G I E R

REPRINT FROM INGENIØREN – INTERNATIONAL EDITION – VOLUME 5 DECEMBER 1961 NO. 4

P. HANSENS BOGTRYKKERI – KØBENHAVN

G I E R

Logical Organization

By T. Krarup, M. Sc., Geodetic Institute,
and Bj. Svejgård, M. Sc., University of Copenhagen

System and Circuit Design

By Henning Isaksson and Bent Scharøe Petersen,
Regnecentralen, Danish Institute for Computing Machinery,
Danish Academy of Technical Sciences

GIER

The article gives an account of the logical principles on which the electronic computer GIER is based. The command structure and the various automatic address modifications are reviewed, with the chief emphasis being laid on the points in which GIER differs from earlier computers. The operations carried out by the computer are listed in full.

The central unit of GIER is accommodated in the cabinet in the background. On the table, to the left is a reader for paper tape, in the middle the control panel, to the right a typewriter for input and output, and behind the typewriter a rapid perforator for output.

Logical Organization

By T. Krarup, M.Sc., Geodetic Institute, and
Bj. Svejgård, M.Sc., University of Copenhagen

518.5:621.38:681.14

The logical design of GIER (the Geodetic Institute's electronic computer) differs in various points from more conventional computers. The basic principles are of course well known, these being largely the same in all one-address computers. However, if a computer is provided with the necessary functions, and with these only, its programming will generally become a very troublesome affair. In GIER it has been attempted to arrive at a more convenient programming by making automatic a number of the coding principles which have been developed during the decade since the completion of the first electronic computer, and which have become purely ritual.

The computer is of medium size with 1024 cells in a magnetic core memory and 12800 cells on a magnetic drum. Each cell accomodates 40 bits plus two special bits serving particular purposes. The computer works in the parallel mode and its operating speed is roughly characterized by the addition time being about 50 μs and the multiplication time averaging about 180 μs. These figures apply to fixed point operations. The built-in floating point operations have about twice the addition time but the same multiplication time.

In the basic form the computer has its input from a punched paper tape reader, which reads 500 characters per second, and its output to a perforator which punches 150 characters per second. A typewriter permanently connected with the central unit can be used both for input and for output. However, technical provision has been made in the design for connecting up other external units such as magnetic tape and punched card units and recording pens. A Flexowriter (off line) serves for the production and transcription of punched paper tape.

As the computer is a purely binary one, the contents of each cell will be a configuration of ones and zeroes. Such a configuration is not ascribed any significance, a priori, this only becoming evident from the treatment to which the information contained in the cell is subjected by the programme. A cell can thus be used for the storage of information of any type, but there are certain types of information which are of special interest, and these will be reviewed in the following.

1) *Fixed point numbers.*

These are numbers within the limits $-1 \leq x < +1$. A positive fixed point number has 39 binary positions, which is to say that the least significant position receives a value of about 2×12^{-12}. The negative fixed point numbers are represented by their 2-complement. This results in positive numbers and zero getting a zero in their highest position, while the negative numbers have a one in the highest position. In the external language such a number is written as an ordinary decimal possibly followed by a power of ten $(-356.43_{10}-4)$.

2) *Floating point numbers.*

Any number other than zero can be written $p \times 2^q$, where $1 \leq p < 2$ or $-2 \leq p < -1$, p being termed the mantissa and q the exponent. We call the most significant position of a cell that of 0, and the least significant 39. Then p is now placed in the cell so that the binary point is located between position 11 and position 12. Positive p is represented by their binary equivalents. The mantissa thus occupies 30 positions of the cell. The exponent is similarly located in the 10 first positions of the cell. Thus q comes to lie in the interval $-512 \leq q \leq +511$, i.e. that the absolute value

of a floating point number must belong in the interval from about $0.74_{10} - 154$ to $1.34_{10} + 154$. In the external language such a number is written as an ordinary decimal number, with or without sign, possibly followed by a power of ten (or perhaps as a power of ten alone with or without the sign).

3) *Commands.*

A cell can contain one or two commands. In the first case it is said to be a fullword command, in the second, halfword commands. How an order is to be written in the external language is not fixed by the logical structure of the computer but is determined by the reading programme which translates the external language to internal representation. However, the external representation of commands as used in practice has a simple relation to the internal one. A fullword command consists of various elements of which the fundamental ones are an operation part and an address constant. The basic operation is designated by two capital letters which are a stenographic expression for the effect of the basic operation. Thus for instance AR means »add to result register«. (The result register is the part of the computer where the result of an arithmetic operation will normally stand when the operation has been performed). The address constant in its simplest form is a figure which indicates the number on the cell in the magnetic core memory from which an operative number must be obtained or to which it must be sent. However, with certain commands the address component has a somewhat different significance.

In its simplest form a command may, for instance, look like this

AR 136.

This means that the contents of cell number 136 (considered as a fixed point number) must be added to the momentary contents of the result register and the result thereof is to be located in the same place.

The basic operation designation is translated by the reading programme into a configuration of ones and zeroes comprising a total of 6 bits, which are placed in positions 20—25 of the cell. The presence of a one in position 26 is denoted in the external language as an S and involves resetting of the result register before the operation is performed. This applies to any operation whatsoever. The address constant is translated by the reading programme to its binary representation and placed in positions 0—9.

Before a command is carried out, its address constant can be subjected to various modifications. There is such a modification if we write for example AR r + 136. The meaning of this r, which in the internal representation is marked by a one in position 28 and a zero in position 29, is that the number on the cell in which the command stands is added to the address constant before the command is carried out. If the said command stands in cell 17, it will thus act as if it had been written AR 153. However, the command is not altered by this modification, so that its address constant continues to be 136. The significance of this modification is that it is not wished to indicate the absolute location of the operative element in storage, but only the relative location in relation to the command which refers to it. This has the great importance of allowing a programme to be written in such a way as to be independent of its location in storage.

In the same manner, for instance, we may of course write AR r - 56 which, if placed in cell 100, is synonymous with AR 44.

A corresponding modification of the address is obtained if we set position 28 = zero and position 29 = one. This modification is noted externally as s. Thus we may write, for instance, AR s+2 or ARS s - 78. The effect of this s is that the address is modified by the contents of a special register called the sequential register. The contents of this register are automatically altered during the execution of the programme when certain jump commands are carried out, but are constant within each part of the programme. As with the r-modification it holds good that the address constant is unchanged after the command is carried out.

If both position 28 and position 29 are equated to one, we have the so-called p-modification. Thus AR p - 45 means the same as AR 5 if the p-register contains 50. The p-register is a register whose contents are determined by the coder through the aid of special commands. Thus it is quite a normal index register (the only one in the computer) but in addition each cell in the magnetic core storage can act as index register.

The p-modification is of the same nature as the r- and s-modifications in that it does not involve any lasting change in the address constant.

The four forms of addressing mentioned here can be enclosed in parenthesis in the external language, which involves setting bit 27 equal to 1. The significance of this is that the address given does not become the address of the operative element, but on the other hand the address of an address. This again can be marked by parenthesis and so on until an address is found which is not in parenthesis. This so-called indirect marking has the advantage that it is possible to change an address in many commands by simply changing the address in a single cell and thereafter referring to this cell by the indirect marking.

If a command does not contain other elements than those already mentioned, it can find a place in a half cell. Two such commands placed in the same cell will be located so that the first occupies the positions 0—9 and 20—29, and the second the positions 10—19 and 30—39. The cell is shown to contain two halfword commands by setting position 40 equal to 1. If position 41 is also equated to 1, the command or commands in the cell having significance in connection with the floating point operations will be carried out in accordance therewith. This is shown in the external language by adding an F to the statement of the operation. For instance a cell content can look like this:

ARF p - 3, HV (r + 4)

This means that the computer, after having carried out the floating point addition to the floating point result register, must take the next command in the cell whose address stands in cell r+4.

If a whole cell is used for a command, there is yet another form of address modification possible which in contrast to those already mentioned is a lasting change, so that after being carried out the command actually has had its address constant changed in the storage.

If we write for example the command

AR 17 + 2

the first time the command is carried out it will have

the same effect as AR 19, after which it will be changed to AR 19+2. Next time it is carried out it will have the effect of AR 21 and be changed to AR 21+2, and so on. The constant, positive or negative, with which calculation is done is placed in positions 10—19. If the address is determined through a chain of parentheses, as described above, the computing takes place in the remotest element not marked by parenthesis.

In a fullword command the remaining positions fall into two groups. The positions 30—32, which in succession are designated X, V, D, bring about certain modifications of the basic operation. X results in the contents of the result register and those of the multiplier register exchanging places when the operation is performed. The multiplier register is the register where a multiplier is usually located during multiplication. The V-modification means that the contents of the next cell must be skipped. This modification results in the convenience that storage of an intermediate result can take place directly after the storage command which in such case is V-marked. It also finds application in connection with conditional commands whereby the possibility arises of conditioning commands of various logical functions. The significance of the D-modification is not quite the same with all basic operations but its fundamental meaning is that it makes the the case in most electronic computers, it is not only the jump commands which can be conditional.

A great proportion of the logic in a programme can be managed by means of a device called the indicator. This device, which is something peculiar to GIER, is a register which serves to keep information about overflow, zero situations, signs and markings (the two last positions in a cell). Each command can be made conditional to the momentary contents of this indicator in addition to allowing each command to be made conditional to the momentary contents of the result register. Thus it must be stressed that, unlike the case in most electronic computers, it is not only the jump commands which can be conditional.

The indicator contains 10 positions which fall into 5 pairs. One of the pairs serves to retain information on overflow and zero situation, one pair for keeping signs and three pairs for marking. These positions are designated in the stated order:

Oa, Ob, Ta, Tb, Pa, Pb, Qa, Qb, Ra, Rb.

The performance of an order can now be made dependent of the contents of the indicator by the addition of for instance LPA. The command will then, and only then, be carried out when Pa contains a one. LPB allows the command to be performed when Pb = 1, and LPC (= LPAB) makes the command conditional upon Pa, Pb = 1,1. Similarly NPA means that the command shall be carried out for Pa = 0, NPC for Pa, Pb = 0,0. In a similar manner another indicator pair can be utilized.

Information can be led to the indicator in connection with any command by means of indicator operations which begin with and I. ITA means that Ta is made equal to 1, if the result of the operation just performed is negative, and equal to 0 if the result is positive.

ITB leads the information to Tb, and ITC to both Ta and Tb. Overflow is stored by analogous indicator commands in Oa and Ob, and marking in one of the last three pairs of indicator cells. Simultaneous with the cell contents being led to the arithmetic unit, the marking of a cell will be led to two special cells in the result register, though without taking part in any way in the arithmetical processes. The marking of the cell can be led in the same command to the indicator by commands like IPA, IPB, IPC, etc. The performance of a command can also be made conditional upon the contents of these two positions in the result register, through commands like NA, NB, LC, which are quite analogous with those commands whereby an operation is conditioned by sign and overflow, these being called for instance LT, NO, etc. When the contents of a register are stored in the storage, it is possible at the same time to decide the marking of the cell, either unconditional as with the commands M, MA, MB, MC which all give the cell marking indicated, or with commands like MTB which make the cell marking dependent upon the contents of one or a pair of the indicator cells.

The last 7 positions in a fullword command is now used to specify all these operations, positions 33—34 accomodating one of the operations I, M, N or L, positions 35—37 one of the adresses K, Z, O, T, P, Q or R. The two last positions 38—39 indicate A or B (and their combination C).

The three types of information enumerated up to now: fixed point numbers, floating point numbers and commands, are those whose form of storage is associated with the design of the computer. Two other types of information, the so-called groups of integers which allow storage of up to four parameters in a single cell, and text for which storage usually extends over several cells, have a form of storage which is solely determined by the reading programme.

The marking of cells described above has, as said before, its special task in connection with commands. The marking in cells not containing commands can be freely used by the coder for the most diversified of purposes. A frequent application will be for specifying various subdivisions of numerical material, whereby the coding can sometimes be facilitated considerably. A simple example may be visualized in the following assignment:

We are given a number of observations which fall into certain groups with a varying number of observations in the various groups. During the handling of these observations it is desired, among other things, to arrive at the sum of the observations in each group by itself.

This can be realized in the following way: During the input of the observations, the last observation in each group can be marked with an a, and the last observation in the whole material marked with both an a and a b. We can suppose that the observations are stored in consecutive cells starting in cell 100. The sums are desired to be stored in consecutive cells beginning in cell 1, and it is required that the last sum be b-marked. The following piece of programme will then solve the problem presented

```
AR     99      IPC   +1
HV   3-1        NA
GR     0       MPB   +1
HVS  r-3       NPB
next command
```

assuming that the result register contains zero when the programme is entered.

When the first command is carried out for the first time, the contents of cell 100 is added in the result register and the command is changed to AR 100 IPC +1. At the same time the marking of cell 100 is led to the two marking positions in the result register and an indication of this marking occurs. The next command is carried out if the cell just called for is not marked with an a, i.e. the first command is jumped back to and carried out again, having its address increased by 1. This is continued until the computer comes to an a-marked cell. When this has occurred the the command HV r – 1 NA is skipped, and the next command stores the sum found in a cell with perhaps a b-marking. The fourth command is only performed if the last cell contents added are not b-marked. The command resets the result register, jumps back to the first command and the process begins again from the start until a new a-marked cell is reached and the sum will be stored in cell 2. It continues in this way until it reaches the last observation, which was of course both a- and b-marked. Thereafter the computer proceeds further into the programme. It is thus seen that it is quite unnecessary to provide the computer with information on how many groups of observations there are, or how much information is to be found in the individual groups. The same programme therefore can be applied towards a new series of observations of a different length and a different subdivision into groups. This series of observations can be located elsewhere and the sums desired perhaps also stored in another place. To indicate the start address, the command PA is used (see list of operations).

If the list of operations is examined, those who are familiar with other computers will notice the absence of special commands for right and left shift. The explanation of this is that the shift commands (TK, TL, CK, CL) with the sign for their address constant indicate the direction of the shift. Thus TK-6 means for instance that the contents of the result register shall be shifted 6 places to the right, i.e. must be multiplied by 2^{-6}.

Among the many jump commands there are two particularly interesting ones, namely the commands HS and HR. These are used with a jump to subroutines (HS) and with the return jump (HR) from these. During this, the previously mentioned sequential register comes into application, since during the performance of an HS-command among the things happening is that the contents of the sequential register are transferred to the HS-command counting constant whilst the address on the jump command is transferred to the sequential register. During the performance of an HR-command, the counting constant of the cell whose address stands in the sequential register is transferred to the sequential register. Hereby a fully automatic administration is realized of subroutines in any number of levels.

In arithmetic GIER also exhibits features which serve to facilitate coding. Thus if an overflow arises with an arithmetical operation, the sign for the result will be correct, and the half result can be obtained by a normalization or a right shift of one place. A characteristic peculiar to GIER normalization commands is that they can be effected not only to the left but also to the right. The latter is particularly of interest with an overflow. (To normalize a number in a binary computer will mean to multiply it by such a positive power of two as to make the number lie in one of the two intervals $-1 \leq x < \frac{1}{2}$ and $+\frac{1}{2} \leq x < +1$).

As regards division, it is always possible to divide with a normalized number. Thus, unlike the case in most computers, it is not necessary with division of, and by fixed point numbers to require that the dividend must be smaller numerically than the divisor.

The two multiplication commands ML and MK can carry out accumulating multiplications. ML is designed like DL, in such a way that calculation with integers is a directly accessible process.

It can generally be said about the command structure that though at first glance it may seem somewhat complicated, yet it often makes possible programmes which are far simpler than the corresponding ones in a computer with a less advanced programming system, since it is possible to a great extent to carry out the logical administration of the programme simultaneously with the actual calculations.

In this article it has naturally been possible to present only the more characteristic features about GIER. A detailed review of the command system will be outside the scope of the article and those interested therein must be referred to »Lærebog i kodning af GIER« (Textbook in coding of GIER), published by Regnecentralen (Computer Centre).

0 9	10 39	40	41
Exponent	Mantissa		

Placing of floating point numbers in cell.

0 9	10 19	20 25	26 32	33 39	40	41
Address constant	Counting constant	Operation	S ( ) r s X V D	Indicator part	0	F

Representation of fullword commands.

0 9	10 19	20 25	26 29	30 35	36 39	40	41
Address constant 1	Address constant 2	Operation 1	S ( ) r s	Operation 2	S ( ) r s	1	F

Representation of halfword commands.

LIST OF OPERATIONS FOR GIER

BASIC OPERATIONS

Addition

AR	(F)	Add to R
AN	(F)	Add abs. to R
AC		Add R to cell
AB		Add Boolean

Multiplication

MK	(F)	Multiplication, short
ML		Multiplication, long
MT		Multiplication with cell sign
MB		Multiplication, Boolean

Placing in register

PM	Place in M.
*PI	Place result. addr. in indicator with the command counting constant as a pattern
*PP	Place result. addr. in p-reg.
*PS	Place result. addr. in s-reg.

Insertion

PA	Insert command counting constant in cell address constant
PT	Insert command counting constant in cell counting constant

Jump

*HV	Jump to left halfword
*HH	Jump to right halfword

Modifications

S : R is reset before basic operation is executed
X : R and M exchanged after basic operation is executed
V : Next whole cell skipped
D : Operation with address constant, shortening of parenthesis chains. The basic operations marked * are automatic D-modifications
F : The 9 commands thus marked can operate with floating point numbers

Indicator

(Ka Kb)
Oa Ob Ta Tb Pa Pb Qa Qb Ra Rb

Indicator operations

I : Indicating of marking (PQR), overflow and zero situation (O) and sign (T).
N : Command only executed of the stated indicator orbit is 0.
L : Command only executed if the stated indicator orbit is 1.
M : The stated cell is marked, either definitely or dependent upon the indicator.

Useful commands

TK 10 F : Floating point numbers in R_F to fixed point numbers in R
NK 0 F : Fixed point numbers in R to floating point numbers in R_F

Subtraction

SR	(F)	Subtract from R
SN	(F)	Subtract abs. from R
SC		Subtract R from cell

Division

DK	(F)	Division, short
DL		Division, long

Shift

NK	(F)	Normalization, short
NL		Normalization, long
*TK	(F)	Number shift, short
*TL		Number shift, long
*CK		Cyclical shift, short
*CL		Cyclical shift, long

Store register

GR	(F)	Store R
GM		Store M
GA		Store R's addr. constant in the cell addr. constant
GT		Store R's counting constant in cell counting constant
GI		Store in. in cell address constant
GK		Store in. in cell address constant and track reg. in cell

Auxiliary commands

QQ	No basic op. Modif. executed
ZQ	Stop. Modif. executed
XR	M and R exchanged.

Satellite commands

*IS	Use as s-value. X, V mod. not effective
*IT	Use as counting constant. X, V mod. not effective

Conditional commands

*BS	Conditional without counting	if resulting addr. of command ≤ counting constant, then skip following right half cell.
*BT	Conditional with counting	

Coincidence

CM	If coincidence	then skip following right half cell
*CA	If not coincidence	

Execute

UH	Execute now with s and r here
UD	Execute now with s and r there

Contact with external units

*VY	Select external unit
*SY	Write with external unit (Res. addr. Pos. 3—9)
LY	Read from external unit to storage and R.

Drum commands

*VK	Select drum track
SK	Write on drum track from storage
LK	Read from drum track to storage.

LIST OF OPERATIONTIMES FOR GIER

The different parts of the operation are executed in the following sequence:

	Time in μsec.
1. *Indicatorcondition checked.*	
a) Condition not fullfilled-time spent	15
b) Condition fullfilled	0
2. *Calculation of resulting address.*	
Minimum (including r-, s- or p-marking) .	27
If the countingpart is not zero ..... add.	9
If indirect marking but not s-marking add.	12
If indirekt marking and s-marking ... add.	26
3. *S-modification*	0
4. *Basic operation* ..... add.	see list below
5. *Indicatoroperation and V-modification* ...	0
6. *X-modification* ..... add.	4

The executiontime of section 4. depends on the basic operation in the following manner:

Operation	Time in μsec. Fixed point	Floating point
QQ—PS—PP—VK*	2	—
GR—GM—GI—GA—GT—GK—PM—PA—PT—BS—BT—UH—UD—XR—LK*—SK*—HV—HH—HK	9	only GR:9
IS—IT—NS—NT—PI—VY—MB—HS	16	—
AR—SR—AN—SN—AC—SC—MT—CA—HR	22	66 + 2.2 × exponent-difference + norm-alisationshifts
AB—CM	27	—
TK—TL—NK—NL—CK—CL	20 + 2.2 pr. shift	30 + 2.2 pr. shift
MK—ML	155 average	140 average
DK—DL	240 average	190 average

*) In case the drum is free, i. e. earlier drum operation finished.

System and Circuit Design

by

Henning Isaksson and Bent Scharøe Petersen,

Regnecentralen, Danish Institute for Computing Machinery,

Danish Academy of Technical Sciences.

The article outlines the conception of the GIER scheme and the considerations leading to the elaboration of the register system. A review is also given of the circuit technique employed, the principles governing dimensioning and practical construction features. Finally the production of a small number of copies of GIER is mentioned.

518.5:621.38:681.14

Introduction.

GIER is an abbreviation of Geodætisk Instituts Elektroniske Regnemaskine (Geodetic Institute Electronic Computer) and is the first solely Danish designed transistorized computer. It has been built as the result of co-operation between the Danish Geodetic Institute — a Government Institute, and Regnecentralen — The Danish Institute for Computing Machinery, which is a non-profit making organisation established by the Danish Academy of Technical Sciences.

When the work on the computer actually started in the latter half of 1959, only limited funds were available. With the limited amount of money and the prices then current, it was decided that only a ferrite core storage of 1024 42-bit words could be used and consequently it was necessary to use a back-up drum. However, an interrupter device should be included to minimise access time to the drum. Further, it was evident that mechanical equipment such as drum and input/output devices for paper tape should be purchased, so that the work could be limited to the design of the electronic circuits.

Under the circumstances mentioned above, it was necessary to consider the advantages and disadvantages in the choice of the following:

1. Parallel or Series Machine.
2. Register Structure.
3. Word and order length.
4. Micro-programming system.

The first problem was solved easily. It was evident that the series machine was the cheaper but also slower in operation than the parallel machine (with the same type of components). Because experience gained earlier with a parallel tube machine (DASK), the parallel machine was chosen.

Register Structure.

Earlier experience had indicated that the order list of a machine is always subject to changes or at least wishes to do so. In the case of GIER, work could begin from scratch with new ideas that had not been tested before. Consequently, it was most reasonable to try to find a register structure that gave high flexibility together with good economy, i. e. to perform as many functions as possible with the same registers.

Functionally a computer can be regarded as a number of registers each of one word capacity. The content of any register can be transferred to any other register, usually via a processor unit, the basic processes being addition and shift. These needs can be fulfilled with the structure shown in Fig. 1.

Communication among the registers is via the common busline. The storages, which in principle consist of a large number of registers which can only transfer information and not process information, are connected via an address register and an information register. To make an information transfer, the address of the proper storage cell is transferred to the address register, and the information transfer now takes place between the selected cell and the busline via the information register.

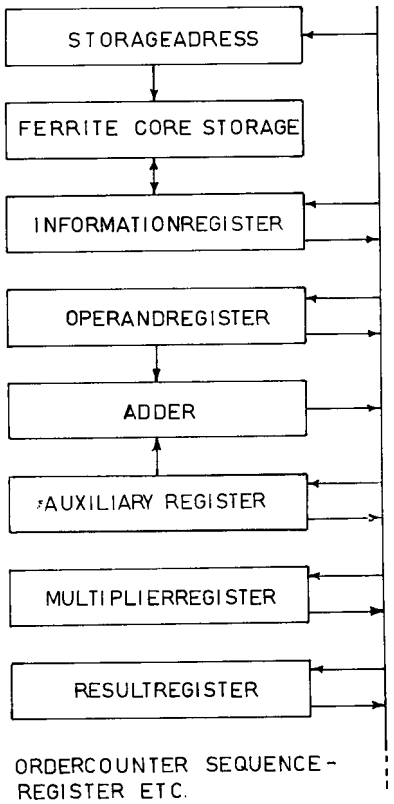

Fig. 1. Structure of the GIER.

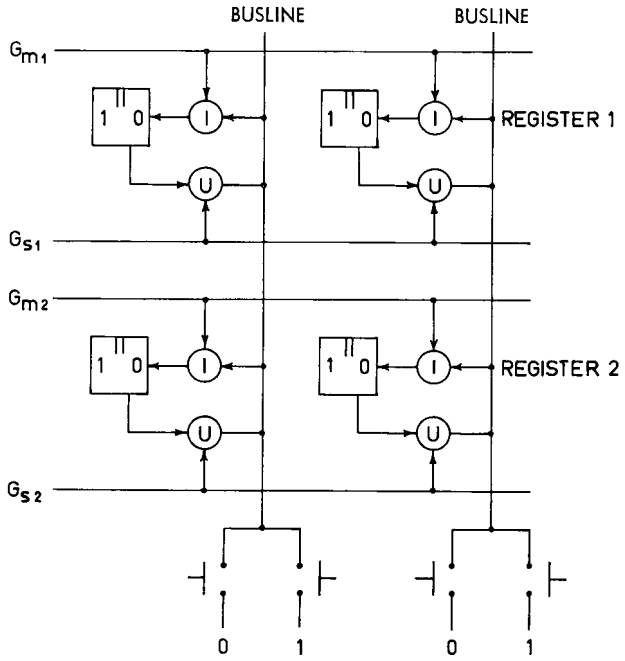

Fig. 2. Connection of the register elements to the transport lines.

If a process is to be executed, e. g. addition, the two addends are transferred to the operand register and the auxiliary register respectively, and in a following step, their sum is transferred over the busline to a register. The other basic process, shift, can be executed in the auxiliary register and in the multiplier/quotient register, in either one or the other or in both coupled together as one double length shift register, which is used e. g. in multiplication.

The register structure described above can be used in serial as well as parallel machines. In a parallel machine, the busline consists of the same number of wires as there are bits in the operands, i. e. in GIER, 42. Fig. 2 shows in a logical block diagram how two registers are coupled.

Every register element consists of a bistable multivibrator which, via an input gate, can receive information from the buswire. Likewise, information from the bistable multi-vibrator can be put out on the wire, via an output gate. To transfer from register 1 to register 2, a read out pulse G_s1 must be sent to register 1 and a receive pulse G_m2 must be sent to register 2. Usually there will only be one register sending and one register receiving simultaneously. In principle, however, there may be an arbitrary number but practical circuits will limit the number. The effect of simultaneous sending depends on the circuits and will form either the logical sum or the logical product of the contents of the sending registers. In GIER, the circuits are chosen so that the logical product is formed on the busline.

With the register structure mentioned, it is particulary easy to attach the control board. To indicate or set the content of a register, this register is sending and receiving simultaneously so that its content is not changed. An indicator on the wire will now show the bit value, and the register content can be changed by pressing the set or reset buttons which force the wire to a voltage corresponding to 1 and 0 respectively.

Word and Order Length.

The word length in a computer is determined by the accuracy with which it shall compute, and of how many orders shall be contained in a word. In GIER the word length was chosen to the "standard" value of 40 bits for operands. In addition, 2 bits are attached to the words for special indications. With this word length it is possible to obtain two one-address orders in a word. However, it was desired that an increment part should be had in addition to the address and some special indications, so at first, one order per word was chosen. Later it was found desirable to include a half-word order with only one address, so now GIER has both whole and half-word orders.

Micro-Programming.

As mentioned previously, it was highly essential that GIER should be very flexible and, realizing that essentially a control unit is a fixed store housing the micro-programmes that execute the actual orders, it was decided to build the control unit as a fixed wired store. The use of a normal ferrite core store for the control unit was considered but it was found that a fixed wired store would be far superior in speed and would also be much cheaper, so a store, built with linear transformers as shown in Fig. 3, was used.*

Fig. 3 shows only 3 time step generators (in all 24 are available) 6 micro-functions (180 in all) together with the coupling for three simple micro-programmes. For each micro-function, GIER has a linear transformer with its secondary winding coupled to an amplifier, which gives a pulse (e. g. send pulse) to the circuits when it is excited by a current in one of the primary single turn windings (Fig. 3 is drawn with the usual mirror symbols). Each of the primary wires corresponds to a certain time step in a certain micro-programme. The selection of a primary wire

* Wier: »A High-Speed Permanent Storage Device«. IRE Transactions on Electronic Computers vol. EC 4, pp. 16—20, 1955.

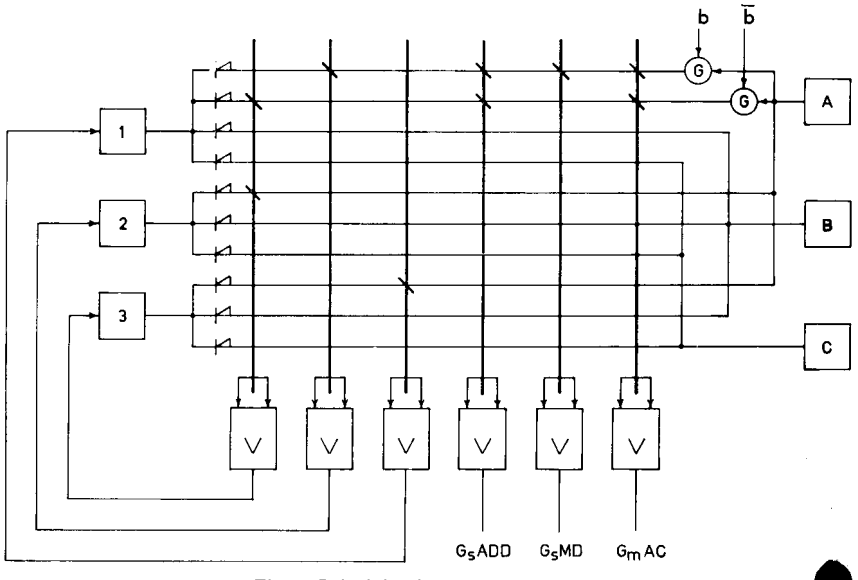

Fig. 3. Principle of the control unit.

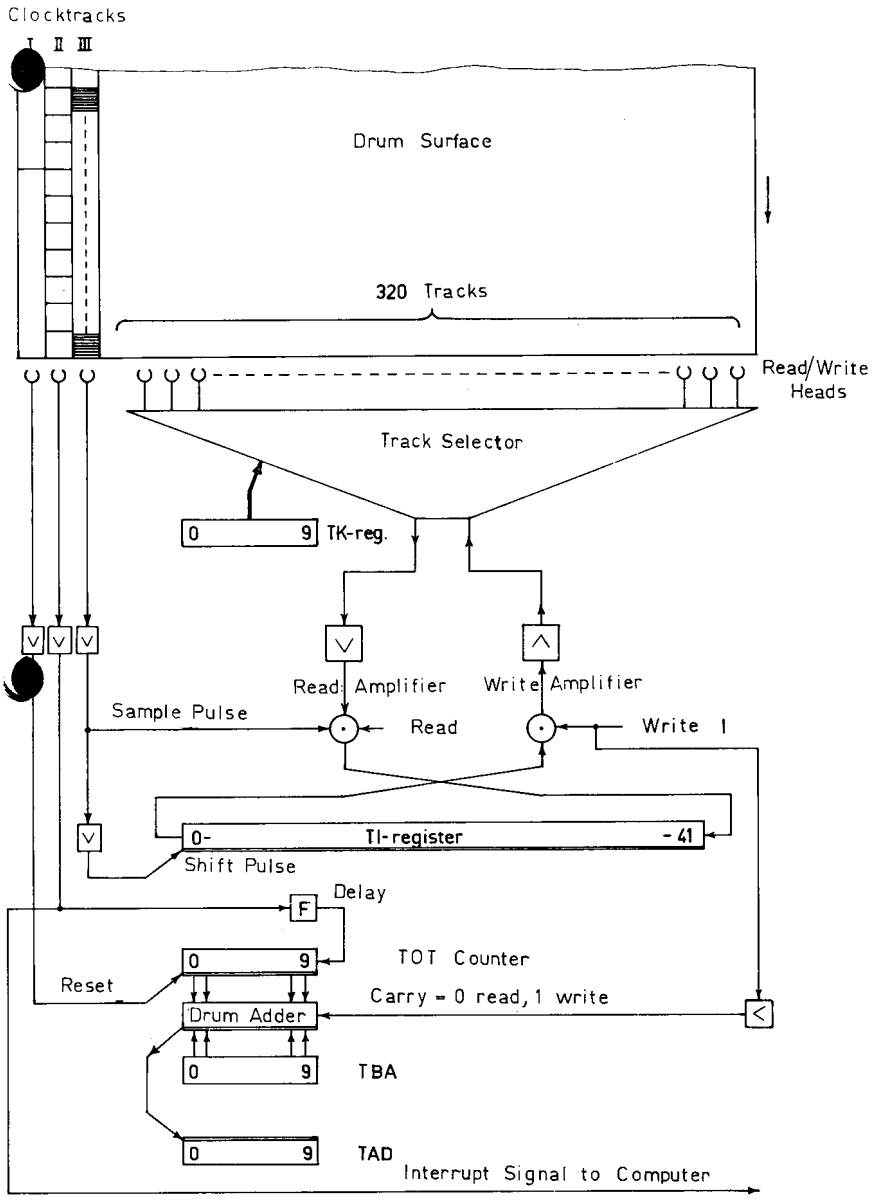

Fig. 4. Schematic diagram of the drum circuits.

is made by voltage coincidence between timing generators, 1, 2, 3 and micro-programme gates A, B, C, ... Conditions are introduced by steering the current between two wires by means of the condition gates G. For example, if b = 1 in time step 1 in micro-programme A, pulses G_sADD, G_sMD and G_mAC will be generated and the next time step will be 2. If b = 0 there will only be generated G_sADD and G_mAC and the next step will be 3.

Transfers between the Two Storage Levels.

As stated previously, GIER has a ferrite core store of 1024 words and a drum, which stores 12800 words. As stated previously, the ferrite core store is of the usual coincidence type. Information on the drum is stored serially with 40 words in each of 320 tracks. One revolution of the drum takes 20 mS. Transfer to and from the drum is carried out in blocks of 40 words = 1 track. To save time the transfers are carried out one word at a time in parallel interlaced with normal computer operation. It is only necessary to interrupt computation 26 μS per word transfered.

Fig. 4 shows the block diagram of the drum circuits. TI is a 42 bit shift register, which is connected to the busline for parallel transfers and to the drum for serial transfer writing on, or reading from the drum.

When a transfer to or from the drum is to be made, the number of the selected track is first transfered to the register TK. The content of TK is decoded by the track selector, which is essentially a voltage coincidence selector, and the head of the selected track is connected to the write or read amplifier.

If the drum is not available (earlier transfer order not yet finished) the central unit will wait till the drum is ready. The drum circuits will then proceed as above.

After a track selection order, an order to read or write must follow. The address of this order is the address of the first word of the block in the ferrite core storage. This address is transfered to the register TBA and then the central unit is free to proceed with the next order.

The clock-tracks on the drum contain one mark per revolution in I, one per word in II and one per digit in III. In addition to the registers mentioned above, the drum circuits contain the TOT-register, which is a counter recording the number of words passed since the reference mark on clock track I, so that the sum of TOT and TBA will be the address in the ferrite core storage to or from which actual transfer is to be made. When a word has been assembled in or written from TI, a request of access to the ferrite core store is sent to the priority circuit and as soon as the actual ferrite storage cycle is finished, the transfer between drum and store takes place. In this way the drum transfer only occupies the central unit 40 × 26.4 μS = 1056 μS during a drum revolution of 20 mS.

Circuit Technique in GIER.

The register elements in GIER are built around a conventional flip-flop circuit with transistors operating in the saturation mode. With the transistor types used, the clock-frequency could not be much higher than 500 kcs. In a few circuits, the use of an anti-saturation circuit was considered necessary. The logical circuits are transistor-diode-logic, but do not use any standard building blocks in the form of cards. However, the same circuits are used many times but the component values are chosen for the particular application. This reasoning implies that the number of different printed circuit cards is relatively high and that more work goes into the design, but the total number of components is reduced considerably and the circuits as a whole can be made to work faster. Some circuits utilize effects which are usually considered unwanted. Thus the hole-storage in diodes and transistors is utilized as are also the symmetrical properties of the transistors. The use of the transistor as a three-terminal device has been emphasized.

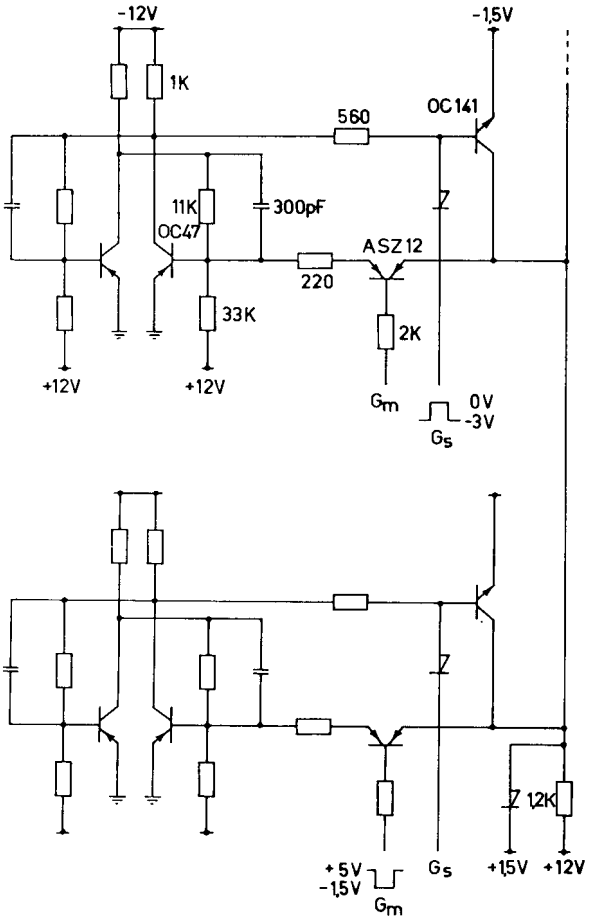

Fig. 5. Register elements with transmitter and receiver.

Design Principles.

All circuits are designed to cover the worst case, i. e. the circuits should work at simultaneous maximum deviation of all circuit parameters in the worst combination. Transistors are assumed to have a current amplification factor β (grounded emitter), which is only half the minimum value stated by the manufacturer, and the collector leakage current I_{CBO} is assumed to be twice the manufacturer's specified value at 45° C. In no case is the assumed β greater than 20. Diodes are assumed to increase the leakage current by a factor of 2 in the same way as transistors and the spread in the forward characteristics is also taken into account. Resistors are assumed to deviate ± 10 % from their nominal value. Finally, a variation in the voltages (regulated) of ± 5 % from the nominal value, is permitted.

Component Types.

GIER is built, to some extent, with complementary circuits, p-n-p and n-p-n transistors are used in a ratio 2 : 1. The most frequent types are Philips OC 47 and OC 141, which are alloyed junction types with alpha cut-off frequencies of 4.5 MCs and 9 MCs respectively, and with DC current amplification factors of 50 and 80 minimum. In addition to these types are used Philco 2N1754, a MADT transistor with alpha cut-off frequency of 70 MCs and current amplification 20. This type has a very small hole storage time constant. Diode types are mainly Philips OA 85 and Telefunken OA 160, point contact diodes, and for special purposes are used silicon junction diodes, germanium junction diodes and gold bonded diodes.

All semi-conductor components are tested before insertion in the circuits. Transistors are tested for current amplification and collector voltage drop at specified current. Diodes are tested for leakage and forward voltage drop. These tests have proved to be necessary even if "professional" components are used throughout.

Information Transfer System.

All information between registers is sent via 40 bus wires. In all about 500 register elements are used, all of which are coupled to their bus-wire with a send gate and a receive gate, which can transfer the content of the registerelement to the wire, respectively from it. In the neutral position, all send gates are maintained at − 3 V and all receive gates at + 5 V (see Fig. 5). The buswire is then at + 1.5 V. When sending, a G_s-pulse will open the send gate so, that if a 0 is sent, the readout transistor will force the buswire to − 1.5 V. In the receiving register element, the read-in transistor, which is symmetrical, will act as an emitter-follower with the collector on the buswire and the emitter coupled to the base at the flip-flop. Thus this is set to 0 and the symmetrical transistor does not saturate. If a 1 is sent, the wire is high, + 1.5 V and the symmetrical transistor now acts as a grounded emitter with the emitter on the wire and the collector to the base of the flip-flop, which is thus set to 1. The transistor in this case is driven into saturation and at the end of the gate pulse the stored charge is driven out through the collector and emitter, thus also functioning to set the flip-flop to 1. The line will remain at a voltage level corresponding to 1, and is not allowed to change until the transistor is out of saturation.

The Adder.

The Adder, Fig. 6, consists of three parts: A complementer, a sum digit and a carry circuit. The two first are built conventionally while the latter uses the

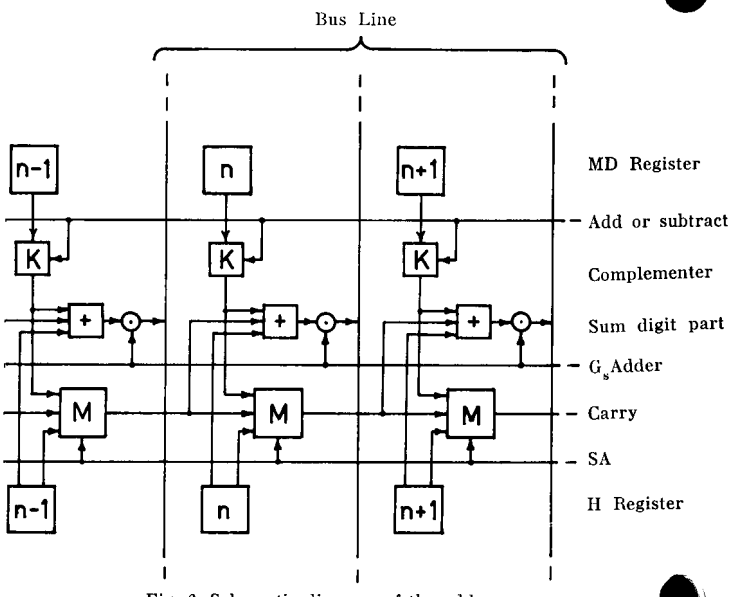

Fig. 6. Schematic diagram of the adder.

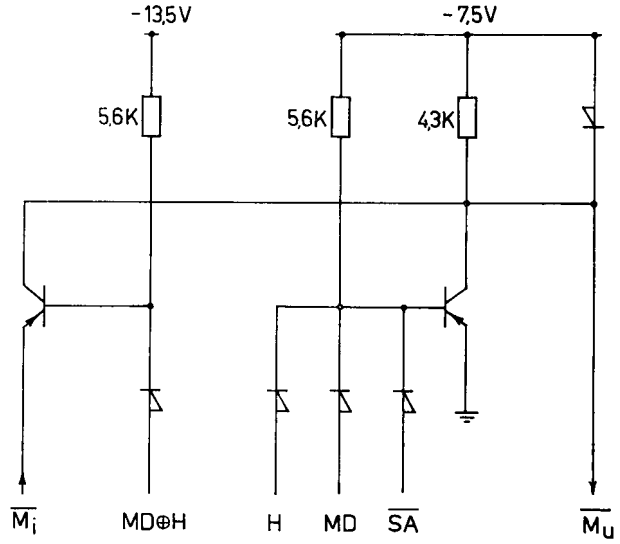

Fig. 7. The carry circuit of the adder.

transistor in a special way*. The adder forms continuously sum or difference of the operand register and the auxiliary register, and the result is read out by a send pulse to the adder-send gate.

The principle of the carry circuit is seen on Fig. 7. A carry is always formed when both H and MD are 1. A carry should be transmitted from the next lower position to the next higher position, when H and MD are different. If both are 0, the carry should not be transmitted, if they are both 1 it need not be transmitted, as a carry is generated directly. As can be seen from Fig. 7, a positive voltage on the terminal $\overline{M}_i$, corresponding to a carry from next lower position, will be transmitted when the function MD $\oplus$ H = $\overline{MD} \cdot H + MD \cdot \overline{H}$ is 1, i. e. negative. The carry will then be transmitted, and it is important that this does not happen when MD = H, because a carry generated in the right transistor of Fig. 7 would then not only be transmitted to the next higher position but also to the next lower, as the left transistor also conducts in reverse direction. The carry delay in the described circuit is essentially 0. The function MD $\oplus$ H is formed in all positions simultaneously while the pulse $\overline{SA}$ is positive, so that no carries are generated at all. In the positions where a subsequent carry should be transmitted, the base of the left transistor is charged by a base current flowing from the – 7.5 V clamp through collector and emitter to the base and through the base resistor to – 13.5 V. When the transistor bases are charged, the pulse SA becomes negative and the carries are generated. Carries will now be transmitted without delay, as a transistor with the base charged with holes will act as a closed contact as long as the current does not exceed the value corresponding to the base charge.

* Similar circuits have been developed by others: Edwards: »Parallel Addition in Digital Computers«, Proc. IEE, vol. 106B, pp 464—466, 1959. Salter: »High-Speed transistorized Adder for a Digital Computer«, IRE Transactions on Electronic Computers, vol. EC 9, pp. 461—464, 1960.

Control unit.

The control unit is built around a fixed wired store containing one wire for every microorder and one transformer (linear) for every micro-function. For every operation there is one operation transistor, the emitter of which is on – 4.5 V as shown in Fig. 10. From the collector a group of wires start, one for each micro-step in the operation. The wires pass through the cores, which are to be excited, and connect to a group of drivers (n-p-n emitter-follewers), which are exited by the clock-pulse G-prime, when the corresponding micro address flip-flop is set. One and only one of these is set. A wire is pulsed with a current of 30 mA, when its operation transistor and its micro address are selected. Conditions are introduced by means of the condition transistors, one of which is shown in Fig. 10. This transistor has its emitter on — 1.5 V, and will be open when both operation and condition are 1. As the emitter is at a higher potential than the emitter of the operation transistor, the current will run in the condition wire when the condition transistor is open.

The cores are coupled as current transformers, so that one wire can pass a great number of cores. However the capacity between wires necessitates a restriction in the micro programming, as only a limited number of wires may pass from one operation transistor through a core. The stray capacity otherwise will give rise to so high a current when the operationtransistor is opened, that unwanted outputs can come from the core. This diffuculty can be overcome by distributing on more operation transistors, which are opened at different times. The clock pulse is a symmetric square wave 450 kCs. There are 180 cores in the control unit.

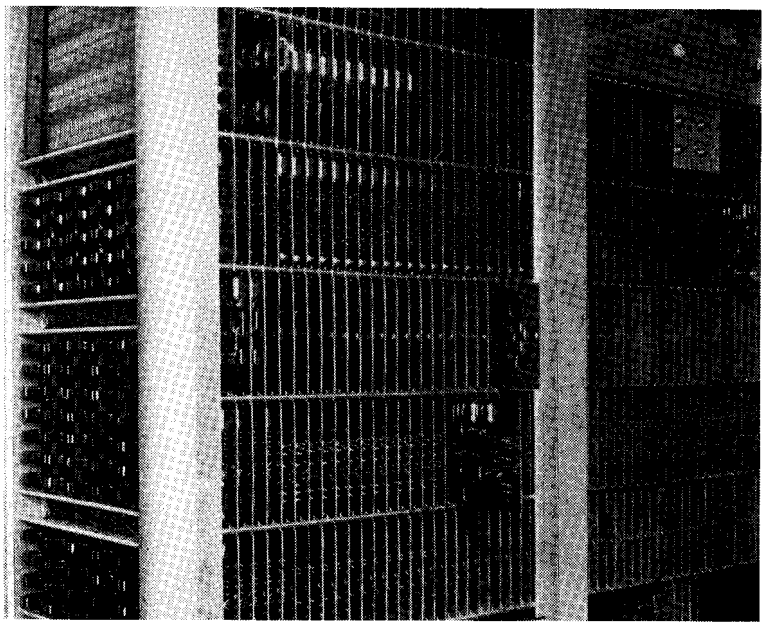
Fig. 8. The practical construction of the GIER central unit. The plates with printed circuits are fitted in drawers and provided with 30-pole plugs. At extreme left is seen the control unit with toroid cores and microprogramme wires. Above, in the control unit which is suspended on a hinge and can be swung out of the cabinet, is seen a plate with diodes for coding the wires chosen.

Gate Pulse Amplifier.

The secondary winding of the cores is connected to an amplifier, shown in Fig. 9. The amplifier is a two transistor amplifier with negative feedback through a diode. The

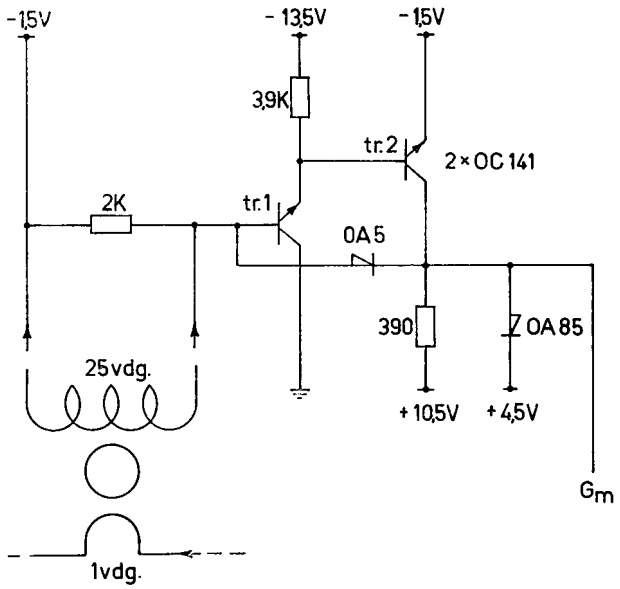

Fig. 9. Gatepulse amplifier for control unit.

first transistor functions as emitter follower, the second as grounded-emitter.

When the amplifier is non-excited, the transistor tr. 2 is closed, as the emitter voltage of tr. 1 is approximately 0·2 V more negative than the emitter of tr. 2. The feedbackdiode is also closed. When the amplifier is excited by a current pulse through the transformer, the base of tr. 1 becomes more positive until the emitter voltage of tr. 1 is so high that tr. 2 starts to conduct. tr. 2 is now driven hard towards saturation, as tr. 1 delivers a high base current to tr. 2. When the collector voltage of tr. 2 reaches approximately – 1.5 V the feedback-diode starts conduction and this will reduce the base current of tr. 1, until, at equilibrium, tr. 1 delivers just the necessary base-current of tr. 2, to keep the collector of tr. 2 at – 1.5 V + 0.2 V = – 1.3 V. The voltage drop over tr. 2 is thus just enough to keep tr. 2 out of saturation. In principle this coupling can be regarded as one transistor with very high current-amplification and no hole storage.*

Mechanical Construction of GIER.

GIER is built in a rack 180 cm × 160 cm × 40 cm. In one end of the rack the power supply is mounted (see Fig. 11). Instruments and knobs for the regulators are accessible from the end.

Approximately 400 printed circuit cards are used. These are connected via a 30 pole plug and are 20 cm × 12 cm. At the end opposite the plug there are 6 test points so that measurements can be made when the doors of the rack are opened.

The drum (made by Standard Electric Ltd.) is placed in the bottom of the rack, and the ferrite core stack (made by Philips Ltd.) is placed in the top of the rack. The control unit consists essentially of the transformers, which are mounted on a hinged plate

* The circuit was developed at Regnecentralen. It was later brought to our knowledge, that essentially the same circuit was developed by Blair and Harris of the Bell Laboratories, US patent 2.887.542.

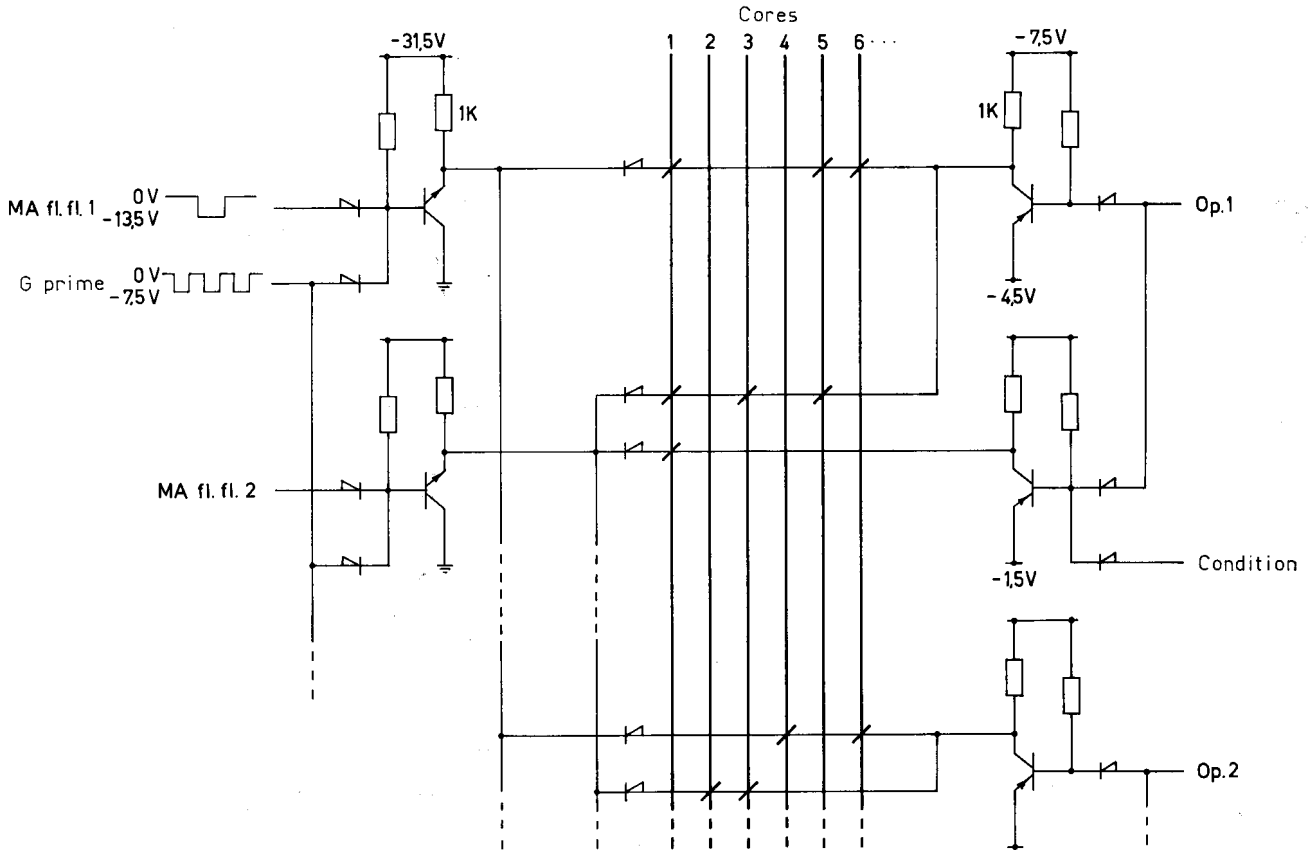

Fig. 10. Operating circuit for control unit.

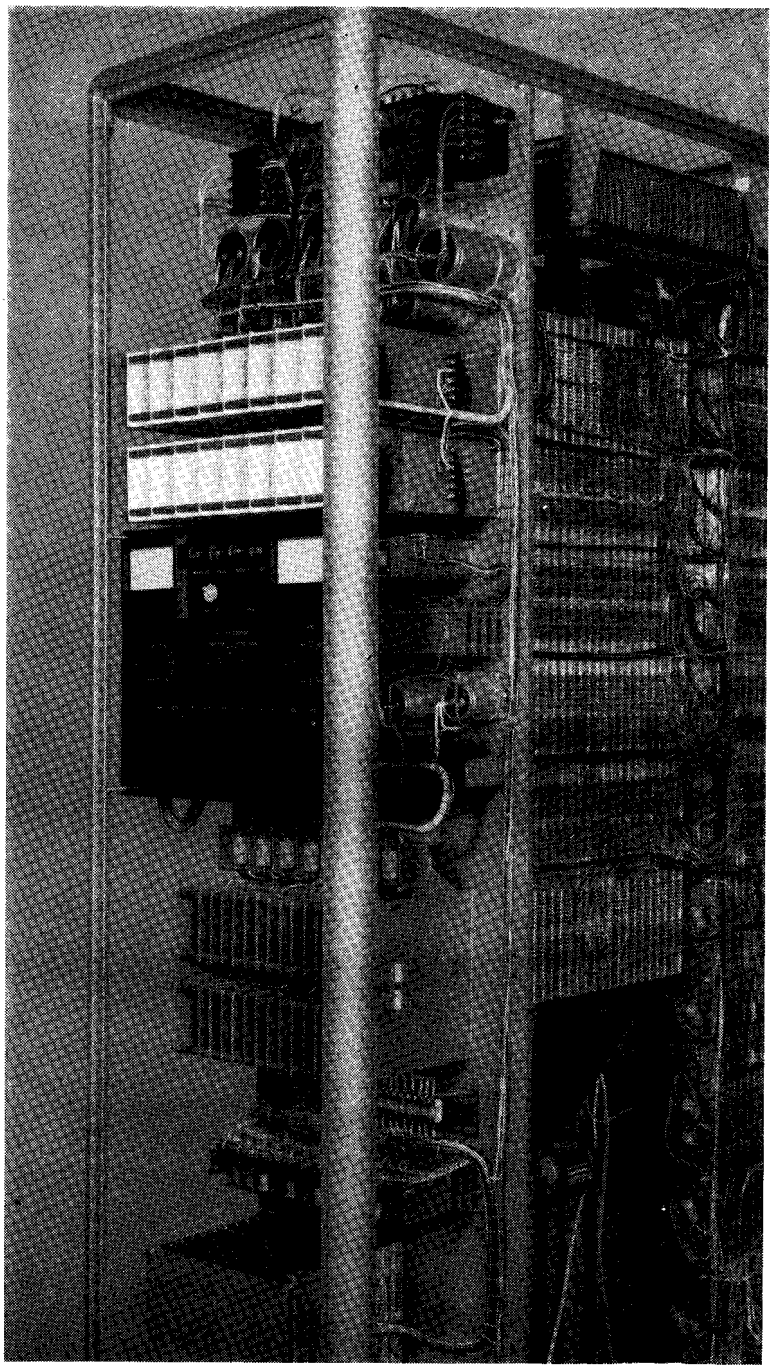

Fig. 11. Current supply is located at one end of the frame. Above are the contactors which cut in the 10 regulated voltages to the central unit. Below these are decouplers, regulating transistors on cooling finns, control panel for current supply, rectifiers on cooling finns control relays, plates with voltage references, and circuits belonging to the electronic excess current cutout. At the bottom is the supply transformer with mains contactors.

in the end opposite the power supply. The unit is connected with multi-pole plugs and can be replaced quite easily, if changes in the micro-programme are required.

The machine is cooled by forced air, which is blown into the bottom of the rack and passes up along the cards and escapes from the top of the rack. The power in the computer is 500 w and the air is heated 5—10° C passing through the machine. It is possible to use air at room temperature but up till now, it appears from practical experience that the life of the transistor can be prolonged by a factor of 2 per 10° C temperature decrease, therefore it was found to be economical to supply the machine with cooled air at 10° C.

The input/output devices are a high speed paper tape reader for 500 characters per second, a high speed perforator for 150 characters per second and a typewriter. (The tape devices are made by FACIT Electronics and the typewriter by IBM). The output devices are placed on a special table which also contains the associated circuits. The tape reader and the control panel are placed on a separate table.

Production.

At the outset it was hoped that the GIER prototype might be a basis for danish industrial production. To prove the producibility and obtain a few cheap computers for use at research laboratories, it was decided that Regnecentralen should build 8 reproductions of GIER. Four of these are to be delivered to other research institutes and four to be used by Regnecentralen in establishing a computer service at the Technical University in Copenhagen and at Aarhus University. Whether there is an industrial future for GIER is not yet known.

Acknowledgements.

The authors whish to acknowledge the valuable help of all who have contributed to the development of GIER, too numerous to name individually but special mention should be made of the fact, that the initiative to build GIER came from the Director of the Danish Geodetic Institute, professor, dr. phil. *Einar Andersen* and Messrs. *T. Krarup* and *B. Svejgaard,* both of the Geodetic Institute, who also designed the order list. Finally the authors want to thank their colleagues, especially *Kurt Henrik Andersen, Per Pedersen* and *Henning Worsøe* for their contributions to the circuit design and *Tage Vejlø* for practical production of the prototype and the reproductions.