



**FEATURES OF THE
FERRANTI ATLAS COMPUTER**

FEATURES OF THE FERRANTI ATLAS COMPUTER

CONTENTS

	Page
1. INTRODUCTION	1
2. THE STORE HIERARCHY FROM THE DESIGNERS' VIEWPOINT	2
3. THE UNIFIED STORE FROM THE PROGRAMMERS' POINT OF VIEW ..	2
4. THE STRUCTURE OF ATLAS COMMANDS	3
5. ENTERING FIXED STORE ROUTINES	4
6. FLOATING POINT ARITHMETIC	5
7. INDEXING OPERATIONS	7
8. BRANCHING	7
9. THE PERIPHERAL SYSTEM FOR ATLAS	8
10. EXTRACODE COMMANDS	10
11. PROGRAM TIME-SHARING	11
12. ADAPTABILITY OF THE SYSTEM	12
13. PROGRAM SOURCE LANGUAGES	12

FIGURES

- Fig.1 Designers' view of Atlas
- Fig.2 Users' view of Atlas
- Fig.3 Core store selection of Atlas
- Fig.4 Command format
- Fig.5 Structure of the command code
- Fig.6 Floating point number

PLATES

- Plate 1 A typical Atlas Computer installation
- Plate 2 Professor T. Kilburn and Mr. M. Lanigan with the index store of the prototype Atlas at Manchester University
- Plate 3 Dr. R.L. Grimsdale with the fixed store of the prototype

ACKNOWLEDGEMENT

Like the earlier successful Ferranti Mark I and Mercury Computers, Atlas is the result of a close collaboration between Ferranti Ltd. and the University of Manchester, England, to whose original research Atlas owes many of its striking features.

Features of the Ferranti ATLAS Computer

1. INTRODUCTION

Atlas has been designed to meet the needs of the large scale computing and data processing centre, which must be able to handle the widest range of work, from computation at the highest speeds to routine data processing, at a low cost per operation.

The design of Atlas exploits to the full the use of modern programming techniques to provide a powerful, comprehensive computing system for the user, by combining carefully planned built-in system programs with fast but economical basic hardware. This design philosophy is reflected in every aspect of Atlas, so that the loss of effective speed often associated with the use of interpretive routines is avoided. This is seen most clearly in the routines for automatic storage allocation, which could only be provided in other existing machines at an enormous cost in operating efficiency.

Atlas has a hierarchy of stores of exceptional size at each level of access time. In order to enable it to use its speed and capacity to maximum effect, the system is designed to allow simultaneous operation of several programs, each with its own set of peripherals (i.e. input-output equipment). However, the user can write his program as if it were for a single-level store machine, and as if he were the only user of this machine.

The effective unification of the store from the user's point of view has been achieved by introducing an entirely new method of handling the transfers of information between different levels of storage, by system routines assisted by special hardware. This also provides complete protection against mutual interference of programs sharing the machine.

Atlas is provided with a relatively simple basic command code of the single address type which provides for all the elementary operations, in both fixed and floating point form. In addition to these simple basic commands the programmer may also use a number of additional commands of a more sophisticated type called 'extracodes', which cause entry to one of a large number of built in subroutines. The extracodes look exactly like the basic instructions and may be mixed with them without complication. They provide all the common mathematical functions (such as sin, cos, tan, etc.) in command form together with all the control commands required for the widest range of peripheral equipment, and for input-output conversion and assembly operations.

The extracode subroutines, together with all the system programs for storage allocation, monitoring, control of peripheral equipment etc., are held in an extremely fast fixed ("read only") store of novel design. A special variable store (the "subsidiary" store) provides working space for all these programs; this is protected against interference by other programs in the machine.

The extremely high speed of the computer (which is of the order of a million commands executed per second) is achieved by fast access to storage combined with extensive parallel operation. For example, at any one time as many as three different commands may be in different stages of execution. This is made possible by providing several access systems to different parts of the store, and also by the parallel operation of the floating point accumulator and the index registers.

2. THE STORE HIERARCHY FROM THE DESIGNERS' VIEWPOINT (see figure 1)

The word length throughout the machine is 48 bits, which can conveniently be subdivided, for example, into 8 characters of 6 bits each. Each 24 bit half-word has a 25th parity bit associated with it, and the parity is checked on every store transfer.

The fastest store is a fixed store constructed from a woven wire mesh into which ferrite rods representing digits are inserted. It has an access time of 0.3 microseconds, and it contains 8192 words in the standard Atlas although it may be extended up to a theoretical maximum capacity of 262,144 words. It is this store which holds all the extracode routines, and certain fixed programs concerned with drum transfers, time-sharing, monitoring and the operation of peripherals.

Next in speed is the B-store, which contains 128 half-word (24 bit) index registers, to which access may be made in 0.35 microsecond (cycle time 0.7 microsecond). As the arithmetic unit used for indices is quite distinct from the floating point accumulator, it is possible to execute indexing commands while a floating point accumulator operation is in progress.

The main core store of the computer has a cycle time of 2 microseconds. However, this time is effectively reduced by the provision of a number of distinct access systems to sections of the store. Thus on the first machine, the core store of 16384 words will have 4 access systems. The overlapping of commands is made possible by these independent access systems which are so arranged that consecutive commands are read from the even and odd registers in the store through separate access systems, so that while one command is using the accumulator, another command is being completed by sending information to the store, and yet a third is being initiated by sending the command from the store to the control.

The core store is backed up by a drum store containing 24576 words per drum. The first machine will have 4 such drums, but many more can be provided. The unit of transfer is one "block" of 512 words which are read or written in 2 milliseconds; the revolution time is 12 milliseconds. Transfers are made directly to or from the main core store, and computing can occur simultaneously. Drum transfers are initiated by fixed store routines.

The fixed store uses a subsidiary core store (of 2 microseconds cycle time) as working space for the fixed store routines. This subsidiary store is locked out from commands in the main store, and it is thus private to the fixed store routines.

Finally there is a set of registers which have addresses in a particular part of the address range and which are referred to collectively as the V-store. These consist mostly of the information and control signals passing to and from the magnetic drums, magnetic tapes and other peripheral equipments. This part of the store is also private and accessible only to the fixed store routines.

3. THE UNIFIED STORE FROM THE PROGRAMMERS' POINT OF VIEW (see figure 2)

Although the system makes use of the various parts of the store described above in order to achieve the best balance in the use of equipment, it is not in fact necessary for the ordinary user to be aware of them all. Thus the fixed store, together with its subsidiary store and the V-store, can be regarded merely as carrying out some of the functions of the control unit, in a way which need not concern the user. Moreover the core store and the drum store are together organised in such a way

that they appear as a single level main store to the programmer, which together with the index registers in the B-store, and the accumulator, are the only parts of the machine with which he is directly concerned. All the other parts of the store (the fixed store, the subsidiary store and the V-store) are dealt with by the extracode routines, which are used by the program as commands. These other parts of the machine are referred to as the private store of the machine. Consequently, the machine appears to the programmer as shown in figure 2, in which the core and drum stores have been unified, and the private store is treated as part of the control of the machine.

The unification of the main store is accomplished by means of sets of registers in the V-store known as the page address registers (see figure 3). The information in the unified main store is regarded as contained in a series of numbered blocks, each of 512 words. These blocks are transferred between the core store and the main store from time to time, and a list of the blocks currently in the fast store is contained in the page address registers. The "addresses" referred to in the commands written by the programmer are the block numbers and word numbers within the block.

When a particular command is obeyed, the page address registers are scanned extremely rapidly by special hardware. If one of these registers contains the number of the block in which the required word lies, it emits a recognition signal, and the corresponding page in the store is used for the command. Thus, so long as the required commands and their data are in the core store, access can be made at the full speed of the core store. If ultimately a word is called for which is not in the core store, then none of the page address registers signals agreement. When this happens the program is automatically interrupted and control is transferred to a special routine in the fixed store, called the drum transfer routine.

The drum transfer routine selects which of the blocks in the core store is to be replaced by the block required, and it arranges to write the replaced block on a suitable position in the drum store. It keeps a directory in the subsidiary store of the actual drum addresses corresponding to each block number. Thus a block is not always written onto the same part of the drum store, but can be put into the next available space to reduce waiting time. In order to enable the drum transfer routine to do this, the current angular positions of each drum are available in certain registers in the V-store. The optimising functions of the drum transfer routine are carried out by a "learning" program which is being developed for this purpose. Having made the necessary transfers, the drum transfer routine adjusts the contents of the page address registers, updates its own records, and allows the program to continue.

The effect of these arrangements is that the programmer benefits from what appears to him to be a very large store, supplied at a fraction of the cost which would result if magnetic cores were used throughout. Moreover, the block numbering system will be used automatically to ensure that a program can only refer to its own area of the main store, thus making effective time sharing possible.

4. THE STRUCTURE OF ATLAS COMMANDS

The command format is shown in figure 4, and more detail is given in figure 5. The function number occupies 10 bits. The first of these distinguishes between machine code functions and extracode functions. The machine code functions fall into two classes (see figure 5):-

Accumulator functions ("A-Codes") involving the accumulator and a word in the store whose address has been modified by the addition of the contents of two index registers designated B_a and B_m .

Index functions ("B-Codes") involving an index register and a half-word in the store whose address has been modified by the addition of the contents of one index register, the one designated B_m . (However, some branching commands are unmodified).

B_a , B_m occupy two 7-bit parts of the command as illustrated in figure 4, allowing for 127 indexing registers and zero. Of these 90 are available to the programmer (including number zero which always contains zero), 30 are used by the extracode routines and 7 are special purpose registers.

Addresses in the main store refer to words and characters within words. As there may be eight characters in a word, it is convenient to regard the character address as an octal fraction. The notation used is exemplified as follows:-

Character 5 in word 1234 has address 1234.5

The second half-word in word 1234 has address 1234.4

The first half-word in word 1234 or the whole word 1234 has address 1234.0 which may be written as 1234

There is a maximum of 2^{20} whole word addresses in the main store. The interpretation of these addresses involves the page address registers as described above.

The addresses in the private store are numbered separately, and are distinguished by a 1-bit at the most significant end of the address, as illustrated in figure 4. Figure 5 shows how the various sections of the private store are distinguished.

5. ENTERING FIXED STORE ROUTINES

There are three distinct command address or control registers in Atlas known as main, extracode and interrupt. They are respectively used and advanced when obeying the programmer's commands, when obeying extracode routines, and when obeying interrupt routines. (These interrupt routines are fixed routines dealing with certain aspects of the organisation of peripheral transfers etc.). The three controls facilitate return to the appropriate routine after leaving it for any reason, and they are available in the special purpose index registers 127, 126 and 125.

When an extracode command is encountered, the machine performs a standard procedure for entering the appropriate fixed store routine. First the address part of the command is copied into the index register B 119, after being modified by the addition of the contents of one or both of the specified indices. In the case of B-extracodes, the B_a part of the command is copied into B 121. Then the function digits are copied into the extracode control register (B 126) and the machine proceeds to obey this control. The result is that the machine enters a routine determined by the function digits of the extracode command, and can make use of B 119, and B 121 as arguments. On exit (which is caused by a 1 in the second function bit of a basic instruction) the machine merely reverts to main control, so that the main program is resumed at the command following the extracode.

B 121 incorporates a special facility which enables a fixed store routine to operate on an index register specified during the execution of a program. If the number 122 appears in the B_a part of an instruction the machine will operate, not on B 122 (which does not exist as a separate register), but on the index register whose address is given in B 121.

There are two flip-flops, known as M/E and I/ME, which determine whether the machine is obeying main, extracode or interrupt control. The second indicates whether or not interrupt control is in use; if not, the first indicates whether main or extracode control is in use. When a cause for interrupt occurs, the I/ME flip-flop is set to the 'I' state and the interrupt control number in B 125 is set to a standard value. So long as interrupt control is in use, further interrupts are inhibited. When the interrupt routine is finished it resets the I/ME flip-flop, and the machine then resumes whichever control it was previously obeying, as determined by the M/E flip-flop. (If further alarms have arisen in the meantime, another interruption will occur immediately.)

There are about 150 distinguishable causes for interruption, associated with various conditions of drums, peripheral equipment, and internal checks, all calling for urgent attention by fixed store routines. Each is associated with an alarm flip-flop in the V-store. The arrangement of these, in conjunction with a special facility associated with index register B 123, is such that control can be directed within a very few microseconds to the particular routine that deals with the most urgent matter requiring attention at any moment. The computing speed of Atlas is such that all interruptions can be dealt with in the time available.

Interrupt and extracode routines are allowed access to the whole store of the machine, but when main control is in use, the private store is locked out. Thus, for example, main programs are denied access to the V-store which contains the control links with peripheral equipment. Ordinary programs therefore cannot affect the peripheral equipment directly. Instead, they must request a fixed store routine to do so. This they do by entering an extracode routine, which contains several safeguards against erroneous operation of the equipment.

In general, a peripheral transfer involves first the use of an extracode routine, to set the equipment in motion and initiate the transfer. Control then reverts to the main program, but there may be several interrupts from time to time to allow a fixed store routine to attend to the details of the transfer as it progresses.

6. FLOATING POINT ARITHMETIC

In a floating point number, 8 bits are used to represent the exponent, and the remaining 40 bits of the Atlas word represent the mantissa including sign, as shown in figure 6. The exponent is an octal one, so that the number represented is $x.8^y$. True complements are used for negative numbers, so that when normalised $\frac{1}{8} < x < 1$ or $-1 < x < -\frac{1}{8}$, and $-128 < y < +128$.

Floating point arithmetic makes use of a floating point accumulator, in which the mantissa is double length (sign + 78 bits).

Many variants of the basic arithmetical commands are provided, giving options concerning rounding off and normalising (or standardising). Thus for example fixed point arithmetic can be performed by using commands which do not cause standardisation.

The following arithmetical commands operate on the contents, s, of a register in the unified store, and the previous contents of the accumulator, a, placing the result in the accumulator. The left hand column gives the machine function code in octal form.

Function	Type of Command	Result	Remarks
0300	floating, unrounded	a + s	
0301	" "	a - s	
0302	" "	s - a	
0320	floating, rounded	a + s	
0321	" "	a - s	
0322	" "	s - a	
0330	fixed point	a + s	
0331	" "	a - s	
0332	" "	s - a	
0324	floating point	s	standardised
0325	" "	- s	
0334	fixed point	s	unstandardised
0335	" "	- s	
0342	unrounded floating point multiplication	a.s	
0343	" " "	-a.s	
0362	rounded floating point multiplication	a.s	
0363	" " "	-a.s	
0352	single length multiplication of integers	a.s	in least significant half of acc, signed.
0353	" " " "	-a.s	
0372	double length multiplication	a.s	double length in whole accumulator.
0373	" " " "	-a.s	

Rounding is adding a digit to the least significant digit of the most significant half of the accumulator if this digit is zero, and if the least significant half of the accumulator is not zero.

In addition to the above, there are further basic commands for division, storing the result from the accumulator, testing the sign of the accumulator, forming the modulus of a or s, etc.

Overflow of the exponent causes a program interrupt leading to a monitoring routine in the fixed store. This will however give an individual programmer the option of overriding the standard monitoring procedure. Overflow of the mantissa in fixed point operations sets an indicator which can be tested by program.

Floating point addition takes about 1.4 microseconds and multiplication about 4.6 microseconds. If the addition instruction is modified it takes about 1.8 microseconds; if it is doubly modified it takes about 2.3 microseconds. During multiplication there is more opportunity for overlap with modification. These times can however vary, depending on the amount of overlapping of commands which results from the use of the multiple access system and the independent access to the index registers. It takes about 10 n microseconds in all to form the scalar product of two n-vectors.

A typical accumulator operation may be coded as follows:-

0320, 51, 52, 1234

"add the floating point number in register 1234 + i + j to the accumulator, and round off, where i and j are contained as word addresses in index registers 51 and 52."

7. INDEXING OPERATIONS

The indexing commands operate on the contents of the 24-bit index register whose address is B_n and whose contents are denoted by b . The other operand may be the half-word (s) in the store designated by the address in the command (after modification by the addition of the contents b_m of the index register B_m), or it may be the address part (n) of the command itself.

Some of the elementary operations on the contents of the index registers are as follows:

Command	Operation	Command	Operation	Command	Operation
0100	s-b to b	0110	s-b to s	0120	n-b to b
0101	s to b	0111	-b to s	0121	n to b
0102	b-s to b	0112	b-s to s	0122	b-n to b
0103	-s to b	0113	b to s	0123	-n to b
0104	b+s to b	0114	b+s to s	0124	b+n to b
0106	b≠s to b	0116	b≠s to s	0126	b≠n to b
0107	b&s to b	0117	b&s to s	0127	b&n to b
0147	bvs to b			0167	bvn to b

It may be noted that modified operations of the type involving n may be used for adding (or subtracting) the contents of one index register to (or from) another. For example, the operation

"Form $i+j+3$ in the word address position of index register 51, where i is in index register 50 and j is in index register 51" may be coded as:-

0124, 50, 51, 3

Although modification is normally performed by adding b_m to n , there are two special functions in which this addition is replaced by a logical "&" operation. These are functions 0164 and 0165, which are otherwise equivalent to 0124 and 0121. These functions are of great value in data manipulation; for example, the operation

"Add to index register 60 the last 6-bit character from index register 59" requires one instruction only:

0164, 60, 59, 7.7

In addition to the above basic commands, there are also basic commands causing cyclic shifts of index registers either one place right or six places left. The latter, being a high speed operation completed in about a microsecond, is particularly useful for operations on 6-bit characters. Shifts of arbitrary numbers of places are accomplished by extracodes using combinations of the above instructions and taking a total time averaging 12 microseconds.

8. BRANCHING

It has been mentioned that the three command address or control registers are identical with index registers 127, 126, 125 (main, extracode and interrupt). Branching is effected by altering the contents of the appropriate control register.

An unconditional branching is obtained by copying the address of the next command to be obeyed into the appropriate command address index register. Thus in the main program, the effect of the instruction,

0121, 127, 0, 1234

will be to cause the program to jump to the command contained in address 1234.

To permit conditional branching a set of commands is available which will test the contents of the accumulator, or of index register B_m , and will copy the address digits n of the command into b if the condition is satisfied. These commands are as follows:-

Command	Effect	Command	Effect
0214	If $b_m=0$, copy n to b	0234	If $a=0$, copy n to b
0215	If $b_m \neq 0$, " " " "	0235	If $a \neq 0$, " " " "
0216	If $b_m \geq 0$, " " " "	0236	If $a \geq 0$, " " " "
0217	If $b_m < 0$, " " " "	0237	If $a < 0$, " " " "

Thus to jump to 1234 if index register 50 is negative, the command in the main program would be:-

0217, 127, 50, 1234

It will be appreciated that in addition to its use for conditional branching, this command may also be used for conditionally re-setting an index.

In order to facilitate counting operations, a set of four commands is available which will test an index, and if it is not zero will either step on or will step back the index by a word or half a word and will then copy the address digits of the command into b .

These commands are as follows:-

Command	Effect
0200	if $b_m \neq 0$ add 0.4 to b_m and copy n to b
0201	if $b_m \neq 0$ add 1.0 to b_m and copy n to b
0202	if $b_m \neq 0$ subtract 0.4 from b_m and copy n to b
0203	if $b_m = 0$ subtract 1.0 from b_m and copy n to b

Furthermore, in order to make it possible to test the size of an index without altering it, a special register is provided which can be used to record the sign and zero-equivalence of either $s-b$, $b-s$, $n-b$ or $b-n$. This register is called b_t , and four commands are available to test b_t to see if it represents the states designated = 0, $\neq 0$, ≥ 0 , or < 0 . It should be noted that in these cases, the address n to which a branch goes may be modified by the addition of the index b_m .

A set of counting commands analogous to those already described above, but operating by testing the state of b_t is also available.

9. THE PERIPHERAL SYSTEM FOR ATLAS

The table below lists the peripheral equipment to be attached to the first Atlas to be installed, and also that for which connections are built in. Additional equip-

ment beyond that allowed for by the built in connections can be incorporated but will require additional hardware

Input Equipment	First Installation	Built in Connections
Clock	1	1
I.C.T. card reader (600 cards/min.)	1	4
TR5 paper tape reader (300 characters/ sec.)	4	12
TR7 paper tape reader (1000 characters/ sec.)	-	4

Output Equipment

Teletype punch (110 characters/sec.)	4	12
I.C.T. card punch (100 cards/min.)	1	2
I.C.T. printer (600 lines/min.)	1	2
Teleprinter (10 characters/sec.)	2	16
Creed high speed punch (300 characters/ sec.)	-	4
Xeronic printer (3000 lines/min.)	-	2
Graphical Output (Cathode ray)	1	2

Magnetic Tape

Ampex TM2 units (90,000 characters/ sec.)	8	32
--	---	----

The paper tape equipment is capable of handling 5-hole, 7 hole or 8-hole tape. The I.C.T. card equipment will read and punch standard I.B.M. or Hollerith 80 column cards, and includes check reading stations which are used to check all reading and punching. The I.C.T. printer has a set of 50 characters and prints 120 characters per line.

The Ampex TM2 units use 1" magnetic tape. Fixed length blocks of 512 words are recorded, and each block is addressed. The addressing of new tapes is done by Atlas itself as a preliminary operation.

A 24-bit end-around-carry check sum is automatically recorded at the end of each block and is checked at the end of each transfer; writing is checked by a trailing reading head. Faulty transfers are repeated up to three times; if still unsuccessful monitoring occurs.

Individual blocks on a tape may be rewritten without disturbing neighbouring blocks. Reading may occur with the tape travelling either forwards or backwards; writing is always done forwards.

Up to eight magnetic tape transfers can occur simultaneously; if more transfers are requested they are held in a waiting list. The speed of Atlas is such that a magnetic drum transfer, eight magnetic tape transfers and input-output transfers involving *all* of the equipment listed above can occur simultaneously.

Atlas will also read and write I.B.M. half inch magnetic tape if circumstances require this facility.

The method by which the peripheral equipment is attached ensures the utmost flexibility in operating procedure. All signal and control lines to and from this equipment are given addresses in the V-Store. Thus all communication with the equipment is carried out by fixed store routines (which have access to the V-Store). The only exceptions to this rule are the transfers to and from the drums and tape, which once started, are automatically allowed direct access to the main store for the periods required to transfer words, without requiring the attention of a fixed store routine throughout the transfer. However, they can only be initiated through control lines in the V-Store, and are therefore effectively under the control of fixed store routines.

10. EXTRACODE COMMANDS

The basic commands listed in sections 6, 7 and 8 are supplemented by approximately 250 extracode commands to provide a truly comprehensive list of functions, all available to the user as single machine commands. These include commands for conversion to and from decimal or mixed radix numbers, operations on strings of characters, double length arithmetic, complex arithmetic, half-word arithmetic, double length index operations, indirect addressing, table look-up, simple functions such as square root, logarithm, exponential, sine, cosine, tangent, arcsin, arcos, arctan, integral part, fractional part; also the generation of random numbers, summation of polynomials, operations on vectors, and integration of differential equations by the Runge-Kutta-Gill method. In short, the range of operations normally found in the subroutine library of a well-equipped computer is provided permanently within the fixed store of Atlas.

In addition to these, there are extracodes for every operation involving peripheral equipment. These provide for all necessary code conversions, allowing for all punched tape and card codes in common use, and incorporate suitable checks on the correct functioning of all the equipment.

Included in the list of extracodes are commands enabling the programmer, if he wishes, to assist the drum transfer routine by indicating the numbers of blocks which should be brought into the fast store, or which are no longer required in the fast store. Furthermore, he may instruct the system to make deliberate changes in block labels, thus giving the effect of copying blocks from one part of the store to another but taking practically no time to do so; in this way much modification in programs can often be avoided.

The routines controlling magnetic drum and magnetic tape transfers are arranged so that they prevent programs from referring to a block in core store while it is in process of transfer. This is accomplished by setting a special lockout bit associated with each page address register (but not shown in figure 3) which prevents recognition of that block by the program, while still permitting reference to it by magnetic transfers and fixed store routines.

One result of this arrangement is that, for example, a block of data may be read from magnetic tape into the core store alongside the data which it is intended to replace, without actually replacing it. The latter may be used by the program right up to the moment when the new data is wanted. At this stage an extracode routine is entered which has merely to revise the block labels in the page address registers to make the new block immediately take the place of the old.

Some of the fixed store routines controlling peripheral transfers require more space for buffering purposes than is available in the subsidiary store; in such cases a block of main store is temporarily appropriated and protected by setting its lockout bit.

There are special extracode routines for building up strings of items of variable length to be recorded in complete blocks on magnetic tape, and for separating these items on reading.

11. PROGRAM TIME-SHARING

Without any extension of the basic hardware described above, Atlas permits the concurrent operation of any number of programs, without there being any possibility that an error in one of them will interfere with any other.

Supervisory routines are being prepared for the fixed store which will be called into action as soon as a program is held up pending the completion of a peripheral transfer, or is halted by an alarm such as an arithmetical overflow. These routines will hand over control to another program if appropriate, after making the necessary adjustments to prevent interference between two programs. Such adjustments will include storing the contents of the accumulator, all index registers used by the program, the index test register and certain other indicators, and they will also effect the necessary changes to the contents of the page address registers. By setting lockout bits, the supervisory routine can ensure that only those blocks belonging to the program now operating can be referred to.

The priority accorded to a program can be made subject to continual review by a supervisory routine in the fixed store. This routine will keep an account of the references to different types of peripheral equipment, and on the basis of this information it will review the program priorities from time to time. Thus the actual priorities accorded to programs will depend partly on the operators' instructions (which may have an overriding effect if necessary), and partly on the operating experience accumulated so far by the supervisory routines. In this way the system as a whole will be managed so that the operators' intentions are fulfilled while maintaining a high level of working efficiency at all times.

The supervisory and drum transfer routines maintain a directory of all blocks of information within the machine, identified by program number and block number. In this way all blocks are distinguished, even though there may be two blocks in different programs with the same block number. In such a case the supervisor would ensure that only the block belonging to the currently operating program is left unlocked in the core store at any time.

Thus any program may refer to any word address in the range from 0 through $2^{20} - 1$, regardless of the addresses used by any other program, and regardless of the actual size of the store. The only restriction is that the total number of different blocks referred to by all the programs in the machine at any time cannot exceed the number of blocks actually existing. The programmer is thus afforded a degree of freedom in store referencing which has hitherto been unknown.

Another task of the supervisory routines is to review the storage and peripherals required by all job requests submitted to the system, ensuring that so far as possible the programs are run in a sequence that makes the maximum utilisation of the available equipment. Loading and unloading instructions are issued to the operators when necessary. Data tapes and cards are identified by headings, and a directory is kept by the system of the current role being played by each peripheral device, so that no confusion can arise between programs.

It will often be desirable to record output information initially on magnetic tape, and to print it out subsequently as a separate operation; in this way the tape is used as a buffer to smooth the flow of information to the printer. The printing operation, usually done hitherto by special "off-line" equipment, can be done by Atlas itself as one of its parallel activities. The simple programs required for this form a permanent part of the Atlas system and absorb only a few per cent of the computing time of Atlas - time which might otherwise be unoccupied - thus removing the need for extra special-purpose machines which can themselves be very costly.

The same provisions are made for the "off-line" reading and punching of cards and punched tape via Atlas itself. In general, a common character code is used within the system for information from all sources, so that programs and data may be handled without regard to their original form, and any output information can be sent to tape, cards or printer.

If there is any computing time available after all these activities have been attended to, it will be used by a comprehensive general test routine in the fixed store, which will test in rotation as many of the internal operations of the machine as possible.

12. ADAPTABILITY OF THE SYSTEM

The approach which has been adopted to the design of Atlas makes it a system which is extremely adaptable to a wide variety of conditions. The arrangement of the store means that any combination of fast and slow stores can be provided to suit the requirements for the efficient running of the problems met with at any installation, without invalidating programs developed at other installations. At any time, the size of the fast store can be increased, and all earlier programs can then immediately be run with the increased efficiency corresponding to the new configuration, without any amendment of the programs whatever.

The method of attaching peripheral equipment makes it relatively easy to handle new types of equipment as they are developed, even though their control may be complicated, because fixed store routines can be provided to attend to them. Analog-digital converters, character readers, graphical plotters and remote links may for example be connected directly to Atlas, which can itself observe synchronization signals, apply redundancy checks, etc. It is worth noting in this regard that although the interrupt and extracode routines are normally held in the fixed store, the design of the system will permit them to be extended into the main store if desired. Thus special real-time processing routines can be put in the main store when required and can then have access to the peripheral connections in the V-Store; this however does not prejudice the mutual protection of other programs in the machine. (This is in fact the way in which extracode routines have been checked out before being incorporated in the fixed store.)

13. PROGRAM SOURCE LANGUAGES

The general purpose compiler being provided for Atlas may be described as a compiler to write compilers. More precisely it is a system which enables the user to define the form and meaning of the statements which are to be used in the source language.

Most users will not want to define their own source languages and standard definitions will be available for commonly used languages such as Fortran, Algol and Mercury Autocode. The user may adopt these languages as they stand or he may add further definitions to extend them into his particular problem field.

© Ferranti Ltd., 1961

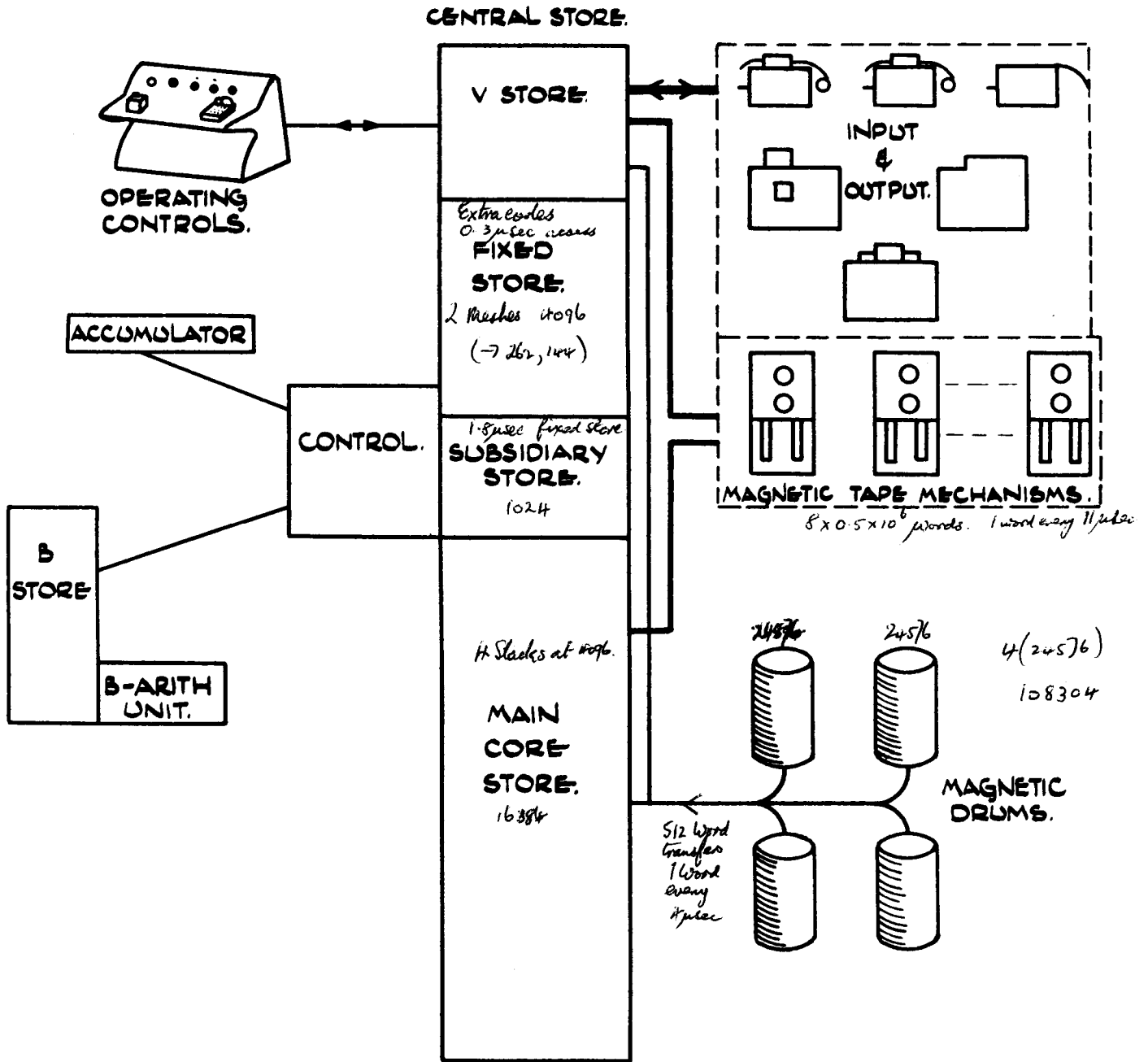


FIG. 1. DESIGNERS' VIEW OF ATLAS.

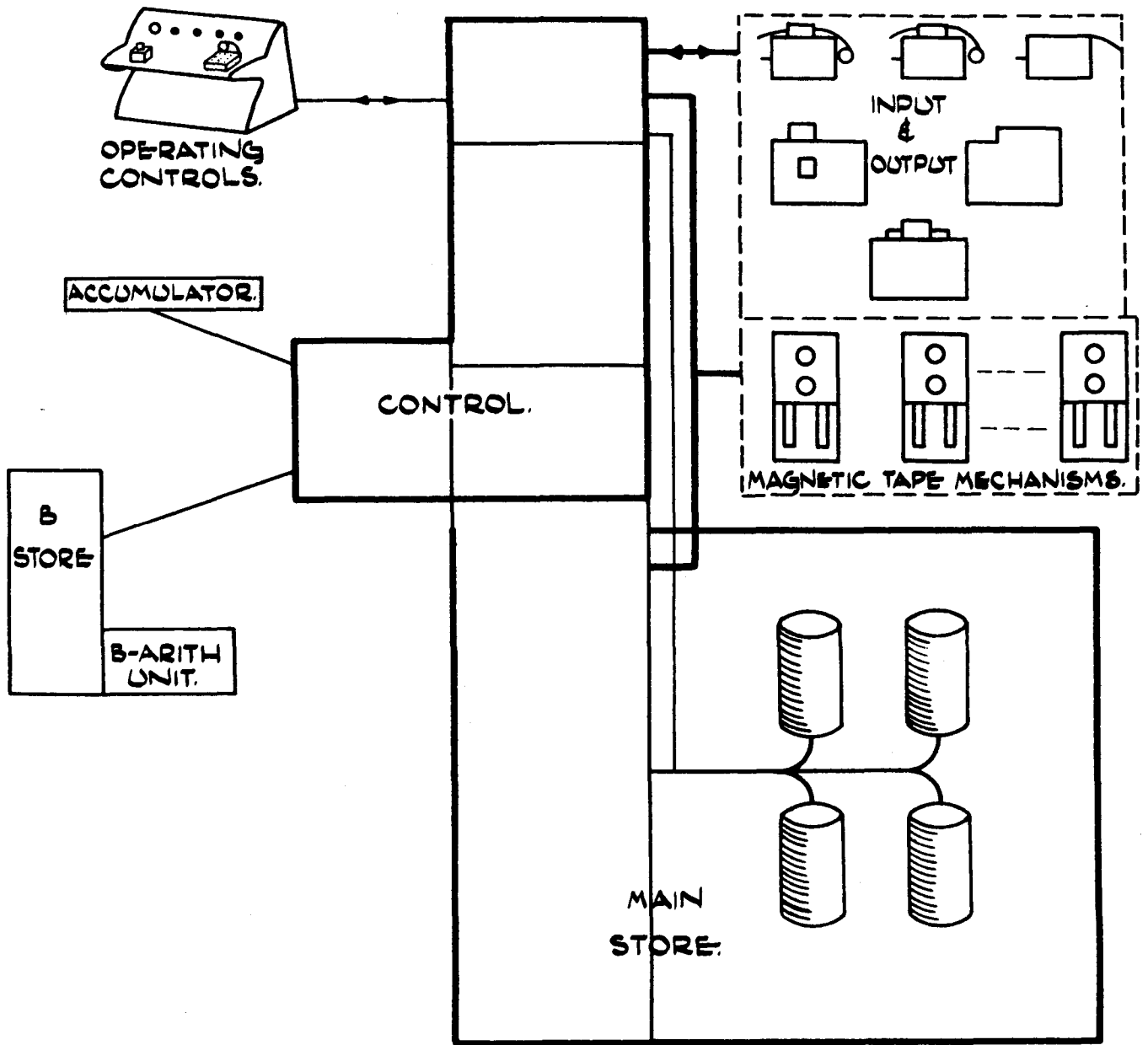


FIG. 2. USERS' VIEW OF ATLAS.

MAIN STORE WORD ADDRESS.

(AFTER MODIFICATION, IF ANY.)

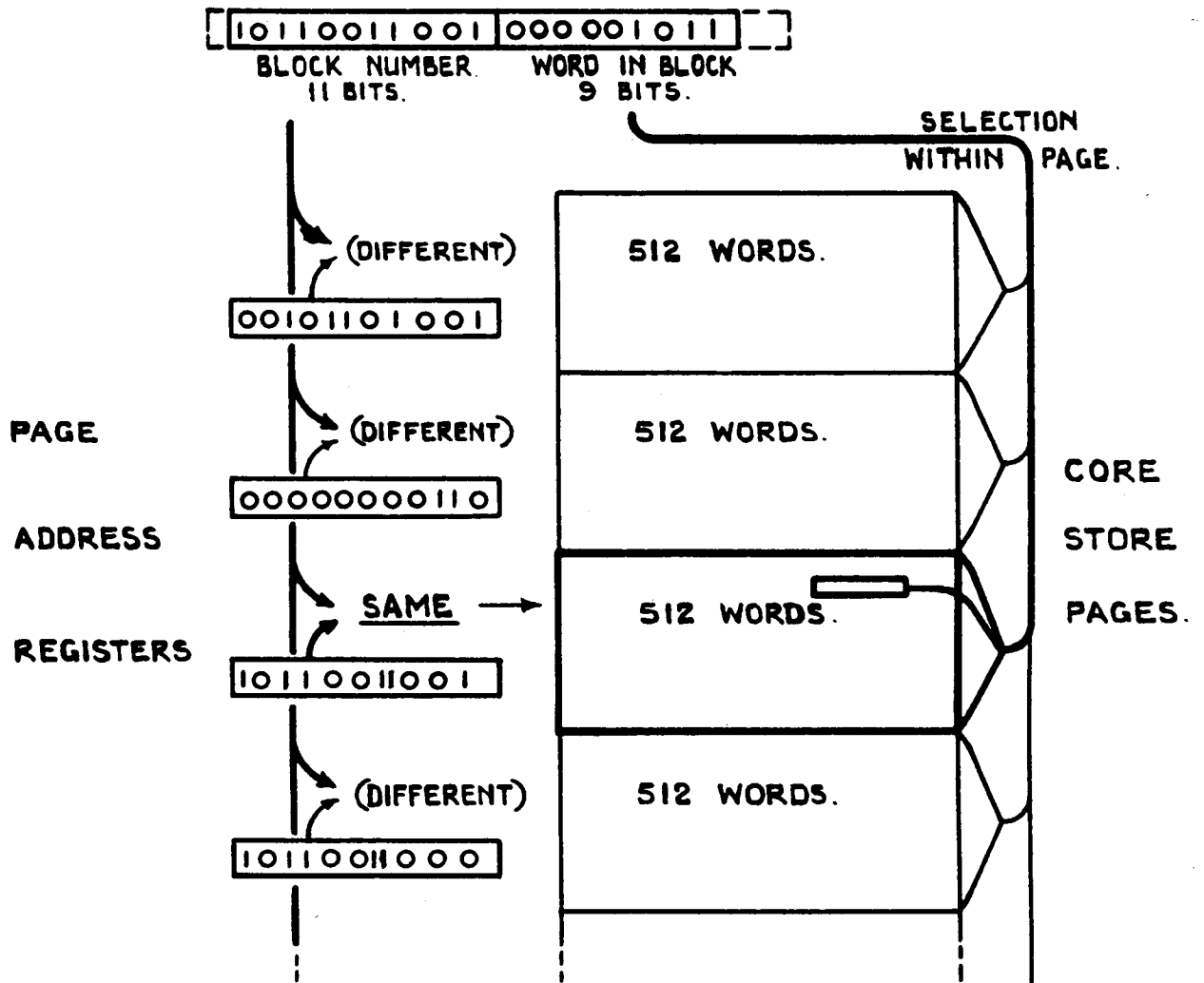


FIG. 3. CORE STORE SELECTION IN ATLAS.

M/S
H

L/S
0

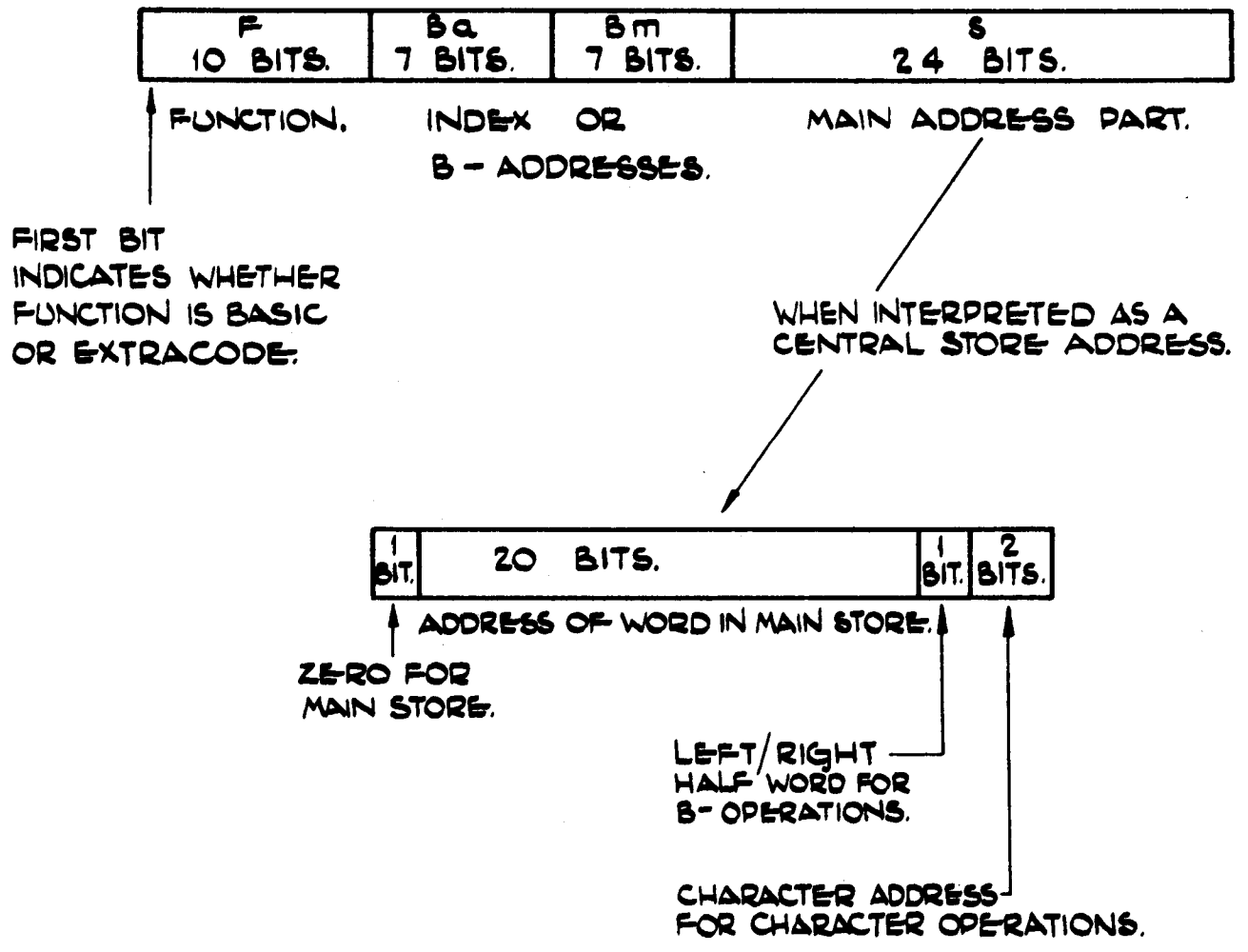
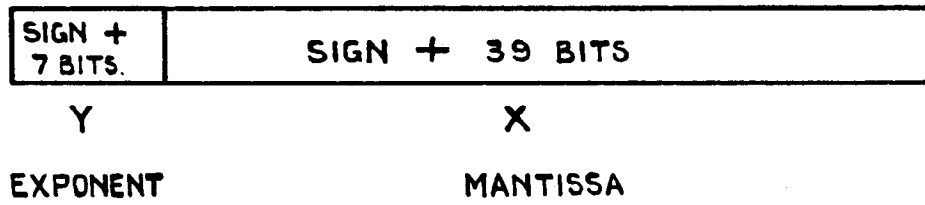
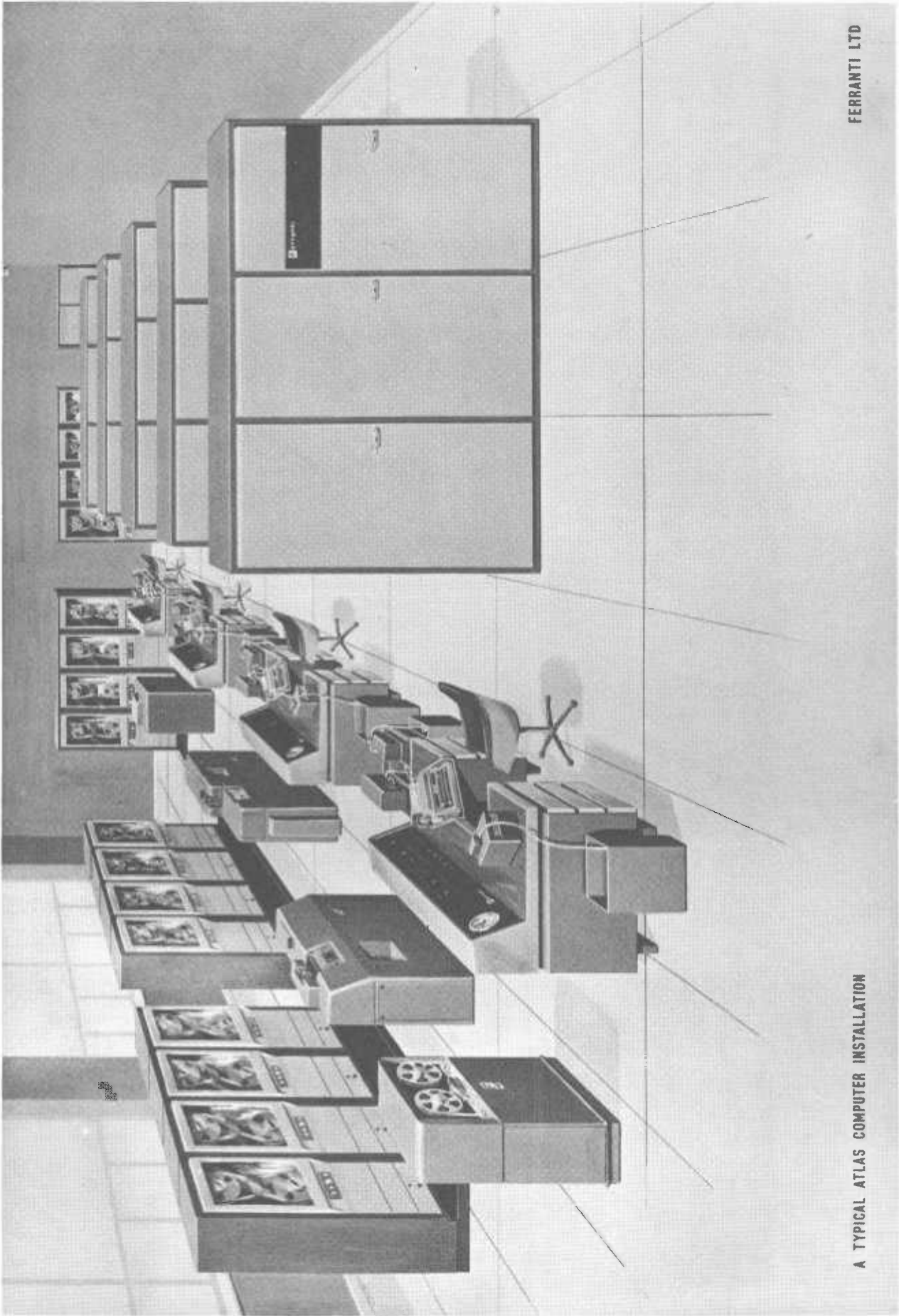


FIG. 4. COMMAND FORMAT.



NUMBER REPRESENTED = $X \cdot 8^Y$

FIG. 6. FLOATING POINT NUMBER



FERRANTI LTD

A TYPICAL ATLAS COMPUTER INSTALLATION



