Chapter 9

The design philosophy of Pegasus, a quantity-production computer[1]

W. S. Elliott / C. E. Owen / C. H. Devonald
B. G. Maudsley

Summary The paper gives an historical account of the development of the packaged method of construction of computers, and the advantages of this method are discussed. The packages used in the computer Pegasus are described from both an electronic and a mechanical point of view. The specification of the machine is given and the arguments which led to this specification are discussed. The detailed logical design procedure leading from the specification to the wiring lists is described. The method of maintenance and some reliability figures are given.

Introduction

The development of standard plug-in unit circuits ('packages') for digital computers began in this country [England] in 1947, and some of the advantages of the method have been discussed in earlier papers [Elliott, 1951; Johnston, 1952; Elliott et al., 1952; Elliott et al., 1953]. The advantages start in the design stage of a new computer project and follow through production and commissioning to maintenance.

In the design stage, what is known as 'logical' design is separated from engineering design. Once the packages have been designed by electronic engineers and the rules for their interconnection have been laid down, the 'logical designers' (usually, but not necessarily, mathematicians) can begin organizing the packages into various computers to carry out different functional requirements. The electronic and mechanical design work invested in the packages is thus drawn on for more than one computer design, and each computer can be assembled from stock parts without further engineering effort. Design time and cost are therefore much reduced.

In production, whether we consider one design of computer or several designs using the same packages, costs and time are also much reduced. Quantity production lines for the relatively few types of standard package are set up, and are common to different computer designs, thus reducing inspection and planning costs. Standard cabinet work has been designed for Pegasus, and this too can be taken from stock or established production lines to make other computers.

In commissioning a computer, because all the packages have been pretested, when power is first applied to the complete machine it is known that a large part is already fault-free. It remains to detect a few errors which may have been made in the interconnections.

Perhaps an even more important consideration is ease and speed of maintenance. Test programmes will usually indicate the part of the machine in which a fault is occurring. Several monitor sockets are located on the front of each package, and by inspection the faulty package is speedily found and replaced.

The package method has been criticized on the grounds of the cost and questionable reliability of plugs and sockets, and some redundancy of components.

The authors believe that the many advantages far outweigh the cost of plugs and sockets. The present trend is to use copper-etched printed circuits, and these fall naturally into the plug-in unit idea, the plug contacts being part of the printed wiring; there has been no trouble in Pegasus from plugs and sockets. Component redundancy in Pegasus is about 10% of the diodes and a few resistors, the cost of redundant components being about £150.

Electrical design of the packages

Circuits used for arithmetic and switching operations

Historical. A previous data-processing machine [Elliott et al., 1952; Elliott et al., 1956b] used 330 kc/s serial-digital circuits; they had originally been designed for 1 Mc/s operation, but 330 kc/s was chosen to suit an anticipation-pulse cathode-ray-tube store. This frequency has been retained to the present time because it suits the magnetostriction delay-line store [Fairclough, 1956] and the magnetic-drum store [Merry and Maudsley, 1956]. Experience with the data processor led to work (commenced in 1951) on a new set of circuits [Elliott et al., 1952], particular emphasis being

[1] *Proc. IEE*, pt. B, vol. 103, supp. 2, pp. 188–196, 1956.

laid on flexibility of use and ability to work without error in high electrical interference fields. These circuits form the basis of those in Pegasus.

Operations to be carried out. The following well-known operations are used to build up the logical structure of the computer:

- *a* '*And.*' This operation, which may be carried out between two or more input serial trains of pulses, produces an output train in which pulses occur only when pulses are present at the same time on all inputs.
- *b* '*Or.*' This operation produces an output train in which pulses occur at all times when a pulse is present on any of a number of inputs.
- *c* '*Not.*' 1's are changed into 0's and 0's into 1's; this is achieved by inverting the pulse train.
- *d* *Digit Delay.* The passing of a pulse train through a digit delay produces a pulse train similar to the input, but each pulse is one pulse position later in timing and restandardized in shape.

All operations in the computer, including addition, subtraction, and staticizing, are carried out by combinations of these elements. There is no circuit specifically for addition, and there are, in general, no flip-flops such as are often used for staticiziug or storing a single digit. A similar philosophy was arrived at independently by the designers of SEAC and DYSEAC [Elbourne and Witt, 1953], but the detailed working out is considerably different.

Digit waveforms. The timing of digit pulses throughout the machine is controlled by a common 'clock' waveform—a 3 microsec square wave (Fig. 1*a*) in which the positive-going portions define digit positions.

The digit pulses, which are routed about the machine and applied to logical circuits, are generally of the form shown in Fig. 1*b*; as generated, they have their leading edges well in advance of the clock pulse and are of a greater amplitude. This means that considerable distortion of the pulse is tolerable, since only the portion which coincides with positive clock pulse is of consequence. Digit pulse trains are 'clocked' ('and' operation with clock) only at their entry into a storage system or into a digit-delay circuit.

Inverted pulses are also employed: as an illustration, consider the operation 'A and not B'. Pulses A and B (Fig. 1) are on two lines and are of the same nominal timing, and we wish to form A . $\overline{B}$ (symbolic representation of 'A and not B'). To do this pulse B is inverted (forming $\overline{B}$, or 'not B') and is used to gate pulse A and prevent its passage. The inverted pulse $\overline{B}$ will be a little late on B, which also may have been later than A, as shown in Fig. 1*c*; thus when A and $\overline{B}$ are 'anded' together a spike may be produced, as shown in Fig. 1*e*. This spike, however, lies between clock pulses and so will be rejected on clocking.

The pulse system used allows several logical operations to be performed in cascade without any loss in nominal timing, so easing the problem of logical design (particularly by permitting afterthoughts). The maximum number of logical operations performed

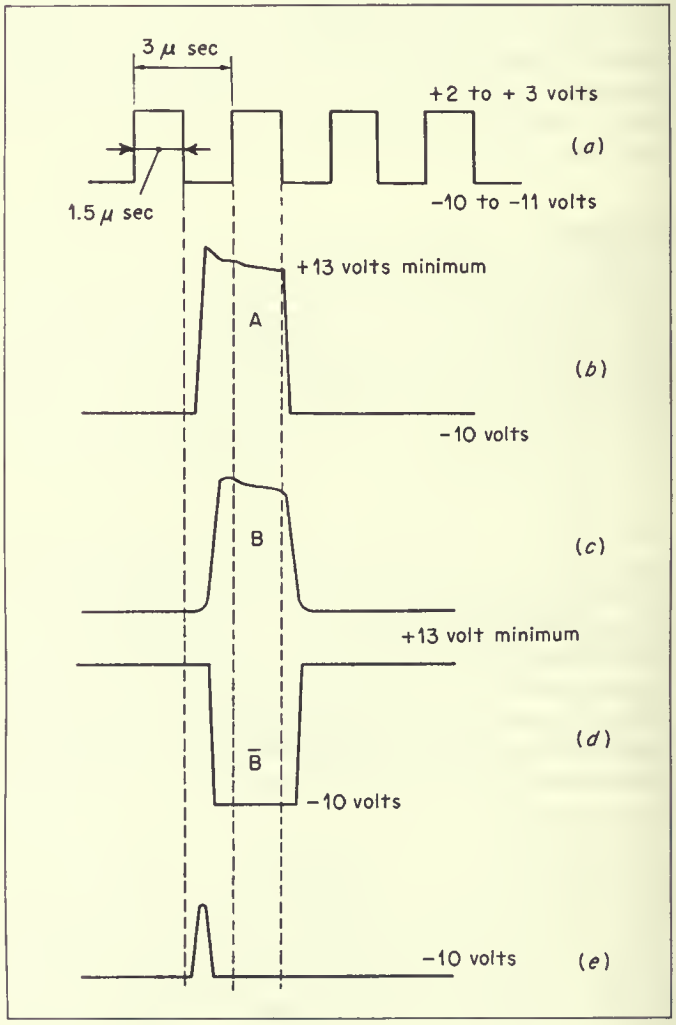

Fig. 1. Basic waveforms.

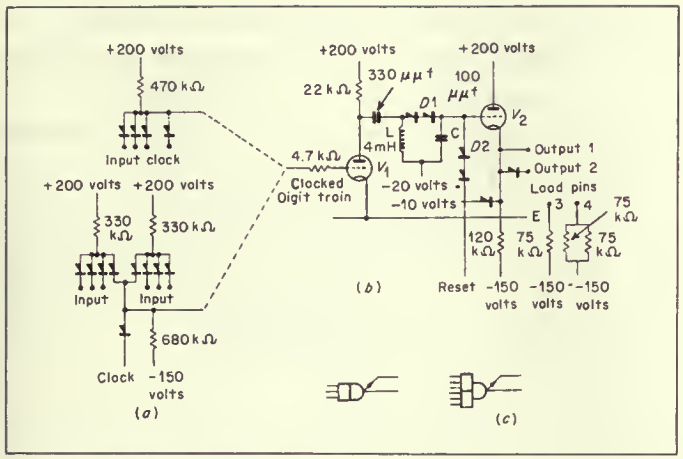

Fig. 2. Digit-delay circuit.

in cascade in Pegasus is five, though up to 12 could be performed in special circumstances.

The logical circuits. Each of the logical packages has more than one circuit unit. A circuit unit is defined as that part of a package which has input and output pins, and no connections to other parts of the package other than supplies. We may make the following generalizations:

- *a* Each unit has an 'and' gate at its input.
- *b* Each unit has a cathode-follower output (half a 12AT7 valve).
- *c* Each unit has an additional output via a germanium diode for making 'or' gate connections.

[Note: There are exceptions to (*a*) and (*c*) on one package type.]

There are three possibilities for the part of the circuit unit between the input 'and' gate and the output cathode-follower, namely a digit delay (half a 12AT7 valve), an inverter (half a 12AT7 valve), and a direct connection. Space does not permit a description of all the circuits, so it is proposed to deal only with the digit delay.

The circuit is shown in Fig. 2, and some typical waveforms are shown in Fig. 3. The input circuit can be of two forms, namely a 3-input 'and' gate and two such gates with their outputs 'or-ed' together. In both cases there is a further gating with a clock pulse. The clocked digits from the gate input circuit are applied to the grid of V_1, the anode voltage of which falls, so building up a current in L. When V_1 is cut off at the end of the digit, this current flows through diodes D_1 and charges up a storage condenser, C, which is discharged at the end of the next clock pulse by a 'reset' pulse applied through D_2. The reset pulse supply is a common computer supply whose amplitude and phasing relative to the clock pulse is shown in Fig. 3.

It will be noted that the reset pulse is also present at a time, just after V_1 is cut off, when the current in the inductor is about to charge the storage condenser. This merely has the effect of deferring the charging of C until the end of the reset pulse, the

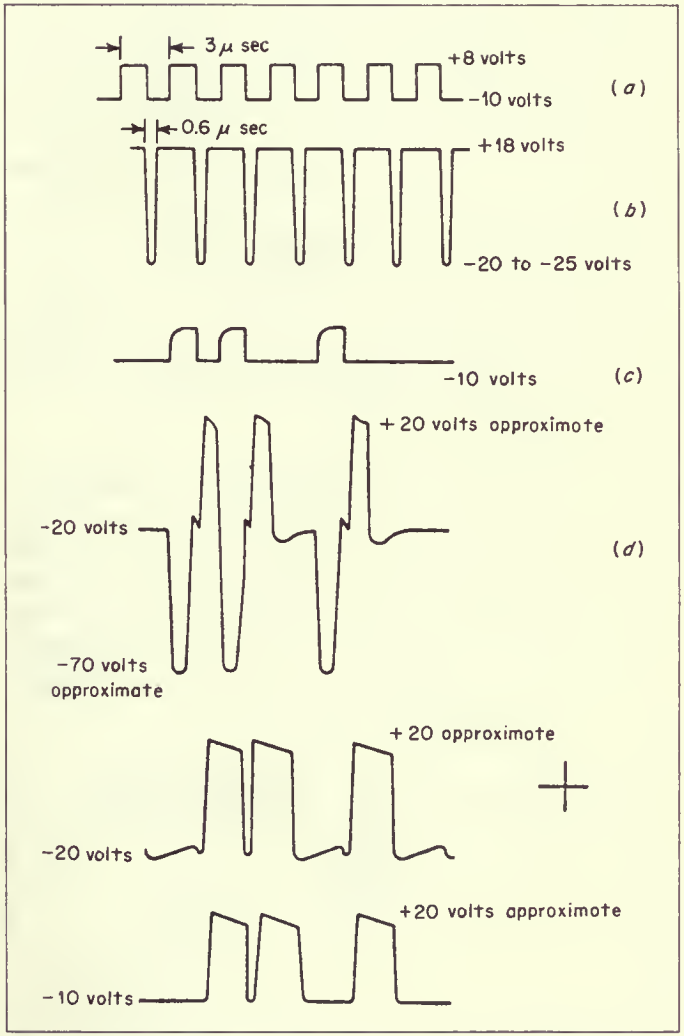

Fig. 3. Digit-delay waveforms.

current in the meantime continuing to flow through the diodes with little loss in the stored energy of L, since the voltage across L is low at this time.

The output cathode-follower V_2 is caught at -10 volts in the negative direction by a diode; this safeguards the crystal-diode circuits driven by it in the event of failure of the h.t. supply or V_2, and it removes residual ripple on the bottom of the input waveform, and thus reduces the back voltage and hence leakage in diodes of gates driven by the output.

The second output through a diode can be used in conjunction with similar outputs from other circuits and a resistor (pins 3 and 4) to make an 'or' (up to about 16-way).

In general, each output circuit has two available load resistors, disposed between direct and 'or' outputs according to a set of rules which are applied for each case. The number of units which can be driven by an output can vary between three and 16 according to circumstances; where more have to be driven than the rules allow, use is made of 'booster' cathode-followers available on one of the packages.

Some examples of the use of the logical circuits

Two examples will be given, the first being a simple arrangement—the staticizor—which is used frequently, and the second being a complicated arrangement—the adder/subtracter—which is used infrequently. The symbols used to indicate the circuit units are shown in Figs. 2*c* and 5*b*.

The staticizor. The function of a staticizor is to remember the fact that a digit occurred at a particular time, for an indefinite period, the method generally used in Pegasus being shown in Fig. 4. A digit delay with a twin 'and' gate input has its output connected to one of its inputs. It is turned on by gate 1, which causes a digit to circulate as long as the inputs to gate 2 remain positive.

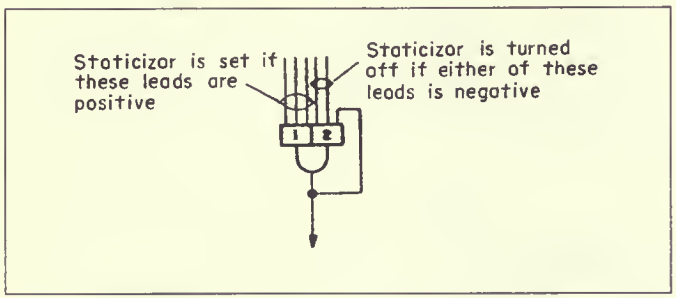

Fig. 4. The staticizor.

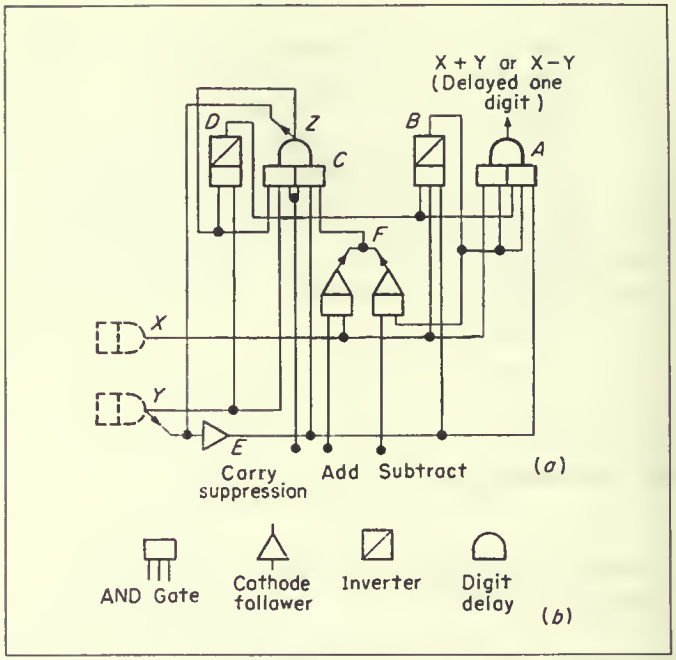

Fig. 5. The adder/subtracter.

It is normally turned off by an inverted pulse (a '0' following a series of 1's) on one of the gate 2 inputs.

The adder/subtracter. Figure 5 shows an adder/subtracter unit with inputs X and Y and an output $X + Y$ for the sum or $X - Y$ for the difference. There are two further input control leads marked 'add' and 'subtract'. If the 'add' lead is held positive while the 'subtract' lead is held negative, the unit acts as an adder. If the 'subtract' lead is held positive and the 'add' lead negative, the unit acts as a subtracter. Carry suppression is controlled by the lead marked 'carry suppression'. Carries are allowed to propagate when this lead is held positive, so that a negative signal on this lead will suppress carry.

Table 1 gives the digits appearing at the outputs of logical elements in the adder/subtracter unit for all combinations of input and carry digits when the unit is operating as an adder.

Arrangement of circuits based on packages

It was required to base the logical circuits on a standard size of package which could also be used for other circuits, e.g. a nickel-line 1-word store [Fairclough, 1956]. A unit which could accommodate three valves and had a 32-way plug was decided on; the

Table 1 Digits at various internal points of the adder/subtracter unit when set to add, for all combinations of the input and carry digits

Inputs digits		*Present carry digit*	*Digits at internal points*					
			A (*Sum*)	*B*	*C* (*Next carry*)	*D*	*E*	*F*
X	*Y*	*Z*						
0	0	0	0	1	0	1	0	0
0	0	1	1	1	0	1	1	0
0	1	0	1	1	0	1	1	0
0	1	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0	1
1	0	1	0	0	1	1	1	1
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	1	1

Note.—*A* and *C* are at the grids of the digit delay units.

problem then was to arrange the various circuits in such a way as to enable a computer to be designed using a minimum total number of packages without too many types. Five types were arrived at and these are shown in Fig. 6.

As an example of the factors involved, consider package types 1 and 2. The circuit units based on package type I can perform all the functions of those on type 2. However, there are many uses for a digit-delay circuit with a single 'and' gate input (package type 2), and since three units of this kind (instead of two for a 2- 'and'-gate input delay) can be based on one package, a saving can be effected. In Pegasus this saving amounts to 32 packages, which is considered to be well worth an extra package type.

In addition to the five logical packages, a further I6 types (three of which are peculiar to each computer) are required. The numbers used for the various functions are given below:

		Number
Logical types	Type 1	113
	Type 2	64
	Type 3	55
	Type 4	45
	Type 8	37
Nickel-line 1-word store		61
Drum-store packages (8 types)		38
Input/output packages (3 types)		17
Clock and reset waveforms (3 types)		14
Total		444

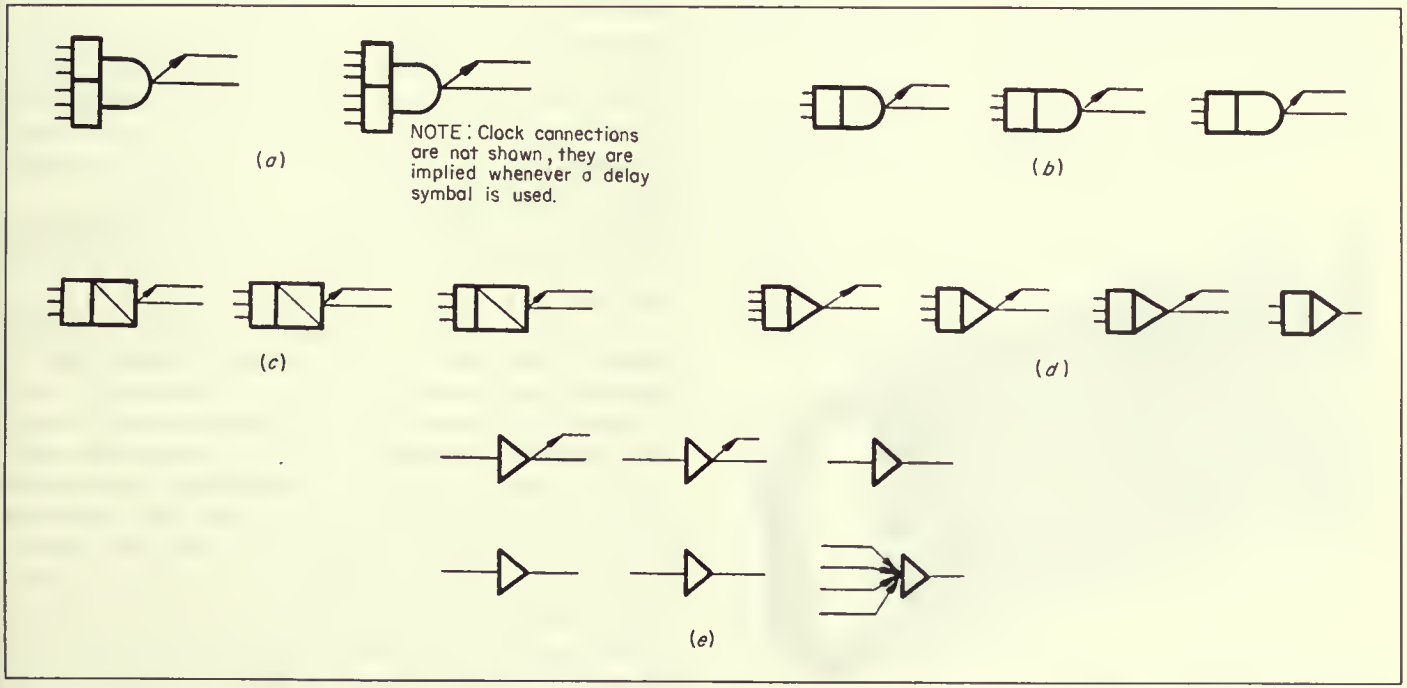

Fig. 6. Contents of logical packages. The arrowhead on an output lead denotes the presence of an OR crystal connection.

The magnetic-drum store and the circuit packages used with it are described in another paper [Merry and Maudsley, 1956], as is the nickel-line store [Fairclough, 1956].

The mechanical design of the packages

General form

Each standard package consists of three main parts, namely the valve panel, the component panel and the plug.

The valve panel is an aluminium pressing, there being three types—a 3-valve type, a 2-valve type and a blank. The package type number is marked on the panel by two dots according to the standard resistor colour code.

The component panel houses up to 100 components, including small transformers, chokes and coils, the panel and the handle being made in one piece from sheet insulating material. This design provides a minimum resistance to airflow over the valves and gives ample protection to the valves against accidental damage.

The plugs and sockets are used in multiples of eight connections. Most of the packages have four plugs providing 32 connections, but up to 64 are possible in each package. The plug contacts are made of brass and are heavily silver-plated. The socket uses a proprietary valve-holder contact, which can readily be replaced if damaged.

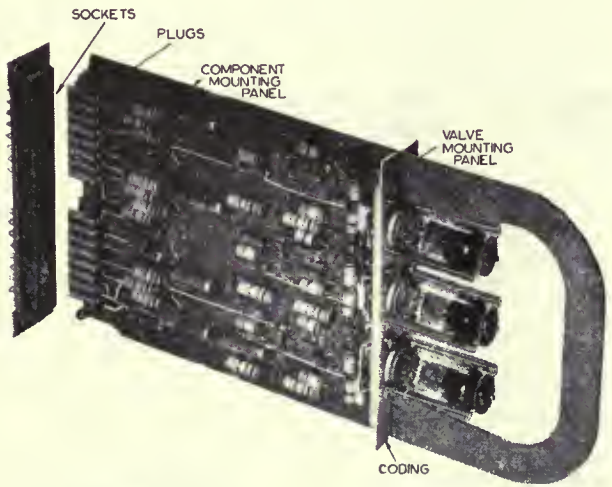

Fig. 7. Standard package.

This combination of plug and socket has a consistently low contact resistance (0.003 ohm at 1 amp); the insertion and withdrawal force is about 4 oz per contact.

The wiring of the packages

At present packages are wired and soldered by hand. The wiring is point-to-point, and within the limitations of layout for efficient performance, wire lengths are standardized for mass production on automatic wire-cutting and stripping machines. The symmetry of the eyelet positions makes it possible to use components which are preformed to a standard pitch and would allow for automatic preforming and insertion of components.

Experimental packages have been produced by photo-etched wiring and dip soldering.

Specification of the computer Pegasus

Summary specification

A detailed specification would cover the ground of the programming manual [Pegasus Programming Manual, Ferranti Ltd., London] and would be out of place here.

Pegasus is a binary serial-digital computer. The word length is 42 binary digits, of which 39 digits are used for a number and its sign (negative numbers are represented by their complements with respect to two), one digit is used for a parity check and the other two are gap digits. The length of an order is 19 binary digits, so that one word may consist of two orders, the remaining digit being a 'stop-go' digit. If the 'stop-go' digit is a '0', the computer will stop before obeying the orders in the word, but will proceed unhindered if the digit is a '1'.

There is a 2-level store, a magnetic drum holding 5120 words and an immediate-access or computing store of 55 single-word magnetostriction delay lines.

An order is made up of seven *N*-digits, three *X*-digits, six *F*-digits and three *M*-digits, the *N*-digits being the most significant and the *M*-digits the least significant. The *N*-digits allow 128 addresses in the immediate-access store (of which only 63 are used). The registers in this store are shown in Fig. 8. The *X*-digits refer to one of the accumulators, the registers corresponding to *N*-addresses 0–7. Thus the order code is a 2-address code with one address referring to only a limited part of the store. The *F*-digits indicate the function of the order. A list of functions and their corresponding *F* values are given in the appendix of this chapter. The *M*-digits indicate a modifier for the order: they select one of the accumulators, and the modification process is to add certain parts of the contents of the selected accumulator to the order before it is

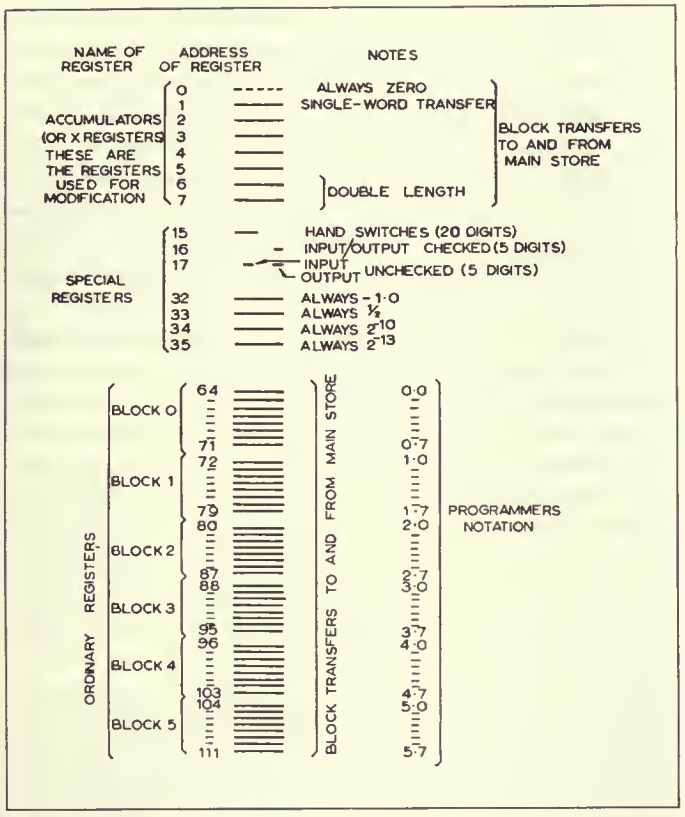

Fig. 8. Allocation of addresses in store.

obeyed, the part chosen depending on the function of the order to be modified. Figure 9 gives a schematic representation of the modification process. The effect of modifying an order depends on the function of the order and can be to make the effective order length 22 digits. This extension is necessary when specifying an address in the main store.

Transfers of information can take place between the computing store and the main store, and vice versa, either in single words or in blocks of eight words. For single-word transfers, only the register with address 1 in the computing store is involved. For block transfers the address on the drum of the first word of the block must be divisible by eight, and the registers in the computing store that are involved will be one of the discrete blocks indicated in Fig. 8.

Input and output is by means of punched paper tape. An 'external conditioning' order is included in the code to enable a choice of input and output equipment to be made. In the standard machine, two tape readers are used.

All stored information is checked (when read) by means of a parity digit, which is such that the total number of 1's in any correctly stored word is odd. The input and output of decimal characters on tape can be checked by a similar process.

The considerations which led to the specification and the logical design

The main features of the design are

a The use of a computing store from which all orders and numbers are taken while computing

b The provision of multiple accumulators

c The provision of special orders and facilities for dealing easily with 'red tape'[1]

The computing store. The use of a fast-access store from which all numbers and orders are taken increases the speed of the machine and eliminates the need for optimum programming. It is this computing store which makes it possible to use an inexpensive magnetic drum (with a relatively long access time) as the main store, and yet have a machine which is fast and relatively simple to programme. On the other hand, programmes have more 'red tape' and are not as simple as with single-level storage.

Transfer between levels is in blocks of eight words; this is a simplification and saves time. One block holds a reasonable amount of programme and other blocks hold data. Four blocks in all (32 words) would be just sufficient, and Pegasus was originally designed with this number. The design was subsequently modified to six blocks, which is quite adequate, in conjunction with the seven accumulators. Any further increase in the size of the computing store would be achieved by increasing the size, not the number, of blocks. As it is there is an economic balance between the usefulness and the cost of the computing store.

[1] 'Red tape' is an expression for the non-arithmetic orders in a programme.

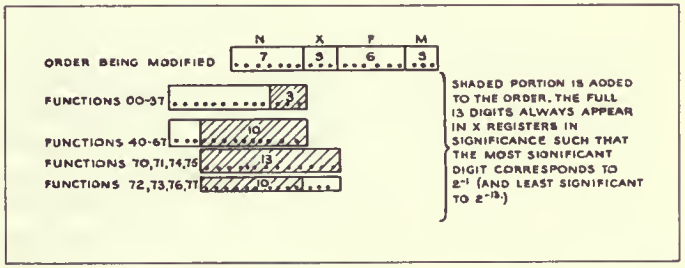

Fig. 9. Order-modification process.

The provision of several accumulators. This is the most novel feature of the logical design of Pegasus. It is generally agreed that the simplest order code from the user's aspect is the 3-address code with orders of the form, $A + B \rightarrow C$. An examination of this form of code, however, shows that in many cases two of the addresses are the same, so that the order takes the 2-address form, $A + B \rightarrow A$. A further examination shows that in a large proportion of cases the address A is confined to a very few addresses. This leads to the suggestion of a code of the form $N + X \rightarrow X$, where X covers only a small part of the store while N covers the whole store. This will have the advantage of yielding a reasonably short order. In Pegasus two such orders are incorporated in one word, leaving sufficient digits to specify a modification register (a Mancunian *B*-line) in each order.

The extreme case of this code is, of course, the single-address code, where X is confined to one address, the accumulator. However, experience had convinced the programmers collaborating in the design of Pegasus that, with single-address codes, a large number of orders are concerned solely with transfers of numbers from one register to another; the single accumulator is a restriction through which all numbers must pass and in which all operations have to be performed.

In the Manchester University computer the *B*-lines serve two very valuable but distinct purposes: they allow order modification and rudimentary arithmetic (such as counting) to be done without disturbing the accumulator. It was felt that fuller arithmetic and logical facilities on these *B*-lines would have been extremely valuable. The seven accumulators in Pegasus, used for modification and arithmetic, are a development of the *B*-line concept.

Special facilities for dealing with 'red tape'. The difficulties associated with the 2-level storage system have been greatly reduced by having an order-modification procedure which depends on the function of the order (Fig. 9). This method of modifying orders, used in conjunction with order 66 of the code (the unit-modify order), enables the counting through blocks of information to be done with relative ease.

The use of the group-4 orders of the code enables counters to be set conveniently and a constant (up to 127) to be placed in an accumulator, the constant being the value of the *N*-digits of the order. Order 67 (the unit-count order) enables the counting of cycles of operations to be dealt with in a simple way. A jump to another part of the programme can be programmed to take place automatically when the required number of cycles has been performed.

Having a large number of jump instructions greatly helps in organizing a programme. In particular, one order enables a jump to be made depending on the condition of an accumulator (being zero, for example), and another order on the complementary condition (being not zero). When only one of these orders is available it is necessary to think ahead to see whether or not the correct condition will be satisfied. Although the eight jump instructions included in the code were felt initially to be enough, it is now suggested by programmers that even more such orders would be helpful.

The logical shift orders, 52 and 53, are also included to simplify 'red tape'. In particular, they are used for packing and unpacking words holding several items of information.

As a result of including these various orders, the order code of Pegasus is quite large. It is worth remarking, however, that by a sensible grouping of the orders in the code the remembering of the code is a very simple task. A sensible arrangement of the code tends to reduce the amount of equipment needed to engineer it. For example, when the equipment for dealing with group 0 of the code has been allocated, groups 1 and 4 require the addition of only three gates.

Facilities for checking programmes. The features mentioned above make the computer easier to programme, and there are other facilities in Pegasus that make it easier to check out and develop new programmes. These include causing the machine to stop obeying orders, either under programme control or when the programme is in error. In particular, the machine stops if an order for writing in the main store is reached and an overflow indicator is set. A further aid when testing new programmes is the automatic punching out of all main-store addresses appearing in block-transfer orders. When this information is examined an indication of the course of a programme is readily obtained. The punching can be inhibited by a switch when a return to full-speed running is needed.

Machine rhythm

The logical design of Pegasus is built around a nucleus that deals with the simple arithmetic orders, groups 0, 1 and 4, of the code. This nucleus contains the control section, i.e. the order register and order decoding equipment, and the mill in which these orders are executed. The design of this nucleus could not begin until a basic rhythm for dealing with the extraction from the computing store and the execution of such a pair was determined. When the outline of this nucleus was clear, the equipment for dealing with the remaining orders in the code was designed to fit it.

The following arguments led to the basic rhythm. Since the orders of groups 0, 1 and 4 are similar in many respects, for definiteness, it will be sufficient to consider a particular order, 11 of the code, say. This is an order which takes two numbers from the computing store and replaces one of them by their sum. It would take a prohibitive amount of equipment to extract these numbers, add them together and have the least significant digit of the sum available for replacing in the store in the same digit time as the least significant digits of the two components taken out of the store. In practice, some four digit times at least would be needed for this sequence of operations. Thus, it would be impossible to return the sum to the store in the same word as the operands are extracted without having an entry point to each register which is in a different timing from the normal circulation entry. To produce two such entry points to each register would mean more equipment associated with each register, which was considered an uneconomical use of extra equipment. Instead, it was decided to delay the sum so that it could enter the register in the computing store in the next word time in standard timing. This involves one common delaying circuit instead of one for every register. Such an order therefore takes two word times to execute. It may be argued that this second word time could be made to overlap with the first word time for the next order. Two reasons oppose this: the new contents of the register being changed might be required by the next order; and two different sets of equipment for selecting a storage register would be needed if numbers were to be extracted from one and replaced in another register in the same word time.

Thus, the execution of a pair of orders taken from the computing store requires four word times. The reasons for opposing the overlapping of the execution of two orders also oppose the extraction of an order pair while the previous pair is being dealt with. Five word times are therefore needed for the process of extracting and obeying a pair of simple arithmetic orders. More time may be needed for some of the other orders in the code.

The basic 3-beat rhythm is thus established:

a Extract the order pair from the computing store.

b Obey the first order of the pair.

c Obey the second order.

The duration of beat (*a*) is one word time; beats (*b*) and (*c*) are each two word times long for orders in groups 0, 1, 4 and 6 of the code, but may be longer for other orders.

Times for typical operations

The times for the various arithmetic operations are:

	millisec
Addition and subtraction	0.3
Multiplication	2.0
Division	5.4

These times include an allowance for the time to extract the orders.

Some times for standard subroutines are:

	millisec
Exponential function	29
Sine function	24
Logarithmic function	34

Finally, to give some indication of the time for a typical problem, a set of 50 simultaneous equations (with a single right-hand side) takes about $10\frac{3}{4}$ min. Of this time, 3 min 8 sec is for input, 7 min 17 sec is for calculation and 18 sec is for output.

Realizing the specification

The detailed logical design

It would take too long to describe fully the detailed logical design. One aspect is worth mentioning, however, namely the avoidance of all 'exceptions' in the results of orders. As an example of an exception consider the overflow indicators, which should be set whenever the final result of an order is outside the permissible range of numbers. In multiplication this can occur only when both the multiplier and the multiplicand are -1, and this is likely to occur very infrequently. Rather than provide equipment to sense this infrequent case, it is easier to put a footnote in the programming manual, where the overflow indicator is described, pointing out the exception. It was felt, however, that such exceptions should be avoided even at the expense of extra equipment or extra complication. For this and other reasons concerned with facilitating machine use, the logic of Pegasus is quite complicated.

The end-product of the detailed logical design is a series of diagrams with symbols corresponding to the circuit units of the packages, as shown, for example, in Fig. 5. The inputs and outputs of the units on these diagrams correspond to the pins of the sockets into which the packages plug. Thus, the wiring lists of connections of these pins can be produced from these logical diagrams. The first step in the production of these lists is to allocate a position

in the cabinets to each logical circuit in such a way as to reduce the amount of wire needed. When the layout has been completed, the last stage of producing the wire lists can proceed.

General construction of machine

The main units are shown in Fig. 10.

The package frame. This unit is a simple light-alloy frame supporting diecast light-alloy frame racks to which the back socket panels are fixed. The packages slide into grooves in the rack and plug into sockets at the back, a polarizing feature preventing the insertion of a package upside down. If electrical or magnetic screening is necessary between any packages, a special metal plate is inserted in slots in the cast rack and is fixed by a single screw in the back panel. Coded aluminium strips containing coloured plastic studs which identify the position of each package are fixed to the front of each casting.

Arrangement of the packages. There are 200 packages per cabinet, arranged in ten horizontal rows of 20 units per row. The metal valve panels are placed so that the edges almost touch. The component panel of each unit is in register with the unit in the corresponding position in each of the other rows, thereby providing vertical chimneys for cooling the components secured to these

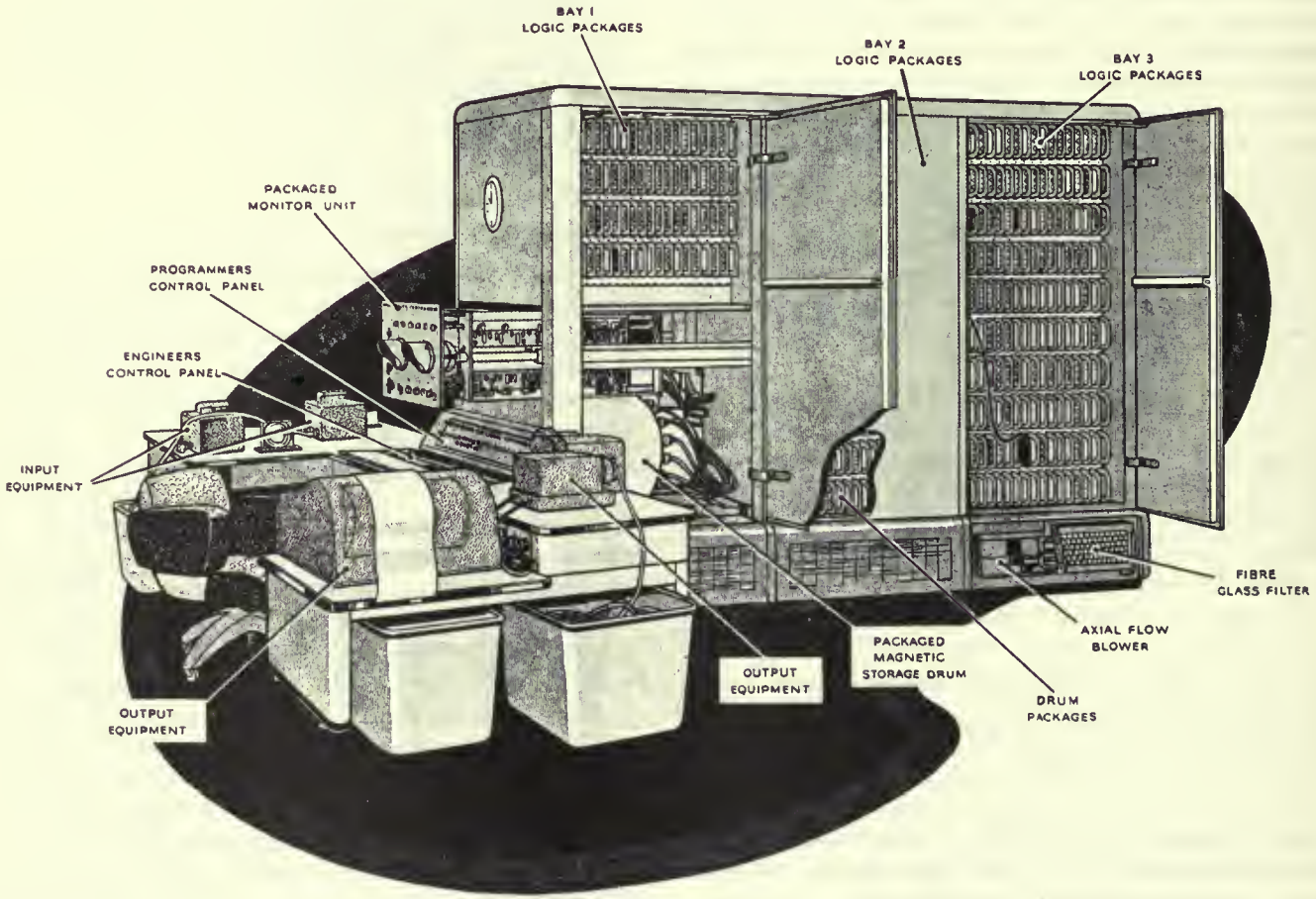

Fig. 10. Main units.

panels. Warm air from the main source of heat, the valves, is prevented by the valve panels from reaching the more temperature-sensitive components, such as diodes, secured to the component panel.

The back panel wiring. For locating long signal wires between sockets a system of plastic strips is used, which hold the wires at definite positions given by the instructions on the wiring lists. The exact route of every wire is predetermined, thus making wiring and inspection more reliable and fault finding and maintenance easier.

Final assembly. The completely wired frame is assembled in its cabinet, which has already been fitted with the control and auxiliary supply circuit unit, heater transformers, fuses, cooling assembly and cableforms. The work of connecting the cableforms, heaters and earths can be done by relatively unskilled labour working to clearly written instructions and diagrams.

The cooling system. Each cabinet has its own cooling system as an integral part of the construction; there is therefore no difficulty in cooling cabinets added to existing computers. Two axial-flow turbo blowers are mounted in the base beneath an airtight pressure chamber, each providing 300 ft^3/min of air at a total pressure head of 1 in (water gauge). The maximum temperature rise is 10° C.

The power supply. A separate cubicle houses metal rectifiers, shunt stabilizing valves and control circuits. The power is obtained from the mains through a motor-alternator set, the output of which is stabilized to 2%, the main purpose of this set being to act as a buffer against switching surges and other mains voltage variations. The valve heaters in the computer are energized from the stabilized alternator output, which is expected to extend the valve life.

Maintenance

General

All digital computers so far have a fault rate which cannot be ignored. When the best has been done in the choice of components, circuits and mechanical construction, attention must be paid to the following points to get the best out of a machine:

a Rapid fault location

b Getting the machine working again as soon as possible after locating a fault

c Preventive maintenance

Fault location

There are parity-checking circuits on both the main and the high-speed stores. Errors of a single digit in the stores stop the machine. The fault can then be quickly located by examination of the monitors.

For other faults the general method is to run a test programme (assuming the fault is not in the main control) which will indicate the area of the fault. Detailed examination can then be carried out with the monitors.

All outputs of circuit units are readily accessible at monitoring sockets on the front of each package, and in addition about 80 points can be directly selected by switches from the monitoring position: these include all store lines and a number of key waveforms. Fault-finding is normally a matter of tracing 0's and 1's through the machine with reference to logical diagrams rather than electronic circuit diagrams.

A variety of triggers can be selected for the monitor time-bases, these including

a Trigger at any word position within a drum revolution (128 different times selectable by switches)

b Trigger at any word time of any selected order

These triggers and some other monitoring facilities are produced by 19 standard packages and are found to be well worth the extra equipment.

Fault repair

Once a faulty package has been located, the machine can be got working again immediately by replacement of the package with a spare; repair of the faulty package can be done at leisure with the aid of a package tester. With this equipment a package can quickly be given a series of standard tests; each is selected by switches, and the performance is measured either by observation of meters or a built-in oscillograph.

During commissioning not one case was found of the first machine doing other than what one would expect from the logical diagram (except for a very few cases of incorrect wiring).

Preventive maintenance

The machine h.t. supplies are reduced while the test programmes are being run. This marginal testing shows up incipient faults such as deterioration in valves, crystal diodes or resistors. The machine is at present kept in good running order down to 10% margins

(the supplies are normally controlled to about 1% of nominal), although correct running at about 20% reduction has been observed.

Conclusions

The first machine has been computing regularly for only a few months and has been on regular preventive maintenance (about 1 hour per day) for a few weeks. Error-free runs of over 30 hours are common, and at the time of writing there has been no error for $55\frac{3}{4}$ hours' running. The majority of package replacements are done during routine maintenance.

The packaged method of construction of computers has proved to have great advantages in design, construction and operation.

References

ElliW56a; ElboR53; ElliW51, 52, 53, 56b; FairJ56; JohnD52; MerrI56; Pegasus Programming Manual, Ferranti Ltd., London; Pegasus Maintenance Manuals, Ferranti Ltd., London.

APPENDIX

The Pegasus Order Code

00 $x' = n$
01 $x' = x + n$
02 $x' = -n$
03 $x' = x - n$
04 $x' = n - x$
05 $x' = x \,\&\, n$
06 $x' = x \not\equiv n$
07 Not allocated

10 $n' = x$
11 $n' = n + x$
12 $n' = -x$
13 $n' = n - x$
14 $n' = x - n$
15 $n' = n \,\&\, x$
16 $n' = n \not\equiv x$
17 Not allocated

20 $(pq)' = n \cdot x$
21 $(pq)' = n \cdot x + 2^{-39}$
22 $(pq)' = p + 2^{-38}q + nx$

23 $(nq)' = n + 2^{-38}q$ { this order assumes that any overflow is due to operations in 7. Clears overflow unless n' overflows

24, 25 $q' + 2^{-38}\left(\frac{p'}{n}\right) = \frac{x + 2^{-38}q}{n}$ { $0 \le p'/n < 1$ (unrounded division); $-\frac{1}{2} \le p'/n < \frac{1}{2}$ (rounded division)

26 $q' + 2^{-38}\left(\frac{p'}{n}\right) = \frac{x}{n}$; $-\frac{1}{2} \le p'/n < \frac{1}{2}$ (rounded single-length division

27 Not allocated

30, 31, 32, 33, 34, 35, 36, 37 } Not allocated

40 $x' = c$
41 $x' = x + c$
42 $x' = -c$
43 $x' = x - c$
44 $x' = c - x$
45 $x' = x \,\&\, c$
46 $x' = x \not\equiv c$
} $c = N2^{-38}$
47 Not allocated

50 $x' = 2^N x$ } single-length arithmetical shifts
51 $x' = 2^{-N}x$ (rounded) } single-length arithmetical shifts
Note: $x' = x$ if $N = 0$
52 Shift x up N places } single-length logical shifts
53 Shift x down N places } single-length logical shifts
54 $(pq)' = 2^N(pq)$ } double-length arithmetical shifts
55 $(pq)' = 2^{-N}(pq)$ (unrounded) } double-length arithmetical shifts
Note: $p' = p$ and $q' = q$ if $N = 0$

56 (Normalize) $(pq)' = 2^{\mu}(pq)$;

$$x' = x - 2^{-38}\mu \begin{cases} \text{either (1) } \frac{1}{4} \leq (pq)' < \frac{1}{2} \text{ and} \\ \qquad -1 \leq \mu \leq N - 1 \\ \text{or (2) } -\frac{1}{2} \leq (pq)' < \frac{1}{4} \text{ and} \\ \qquad -1 \leq \mu \leq N - 1 \\ \text{or (3) } -\frac{1}{4} \leq (pq)' < \frac{1}{4} \text{ and} \\ \qquad \mu = N - 1 \end{cases}$$

57 Not allocated

60 Jump to N if $x = 0$
61 Jump to N if $x \neq 0$
62 Jump to N if $x \geq 0$
63 Jump to N if $x < 0$
64 Jump to N if overflow staticizor clear; clear overflow staticizor.
65 Jump to N if overflow staticizor set; clear overflow staticizor.
66 (Unit-modify) $x'_m = x_m + 1$. Jump to N if $x'_m \not\equiv 0$ (mod. 8)
67 (Unit-count) $x'_c = x_c - 1$. Jump to N if $x'_c \neq 0$

70 Single word read to accumulator 1. $1' = s$
71 Single word write from accumulator 1. $s' = 1$
72 Block read from main store $u' = b$
73 Block write into main store $b' = u$
74 External conditioning
75, 76 Not allocated
77 Stop

The notation used here is as follows:
N is the first address (the register address) in an order.
X is the accumulator specified in an order.
n is the word in N before obeying the order.
x is the word in X before obeying the order.
p and q are the words in 6 and 7 before obeying the order.
$(pq) = p + 2^{-38}q$, with $q \geq 0$. This is a double-length number.
x', n', p' and q' are the corresponding values after obeying the order.
B is a block in the main store (the drum).
U is a block in the computing store.
P is the position number of a word within a block.
OVR is the overflow indicator.
xm is the modifier in X, i.e. an integer represented by the digits 1 to 13 of x.
xc is the counter in X, i.e. an integer represented by the digits 14 to 38 of x.