)

(2) Transfer the P.I. to a temporary register so that (3) can be obeyed.

(3) Select the storage location of the number to be used in the P.I. and set-up the function to be carried out.

(4) Obey the P.I.

Repeat (1) to (4) as long as required.

The duration of operations (2) and (4) in a serial

/machine

machine will normally be equal and determined by the word length w and the periodic time τ secs of the clock pulse, so that these operation times will be $w.\tau$ secs each. In some cases, however, such as multiplication and division this time interval will usually have to be increased.

In the case of a single-address computer, a counter is used for the P.I. location, unity being added to this control counter prior to the P.I. address selector being set up (operation (1)). This whole operation is quite fast and takes a time of about $2.\tau$ secs in a medium-speed computer.

Operation (3) is similarly fast, the number address being in the P.I. temporary register, or control register as it is commonly called, and the time taken will again be about $2.\tau$ secs.

Thus, for the normal operation, the total time taken for the four operations will be

$$T = 2.w.\tau + 4.\tau \text{ secs}$$

and if $w = 15$, then

$$T = 34.\tau \text{ secs.}$$

If the computer is not to be too slow, then τ must be kept reasonably short, that is, the clock pulse repetition rate must be kept as high as possible.

When reviewing the clock frequencies of previous computers, it was observed that the majority operated at clock frequencies of less than 150 kc/s whilst a few operated at 1 Mc/s

/or

or above. Generally speaking, pulse circuit techniques become more difficult as the frequencies become in excess of 100 kc/s, due to the effects of stray capacitance of the circuits and, at very high frequencies, trouble may be found with delay times in long cables. Further, the time taken to extract a number from the magnetic core store is of the order of 5 - 10μs[21], unless special techniques are used.

As a result of the above reasoning, a clock frequency of 100 kc/s was chosen, giving τ = 10μs and a normal operation time of 340μs, or about 3000 operations/sec. In addition, it would be arranged that slow-speed operation (about 1 c/s) would be available for demonstration purposes.

4.6 Input-Output Facilities

Normally, input and output of information is carried out by means of punched-cards or paper tape, the tendency being towards the latter medium. One of the advantages of the latter is convenience of storage; on the other hand, the former is probably the easier to read visually but, in general, appears to be more bulky. It was decided, therefore, to use standard 5-hole paper tape equipment. In addition, lamp indication would be used for the purposes of demonstration so that the observer would be able to follow the mode of operation of the computer at slow speed.

4.7 Final Arrangement

To summarize the foregoing sections, the resulting computer specification was as follows:-

(a) Single-address in serial operation.

(b) Numbers in fixed point, within the range $1 > x \geq -1$

(c) 16 possible functions, as given in Appendix III.

(d) Main store: 256 words of 15 digits length.

(e) One B-register for order modification.

(f) Input and output by means of 5-hole paper tape.

(g) Clock frequency 100 kc/s, giving a normal operation speed of about 3000 per second.

(h) Active elements to be thermionic triodes, and in some cases pentodes, and semiconductor point contact diodes.

(i) The estimated power consumption was 2.5 to 3kW and the cost £2,200, excluding the input and output equipment.

The block schematic diagram for this computer is given in Fig. 2 and is briefly described below.

As can be seen from the diagram, the contents of the main store can be selected through a decoder SDC from either the control counter, CC, or control register, CR, according to the switches marked I and O, corresponding to the instruction selection time and number selection or operation time.

/Assume

Assume that initially the control counter is set to zero and that the instruction-switches are closed. The store decoder then selects the first location of the store (0) and, by means of the timing pulses (not shown in the diagram) transmits the instruction P.I. therein, via I_2, to the control register. As the first digit (the B-modifier digit) emerges from the store it is tested and opens or closes the "Mod" switch, according to whether the digit is "0" or "1" respectively. In the latter case, the contents of the B-register, BR, are added to the output of the store, modifying the instruction, before passing it into CR. The contents of BR are also recirculated, so that they can be used over a succession of instructions. This completes the instruction period with the P.I. in the control register and the I-switches opened.

The operation obeying switches, O_1 to O_3, are now closed, setting up the location of a number in any storage location S, given by the present source or destination PS digits, and the function being decoded in the function decoder FDC. The output of the decoder, which is fed to the arithmetic unit AU, is used to control the gates into and out of the accumulator A, a register R, B-register and input and output equipment, which is assumed to be part of AU in the diagram.

In the case of shift and jump orders in which the store is not used, the PS digits are treated as integers passed directly to the AU through On or to the control counter through

/Oj.

0j. Finally a stop order simply prevents the generation of the timing waveforms at the end of the instruction. On completion of the order being obeyed, a single pulse is applied to the control counter at OC, except in the case of jump orders which inhibit the +1 pulse. The cycle of operations is then repeated with the next instruction being extracted from the store.

In fact, this computer was never constructed due to pressure of other work, mainly the construction of an Analogue Computer[56], * at the time. By the time the digital computer project was resumed in 1958, fresh thoughts on the matter led to the conception of a somewhat different computer, mainly due to the impetus in the semiconductor field.

The reason for including the contents of this chapter is mainly to show something of the history of the final computer, and also what could be reasonably considered as a workable minimum size.

---

* See Appendix VII.

5. Transistor Computer - General Considerations

Before considering the details of the Transistor Computer, the historical development and reasons for changing the computer design will be dealt with in the following section.

5.1 Historical Development

When the digital computer development was resumed in 1958, it was noted that the prices of transistors had fallen considerably and the possibility of using transistors instead of valves was more seriously considered, due to their other advantages already enumerated in Section 3.3.2. Upon investigation, it was found that the cheapest transistor available having reasonable properties appeared to be that manufactured by The General Electric Company Ltd. under the type No. GET3. In particular, it had a reasonable grounded emitter current gain cut-off frequency f_{α} of about 1 Mc/s and a price of 10/- compared to the double-triode price of 18/-.

Also, during 1958, the Computing Department of the University was set up under the directorship of Dr. D.C. Gilles. When the Author made the acquaintance of Dr. Gilles towards the end of that year, he made several suggestions, based on mathematical grounds, which would make the computer more versatile. These were:-

(1) To increase the word length and hence the accuracy of calculation.

/(2) To

(2) To provide additional conditional-jump orders, possibly some logical orders and a standardise or normalise order, so that floating-point arithmetic could be more easily programmed. In addition, space in the order code should be left for the addition of block transfer orders to and from a backing-store, such as a magnetic drum, to be added at a later date.

(3) To use function numbering rather than lettering, as this would enable standard teleprinter equipment to be used without the continual use of letter-shift and figure shift instructions having to be punched on the paper tape, which is both time-consuming and wasteful of tape.

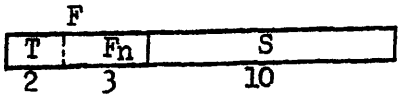
As it was considered desirable to maintain the possibility of 1024 storage locations (though only 256 would be included initially) and the cost of the main store being one of the major items (about 25%) and proportional to the word length, it was decided to consider double-length number operation to increase the accuracy of working similar, in principle, to that used in EDSAC I and the Manchester computers, which had word lengths of 17 or 34 digits and 20 or 40 digits respectively. Thus the space occupied by an instruction would still be one location of 15 digits but arithmetic could be carried out with 15 or 30 digits, according to the function digits of the instruction; the two halves of a double-length number being stored in adjacent storage locations.

/This

This means a doubling of the arithmetical orders in addition to those brought about by the suggestions in (2) above. It was impractical to accept a maximum of 16 functions and this figure was accordingly increased to 32, requiring 5 binary digits in the instruction.

Assuming the maintenance of 10 digits for the storage address locations, this now filled the 15-digit word length without leaving a space for instruction modification by a B-register. In order to overcome this without increasing the word length, which might, in fact, have been worthwhile, a modify instruction was introduced so that the contents of any specified storage location could be added to the next instruction before being passed to the control register. The B-register, however, was to be retained as a "counting" register for use in iterative processes requiring a number of prescribed cycles.

The instruction form now became:



Due to the increased number of functions, the function digits were separated into two decimal numbers T and Fn, ranging from 0 - 3 and 0 - 7 respectively, rather than one decimal number ranging

/from

from 0 - 31, thus aiding the human memory akin to the letter system in the earlier computer.

In this system, T gave the type of order thus:

0 = control orders

1 = single-length number orders

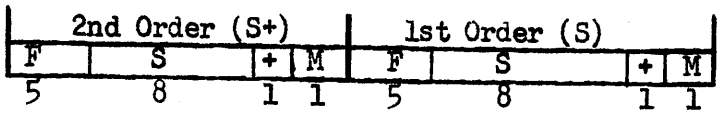
2 = double-length number orders

3 = auxiliary orders,

whilst Fn gave the actual functions,- add, subtract, jump, etc.

It seemed, however, that, in general, if both single- and double-length arithmetic were to be available, the tendency would be to use the latter, particularly if standard subroutines had been prepared for double-length working. Also, it was felt that the absence of the modifier digit was to be deprecated, due to the inconvenience of having to insert an additional instruction before every instruction to be modified, and that this would considerably increase the number of storage locations required in the average programme.

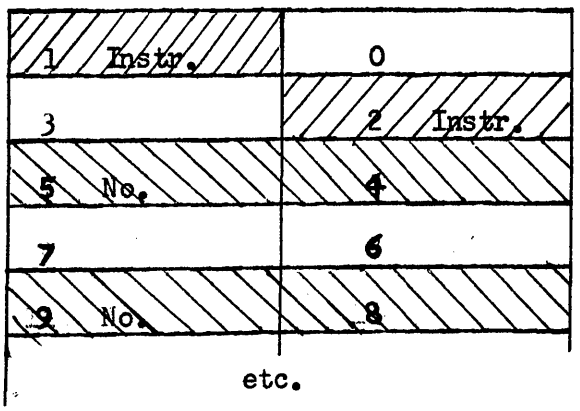
An order code was thus devised, in which the word length was doubled to 30 digits, giving the same accuracy as the double-length working of the previous scheme, but two instructions were placed in one storage location. The B-modifier was reintroduced and the number of functions maintained at 32. This gave an instruction form composed thus:



digits

This limited the size of store to a maximum of $2^9 = 512$ if only instructions were stored, and $2^8 = 256$ if only numbers were stored; in practice, this would mean a storage capacity of about 400 locations. With this system, all numbers would be located in odd number locations, i.e., 1, 3, 5 ..... etc., whereas instructions would occupy all locations taken in sequence, 0,1,2, 3,4,... etc., as shown below:

Instruction in each space or number in any pair



It might seem more logical to interchange the columns in the above diagram but for the fact that the first order of a pair is on the right when transferred to the control register, unless a parallel system of transfer were used.

At this time (early 1959) Mr. T.H. O'Beirne of Messrs. Barr & Stroud Ltd. (Glasgow), approached Dr. Gilles in order to ascertain the possibilities of carrying out some work for him in

/the

the digital computer field, in order that the firm might gain experience and enter the computer technique market. Dr. Gilles said that, although he could not help in this matter, the Engineering Department might be interested in view of the fact that the computer to be built there might be held up due to shortage of technician staff, and a meeting was arranged between Mr. O'Beirne and the Author.

It was explained to Mr. O'Beirne that the computer was basically designed apart from details and that, as far as Messrs. Barr & Stroud Ltd. were concerned, it would be mostly constructional, with some development mostly to suit their production techniques. After several meetings, it was agreed that Messrs. Barr & Stroud Ltd. would construct the computer and, as a result, the details of the computer were completed by joint consultation.

5.2 Size of Computer

By this time, it was becoming apparent that, though intended primarily as a teaching device, the computer was acquiring more facilities and becoming much more a small full-scale computer. In this light, the whole question of size was reviewed, as it seemed that the original word length of 15 digits was too small and, also, it was desirable to have more than one B-modifier register; at the same time, however, the cost had to be kept as near as possible to the original estimate.

/After

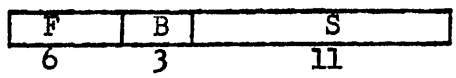
After some considerable discussion, it was agreed that the 15-digit number was too short for reasonable accuracy when making allowance for small differences of relatively large numbers, whereas a 30-digit number was unnecessarily large for most engineering calculations and an intermediate number of digits might be more suitable. It was finally decided on a standard word length of 21 digits,- i.e., a number of 19 digits (approximately 6 - 7 decimal equivalent accuracy) plus sign and overflow digits. The overflow digit is only used in the registers to detect when a number has overflowed or exceeded the permitted range. The normal word length as stored in the main store is only 20 digits long, so that the effective word length is only 20 digits though 21 shift pulses are required for the serial movement of the digit pattern. This word length also considerably eased the instruction content, which was already overfilled, by gaining the extra 5 digits over the original 15.

The decision to increase the number of B-registers beyond one meant an increase in the number of B-register digits, 3 appearing to be a reasonable value. This would allow up to 7 (only 4 initially) B-registers, plus the non-existent zero register for non-modification. It would have been possible, of course to allow only 2 digits corresponding to 3 real B-registers, but this was discarded as there appeared to be plenty of digits available and this method would not allow for future expansion of the number of B-registers.

/It

It was seen from the order codes proposed at the end of 1958 that most of the function numbers were already used and again did not allow for the addition of orders that appeared desirable, if not essential. An additional digit was thus allocated to the function portion of the instruction, giving space for a possible 64 functions with 6 binary digits. The remaining 11 digits (20 - 9) were adequately set aside for the address locations, which would allow a total storage space of 2048 words,- considerably more than had been originally intended.

This gave a final instruction form thus:

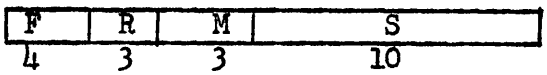


The B-digits of this instruction give the address of the B-register, which will be used either as an instruction modifier or as a register address in instructions which are carried out on the contents of a B-register. In the latter case, the instruction becomes of two-address form which cannot be B-modified; a special modify order is provided, however, to overcome this deficiency.

By this time, the number of registers had increased to seven, consisting of 4 B-registers, 2 registers, M and L, for a (most and least sign) double-length accumulator and a seventh register, D, primarily for a multiplier in multiplication or the quotient in division. This suggested an alternative approach to

/the

the function numbering 0 - 7, in which the seven registers were numbered 1 - 7 (zero being the non-existent B-register). Four binary digits would be allocated to the functions, finally reducing the maximum number of storage locations to 1024 words, which was still quite adequate, and producing an instruction of the form:



in which R is the register to be used. However, this system, as with most systems of coding, had several anomalies and also limited the total number of registers, including B, to seven, which was not the case with the previous scheme. It was decided, therefore, to adopt the earlier form given on page 52.

The decision on this instruction form having been taken, during the following year various schemes of coding were considered before the final order code, reproduced in Appendix IV, was settled. Before this was compiled, however, the overall estimated cost was reviewed because of the increased word length and number of registers. It was found that, due mainly to the falling prices of transistors, as shown by the graph Fig. 3 , the overall cost of the computer could be kept within the original figure. This was possible, providing that the main store was limited to 512 locations and there were only 10 registers, these being B1 - B4, M, L, D, I, S and C, where the nomenclature is as

/before

before, with the addition of I, an Indicator Register for examining the contents of any storage location; S, a store buffer register between the core store and the computer and, C, the control register for holding the present instruction in the control unit.

As with the earlier Valve Computer propounded in Chapter 4, up till now it was envisaged that the computer cycle would be constant,- that is, there would be a fixed time for reading out and obeying the instructions, and also for the times between these operations. It was now realised that this was not essential and further that, in the majority of cases of order modification, it is the address portion of the instruction that is modified rather than the function or modifier address. It seemed logical, therefore, to reduce the length of the B-registers to that of the address portion of an instruction, viz. 11 digits or approximately half a word length. This meant in turn that for the same cost it would be possible to almost double the number of B-registers so that, in fact, 7 B-registers could be provided instead of the original 4. It was considered desirable, however, still to be able to modify an entire instruction, and this was provided for by means of a special instruction. As the scheme of constant time intervals mentioned above had now been broken by means of using half-word operations, the whole question of timing was investigated.

5.3 Speed of Operation

In the earlier scheme discussed in Chapter 4,

/it

it was seen that a clock frequency of about 100 kc/s was reasonable when using thermionic valves. When transistors are used, it is necessary to consider the effect of frequency on the current gain. A number of binary units were constructed, using the GET103 transistors, and these were found to operate successfully up to a frequency of about 120 kc/s; however, when they were connected in the form of a shifting register with unbalanced loading, the speed of operation was limited to about 80 kc/s. The choice, therefore, lay between a lower clock frequency of about 50 kc/s or the use of higher frequency transistors; as the cost of the latter would have been quite considerable, it was decided to reduce the clock frequency as speed of operation was not the primary feature of the computer.

As was seen in Section 5.2, it was decided to have variable operation times to accommodate full- and half-word length operations and, as the clock frequency had been reduced, it seemed reasonable to investigate the possibility of further reduction in operating time by having minimum operation-times for all functions.

5.3.1 Minimum Timing of Operations

The basic machine rhythm is as shown in Fig. 4, in which it is seen that there are 4 periods, IWP, IWA, OWP and OWA, these being a preparation and action period of each of the instruction word and operations word respectively.

/During

During IWP the instruction, PI, whose location is given by the control counter, is read into the S-register and, if the order may be B-modified, the B-digits of the instruction are transferred to a 3-stage, B-address (BA) register. The contents of the BA-register are then tested to see whether or not the instruction is to be B-modified, according to whether the contents of the BA-register are not-zero or zero respectively. As was pointed out earlier, it would also be possible to modify the whole instruction by means of the previous instruction function [17]*. Thus we see that the action time, IWA, can have three durations:

(a) Short (Fast), corresponding to a parallel transfer from the S-register to the C-register, if there is to be no modification.

(b) Half-word duration, corresponding to normal address modification by a B-register or function [16], or,

(c) Full-word duration (Normal), corresponding to complete modification after function [17].

The corresponding IWA durations are 3 pulses, 14 pulses and 24 pulses respectively, as shown in Fig. 5. These pulses consist

/of

---

* Numbers given in square brackets, thus [ ] refer to the Final Order Code function numbers given in Appendix IV.

of two groups,- a fixed group of three pulses following a variable group of Shift Pulses (S) equal to 0, 11 and 21 respectively for the above three durations; thus all instructions are transferred to the C-register in the minimum time.

During OWP, by which time the function of the PI has been set up, unity is added to the control counter preparatory to selecting the next instruction and then the number in the store selected by the PI is read into the S-register. If reading of a number is required and if the order is non-B-modifiable, the B-digits are now transferred to the BA-register to select the B-register for B-instructions. As with the instruction period, the action time varies according to the function as shown in Fig.5, there being five possibilities with the durations given in numbers of pulses as before:

(a) Short (Fast) orders, which involve only parallel transfers (3 pulses).

(b) Half-length orders, involving the B-registers (3 + 11 pulses).

(c) Normal full-length orders (3 + 21 pulses).

(d) Double-length orders associated with the double-length accumulator (3 + 40 pulses).

(e) Variable (Long) length orders, such as multiplication, etc. (Variable number of pulses).

5.4 Order Code

The complete list of functions is given in Appendix IV and, in addition to the basic functions of transfer of numbers to and from the registers and the main store, addition, subtraction and logical operations, the following instructions are considered worthy of special mention.

5.4.1 Multiplication [56]

Because of its relative importance, it was decided to build in an automatic multiplier rather than to use a subroutine, as was used in some of the earlier machines, which would also have to occupy valuable storage space.

The two most common forms of automatic multiplier, which are described below, are (a) the serial multiplier of the form shown diagrammatically in Fig. 6, and (b) the parallel type shown in Fig. 7. Unfortunately, these multipliers only handle positive numbers, a correction being introduced by means of a programme subroutine when negative numbers are also occurring.

The mode of operation of the serial multiplier is one of continued addition, as will be described with reference to Fig. 6, which does not show the auxiliary connections to the store or the sources of shifting pulses. It is assumed that the multiplicand and multiplier are in the registers shown and

/that

that the accumulator is initially cleared. The two-gate shown* is controlled by the digit in the least significant end of the multiplier register and it is arranged that, if this digit is zero, the gate is closed, whereas if it is unity, the gate is opened.

Let us assume that the least significant digit of the multiplier is unity so that the two-gate is open; a train of shift pulses causes the contents of the multiplicand register to be passed through the two-gate to the Adder, the other input of the Adder being supplied from the most significant half of the double-length accumulator (M.S.) to which the same shift pulses are applied, the resulting sum being returned to the input end of the accumulator. At the same time, the multiplicand is recirculated back into its own register to be used again.

A single right shift pulse is now applied to the multiplier register and double-length accumulator so that the least significant digit of the product enters the least significant half of the accumulator (l.s.) and the next most significant digit of the multiplier controls the two-gate. If this happens to be a zero, then the two-gate is closed and the multiplicand is not added to the accumulator contents. The process continues for the entire word length of the multiplier,- i.e., it requires w cycles for a word length of w digits, and such a method is slow. Some speeding-up can be made by

/arranging

---

* See the list of symbols in Appendix II.

arranging that a zero in the multiplier causes an immediate right shift until a one occurs, thus saving the unnecessary addition times.

The faster method of the parallel multiplier shown in Fig. 7 is to be preferred. The multiplier is stored in a register (not shown) whose output is supplied to w two-gates, one for each digit $R_1, R_2 \ldots R_{w-1}, R_w$ of the w-digit multiplier, and the multiplicand is supplied serially at input D. It is seen that the output of each two-gate is added to the unit-delayed output of the next more significant adder, the product appearing at the output of the least significant end adder, P. The mode of operation is best seen by means of the table of Fig. 8, in which a 'c' represents a carry stored in the adder unit when the two 4-digit numbers are multiplied together. It will be observed that the operation takes a time of only two word-lengths; on the other hand, it requires far more equipment, there being (w-1) adders and unit delay elements.

With these limitations in mind namely, positive numbers only and either slow speed or high cost, other methods were investigated. The answer appeared to be found in a scheme suggested by Drs. A.D. Booth and K.H.V. Booth and described on pages 46 to 48 of their book (Ref. BB), which gives a serial multiplier whose output is a correctly signed product of two signed input numbers. The method consists basically of

/comparing

comparing the two least significant digits of the multiplier register and either adding or subtracting the multiplicand to the partial product in the accumulator if these two digits differ, or leaving the partial product unaltered if the two digits are the same. The contents of the multiplier register and accumulator are then shifted one place to the right as with the serial multiplier given on page 58, and the process repeated until the multiplication is completed; the time is similar to the earlier method but negative numbers can be handled.

Later, J. Leech of the University Computing Laboratory, suggested a faster method in which the three least significant digits of the multiplier register were compared and the multiplier and product shifted two digits at a time, thus approximately halving the multiplication time without much increase in cost. A full description of this method, as adopted, is given later in Section 6.6.

5.4.2 Division [58]

Less attention seems to have been given to automatic dividers in the past, apparently mainly due to there being relatively few division operations as compared to other operations. In most of the earlier machines, division was programmed or else the so-called trial-and-error method was used, which again only operates with positive numbers. In this system,

/the

the divisor is compared with the partial remainder and, if it is less than the partial remainder, a subtraction is performed and unity entered into the least significant end of a quotient register. If it is greater than the partial remainder, then no subtraction is carried out and a zero is entered into the quotient register; the partial remainder and quotient are then shifted one place to the left and the process repeated. The only convenient method of carrying-out the test is by subtracting the divisor from the partial remainder, testing the sign of the remainder and, if negative, adding the divisor back into the partial remainder accumulator, as shown in the schematic diagram of Fig. 9. The method is slow, taking a time of anything from w to 2w word-lengths duration, according to the number of re-additions taking place.

Again a superior method was found on pages 48 - 50 of Ref. BB, using a non-restoring method, thereby avoiding the subtraction followed in some cases by addition as well. In this method, a direct comparison of the sign digits of the divisor and partial remainder dictates an addition or subtraction, thereby reducing the time to w word lengths for all division operations; also, the method can be used equally well with positive and negative numbers. This method was adopted with a modified form of rounding-off, as described later in Section 6.7

5.4.3 Shift Operations [36] to [39]

In any shift operation a negative value of shift is permissible and is interpreted as a positive shift in the opposite direction. Thus, strictly speaking, [37] and [39] are redundant but it was considered convenient to have both left and right shift functions with positive values rather than one shift operation with positive and negative values giving the direction of shift, as this is less likely to give programming errors. Care is taken to observe that the sign digit is preserved in right shifting and, if excessive left shifting takes place, the computer normally stops due to the number having overflowed the permissible range.

The Logical shifts are identical to the mathematical shifts, excepting that overflow is ignored with left shift and the most significant end of the accumulator fills with zeros with right shift, irrespective of any original sign digit; that is, the digit pattern is literally moved, the remaining portion being occupied with zeros.

5.4.4 Normalise Operation [35]

This instruction was included to speed up floating-point arithmetic, which will normally be used. Though the computer is a fixed-point machine, floating-point arithmetic can be programmed as indeed can the function of normalising or standardising, but it was considered worth engineering the function which would speed up this operation. As it will be

/seen

seen later, it takes only 1 to 3 word lengths against a considerable programming time.

The operation of normalising converts a number in the double-length accumulator into standard form, $x.2^y$ where x lies in the range $\frac{1}{2} \leqslant x < 1$ or $-\frac{1}{2} > x \geqslant -1$ and y can lie in the range $2^{19} > y \geqslant -2^{19}$. This is achieved by a process of shifting the contents of A until the number is normalised and then altering the value of y in the D-register by subtracting the amount of shift carried out.

As the function is addressless, i.e., the store is not used, the address portion of the instruction is used as an overshift indicator. It may be desirable in some problems to specify a maximum accuracy by indicating that a small number can be regarded as zero. This is equivalent to noting a shift of the accumulator greater than any value N specified in the order and should be followed by a jump instruction [24] which could then direct the clearance of the contents of the accumulator, as otherwise the computer will stop. If no limit is required on the amount of shift, this can be accommodated by putting $N \geqslant 40$, the maximum possible shift, or more conveniently, N = 0, which is interpreted as N = 40; any other meaning of N = 0 would be useless and this saves punching and reading-in time, as the zero does not have to be entered.

/5.4.5 Input and Output Operations

5.4.5 Input and Output Operations

(i) Input [10]

When reading-in information from a paper tape, the data is temporarily held in a buffer store associated with the input reader. When the function [10] is called, the five digits stored in the buffer, if ready, are transferred in parallel to the least significant end of the store register, from whence it is transferred into the main store and the specified B-register; the remaining portion of the B-register and storage location being cleared. If the buffer store is not ready to give out its contents, the computer waits until the buffer store is ready and then carries out the operation.

(ii) Output [20]

In a similar fashion to the input, all output data is passed through a buffer store so as not to slow down the computer more than necessary. When the function [20] is called, the number in the location specified is transferred to the S-register and the five digits at the least significant end of the register are then transferred to the output buffer store if already cleared; the output punch is then activated, the computer proceeding to its next instruction. If the buffer store is not empty, then the computer waits until it is free to proceed. If the reel of tape on the punch ends, the computer stops and indicates "tape-out" so that a new reel can be

/inserted

inserted and, upon operation of the start button, the computer continues to operate,- this being one of the Internal Stops mentioned in Section 5.4.1 (d).

(iii) Terminal Unit Change Orders [28] and [29]

In practice, it may be desirable to have alternative input and output units and, to enable these units to be changed, these functions are provided. [28] connects the specified terminal equipment, the "address" of which is given by N, odd numbers being used for input devices and even numbers for output devices. The computer can thus detect whether the peripheral equipment is an input or output device. In the former case, the computer automatically changes over from one unit to the other, as it is only possible to have one input unit at a time. It is permissible, however, to have a number of output units connected at the same time, so that, in this case, the selected output unit is connected in parallel; the order [29] being used to disconnect a previous output unit. When initially set, the computer automatically selects output unit, 0, and input unit 1, which can then be changed as required.

(iv) Decimal Input to the Store

A telephone dial is provided for putting into the store any decimal integer up to the permitted maximum value of $(2^{19}-1)$. When the dial is operated, each digit of a decimal number adds unity into the least significant end of the M-Register. The digit dialled in is then automatically multiplied by 10 before

/the

the next decimal digit is added in; this process continues until the whole number is dialled in, the result being correctly converted into binary form. If a negative number is required to be put in the store, this can be achieved by putting in the modulus of the number and following it by a subraction from zero, using function [44] and the cleared accumulator.

5.4.6 Modify Orders [16] and [17]

In addition to the normal B-modification these two functions are provided for the following possibilities:

(a) Modification of a non B-modifiable order (functions [0] to [15] which can have the address portion modified by any value of N within the permitted range by preceding the present instruction by the modify instruction [16] . Alternatively, the instruction address may be modified by the contents of any storage location by preceding the present instruction by the modify instruction [17] .

(b) Modification of any part of any instruction may be carried out by preceding the present instruction by the modify instruction [17] . The present instruction may be doubly modified, that is B-modification and [16] or [17] modification may take place simultaneously.

5.4.7 Jump Instructions

In addition to the more obvious jump instructions, there are added two counting jumps [14] and [15] ,

/excess

excess jumps [24] and [25] and the change store jump [27]. In the case of the count jumps 1 or 2 is subtracted from the specified B-register before testing if its contents are zero. Normally, the single count [14] would be used, but the double count [15] is useful for double-length working if one wishes to use 40 digit numbers or floating-point arithemtic where the exponent and mantissa will normally be stored in adjacent store locations.

Functions [24] and [25] are used to test whether an arithmetic operation has caused a number to exceed the permissible range, as mentioned in Sections 5.4.6 and 5.4.7 respectively. If these or certain other functions do not follow the excess, the computer stops and may be re-started by the operator.

Function [27] is an unconditional jump like [26], only in this case the reading from the store is transferred to the Initial Order store (see below) or vice versa.

5.4.8 Stops

There are basically four forms of stop:

(a) Absolute Stop called by an instruction [0,0,0], i.e., F-, B- and S-digits all zero which will normally only occur due to an error, such as an incorrect jump order to an unused part of the main store. Under these circumstances, the computer

/can

can only be re-started by means of the Initial Set button (see in Section 5.5).

(b) Programme Stop called for by an instruction [0,0,N] where the address portion may have any value other than zero and, in fact, a number of programme stops can then be easily identified by this number N. The computer may then proceed by operation of the Run button.

(c) Programme Optional Stop called for by an instruction [0,B,N] where B must have a value other than zero, in which case the instruction is either carried out as at (b) or ignored, according to the setting of a switch on the control panel.

(d) Internal Stops. A number of conditions inside the computer can cause a stop, the reason for which is indicated on the control panel, so that the operator can decide on whether to return to the initial orders or to continue from the stopped condition (see Section 6.15).

5.5 Initial Orders

A set of initial orders is required in order to read-in and process the programme punched on the paper tape. Originally, a fairly simple system based on that given in Ref. W93 was to have been adopted and prewired either on a uniselector, which could then be transferred at will into the

/store

store, this being the more common method, or alternatively, by prewiring a part of the store,- a so-called wired programme as used in some large computers for commonly used subroutines[57,58].

After much collaboration, Mr. Leech and Mr. O'Beirne produced a very much superior set of initial orders given in Appendix V, consisting of 96 orders. This rather eliminated the idea of using uniselectors and it was decided to build a set of store memory planes to contain the initial orders and arrange that, when in use, instructions were taken from this auxiliary store but, at the same time, the processed programme could be written into the same location (0 - 95) of the main store. By means of the jump instruction [27] mentioned above, it was then possible to transfer reading from any available location in the one store to the other store prior to or during the actual programme. This virtually increased the size of the store but did not seriously affect the cost, as the reading and writing equipment was common and in the auxiliary store, cores are only required where there is a "one" in the instruction.

6. Transistor Computer - Logical Design

6.1 General

Before proceeding with the details of the design, it seems fitting to give a simplified general block diagram, Fig. 10, which shows the basic units and interconnections of the final computer, without showing the control gates.

It will be seen to consist of six full-length registers, one of seven half-length B-registers, a third-length register, two counters, two stores, three decoders and a logic unit. Three main highways carry the information between the units, these being designated H1, H20 and H2I. The outputs of the registers are normally connected to the Output Highways and the inputs to the registers to the Input Highways through gates. The main exceptions to this are parallel transfers between the Store, Store Register and Control Register, Input and Output buffer registers and the handswitches.

6.2 Basic Timing

Before discussing the logical design of the computer, it is necessary to consider the basic timing arrangements.

The basic unit of time is conveniently a 7-digit pulse-train, this being one-third of a word length (21 digits) and gives a total IW or OW normal period of 28 digit times, as was seen in Fig. 5. In the other cases, there was always a 4-pulse

/period

period at the beginning and a 3-pulse period at the end of W. It is desirable to be able to refer to these pulses and they are therefore assigned the nomenclature given at the foot of the diagram of the main timing unit, Fig. 11. The first letter designates the word period in which the pulse occurs and the second letter its alphabetical position within the word beginning with A and ending with Z. E to W do not exist, but are replaced by the shifting pulses S, whose number depends on the function, and W is used to designate the complete word. The C-pulse does not exist in the computer, allowing a double-pulse time between B and D for the read-write cycle at B. Although the 4- and 3-pulse groups of a word are separated by the S-pulses, the 4- and 3-pulse groups of adjacent words form a complete 7-pulse group extending from OX to ID or IX to OD.

The main timing unit is required, therefore, to count up to a maximum of seven groups, each of seven pulses. These are provided by means of two timing chains, T1 to T3, providing the seven-pulse times and T4 to T6 the groups of seven pulses, the entire chain being supplied by the external clock pulses, CP. The method used to select the number of groups is to gate the length required F, H, N or D with the appropriate conditions of T1 to T6, as shown by the gates in the centre position of the figure. The counter is then reset to zero by

/means

means of the monostable pulse generator, MS, preparatory to the next cycle. The method used to produce the individual pulses required between the S-pulses is indicated at the top of the diagram without showing the details of the matrix, which follows standard practice.

6.3 Basic Instruction Periods

The instruction period may be one of three durations, as mentioned in Section 5.3, and is fast, with the exception of a modified instruction when the duration is a half-word length or when the special S-modify instruction has preceded the present instruction, when the duration becomes full length. The process is described as follows in conjunction with the flow and logical diagrams of Fig. 12, assuming an operation period to have just been completed:

During the first three pulse times IA - IC the instruction is parallel transferred from the store into the S-register to be ready for pulse ID to test whether the function is B-modifiable ($F > 15$) or not and, if so, to parallel transfer the B-digits of the P.I. into the three-stage BA-register, which has been cleared with IA. At this point, three choices are available, as shown in the flow diagram:

(i) Unmodified instruction, in which no action takes place (Fig. 12(b)).

(ii) Short modification in which the address portion

/of

of the instruction is modified by serially adding together the contents of the S1-register, a B-register and/or the C1-register, this being carried out during the IS-period of 11 pulses duration, as shown in Fig. 12(c) (dotted lines).

(iii) Long modification, in which the instruction is modified by serially adding together the contents of the S-register, the C-register and, if required, a B-register, this being carried out during the IS-period of 21 pulses duration, as shown in Fig. 12(c), (dashed lines). In this case, a complication arises due to the short length of the B-register. It is arranged, therefore, that, at the end of the eleventh shift pulse, the BA-register is cleared by the HT waveform thereby removing the B-register shift pulses. After this stage, the true P.I. is in the S-register and at IX the C- and BA-registers are cleared prior to transferring, at IY, the contents of the S-register into the C-register and, if the function number is less than 16, the B-address to the BA-register before transferring to the operation period at IZ.

6.4 Basic Operation Periods

The timing of the operation period is more variable than the instruction period, due to the different facilities required. However, there are three groups of operations, each of which have the same duration, and a fourth group whose operation period is essentially variable, as shown by Table I.

TABLE I

Grouping of Functions according to Type

	Group 1, Fast	Group 2, Half	Group 3, Normal	Group 4, Variable
Basic		1 - 9	40 - 55, 59	
Special	0, 12, 13, 16, 17, 20-22, 24-29, 61.	10, 14, 15.	23, 31, 32, 60.	33-39, 56, 58.

It is proposed to deal first with the basic functions given in Groups 2 and 3 and then the special cases will be considered individually.

As with the instruction period, the basic timing pattern of these first groups is similar and is described with the aid of the flow diagram, Fig. 13. When the function appears from the function decoder at the end of IZ, it is immediately tested for impermissibility (see Section 6.15) and, assuming that it is permissible, OA is used to clear the S-register. OB advances the control counter by unity and also carries out a reading operation if a number is to be extracted from the store. This is followed by a series of shift pulses, OS, and in the case of certain functions, the resulting number is written into the store, the operation being initiated at OX. OZ then transfers control to the next IW period.

(1) B-Register Functions [1] to [9]

These functions use the B-, C- and S-registers only and, in the latter cases, only the least significant stages, thus requiring only 11 OS-pulses. The logic diagram is divided into two for convenience, according to (i) functions requiring input to the S-register or the C-register, Fig. 14(a), and (ii) the other functions requiring the S-register, Fig. 14(b). It should be noted that the parallel transfers to and from the store have been omitted to avoid confusion, only the flow of pulses during OS being shown.

(2) M-Register Functions [40] to [49]

These functions use the full length of the C- and S-registers together with the M-register, and require 21 OS-pulses. Two logic diagrams are produced, according to (i) functions producing a change of the store content, Fig. 15(a), and (ii) M-arithmetic operations, Fig. 15(b). In the case of function [40] the L-register also receives shift pulses to clear its contents.

(3) D-Register Functions [51] to [55] and [59]

These are similar to the previous group and the logic diagrams are given in Fig. 16. Due to the short length of the C1-register, its S-pulses are stopped after the eleventh by means of the HT waveform from the timing unit.

/The

The remainder of the functions can now be considered individually, due to their variable nature.

6.5 L-Register Operations [31] to [34]

The L-register is normally regarded as the least significant half of a double-length accumulator with the M-register as the most significant half. This means that the length of the L-register is only 19 digits as no sign digit is required and, when transferred to the store, the number is considered as a positive value; that is, the sign digit is always stored as zero. From design considerations, this means only 19 shift pulses are required instead of the normal 21 for the L-register, whilst 21 are still required for the S-register. Also, when carrying out addition or subtraction, a carry or borrow may be propagated into the M-register, requiring a further 21 pulses. Due to these differences, it is convenient to consider the L-register functions separately.

(1) Copy the L-Register into the Store [31]

This is a normal length operation of 21 OS-pulses, as given by the flow diagram, Fig. 13, and to enable the contents of the short L-register to be correct at the end, two additional stages, cleared at OB, are added in the recirculation path, as shown in Fig. 17(b). As will be seen later in Section 6.9, one of these additional stages is required in the normalise operation so that only one stage has had to be added to produce this delay.

(2) Read the Store into the L-Register [32]

When reading a number from the store, the sign digit is placed throughout the M-register in addition to placing the number in the L-register. This is most conveniently carried out by initially clearing the M-register and then setting all the stages to unity, if the sign digit of the number when transferred into the S-register is unity. Thus the number of OS-pulses required is only 19 to fill the L-register and the two pulses, OX and OY, are used to complete the recirculation of the S-register, as shown in Fig. 17.

(3) Addition and Subtraction into the L-Register [33] and [34]

As was mentioned above, the complication here is that, at the completion of the addition or subtraction of the number into the L-register, a digit may have to be propagated into the M-register and this would almost double the operation period to 40 pulse times. If small integers are being used without this overflow into the M-register, this would mean a considerable waste of time. To avoid this, these functions are carried out in a two-phase manner, the second phase taking place only when required; thus the number of OS-pulses may be either 19 or 40, as shown by the flow diagram of Fig. 18. A counter unit is used as a phase selector to change over the addition operation from the L-register in phase 1 to the M-register in phase 2.

6.6 Multiplication [56]

The method of operation is basically one

/of

of testing the digits of the multiplier and carrying out addition or subtraction of the multiplicand to the partial product in the double-length accumulator, according to Table II.

TABLE II

Multiplication Rules

Borrow Digit	Multiplier Digits		Action	
b	m	l		
0	0	0	Do nothing	
	0	1	Add	
	1	0	Subtract after one shift	Set borrow to unity
	1	1	Subtract	
1	0	0	Add	Set borrow to zero
	0	1	Add after one shift	
	1	0	Subtract	
	1	1	Do nothing	

Note: The m and l digits of the multiplier are the most and least significant digits of the least significant pair of the multiplier digits.

The following results are observed:

(a) The borrow digit of the nth operation is identical to the m-digit of the (n-1)th operation.

/(b)

(b) Having tested the 3-digit pattern, if m = l = b no addition or subtraction is required and the cycle becomes a Fast operation.

(c) If m ≠ l but l = b, a pre-shift pulse is used before the addition or subtraction but, in any case, a total of two shifts is necessary except in the last cycle, where there is only one shift. Thus optional pre-shift and post-shift pulses are required in addition to a compulsory post-shift pulse.

It is now possible to draw up the required flow diagram for the whole operation and Fig. 19(a) shows that there are 8 similar intermediate cycles, together with slightly modified first and last cycles.

In the intermediate cycles the partial product in the accumulator may be optionally preshifted one place to the right at OD, this being remembered by the Shift Memory bistable unit, shown in Fig. 19(b). At the same time, the A-counter is advanced by two, counting the cycle being performed. If the m, l and b digits of the D-register are identical, the operation is made Fast and, if not identical, an addition or subtraction of the S-register to the partial product in the M-register is carried out. After this, a post shift pulse, OX, is applied through the post shift gate, PSG, to the accumulator if it has not been pre-shifted; the second shift of the

/accumulator

accumulator then takes place, using OY. The contents of the D-register are also recirculated two places to the right by OX and OY and also sent to the "D-borrow" store, D_o, only the last digit of the pair being retained. This completes the basic cycles 2 to 9.

In the first cycle, an additional operation is carried out, namely, OB clears the whole accumulator, adds two to the D-borrow store and A-counter, causing the operation to become Long. At the same time, the contents of the selected storage location are transferred into the S-register for use in the multiplication operation.

Finally, in the last cycle, OY is inhibited from producing a shift of the accumulator, due to the closing of the accumulator 2nd shift gate, A28G, but not of the D-register; this leaves the contents of this register ready for another multiplication if required, particularly in the case of decimal to binary conversion.

Before leaving the operation of the multiplier, it is necessary to explain the method used of cycle counting in the arithmetic counter, shown in the logical diagram of Fig. 19(b). The total count required is 10 cycles and the choice lies between beginning with zero and detecting +10 or alternatively, beginning with -10 and detecting zero. As such, there is little to choose between the two methods. It must be

/remembered

remembered, however, that the counter is also used in other operations (division and shifts) and, in practice, it is more convenient to detect one condition rather than a whole range of values; the latter method was therefore adopted. Furthermore, as will be seen in the next Section, it is necessary in division to detect the 21st and 22nd cycles, so that beginning with -21 in the counter, the conditions of 0 and +1 are reasonably easily detected. To avoid a different initial setting of the counter for multiplication, a double count from -21 with an additional double count gives a final counter value of +1 for the required 10 cycles; thus -21 is set into the A-counter at IB prior to every operation.

6.7 Division [58]

As mentioned in Section 5.4.2, the divider uses a non-restoring procedure similar to that described in Ref. BB (pages 48 - 50), the rules being as shown in Table III, the results of testing the overflow digits, OF, of the accumulator and S-register.

TABLE III

Divider Rules

Like OF digits	Subtract C(SR) from C(MR) and add unity to C(DR)
Unlike OF digits	Add C(SR) to C(MR) and add zero to C(DR)

/It

It will be seen from the flow diagram, Fig. 20(a), that there are 19 standard cycles and three (1st, 21st and 22nd) with slight modifications. In the standard cycle, the overflow digits of the accumulator and store register are compared at OD which sets a bistable unit, as shown in the logic diagram Fig. 20(b), to add or subtract the contents of the store register to or from the contents of the M-register and, at the same time, add unity or zero to the contents of the D-register using the D-half adder during the OS period. Also at OD, unity is added to the A-counter recording the cycle and, finally, at OY, the contents of the accumulator and D-register are shifted one place to the left.

In the first cycle, the only difference from the standard cycle is the initial clearing of the D-register at OB and reading the required number from the store into the S-register. In the last division cycle (the 21st) the left shift operation is omitted from the standard cycle, due to AC6 changing to zero and closing the left shift gate, LSG. The final cycle (number 22) is used for rounding-off the quotient, the main difference being that, after the standard addition/subtraction operation during OS, the contents of the accumulator are shifted back one place at OY, due to the extra division cycle, and also the overflow digit of the quotient in the D-register is changed.

/As

As with all long operations, a temporary overflow condition may take place, but this does not have any effect until the last cycle when it may become permanent. In this case, the computer will be stopped, as the number range has been exceeded and information is lost.

6.8 Shift Operations [36] - [39]

The duration of the shift operations is variable, ending when the required shift is completed. The only difference between the operations is that, in the case of the logical shifts, the sign digit is not repeated in the case of right shifting nor is there any recording of overflow that might take place in left shifting.

Apart from this, all shifting operations are basically the same and the desired amount of shift given by the "address" digits N may be positive or negative, a negative value being interpreted as a positive shift in the opposite direction. In this sense, both left and right shift instructions are not strictly required but are provided more for convenience of programming.

The method adopted is basically one of counting the shifts using the A-counter, beginning with the complement of the required count and finishing when zero is reached; negative values are actually represented as modulus 64 without a sign digit being present.

/Fig.21

Fig. 21 shows that, in addition to the counting, some organisation has also to be carried out. Firstly, if the required shift is positive, the accumulator is shifted one place in the opposite direction to that given in the instruction. This is necessary due to the number N being digitally complemented when transferred to the A-counter, which would result in an error of one in the count. If the shift is a negative value, the numerical portion is digitally complemented at this time, no pre-shift being required due to the double complementing.

At OD the six least significant digits of N are transferred in parallel to the A-counter, being automatically digitally complemented at the same time. If, however, in error or otherwise, a shift of greater value than 38 is requested, -48 is conveniently transferred to the A-counter instead, as the operation is automatically limited to 40 by the main timing unit.

If a right shift in excess of 38 results from the above, it is unnecessary to carry out the shift which would only reduce the contents of the accumulator to zero or $-2^{-38} \neq 0$. Therefore, the operation is made fast, as shown at (i) in Fig. 21(a) and the accumulator is cleared at OX.

In all other cases, alternative (ii) is carried out, in which case shifting takes place until the content of the A-counter is zero and the main timing unit is reset to give OX.

/In

In the case of logical shift instructions, the overflow digit of the accumulator is made zero at OD to ensure a true logical right shift and, at OX, it is made a copy of the sign digit to prevent an overflow condition from being recorded.

6.9 Normalise Operation [35]

The normalise operation shifts a number x in the accumulator until it is in standard form and then subtracts the required shift from the value of y in the D-register. There are thus two operations to be performed:

(i) Shifting of x until it is in the specified range (phase 1).

(ii) Subtracting this amount of shift from y (phase 2).

A third phase may also be required in exceptional cases, due to rounding off; the operation will be clarified with the aid of Fig. 22.

Phase 1

The operation commences by clearing the carry stores at OA and the N-register at OB. Then, as seen in Fig. 22, at time OD, the six least significant digits of the C-register containing the maximum acceptable shift, are transferred in parallel to the A-counter, being automatically digitally complemented, as with the shifting operations, and at the same time, the contents

/of

of the accumulator are shifted one place to the right. This is necessary in order to allow a number which has temporarily overflowed to be normalised, requiring one shift to the right, as shown below:

00.110 ..... (3/4)

\+ 00.110 ..... (3/4)

01.100 ..... ($1\frac{1}{2}$) normalised by a single right shift, thus

00.110 ..... x 2^1

In this case, no left shifting takes place, as the number is now normalised and the previous exponent requires unity to be added to it.

In any other case, the number will still not be normalised and can only be normalised by left shifting, as x must be in the range $\frac{1}{2} > x \geqslant -\frac{1}{2}$. This shifting takes place during OS (maximum 40) until the number x is normalised, which must occur unless the content of the accumulator is zero. At OY, two courses are open:

(i) If the number x had been zero, the normalise operation is complete and OZ transfers to the next IW period.

(ii) If the number x were not zero, then it will now be normalised and OY changes the operation from phase 1 to phase 2, using the phase selector and also

/transfers

transfers the contents of the A-counter to the N-register, the latter having been cleared at OB.

Phase 2

In this phase, the actual shift count is subtracted from the previous exponent in the D-register and the most significant half of the accumulator is rounded-off by adding the most significant digit of the L-register to the contents of the M-register.

It should be noted that, at the beginning of this phase, the contents of the N-register equal the actual shift count added to the digital complement of the six least significant digits of the C-register or, in symbols -

$$C(NR) = -C(CR) + 1 + SC$$

To obtain the correct exponent in the D-register, we require to subtract the (actual count - 1) from the contents of the D-register, the one being due to the initial right shift at OD of phase 1, that is

$$C(DR)' = C(DR) - (SC - 1)$$

This is obtained by subtracting from the D-register the sum of the contents of the C- and N-registers, thus

$$\begin{aligned} C(DR)' &= C(DR) - \left[C(NR) + C(CR)\right] \\ &= C(DR) - \left[-C(CR) - 1 + SC\right] - C(CR) \\ &= C(DR) - (SC - 1) \end{aligned}$$

as required.

/These

These two operations are carried out during OS, as seen in Fig. 22(b), the M-register half-adder carry store having been previously set equal to the most significant digit of the L-register at OD. Due to there being only 11 N-digits in the C-register, the shift pulses supplying this portion of the register are stopped by means of the CRN selector, controlled in turn by the waveform HT generated in the main timing unit.

At time OY, the resulting number in the M-register will usually be still normalised and the operation is complete. If the number in the M-register were just less than +1 or $-\frac{1}{2}$ at the beginning of phase 2, then the addition of the most significant digit of the L-register, if unity, to the M-register will cause the contents of the M-register to overflow and to require re-normalising, this being carried out in phase 3.

Phase 3

Considering the two possible conditions:

Case (a) x = just less than 1:

x = 00.111111111111 and I_m = 1 .....

gives

x = 01.000000000000 requiring 1 right shift

thus

x = 00.100000000000 normalised and exponent increased by unity.

Case (b) x just less than $-\frac{1}{2}$

x = 11.011111111111 and I_m = 1 .....

gives

x = 11.100000000000 requiring 1 left shift

thus

x = 11.000000000000 normalised and exponent decreased by unity.

It is observed that, in case (a), the most significant or overflow digit after the addition of I_m = 1, is zero and a single right shift is required to normalise the number, unity being added to the exponent; in case (b) the opposite is so, namely, the overflow digit is unity, a left shift is required together with a further reduction of unity from the exponent.

It will be seen in the flow diagram, Fig. 22(c), that the operation required is detected and partially carried out at OD, after which the contents of the N-register are subtracted from the exponent in the D-register, whilst the contents of the M-register are recirculated through the M-half-adder, the carry store having been cleared at OD. This causes no harm and simplifies the circuitry, as the operation then becomes identical to phase 2 during the OS period. The correctly normalised number now stands in the M-register and the operation ends on OZ, due to the removal of the Long operation signal at OY.

Referring back to phase 1, it will be seen that

/the

the digital complement of the number N in the C-register is the starting point for the A-counter. The purpose of this is to note any overshift, as was explained in Section 5.4.4. If the A-counter reaches zero, this means that the overshift specified has been reached and a binary unit is changed over. Unfortunately, this is complicated by the rounding-off in phase 3 if case (a) occurs after just reaching the overshift value, as the one right shift would mean that the nett shifting had not reached the overshift condition. This is overcome by providing a means of changing the binary unit back again, as shown in Fig. 22(d).

6.10 Input and Output Operations

Due to the different modes of operation, the three methods will be dealt with separately.

(1) Tape Input and Output [10] and [20]

Input and output of information, using 5-hole paper tape, consists of a parallel transfer during OD to or from the 5 least significant digits of the S-register and a buffer register, as shown in Fig. 23. In the case of an input order, the address portion of the S-register is transferred to a B-register and also recirculated in preparation for transferring into the store itself at OX. In the case of an output order, the number is first read from the store into the S-register during OB, ready for the transfer, in OD.

/In

In addition to the above processes, two precautionary actions have to be carried out. These are (a) that the external unit is ready when required, and (b) that the paper tape has not run out of the unit in use. These are both tested during OA, as shown in Fig. 21 and, if satisfactory, the above process is carried out. If not, then one of the following actions takes place.

If (a) the unit is not ready, the content of the Store location is read into the S-register at OB, if an output order, and at OD the function decoder is closed to make a Fast operation period, in which the control counter is prohibited from advancing and the unit is repeatedly tested at OY until found ready, when the operation is completed as before and the Long selector is changed back to off at OX.

If (b) the tape has nominally run out, the computer stops but can be re-started after inserting a new reel of tape or using the tape override pushbutton, which permits the remainder of the reel to be used by setting a bistable unit. This bistable unit is cleared, however, by the Initial Set button (see Section 6.13), so that a later operator will be warned that the tape unit needs to be refilled with tape.

(2) Input/Output Unit Change [28] and [29]

There are two functions available for changing the input or output unit as required, or more strictly speaking, its

/ouffer

buffer register. In practice, only one input device can be read at a time but any number of output devices may be used simultaneously. The address portion of the instruction gives the address of the buffer register, odd numbers (ending in 1 in the binary form) being reserved for input devices and even numbers (ending in 0 in the binary form) being the output devices.

When the function [28] is called, the new device is connected during OD and, if the address is odd, the previous unit will have been disconnected during OB; function [29] is used during OD for disconnecting an output unit. This is clarified by the logical diagram of Fig. 25.

(3) Manual [60] and [61]

With regard to the output function [60] , the store content is transferred in parallel to the S-register and then serially to an Indicator register, which is equipped with an indicator lamp on each stage. The operation is quite normal but is given here to keep all input/output operations together.

In the case of manual input [61] reference will be made to the logical diagram, Fig. 26. The number to be inserted is set up on a row of handswitches and, at time OB, the number is transferred in parallel to the S-register instead of reading from the store. It is then transferred to the storage location specified, during OX, as for any

/normal

writing operation. An instruction may also be set up manually, using the handswitches, in the following manner. With normal running, an instruction is read from the store into the store register during IB (see Section 6.3), which may then be modified before being transferred to the control register. When the instruction is taken from the handswitches, the reading operation is inhibited during IB and instead, the instruction set up is transferred in parallel to the S-register during IB, so that the remaining portion of the IW period is the same as for a normal instruction. Further, the control counter advance is inhibited by the normal/manual instruction switch, which has to be operated to bring in this manual instruction.

(4) Dial Input

This form of input is purely manual and permits the entry into the store of any decimal number within the range of the computer, the number being inserted by means of a standard telephone dial. After an initial link programme the computer stops and the decimal number is entered, beginning with the most significant integer. The integer is dialled and, as the dial returns to its stationary position, unity is added to the contents of the M-register as each pulse of the dial switch is produced. Thus, at the end of the integer being dialled, its binary equivalent

/is

in the register. Upon completion of this integer, the control counter is advanced and a subroutine programme such as that given in Appendix VI is entered to multiply the integer by 10 prior to accepting the next integer. The process is repeated and continued until the entire decimal number is entered. The normal programme is then continued by means of the ordinary start button.

The method of entering a decimal number is now described with the aid of Fig. 26. Before stopping the computer to dial in a decimal number, two storage locations must be given, namely, (1) the location of the reset instruction after the dialling has taken place (link) and (2) the store location into which the number dialled will be placed. Having stopped, the computer is automatically restarted when the dial is used. Two sets of contacts are used, one producing an impulse for each digit of the decimal numbers (DIGIT) and the other producing a pulse at the end of the integer (INTEGER). When the first digit pulse is received from the dial, the carry store of the M-half-adder is set at unity and, during IS, the computer adds this to the previously cleared M-register. At OA, the M-half-adder carry store is cleared and the contents of the register recirculated during OS before the computer again stops at OZ. It will be noted that the content of the M-register is unnecessarily recirculated during OW after carrying out the

/addition

addition of unity during IW. This was done for reasons of economy, as there is plenty of time between digit pulses and this saves the addition of a number of gates.

At each of the digits the above is repeated until the INTEGER pulse occurs at the end of the integer and the C-Counter is advanced by unity, due to the changing of the Dial selector at OX, to enter the subroutine which places the integer in the specified storage location. Again the computer stops and any further integers are treated in like manner. The integer zero in fact produces 10 pulses, so that a test can be carried out to determine when the complete decimal number has been entered. The method of continuing the normal programme is by means of the start button, which effectively enters zero pulses as opposed to decimal "0" with its 10 pulses, the subroutine differentiating between the two values as shown in the Appendix.

6.11 Modify Orders [16] and [17]

These two orders, shown in Fig. 27, are fast and consist of holding a modifier until the next IW period, when it is used as described in Section 6.3

In the case of [16] an N-Mod. bistable unit is set during OB to retain the information that the next instruction is to have its address modified by the contents of the C1-register.

/With

With [17] , anS-Mod. bistable unit is set during OB and the contents of the specified store location are read into the S-register. The content of the S-register is then transferred to the C-register during OX and OY, either at the end of the order or, in the case of single shot operation, when the computer is restarted.

6.12 Jump Instructions

As seen from the order code, there are a number of jump instructions and in all of these orders, when a jump of control is to take place, the contents of the C-register are transferred in parallel to the C-counter during OY, the counter having been cleared at OX, as shown in Fig. 28. In the case of some of the test jump instructions, a shift operation has first to be carried out as will be seen below. The two groups are thus:

1. Fast Jump Instructions [12] , [13] , [21] , [22] , [24] to [27] .

2. Pre-shifted Jump Instructions [14] , [15] and [23].

(1) Fast Jump Instructions

Apart from the above remarks, the only additions in this group are to the functions [24] , [25] and [27] .

In the case of [24] and [25] , if the test is satisfied, it is necessary to carry out an early transfer, as

/the

the tested bistable unit has still to be cleared after this prior to the next instruction, in case the tested condition is not cleared. OB is used to change over the Jump bistable unit if the test is satisfied and the tested bistable unit is then cleared at OD.

Function [27] is an unconditional jump, but requires the store reading facilities to be transferred from one store to the other at OB.

(2) Pre-shifted Jump Instructions

In the case of [14] and [15] a subtraction of integer 1 or 2 from the nominated B-register takes place before the test is made, so that the duration of the operation is half length, as in Fig. 28(b).

At first, it might appear that [23] should be fast, due to its similarity to [13] , but it requires to be a full length operation to enable the contents of the accumulator to be recirculated, as it is possible for the contents to become zero without feeding information into the accumulator (e.g., a right shift) so that the flow diagram becomes similar to that for [14] and [15] .

6.13 Starting Procedure

When the power supply is first switched on, the computer is static apart from the main oscillator producing the

/timing

timing pulses. Before operating the Run control, it is necessary to set correctly all the bistable units which will be arbitrarily set when the main power is connected. To achieve this, an Initial Set control is first operated which sets all the bistable units to their correct states.

To run the computer, the Run Control is then operated which permits the passage of the main oscillator pulses to the computer circuits.

It is necessary to start the computer in an IW period, as nothing can be done until an instruction is withdrawn from the store. It will be observed from Fig. 11 that the change-over from OW to IW takes place during the 7-pulse group beginning with OX, corresponding to the initial setting of the timing counter. As the control register is initially cleared, the computer would carry out an absolute stop instruction immediately stopping the computer on OZ (see below in Section 6.14). As this cannot be permitted, a Start bistable unit is used to prohibit this action taking place, as shown in Fig. 29. This bistable unit is also used, when restarting after a normal stop, to advance the C-counter by unity at OY instead of using OB, which is inhibited by the stop call. When the computer is first started, the OY pulse is inhibited by using the Jump bistable unit as shown. The

/reading

reading from the store is also automatically arranged to begin at instruction zero of the wired Initial Order subroutine described in Section 5.5.

6.14 Stopping Precautions

In the case of stopping the computer, it is important that it should stop at the end of the OW period for which it has been called. If this was not arranged, then restarting would not occur at the correct instant in the timing cycle. In order to accomplish this, stopping always occurs at the beginning of OZ, that is, at the last instant of OW permitting the operation to have been completed in the case of Single Shot operation and before the change-over to IW, when the function would have been removed at IA in the case of a programme stop. In addition, when a stop is called, it is desirable that the computer should indicate the true conditions at that time, in particular the state of the control unit. Normally, the C-counter would be advanced with OB, but this is inhibited when a stop is called, the advancement taking place at OY when the next start takes place. Similarly, the contents of the C-register are not substituted by the contents of the store register for S-modification [17] until restarting, if a stop is called.

6.14.1 Causes of Stopping

There are a number of reasons why the computer

/should

should stop apart from fault conditions, and these may be divided into two groups (a) intended stops and (b) unintended stops.

The intended stops (a) consist of Single-shot operation and the programme stop orders [0]. In the case of single-shot operation, the computer is operated at the rate of one instruction for each operation of the start button and, apart from the change of timing of the control counter advance pulse, the computer functions similarly to the normal operation, only at very low speed. With the programme stops [0], the normal and optional stops act in like manner to the single-shot operation, the optional stop being brought in or out or use by means of a panel switch. In the case of an absolute stop, not normally programmed and perhaps therefore belonging to (b), the only means of restarting the computer is to return to the Initial Set conditions, due to the inclusion of the Absolute Stop bistable unit of Fig. 29. This would not normally be programmed and is mainly intended as a precaution against instruction source errors, for example, if in error, the computer should jump to an empty portion of the store, the computer will stop indicating on the control counter from whence the absolute stop was extracted, almost certainly incorrectly, thus permitting the programme to be checked before restarting.

/In

In addition to the absolute stop mentioned above, the computer may stop due to an order being impermissible on account of an overshift or overflow having taken place unexpectedly. Thus every instruction is first tested at OA for permissibility, as shown in Fig. 30. If the instruction is impermissible, the computer will stop and can be restarted by means of the Initial Set and Run controls or by use of a special Impermissible Override start control, an indicator lamp reminding the operator that the results obtained may be false.

Certain orders, however, are permissible, as shown in Table IV, as these orders either modify the following instruction or may clear the impermissible condition.

TABLE IV

CONDITION	PERMISSIBLE ORDERS
(a) Overshift after Normalising	16, 17, 24
(b) Overflow in the D- or M-registers	16, 17, 25, 35 - 39, 59
(c) Overflow in the S-register or in the M-register with an arithmetical left shift after (b)	No permissible order, as information is permanently lost and cannot be retrieved.

Finally, there is the possibility of an error, probably due to punching, in which a vacant function number VAC is called for, such as [11], [18], etc. As it does not exist, the instruction is treated as an absolute stop and requires the operator to return to the programme and thence to the Initial Setting.

7. Transistor Computer. Engineering Design

From the above logical considerations, it will be observed that, apart from the main store and logical units, the majority of the computer consists of gates, bistable, counter and register units, and it is necessary to consider some of the engineering problems concerned with these items.

7.1 Gates

In Section 3.4 the different combinations of diodes and triodes (both vacuum and semiconductors) were considered and, at the time, it was decided to use semiconductor diodes together with triode valves where power was required. In this final design, the only difference is to use transistors in place of the triode valves, due to the reduced cost.

A typical diode AND gate is shown in Fig. 31. If any of the inputs to the circuit shown at (a) is at zero potential, then the particular diode will conduct and the output voltage will be approximately zero; on the other hand, if all the inputs are at a negative potential greater than or equal to the bias voltage E_b, then the output voltage will be equal to E_b (negative); hence the circuit gives a negative output only when all the inputs are negative, that is, the logical function AND is performed. If an external load is connected to the output, then this is equivalent to

/a

a resistor R_1 connected to zero potential, as shown at (b). This does not affect the zero voltage condition but it does affect the AND condition of a negative voltage which, if we assume non-conduction of the diodes, reduces the output voltage to $V_o = \frac{R_1.E_b}{R_b+R_1}$. Thus, to maintain the output voltage high, it is desirable to keep R_b small compared with R_1. Unfortunately, this means increasing the current I_b when the gate is closed, as it will then be $I_{bo} = \frac{E_b}{R_b}$; unless E_b is reduced. As will be seen later, it is usually necessary to keep V_o high and hence E_b large. Thus we have conflicting conditions and, as it is desirable to satisfy both, it is generally necessary to use some form of power amplifier. The two forms commonly used:

(i) the common emitter amplifier of Fig. 31(c) and

(ii) the common collector amplifier of Fig. 31(d).

These circuits are quite standard and are capable of high power gain and, in the present instance, the transistor is being used essentially as a switch; hence the need for the coupling components R_b and C_b.

Consider now the application of a negative step input to each of the amplifiers as the result of the output of an AND gate such as at (a), producing the given waveforms corresponding to the circuits (c) and (d). The following points should be noted:

(A) Common Emitter	(B) Common Collector
(i) Output waveform is inversion of input so that logically $V_o = \overline{V_i}$.	Output waveform is not inverted, so that logically $V_o = V_i$.
(ii) During the OFF time V_o is dependent on the load resistor R_l, i.e., $V_{ceo} \doteq \frac{R_l \cdot E_c}{R_l + R_c}$.	During the OFF time $V_o = 0$, independent of circuit parameters.
(iii) During the ON time, V_o is almost constant and equal to $-V_{ces}$.	During the ON time, V_o is approximately proportional to V_i and almost independent of R_l due to the low output impedance of the amplifier (100% negative feedback).

Normally (A)(ii) is not serious as R_c can be small and (B)(ii) is desirable, whilst (A)(iii) is preferable to (B)(iii) due to probable variations in V_i, so that (A) appears preferable to (B).

The most serious difference, however, occurs in (i) in which it is seen that (A), unlike (B), gives a normally undesirable inversion. This can be overcome by cascading two stages of (A) but this is relatively expensive, and it was thus decided to use (B) for power amplification. However, (A) also has its use where the logical function NOT is required and, as this often follows an

/AND

AND, this causes a saving of the power stage type (B). The resistor R_b is used for current limiting due to the relatively large driving signal and C_b is used to give a good switching response, as discussed in Section 7.3.2.

The other form of gate used is the one-gate or OR gate, a typical arrangement being as shown in Fig. 32, in which it will be observed that if any input signal becomes negative, the output also becomes negative due to the conduction of the corresponding diode, the remaining diodes being reversed biassed and preventing the input signal feeding back into the other input circuits. It will be seen, therefore, that the effective output resistance of the OR gate will be approximately equal to the output resistance of the circuit feeding the gate, as the resistance of the conducting diode is normally low and hence, with reasonable care in design, it is not normally necessary to connect an output amplifier to the OR gate unless the load is excessive. Also, the value of R is not critical, providing it is not made so large as to cause an excessive time constant with the total stray capacitance between the output terminal of the gate and zero potential.

7.2 Logical Units

Apart from the individual gates, there are a number of composite logical units in the computer. These consist of decoders, adders, and a logical unit.

(i) Decoders

(1) Decoders

Normally decoders consist of a matrix of AND-gates but, as was seen in Section 7.1, OR-gates are easier to use, due to their relatively low output impedance. It is, therefore, convenient to use "inverted logic" for the decoders, that is, to use OR-gates throughout instead of AND-gates and invert the signals at the input and the output. The former is particularly easy, as both the signal and its inversion are available at the source bistable unit. This method is clarified by Fig. 33, which shows a simple case of two inputs giving four outputs and is also verified by the use of one of De Morgan's Laws, namely,

$$\overline{A + B} = \bar{A}.\bar{B}$$

For, considering the condition 00 in Fig. 33(a),

$$00 = \overline{I_1}.\overline{I_2}$$

and from (b)

$$00 = \overline{I_1 + I_2}$$
$$= \overline{I_1} \cdot \overline{I_2}$$

From Fig. 33, it appears that no advantage is obtained by using the inverted logic. The main advantage is that the output waveform from the NOT-element is standardised unlike that from the common emitter power amplifier P.

(2) Adder and Subtractor Units

The method used in all these units is the standard practice of combining half-adder or -subtractor units as in Fig.

/34.

34. The only difference between the two units is the production of the carry or borrow digit, the one requiring an additional NOT-unit. The full adder/subtractor unit in the main logic unit requires a half-adder and a half-adder/subtractor connected, as in Fig. 35(a). Due to the findings of Section 7.1, the arrangement of Fig. 35(d) was devised using NOT AND rather than AND logic. It will be noted that this results in all inputs and outputs being NOT-digits, which are just as easily obtainable from the register bistable units as the original digits. The auxiliary adder is simplified by the omission of the subtract facility and the D- and M-half-adders are as shown in Fig. 35(b).

(3) Logic Unit

In addition to providing the above facilities of the Main Adder and Subtractor, the logic unit performs the logical functions of AND and NOT EQUIVALENT as shown in Fig. 36, NOT AND logic being used again.

7.3 Bistable Unit

The bistable unit is the fundamental storage element used in the computer apart from the main store. The units are used individually for controlling various operations or parts thereof when there are a number of phases and also

/as

as the basis of the counters and registers; it is thus a very important unit.

In its simplest form, the transistor bistable unit is shown in Fig. 37 and it will be seen to consist of two cross-connected common-emitter amplifier stages, similar to that of Fig. 31(c). The mode of operation has been described in most books on electronic switching circuits, including those given in the Bibliography. The main requirements of the bistable unit are:-

(A) Stability in the steady state or d.c. condition, which implies

(i) that the conducting transistor should be bottomed, as the collector current will then be less dependent on the transistor characteristics and temperature, and

(ii) the non-conducting transistor should have some reverse bias on the base rather than zero voltage in order to reduce the cut-off collector current, which is also temperature dependent.

(iii) Some load must be able to be connected to the collector without bringing the conducting transistor out of saturation.

(B) It is desirable that the output voltage of one bistable unit may be used to drive a second bistable unit, as this will be commonly required in counters and registers.

(C) High-speed switching is obviously required on condition that a safe margin of stability is maintained.

7.3.1 Steady State Stability

The bistable unit must have two, and only two stable states, which are connected by an unstable state; the unstable state occurs when both transistors are conducting as distinct from the stable states when only one is conducting. In all, there are five conditions which will normally exist if bottomming takes place. These are shown in the d.c. input characteristic of a typical transistor bistable unit, Fig. 38, in which V_i is the voltage applied to one base and I_i the current supplied. This characteristic has been fully developed by Neeteson[N] and makes the following reasonable assumptions:

(1) that the base-emitter characteristic is a straight line, and

(2) that the collector-emitter circuit behaves as an ideal switch, that is, no resistance when conducting and no current when non-conducting.

To determine the operating conditions when used as a bistable unit, a load line corresponding to the base resistor R_b and supply voltage E_b is added, as shown by a typical value on Fig. 38, in which the two stable states are given as X and Y. For the bistable unit to be stable, the load line must obviously cut the negative resistance portion C and, as the

/change

change in voltage between these two points is small, it is convenient to define the degree of stability, S, in terms of the current at O compared to the values at 2 and 3. It will be seen that the maximum stability will occur when the value I_o is the mean of I_2 and I_3 and the slope of the load line is infinite which, in practice, means making R_b as large as possible, at the same time increasing E_b as $I_o \doteq \frac{E_b}{R_b}$ as the line CD is almost coincident with the current axis.

The Stability, S, is thus defined as

$$S = \text{smaller of } \frac{I_3 + I_o}{I_m} \text{ or } \frac{I_2 - I_o}{I_m}$$

where I_m is the mean value of I_2 and I_3,

which means that the maximum stability is unity.

7.3.2 Transient Analysis

The transient condition theory is far more difficult to deal with but has partially been considered by a number of writers, in particular Early[59,60], Ebers[61] and Moll[61,62], who approached the problem by considering the transistor as a current-controlled device but this, unfortunately, gives rise to some difficult measurements of current gain and cut-off frequencies. In 1957, Beaufoy and Sparkes[63] approached the problem by considering the transistor as a charge-controlled device, and this leads to a

/more

more satisfactory solution to the problem. The basis of their method is to consider the charges Q_B and Q_{BS} stored in the base region and saturation region respectively. Equivalent time constants may be derived in terms of these charges and hence the switching times for the transistor.

In the case of the common emitter configuration as used in the bistable unit, the relevant time constants[63] are:

The base time constant, T_b, $$= \frac{\Delta Q_B}{\Delta I_B} = \frac{1.22}{\omega(1-\alpha)} \qquad (1)$$

The collector time constant, $$T_c = \frac{\Delta Q_B}{\Delta I_C} = \frac{1.22}{\omega . \alpha} \qquad (2)$$

with constant collector voltage.

The variable voltage base time constant $$T_v = \frac{\Delta Q_B}{\Delta I_B} + \frac{\Delta Q_{Bn}}{\Delta I_C} \qquad (3)$$

where ΔQ_{Bn} is the change of base charge due to the change of the depletion layer caused by the change of collector voltage.

The saturation time constant $$T_s = \frac{\Delta Q_{BS}}{\Delta I_{BS}} \qquad (4)$$

Having arrived at these time constants, the switch-on and off times can be readily calculated, assuming the base input current to be constant, the resulting expressions being given below, using the symbols shown in Fig. 39:

$$T_r = T_v \log_e \frac{I_{b1} T_b}{I_{b1} T_b - 0.9 I_c T_c} \quad (5)$$

$$T_f = T_v \log_e \frac{I_c T_c + I_{b2} T_b}{0.1 I_c T_c + I_{b2} T_b} \quad (6)$$

$$T_o = T_s \log_e \frac{I_{b1} T_b + I_{b2} T_b}{I_c T_c + I_{b2} T_b} \quad (7)$$

In the case of the bistable unit, however, the problem is complicated by the base current not being constant, due to the transient effect feeding back on itself due to the coupling components.

Neeteson has tackled this problem at length and shown that it is in general quicker to change over the bistable unit by means of a positive drive to the conducting transistor rather than a negative drive to the non-conducting transistor[N120]. He does have some difficulty with his value of fall time, which he later explains as being due to the influence of the depletion-layer capacitances. This appeared to agree with Beaufoy and Sparkes[63], where allowance is made for the variation of charge due to the variation of the depletion layer by using T_v rather than T_b. It was decided to carry out a similar test to Neeteson in order to confirm this. Before giving the experimental results, it is necessary to add some further theory in order to obtain the total change-over time and hence the maximum speed of operation.

/Neeteson

Neeteson regards the times from the start of the trigger pulse applied to one transistor until the time is reached when the other transistor is affected[N108]. Useful though this may be, it does not give the full information as shown by Fig. 40, in which he only considers times T_o and T_i (corresponding to his t_d and t_2). Having calculated the value of T_i from Neeteson's equation (5.145) which is derived from the equivalent circuit reproduced in Fig.41(a), it is possible to proceed in the following manner with reference to Fig. 41.

$$i_{c1} = -\left\{I_{c1} + \beta I_{b2}\right\}(1 - e^{-t/T_v}) \qquad (8)$$

where i_{c1} is the transient term of the collector current

I_{c1} is the initial value of the collector current

I_{b2} is the final value of the base current

β is the common emitter short circuit current gain and equals T_b/T_c from equations (1) and (2)

$$i_p = -\left\{I_{c1} + \beta I_{b2}\right\}\frac{R_c}{R_b+R_c+R}\left[1 - \frac{T-T_1}{T_v-T_1}e^{-t/T_1} - \frac{T_v-T}{T_v-T_1}e^{-t/T_v}\right] \qquad (9)$$

where T_v has been substituted for T_b to allow for the variation of collector voltage,

$$\left.\begin{aligned} T_1 &= \frac{R_c + R_b}{R_c + R_b + R}\cdot T \\ T &= CR \end{aligned}\right\} \qquad (10)$$

and $$i_r = i_{c1} - i_p \qquad (11)$$

After time T_i, it is assumed (i) that R_b is short circuited by transistor TR2 conducting and (ii) that its input characteristic is that of a forward biassed ideal diode. The switching characteristic now follows the curve shown in Fig. 40, which is part of an exponential curve starting from an initial value of i_{c1} equal to I'_{c1}. Under these conditions, a similar set of equations to those above are obtained:

$$i'_r = i'_{c1} - i'_p \qquad (12)$$

where

$$i'_{c1} = -\left\{I'_{c1} + \beta I_{b2}\right\}(1 - e^{-t/T_v}) \qquad (13)$$

and

$$i'_p = -\left\{I'_{c1} + \beta I_{b2}\right\}\frac{R_c}{R_c+R}\left[1 - \frac{T-T'_1}{T_v-T'_1}\, e^{-t/T'_1} - \frac{T_v-T}{T_v-T'_1}\, e^{-t/T_v}\right] \qquad (14)$$

where $T'_1 = \dfrac{R_c}{R_c + R}\, T$

Hence time T_F can be calculated from the above equations, but the required fall time is

$$T_f = T_F - T_I \qquad (15)$$

From equations (11) and (12) at time T_i

$$i_R = i_r = i'_r$$

/or

$$- \left\{ I_{c1} + \beta I_{b2} \right\} \left\{ (1-e^{-T_i/T_v}) + \frac{R_c}{R_b+R_c+R} \left[1 - \frac{T - T_1}{T_v - T_1} e^{-T_i/T_1} - \frac{T_v - T}{T_v - T_1} e^{-T_i/T_v} \right] \right\}$$

$$= - \left\{ I'_{c1} + \beta I_{b2} \right\} \left\{ (1 - e^{-T_r/T_v}) + \frac{R_c}{R_c+R} \left[1 - \frac{T - T'_1}{T_v - T'_1} e^{-T_r/T'_1} - \frac{T_v - T}{T_v - T'_1} e^{-T_r/T_v} \right] \right\} \qquad (16)$$

and hence T_f can be calculated.

From Fig. 40, it is seen that T_r is shown as being less than T_f. This, in fact, is so under normal conditions as was observed experimentally and can be checked analytically by assuming the TR2 base voltage to change instantaneously; this being almost true, due to the small change of voltage required.

Thus the total switching time

$$T_t = T_o + T_i + T_f \qquad (17)$$

gives a measure of the speed of operation.

7.3.3 Transient Analysis Experiments

To confirm the above theory, a pair of average transistors of the type used in the computer were connected as a bistable unit, using reasonable values of components. The bistable unit was switched on and off by means of a pulse generator through a pair of semiconductor diodes in the form of a counter stage (see Section 7.4) and the collector voltage waveforms were observed on a 10-Mc/s

/bandwidth

bandwidth oscilloscope.

The results obtained are given in Table V below, where the fixed parameters of the circuit (Fig. 33) were:

R_c = 1kΩ R = 10kΩ

R_b = 5kΩ C = 1000pF

E_b = +3V

Input pulse 10V peak and having a source resistance of 500Ω. Transistor parameters as measured by the method described by Beaufoy and Sparkes[63]:

TR1: T_b = 12μs T_c = 0.14μs

T_v = 20μs T_s = 4.9μs

TR2: T_b = 11.5μs T_c = 0.12μs

T_v = 19μs T_s = 5.1μs

The easiest parameter to vary was E_c, as this was provided by a variable voltage stabilised power supply, and the measurements are only given for the one transistor as the values for TR2 were similar to those of TR1 given in the Table.

/TABLE V

TABLE V

E_c (-volts)	T_o Meas. (μs)	T_o Calc. (μs)	T_i Meas. (μs)	T_i Calc. (μs)	T_f Meas. (μs)	T_f Calc. (μs)
8	.021	.018	.30	.29	1.72	1.65
9	.060	.058	.30	.29	1.85	1.79
10	.129	.127	.31	.29	2.10	1.95
11	.142	.137	.30	.29	2.19	2.07
12	.179	.177	.32	.29	2.22	2.20
13	.217	.217	.31	.29	2.31	2.29

The values for the second transistor were similar and the measured values show reasonable agreement with the calculated values. The experimental values obtained are slightly higer than those calculated which is probably due to the limited bandwidth of the oscilloscope.

7.3.4 Effects of the Circuit Parameters

Having established that the theory agreed with practice, it was decided to attempt to predict the effect of variation of the parameters. The main difficulty lies in the large number of parameters, at least 10, and the problem was resolved into a form of normalising, that is, each parameter was allowed to vary separately whilst maintaining the other parameters constant at the original value. The calculations

/involved

involved were extremely tedious without the use of an automatic computer, so that results were calculated for a ± 20% range of values of each parameter. The results are shown in Figs. 42 to 47, one diagram for each of the calculated terms, rather than the parameters, as the range of values of the calculated values was large and this method avoids the use of a number of scales on any one diagram.

Conclusions

(1) Stability, Fig. 42

The stability, which is the most important factor in the computer, is almost unaffected by the value of R_c or β, but quite drastically by the other parameters. It is not unexpected to find that R_b and R tend to compensate for each other in that they form a potential divider across a transistor, so that with ageing they will tend to change equally and give consistent stability. Similarly, with the power supplies E_c and E_b if stabilisation is poor, it would appear best for them to have the same percentage variation. In an original design, it appears that R and E_b should be kept small whilst R_b and E_c should be kept large. It will be seen later, however, that this conflicts with the speed of operation.

(2) Switching Time, Figs. 43 to 46

Over the range considered in Figs. 43 to 45 the predominant term is always the fall time Tf ranging from 1.6 to 2.3μs compared to the other values, To = up to 0.2μs and Ti

/between

between 0.2 and 0.4μs, though in some cases, these short times may be important in the present instance only the total time, T_t, Fig. 46, will be considered, for which the variations are similar to T_f, Fig. 45.

The voltage E_c, input current I_i and the transistor parameters have the greatest effect but in cascaded conditions, such as counters or registers, E_c and I_i will tend to cancel each other out, as I_i will then approximate to E_c/R_c when driven from the collector of a previous stage. As might be expected, a large value of T_v requires a longer time, whereas a large value of β, giving an effectively larger drive, reduces the time. The effects of C and R cancel each other, as the ideal condition is to have the CR product equal to T_B in a similar manner to a frequency compensated potential divider.

The fact that the base components E_b and R_b have little effect on the speed is mainly due to the small change of base potential, their main effects being on the smaller terms T_o and T_i.

The fall time, T_f, is dependent primarily on removing the base charge, so that the smaller this charge the shorter the fall time. As this charge $Q_B = I_c T_c = \frac{E_c}{R_c} . T_c$, it is to be expected that a large value of E_c or small value of R_c will tend to give a longer fall time and this is shown to be true.

/However,

However, R_c is normally limited to a reasonable value to assist stability and to give a low output impedance.

In an original design where speed of operation is important, it would appear advantageous to keep E_c low, R_c and I_i high and CR equal to T_B. These will be seen to conflict with the stability findings and it is interesting to plot a final set of values, the bistable unit figure of merit.

(3) Figure of Merit, Fig. 47

The figure of merit is the product of the stability and reciprocal of the total switching time. The ideal will obviously be to keep the value as high as possible, providing that the stability is not impaired as this is the more important.

To summarise then, it would seem that

(a) E_b should be kept as small as possible, if the delay time is not important.

(b) R should be kept small though this does increase the fall time.

(c) E_c should be kept high, though care must be taken not to create an excessive switching time. This will commonly be compensated for by I_i being proportional to E_c.

(d) R_b should be kept high, bearing in mind that it is the ratio E_b/R_b which is important.

/(e)

(e) R_C is not too critical, though too small a value may cause a large switching time against which a large value will give poor stability.

(f) The transistor should have a large value of β but small base time constants T_v and T_s.

7.3.5 Transistor Parameters

From Section 7.3.2 it is seen that the relevant transistor parameters are the short circuit current gain β, the variable voltage time constant T_v and the saturation time constant T_s.

Tests were carried out on a random batch of 500 transistors, Type GET113, in an attempt to predict the variation of the parameters, the measurements being carried out by the method given in Reference 63. The results are given in Figs. 48 to 51, which include all values with the exception of about 20 which were completely out of range or had an excessive value of collector cut-off current, I_{co}.

(1) Variation of β, Fig. 48

From the histogram of Fig. 48 it is observed that the majority of the transistors tested had a value of β lying between 150 and 170 with a more rapid cut-off on the lower value of β. This is to be expected, as the manufacturer is more likely to reject values below a certain minimum rather

/than

than above a certain maximum as, in general, a high value of β is to the designer's advantage. Taking the peak of the histogram as corresponding to a β of 160, the majority of the transistors can be said to fall within the range of -25% to +50%.

(2) Variation of T_v, Fig. 49

This diagram is similar to that obtained for β except that it is more symmetrical about the mean value of 20μs, so that the majority of the transistors tested are within a range of ± 40%.

Considering equations (1) and (2)

$$T_v = T_b + \frac{\Delta Q_{Bn}}{\Delta I_c} \qquad (18)$$

and assuming a reasonable control on the impurity content of the germanium used in the transistor manufacture, the second term of equation (18) should be sensibly constant.

A histogram of T_b was plotted, Fig. 50, and shows a similar trend to T_v. Further as

$$T_B = \frac{W^2}{2D_p (1 - \alpha)} \qquad (19)$$

it would be expected that T_B would be principally dependent on the depletion layer width W as the variation of α will be small.

/This

This implies that no selection for this quantity has been made by the manufacturer, with the result that there is an equal probability of the base width being above or below the nominal value.

(3) Variation of T_s, Fig. 51

The histogram for T_s is similar to that for T_b, as T_s is also primarily dependent on W^2. The range of value of T_s, 3μs ± 50% to +75% , is high but this is not serious when it is remembered that T_s only affects the storage time, T_o, which is about 2% of the total switching time.

7.3.6 Conclusions

From the above analysis, it would appear possible to devise a computer programme to optimise the design of the bistable unit, given the possible range and tolerance of all the parameters. This would require considerable effort, and possibly some ingenuity, to achieve and it is hoped that this project may be pursued at a later date.

The final circuit given in Fig. 52 is, therefore, partially intuitively designed, but is based on the arguments of the previous sections.

Various schemes are available for reducing delay times due to storage by preventing the transistors from bottoming, use commonly being made of diode clamps. However,

/in

in the present instance, this was not done as the delay time was comparatively small. The fall time, T_f, was effectively reduced by means of the diodes, D, which hold the collector voltage at E_d as it attempts to reach the normal value of E_c. A full analysis of this circuit has not yet been considered, but it is unlikely to prove worthwhile unless one is interested in very high speed switching, when storage charges in the diode would have to be considered as well as the effects of preventing the transistors from bottoming. This might well be followed up at a later date, but is unnecessary in the present instance.

7.4 Counters

The counter circuit consists of a bistable unit with a suitable pulse-steering network to cause the input pulse to change over the state of the bistable unit whatever its previous condition. The simplest arrangement, which was that used, is to incorporate a pair of diodes in the conventional manner shown in Fig. 53. In this arrangement, the incoming pulse is always steered to the conducting transistor, causing it to become cut-off and the changeover made.

7.5 Registers

In general, the registers consist of a series of bistable units suitably coupled so that, when required, the information can be shifted from one bistable unit to the next at the command of a shift pulse. If information in stage

R_{n-1} is to be transferred to stage R_n, which itself will be transferring to R_{n+1}, then it is necessary to provide some form of temporary storage between the stages in which the information can be held during the transfer process. There appear to be three basic methods of providing this temporary storage, as shown schematically in Fig. 54.

(a) Capacitance Storage

The method is shown in Fig. 54(a) in which the information is stored on the capacitors, C, in addition to the bistable units, the succeeding stage being isolated by means of the diodes D. A suitable shift pulse is applied at S, which tends to cut-off all the stages of the bistable units but, due to the voltage stored on the capacitors, only the required halves of the bistable units are cut-off, resulting in the information being correctly transferred. The CR product will have to be large compared to the shift pulse width but small compared to the periodic time of the pulses, so that normally a compromise has to be made.

(b) Delay Line Storage

In this case, all the bistable units are cleared to the zero state; those stages that were storing unity produce a step function at the input to their delay lines, DL, shown in Fig. 54(b). After a short delay, this step is applied through the diode, D, to change it to the one state as required. This

/system

system appears to have no appreciable advantage over (a) and has the disadvantage of requiring relatively expensive delay lines.

(c) Bistable Unit Storage

In this system, shown in Fig. 54(c), there are two bistable units per digit, one being the main register bistable unit RBU, the other the temporary bistable unit TBU.

Assuming that each TBU is initially cleared, then application of the first shift pulse, S_{RT}, immediately transfers the contents of each RBU to its associated TBU, automatically clearing itself.. The second shift pulse, S_{TR}, now transfers the contents of each TBU into the next RBU, again automatically clearing itself; thus the information has been advanced one place along the register. The main advantages of this system are (i) the lack of time elements compared to CR of (a) or DL of (b), which means that high speeds of operation can be achieved and (ii) flexibility of operation, for example, the comparative ease of making the register reversible; on the other hand, the system is expensive compared to (a), requiring two bistable units for each digit and the provision of two sets of shift pulses.

System (a) was finally adopted as the basis for all the registers, with the following modifications:

/(1)

(1) Reversible registers M, L and D.

(2) Complementing register C

Fig. 55 shows the method adopted for the reversible register, in which the shift pulses are gated to the appropriate input, according to the required shift direction. The method of operation is identical to that described at (a), isolation from the non-required direction being made by the two diodes, D or D'.

In the case of the complementing register, Fig. 56, each stage of the register has added to it a counter control which changes over its contents irrespective of the original condition, in the same manner as described in Section 7.4, isolation between the two operations being provided by the diodes D and D'.

7.6 Main Store

The main store, which is of the magnetic core type, uses the direct selection method rather than the matrix-core arrangement, as this assists the discrimination between "1's" and "0's"[57,64]. The store, consisting of 1024 locations, is divided into 32 planes of 32 lines each, and it is necessary to be able to select any one line with the minimum of components. In the past, when using valves, the tendency had been to use a magnetic core selector in order to obtain a transformation

/ratio

ratio in addition to the switching action. More recently, transistor selectors have been considered[65] on account of their high current capabilities compared to valves.

Basically, only 96 transistors are required,- 32 to select the lines in a plane, 32 to provide the Read-pulse to the planes and a further 32 to provide the Write-pulse to the planes. In addition, it is necessary to provide two diodes for each word to prevent feedback along the lines causing 32 words to be selected instead of only one. The cost of 2048 diodes is quite considerable and other methods of isolation were considered without much success, with the final result that a transistor is used as a double diode, as shown at DD in Fig. 57. The method appears to be quite satisfactory from the preliminary tests, provided that the line selector is actuated prior to the Read or Write pulse being applied.

Fig. 57 also indicates the method of threading the cores on the three wires; the 20 cores are first threaded onto the two horizontal selector wires in the form of a chain and the assembly then threaded over the twenty vertical read/write wires.

7.7 Initial Order Store

The 96 initial orders are prewired on three planes of 32 words similarly to the Main Store, the only difference being that, due to its permanent nature, cores are

/only

only provided where a "1" is required. Where a "0" occurs, a small plastic ring of comparable size is provided to keep the wires in position and the write wire for the entire 96 words is pulsed after each reading, as it is not necessary to select the words individually during writing, all the cores being set to the "1" state at each time.

8. Conclusions

Having considered, in Section 2, the development of automatic digital computers up to 1956 when the present project was begun and, in Section 3, the basic units from which computers were constructed, it was concluded that, at that time, the trend was towards semiconductor and magnetic devices, principally magnetic cores and drums. For this particular project it was desirable to be able to observe easily the digital contents for demonstration purposes so that magnetic cores were ruled out, apart from the main store. A magnetic drum store and, at that time, transistors had to be rejected, due to their cost, and a relatively simple valve computer was designed, based on the arguments of Section 4.

Fortunately, there was then a break of about two years, due to other commitments, during which time the price of transistors fell considerably, making them a far more attractive proposition than valves. Also, at this time, some practical programming advice was given by the University Computing Laboratory and assistance was offered by a local manufacturing firm.

This brought about a change of the idea of a purely teaching computer to the concept of a small low-priced general purpose computer suitable for engineering computations, as was discussed in Section 5. This required more care to be devoted

/to

to the design philosophy, as seen from the work of Section 6. Every attempt was made to keep the cost of this more ambitious computer within the original component estimate of £2000; although the total storage and the size of the word length were increased, it was desirable not to decrease the speed of operation.

To achieve this low cost, it was necessary to investigate the switching performance of relatively cheap transistors, both analytical and experimental methods being used. It was shown in Section 7 that it was not easy to determine the absolute maximum speed of operation which conflicts with the Stability Factor; the ultimate design will thus depend on how low a Stability Factor the designer is prepared to accept. This problem appears to be worth pursuing by attempting to programme the design for use on a computer.

Experimentally, it was found that a speed of 200 kc/s was quite easily obtainable, using a pair of typical transistors tested in the laboratory and therefore, allowing for manufacturing spreads due to components and loadings, a clock frequency within the range of 50 - 100 kc/s should be easily obtained.

Unfortunately, the computer is not yet completely built, but tests have shown that the main pulse timing unit

/operates

operates satisfactorily up to a speed of 120 kc/s, so that the final operation speed may well be nearer 100 kc/s than 50 kc/s.

From the results of considering the effect of variation of the possible parameters on the Stability of the bistable units, it does not appear necessary to provide any form of marginal checking facilities, which would only increase the cost and apparently serve little purpose; time alone will prove whether this was a wise decision or not.

It can be concluded, therefore, that the development of a low price useful computer is possible, provided that considerable care is taken in the logical design to keep the final computer as flexible as possible, reasonably foolproof and yet simple to use. Although the computer is still under construction, the Author is confident that it will fulfil the above requirements and fill a gap in the range of commercially available machines, lying between the small single purpose computers of limited use and the large general purpose computers, which are not economically priced for most engineering purposes.

For the future, the following ideas would appear worthwhile projects:

1. The programming of an optimum design for a transistor bistable unit on the lines discussed in Section 7.

/2.

2. The development of an automatic programme to aid the future operators of the computer.

3. The addition of a magnetic drum or tape backing store to the computer for which space has been left in the order code, and

4. The development of a faster machine, using the same design philosophy but using higher frequency transistors.

9. Bibliography

Books

B	Bowden, B.V.	Faster Than Thought (1955)
BB	Booth, A.D. and Booth, K.H.V.	Automatic Digital Calculators (1956)
E	Engineering Research Associates.	High Speed Computing Devices (1950)
H	Hartree, D.R.	Calculating Instruments and Machines (1949)
I	Ivall, T.E.	Electronic Computers (1956)
N	Neeteson, P.A.	Junction Transistors in Pulse Circuits (1959)
W	Wilkes, M.V.	Automatic Digital Computers (1956)

Articles

1.	Menabrea, L.F.	Bibliotheque Universelle de Geneve No. 82 (1842)
	and translated by Augusta A, Countess of Lovelace.	Taylor's Scientific Memoirs 3.Art.XXIX. (1842)

and more recently reproduced in Reference B above.

2.		Encyclopaedia Britannica 23 460 (1955).
3.	Babbage, H.P.	Babbage's Calculating Engines (1888).
4.	Aitken, H.H. and Hopper, G.M.	The Automatic Sequence Controlled Calculator. Electrical Engineering 65, 384 (1946).
5.	Goldstine, H.H. and Goldstine, A.	The ENIAC M.T.A.C. 2. 97. (1946).
6.	Hartree, D.R.	The ENIAC, an Electronic Computing Machine. Nature, 158, 500 (1946).
7.	von Neumann, J.	First Draft Report on the EDVAC. Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, Report (1945).
8.	Wilkes, M.V., Wheeler, D.J., Gill, S.	The preparation of programs for an Electronic Digital Computer (1951).
9.	Williams, F.C., Kilburn, T., Tootill, G.C.	Universal High-Speed Digital Computers: A Small-Scale Experimental Machine. Proc.I.E.E. 98.II.13 (1951).
10.	Williams, F.C. and J.C. West	The Position Synchronisation of a Rotating Drum. Proc.I.E.E. 98.II.29 (1951).
11.	Williams, F.C., Kilburn, T., Thomas, G.E.	Universal High Speed Digital Computers: a Magnetic Store. Proc.I.E.E.99.II.94 (1952).

12.	Kilburn, T., Tootill, G.C., Edwards, D.B.G. and Pollard, B.W.	Digital Computers at Manchester University. Proc.I.E.E. 100.II.487 (1953).
13.	Barnes, R.C.M., Cooke-Yarborough, E.H., Thomas, D.G.A.	An Electronic Digital Computer. Electronic Engineering 23.286 (1951)
14.	Kilburn, T., Grimsdale, R.L. and Webb, D.C.	A Transistor Digital Computer with a Magnetic Drum Store. Proc.I.E.E.103.B.390 (1956).
15.	Foulkes, R.M.	The "Metrovick 950" Digital Computer. M.V. Gazette 28.111.(1957).
16.	Cooke-Yarborough, E.H., Barnes, R.C.M., Stephen, J.H., Howells, G.A.	A Transistor Digital Computer Proc.I.E.E.103.B.Supp.3. 364 (1956).
17.	Bruce, G.D., and Logue, J.G.	An Experimental Transistorized Calculator. Electrical Engineering 74, 1044 (1955).
18.	Taylor, N.H.	Introduction to "Rapid-Access Storage" Proc.I.E.E.103.B.Supp.2.289 (1956).
19.	Kilburn, T., Edwards, D.B.G., Thomas, G.E.	The Manchester University Mark II Digital Computing Machine. Proc.I.E.E.103.B.Supp.2.247 (1946).
20.	Lonsdale, K. and Warburton, E.T.	Mercury: A High Speed Digital Computer. Proc.I.E.E.103.B.Supp,2.174. (1956).
21.	Robinson, A.A., Newhouse, V.L., Friedman, M.J., Bindon, D.G., Carter, I.P.V.	A Digital Store using a Magnetic Core Matrix. Proc.I.E.E.103.B.Supp.2.295. (1956).
22.		LEO Electronic Engineering 26.162. (1954).

23.	Knight, L.	An Electronic Calculator for Punched-Card Accountancy. Proc.I.E.E.103.B.Supp.2.228. (1956).
24.	Bird, R.	The HEC Computer Proc.I.E.E.103.B.Supp.2.207. (1956).
25.	Guttridge, E.J., Yandell R.P.B.	The Programme-Controlled Computer. Proc.I.E.E.103.B.Supp.2.217. (1956).
26.	Elliott, W.S., Owen, C.E., Devonald, C.H., Maudsley, B.G.	The Design Philosophy of Pegasus, a Quantity-Production Computer. Proc.I.E.E.103.B.Supp.2.188. (1956).
27.	Fairclough, J.W.	A Sonic Delay-Line Storage Unit for a Digital Computer. Proc.I.E.E.103.B.Supp.3.491. (1956).
28.	Scarrott, G.G., Naylor, R.	Wire-Type Acoustic Delay Lines for Digital Storage. Proc.I.E.E.103.B.Supp.3.497. (1956).
29.	Bambrough, B.	A Digital Computer Based on Magnetic Circuits. Proc.I.E.E.104.B.Supp.7.485. (1957).
30.	Githens, J.A.	The Tradic Leprechaun Computer. Proc.of Eastern Joint Computer Conference, A.I.E.S. Special Publication T-92 (1956).
31.	Takahashi, S., Nishino, H., Matauzaki, I., Kondo K.	ETL Mark III, A Transistor Digital Automatic Computer. ETJ.3.143 (1957).
32.	Wilkes, M.V., Renwick, W.	An Ultrasonic Memory Unit for the EDSAC. Electronic Engineering 20.208 (1948).
33.	Newman, E.A., Claydon, D.O. Wright, M.A.	The Mercury-Delay-Line Storage System of the ACE Pilot Model Electronic Computer. Proc.I.E.E.100.II.445 (1953).

34.	Booth, A.D.	A Magnetic Digital Storage System. Electronic Engineering 21.234 (1949).
35.	Morton, P.L.	A Compact Magnetic Memory Proc.I.R.E.38.211 (1950).
36.	Williams, F.C., Kilburn, T. Thomas, G.E.	Universal High-Speed Digital Computers: a Magnetic Store. Proc.I.E.E.99.II.94 (1952).
37.	Merry, I.W., Maudsley, B.G.	The Magnetic-Drum Store of the Computer Pegasus. Proc.I.E.E.103.B.Supp.2.197. (1956).
38.	Clayden, D.O., Page, L.J., Osborne, C.F.	The Magnetic Storage Drum on the ACE Pilot Model. Proc.I.E.E.103.B.Supp,3.509. (1956).
39.	Papian, W.N.	Coincident Current Magnetic Memory Cell for the storage of Digital Information. Proc.I.R.E.40.475. (1952).
40.	Robinson, A.A., Newhouse, V.L., Friedman, M.J., Bindon, D.G., Carter, I.P.V.	A Digital Store using a Magnetic Core Matrix. Proc.I.E.E.103.B.Supp.2.295. (1956).
41.	Widrow, B.	A Radio-Frequency Non-destructive Readout for Magnetic-Core Memories. Trans.I.R.E. EC-3, No.4.12 (1954).
42.	Buch, D.A., Frank, W.K.	Non-destructive Sensing of Magnetic Cores. Comm. & Electronics 72, 822. (1954).
43.	Papoulis, A.	The Non-destructive Readout of Magnetic Cores. Proc.I.R.E. 42.1283 (1954).
44.	Stephen, J.H. and Cooke-Yarborough, E.H.	An Interleaved-Digit Magnetic-Drum Store for a Transistor Digital Computer. Proc.I.E.E.103.B.Supp.3.382 (1956).

45.	Wang, A. and Woo, W.D.	Static Magnetic Storage and Delay Lines. Jour. of App.Phys.21.49 (1950).
46.	Wang, A.	Magnetic Delay Line Storage. Proc.I.R.E.39.401 (1951).
47.	Eugene, A.S.	An Analysis of Magnetic Shift Register Operation. Proc.I.R.E.41.993 (1953).
48.	Perry, G.H., Hoffman, G.R., Shallow, E.W.	A New and Simple Type of Digital Circuit Technique using Junction Transistors and Magnetic Cores. Proc.I.E.E.103.B.Supp.3.412. (1956).
49.	Hoffman, G.R. and MacLean, M.A.	Quiescent Core-Transistor Counters. Proc.I.E.E.103.B.Supp.3.418 (1956).
50.	Eldridge, D. and Dorey, P.F.	A Core-Transistor Logical Element. Proc.I.E.E.104.B.Supp.7.512 (1957).
51.	Eccles, W.H. and Jordan, F.W.	A Trigger-Relay. Radio Review 1.143 (1919).
52.	Thomas, G.E.	The Design and Construction of an Electronic Digital Computer. Ph.D. Thesis, University of Manchester (1954).
53.	Norquist, R.G.	Testing High-Speed Digital Computing Circuits. Electronics. 32.50. (1959).
54.	Moody, N.R. and Harrison, R.G.	Millimicrosecond Digital Computer Logic. Electronic Engineering 31.526. (1959).
55.	Scarrott, G.G., Harwood, W.J., Johnson, K.C.	The Design and Use of Logical Devices using Saturable Magnetic Cores. Proc.I.E.E.103.B.Supp.2.302. (1956).

56. Thomas, P.A.V. — An Analogue Computer for Teaching Purposes. British Comm. & Electronics 6.443 (1959).

57. Remwick, W. — A Magnetic-Core Matrix Store with Direct Selection using a Magnetic-Core Switch Matrix. Proc.I.E.E.104.B.Supp.7.436 (1957).

58. Brown, D.A.H. — A Wired Subroutine Store using Ferrite Cores. R.R.E. Journal, No. 42, 25. (1958).

59. Early, J.M. — Effect of Space Charge Layer Widening in Junction Transistors. Proc.I.R.E.40.1401 (1952).

60. Early, J.M. — Design Theory of Junction Transistors B.S.T. Journ.32.1271 (1953).

61. Ebers, J.J. and Moll, J.L. — The Large Signal Behaviour of Junction Transistors. Proc.I.R.E.42.1761 (1954).

62. Moll, J.L. — Large Signal Transient Response of Junction Transistors. Proc.I.R.E.42.1773 (1954).

63. Beaufoy, R. and Sparkes, J.J. — The Junction Transistor as a Charge-Controlled Device. A.T.E.Jour.13.No.4. (1957).

64. Ridler, D.S., Grimmond, R. — The Magnetic Cell: A New Circuit Element. Proc.I.E.E.104.B.Supp.7.445 (1957).

65. Padwick, G.C. and Cain, A.L. — Transistor Circuits for a Ferrite Store. Proc.I.E.E.106.B.Supp. 675 (1959).

10. Appendices

Appendix I. Glossary of Terms

This list primarily includes those terms used in the Thesis due to there being such a variety of words in computer terminology, the interpretation of which varies from one Author to another.

Access Time: the time required to extract a number from a storage medium measured from the time at which it is called.

Accumulator: the main register of the arithmetic unit in which double-length numbers may be manipulated.

Adder, half: a logical unit which will add together two binary digits.

Adder, full: a logical unit which will add together three binary digits; often consists of two half-adders and hence the name.

Address: a pattern of digits which locates the position of a number in the main store or identifies a particular register.

Arithmetic Unit: that portion of the computer which carries out the arithmetic and logical operation.

Asynchronous: a computer in which the start of each operation is given by a termination signal from the previous operation; see Synchronous.

Automatic Programme: a computer interpretive programme which is provided as an aid to the programmer, saving him a considerable amount of time.

Bistable Unit: a logical unit having two stable states capable of storing one bit of information; also known as a binary unit or flip-flop.

B-Register: a temporary store used to hold a B-modifier; see Modifier.

Bit: a bit of information; a single binary digit.

Buffer Store: a medium duration store to act as a buffer between the computer and peripheral equipment, in which the information is held for the duration required by the latter equipment.

Carry Digit: the resulting digit after adding two or three binary numbers whose sum produces a number in excess of unity which is carried over to the next stage; in subtraction a borrowed digit is sometimes referred to as a carry digit - carried back instead of forward.

Clock Pulse: one of a standard series of pulses whose repetition frequency is maintained constant.

Code: the list of instructions which the machine may carry out and the symbolic or number representation of these instructions.

Control Counter: a counter which is advanced sequentially in a one-address machine to give the location of the next instruction.

Control Register: a temporary store used to hold the present instruction while it is being obeyed.

Control Unit: that part of the computer which controls the operations of the machine.

Decoder: A logical unit, providing an output on one of a number of lines for each pattern of digits fed into the unit.

Encoder: a logical unit, energising one or more output lines for each of its input lines.

Engine: another word for computer.

Fixed Point: used to describe a machine in which the binary point is fixed, commonly between the first two most significant digits.

Floating Point: used to describe a machine in which the binary point effectively floats, the number being stored in the form $a.2^b$.

Function: name given to the portion of an instruction which tells the computer the operation to be carried out.

Gate: a logical unit whose output is some logical function of the input; e.g. an AND gate produces an output when all inputs are present and an OR gate produces an output when any input is present.

Instruction: a set of characters which informs the computer of a particular operation together with one or more addresses referring to the location of numbers or further instructions which are contained in the main store.

Jump: an instruction which may be conditional and commands the computer to select a particular storage location for the next instruction rather than the normal sequence.

Location: a storage position; also known as address.

Logical Design: the stage in the formulation of a computer design which comes between the initial statement of

/requirements

Logical Design: (cont'd) requirements and the final engineering design which makes it practical; at this stage use is made of logical elements or units and shown symbolically in Appendix II.

Logical Unit: a unit whose output is a logical function of one or more inputs, as in Appendix II.

Machine: another word for computer.

Memory: a unit which stores information for a long time until required; also known as the Main Store.

Modifier: a quantity which can be used to change the whole or part of an instruction before it is obeyed.

Monostable Unit: a unit having only one stable state and normally used to provide a single pulse.

Normalise: the process of changing a floating-point number into a standard form and modifying the exponent term to correspond with the original value; also known as standardising.

Optimum Programming: programming in such a way that minimum waiting time is required to obtain information from the main store when it is not of the immediate access type.

Order: an alternative name for an instruction

Overflow: the result of exceeding the capacity of the machine; for example, adding together two numbers which are individually within the range but giving a sum which is in excess of the range of the machine.

Parallel Operation: within the machine, the transfer of all digits simultaneously along a number of wires.

Register: a fast access store, normally holding one word used for carrying out arithmetic and control operations.

Selector: a unit used to select a particular unit or part thereof; thus the store selector selects a storage location; input-output selector selects a particular input or output unit usually through a buffer register.

Serial Operation: the transfer of all digits of a word sequentially along a single line.

Subroutine: a set of instructions that is commonly used, such as the Initial Orders which are built into the machine or some special set of orders that is kept externally on tape or cards.

Transfer: to copy information from one part of the computer to another; the transfer may be either in serial or parallel mode.

Word: a set of digits which occupies one storage location.

Appendix II. Symbols

1. Logical Symbols: the main logical symbols used in the logic diagrams are given in Fig. 58.

2. Reference Letters

A	Accumulator
AA	Auxiliary Adder
AC	Arithmetic Counter
AS	Adder/Subtractor Unit
AU	Arithmetic Unit
BAR	B-address Register
BDC	B-Decoder
BnR	B-Register number n
CC	Control Counter
CR	Control Register
ClR	Least significant or address location portion of CR
CmR	Most significant portion of CR
D	Deficit Operation - 2 pulses short
DR	D-Register
F	Fast Operation
FDC	Function Decoder

H	Half-length Operation
HA	Half-adder
HS	Handswitches for manual input
HT	Half Time Waveform corresponding to 11 shift pulses
IR	Input Buffer Register
L	Long Operation
IR	L-Register; the least significant portion of A.
MR	M-Register; the most significant portion of A
N	Normal Operation
NR	Normalise Register
NZ	Not Zero
OF	Overflow (Indicator)
OR	Output Buffer Register
OS	Overshift (Indicator)
RC	Counter Reset to zero
SIR	Signal Lamp Indicator Register
SR	Store Register
SlR	Least significant or address location portion of SR
SmR	Most significant portion of SR

Appendix III. Valve Computer Order Code

Function Symbol	Operation
ST	Stop computer
A → (S)	Transfer contents of accumulator to the store location S and clear the accumulator.
A → A (S)	Copy the contents of the accumulator into the store location S.
A + (S)	Add the contents of location S to the accumulator
A - (S)	Subtract the contents of location S from the accumulator
R → (S)	Transfer the contents of the auxiliary register to the location S and clear the register.
R = (S)	Copy the contents of location S into the auxiliary register.
R x (S)	Multiply the contents of location S by the contents of the auxiliary register, placing the most significant portion of the product in the accumulator and the least significant portion in the auxiliary register.

A ÷ (S)	Divide the contents of the accumulator by the contents of location S, placing the quotient in the auxiliary register and the remainder in the accumulator.
RA (N)	Right shift the contents of the accumulator N places.
LA (N)	Left shift the contents of the accumulator N places.
RD (S)	Read in 5 binary digits from paper tape and place them in the least significant end of location S.
PU (S)	Punch out the 5 binary digits at the least significant end of location S.
J → (N)	Jump to location N for the next instruction
J+ → (N)	Jump to location N for the next instruction if the contents of the accumulator are positive.
B = (S)	Copy the contents of location (S) into the B-Register.

It will be observed that with the exception of A → A(S) and J+ → (N), all the function symbols as punched require only two symbols and a number (S) or (N).

Appendix IV. Transistor Computer Order Code

When punching input paper tape, only the function number, B-address and store location or integer is required, but to explain the order code the following abbreviations are used:

A = Accumulator

B = B-register

C = Control Counter

D = D-register

L = L-register

M = M-register

N = Integer in address portion of control register

HS = Handswitches

S = Store Location

SL = Signal Lamp Register

An inverted comma (') means contents after the operation rather than before, thus:

S' = B means "the contents of S after the operation equals the content of B before the operation; the latter not being cleared unless stated".

Function No.	Operation
0.0.0	Absolute STOP
0.0.N	Normal STOP
0.B.0) 0.B.N)	Optional STOP
(a) 1. $S' = B$	
2. $B' = S$	
3. $B' = B + S$	
4. $B' = B - S$	
5. $B' = N$	
6. $B' = B + N$	
7. $B' = B - N$	
8. $B' = B \,\&\, S$	
(a) 9. $B' = S$, $S' = B$	
(b) 10. $B' = S' = I$	
11.	
12. $C' = N$ if B is + (≥ 0)	
13. $C' = N$ if $B \neq 0$	
14. $C' = N$ if $B' = (B - 1) \neq 0$	
15. $C' = N$ if $B' = (B - 2) \neq 0$	
(c) 16. Add N to next Order	
17. Add S to next Order	
18.	
19.	

0 - 15 are not B-modifiable

Function No. (cont'd)	Operation
(d)20.	$O' = S$
21.	$C' = N$ if A is - (< 0)
22.	$C' = N$ if A is + ($\geqslant 0$)
23.	$C' = N$ if $A \neq 0$
24.	$C' = N$ if overshift in A
25.	$C' = N$ if overflow in A or D
26.	$C' = N$
27.	$C' = N$ and change store
(e)28.	Connect Input/Output unit N
(f)29.	Disconnect Output unit N
30.	
(g)31.	$S' = L$, $S_S' = 0$
(h)32.	$L' = S$, $M' = 0$
(i)33.	$L' = L + S$, $M' = M + S_S$
(i)34.	$L' = L - S$, $M' = M - S_S$
(j)35.	Normalise A with overshift N
(k)36.	Acc. N-place left shift (Arith.)
(k)37.	Acc. N-place right shirt (Arith.)
(l)38.	Acc. N-place left shift (Log.)
(l)39.	Acc. N-place right shift (Log.)
40.	$S' = M$, $A' = 0$
41.	$S' = M$
42.	$M' = S$

Function No. (Cont'd)	Operation
43.	M' = M + S
44.	M' = M - S
45.	S' = S + M
46.	S' = S - M
47.	M' = M ≠ S
48.	M' = M & S
49.	M' = S, S' = M
50.	
51.	S' = D
52.	D' = S
53.	D' = D + S
54.	D' = D - S
55.	D' = N
56.	A' = S x D
57	
58.	D' = A ÷ S
59.	D' = M, M' = D
60.	(SL)' = S
61.	S' = (HS)
62.	
63.	

NOTES ON THE ORDER CODE

(a) Top digits of S remain unaltered; changes in bottom 11 only.

(b) Both B and S are cleared, apart from input to their bottom 5 digits.

(c) No carry into top digits of next order.

(d) Bottom 5 digits of S are punched out.

(e) N is odd for input units; N is even for output units.

(f) Connecting an input unit disconnects previous input unit.

(g) L is 19 digits; sign digit of S is made 0.

(h) The repeated sign digit of S appears in M.

(i) The sign digit of S is added into or subtracted from the bottom digit of M.

(j) Accumulator has one right shift, then a succession of left shifts, to normalise; associated binary exponent is altered in D; N = 0 means no overshift can be set; otherwise N is smallest impermissible shift.

(k) Left shift can set overflow while right shift repeats sign digit.

(l) Left shift cannot set overflow and right shift does not repeat sign digit.

Appendix V. Initial Orders

The initial orders are provided to read in integers and instructions, provided that the following rules are obeyed:

1. An integer may be preceded by + or - if required, though the former is not necessary.

2. Terminating characters for integers or instructions are Space (Sp) or Carriage Return (CR).

3. The three items of an instruction must be separated by a point (.) and not a space.

4. The Function or B-address may be omitted if zero, but zero store address or integer must be punched.

5. Relative addressing is accomplished by following the address S by v and an integer x. The final address is then the result of adding the contents of store location (100 + x) to S. If the contents of store location (100 + x) is to be subtracted from S, the symbol n is used between S and x. In these cases, if x is zero, it need not be punched and x may also be negative, provided that the final address is positive. v or v0 is reserved for the address of the current instruction.

6. Directives. If the address is followed by an equals (=), the instruction is obeyed at once instead of being placed in the store. These directives are used mainly for instructing the Initial Orders where to place the programme being read in and also when to stop reading in instructions and to begin obeying them when the required accumulator contents must be specified.

7. Titles. All characters following a letter shift (LS) are immediately punched out and not stored. This continues until a line feed (LF) occurs, after which storing again takes place instead of punching out.

A few permissible orders may clarify the above rules:

(1) -720

(2) 5.2.3 Sp

(2) 5.2.3 CR

(4) 26. 60)
) A space would be punched after all
(4) 1.0) instructions but is not indicated in the examples

(5) 23.20v3

(5) 23.20n3

(5) 23.-20v3

(5) 26.-12v

(6) 40.200 =

(6) 27.300 = 0

(7) LS A B C CR LF will be entirely punched out and not stored.

INITIAL ORDERS

0	55. 10	Set D = 10
1	40. 100	[Acc. = 0] Clear 100
2	5.4.2045	B4 = -3 for INITIAL PLUS
3	5.2.0	B2 = 0 sets NOT PARAM
4	5.5.0	B5 = 0 sets INITIAL SHIFT
5	5.3.0	B3 = 0 sets NON ARITH, NOT REFER
6	6.4.2	Add 2 to PLUS/MINUS indicator
7	40. 98	Store or clear in item store
8	31. 97	Store or clear in integer store
9	10.1.96	READ character to B1 and 96
10	32. 86	Put part digit-pattern in L
11.	43. 37	Add other part to M
12.	38.1.0	Shift pattern as per B1
13.	22. 54	Jump if non-digit
14	56. 97	10x c(97) → ACC
15	8.1.30	Remove parity bit in B1
16	1.1.96	True digit → character store
17	33. 96	Add true digit to L
18	6.3.1	Add 1 to B3 for ARITH
19	26. 8	Jump (to store new integer)
20	14.1.57	Jump if not letter shift = 27 (LS)
21	13.5.40	Jump to STOP if SUBSEQ. SHIFT
22	13.3.40	Jump to STOP if ARITH
23	13.1.9	Jump to READ if entry from 94

```
24   20.  96   TITLE OUTPUT from character store
25    6.3.14   Add for M/LF = 13
26   13.3.28   Jump if not M/LF
27   13.1.9    Jump to READ if line feed (LF)
28   10.3.96   TITLE READ TO B3 and 96
29   13.3.31   Jump if not Figure Shift = 0
30    5.1.15   Set for LF condition
31    7.3.27   Subtract for LS = 27
32   13.3.24   Jump to OUTPUT if not LS
33    5.1.0    Set for IS condition
34   26.  24   Jump to OUTPUT
35   14.1.48   Jump if not CR = 30
36   13.3.38   Jump if ARITH
37   26.3.9    Jump to 9 (B3 = 0: for M-pattern at 11)
38   13.2.46   Jump if PARAM
39   17.  99   C(99) = directive becomes next order
40        0    STOP if jump entry
41    5.1.1    Put 1 in B1
42    3.1.99   Add in old directive address
43    1.1.99   Put new address into directive
44    1.1.100    and into address store
45   16.  2046 Make next order 40.2.98 (with B2 = 0)
46   40.2.100  Jump entry, ACC→100 + B2
47   26.  2    Jump (to set INITIAL PLUS)
48   14.1.74   Jump if not Er = 31
```

```
49   26.  9     Jump to READ
50    5.1.1024  INTERLUDE
51   40.1.2047  TO CLEAR        (needs ACC = 0)
                                ( to clear    )
52   14.1.51    MAIN            (   1023      )
53   26.  0     STORE
54    7.1.26    Subtract for + = 26
55   13.1.20    Jump if not +
56   26.  83    Jump (to join with -)
57   42.  97    Integer in 97 →ACC
58   12.3.61    Jump if NOT REFER
59   17.  97    Next order becomes 42.[100 + C(97)]
60   42.  100   C [100 + C(97)] →ACC
61   13.4.64    Jump if not MINUS
62   40.  97    ACC to 97 and clear ACC
63   44.  97    Subtract 97 from cleared ACC
64   14.1.69    Jump if not . = 28
65   38.5.14    Log. left shift, 14 INIT: 11 SUBSEQ.
66    5.5.2045  SET SUBSEQ. SHIFT
67   43.  98    Add previous part-item to ACC
68   26.  5     Jump (with SUBSEQ. SHIFT)
69   43.  98    Add previous part item to ACC
70   14.1.35    Jump if not n = 29
71    5.4.2046  Set for MINUS
72    5.3.1024  Set for REFER AND ARITH
```

```
73   26.  6     Jump (to SUBSEQ. and store item)
74    6.1.21    Add for = = 10
75   13.1.81    Jump if not =
76   13.2.90    Jump if PARAM
77   40.  99    ACC item becomes directive
78    2.1.99    Put directive address to B1
79    1.1.100     and thence to 100
80   26.  4     Jump (with NOT PARAM)
81   14.1.85    Jump if not - = 11
82    5.4.2046  Set for MINUS
83   13.3.40    Jump to STOP if ARITH
84   26.  6     Jump (to SUBSEQ, storing zero)
85   14.1.94    Jump if not v = 12
86   12.4.72    Jump if not INITIAL PLUS
87   13.3.72    Jump if ARITH (to join with n)
88    5.2.2046  Set PARAM (B2 = -2)
89   26.  4     Jump (with PARAM)
90   21.  94    Jump if item is negative
91   40.  98    Item (param. suffix) → 98
92    2.2.98    and address thence to B2
93   13.2.4     Jump for B2 now zero (= positive)
94   15.1.21    Jump if not Sp = 14
95   26.  36    If Sp, treat like CR
```

The above orders will be clarified with the aid of the simplified flow diagram, Fig. 59.

An integer or an integer part of an instruction are converted to a binary number by means of loop I. Following the last digit of an integer or address, there will be a Space, Carriage Return, v or = and, in this case, path II is followed. In the first three cases, the item is stored in the location given by a previous directive, after which the next character is read in by the return path III. If the instruction is a directive, it is placed in location 99 ready to be obeyed and not placed in the main store programme, after which the next character is read in by loop IV.

If letter-shift is read in, then it and the following characters, letters or numbers, are punched out by using the loop V. This separate reading loop is essential if numbers are to be permitted in the title, as otherwise they would be stored. When the line-feed character appears it is punched out and the next character is read in after following the path VI.

The main store is not cleared unless a directive is used to transfer the control to instruction 50 in the Initial Order store.

Appendix VI. Dialling Subroutine

In this programme

B1 is used for the link address,

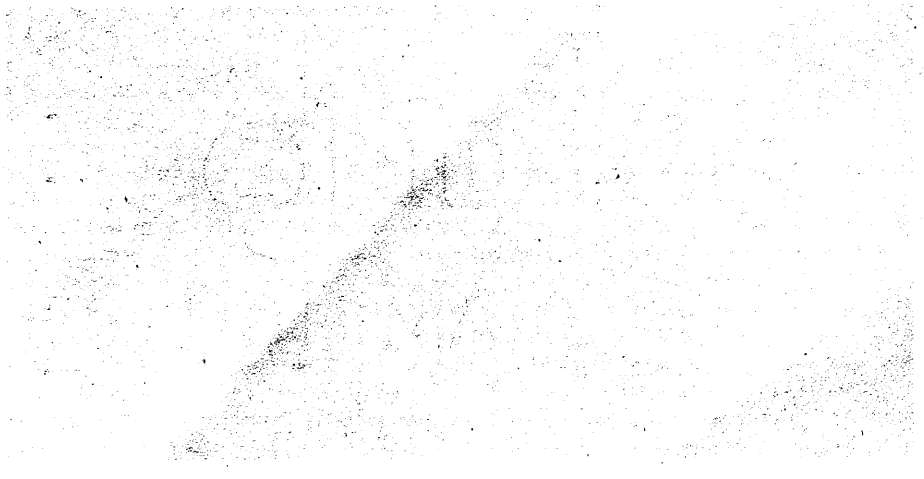
B7 to specify the final location of the number being dialled in.

Loc 1023 is used as working space.

The orders begin in any arbitrary store location m and continue as below:

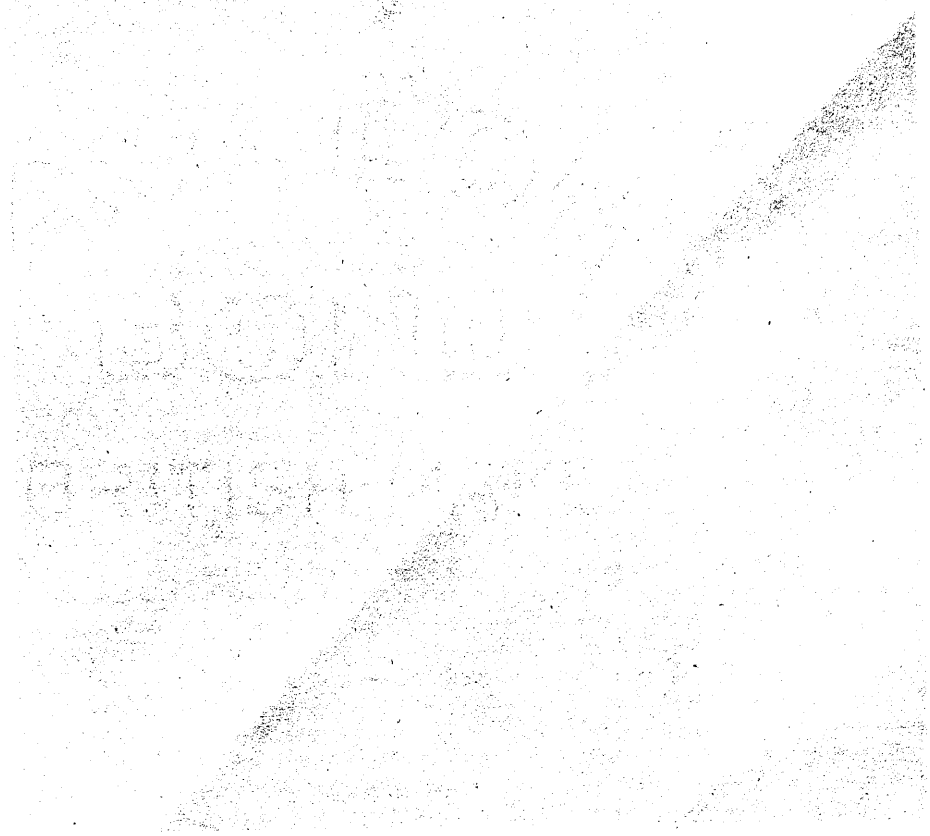
m	55.	10	Set D = 10
m + 1	40.	1023	Clear ACC (into 1023)
m + 2	40.7.0		Clear store specified by B7, by transferring contents of cleared ACC.
m + 3		10	STOP: to await DIAL OR START
DIAL OR START			
m + 4	23.	2v	Jump 2 orders ahead if ACC is NZ (= DIAL)
m + 5	26.1.0		Jump to link address (i.e., obey link) (= START)
m + 6	51.	1023	Copy 10 from D into 1023
m + 7	44.	1023	M' = M - 10: subtract 10 (from 1023) from dial entry.
m + 8	22.	2v	Jump 2 orders ahead if ACC is + (ACC now zero, dial entry was 10 for digit zero).

m + 9	43.	1023	If ACC was -, add 10 from 1023 to M to restore digit.
m + 10	40.	1023	Put new digit from ACC into 1023 and clear ACC
m + 11	56.7.0		10 x (prev. integer: in store designated by B7) L
m + 12	33.	1023	L' = L + new digit = new integer
m + 13	31.7.0		Store new integer (from L) in store designated by B7
m + 14	40.	1023	Clear ACC (into 1023: now finished with).
m + 15	26.	-12v	Jump back 12 orders (to STOP)



Appendix VII

Reprint of Reference 56



681.142:373.63

An Analogue Computer for Teaching Purposes

by P. A. V. THOMAS, B.Sc.(Eng.), A.M.I.E.E.*

Because computers are not available for demonstrational purposes, a low-priced, general-purpose analogue computer has been developed in Glasgow University for explaining the principles of analogue computing circuits.

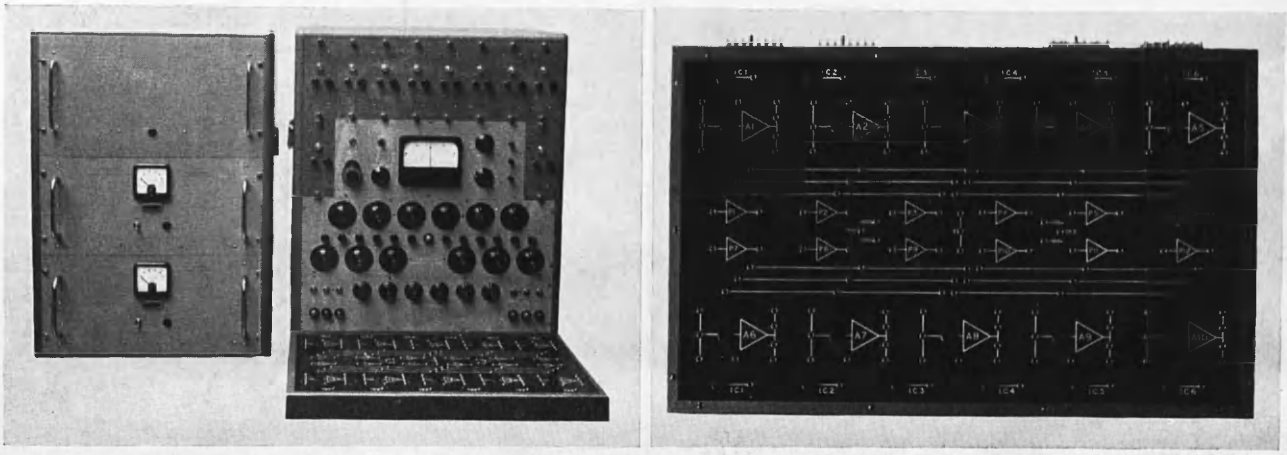


Fig. 1 (left). General view of the Glasgow University Teaching Analogue Computer (GUTAC). (right) Plan view of the problem board.

THE Glasgow University Teaching Analogue Computer (GUTAC) was developed in the Electrical Engineering Department to fulfil two objects: Firstly it was intended to demonstrate the principles of analogue computing circuits, and, secondly it was designed for use as a general-purpose, real-time computer having a reasonable accuracy. To meet these needs, it was considered desirable that the computer should be portable, flexible and easily operated, and at the same time be of moderate cost. The resulting computer is shown in Fig. 1(a). From this it will be seen that the power supply is situated on the left and the computing unit on the right. The lower front portion of the latter consists of a separate plug-in unit. The computer unit is 24 by 12 by 30 in. high and the power unit is 21 by 15 by 30 in. high.

The Computing Unit

GUTAC follows conventional real-time computer principles in consisting of high-gain, d.c.-coupled amplifiers with suitable feedback networks and coefficient potentiometers. Referring to Fig. 1 the ten amplifiers are seen at the top of the computing unit, arranged around the monitor and calibrator panel which contains the central meter. Below this panel is mounted the coefficient potentiometer panel, carrying twelve potentiometers and setting-up switches. Finally, below this, is a panel with six potentiometers which provide the d.c. potentials for the initial conditions required for the integrators, etc. and also the six necessary control switches. Reading left to right, these are, 'a.c. supply on', '$\pm$ 300 volts d.c. on' (for the amplifiers), '$\pm$ 100 volts d.c. on' (reference supply), 'compute/reset', 'hold' and a spare switch.

Careful investigation showed that little advantage was to be gained by designing and constructing the d.c. amplifiers in the laboratory, since those available commercially were obtainable at a strictly competitive price. The type selected had a.c. drift correction circuits included, giving a d.c. gain $>30 \times 10^6$, a bandwidth of 16 kc/s with a gain of 10, a daily drift of $\not> 20\ \mu V$ and an output voltage of ± 100 volts into a 10-kΩ or 20-kΩ load resistor. Each of the coefficient potentiometers is a 50-kΩ wire-wound resistor, the input to each being taken through a three-position setting-up switch, arranged either above or below the corresponding potentiometer. The other end of each potentiometer is connected to earth.

The initial condition potentiometers are also of the 50-kΩ wire-wound type. They are connected across the ± 100 volt reference supply, and are brought into use by means of reset relays operated by the 'compute/reset' switch. A second set of relays (hold) enable the compu-

*Department of Electrical Engineering, The University, Glasgow, W.2.

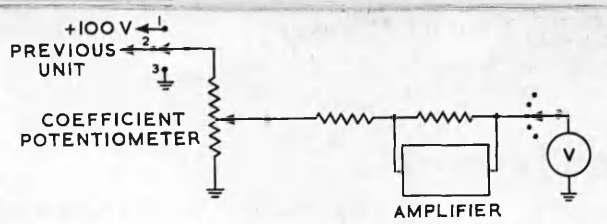


Fig. 2. Setting-up procedure for method 2.

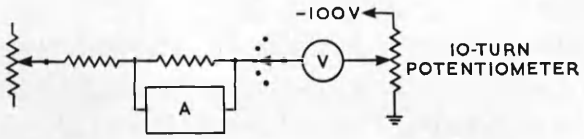


Fig. 3. Balance system used for method 3.

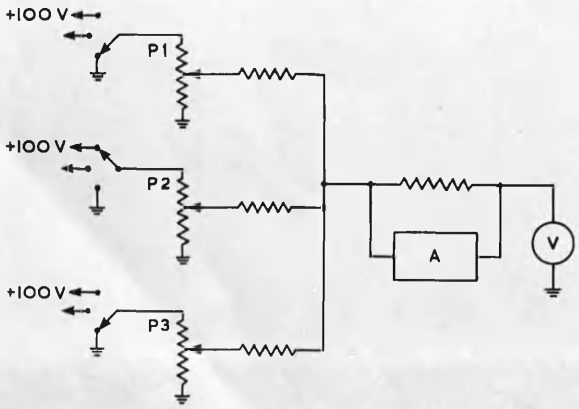


Fig. 4. Multiple input procedure. P2 represents the particular potentiometer being set-up.

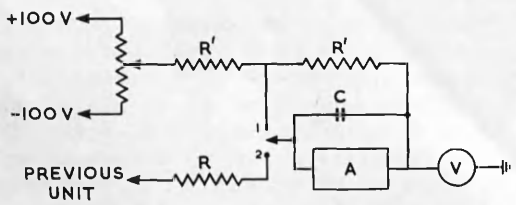


Fig. 5. Integrator set-up.

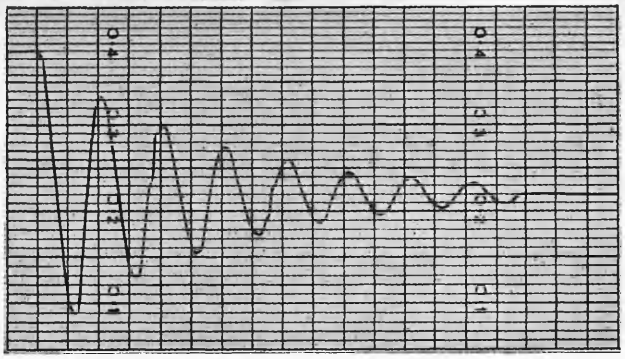
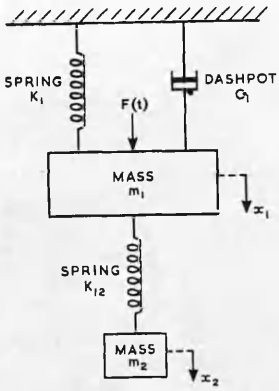


Fig. 6. Solution of a second order differential equation, $\ddot{x}+a\dot{x}+bx=O$.



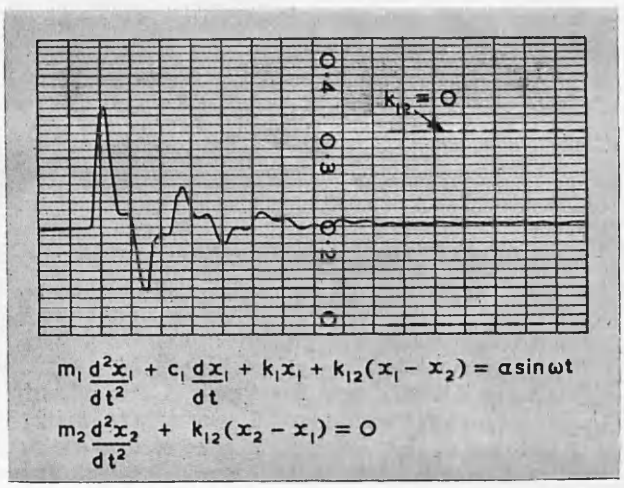


Fig. 7. Solution of a problem which involves the mechanical vibration damper shown in the diagram above.

tation to be stopped at any instant, either to make measurements or to change the coefficients.

The Problem Board

To provide maximum flexibility, the problem is set up on a problem board which is plugged into the front of the computing unit. It will be seen from Fig. 1 that the inputs and outputs of all the amplifiers (A1 to A10) and the coefficient potentiometers (P1 to P12) are brought out to sockets on the insulated panel. In addition to these are added the initial-condition potentials (I.C.1 to I.C.6), twelve bus-bars, ± 100 volts and two test points (T.P.1 and T.P.2). The computing resistors were made up in plug-in tubular form but, because of their greater physical size, the capacitors were assembled in a separate box. The values of the elements were standardized to nominal values of 100 kΩ, 1 MΩ, and 1 μF respectively.

By using a separate plug-in problem board, the flexibility is increased. A number of boards can be available and quickly interchanged for demonstration purposes, or they may be set aside if a particular problem is likely to recur in the near future.

The Power Supply

The power supply consists of three units. Two of these are of commercial manufacture and supply +300 volts at $\frac{1}{2}$ amp and −300 volts at $\frac{1}{2}$ amp for the amplifiers. They are highly stabilized by conventional methods. The third unit, made in the laboratory, consists of two sections giving +100 volts and −100 volts as the reference volt-

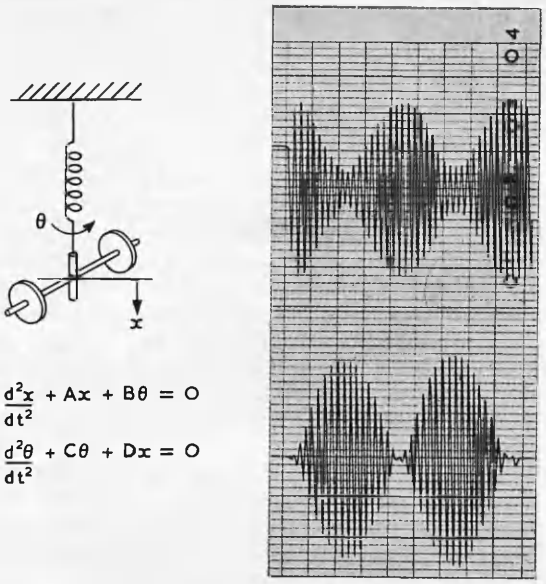


Fig. 8. Solution for the equations of a tuned torsional pendulum.

ages used for initial conditions, setting-up of potentiometers, etc. This unit consists of two stabilizers receiving their inputs from the above ±300 volt supplies. A low-voltage unstabilized supply of about 8 volt for the relay circuits is also provided.

Operating Procedure

The problem is set up on the problem board using the plug-in resistors, capacitors and associated leads. The board is then plugged into the computing unit and the supplies are switched on. After allowing a reasonable warming-up period, each amplifier 'set zero' is checked and all coefficient potentiometers and initial conditions are set up. One of the following three methods is adopted according to the accuracy required:

(1) For demonstration purposes it will usually suffice to read the dials directly.

(2) Where higher accuracy is required, the method shown in Fig. 2 is adopted. Each potentiometer has an associated three-position setting-up switch which is normally in position 2, connecting the potentiometer to the previous unit. To set up the overall constant of the block, the switch is set to position 1, exciting the potentiometer with +100 volt reference. The voltmeter selector switch is then set to the appropriate amplifier output and the potentiometer is subsequently adjusted to give the required value. Thus, if the overall constant for the block is to be 0.8, the voltmeter must read $0.8 \times 100 = 80$ volts. The setting-up switch is then returned to position 2.

(3) For even higher accuracy than method 2, a 10-turn reference potentiometer is used in a balance system, as shown by Fig. 3. In this case, the amplifier output is balanced against the reference potentiometer which is supplied with the −100 volt reference supply—negative due to the phase inversion of the amplifier—the desired overall constant being set up on the reference potentiometer and the voltmeter being used as a null detector.

In some instances an amplifier will receive more than one input with different coefficients. Each potentiometer is then set up individually by earthing the inputs to the other potentiometers by use of the third position of the switch as shown in Fig. 4. Here *P2* represents the particular potentiometer being set-up.

In the case of integrators, the initial output is set up by means of a convenient initial condition potentiometer as shown in Fig. 5. During the set-up procedure the relay contact is in position 1 so that the integrator capacitor *C* is charged to the required voltage, as read on the voltmeter. When the 'compute/reset' switch is operated, the relay contact moves to position 2, connecting the block as an integrator with a time constant CR. Having set up all the required coefficient potentiometers and initial conditions, the recorder selectors are set to the required outputs and the 'compute/reset' switch is operated. The computation can, if required, be stopped manually at any time by operating the 'hold' switch which open-circuits the integrators by means of a second set of relays.

Suitable for Wide Range of Problems

Any linear problem involving not more than a total of ten inversions, additions and integrations can be solved. The overall accuracy obtainable is obviously dependent on the complexity of the problem. As an example, the solution of a second-order differential equation given in Fig. 6 gave results within 2 per cent for the frequency and 5 per cent for the decrement. More complex examples are the mechanical vibration damper illustrated diagrammatically in Fig. 7 and the tuned torsional pendulum shown in Fig. 8.

GUTAC fulfils the requirements of a low-priced, general-purpose, moderate-accuracy analogue computer for teaching purposes, the total cost of material being about £750. It will be observed that no non-linear function units have been incorporated, but it is intended at a later date to construct a third unit housing a multiplier, a function generator and other non-linear units to simulate such components as dead zone and back-lash.

Acknowledgments

The author wishes to express his thanks to the University of Glasgow for permission to publish this paper, to his colleagues for their helpful suggestions and encouragement, and to Mr. G. Boyle who constructed the computer and helped with the mechanical design.

SARAH Helps Oil Survey

SHELL Aircraft Ltd. are using SARAH radio beacons to help oil-survey operations in the Trucial Oman Coast area. SARAH, which is manufactured by the Special Products Division of Ultra Electric Ltd., is the small self-contained radio transmitter which provides a beacon signal on which aircraft, land vehicles and sea vessels can home. Its principal application is in locating survivors of air crashes, and the version of SARAH being used in the Persian Gulf is a variant of the equipment supplied to Commonwealth and N.A.T.O. military forces. In the civilian beacon system, which is designated as RB52K, the speech facility has been omitted.

The survey area which is being worked is entirely featureless, and due both to heat haze and dust it is normally impossible to identify any sort of visual position marker from the air. It is essential, however, that the point at which the previous day's operations were terminated should be located. SARAH receiving equipment has accordingly been installed in two WS.55 helicopters. In the first trial run, with the beacons set up at points unknown to the pilots, homing signals were received at a distance of 15 miles, from an altitude of 1,500 ft, and successful homings made. During the last four months SARAH has proved a valuable aid to surveying in extremely difficult country.

International Transistor Exhibition

Earls Court, London, 21st to 27th May 1959

"SOMETHING for everyone" — that seems a fair summary of the International Transistor Exhibition, which is being promoted by the Electronics and Communications Section of The Institution of Electrical Engineers, in connection with their International Convention on Transistors and Associated Semiconductor Devices. The exhibition is far more than a display of transistors: in fact, the exhibits can be classified under six main headings—semiconductor devices; equipment for the production and testing of such devices; materials; research; components; and equipment using semiconductor devices. The appeal of the exhibition should therefore be wide, for it offers something for everyone in any way concerned with modern trends in electronics.

Although at the time of going to press full details are not available from all the exhibitors, we hope that the following brief survey will be of some assistance to visitors. We would stress that no attempt has been made to mention every exhibit, or even every exhibitor. However, a full list of those who will have stands at the exhibition appears below.

Semiconductor Devices. A good idea of the present-day scope of transistors can be gained from the exhibits of those firms who manufacture these and other semiconductor devices. *The British Thomson-Houston Co. Ltd.* will be showing h.f. and l.f. transistors for amplification and switching applications; germanium point-contact diodes for computers, data processing and television; germanium and silicon junction rectifiers and silicon microwave diodes for radar. A full range of transistors for both computer and h.f./v.h.f. communication applications will also be shown by *Semiconductors Ltd.*

Applications for silicon devices and how they are produced will form the main theme of the display by *Ferranti Ltd.*, which will include low-, medium- and high-power silicon rectifiers, photocells, signal diodes, *p-n-p-n* switches, variable capacitor diodes, fast diodes, and zener reference diodes. The display by *Westinghouse Brake & Signal Co. Ltd.* will include a series of exhibits devoted to semiconductor power control devices, while *English Electric Valve Co. Ltd.* will show examples of air-cooled high-power germanium rectifiers with current ratings in the range 13 to 160 amp. *Newmarket Transistors Ltd.*, in addition to their wide range of transistors, will be showing an automatic measuring and grading machine designed by the company to handle germanium wafers.

From the U.S.A. the display of the *Raytheon Manufacturing Co.* will primarily illustrate the wide range covered in the field of semiconductors, many types now being produced in Europe.

Production Equipment and Materials. The manufacture of transistors has its own problems, and some of the answers to them will be found here. *Radio Heaters Ltd.*, for example, will be showing their 'Radyne' silicon zone-refining unit, and machines for automatically sawing hard materials such as silicon and germanium will be on the stand of *A. & M. Fell Ltd.* A well-known German firm, *Heraeus Quarzschmelze G.m.b.H.*, through their British agents *Fleischmann (London) Ltd.*, will be showing mainly quartz ware for use in the production of transistor materials.

Other companies with exhibits for the manufacturer will be *Edwards High Vacuum Ltd.* (vacuum coating plant, vacuum furnace, potting and degassing outfit, etc.), *Birlec Ltd.*, showing a new pilot dryer designed to dry air and gases for transistor manufacture, and *Elga Products Ltd.*, whose Elgastat Robot type B.106 is a self-contained unit for semiconductor washing. On the testing side, *The Wayne Kerr Laboratories Ltd.* will show transistor adaptors for their r.f. bridge B.601, which are described in New Equipment & Components this month.

A good selection of materials will be seen on the stand of *Johnson, Matthey & Co. Ltd.*—germanium and germanium dioxide, high-purity metals, alloys for semiconductor devices, etc. The General Chemicals Division of *Imperial Chemical Industries Ltd.* will show semiconductor grades of silicon.

Research Exhibits. The stand of the *Research Labora-*

LIST OF EXHIBITORS

UNITED KINGDOM

Barlow-Whitney Ltd.	29
Belling & Lee Ltd.	12B
Birlec Ltd.	34
British Communications and Electronics	12
British Oxygen Gases Ltd.	86
Brush Crystal Co. Ltd.	41
Burndept Ltd.	101
Chapman & Hall Ltd.	18
Dawe Instruments Ltd.	27
Dependable Relay Co. Ltd.	33
Edwards High Vacuum Ltd.	23
Electronic Engineering	73
Elga Products Ltd.	52
Elliott-Automation Ltd.	63
English Electric Valve Co. Ltd.	28
Evans Electroselenium Ltd.	81B
Ever-Ready Co. (Great Britain) Ltd.	30
A. & M. Fell Ltd.	50
Ferguson Radio Corporation Ltd.	11
Ferranti Ltd.	74
General Electric Co. Ltd.	55
Hatfield Instruments Ltd.	17
Fleischmann (London) Ltd.	39
Imperial Chemical Industries Ltd.	104
The Institution of Electrical Engineers	15
Instrument Review	31
International Rectifiers (G.B.) Ltd.	20
Johnson, Matthey & Co. Ltd.	47
Kynmore Engineering Co. Ltd.	62
Livingston Laboratories Ltd.	16
Joseph Lucas (Electrical) Ltd.	70
Mallory Batteries Ltd.	68
Mansol (G.B.) Ltd.	35
Midland Bank Ltd.	19
Mining & Chemical Products Ltd.	51
Newmarket Transistors Ltd.	54
The Plessey Co. Ltd., Components Group	46
Pye Ltd.	24
Racal Engineering Ltd.	82
Radio Heaters Ltd.	72
Rank Cintel Ltd.	80
R.C.A. (Great Britain) Ltd.	85
Rivlin Instruments Ltd.	71
Roband Electronics Ltd.	61
Semiconductors Ltd.	46
Shaw Publishing Co. Ltd.	105
Sintering & Brazing Furnaces Ltd.	78
South London Electrical Equipment Co. Ltd.	93
Standard Telephones & Cables Ltd.	25
Taylor & Francis Ltd.	22
Texas Instruments Ltd.	6
Thermal Syndicate Ltd.	60
Ultra Electric Ltd.	21
Venner Electronics Ltd.	40
The Wayne Kerr Laboratories Ltd.	38
Welwyn Electrical Laboratories Ltd.	67
Westinghouse Brake & Signal Co. Ltd.	12A
Whiteley Electrical Radio Co. Ltd.	77
Wirepots Ltd.	49
Wire Products & Machine Design Ltd.	18A

AUSTRIA

Electrovac Hacht & Huber O.H.G. (Agents – The Roditi International Corporation Ltd., London)	69

BELGIUM

Union Miniere du Haut (Agents – Brandhurst Co. Ltd., London)	63A

FEDERAL GERMAN REPUBLIC

Siemens & Halske A.G. (Agents – R. H. Cole (Overseas) Ltd., London)	36
Te-Ka-De G.m.b.H. (Agents – Neoflex Ltd., London)	57
Telefunken G.m.b.H.	53

FRANCE

Compagnie Francaise Thomson-Houston	32
Compagnie Generale de Telegraphie sans Fil	48

ITALY

Societa Generale Semiconduttori S.p.A.	66B

JAPAN

Sanyo Electric Co. Ltd. (Agents – Marubeni-Ilda Co. Ltd., London)	82A
Tokyo Shibaura Electric Co. Ltd. (Agents – Daiichi Bussan Kaisha Ltd., London)	91A

U.S.A.

Raytheon Manufacturing Co. (Agents – B. & K. Laboratories Ltd., London)	66
Semiconductor Information Service	89
Sylvania Electric Products Inc. (Agents – Sylvania Thorn Colour Television Laboratories, Enfield)	45
Transitron Electronic Corporation (Agents – G. E. Industrial Supplies Ltd., London)	64

RESEARCH SECTION

Associated Electrical Industries Ltd.	81A
British Broadcasting Corporation	92A
British Thomson-Houston Co. Ltd.	59, 100
General Post Office	88
Ministry of Supply	96
Mullard Ltd.	65 & 97
Siemens Edison Swan Ltd.	37
U.K. Atomic Energy Authority	94
University of Birmingham Electrical Engineering Department	92

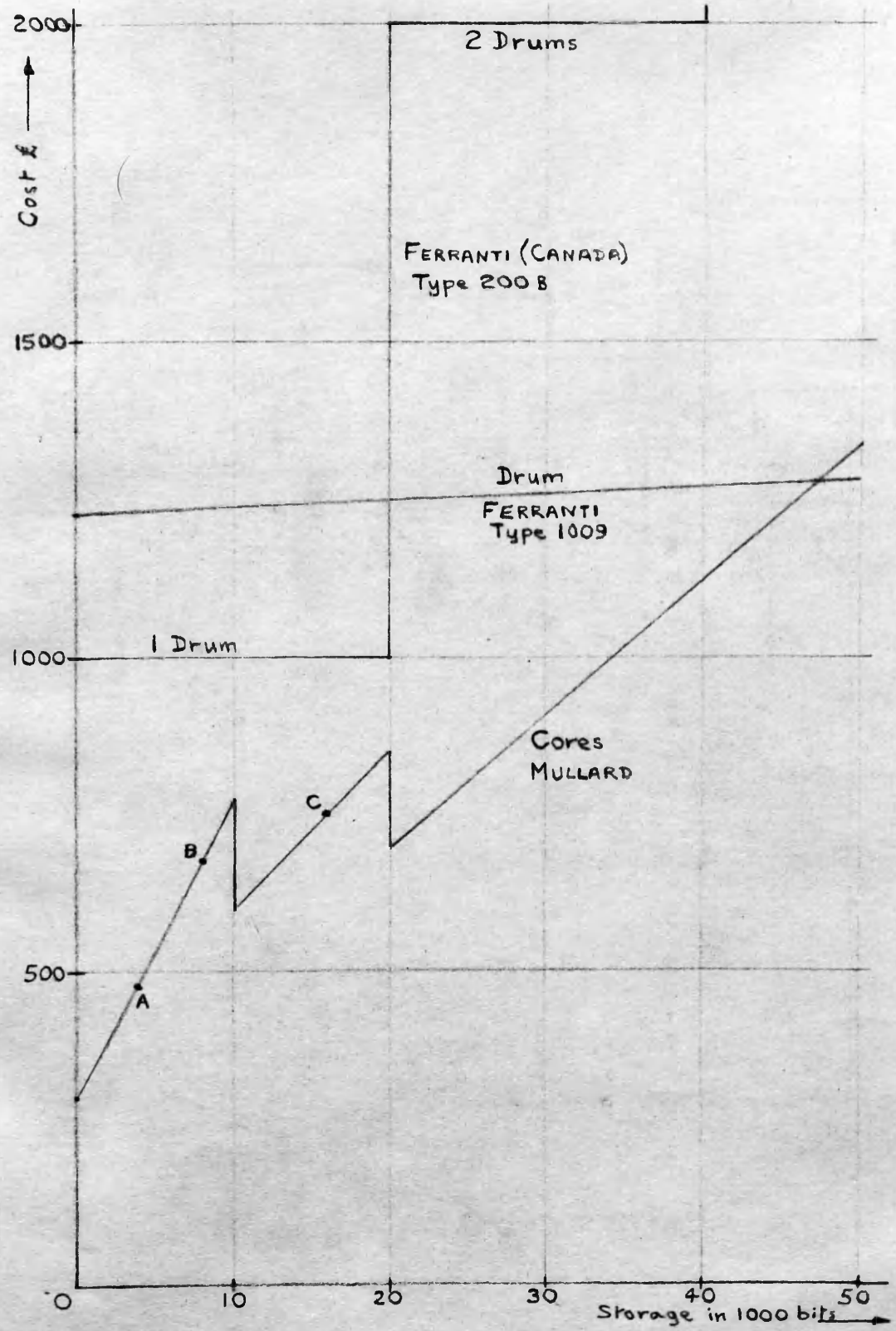


Fig.1. MAGNETIC STORAGE COST (1956)

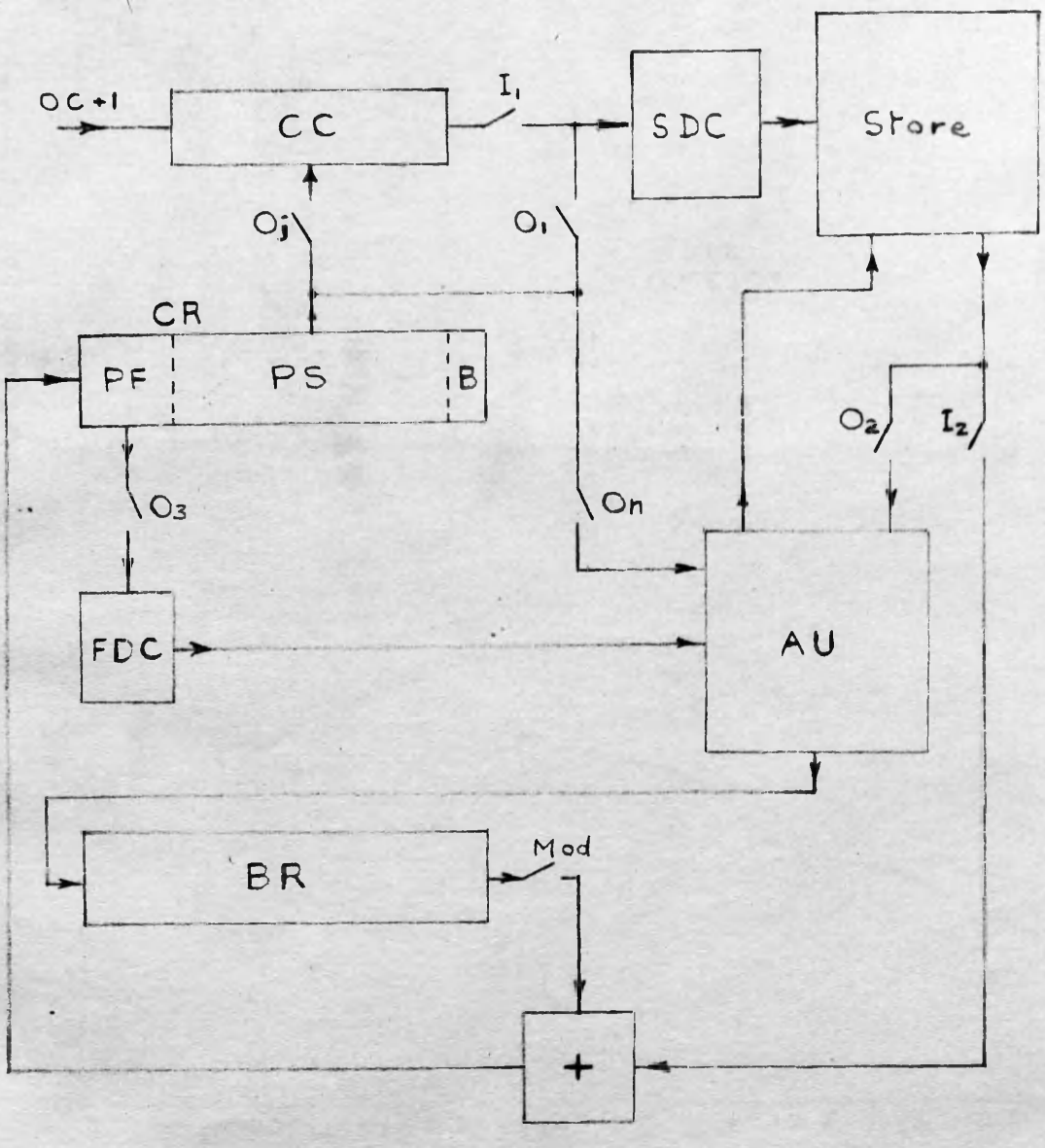


Fig. 2. VALVE COMPUTER BLOCK DIAGRAM

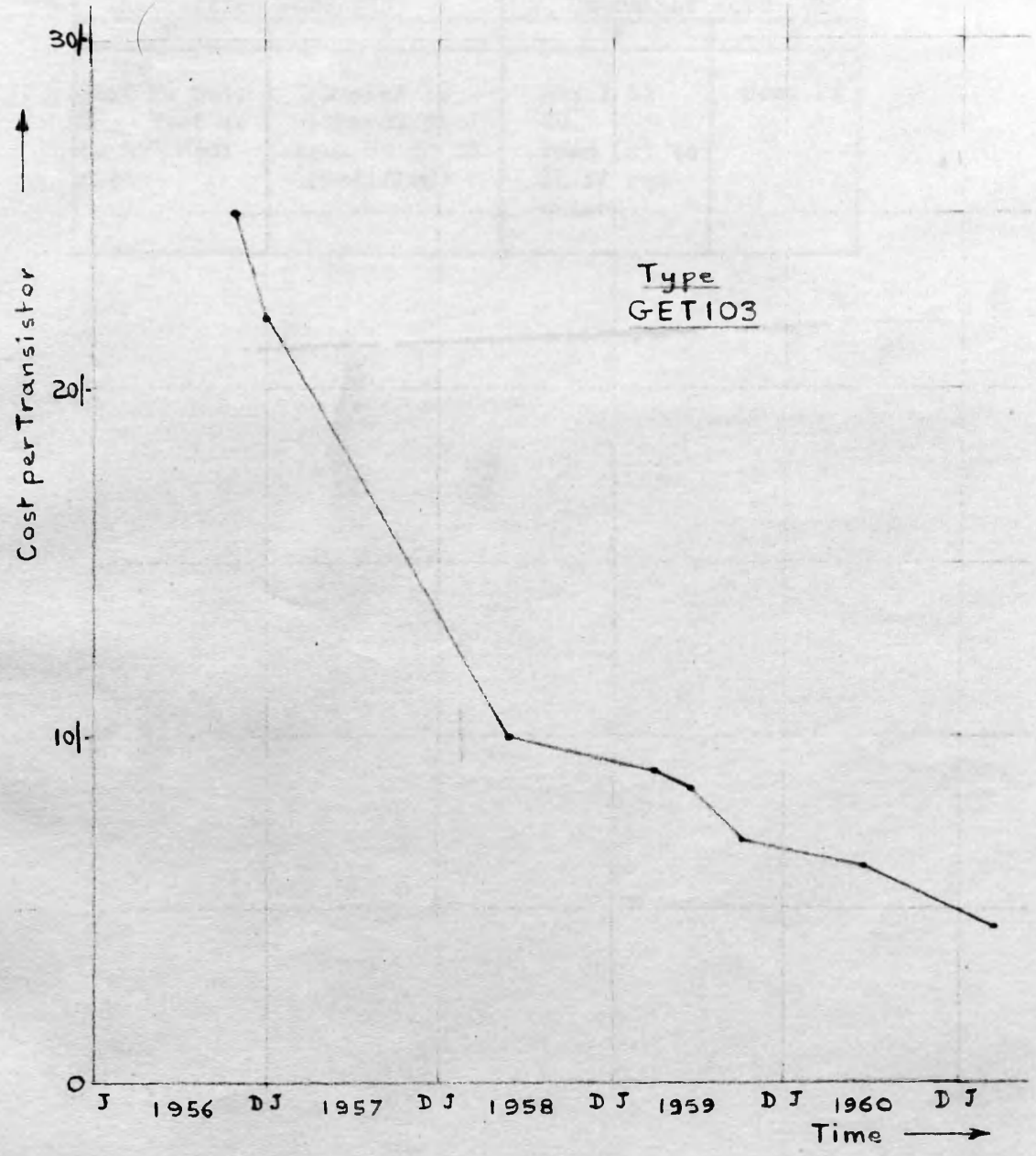
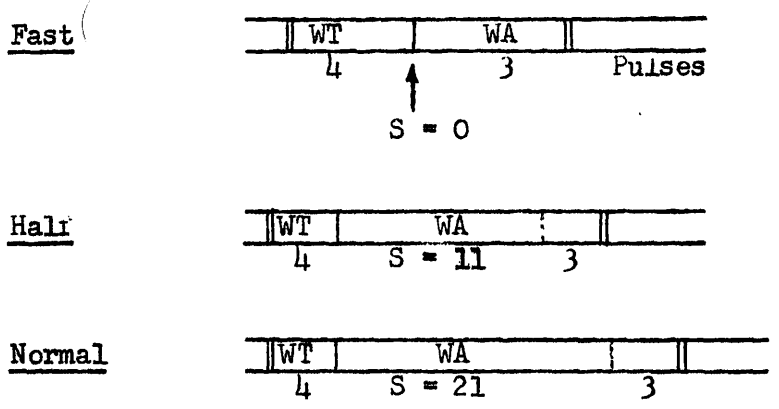


Fig. 3. PRICE OF A SINGLE TRANSISTOR

Instruction Word (IW)		Operation Word (OW)	
P	A	P	A
Read PI into SR. Test if to "B" Modified	Transfer PI (unmodified) from SR to CR (modified)	Add 1 to CC. Read (S) to SR if required	Obey PI

Fig. 4. Basic Machine Rhythm.

Instruction and Operation Words



Operation Words Only

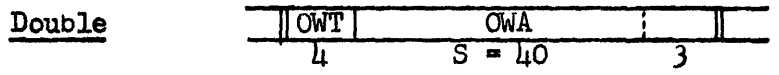


Fig. 5. Instruction Times

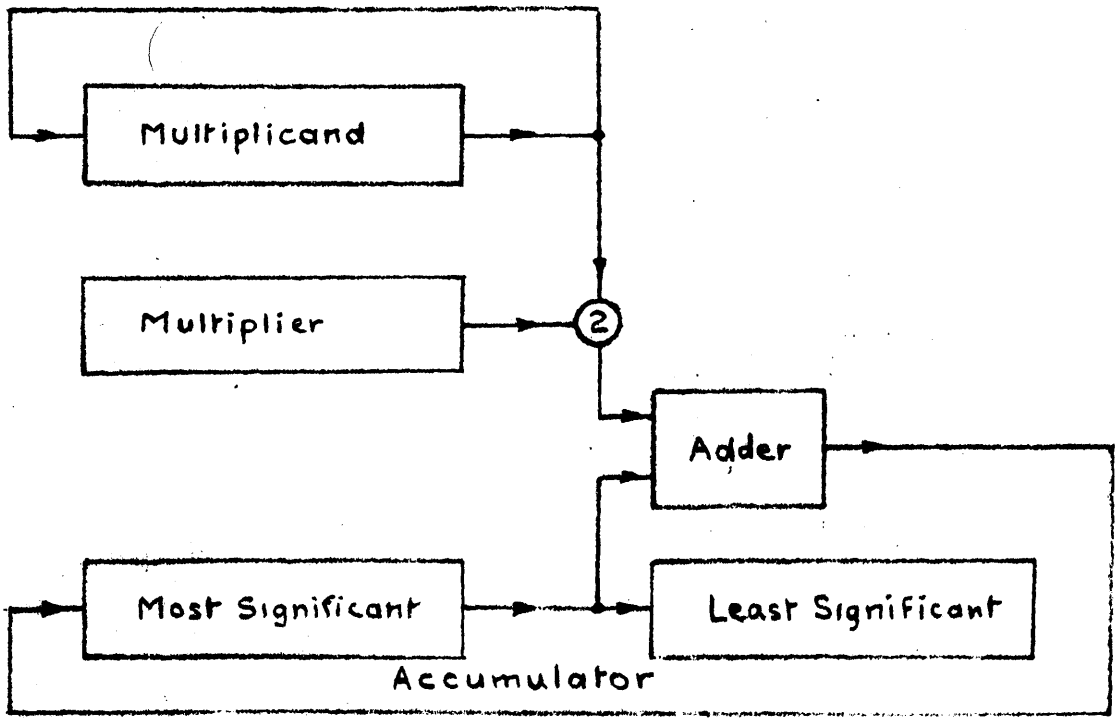


Fig.6. SERIAL MULTIPLIER

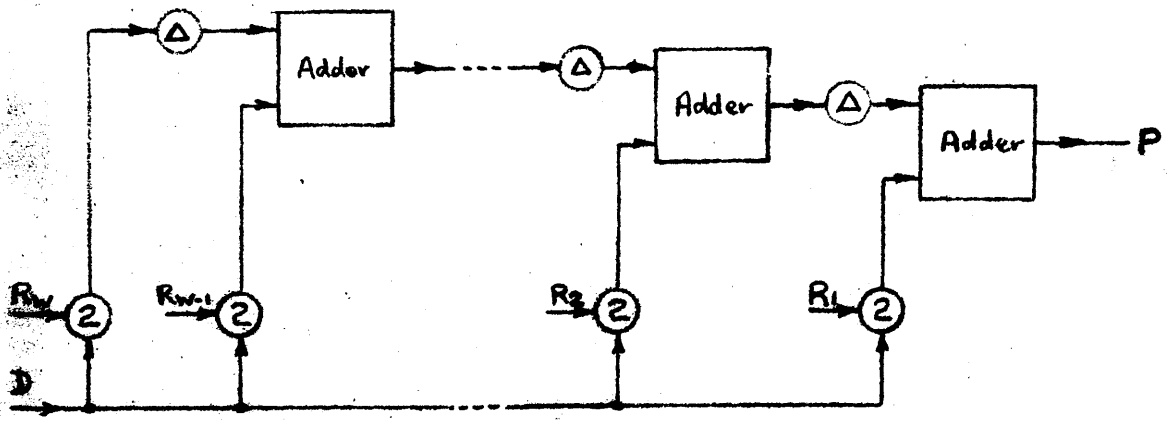
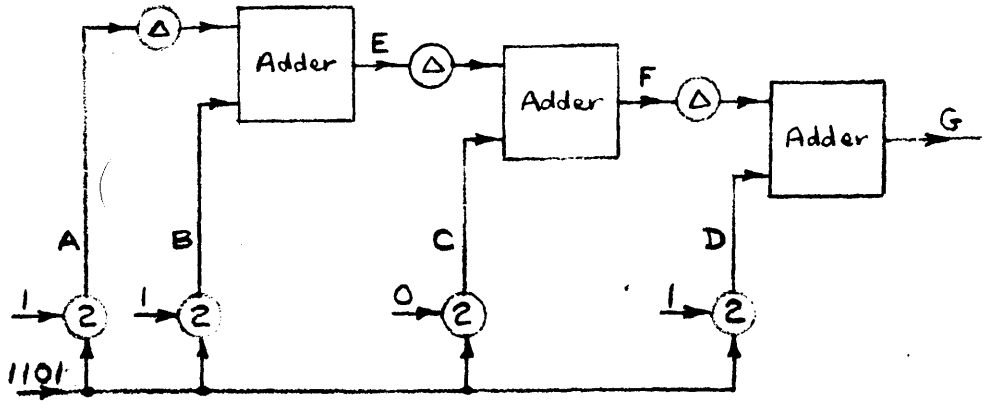


Fig.7. PARALLEL MULTIPLIER



(a) DIAGRAM OF 4-DIGIT MULTIPLIER

Pulse Time	A	B	C	D	E =B+ΔA	F =C+ΔE	G =D+ΔF
1	1	1	0	1	1	0	1
2	0	0	0	0	1	1	0
3	1	1	0	1	1	1	0_c
4	1	1	0	1	0_c	1	1_c
5	0	0	0	0	0_c	0	0_c
6	0	0	0	0	1	0	1
7	0	0	0	0	0	1	0
8	0	0	0	0	0	0	1

Thus output at G, reading upwards, is :

$10101001 = 169 = 13^2$

(b) TABLE OF PULSES

Fig. 8. PARALLEL MULTIPLICATION OF 1101 BY 1101 (13^2)

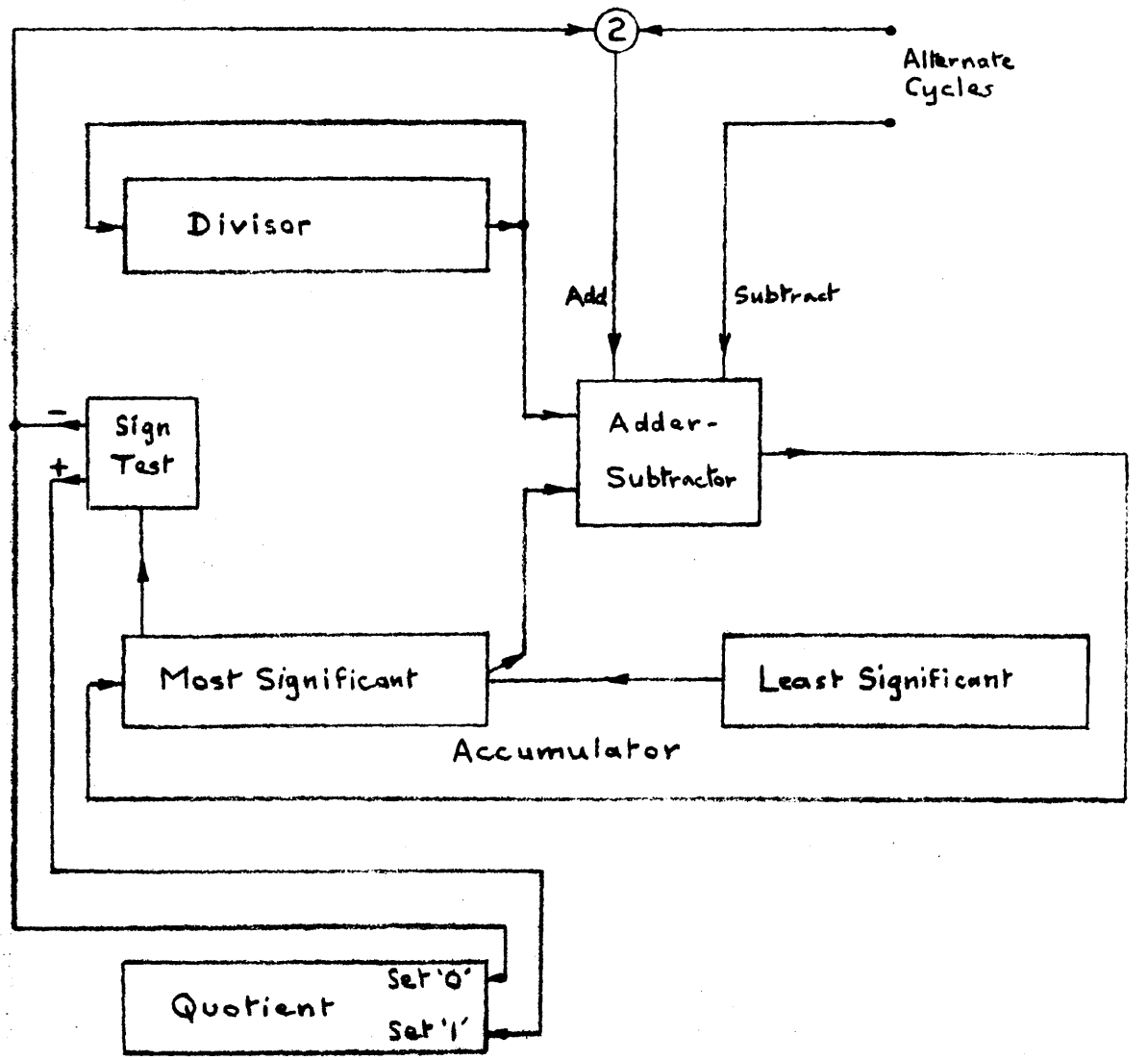


Fig. 9. TRIAL-AND-ERROR DIVIDER

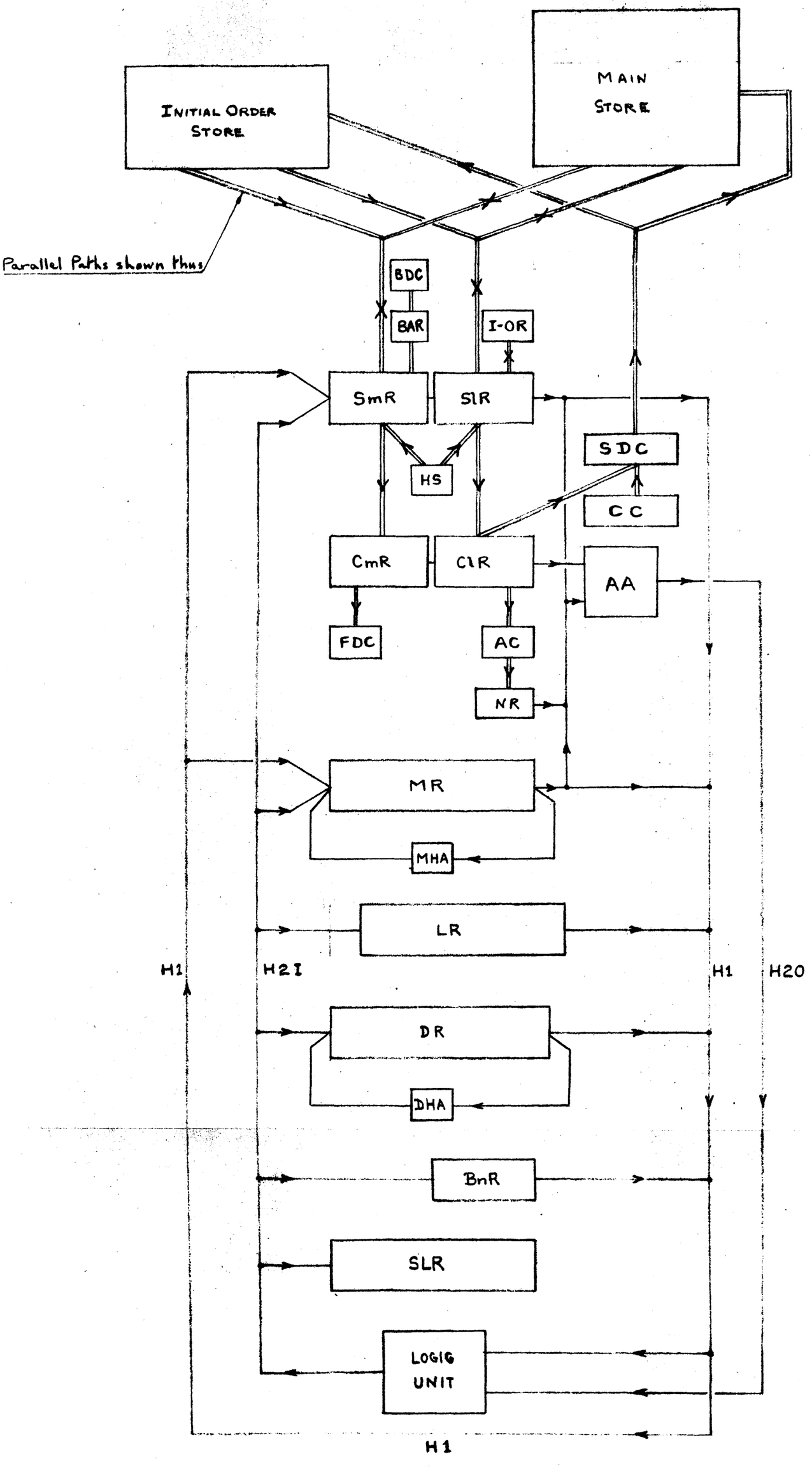


Fig. 10. TRANSISTOR COMPUTER - GENERAL BLOCK DIAGRAM

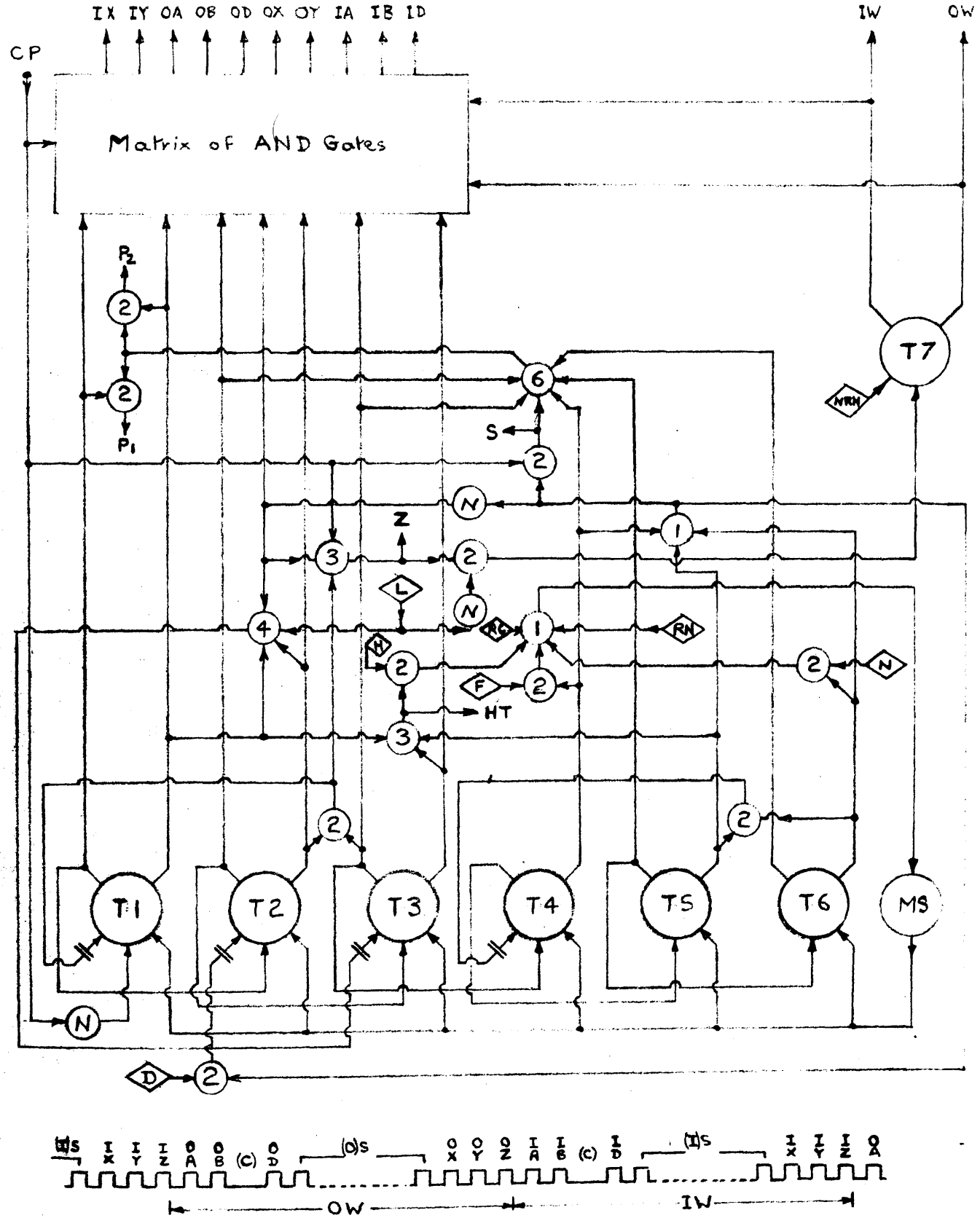


Fig. 11. MAIN TIMING UNIT AND WAVEFORMS

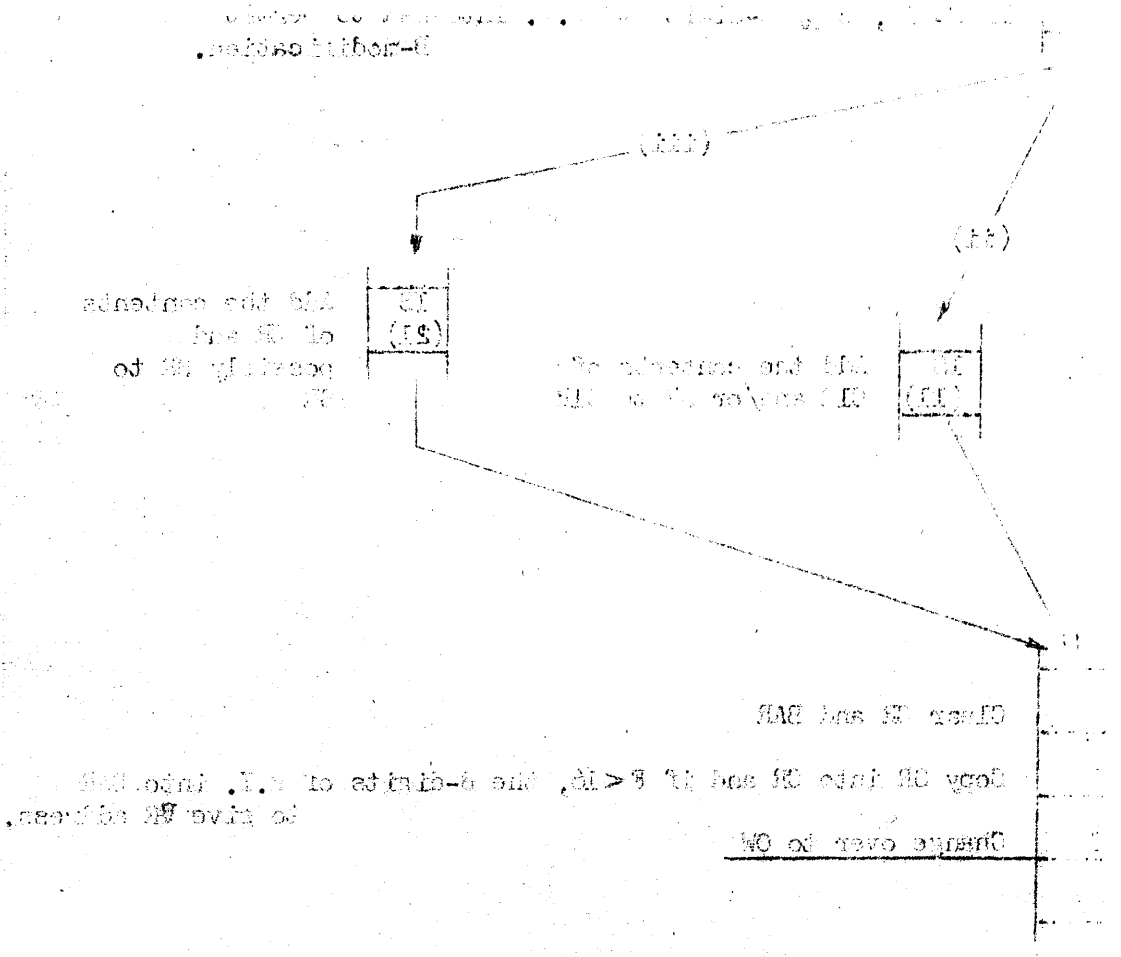
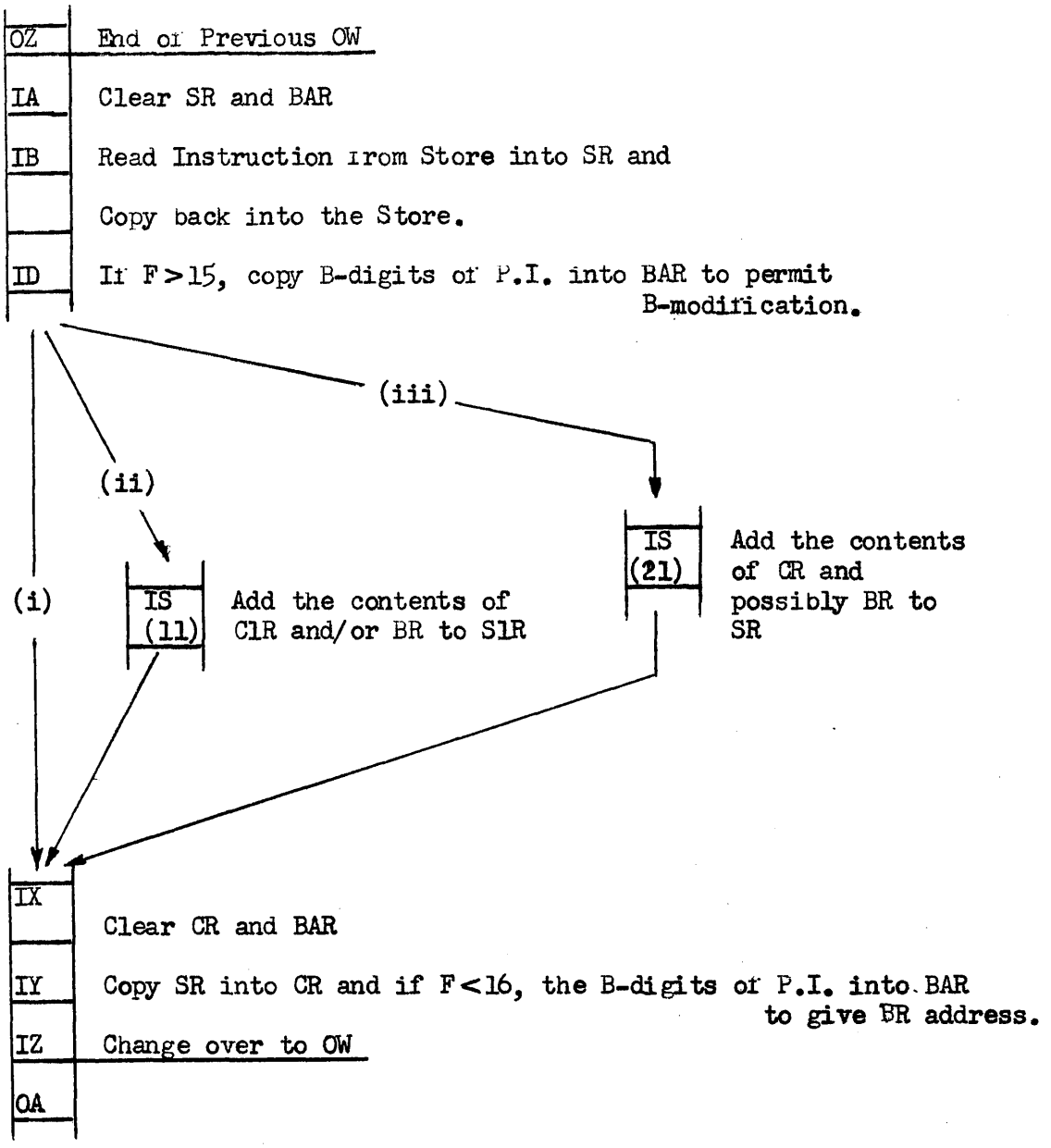
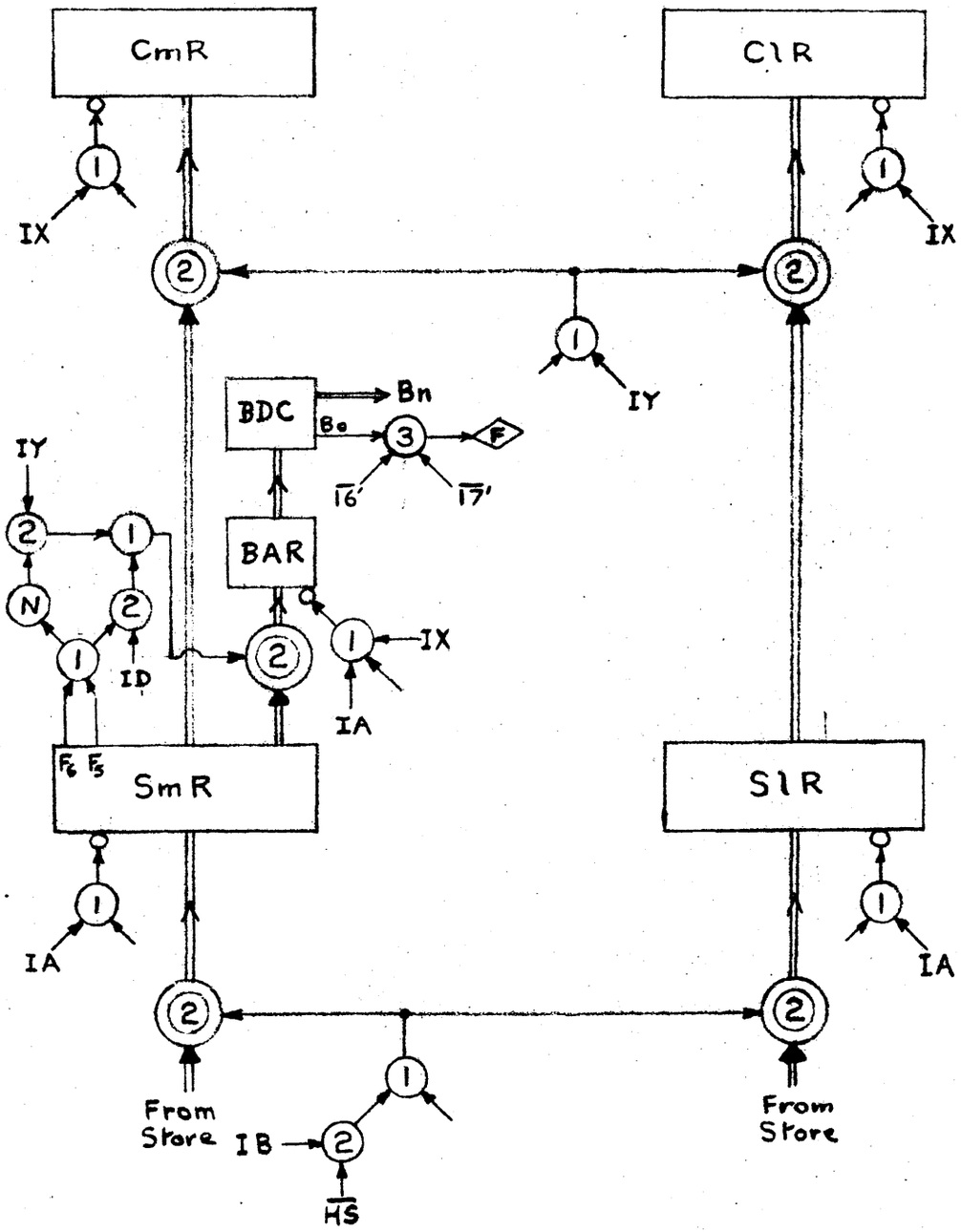
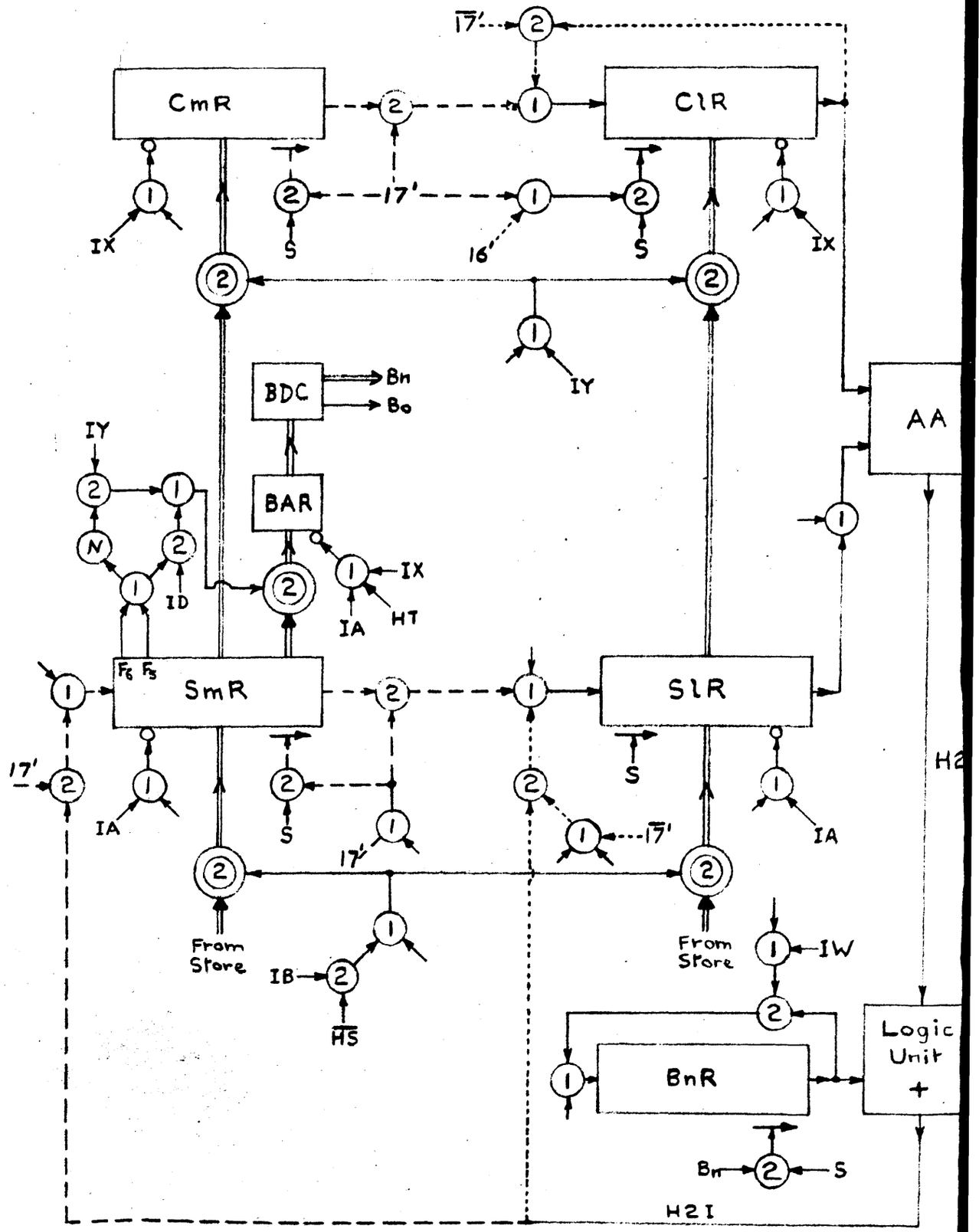


Fig.12. Instruction Word Period



(a) **Flow Diagram**





(b) (Above) UNMODIFIED INSTRUCTION

(c) (Right) MODIFIED INSTRUCTION

LOGIC DIAGRAMS

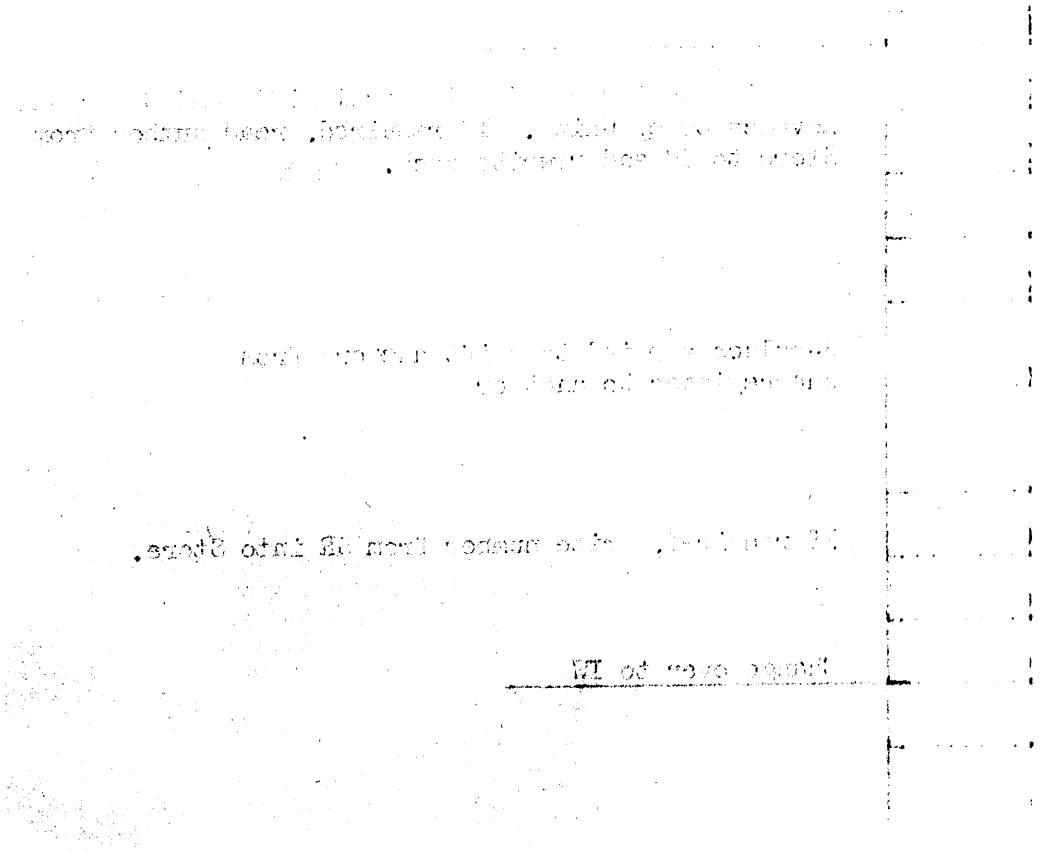
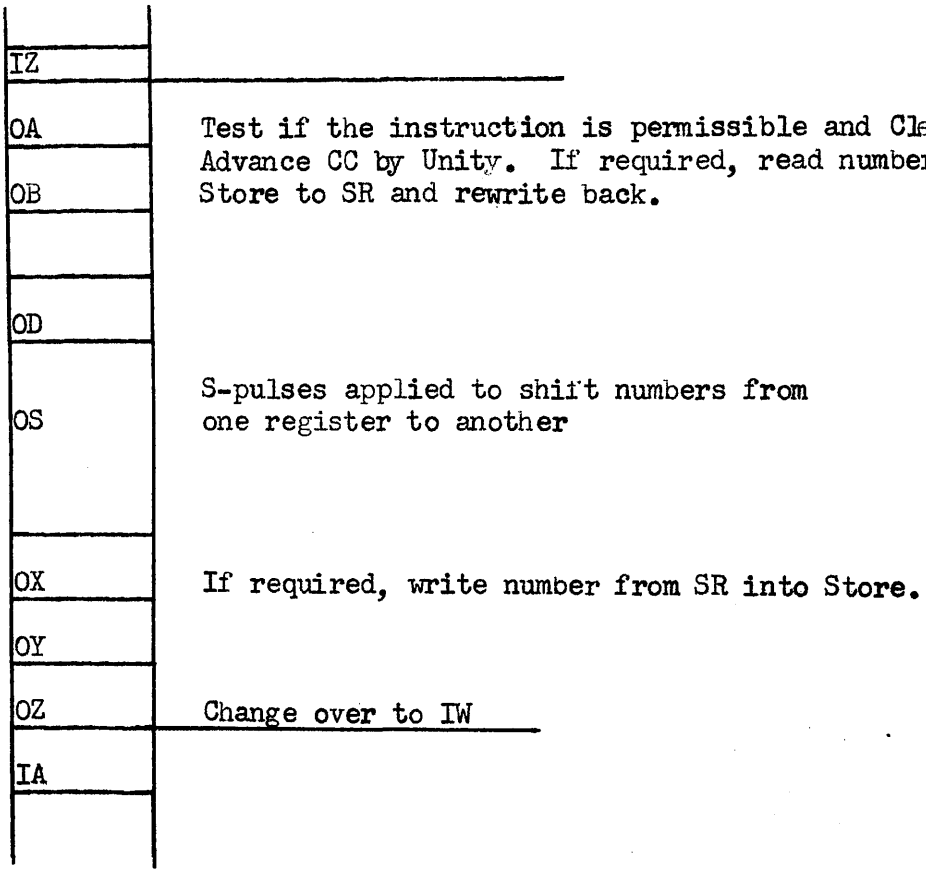
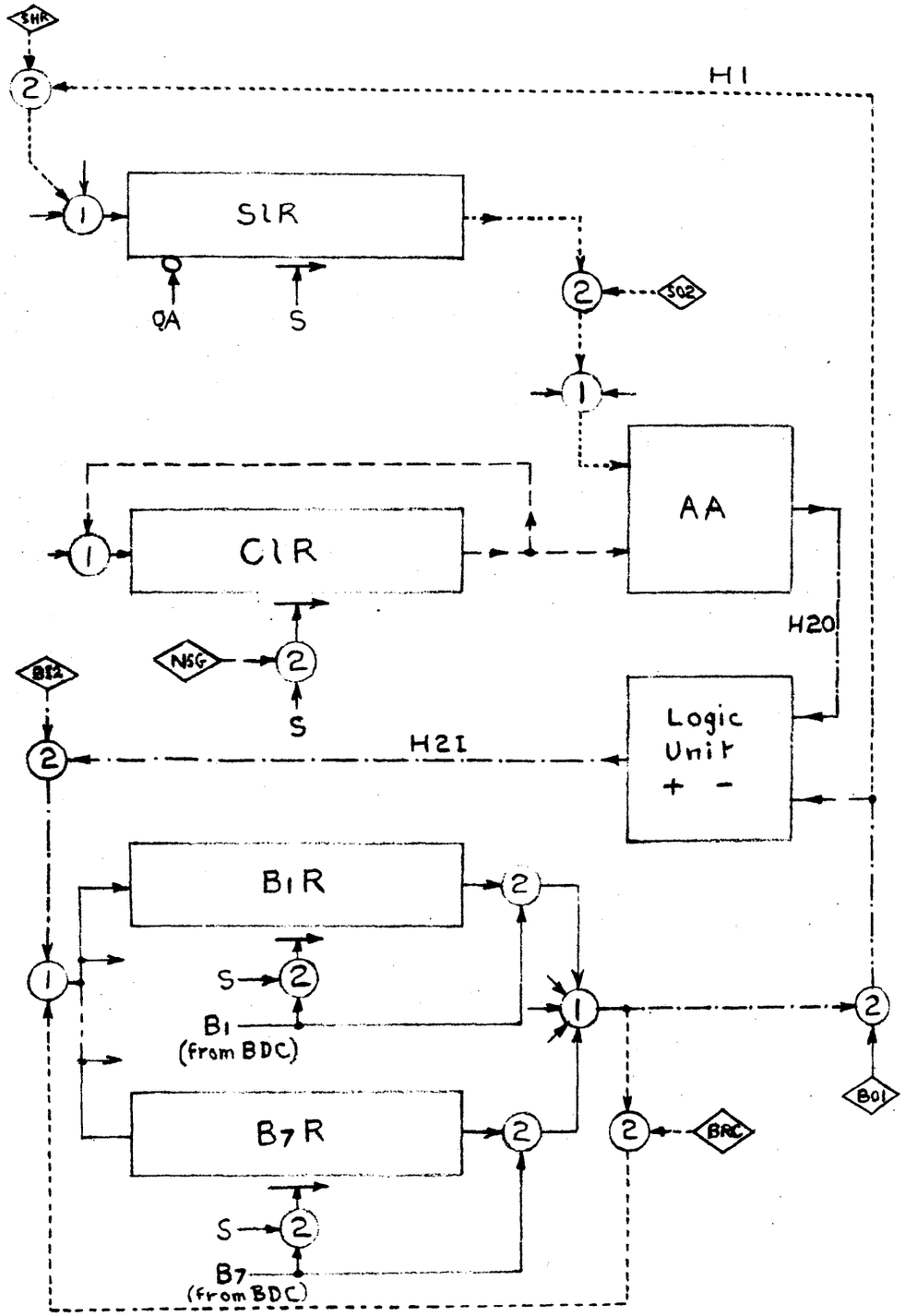
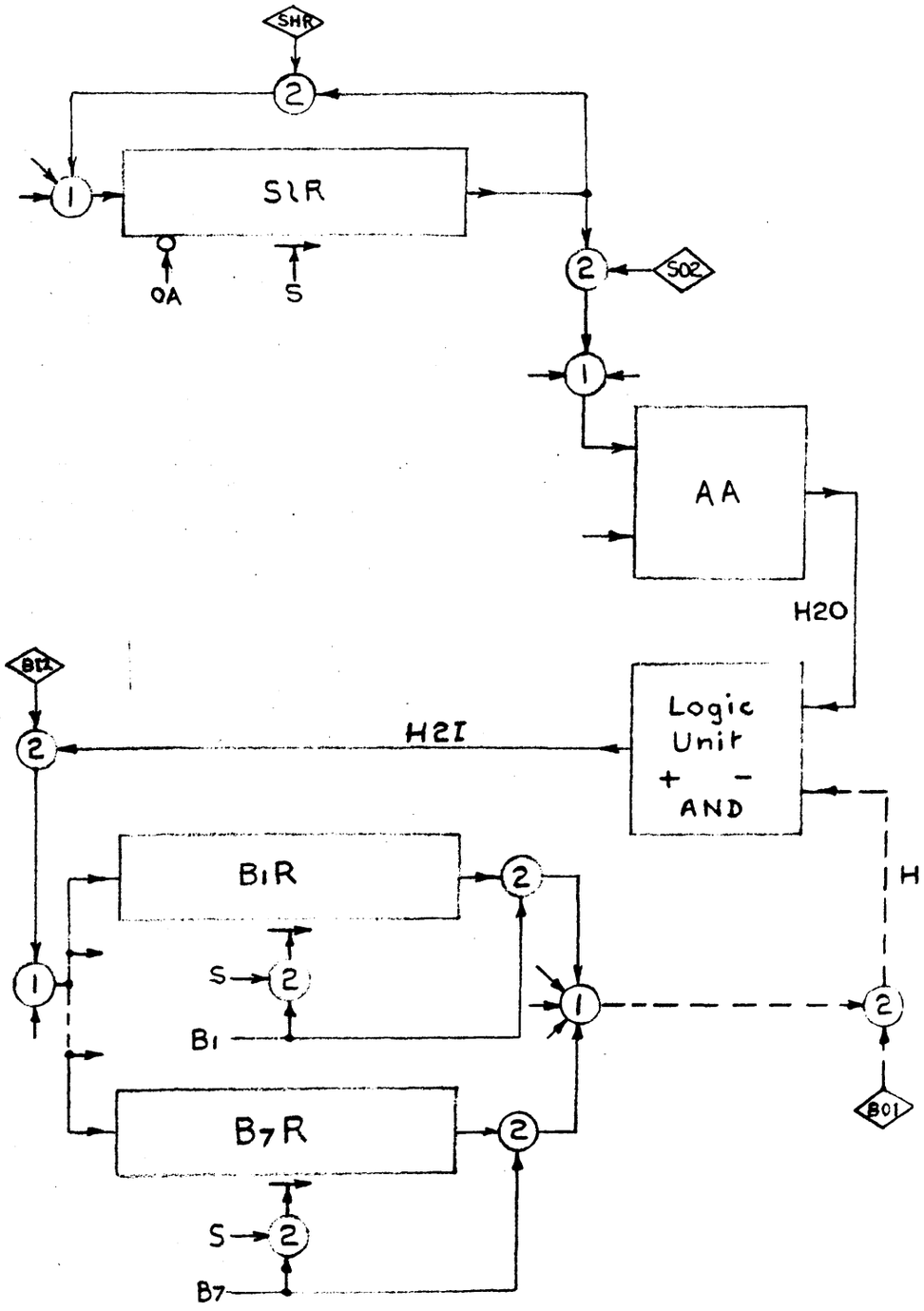


Fig.13. Basic Operation Word Period Flow Diagram


IZ
OA
Test if the instruction is permissible and Clear SR.
Advance CC by Unity. If required, read number from
OB
Store to SR and rewrite back.
OD
OS
S-pulses applied to shift numbers from
one register to another
OX
If required, write number from SR into Store.
OY
OZ
Change over to IW
IA

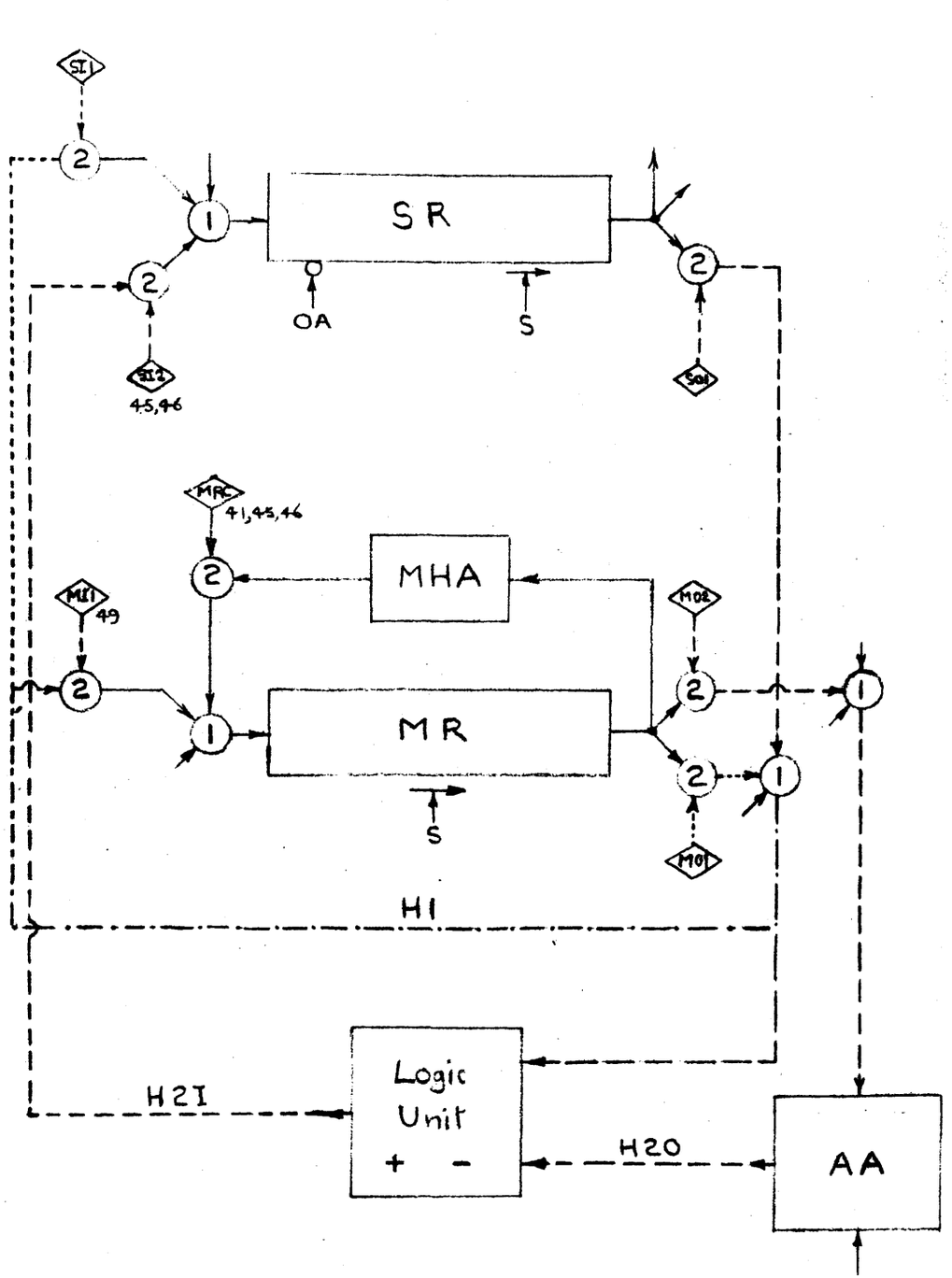


(a) Orders involving Store entry ..... or Control Register ——



(b) Orders involving Store reading and Arithmetic — — — —

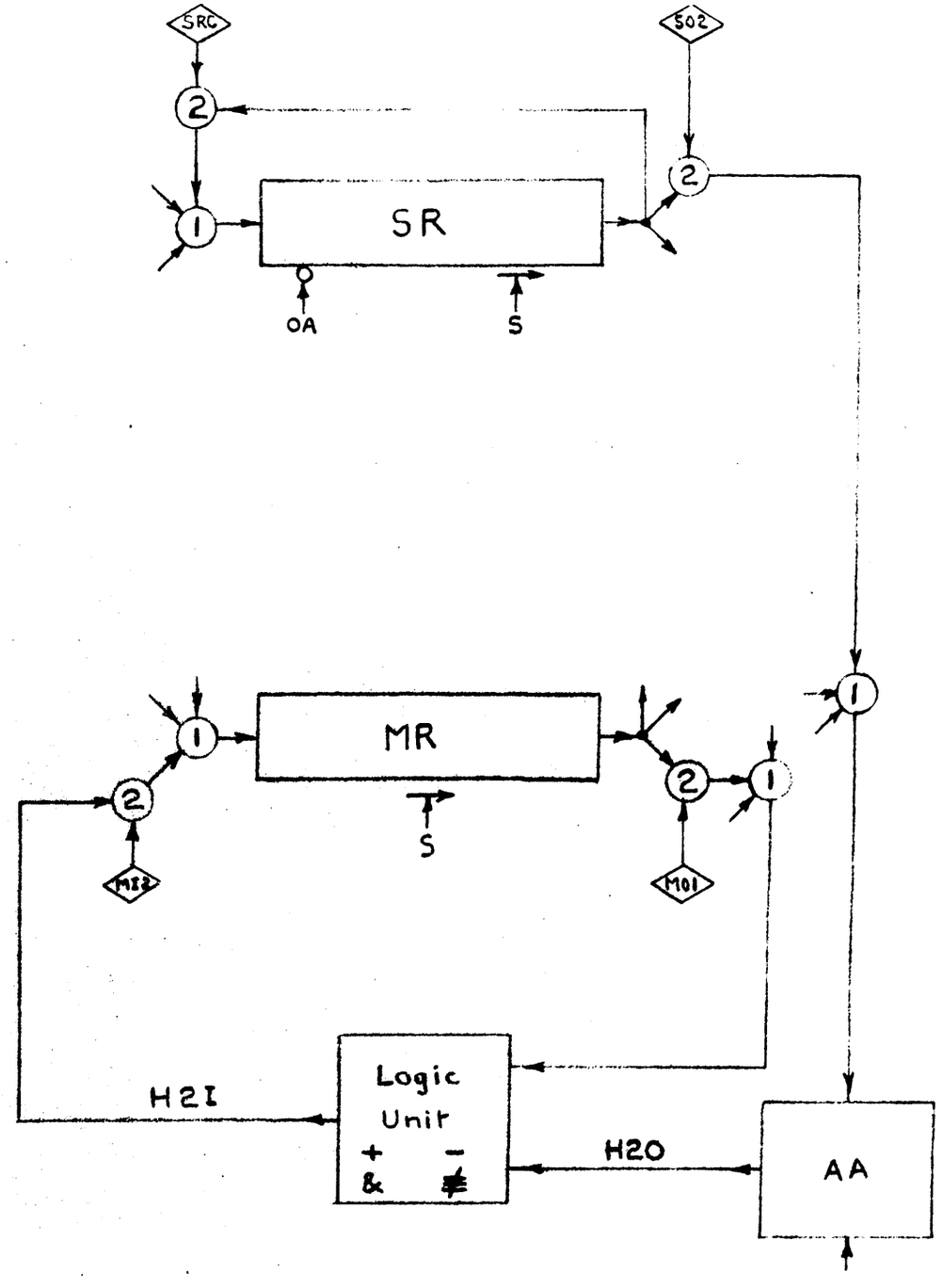
Fig. 14. Logic Diagram for Basic B-Register Instructions



(a) ORDERS INVOLVING CHANGE OF STORE CONTENT

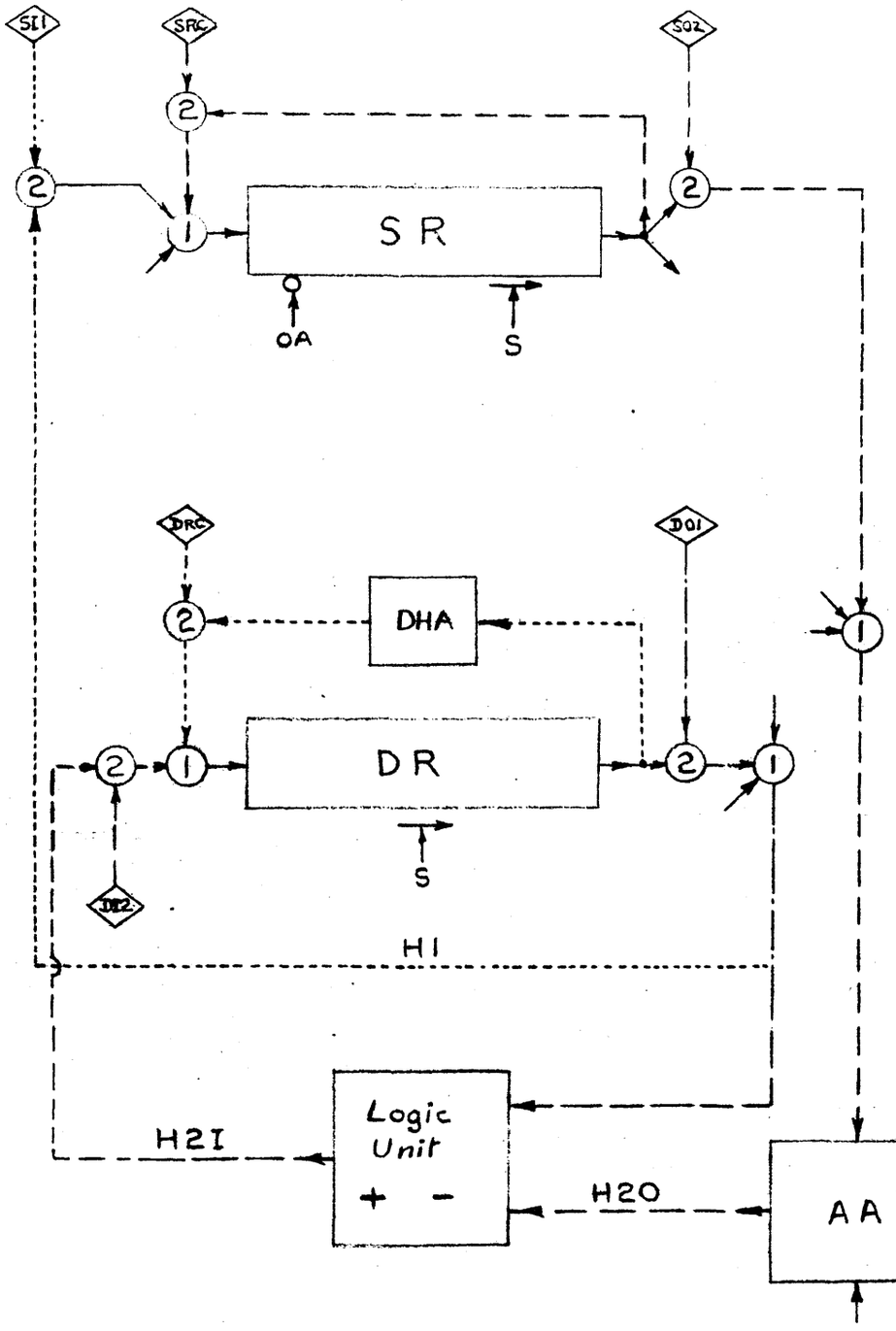
M-REGISTER ONLY ----------

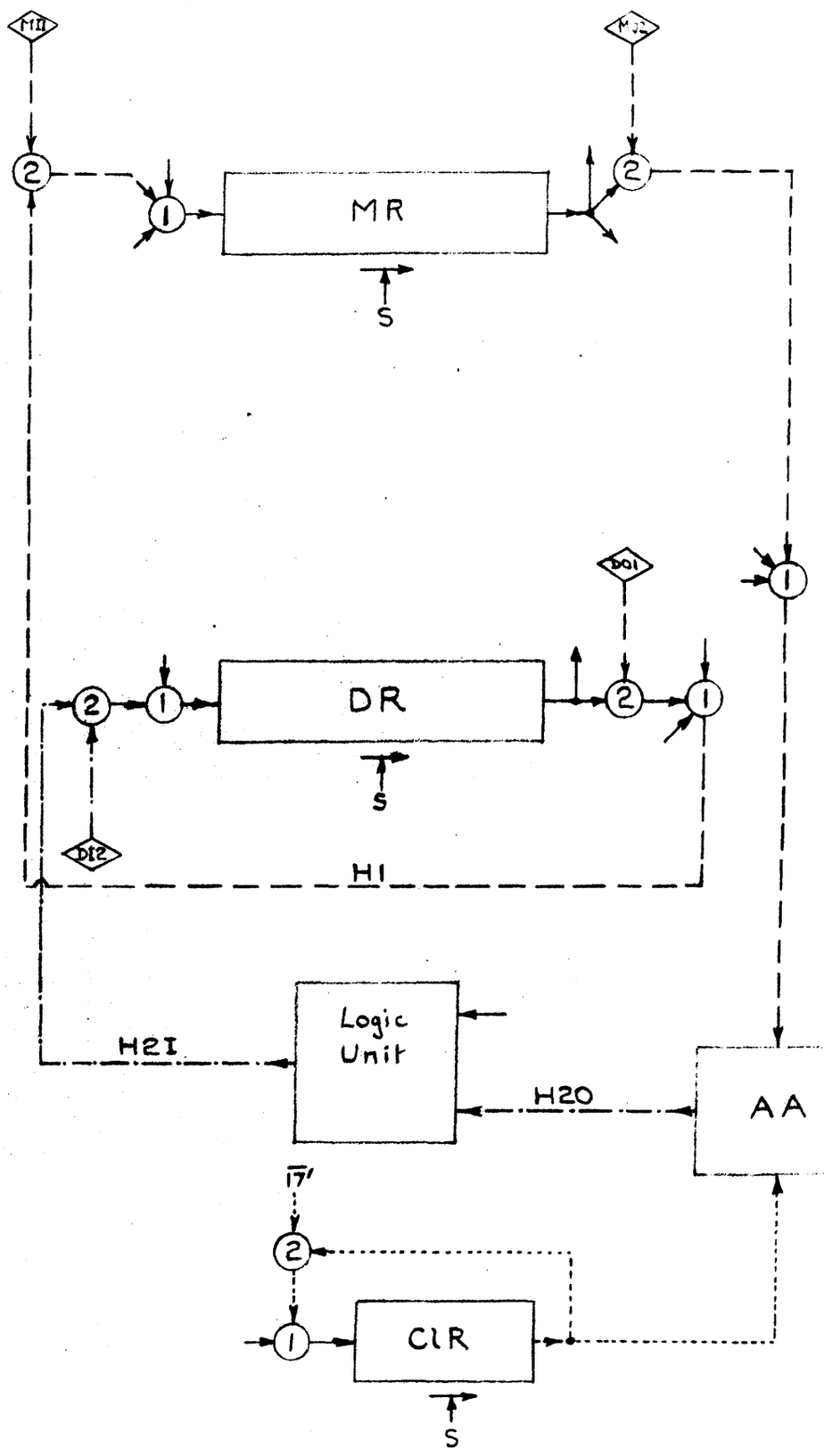
M- AND S-REGISTER – – – – –



(b) ORDERS IN WHICH STORE CONTENT IS UNCHANGED

Fig. 15. LOGIC DIAGRAMS FOR M-REGISTER INSTRUCTIONS





(a) (Above) STORE REGISTER ORDERS WITH INPUT..... OR OUTPUT------

(b) (Right) ORDERS NOT USING THE STORE

Fig. 16. LOGIC DIAGRAMS FOR BASIC D-REGISTER INSTRUCTIONS

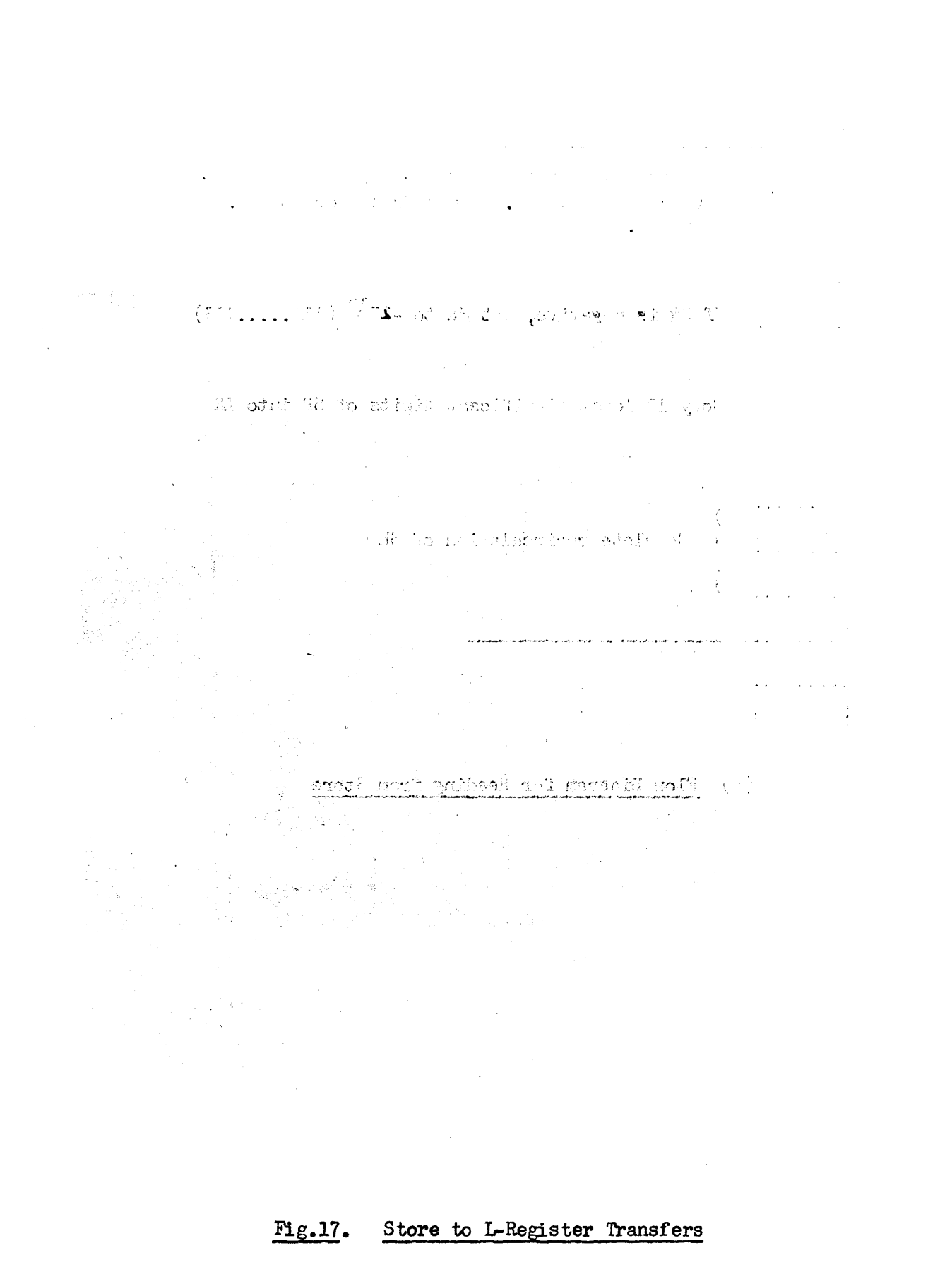
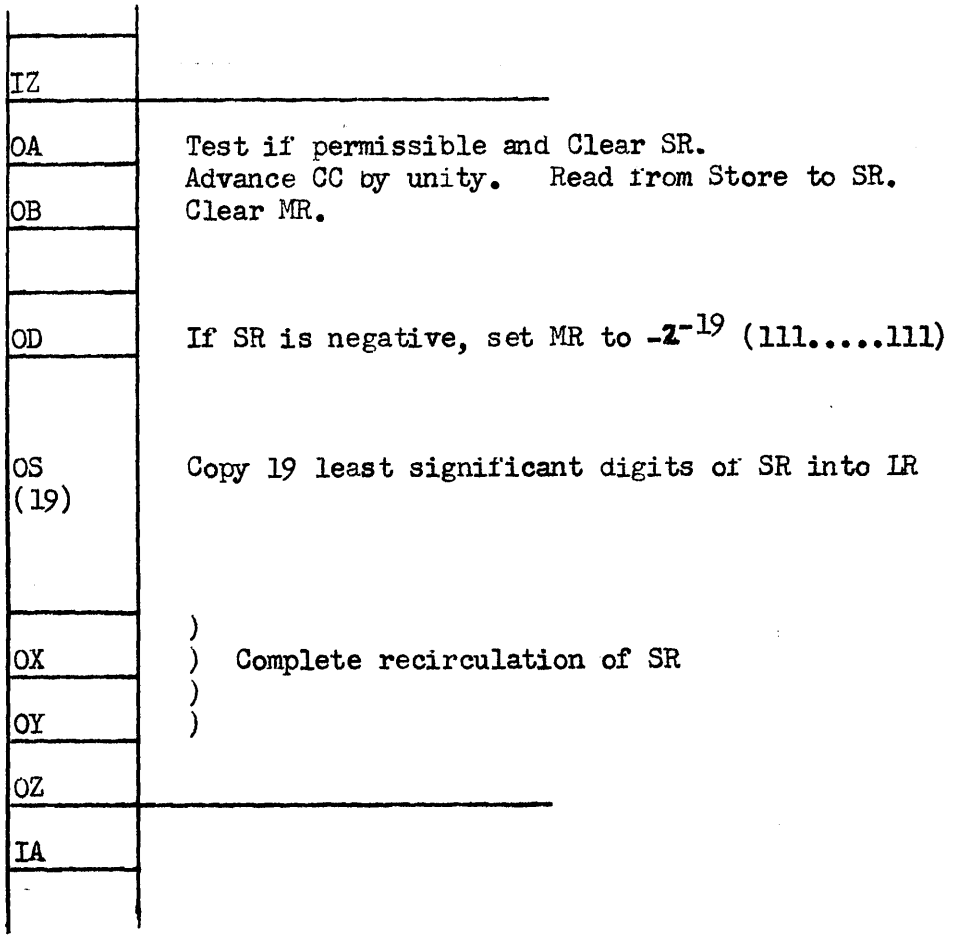
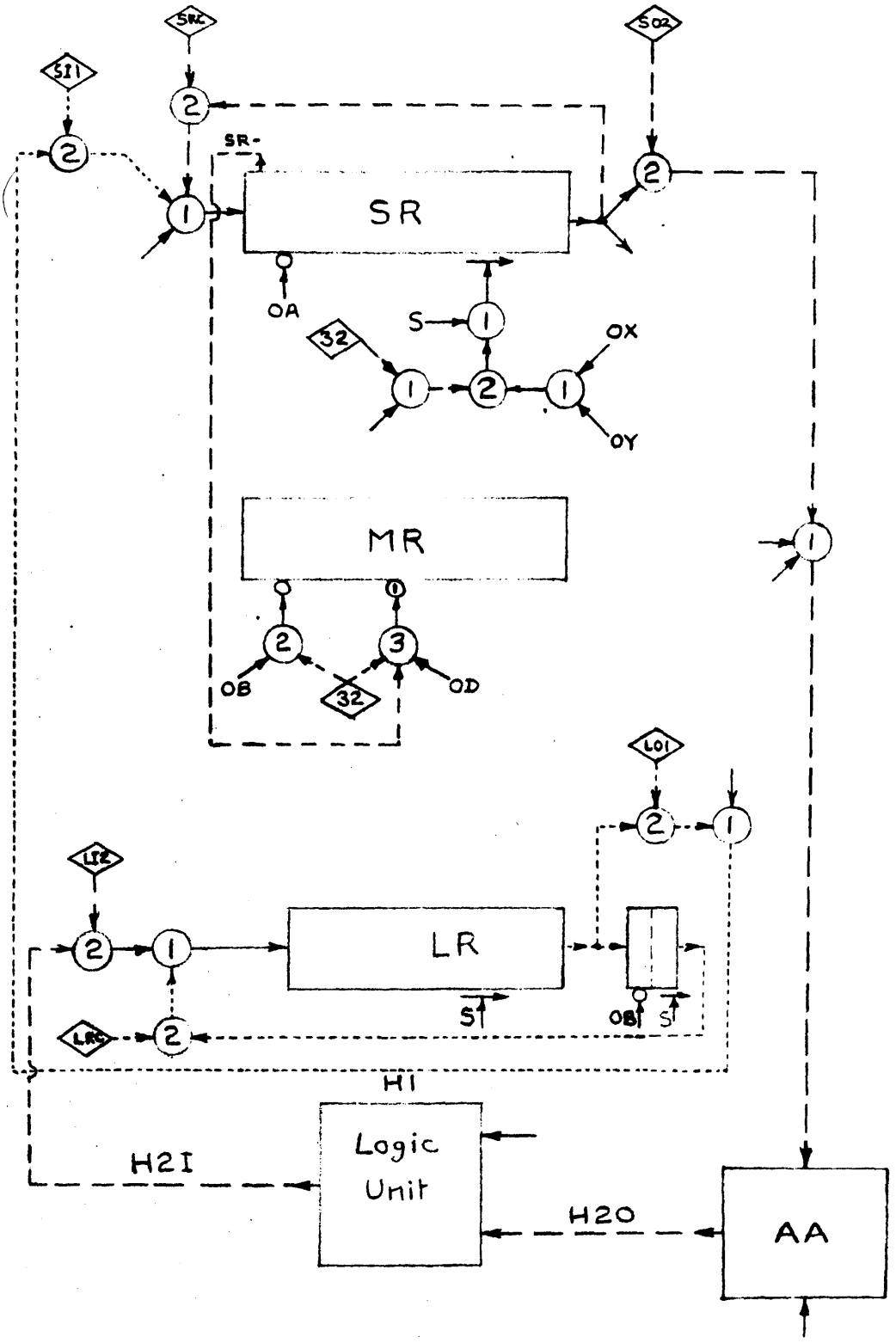


Fig.17. Store to L-Register Transfers



(a) Flow Diagram for Reading from Store



(b) LOGIC DIAGRAM : TO STORE .....
FROM STORE------

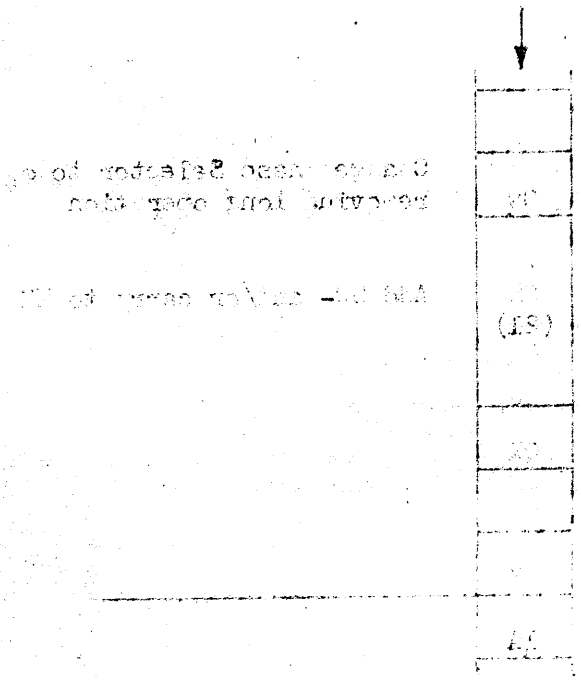
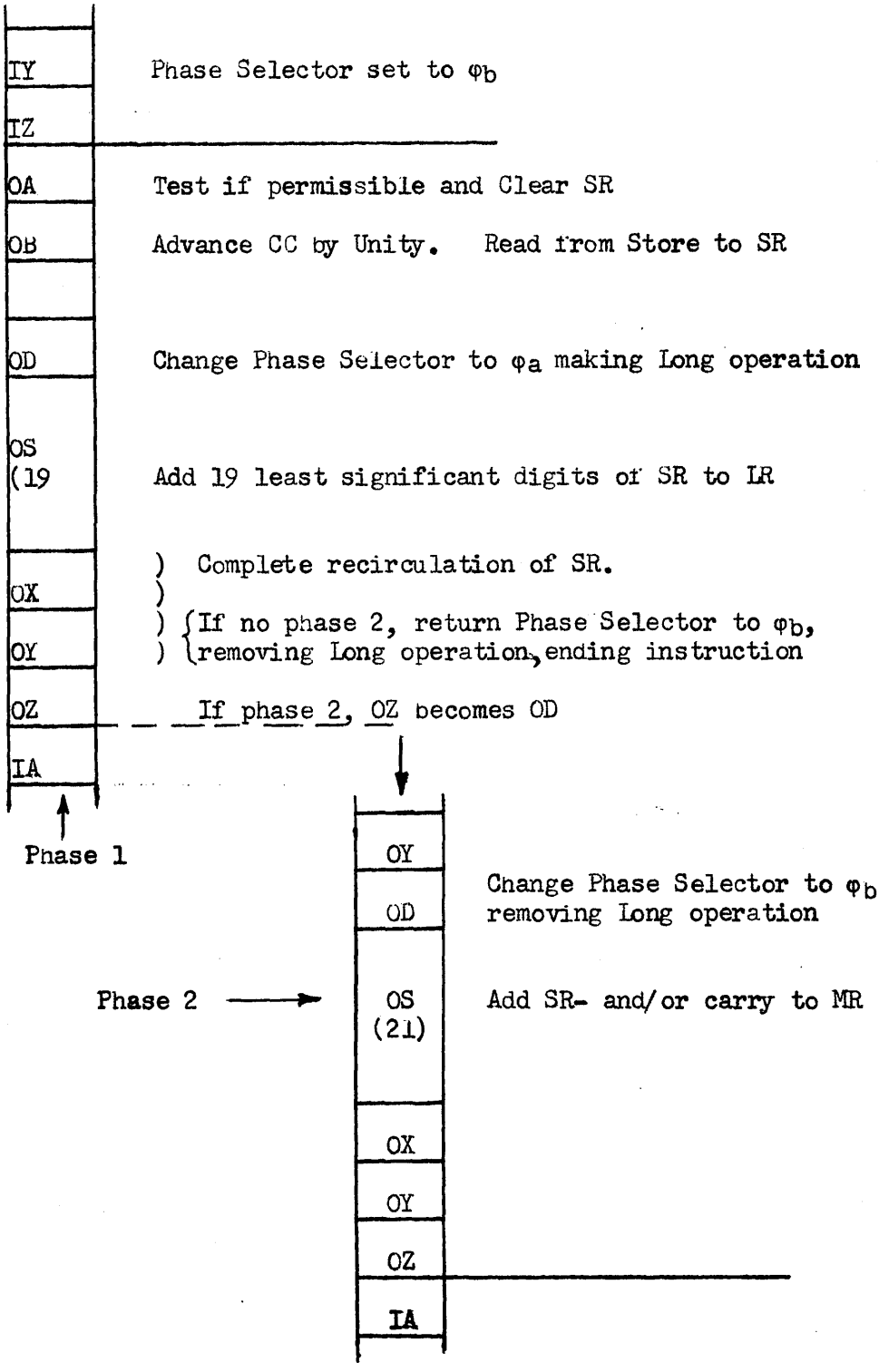
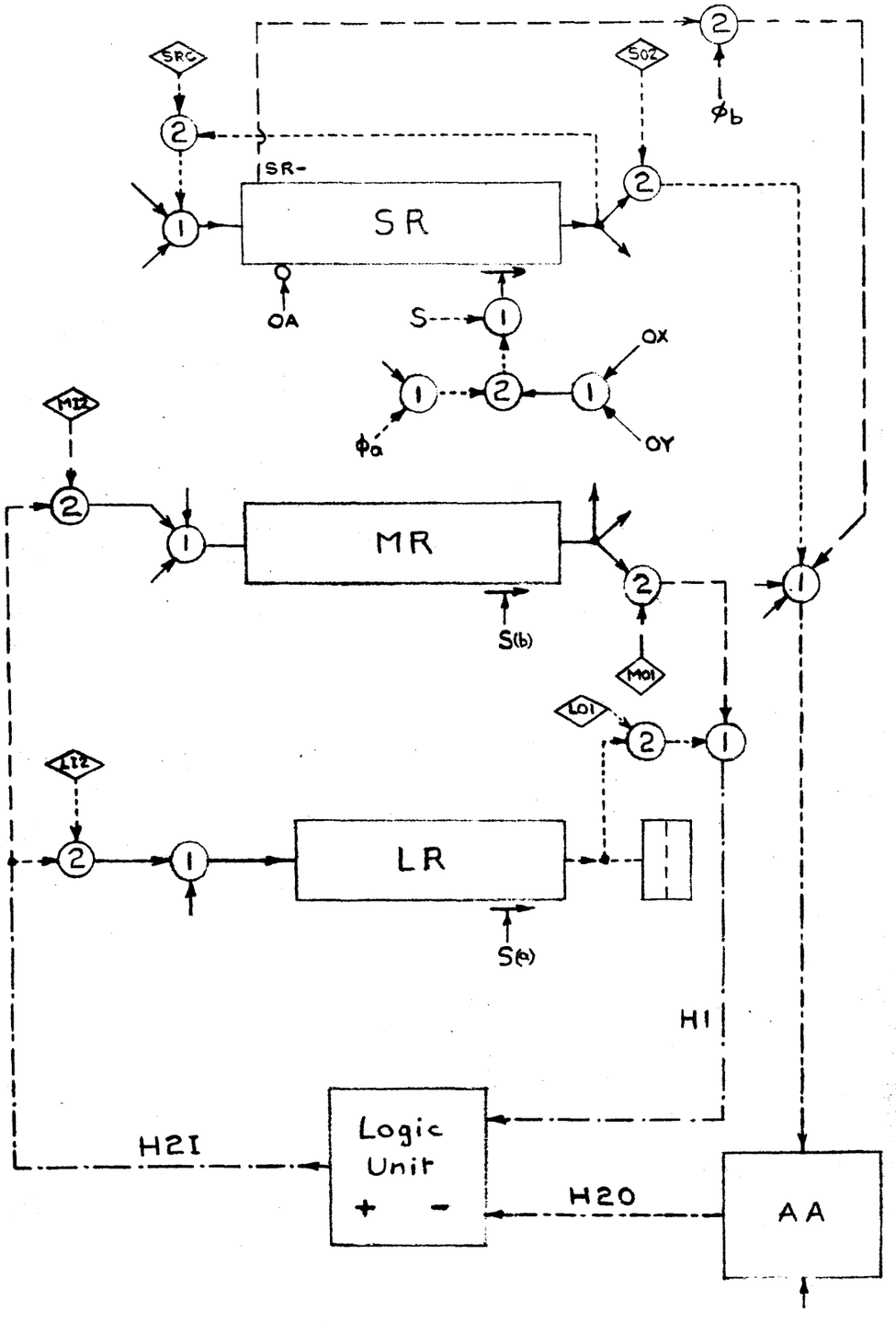
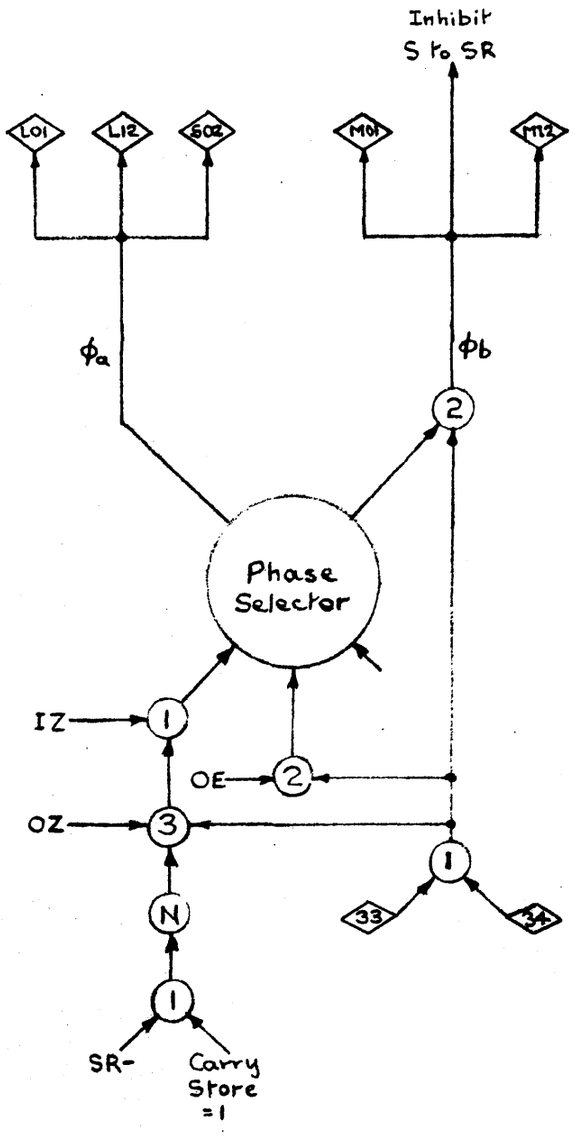


Fig. 18. L-Register Addition and Subtraction



(a) Flow Diagram for L-Register Addition





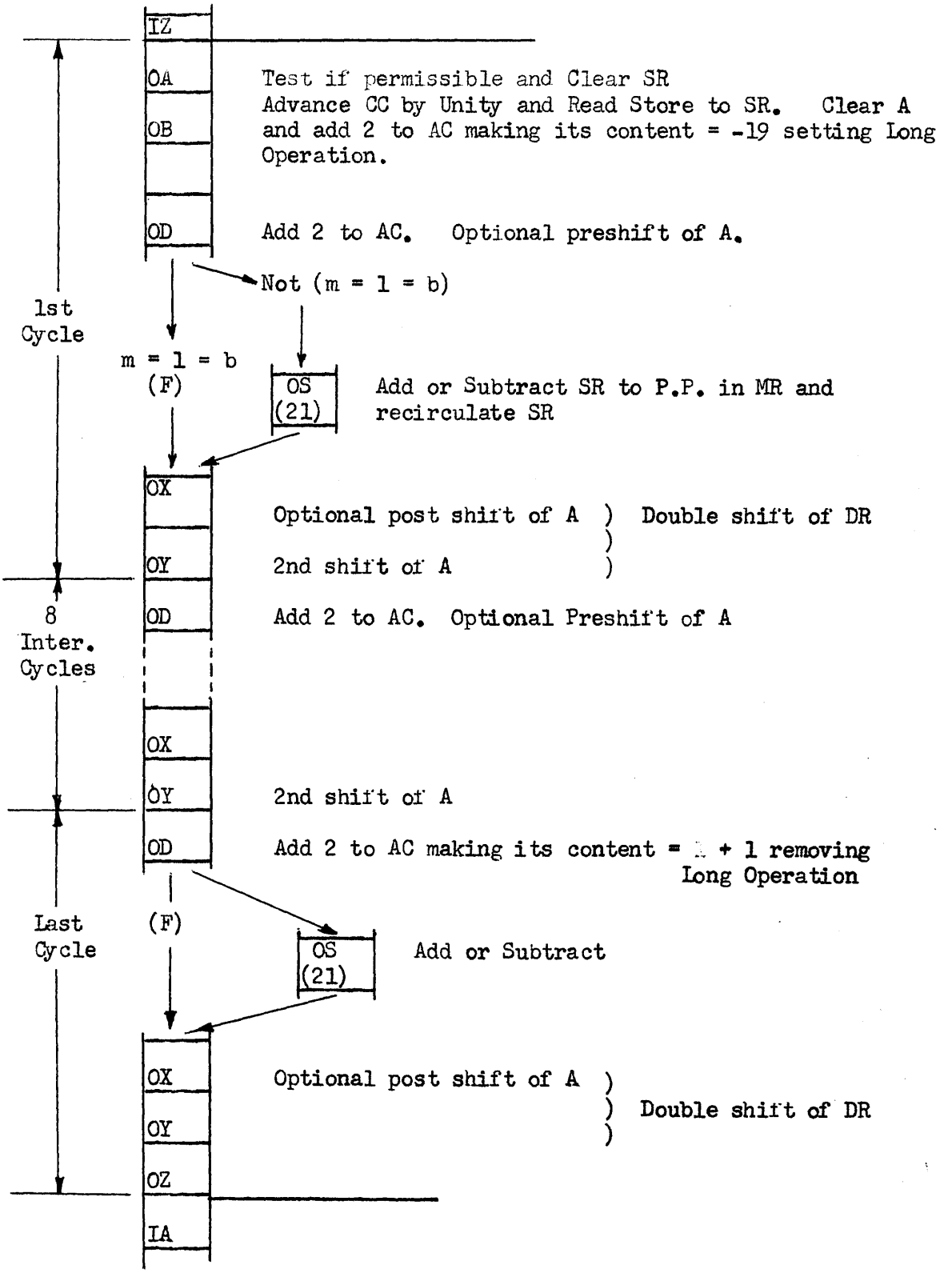
(b) LOGIC DIAGRAM

(Above) OPERATION : PHASE a .....

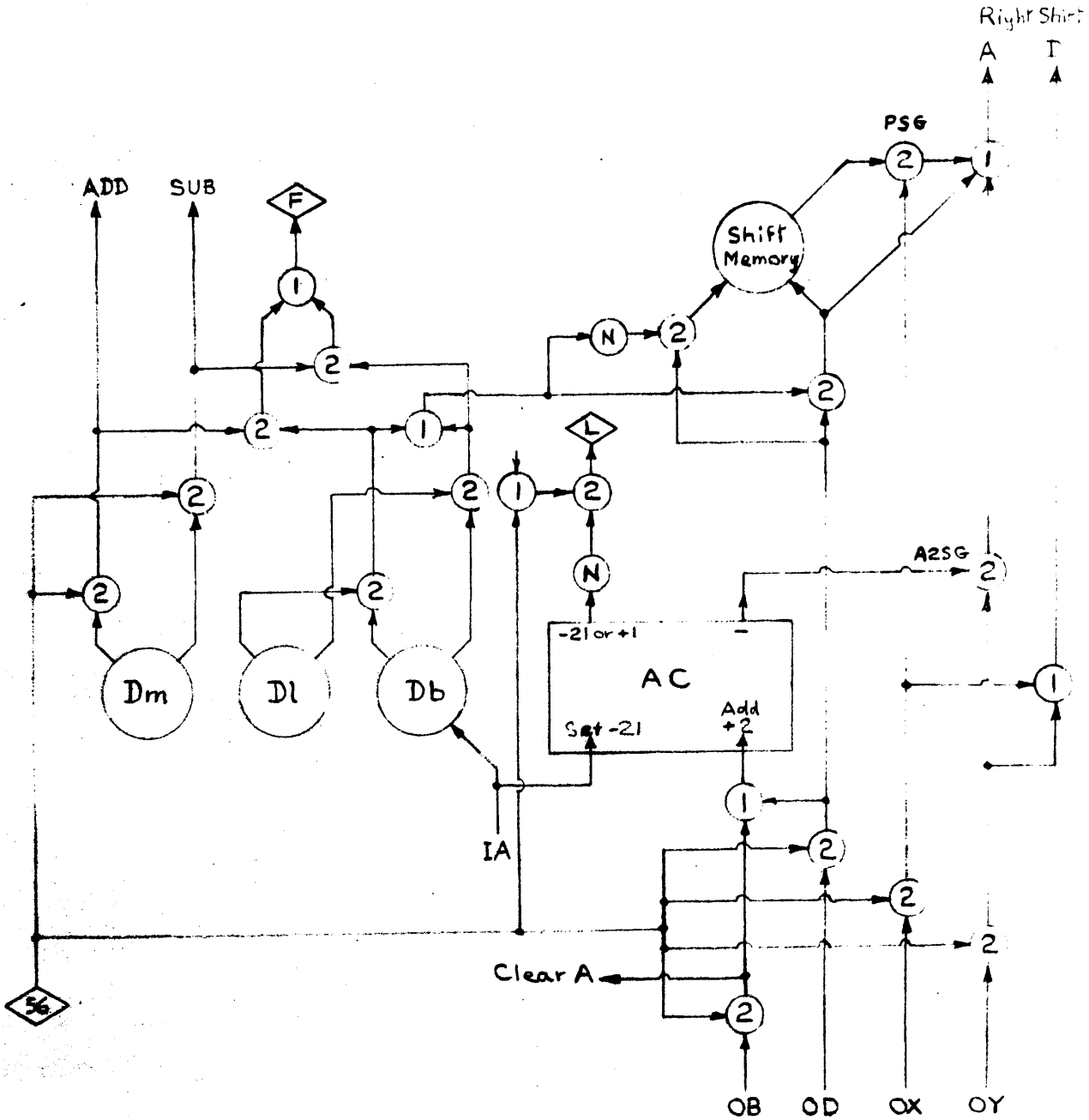
PHASE b – – – –

(Right) CONTROL ARRANGEMENTS

Fig. 19. Multiplication



(a) Flow Diagram.



(b) LOGIC DIAGRAM

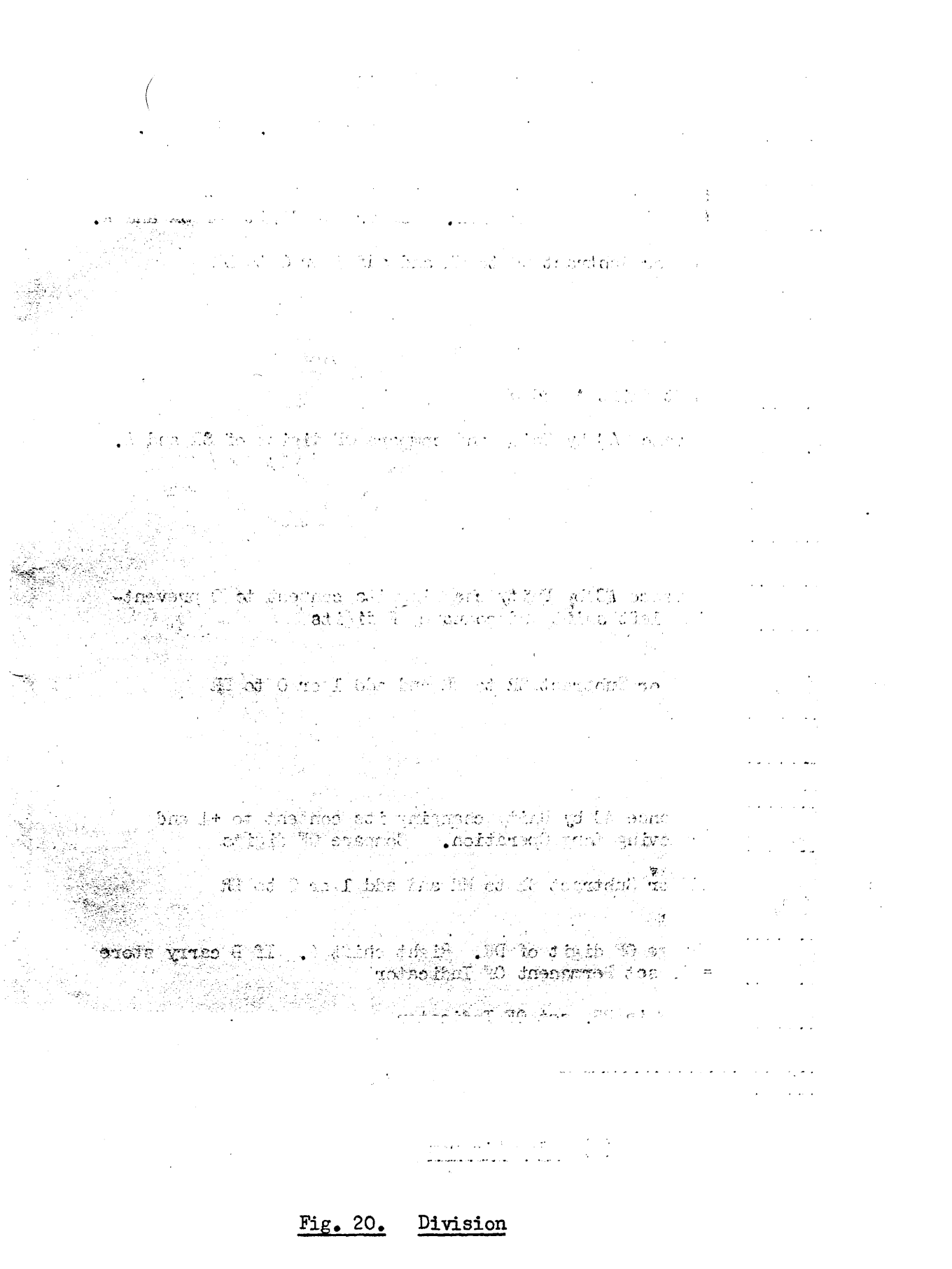
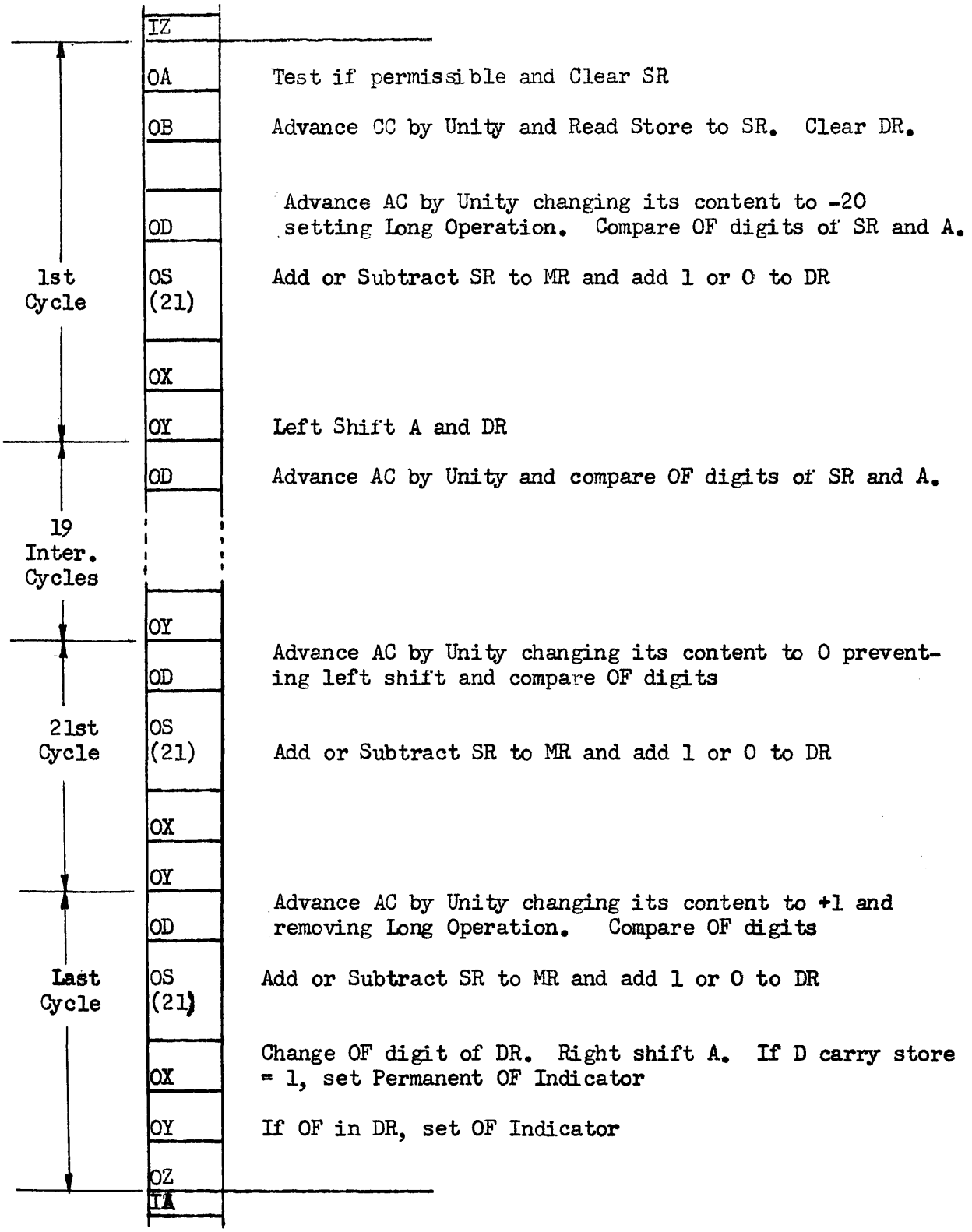
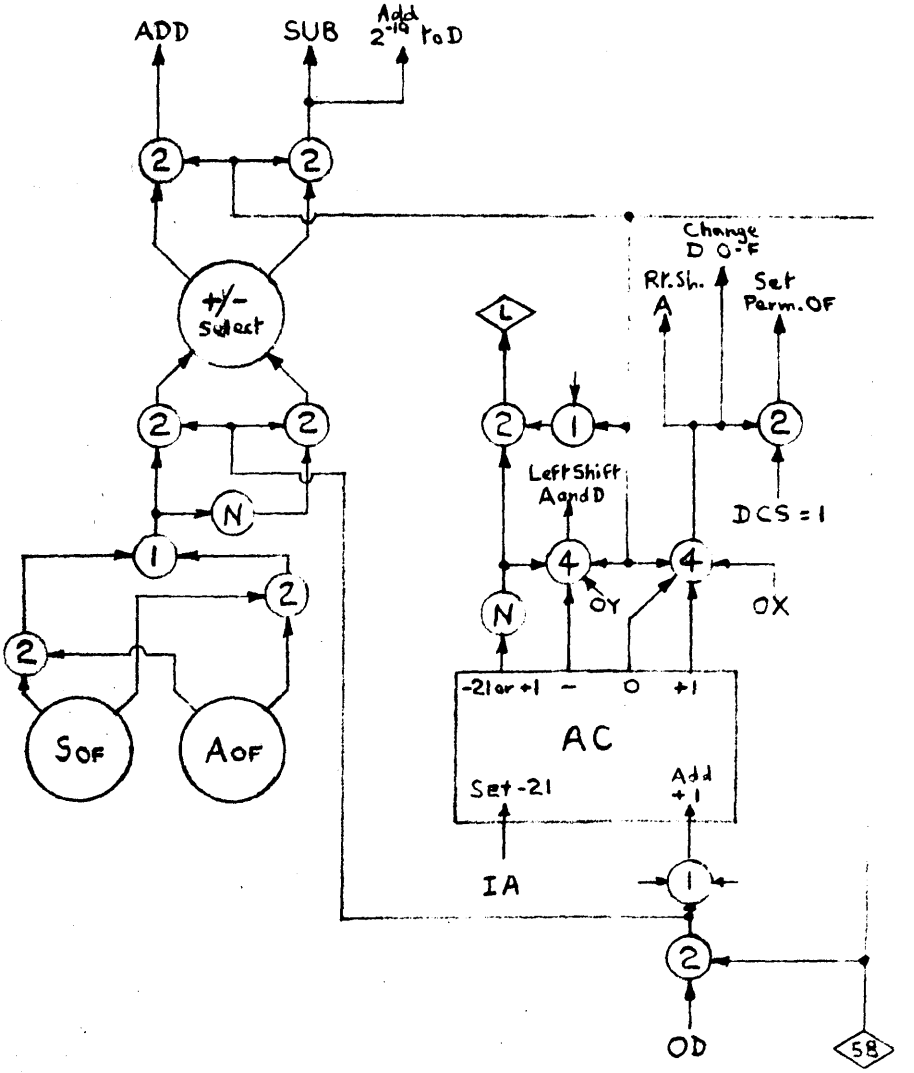


Fig. 20. Division

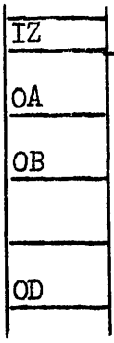


(A) Flow Diagram



(b) LOGIC DIAGRAM

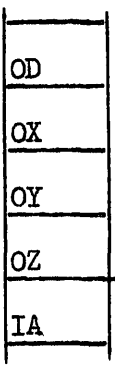
Fig. 21. Shift Operation



Test if permissible and Clear SR.
Advance CC by Unity. (If N+, shift A in opposite
(direction; If N-, digitally
(complement number N.

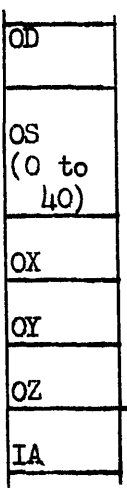
Copy the digital complement of 6 least significant digits of N into AC unless required shift > 38 when -48 is set into AC. If Logical Shift, make OF digit of A = 0.

(i) If Right Shift > 38, operation is Fast



Clear A. If Logical Shift make OF digit of A = sign digit of A,

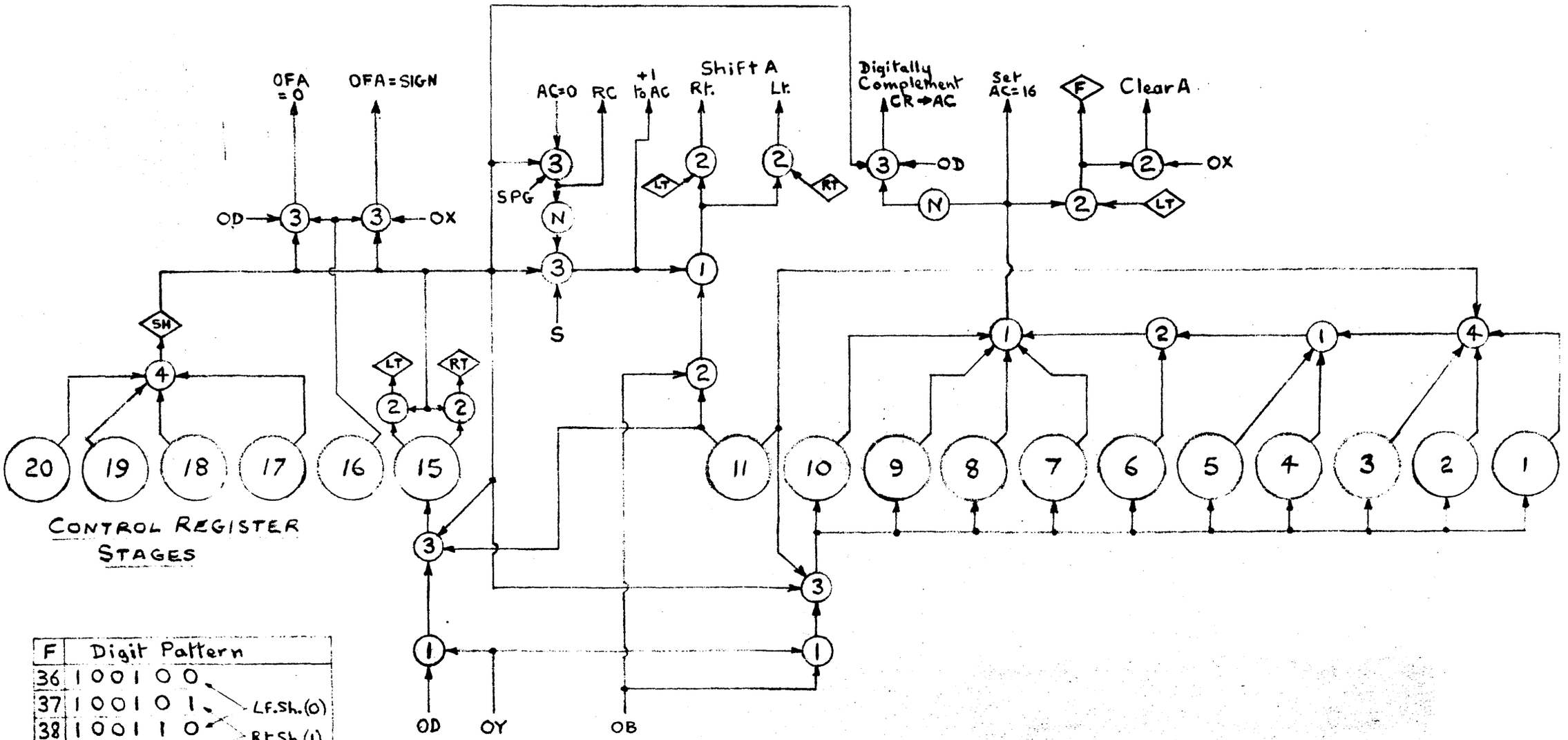
(ii) If not (i)



Shift A, adding Unity to AC for each shift until AC = 0.

If Logical Shift make OF digit of A = sign digit of A

(A) Flow Diagram



CONTROL REGISTER STAGES

F	Digit Pattern	
36	1 0 0 1 0 0	Lf.Sh.(0)
37	1 0 0 1 0 1	
38	1 0 0 1 1 0	Rt.Sh.(1)
39	1 0 0 1 1 1	

Shifts

Log.(1)

(b) LOGIC DIAGRAM

IZ	
OA	Test if permissible and Clear carry stores
OB	Advance CC by Unity and Clear NR
OD	Copy the digital complement of the 6 least significant digits of CR into AC. Right shift A
OS (0 to 40)	Left shift A, adding Unity to AC for each shift, until normalised
OX	
OY	If A is not normalised (A = 0), operation is complete. If A is normalised, copy AC into NR and change Phase Selector to $\varphi 2$ connecting CR shift pulse gate and change over to Long Operation
OZ	

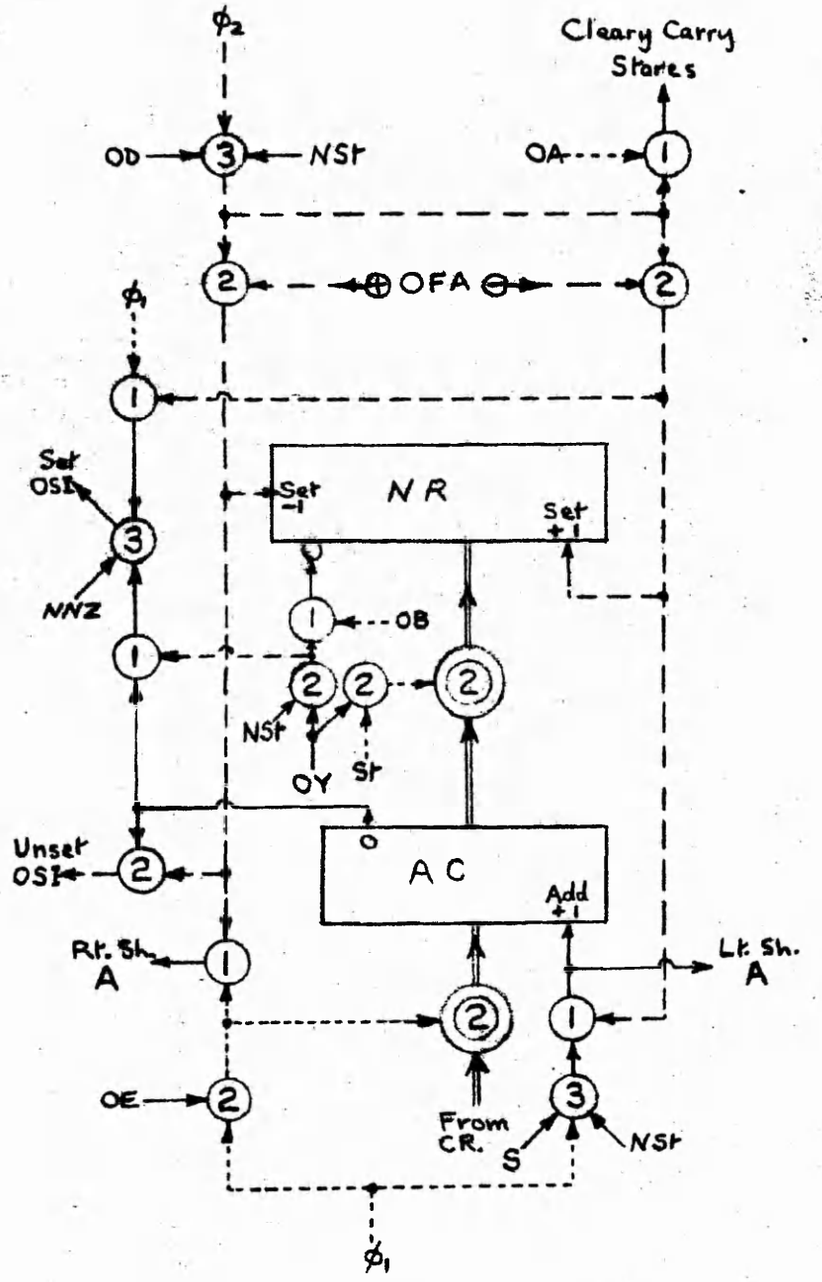
(a) Phase 1. Flow Diagram

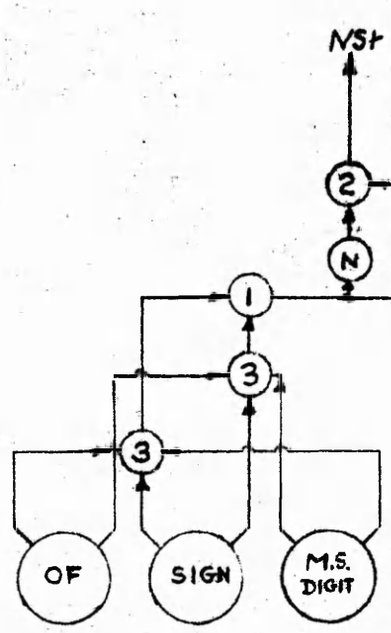
OD	Copy most significant digit of LR to MHA carry store
OS (21)	MHA Carry added to MR. Sum of NR and CR1 subtracted from DR with CR shift pulses stopped at 11 by HT.
OX	
OY	If A still normalised (A≠1 or $-\frac{1}{2}$), operation is complete. If A not normalised, clear NR and continue in phase 3.
OZ	

(b) Phase 2. Flow Diagram

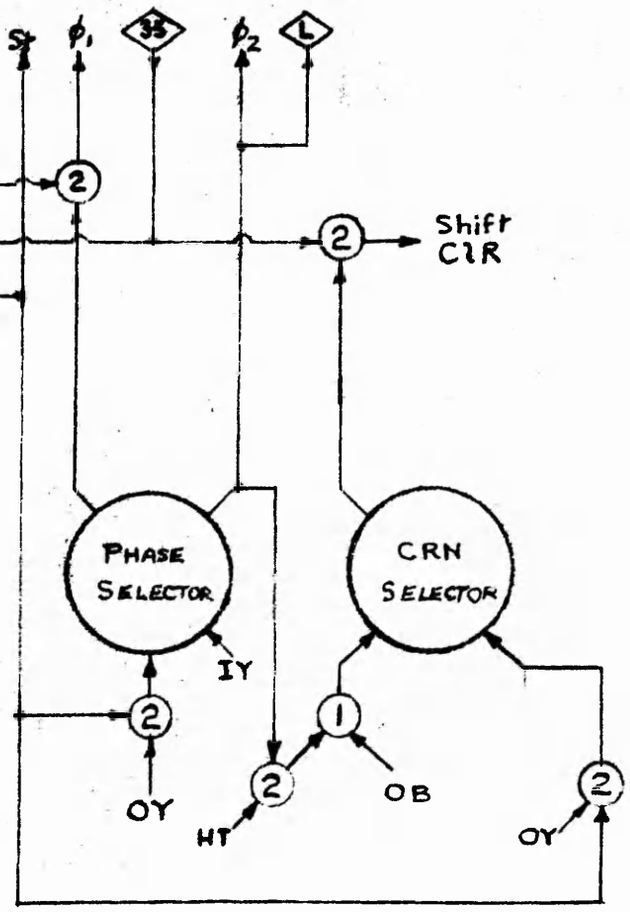
OD	Clear carry stores. If OFM = 0, Right Shift A and set NR=-1 If OFM = 1, Left Shift A and set NR=+1
OS (21)	Subtract NR from DR and recirculate MR
OX	
OY	Change phase selector to φ_1 removing Long operation
OZ	
IA	

(c) Phase 3. Flow Diagram.





MOST SIGNIFICANT STAGES OF M R



(d) LOGIC DIAGRAM

(Above) Phase 1 ---------

Phase 2 and 3 — — — —

(Right) Control

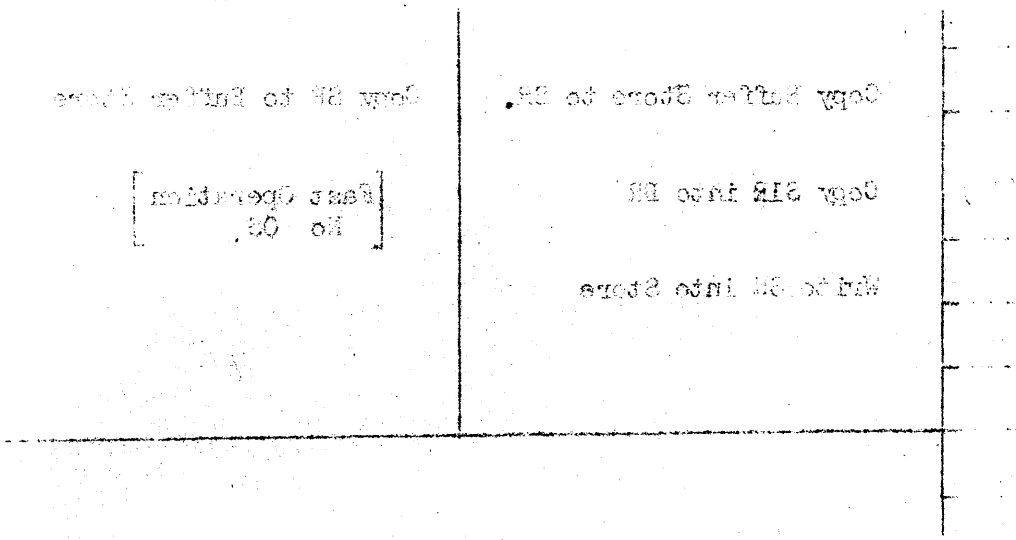
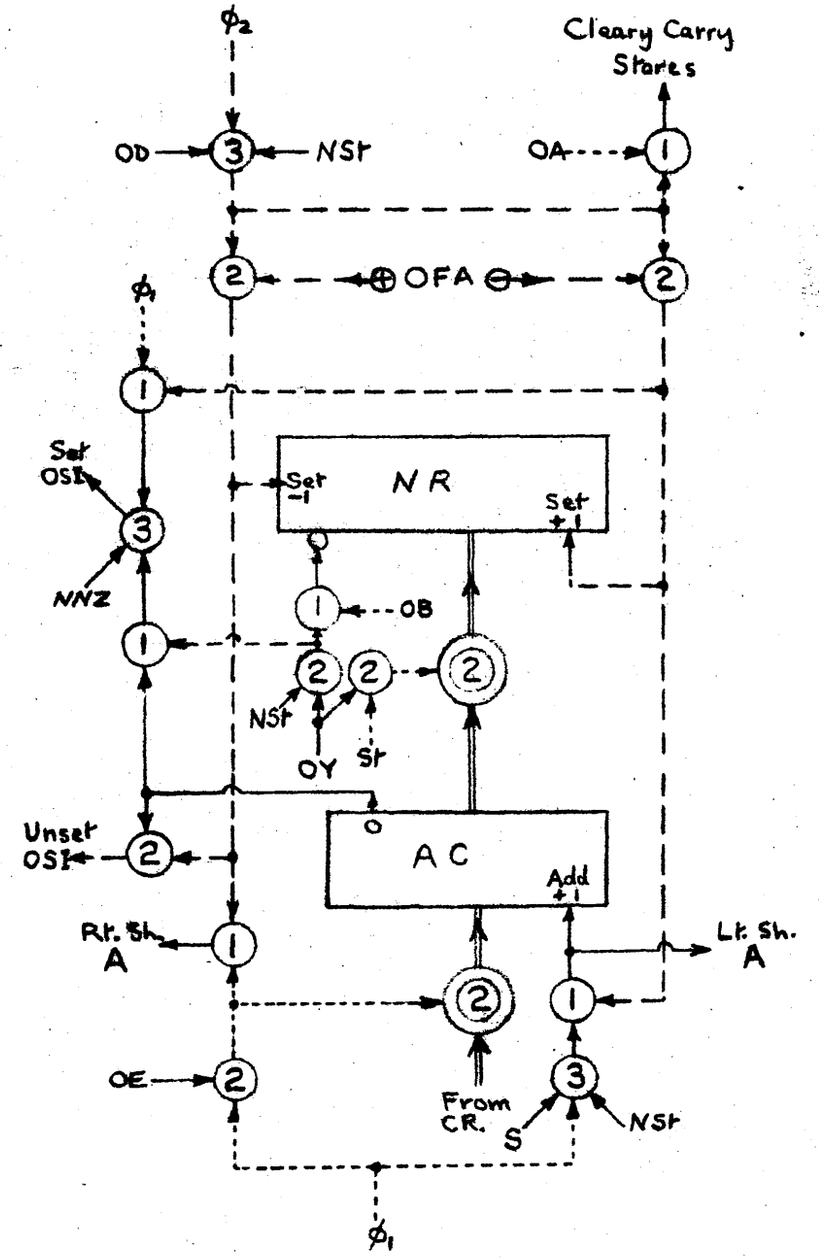
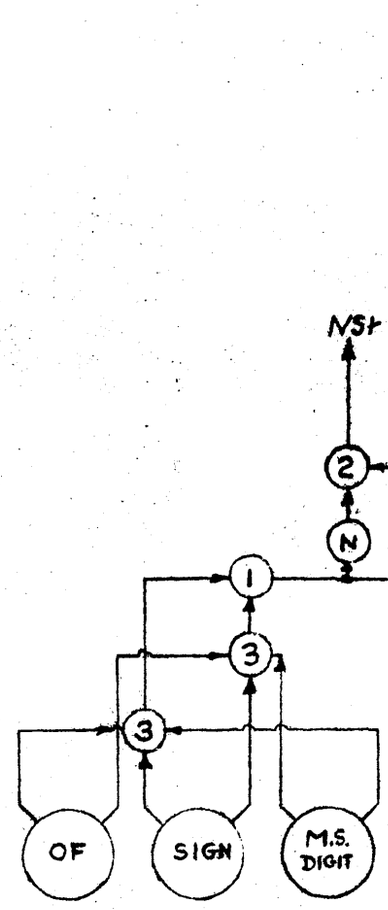
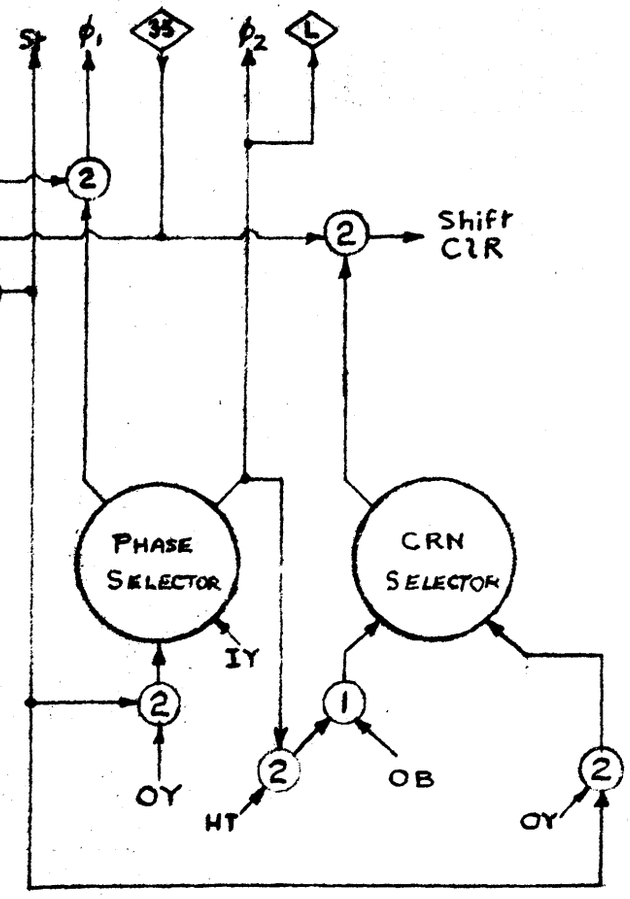


Fig.23. Flow Diagram for Tape Input





MOST SIGNIFICANT STAGES OF M R



(d) LOGIC DIAGRAM

(Above) Phase 1 ----------

Phase 2 and 3 — — — —

(Right) Control

Fig.23. Flow Diagram for Tape Input

	Input [10]	Output [20]
IZ		
OA	Test if permissible and clear SR	
OB	Advance CC by Unity and Test if Unit is Ready.	Read from Store to SR
OD	Copy Buffer Store to SR.	Copy SR to Buffer Store
OS (11)	Copy SIR into BR	[Fast Operation No OS]
OX	Write SR into Store	
OY		
OZ		
IA		

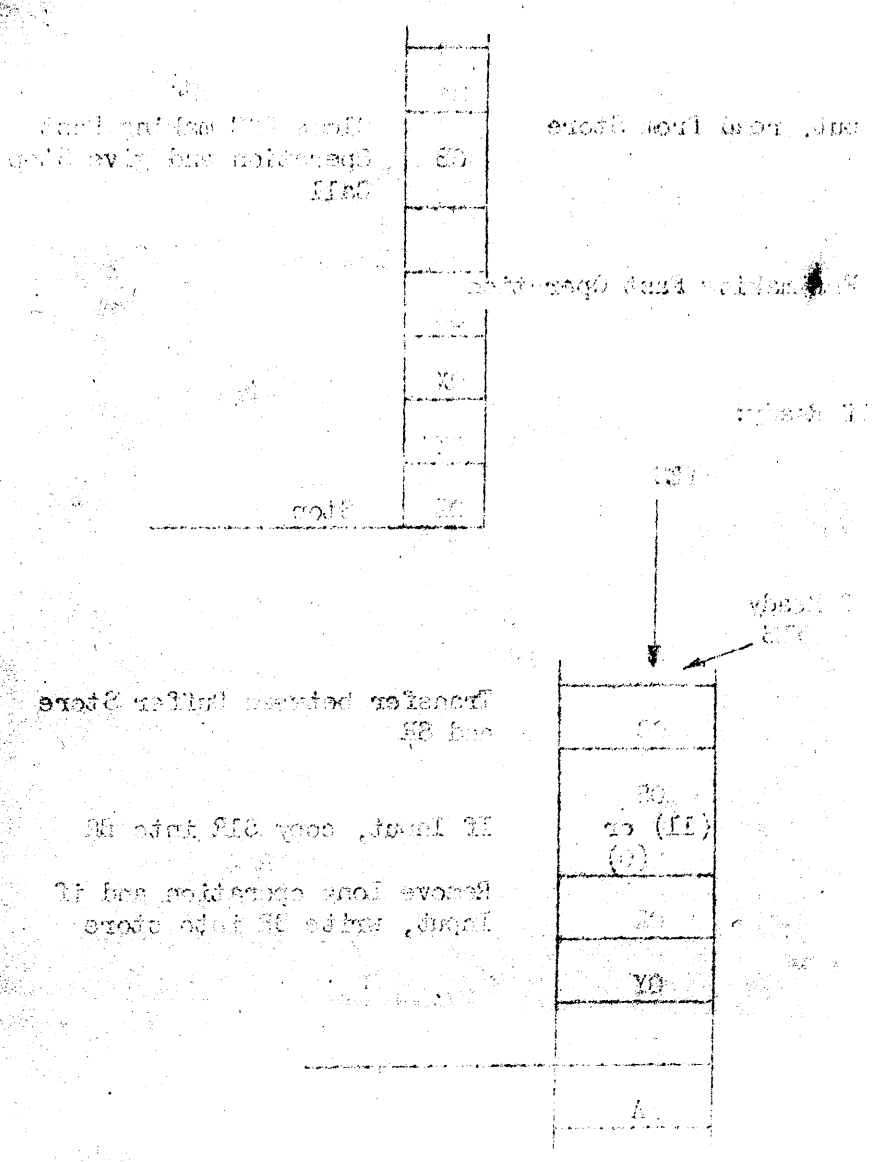
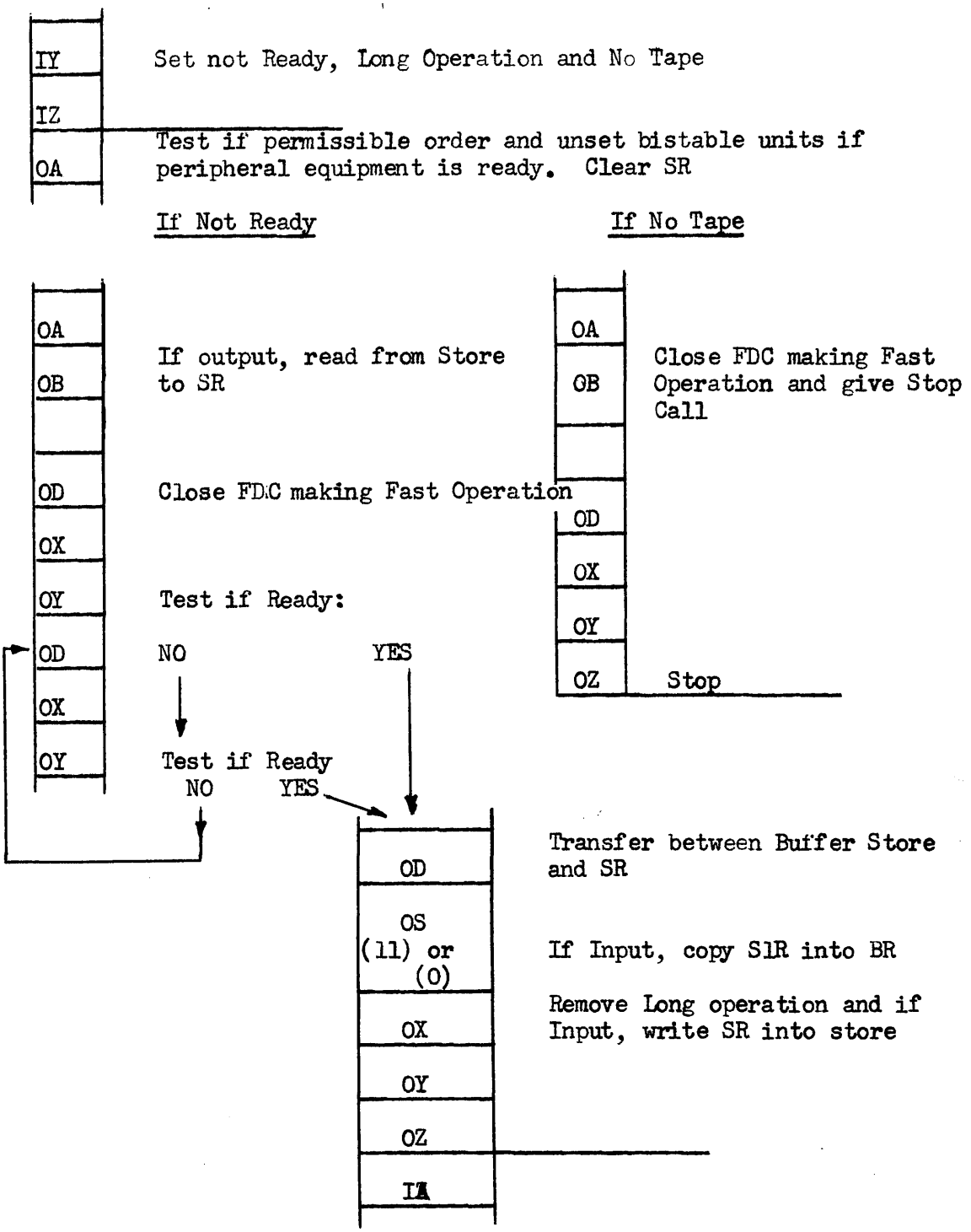
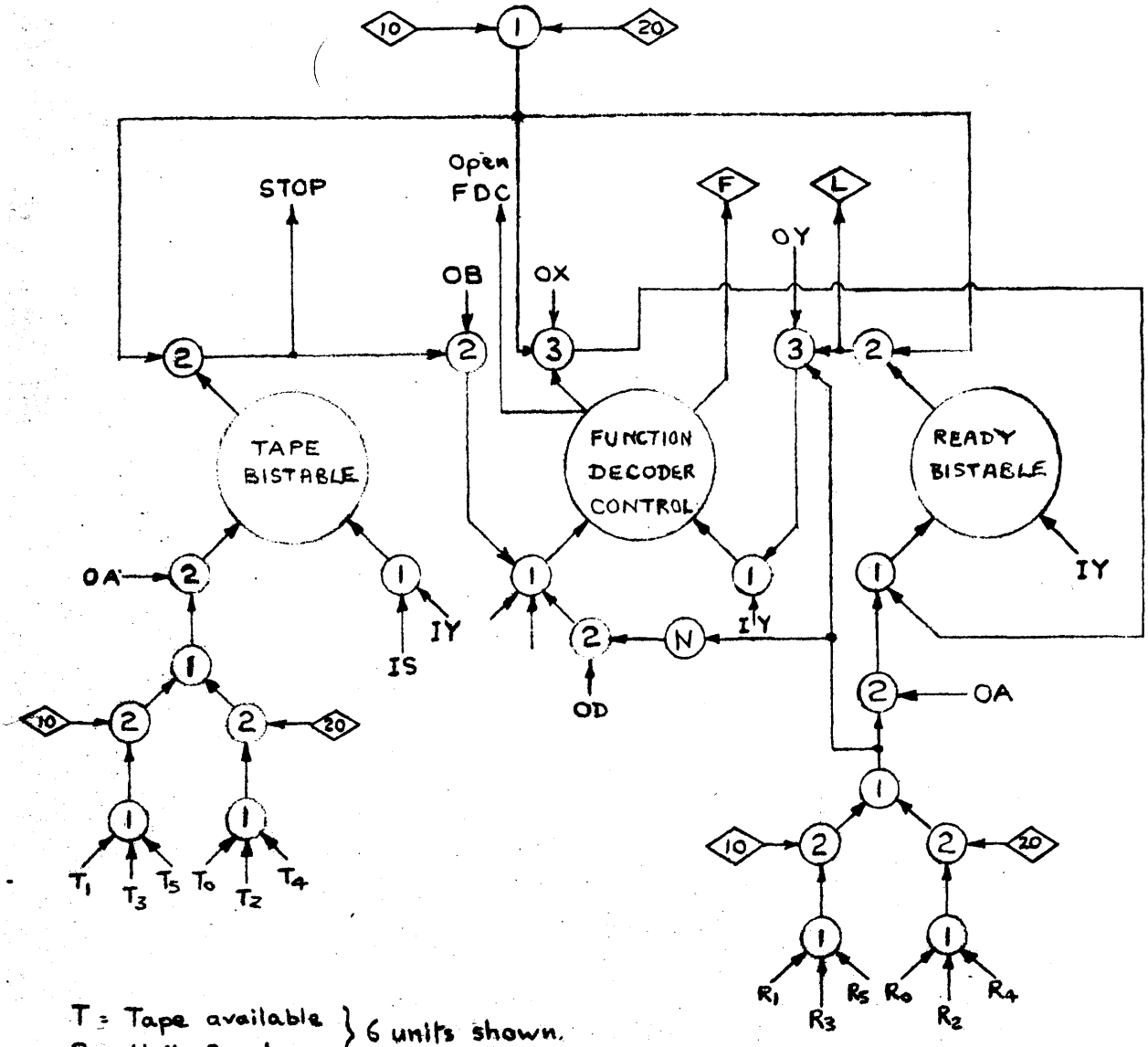


Fig. 24. Testing of Peripheral Equipment



(a) Flow Diagram



T = Tape available
R = Unit Ready
} 6 units shown.

(b) LOGIC DIAGRAM

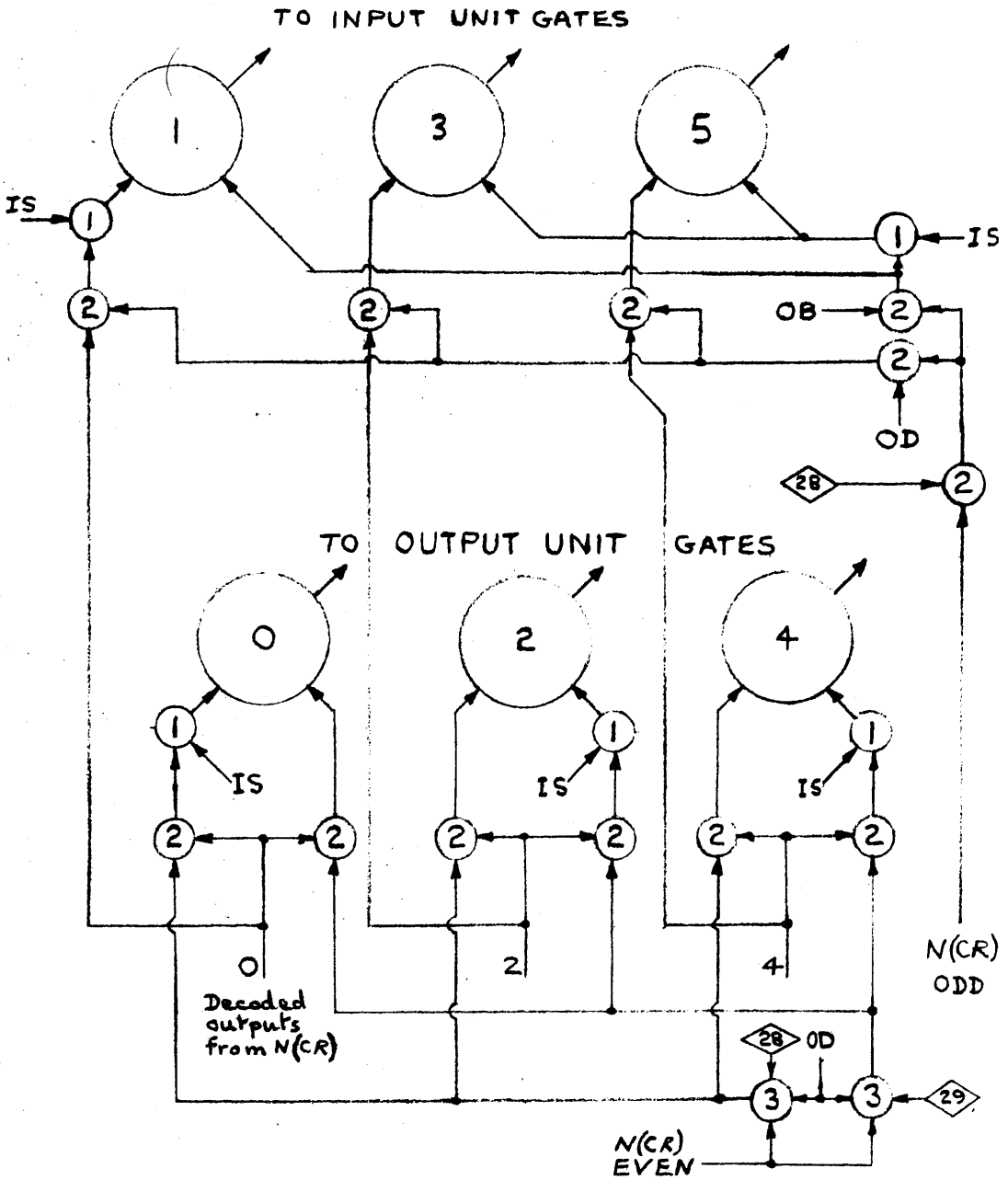


Fig. 25. INPUT - OUTPUT SELECTOR – LOGIC DIAGRAM

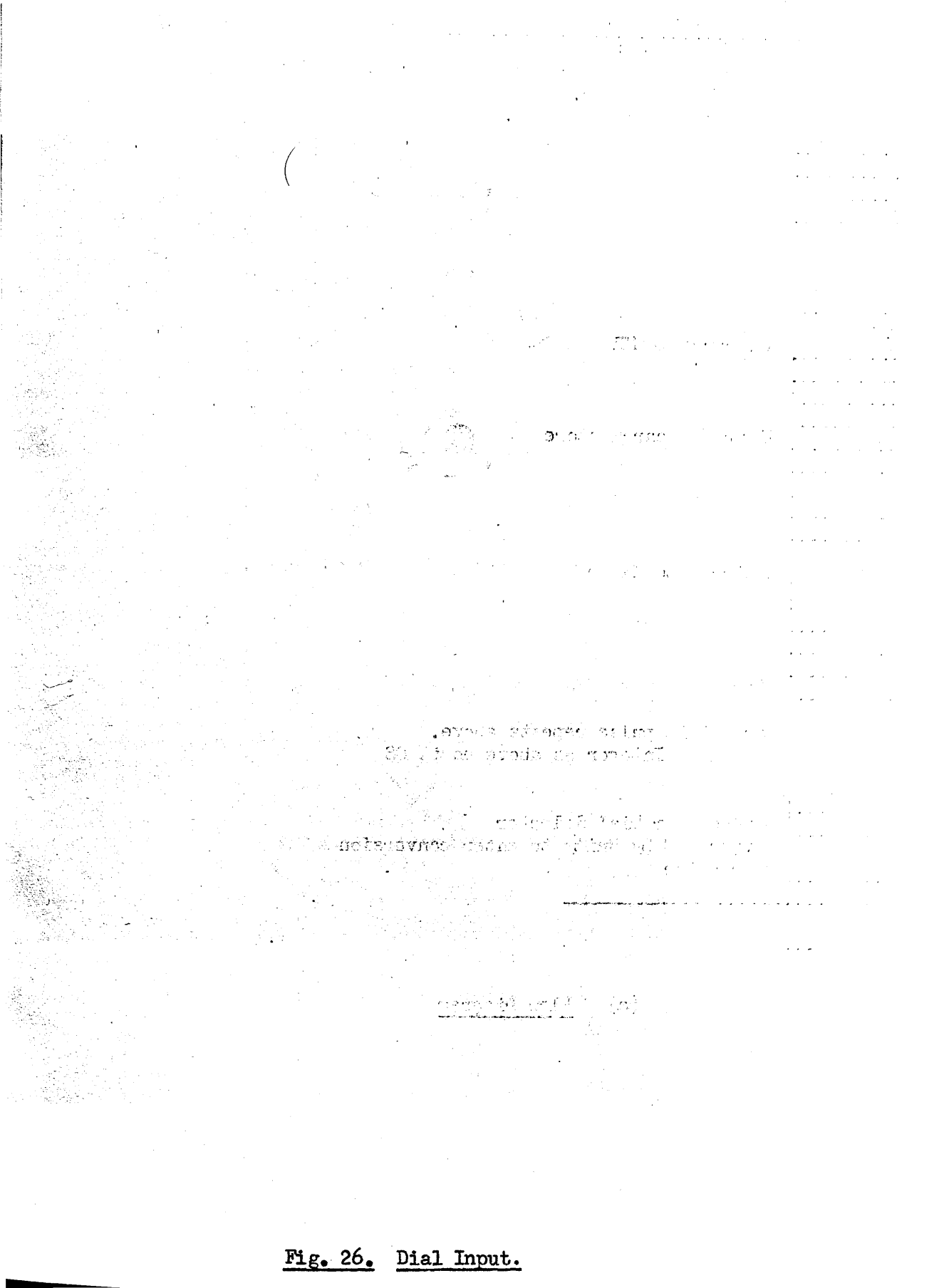
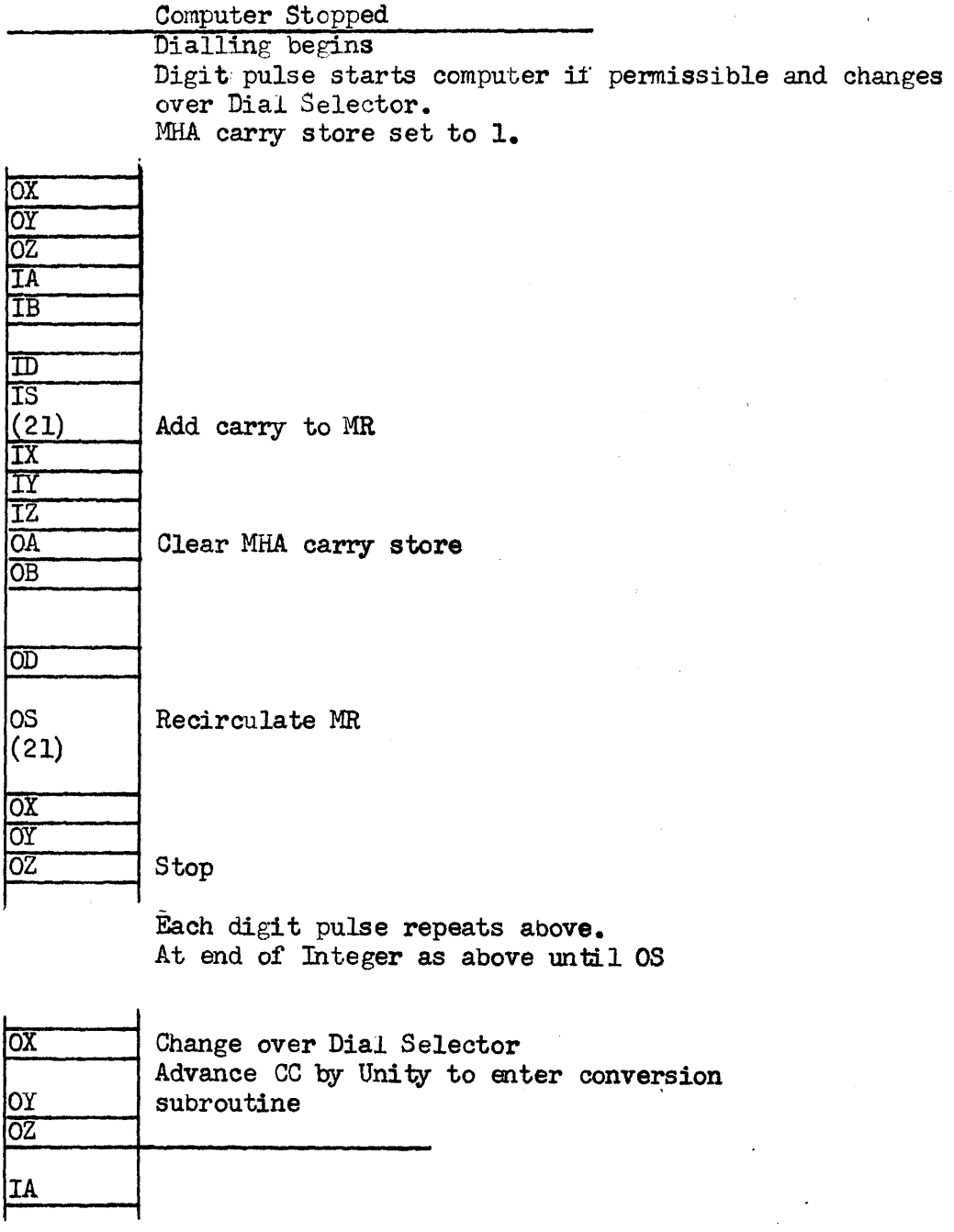
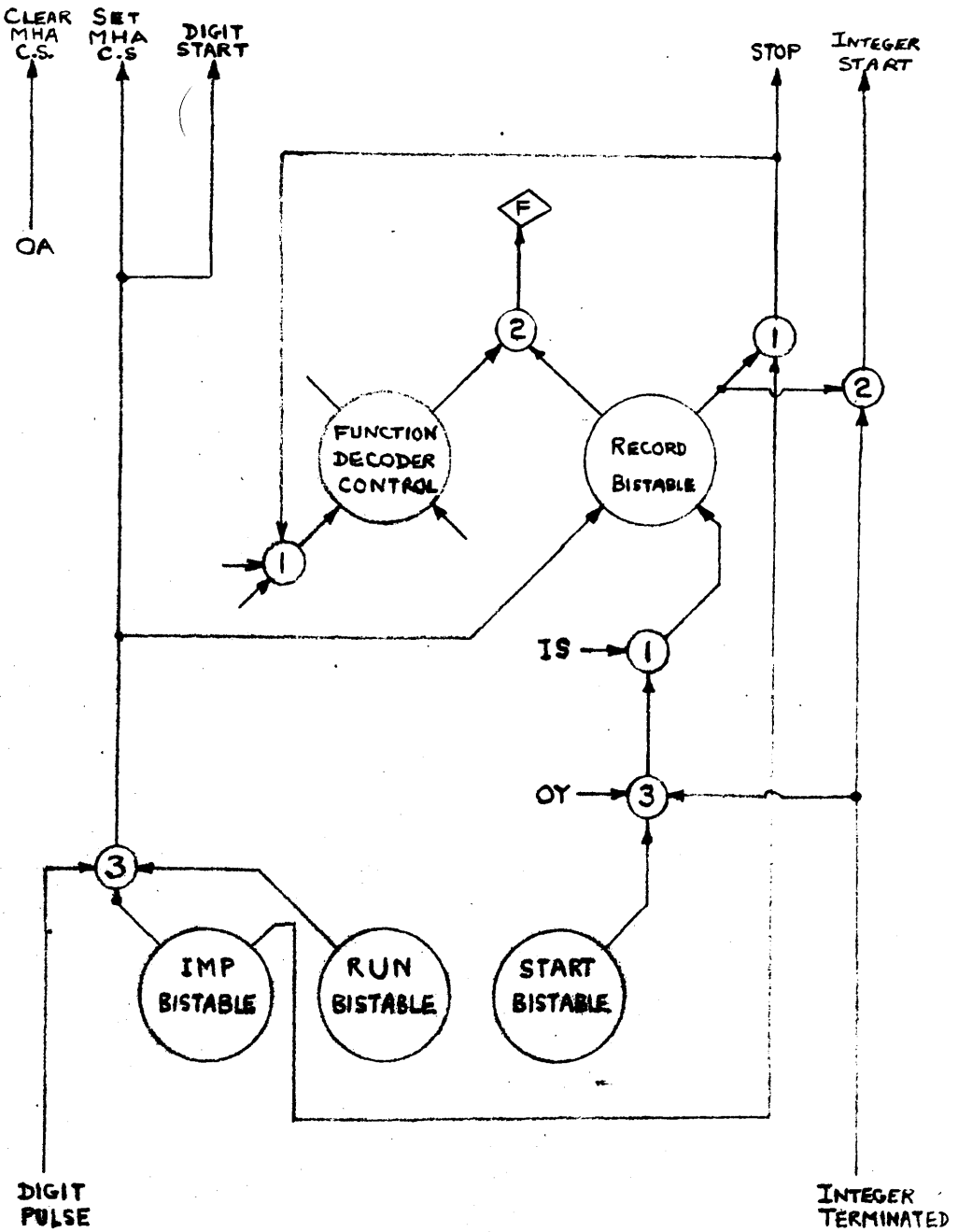


Fig. 26. Dial Input.



(a) **Flow Diagram**



(b) Logic Diagram

	N-Mod [16]	S-Mod [17]
IZ		
OA	Test if permissible and clear SR	
OB	Advance CC by Unity	
		Read from Store to SR
OD		
OX		Clear CR
OY		Copy SR into CR
OZ		
IA		

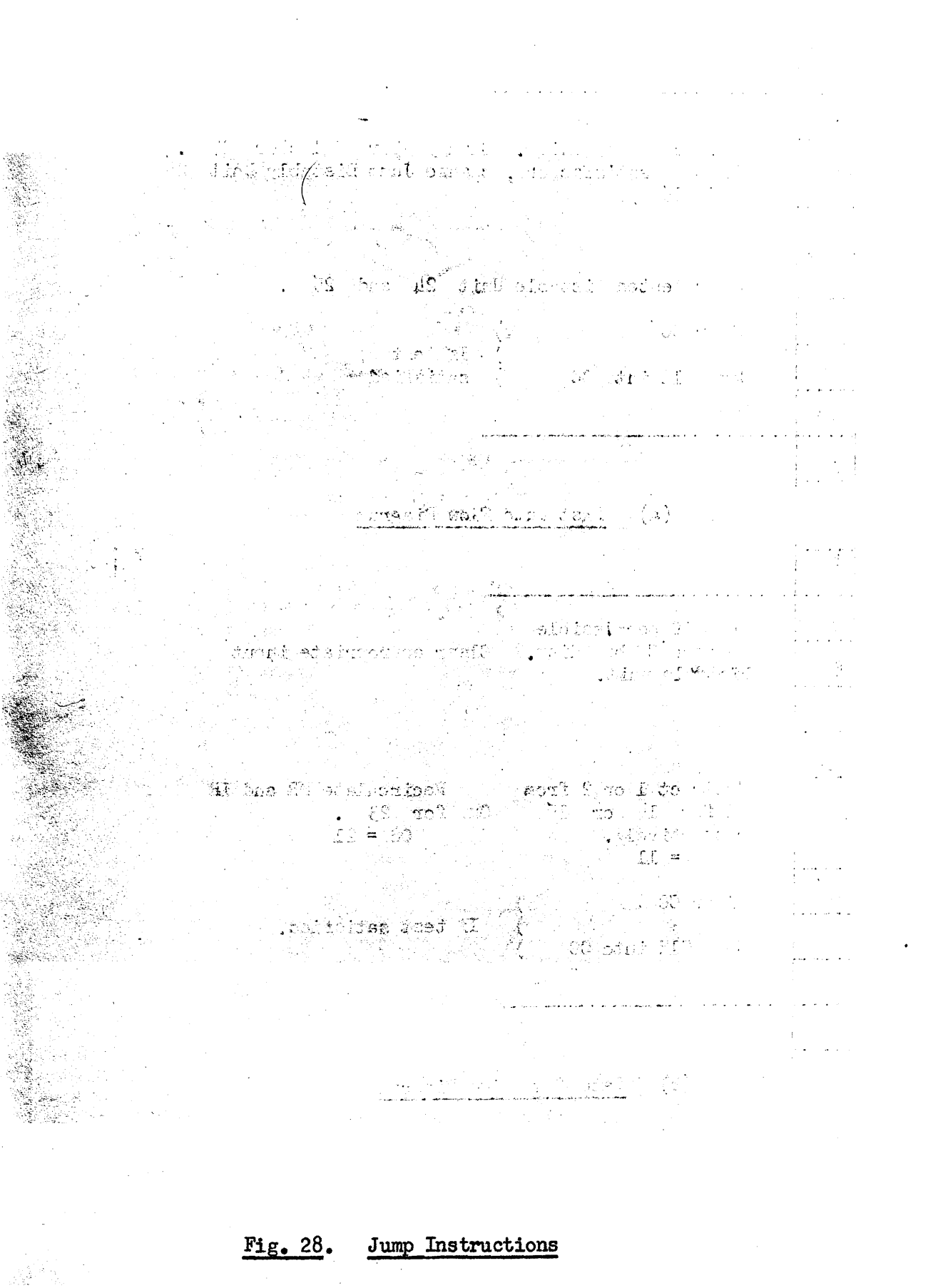
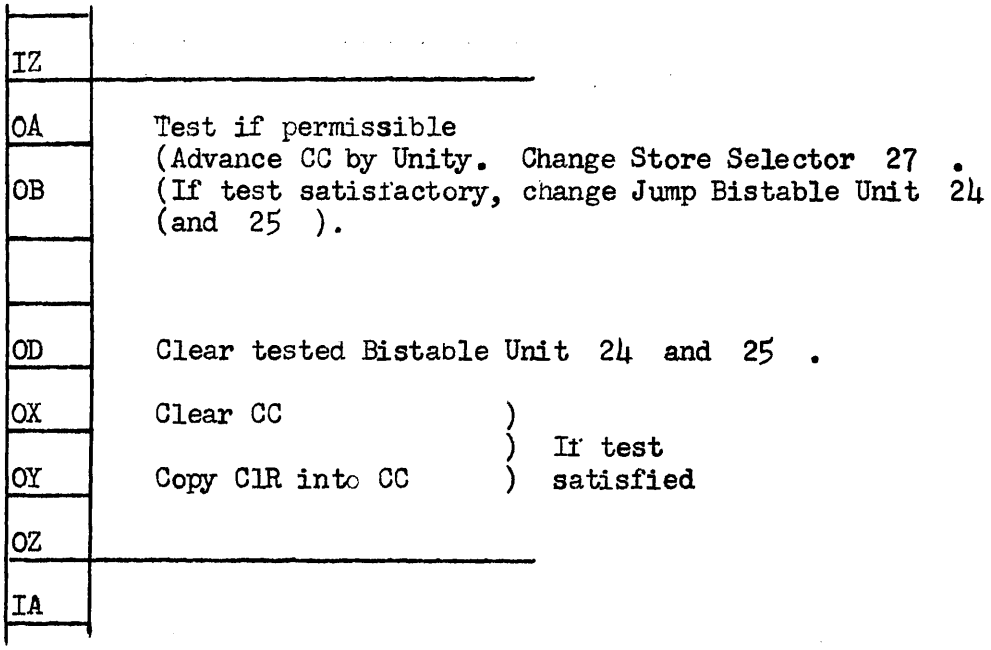
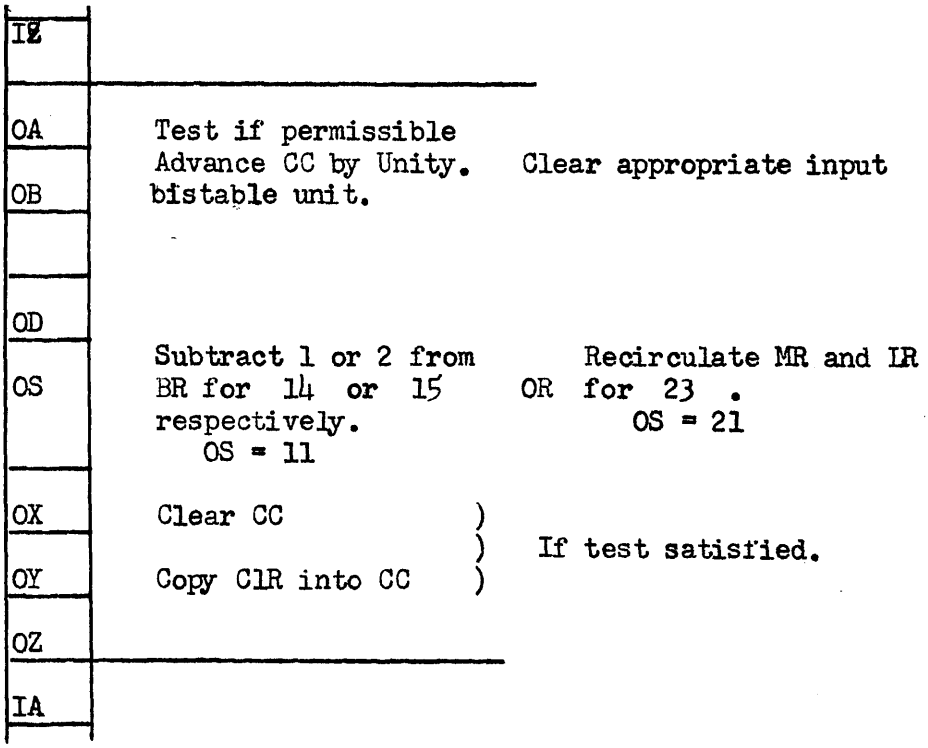


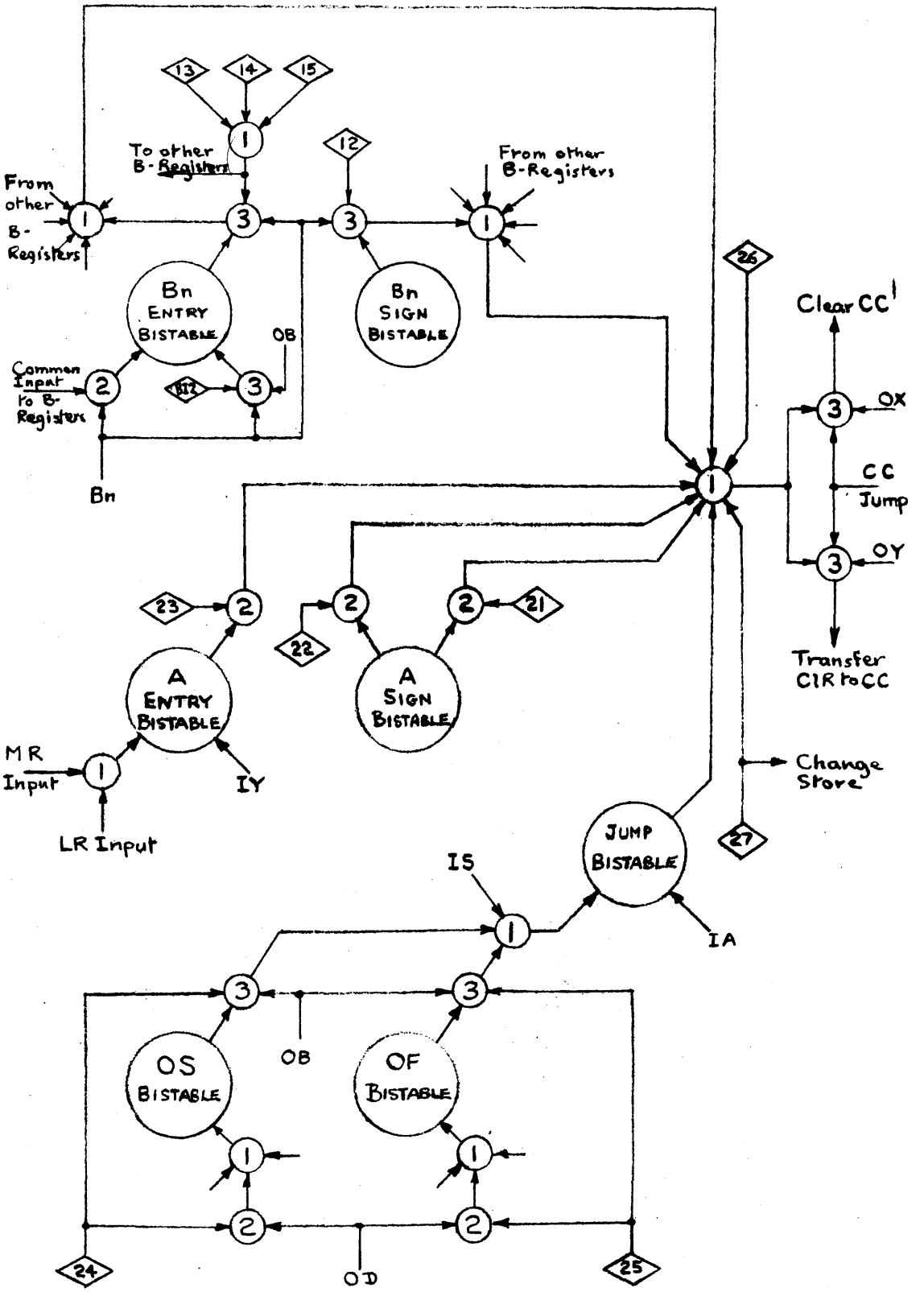
Fig. 28. Jump Instructions



(a) Fast Jump Flow Diagram



(b) Late Jump Flow Diagram



(c) LOGIC DIAGRAM

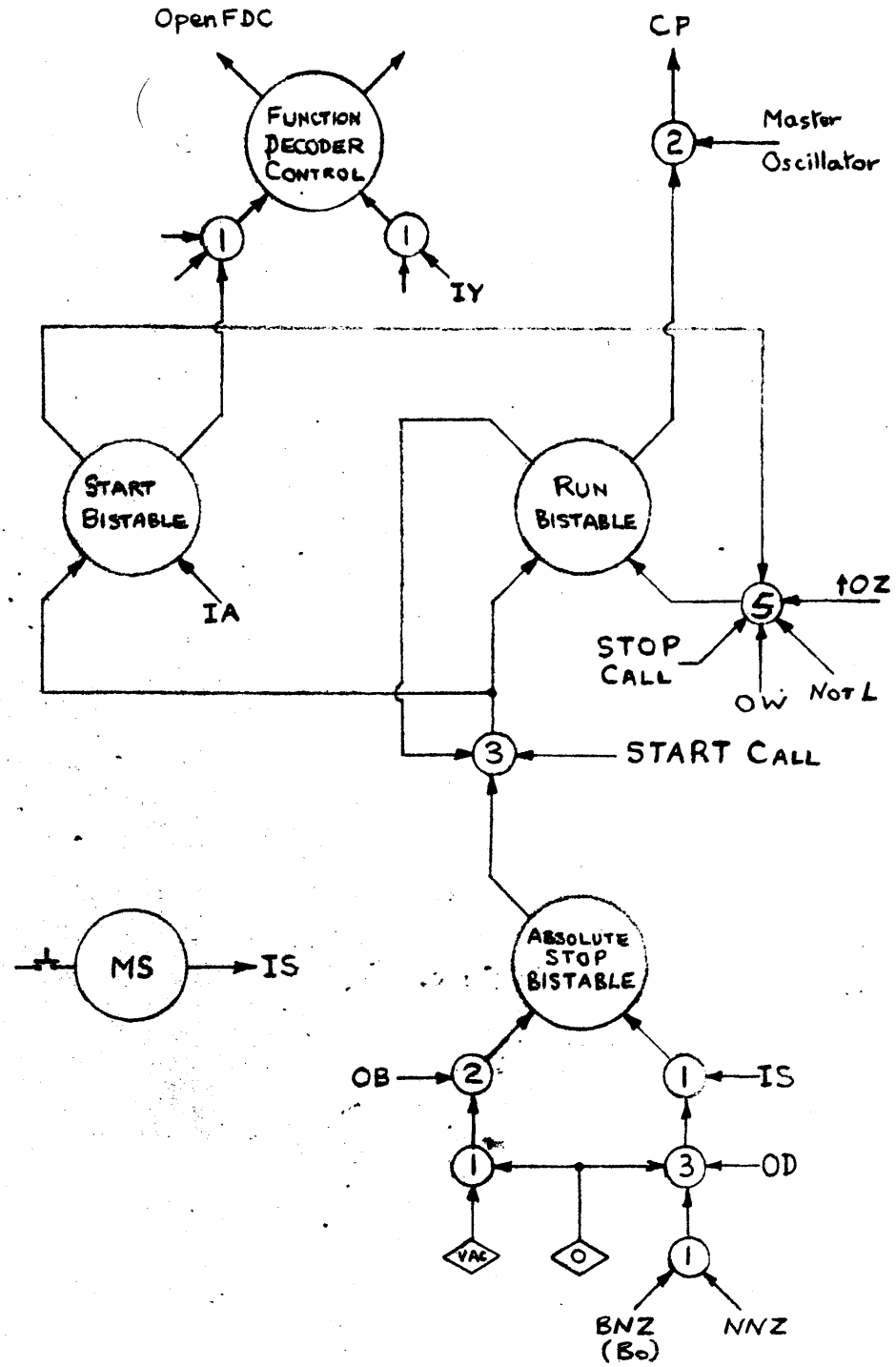


Fig. 29. STARTING AND STOPPING - LOGIC DIAGRAM

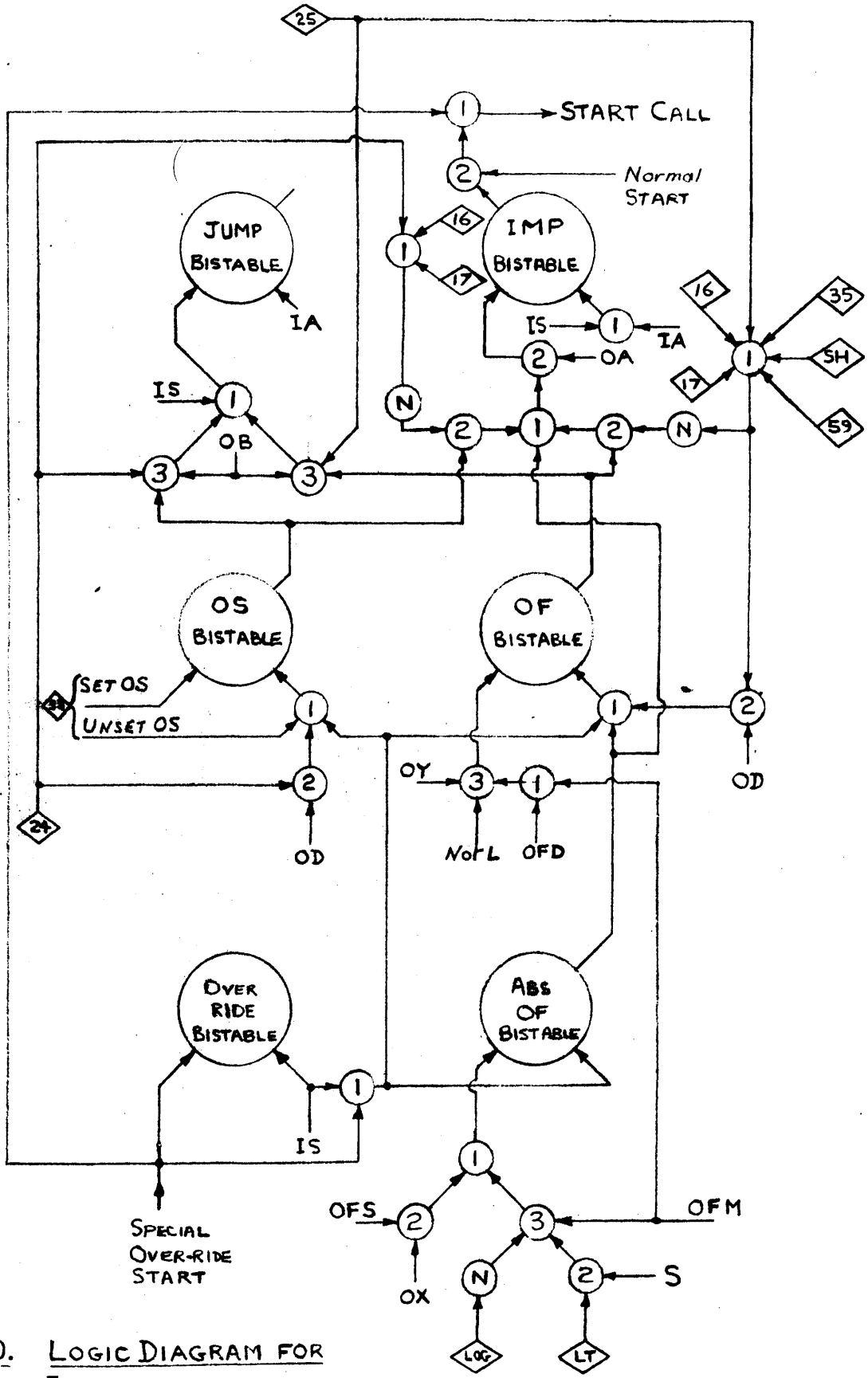


Fig. 30. LOGIC DIAGRAM FOR IMPERMISSIBILITIES

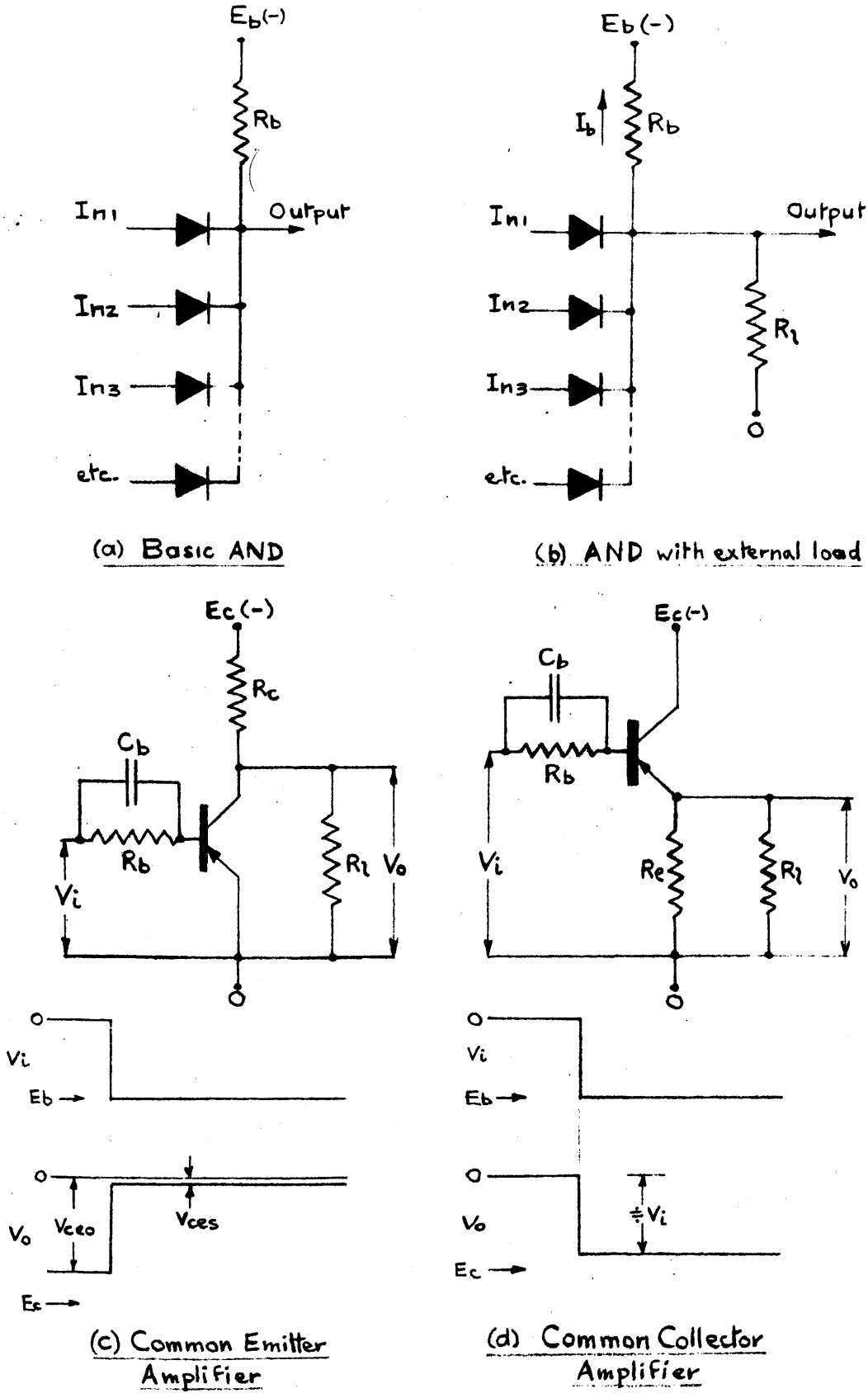


Fig. 31. AND Gates and Amplifiers

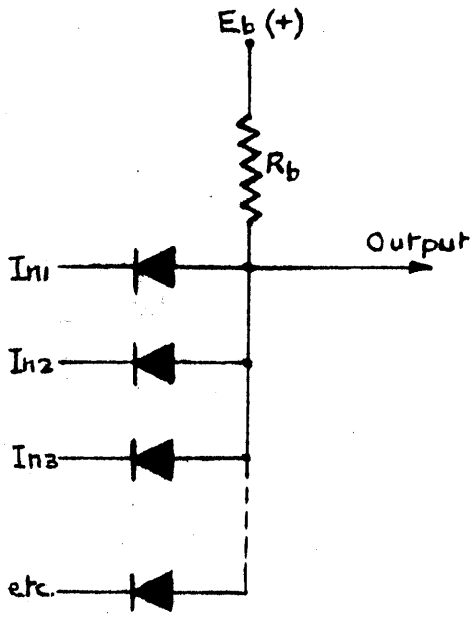
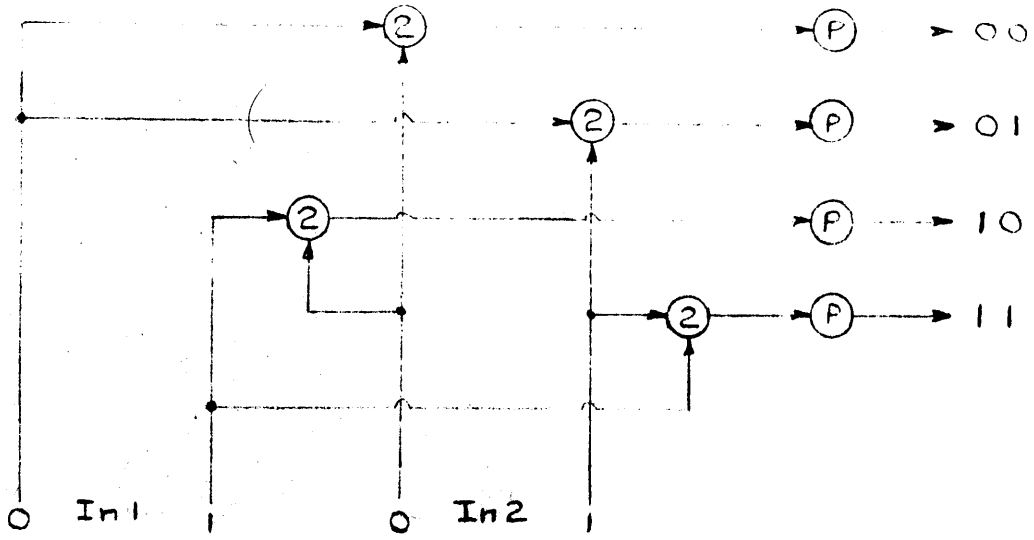
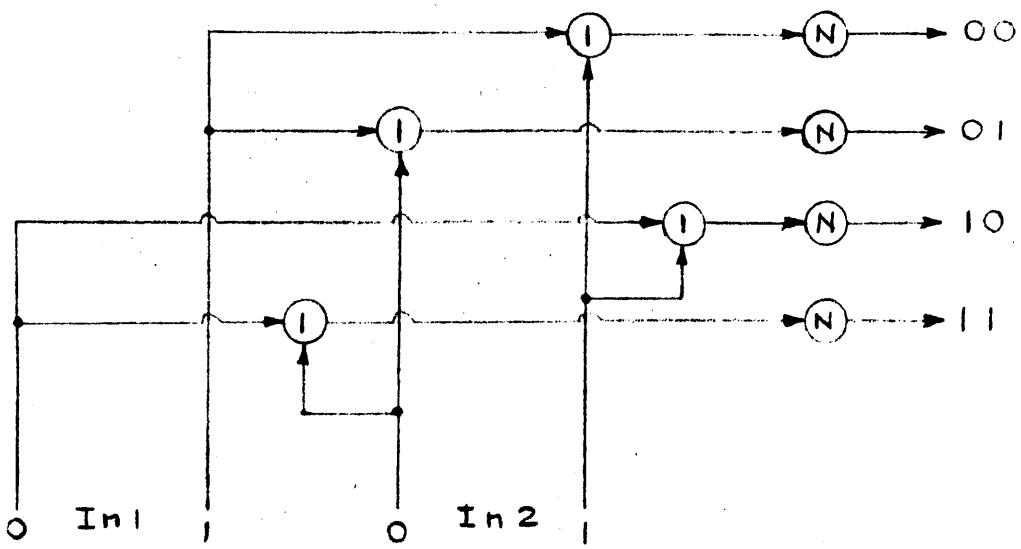


Fig. 32. OR GATE

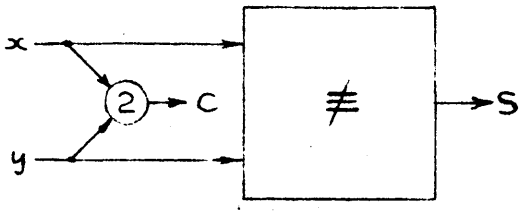


(a) NORMAL LOGIC



(b) INVERTED LOGIC

Fig. 33. DECODER UNIT



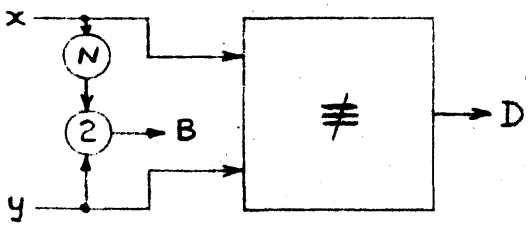
(a) HALF ADDER

TRUTH TABLE $(x+y)$

x	y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$S = x \neq y$

$C = x.y$



(b) HALF SUBTRACTOR

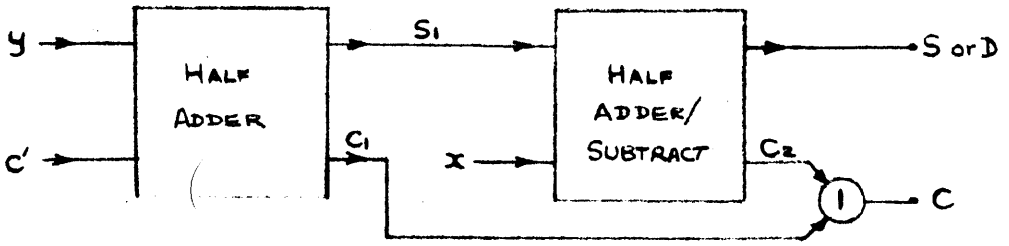
TRUTHTABLE $(x-y)$

x	y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

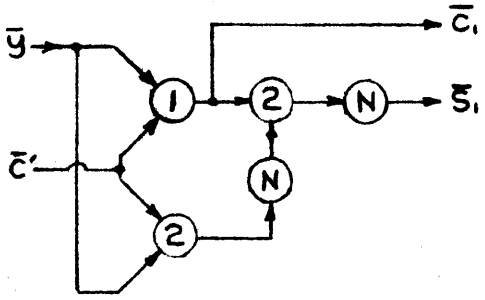
$D = x \neq y$

$B = \bar{x}.y$

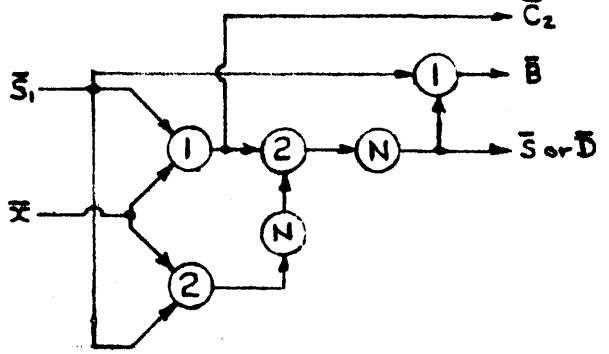
Fig. 34. BASIC ARITHMETIC UNITS



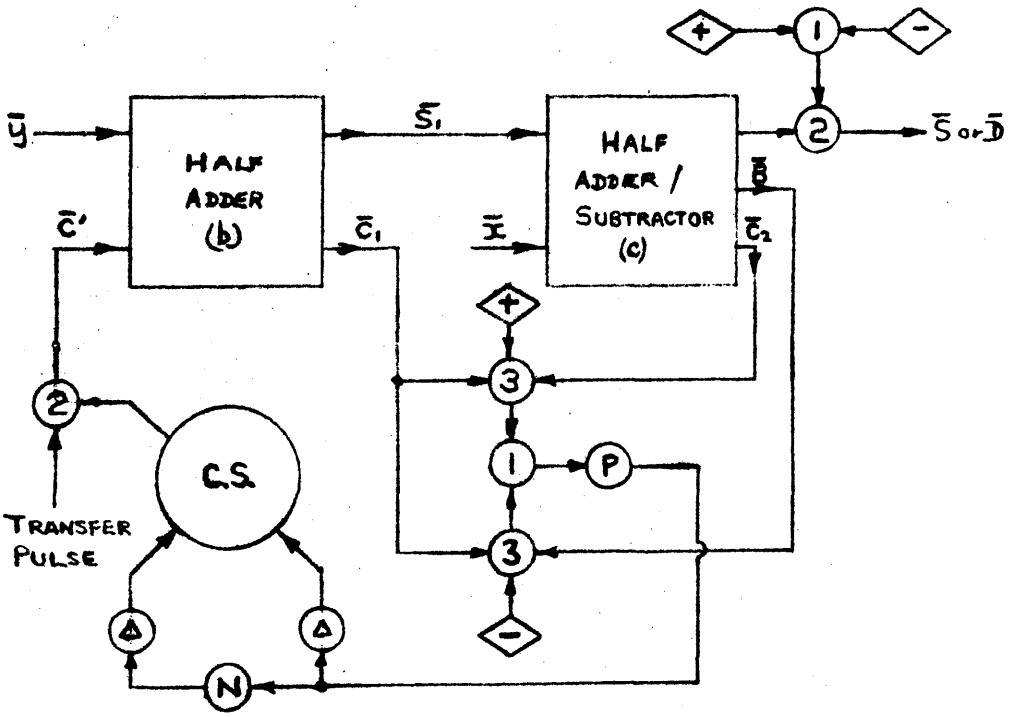
(a) BASIC ARRANGEMENT



(b) HALF ADDER



(c) HALF ADDER/SUBTRACTOR



(d) FINAL ARRANGEMENT

Fig. 35. MAIN ADDER - SUBTRACTOR UNIT

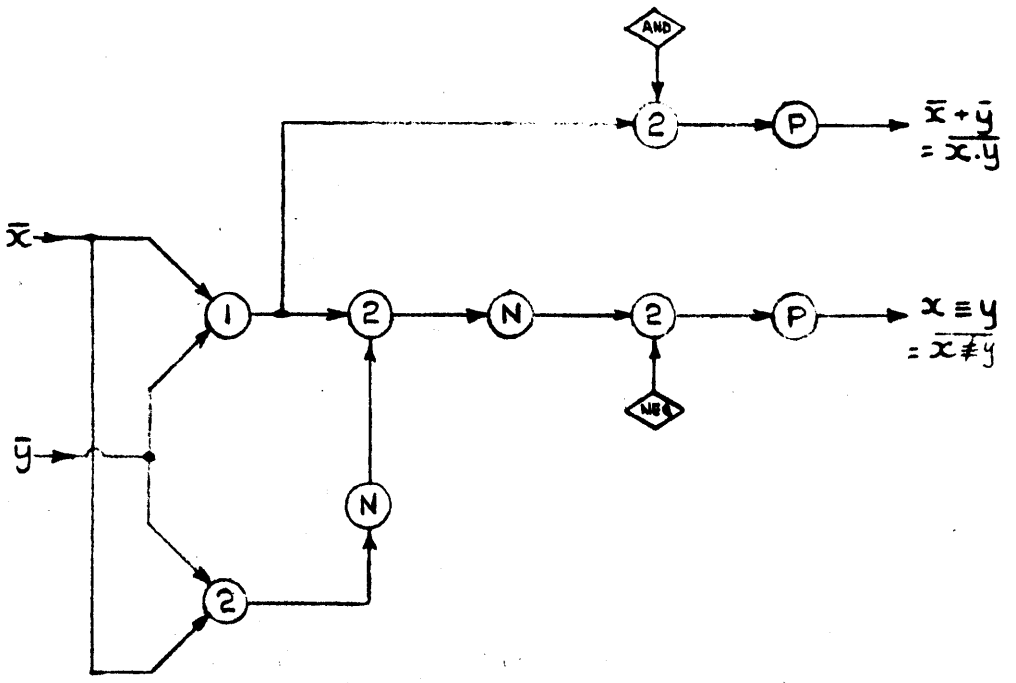


Fig. 36. LOGICAL FUNCTION UNIT

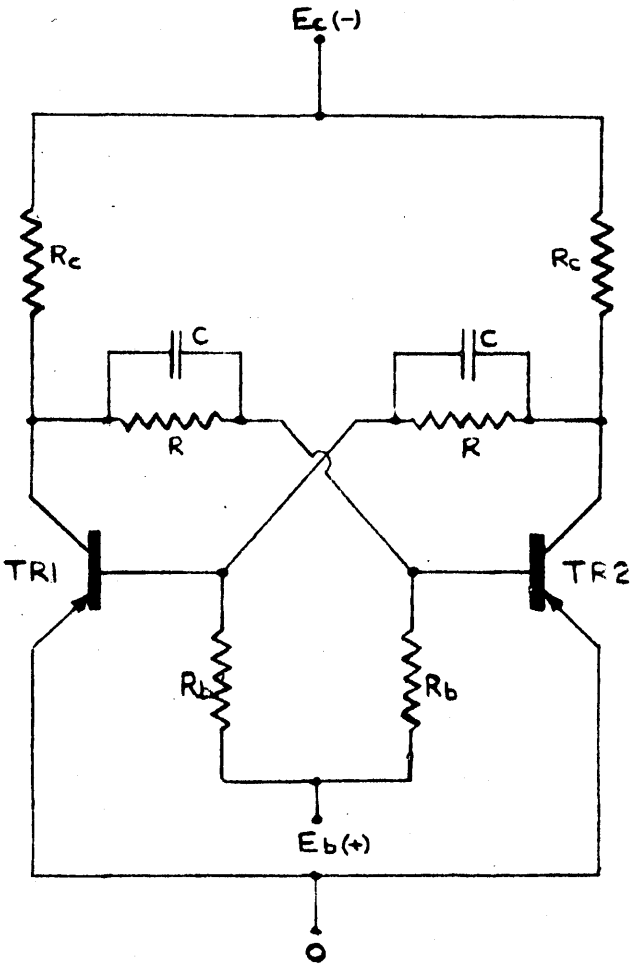


Fig. 37. BASIC BISTABLE UNIT

Region	TR1	TR2
A	Bottomed	Cut-off
B	Active	Cut-off
C	Active	Active
D	Active	Bottomed
E	Cut-off	Bottomed

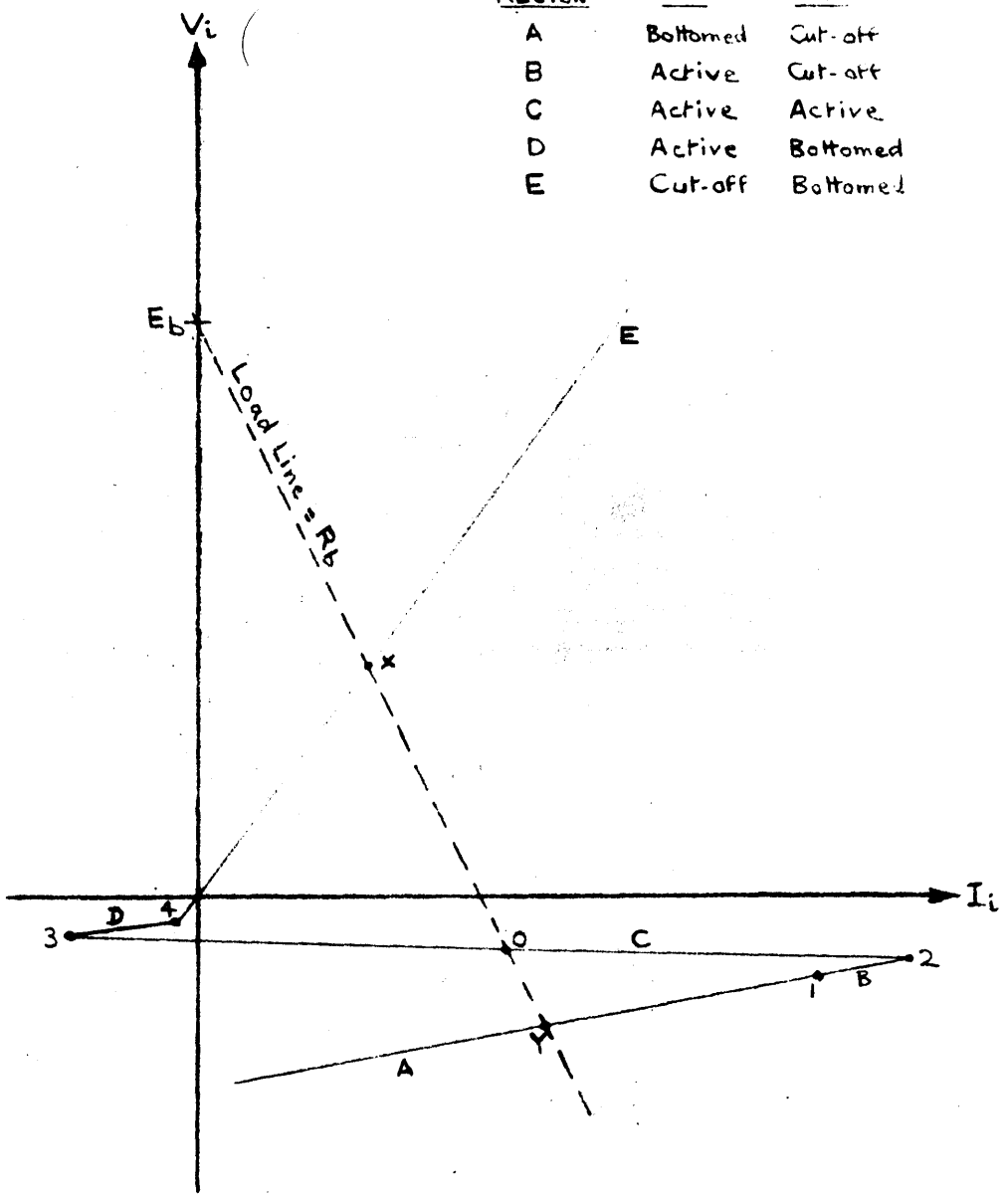


Fig. 38. Input Characteristic of Transistor Bistable Unit

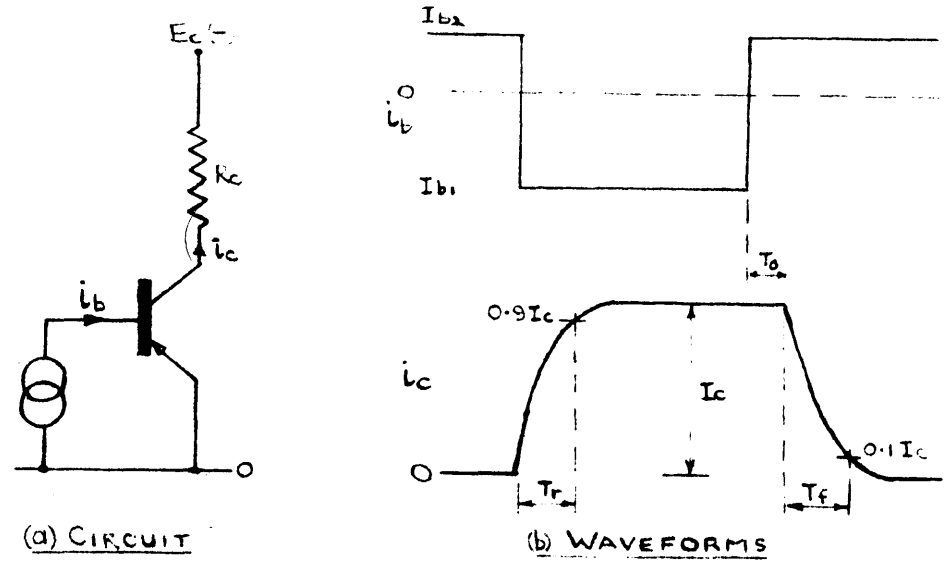


Fig. 39. COMMON EMITTER SWITCH

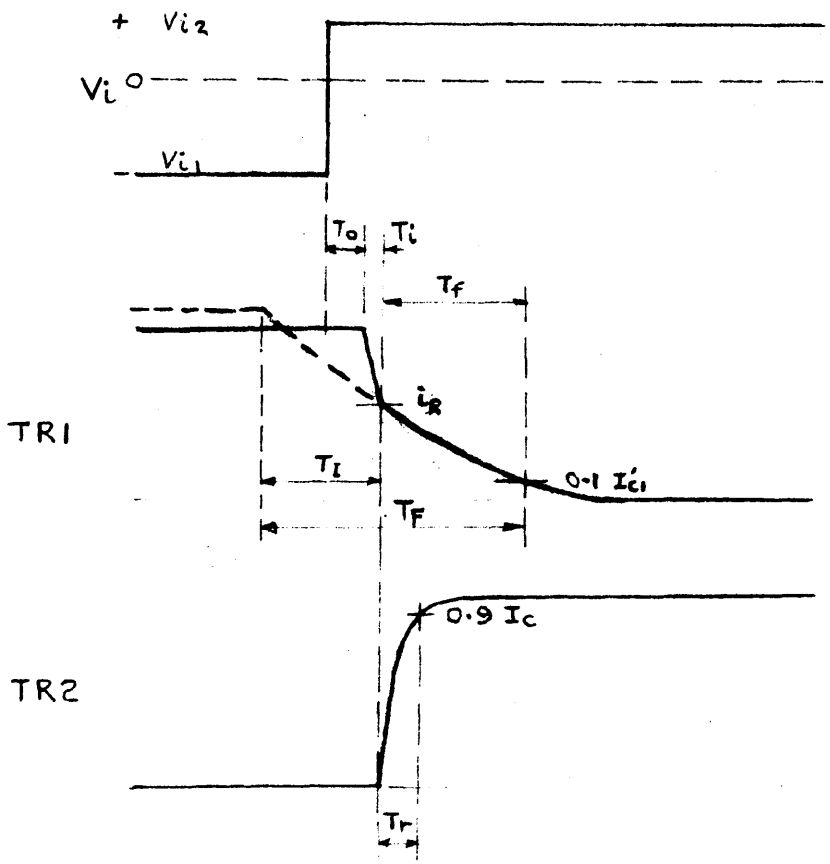
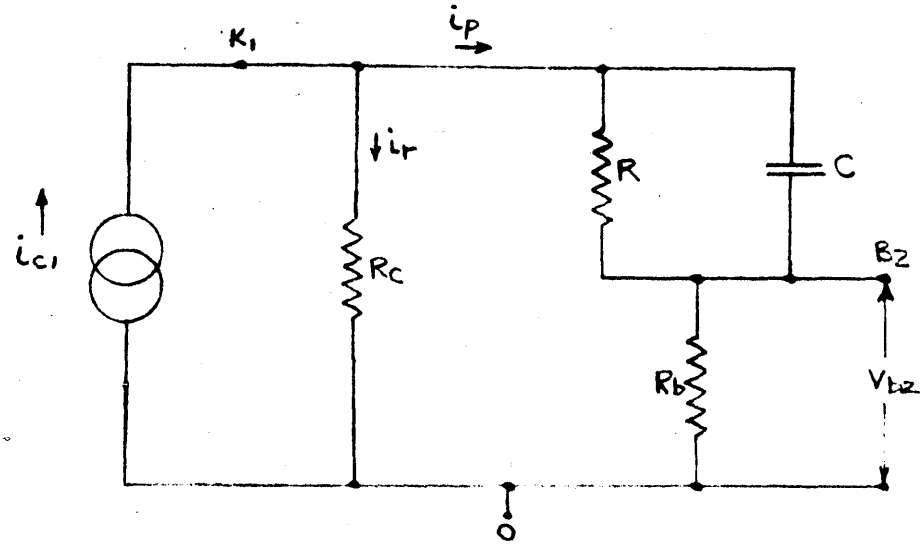


Fig. 40. BISTABLE SWITCHING



(a) During times T_0 and T_i circuit is as above.

(b) After T_i, B_2 assumed connected to 0 due to transistor TR2.

Fig. 41. EQUIVALENT CIRCUIT BETWEEN THE COLLECTOR OF TR1 AND THE BASE OF TR2

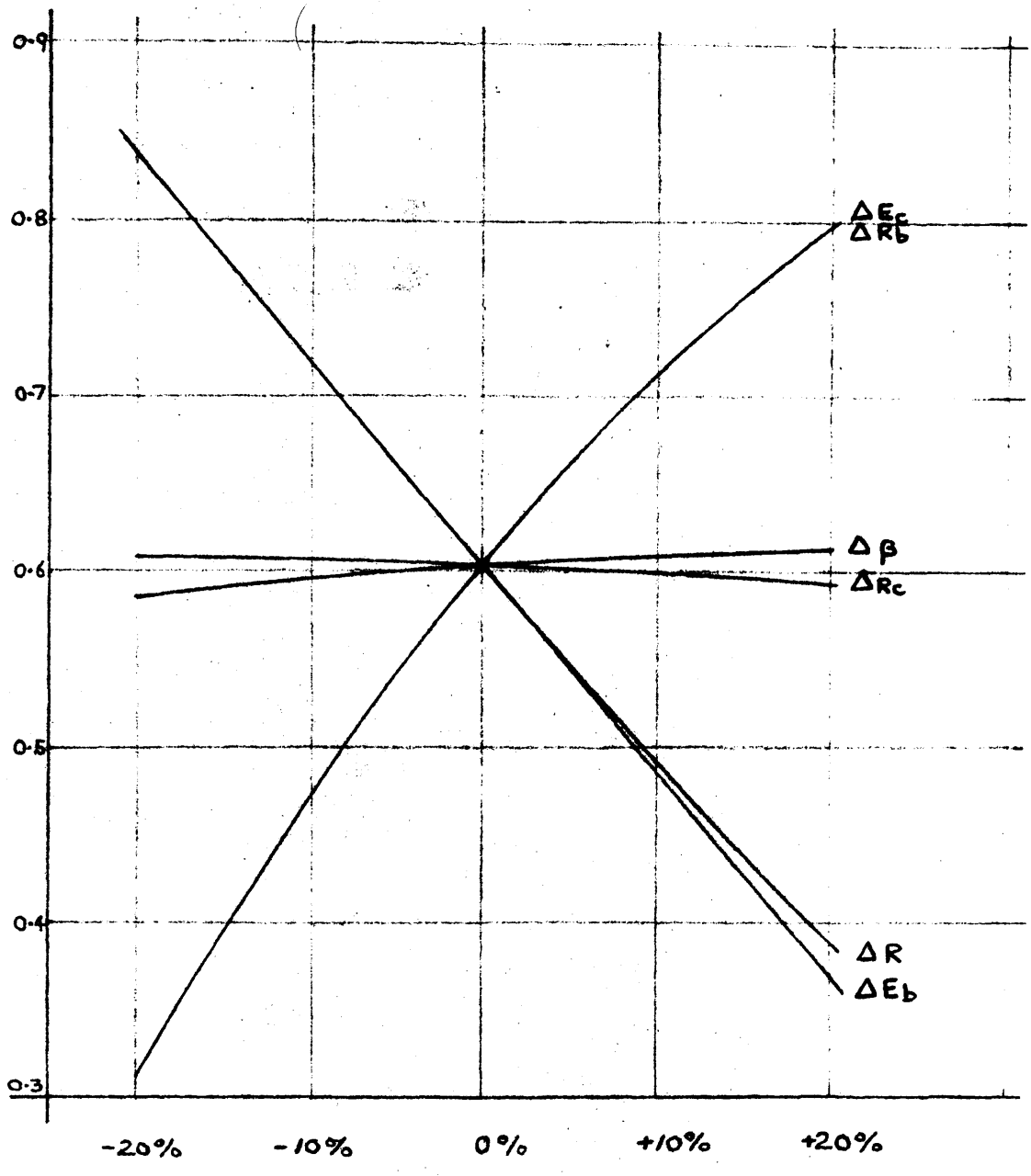


Fig.4.2. STABILITY FACTOR

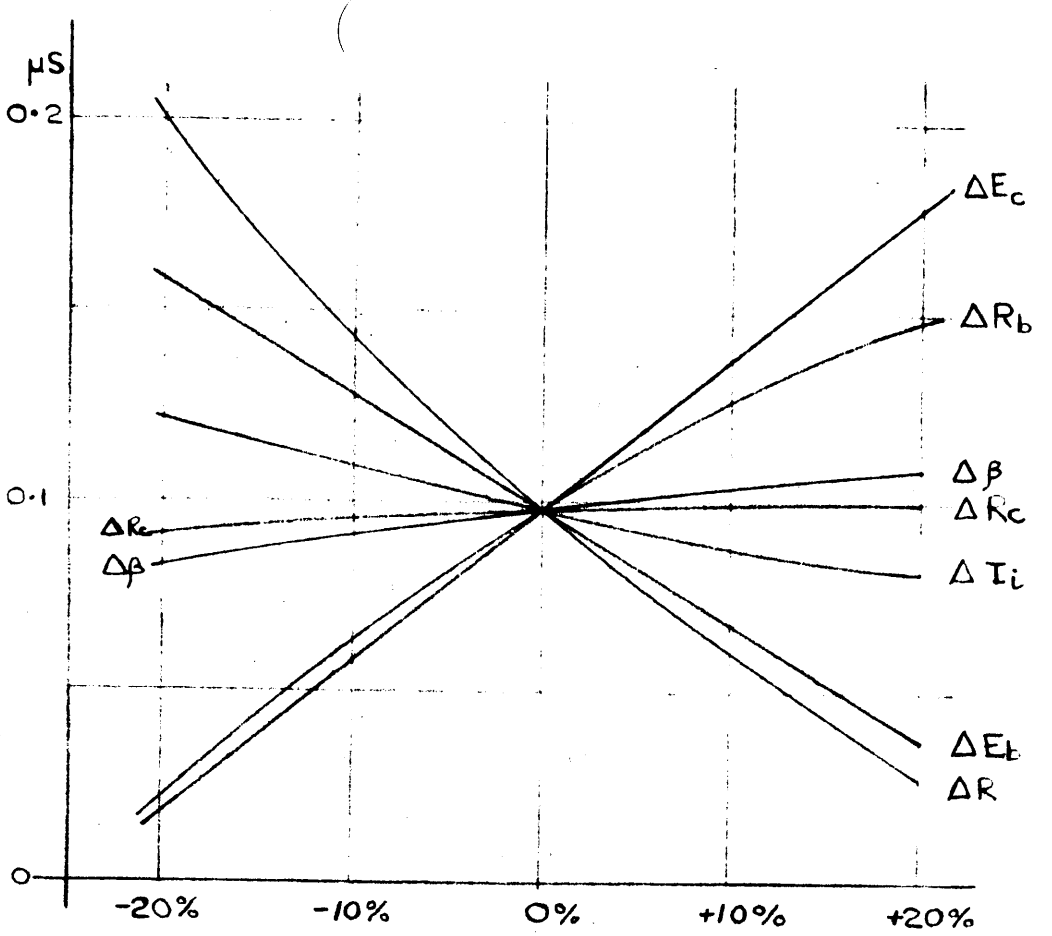


Fig. 43. DELAY TIME. T_0

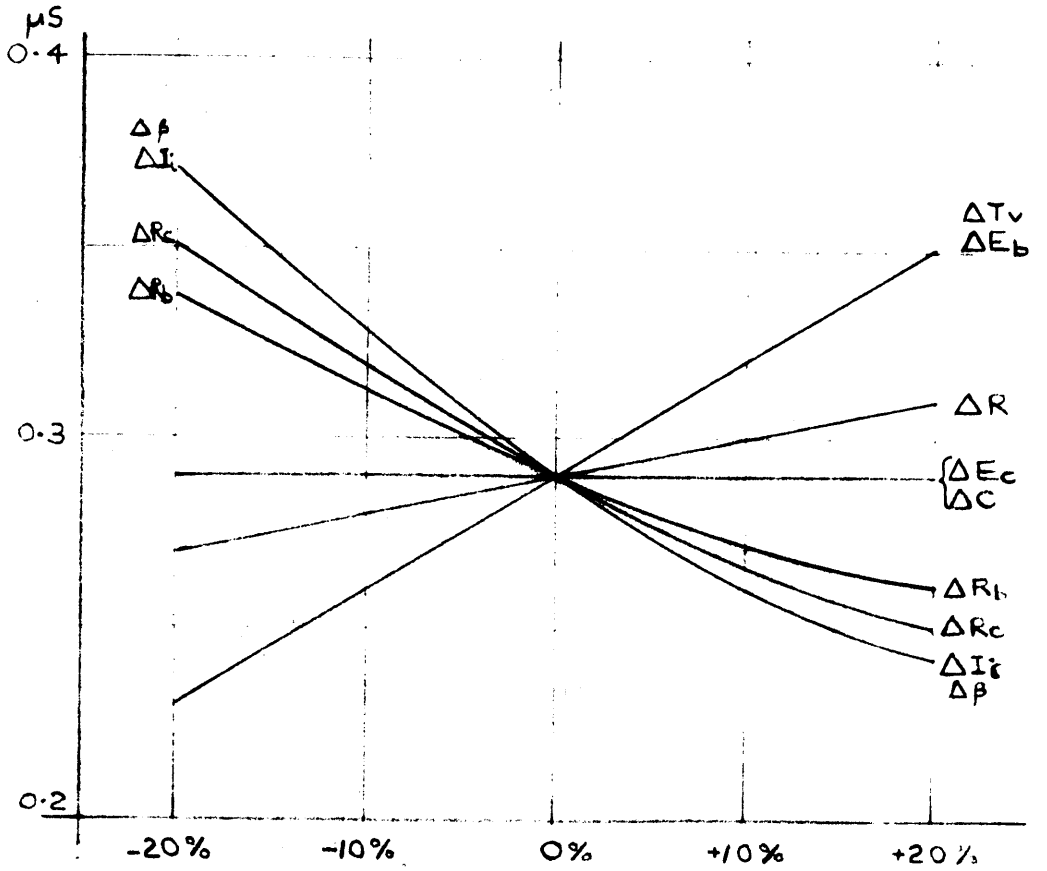


Fig. 44. INITIAL FALL TIME . Ti

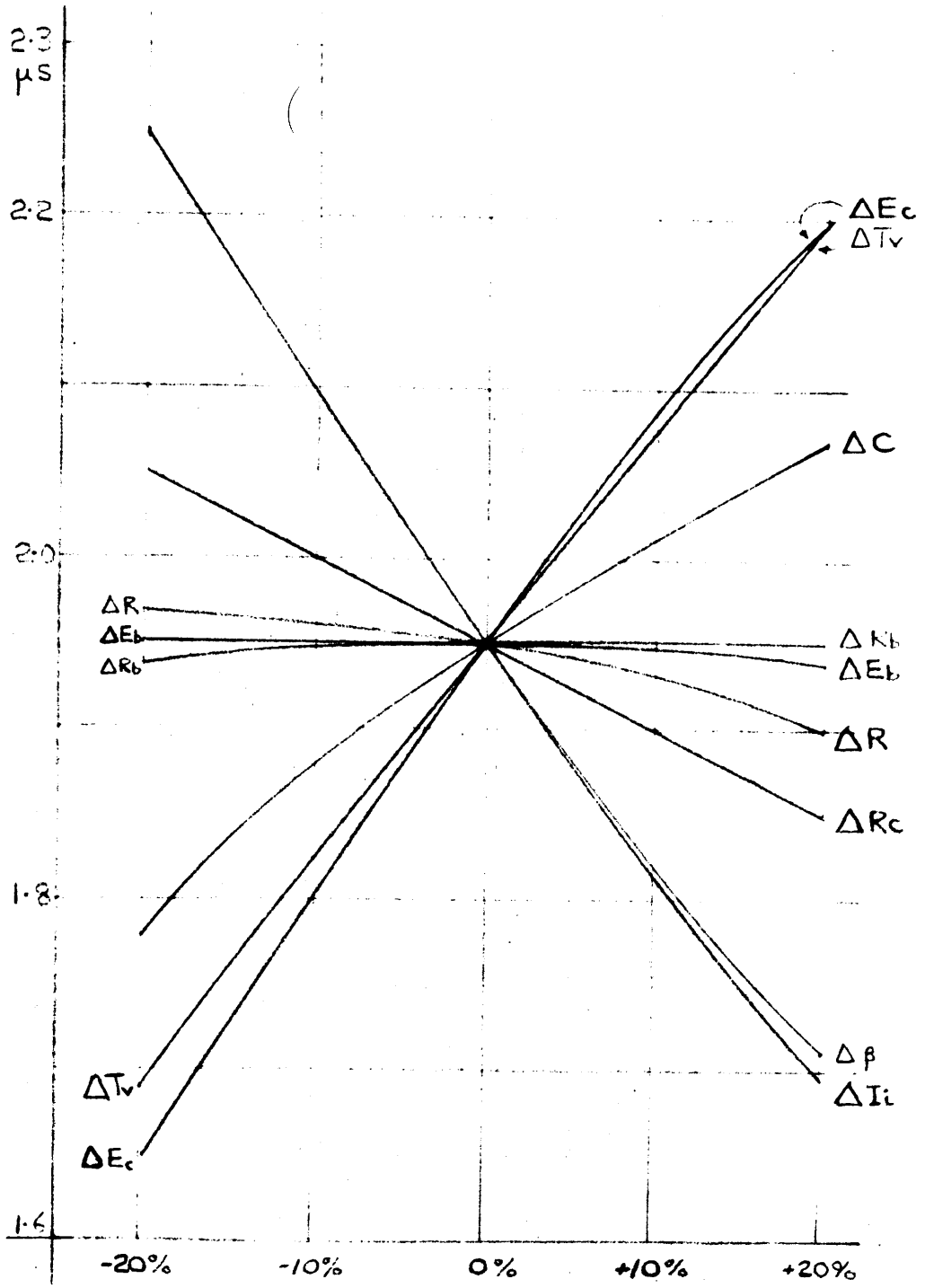


Fig. 45. FALL TIME. T_f

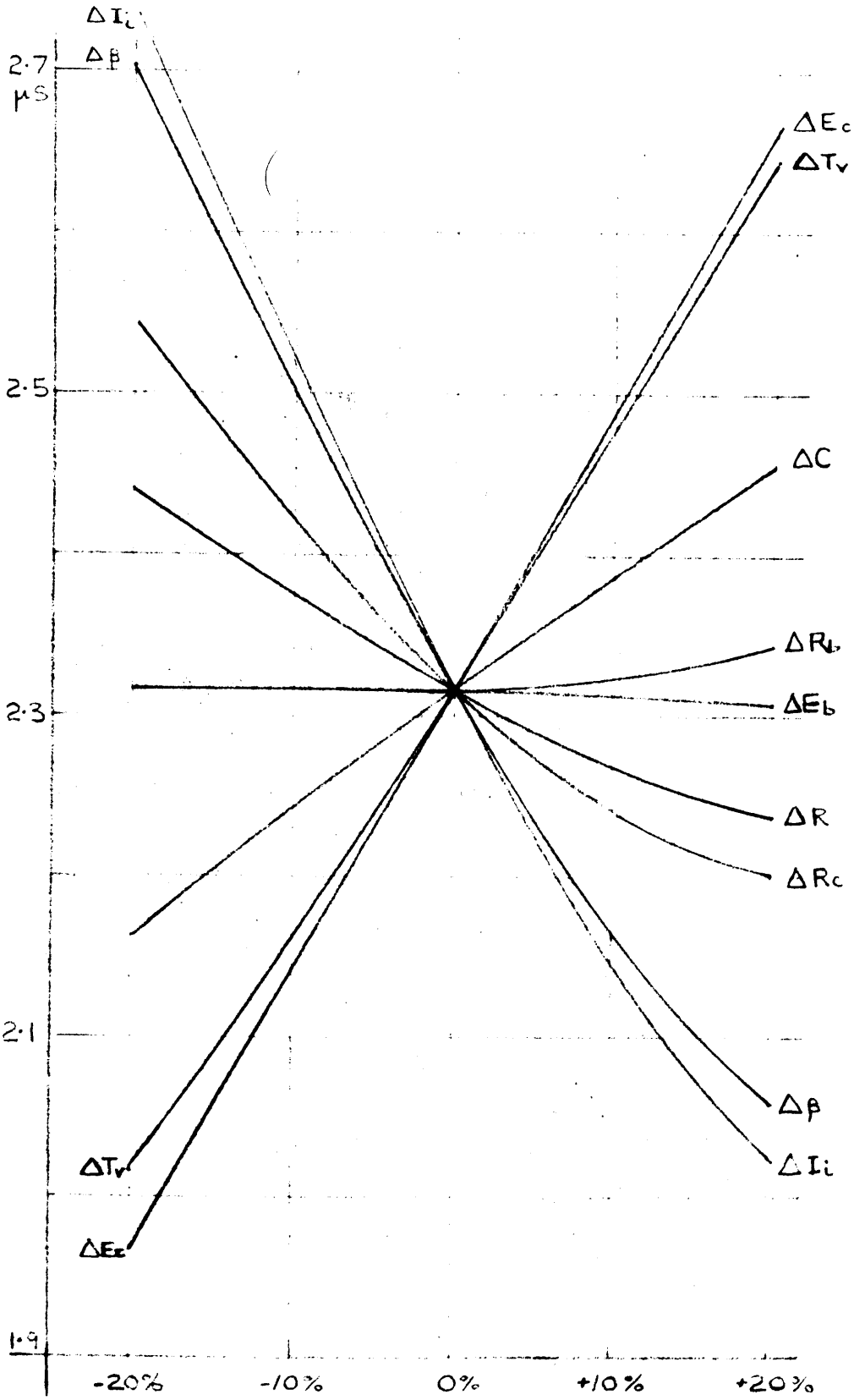


Fig. 46. Total Switching Time. T_t

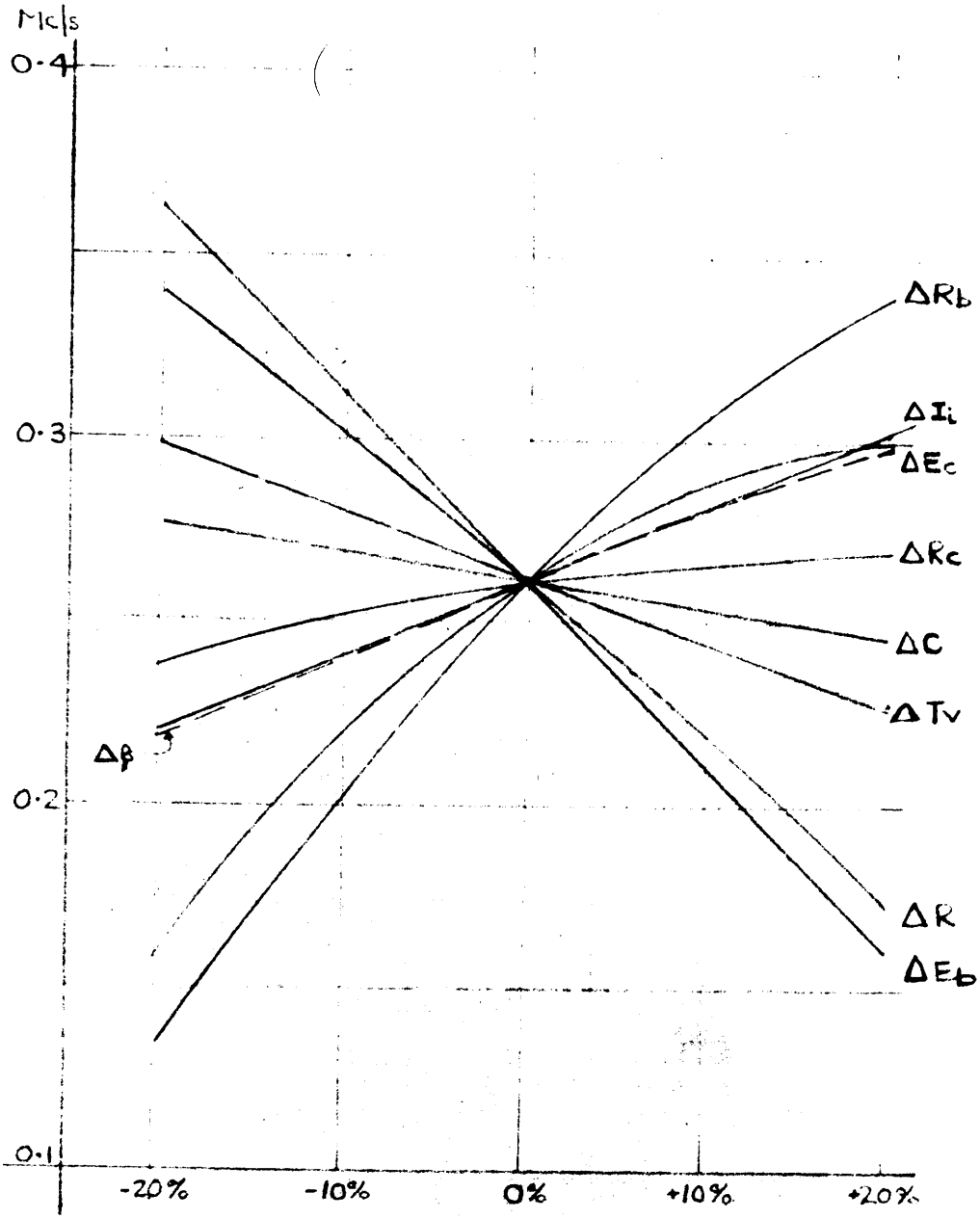


Fig. 47 FIGURE OF MERIT. M

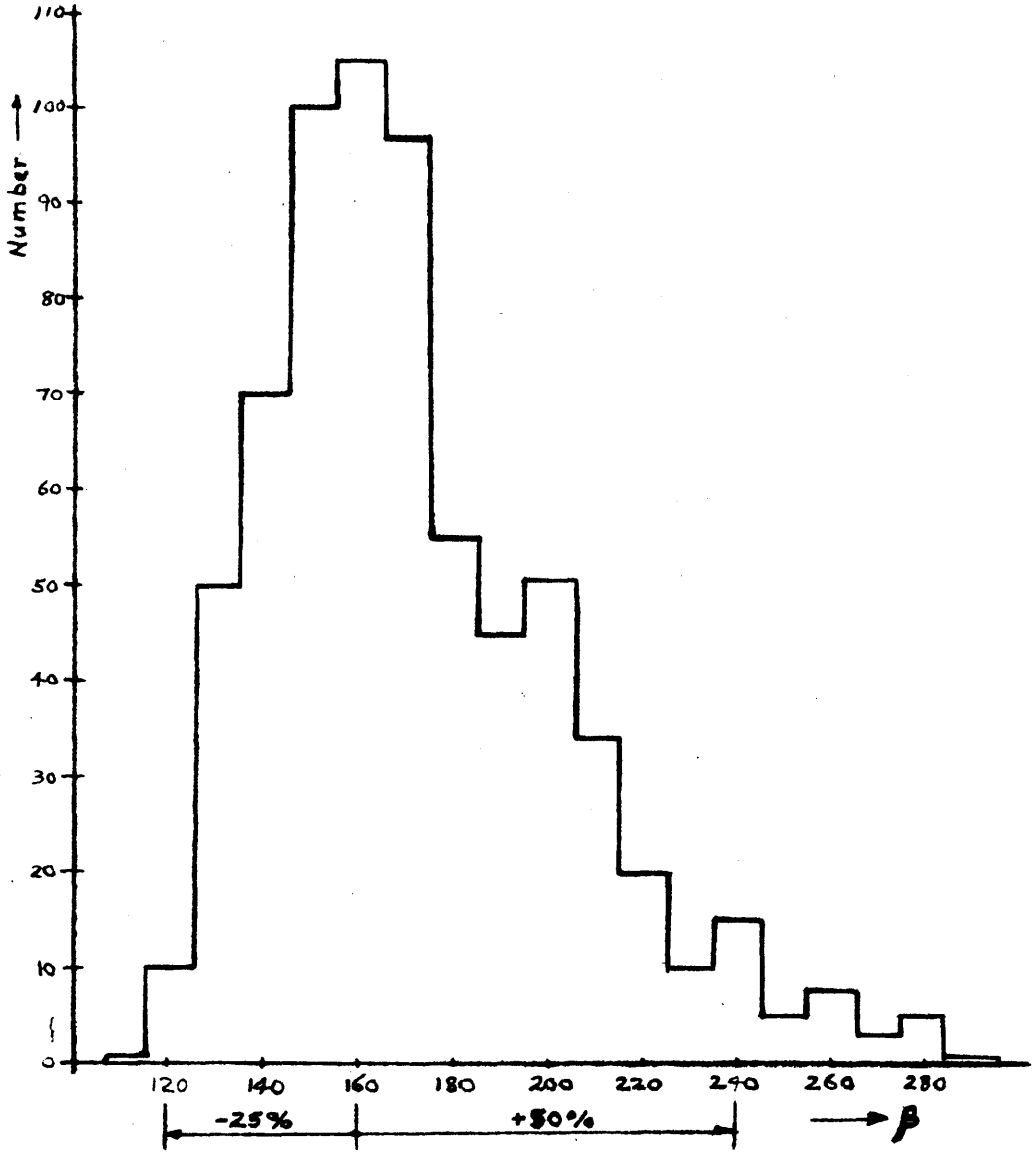


Fig. 48. VARIATION OF β

Fig. 48. VARIATION OF

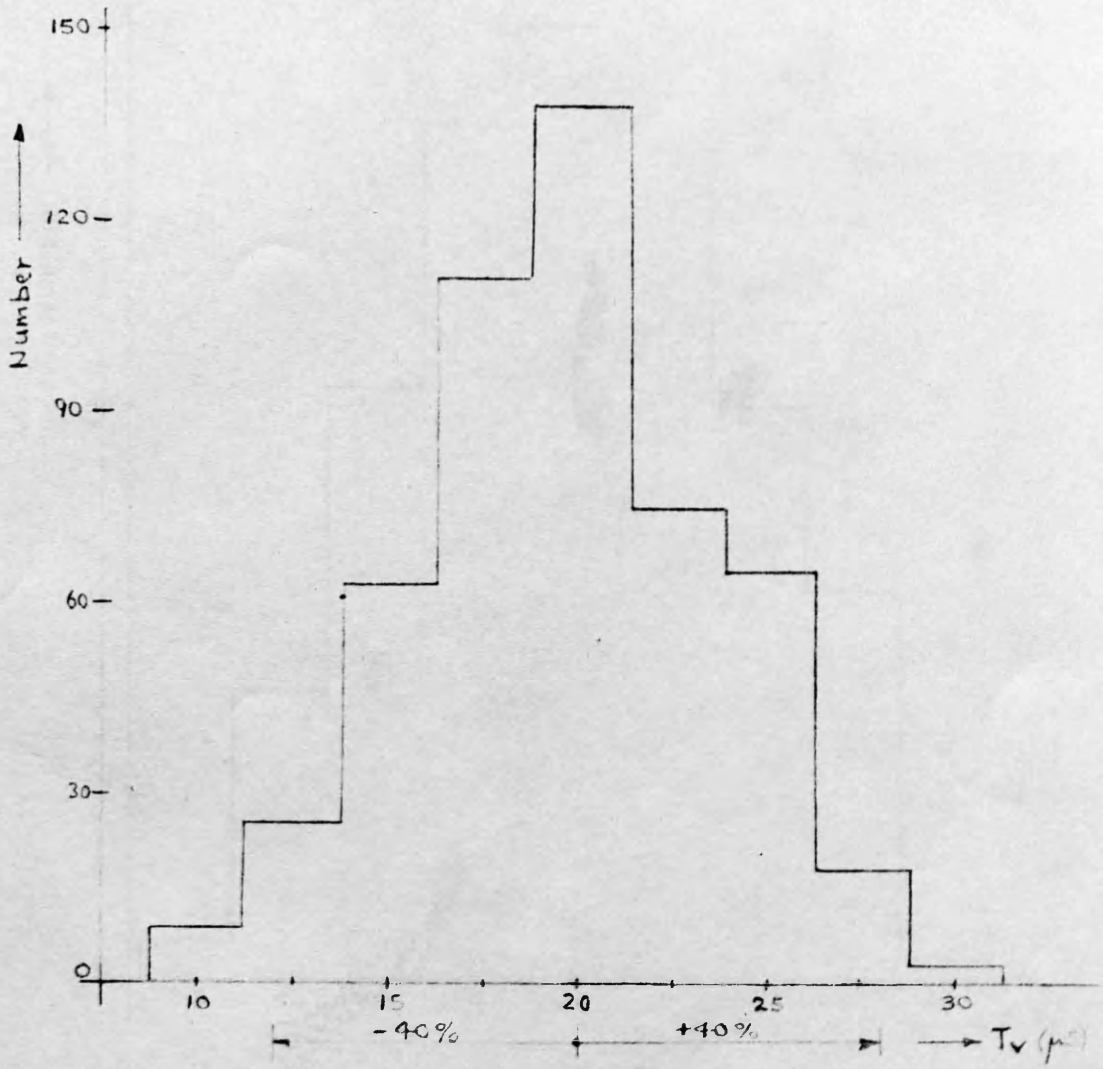


Fig. 49. VARIATION OF T_v

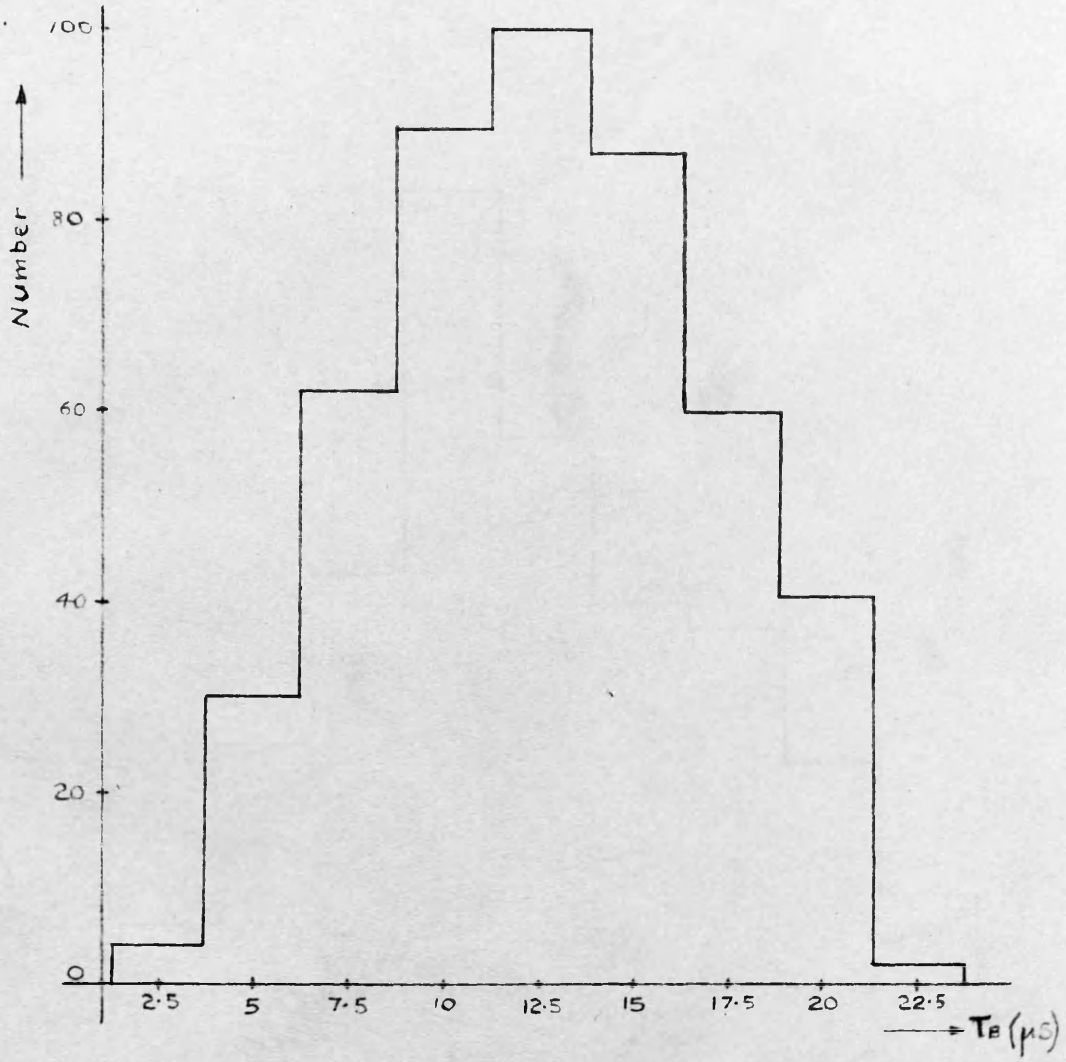


Fig. 50. VARIATION OF T_B

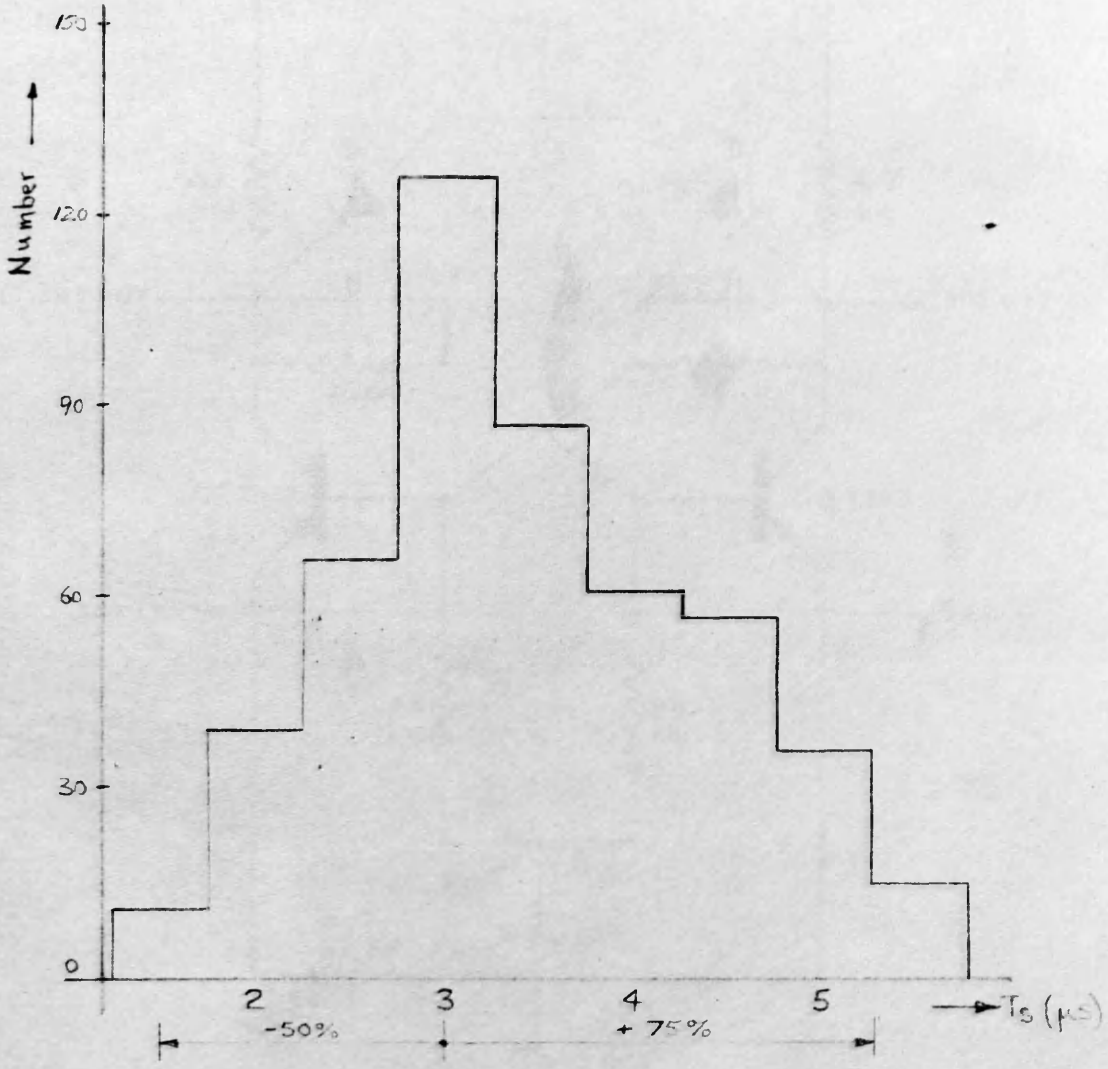


Fig.51. VARIATION OF T_s

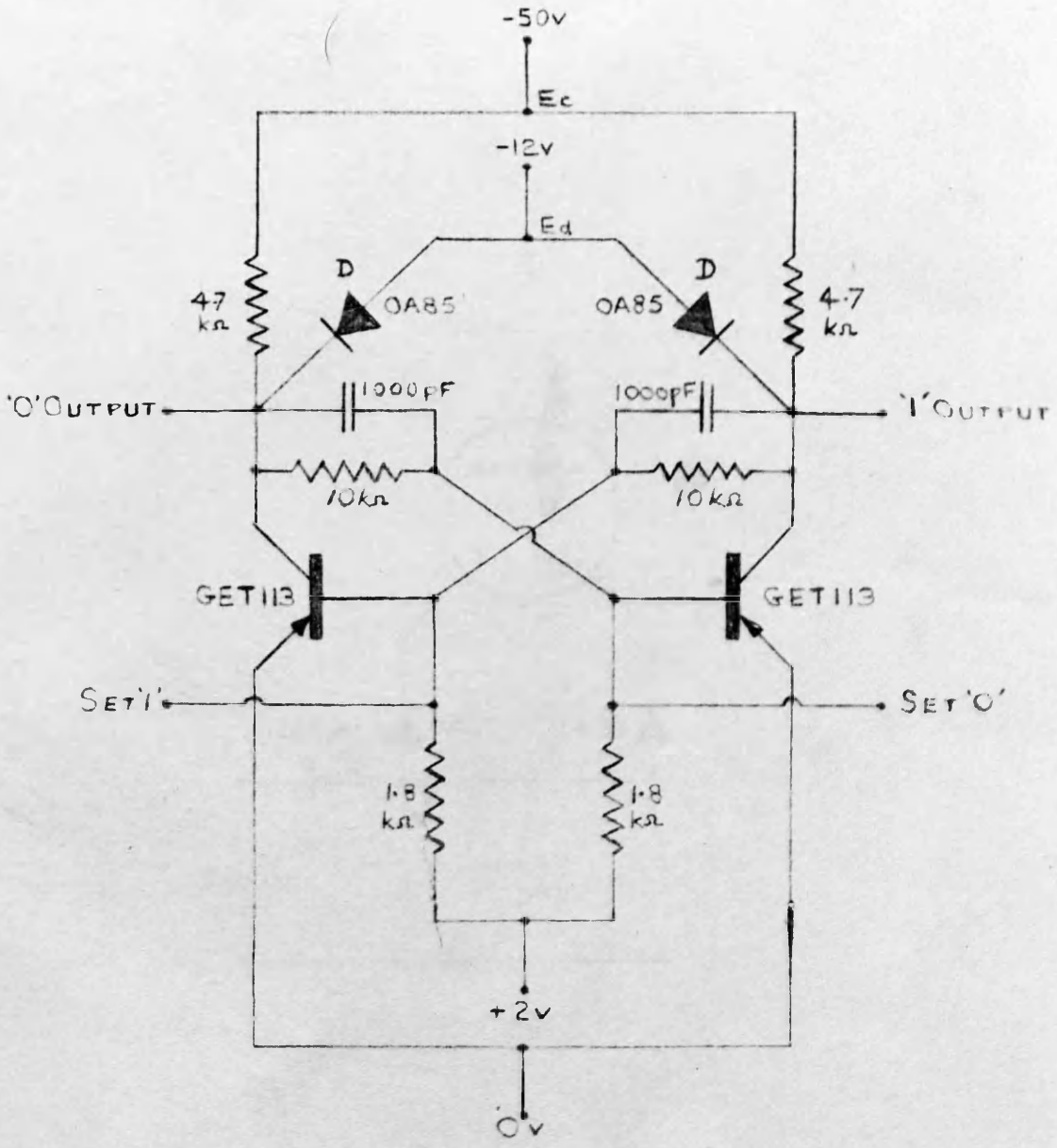


Fig. 52. Final Bistable Unit

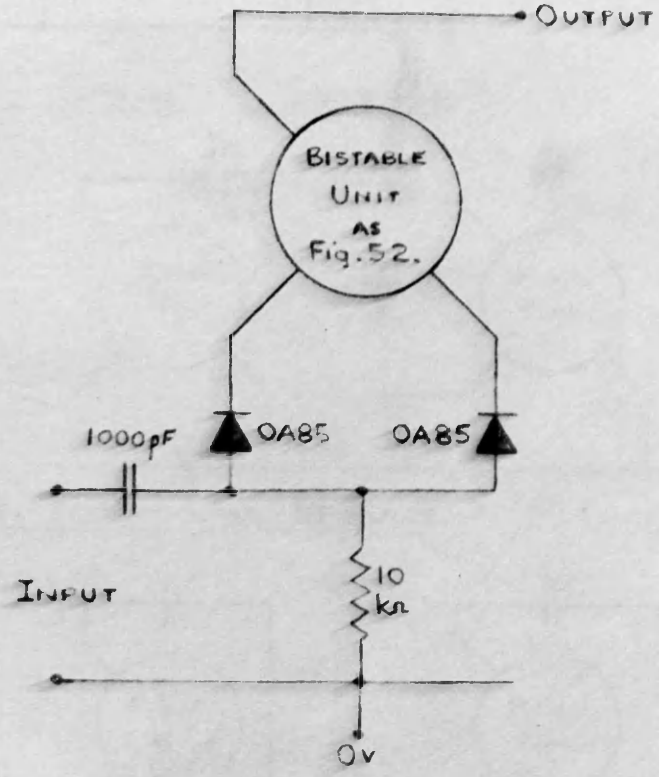
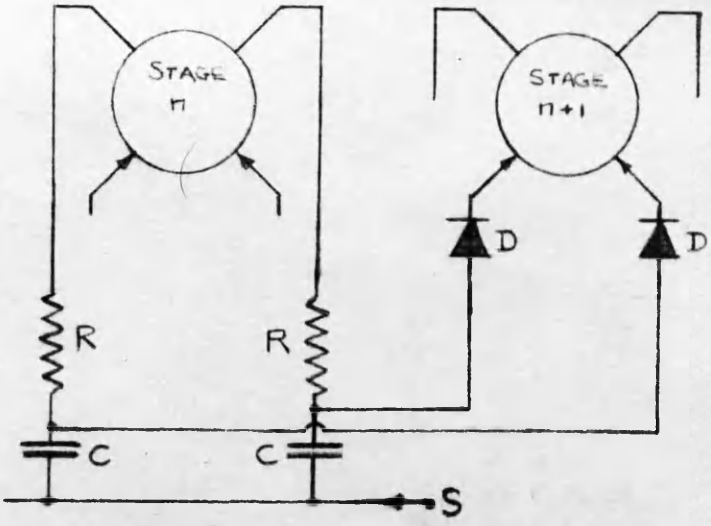
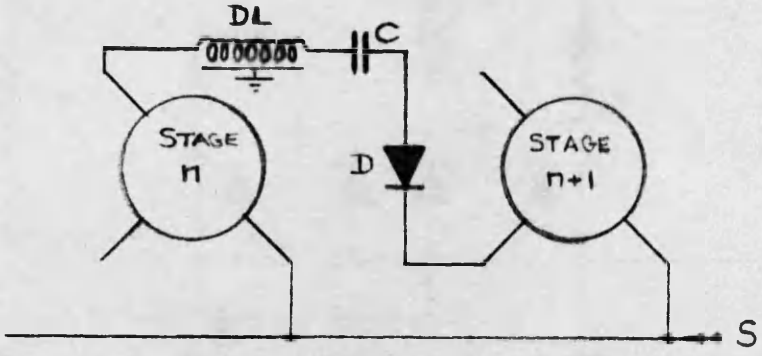


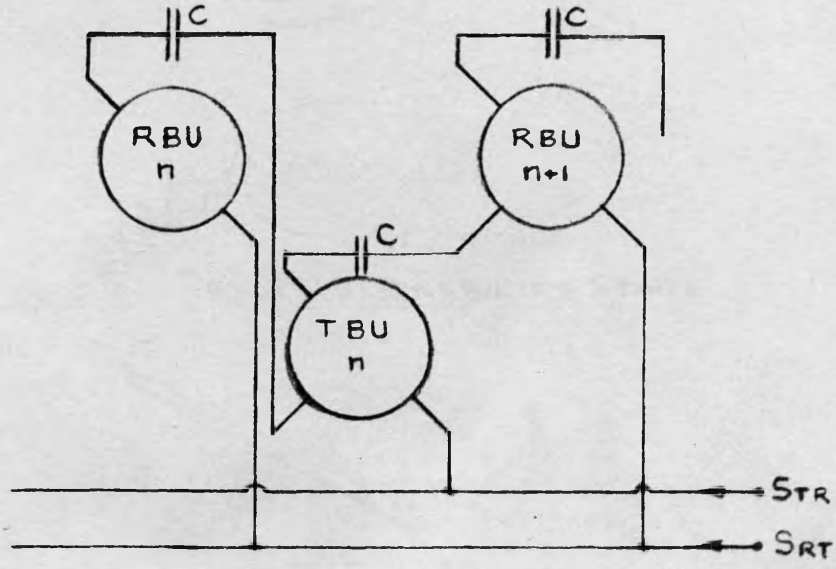
Fig. 53. COUNTER UNIT



(a) CAPACITY STORAGE



(b) DELAY LINE STORAGE



(c) BISTABLE UNIT STORAGE

Fig. 54. REGISTER TEMPORARY STORAGE

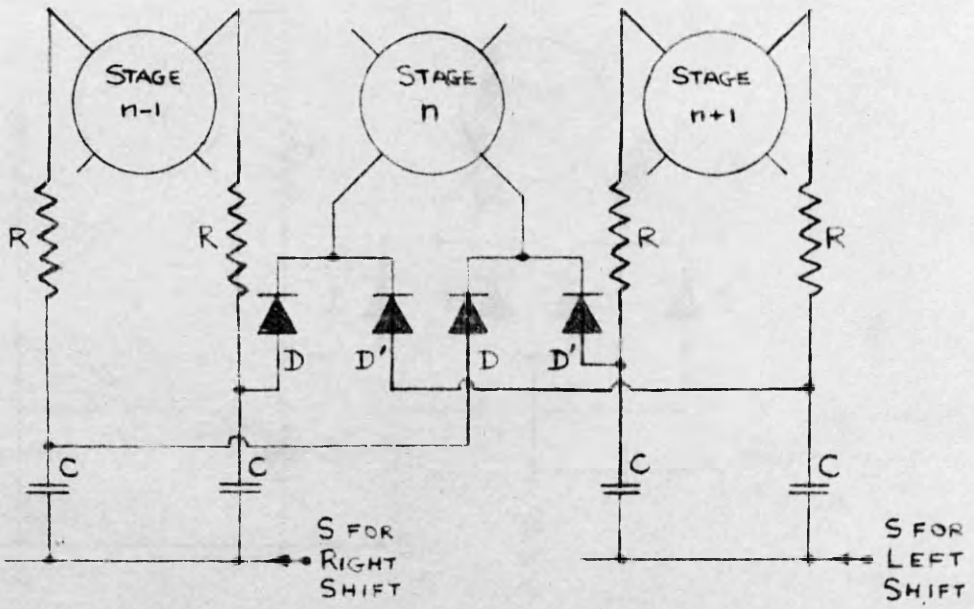


Fig. 55. REVERSIBLE REGISTER STAGE

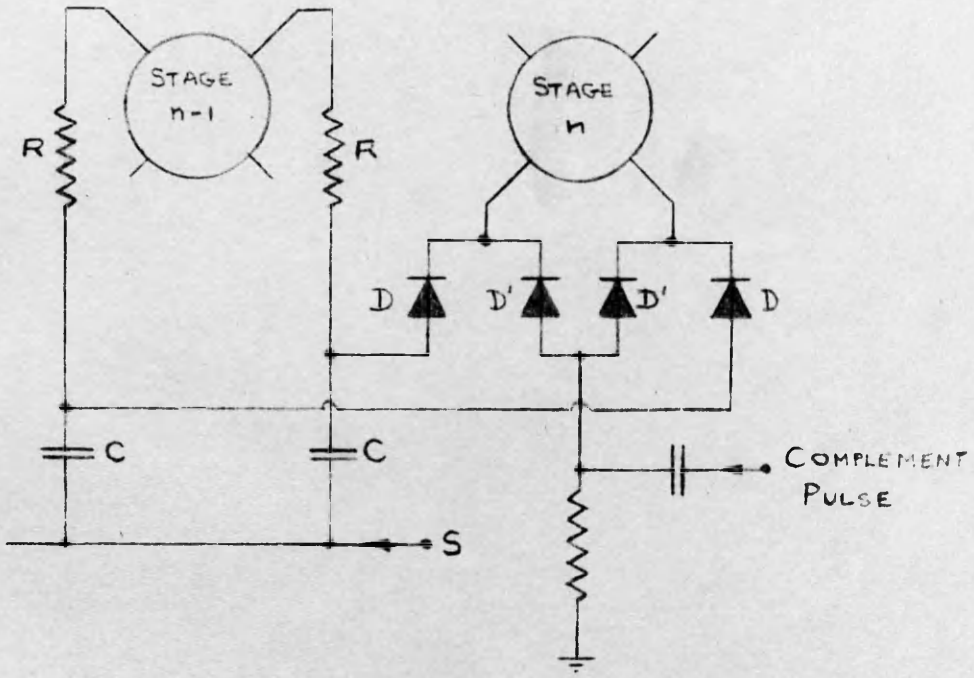
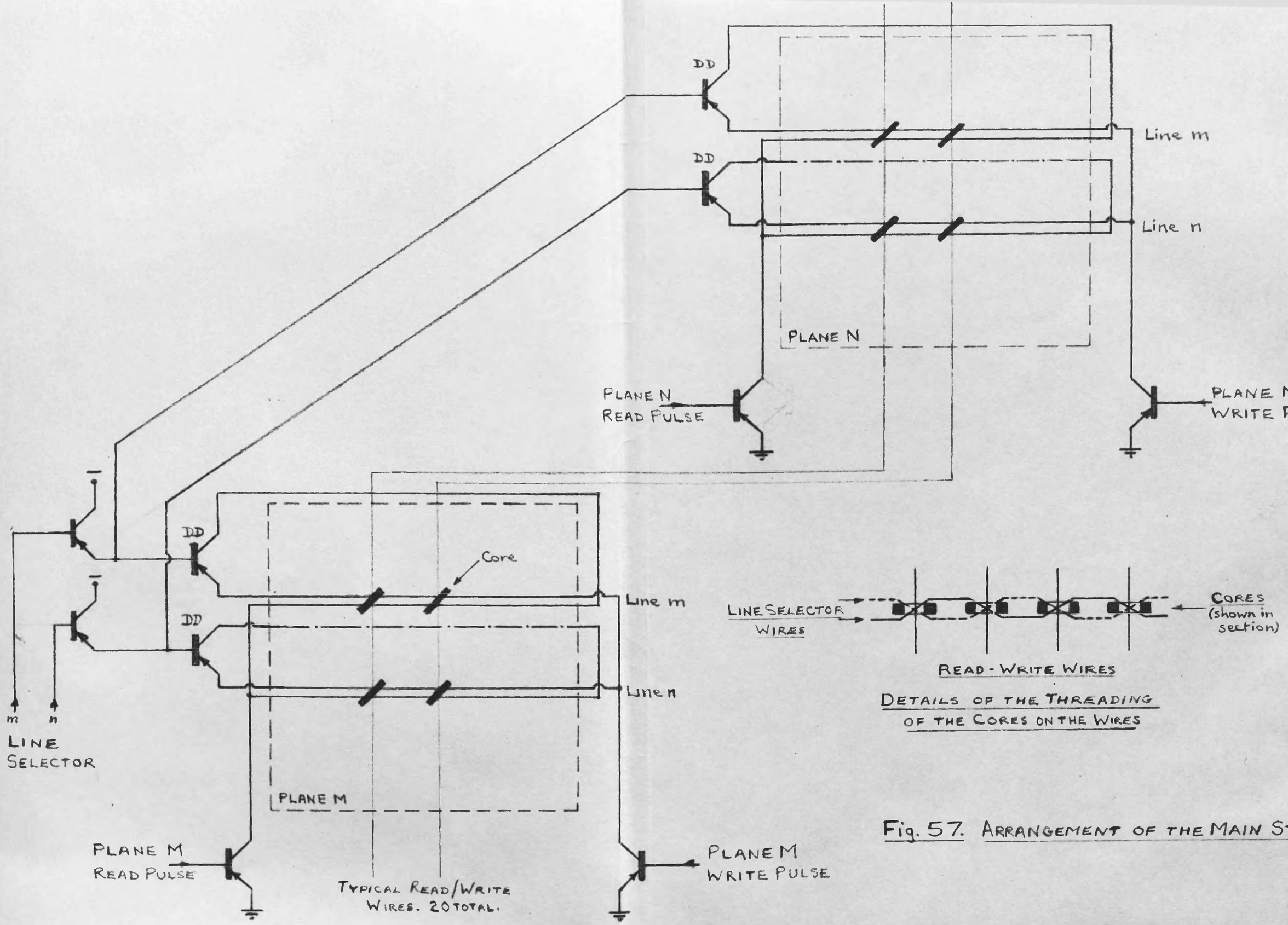


Fig. 56. COMPLEMENTING REGISTER STAGE

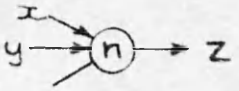


DETAILS OF THE THREADING
OF THE CORES ON THE WIRES

Fig. 57. ARRANGEMENT OF THE MAIN STORE

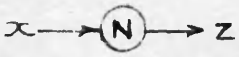
OR $Z = x + y + \cdots$

AND $Z = x.y. \ldots$



n is the number of inputs required to produce an output.

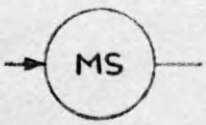
NOT $Z = \bar{x}$



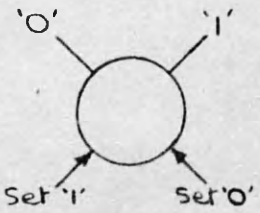
Power Stage

Digit Delay

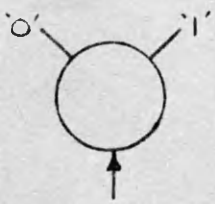
Monostable Unit



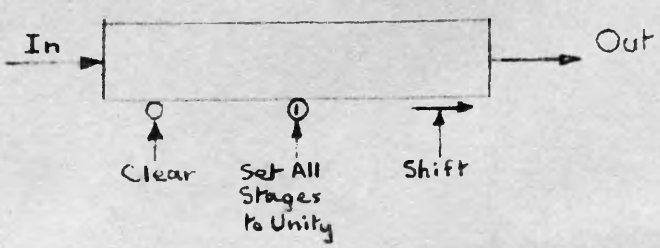
Bistable Unit



Counter Unit



Register



Function References

Fig.58. Logical Sybols

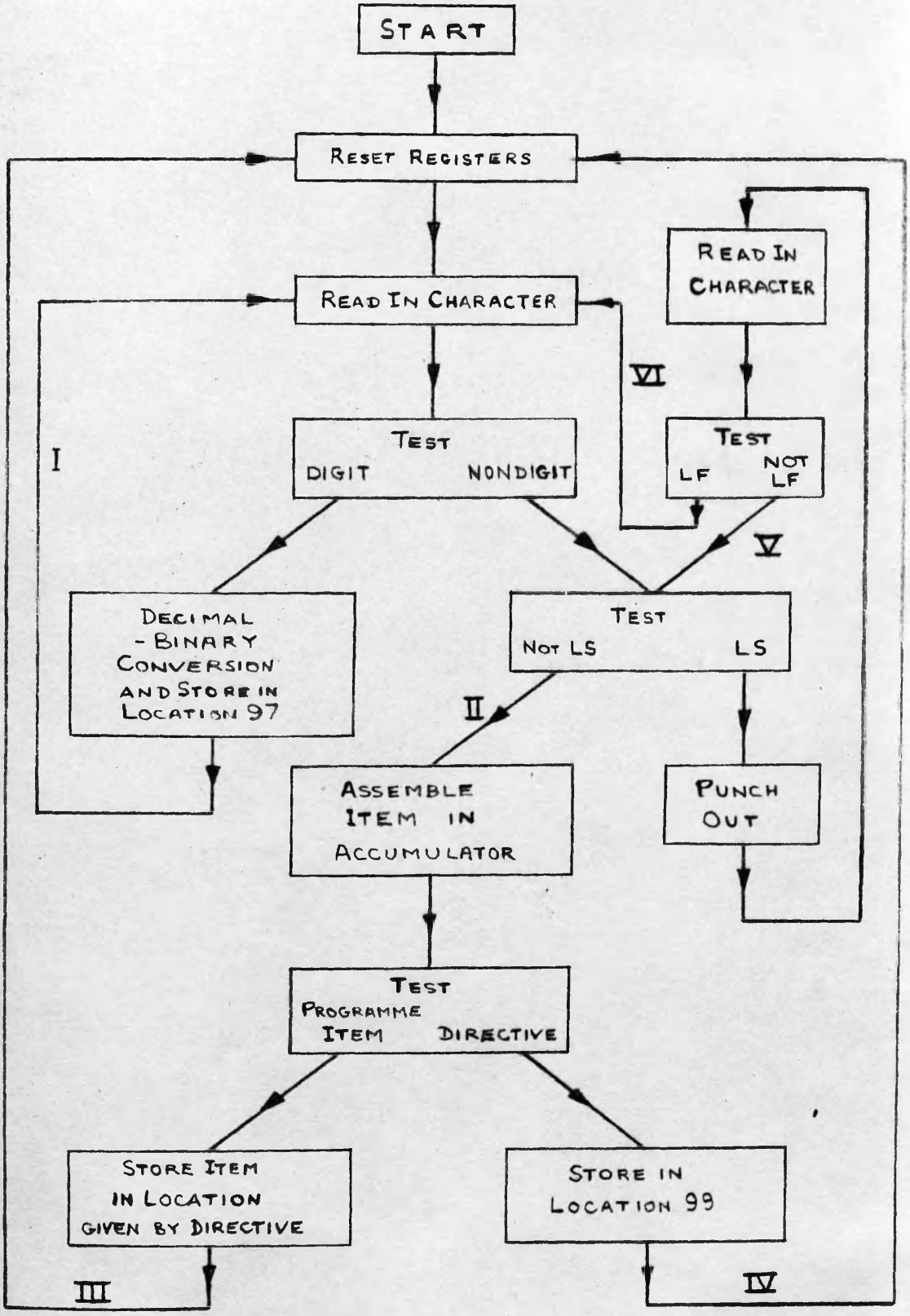


Fig. 59. INITIAL ORDERS - FLOW DIAGRAM