ELECTRODATA

PROGRAMMING AND CODING MANUAL

DATATRON

electronic data processing machine

ElectroData CORPORATION

pasadena, california

DATATRON

ElectroData Corporation has an expert staff of mathematicians and computer specialists whose activities range from the coding of subroutines to the basic mathematical development of solutions to complex scientific and business problems.

Programming assistance, coding, and consultation are available at ElectroData's computing center or at the plant of the Datatron user.

Copyright 1955
by ElectroData Corporation

ElectroData Corporation

Programming and Coding Manual

DATATRON

Electronic Data Processing Machines

Preliminary Edition

Copyright 1954
by ElectroData Corporation

TABLE OF CONTENTS

Programming and Coding Manual

DATATRON
Electronic Data Processing Machines

TABLE OF CONTENTS - continued

TABLE OF CONTENTS - continued

SYMBOLS AND ABBREVIATIONS

L_i = Quick-access Loop i, where i = 4, 5, 6, 7 .

For example, L_7 = the 7000 loop.

( ) = The contents of. For example, (L_4) = The contents of L_4

$(\ \ )_b$ = The contents <u>before</u> an operation

$(\ \ )_a$ = The contents <u>after</u> an operation

ASG, A1, etc. = The individual digits of the A register, where ASG represents the sign digit, and A1..., A10 (the least significant digit) are the successive digits of the A register. Similarly for the D, R, and C registers, of which only the D register contains a sign. The B register is designated B1, B2, B3, B4.

O.F.T. = Overflow toggle

$\equiv$ = Congruent. For example, 2037 $\equiv$ 4037 modulo 20.

ms = Millisecond

mod = Modulo

DEFINITIONS OF TERMS

<u>Absolute value</u>: Numerical or positive value.

<u>Access time</u>: The time interval between the instant at which information is in position to be read from a drum or other storage and the instant at which the next operation can begin; i.e., the read time. Also, the interval between the instant a signal to store information is received and the instant at which storage is completed; i.e., the write time.

Copyright 1954
by ElectroData Corporation

Addend: The number which is added to another number in addition.

Adder: A device capable of forming the sum of two quantities.

Address: A label such as an integer or other set of characters which identifies a register, location, or device in which information is stored. Specifically, the designation of a location from which a number is to be brought for arithmetic operation.

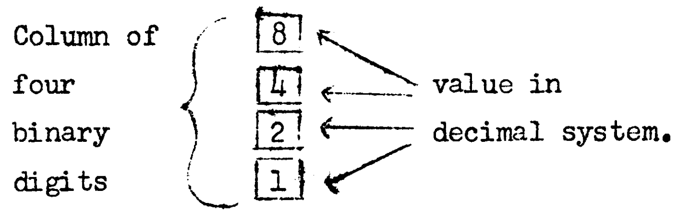
Absolute address (specific address): The label(s) assigned by the machine designer to a particular storage location.

Relative address: A label used to identify a word in a routine or subroutine with respect to its position in that routine or subroutine. Relative addresses are translated into absolute addresses by the addition of some specific "reference" address, usually that at which the first word of the routine is stored.

Symbolic address (floating address): A label chosen to identify a particular word, function or other information in a routine, independent of the location of the information within the routine.

Argument: An independent variable; a given value subject to some operation from which a corresponding result is to be found.

Binary-coded decimal: System by which decimal numbers are represented by a column (or row) of four binary digits. In this computer each binary digit in the column represents a fixed decimal value.



Copyright 1954
by ElectroData Corporation

A zero in one of the four bits indicates that the corresponding decimal value is to be omitted. The decimal values of the "one" bits are summed to find the total decimal value of the four-bit column. For example the column representing a decimal six would be:

$$\begin{array}{ll} \boxed{0} & \\ \boxed{1} & 4 \\ \boxed{1} & 2 \\ \boxed{0} & \end{array} \qquad 4 + 2 = 6$$

Binary number system: Number system in which there are only two characters, 0 and 1, represented by the two states of various types of components. Electronic computers use the binary system or binary-coded systems extensively since the two states of components are "on" and "off", or conducting and non-conducting.

Bit: A binary digit.

Block: A group of words considered or transported as a unit.

Breakpoint: A point in a routine at which the computer may, under the control of a manual switch, be stopped for a visual check of progress.

Cell: Storage for one unit of information, usually one character or one word.

Character: One of a set of elementary symbols such as those corresponding to the keys on a typewriter. The symbols may include the digits 0 through 9, the letters A through Z, punctuation marks, spaces, operation symbols, and any other single or coded symbols which a computer may read, store, or write.

Checkout: Trial run of a completed program to test for any coding, scaling, punching, or keyboard errors.

Copyright 1954
by ElectroData Corporation

Code:

(noun) - A system of symbols and of rules for their use in representing information.

(verb) - To prepare problems in computer code or in pseudo-code.

Command: A word consisting of an order (two digits), an operand address or operational indicator (four digits), and other digits performing various tally and control functions. Also, in this computer, one of the orders (arithmetic, logical, shifting, or transfer operations) which the computer executes on the appearance of the appropriate two digits in the control section.

Console: Desk containing operator's control panel for the computer.

Control counter: A four-digit register, capable of counting, which holds at all times during machine operation the address of the command to be executed next.

Cycle:

(noun) - 1. A set of operations repeated as a unit.

2. A non-arithmetic shift in which the digits dropped off at one end of a word are returned at the other end in circular fashion.

(verb) - To repeat a set of operations a prescribed number of times.

Debug: To isolate and remove all malfunctions from a computer or all mistakes from a routine.

Decade: One decimal digit of a register, represented on the ElectroData console by four neon lamps in vertical column.

Copyright 1954
by ElectroData Corporation

Digit: One of the n symbols of integral value ranging from 0 to n-1 inclusive in a scale of numbering at base n; especially, one of the 10 decimal digits.

Floating-point representation: A notation in which a number x is represented by a pair of numbers y and z (and two integers n and m which are understood parameters in any given representation), with y and z chosen so that $x = y \cdot n^z$, where z is an integer and ordinarily $y = 0$ or $m > |y| \geq m/n$, (where n is usually 2 or 10 and m is usually 1). The quantity y is called the fraction or mantissa; the integer z is called the exponent or characteristic.

Flow chart: A graphical representation of a sequence of operations, using symbols to represent various operations.

Forbidden combination: A combination of binary-coded decimal digits which produces a number between ten and fifteen inclusive.

Format control: Manually adjustable circuits which determine Flexowriter printout form.

In-check point: Point in program where computer can be stopped and a machine computed value can be checked against a known value. Used during checkout of a program.

Instruction: The two digits of an order or command. Also, certain other digits of a command word which perform such functions as controlling format of printout and the operation of input devices.

Keyboard: Device whereby numbers can be loaded into computer or punched on tape simply by depressing a key corresponding to the correct decimal digit or character.

Copyright 1954
by ElectroData Corporation

Loop: Depending upon context, the word may mean:

1. A section of coding repeated a number of times.
2. A section of memory having much shorter access-time than the main memory.
3. A length of paper tape with beginning and end spliced together.

Microsecond: One millionth part of a second.

Millisecond: One thousandth part of a second.

Modify: To alter in a command the address indicating where the operand is stored.

Most (least) significant digit: In decimal numbers, the digit in a number which is multiplied by the algebraically highest (lowest) power of 10 appearing in that number. Also called "highest- (lowest-) order" digit.

Normalize: To shift left a number in the A register until the first non-zero digit is in the first position of the A register. For example, +.0000567832 in normalized form is +.5678320000.

Operand: Any one of the quantities entering or arising in an operation. An operand may be an argument, a result, or a parameter; specifically, in this computer, the number stored at the location designated by the address part of certain command words.

Order: (See "Command".) In this computer, a two-digit instruction. A fundamental distinction for timing and internal operation is that between operand and no-operand orders.

Overflow: (over capacity) - In an arithmetic operation, the generation of a quantity beyond the capacity of the register or location which is to receive the result.

Parameter: In a subroutine, a quantity which may be given different values

Copyright 1954
by ElectroData Corporation

when the subroutine is used in different main routines or in different parts of one main routine, but which usually remains unchanged throughout any one such use.

Patch panel: Multiple-terminal electrical connector(s); specifically, connectors at the rear of format control assembly which route information among various output devices.

Prestore: To store a quantity in an available or convenient location before it is required in a routine. Also, to set an initial value for the address of an operand or a cycle index; to restore.

Pregram: (noun) - A plan for the solution of a problem. A complete program includes plans for the transcription of data, coding for the computer, and plans for the absorption of the results into the system.
(verb) - To plan a computation or process from the asking of a question to the delivery of the results, including the integration of the operation into an existing system. Thus programming consists of planning and coding, including numerical analysis, systems analysis, specification of printing formats, and any other functions necessary to the integration of a computer in a system.

Register: A temporary storage place for a number which has immediate access.

Routine: A set of instructions and data in the proper order, enabling a computer to solve a programmed problem. For this computer, coded routines are punched on tape or cards.

Run (noun): One performance of a program on a computer.

Scaling: Determining of size of numbers likely to be encountered during a computation and making provision in the code to keep them within the range of the computer. Also, adjusting the size of the input data to

Copyright 1954
by ElectroData Corporation

fit the computer.

Serial operation: Usually refers to way in which numbers are transferred between registers or storage locations. Serial indicates digit-by-digit operation.

Subroutine: The set of instructions necessary to direct the computer to carry out a well-defined mathematical or logical operation; a subunit of a routine. A subroutine is often written in relative or symbolic coding even when the routine to which it belongs is not.

Tally: To count the number of passes through a series of commands.

Toggle: Electronic circuit consisting of a bi-stable multi-vibrator or flip-flop. Its "on", or "one" position is indicated on the console by a neon light.

Word: A set of characters which occupies one storage location and is treated by the computer circuits as a unit and transported as such. Ordinarily a word is treated by the control unit as a command, and by the arithmetic unit as a quantity.

Copyright 1954
by ElectroData Corporation

CHARACTERISTICS OF THE COMPUTER

The ElectroData Computer is a general-purpose automatic digital computer capable of being programmed to solve a wide variety of mathematical, engineering, and business problems. In common with other fully automatic digital computers, it stores instructions (coded in numerical form) and the numbers with which it is to operate, interchangeably in its internal memory. It makes comparisons between numbers or instructions and alters the sequence of operations accordingly. These two basic capabilities enable this type of computer to:

> Perform arithmetic and logical operations on either instructions or numbers in accordance with sequences of instructions stored in its memory;
>
> Select alternate sequences of instructions depending upon the results of a comparison between two numbers or instructions; and
>
> Alter or modify one or more instructions in accordance with a fixed sequence of instructions.

The solution of computer problems, in general, involves many repetitions of relatively few basic sequences of operations; the hundreds of thousands of individual operations required for the solution of a typical problem may be carried out entirely automatically by the computer in accordance with an original set of perhaps only a few hundred instructions. In many cases, the original set of instructions may be used over and over again to provide solutions of the same problem for various changes in the parameters, or the same set of instructions, with minor modifications, may be used to provide solu-

Copyright 1954
by ElectroData Corporation

tions for many related problems. The ability of an automatic electronic digital computer to perform arithmetic and logic (including selection of alternate sequences and alteration of instructions) at the rate of several hundred operations per second makes it an efficient tool for solving complex problems, or for reducing large quantities of raw data.

<u>Computer Systems</u>

ElectroData computing systems now available include:

A. Input-Medium Preparation Equipment

B. Input Equipment

C. Basic Computer

D. Auxiliary Storage Equipment

E. Output Equipment

<u>Number System</u>

The ElectroData Computer operates in the decimal number system, representing each digit of the data and coded instructions by a group of four binary digits, called a <u>decade</u>. The position of the binary digit indicates a 1, 2, 4 or 8 in the decimal system. Using this binary-coded-decimal system, it is possible to represent numbers up to 15 (sum of 1, 2, 4 and 8). However, those above nine are not used in the ElectroData computer except for internal checking. Internal operation in the familiar decimal system (in contrast to the binary, octal, or hexadecimal systems often employed by electronic computers) significantly simplifies the preparation of the instructions (coding), eliminates the necessity for conversion to and from the decimal system for input and output, and facilitates the location of coding and machine errors, should they occur.

Copyright 1954
by ElectroData Corporation

The register length is 10 decimal digits and algebraic sign; all numbers are stored in the memory as absolute value and sign. Fixed-decimal-point operation is normally employed, all numbers being treated as if the decimal point were fixed between the algebraic sign and the most significant digit. All numbers lie between -1 and +1. The smallest non-zero absolute number that can be represented is $1 \cdot 10^{-10}$.

Command System

The computer uses one-address instruction code, with each command (or instruction) occupying the same memory space as one normal number. A command is made up of three parts:

(a) the numerical code for the specific operation to be performed (two decimal digits);

(b) the address, usually a location in the memory to be referred to during execution of the command (four decimal digits);

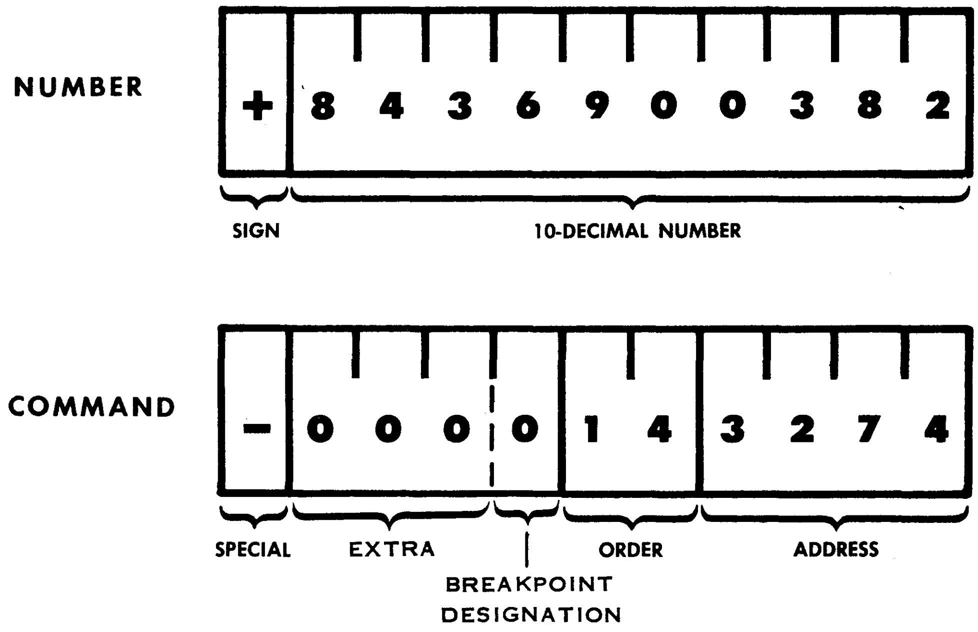
(c) and digits which designate special properties of the command. (These will be described later.)

The address part of a command word occupies the four least significant digits of a register or memory location, which simplifies arithmetic modification of addresses. The two-digit operation code immediately precedes the address. The codes for special operations occupy the other positions.

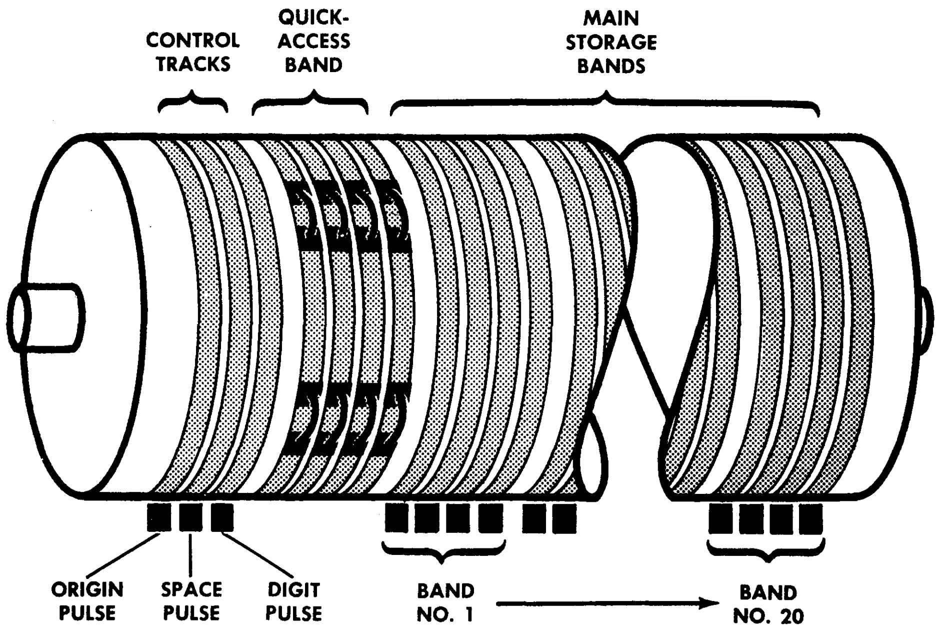
Since the computer operates in the single-address system, the address of the next command to be executed need not be specified explicitly; commands are always executed in the sequence in which they are stored in the memory until this normal sequence is interrupted by the execution of a "change control" command, which specifies the address of the next command to be executed.

Copyright 1954
by ElectroData Corporation

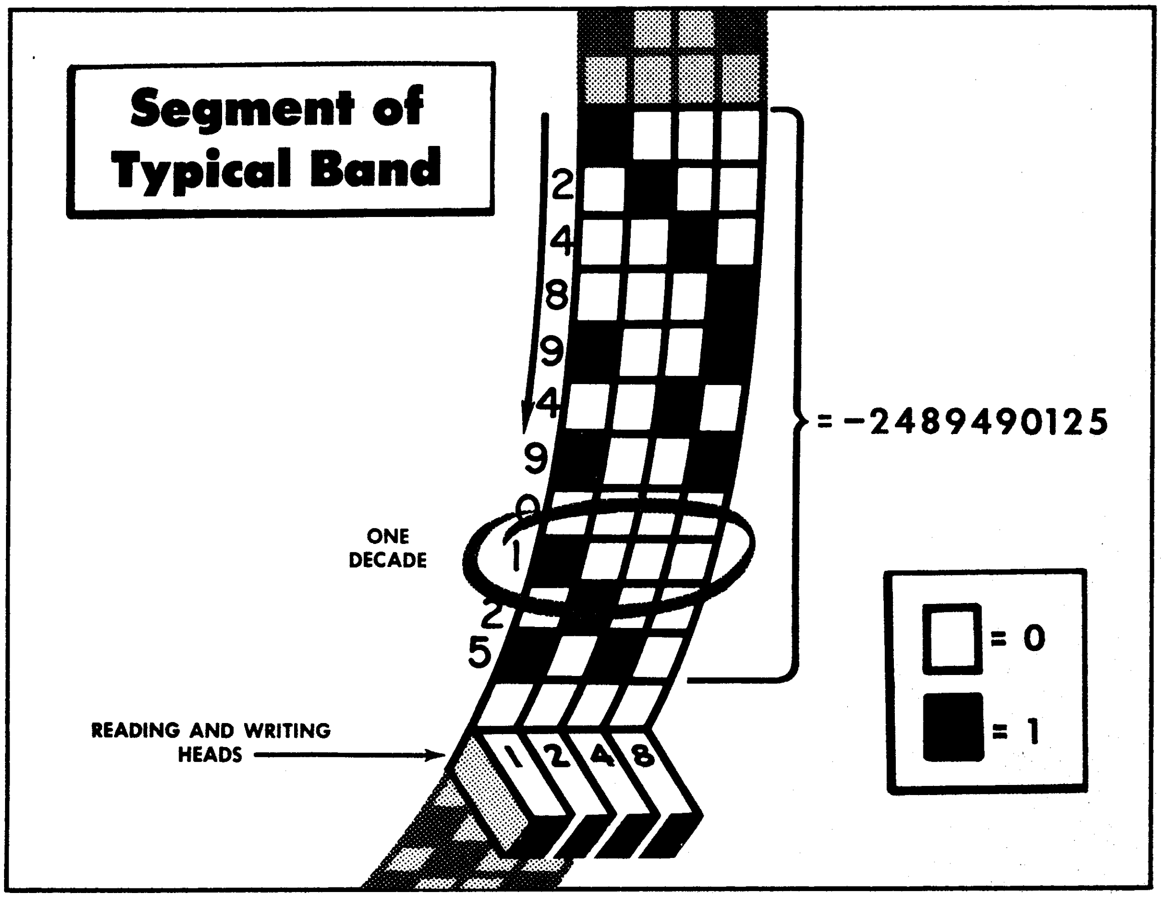
"Word" Structure

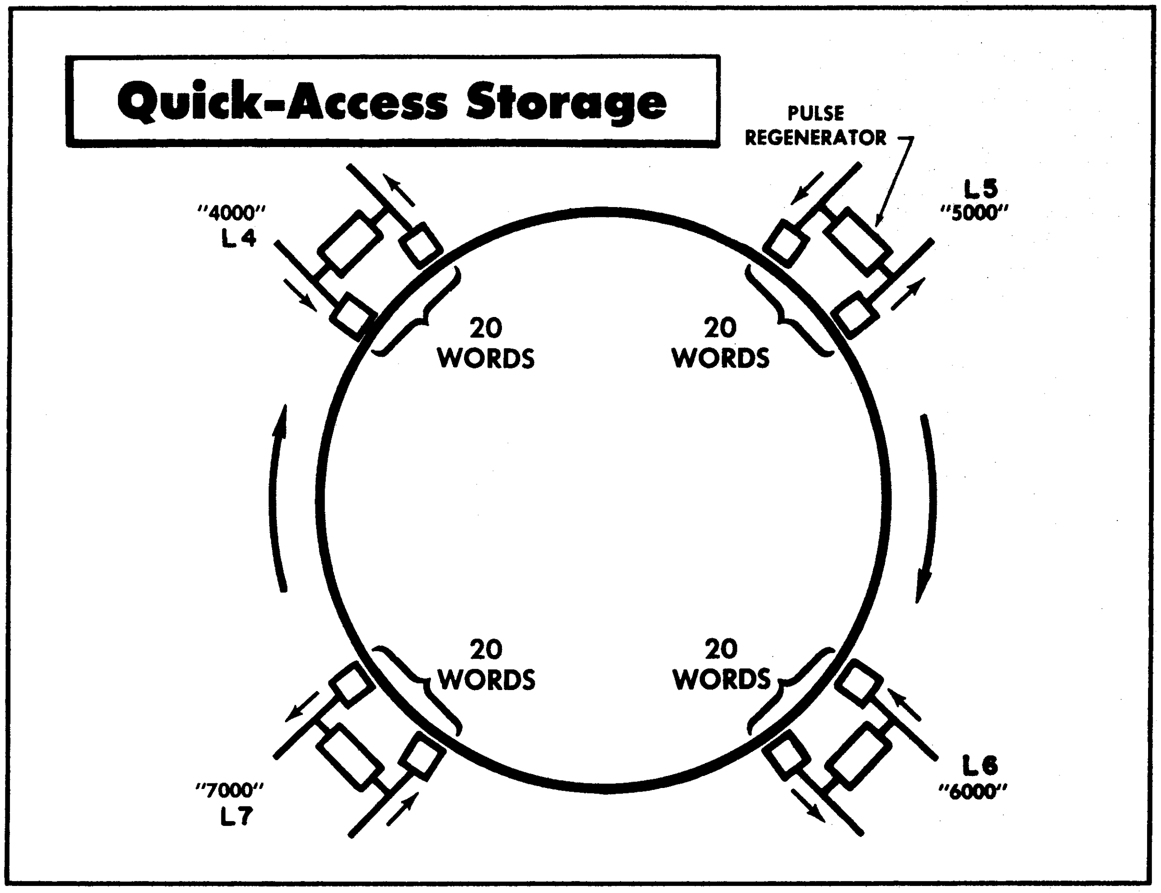


Storage Drum



Copyright 1954 by ElectroData Corporation





Copyright 1954 by ElectroData Corporation

Memory System

A magnetic drum, rotating at approximately 3600 rpm, along with the associated magnetic reading-writing heads and electronic circuits, serves as information storage for the computer.

One section of the drum, called the main memory, has a capacity of 4000 words. (A word is either a number or a command.) Twenty bands along the axis of the drum contain 200 words each around the circumference of the drum. Each band consists of four tracks. Each track, containing one binary digit of the decimal code, is provided with its own reading-writing head and the associated circuits. The four heads for one band read or write simultaneously, handling binary digits in parallel. Since the positions for the eleven decimal digits of each word (including the algebraic sign) occur circumferentially on the drum, they successively come into position under the heads as the drum rotates, handling decimal digits in series. Since the electronic arithmetic section of the computer handles the numbers and instructions in the same way, the computer operates in "series-parallel".

A second section of the drum, the quick-access memory, has a storage capacity of 80 words. It consists of four circulating loops, each loop having two groups of four magnetic heads and associated circuits. There is space for 20 words between the two groups of heads for each loop. As each decimal digit is read, the signals from the reading heads are amplified, reshaped, and sent back to the writing heads, which precede them rotationally. Thus, the block of 20 words in each loop is continuously circulated as the drum rotates. Single words may be read from or written into any of these four loops, or a block of 20 consecutive words may be transferred in either direction between any of the loops and the

Copyright 1954
by ElectroData Corporation

main memory.

Pulses derived from permanently recorded tracks on the drum, in conjunction with the electronic control circuits, synchronize the operation of the computer and positively locate specific word and digit positions around the circumference of the drum. Any particular word position in the main memory may be specified by giving the band (lengthwise along the drum) and the sector (circumferentially around the drum, relative to the origin-pulse position) in which it is located. For the convenience of the coder, addresses of word positions in the main memory are designated by numbers - 0000 through 3999; the translation of an address so designated into the coordinates of band and sector is accomplished electronically. Addresses of word positions in the quick-access memory are designated by decimal numbers of the 4000, 5000, 6000 or 7000 series, referring respectively to the four loops. For convenient reference, these four loops are called the "4000", "5000", "6000", and "7000" loops or L_4, L_5, L_6, and L_7, respectively.

It is important to note that numbers or commands which have been block-transferred between main memory and the loops may be altered during the course of computations in one storage while remaining unchanged in the other. This feature often reduces the coding and operation time which would otherwise be required for restoring sequences of instructions or numbers to their original form in readiness for re-use during a subsequent phase of computation. If, however, alteration of commands or numbers is required, those which have been altered can be block-transferred or separately transferred to their original positions in storage.

Words recorded on the drum surface are destroyed only by having other words written over them. Therefore, information may be retained in memory overnight

Copyright 1954
by ElectroData Corporation

or during a power failure. However, information in the quick-access loops may be garbled during shutdown.

Internal Checking

The appearance of an unexpected arithmetic overflow (or unpredicted setting of the overflow toggle) will automatically stop the computer. In addition, the appearance of a forbidden combination of binary digits - - that is, one representing the numbers 10 through 15 - - at any one of several critical points in the computer, will also stop computer operations. These two internal checks serve to detect the majority of coding and computer errors. A sector alarm guards against garbling of information in the memory. At the instant of the origin pulse or zero pulse referred to earlier, the device which counts the 200 sectors of the drum is inspected to see that it registers zero. If it does not, the machine stops.

Extensive provisions are made for marginal checking of the computer during scheduled preventive maintenance periods. This type of check, made while running special test problems, will assist the operator in locating and replacing marginal components which could cause computing errors within the next few hours of normal operation.

Registers, Counters, and Controls

The command list (to follow) describes, in summary form, the operations automatically carried out by the computer. These descriptions make reference to various electronic registers whose functions are briefly as follows:

A Register: This register contains 11 decimal-digit positions. Ten are for the number or instruction and one for the

Copyright 1954
by ElectroData Corporation

algebraic sign. This register acts as an accumulator and the results of most operations appear here at the end of the operation.

R Register: This register contains 10 decimal-digit positions (no algebraic sign). In several operations, it serves as an extension of the A register, holding the 10 least significant digits of the number contained in the combined A and R registers.

D Register: This register contains 11 decimal-digit positions. Ten are for the number and one for the algebraic sign. All words entering the registers from an input medium or from the drum, first pass through the D register. Although it is not explicitly used in coding, the D register contains one of the operands during an arithmetic operation.

B Register: This register contains four decimal-digit positions, and is used to facilitate the modification of commands and for tallying. As each command is received from memory, it is checked to determine whether its address is to be modified or not. This is determined by a 1 or a 0 in the algebraic sign position. If the sign digit is 1, the four-digit number contained in the B register is added absolutely to the four least significant digits (the address part) of the command, and the command is then executed as modified by

Copyright 1954
by ElectroData Corporation

the contents of the B register. The usefulness of the B register will be pointed out more fully later, but it is important to note that the commands with negative sign, as stored in the memory, are not altered by the operation of this register; they are temporarily modified in the electronic registers immediately before execution. Thus, the same command may be executed many times during a computation, temporarily modified each time by a different number in the B register.

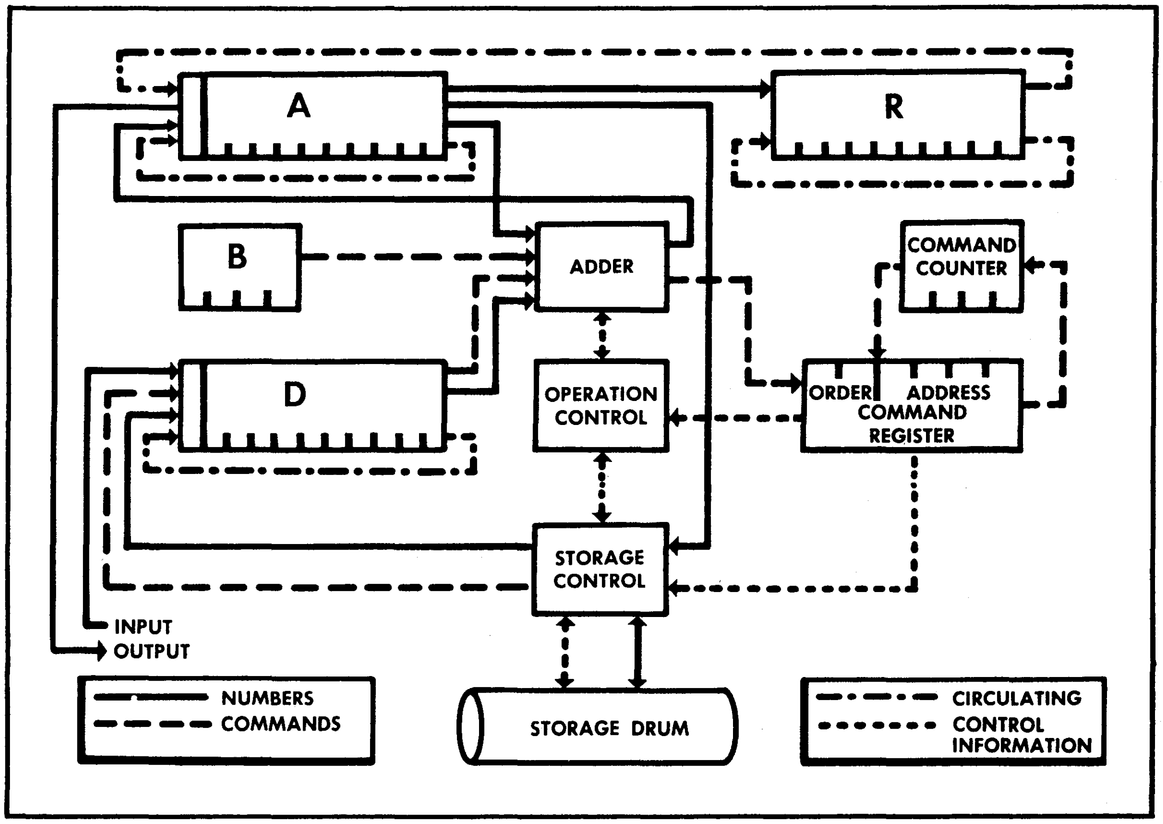
C Register: This register, also called the command register, contains 10 decimal-digit positions. The first two positions contain the order, and the next four contain the address of the operand, if there is an operand. The last four positions are the control counter, which holds the storage address of the command to be executed following the command now in the first six C-register positions (order register and address register),

All information entering the command register is treated as a command. This is the only criterion by which the computer differentiates between commands and numbers.

Special Counter: The special counter is a binary-digit counter that is used

1) to count the number of shifts in multiplication and division

Copyright 1954
by ElectroData Corporation



Copyright 1954 by ElectroData Corporation

2) to count the number of left shifts necessary to normalize the contents of the A register (see spl y command).

Adder: All arithmetic operations are performed in the **Adder**. Also, input and information from the drum destined for the A, B, or C registers pass through the adder. For a complete discussion of adder operation, during addition refer to Section 3.

Operation Control: The operation control senses

1) the order in the command register

2) whether or not an operand is to be brought to the D register.

Storage Control: The storage control regulates the flow of information to and from the storage drum.

A word entering the computer from an input medium goes first through the D register and the adder. If, at this moment, a word is to be interpreted as a tape command (see <u>tape commands</u>), then the adder shunts the word into the C register for execution as a command. Otherwise, the word is sent through the A register and then to the drum.

In the normal sequence of events, the computer fetches the command from the location specified by the control counter, and increases the content of the control counter by 1. When the command is fetched, it is brought to the

Copyright 1954
by ElectroData Corporation

D register through the adder and then to the command register. The command is modified in the adder, if necessary, by the contents of the B register. Now the computer is ready to execute the command.

Full discussion of fetching and executing of commands is found in the discussion of the timing toggle in Section 7?.

Words from the drum, destined for the A and B registers, are sent to the designated register during the execution of the appropriate commands for loading the registers. The path is from the drum to the D register, through the adder, and then to either the A or the B register.

The R register may be loaded from the drum by means of a right shift through the A register.

Information to be punched out on paper tape or printed on the typewriter comes from the A register.

Overflow Toggle and Change of Control

The overflow toggle, referred to in the descriptions of several of the commands, will be set under any of the following conditions:

1) Arithmetic overflow (i.e., a result equal to or greater than 1 in absolute value) during the execution of extract*, addition or subtraction commands, round-off, or adding or subtracting the contents of the special counter into the A register.

* The extract command can cause overflow only if used with digits other than 0 and 1.

Copyright 1954
by ElectroData Corporation

2) Failure of division (i.e., the absolute value of the divisor equal to or less than that of the dividend) during the execution of a division command.

3) Detection of difference between the algebraic signs of the numbers in the A register and that in the memory location during the execution of a "sign compare" command.

Any of the commands which effect conditional change of control, even if not specifically a "cc", will momentarily set the overflow toggle, if that transfer is completed. For example, "zero check", "decrease B", and "special left" commands.

The overflow toggle can be set during the execution of any one of the commands included above. Often the coder will deliberately use one of these commands to compare the number in the A register with some known number and will wish to select one of two alternate sequences of operations in accordance with the results of this comparison. In other cases, the coder may predict the possibility that the overflow toggle may be set during the execution of one of these commands, and wish to change from the normal sequence to some alternate sequence of commands only if this occurs. In either case, the command which may cause the overflow toggle to be set will be followed by one of the "conditional change-control" commands. If the overflow toggle is set, the conditional "change-control" command will be executed; if not, it will be skipped.

At times, however, the coder may not predict the possibility that the overflow toggle may be set during the execution of the commands listed above, or may not consider it worth while to provide an alternate sequence of commands.

Copyright 1954
by ElectroData Corporation

In such cases setting of the overflow toggle, not followed by a "conditional change-control" command, will indicate coding or scaling trouble. Accordingly, if a command which causes the setting of the overflow toggle is <u>not</u> followed by a "conditional change-control" command, the computer will stop and set off an alarm.

Copyright 1954
by ElectroData Corporation

COMMANDS

Arithmetic Commands

Modular Considerations of the Address Parts of Commands. In order to fully understand the operation of the commands, it is necessary to understand the modular correspondence which is involved.

If a number is congruent, or corresponds, to a second number, modulo n, then the difference between the two numbers, divided by the modulus, n, is an integer; in mathematical terms,

$$a \equiv b \bmod n$$

$$\text{If } \frac{a-b}{n} = k, \text{ where } k = \text{integer}$$

For example, $26 \equiv 2 \bmod 6$ or $103 \equiv 3 \bmod 5$.

The address part of most commands is interpreted modulo 8000. Thus 9267 is interpreted as 1267 and 2045 is interpreted as 2045. However there are several important exceptions which are noted in the description of the commands.

In all arithmetic operations the operands are stored in the A and D registers, the number in D having been fetched from the memory location specified in the address part of the command word.

Addition

Add (ad x, 74). Add to the number in the A register the number in memory location x. This command and associated add commands do not affect the contents of the R register.

Subtract (su x, 75). Subtract from the contents of the A register the number in memory location x.

Copyright 1954
by ElectroData Corporation

Add Absolute Value (ada x, 76) Add to the A register the number in memory location x with a positive sign attached.

Subtract Absolute Values (sua x, 77) Subtract from the number in the A register the value of the number in memory location x with a positive sign attached.

Clear and Add (ca x, 64), Clear and Subtract (cs x, 65), Clear and Add Absolute Value (caa x, 66), and Clear and Subtract Absolute Value (csa x, 67). These commands have the same effect as the corresponding commands without the "clear" except that they clear the A register before the addition.

Overflow

The computer considers all numbers to be less than 1 in absolute value. Since two numbers less than 1 may have a sum equal to or greater than 1.0000000000 (and less than 1.9999999999) in absolute value, the machine indicates the presence of a 1 or a number greater than 1 in the sum by setting the overflow toggle. The fractional part of the sum is retained in the A register.

Sign of Zero

All words have signs and zero words are no exception. When the sign compare order (sgc x) is used, it becomes important to know the manner in which the sign of zero is determined.

When the zero is a product or quotient, its sign is the algebraic result of the operation.

When the zero is the result of an addition or subtraction, its sign is the algebraic product of the sign of the operation and the sign of the

Copyright 1954
by ElectroData Corporation

operand; schematically: $\begin{array}{cccc} (+x) & (+x) & (-x) & (-x) \\ \underline{-(+x)} & \underline{+(-x)} & \underline{-(-x)} & \underline{+(+x)} \\ -0 & -0 & +0 & +0 \end{array}$, where the upper number is in the A register.

Adder Operation

All arithmetic operations are performed in the adder by means of serial addition, which is to say that addition takes place one decimal place at a time, the four binary components of the two digits being added in one operation. The number transferred from memory to the D register for addition to the number in the A register may be modified as a result of the execution of the variants of the add operation which are listed below:

ad x	The sign of the number in the D register is unchanged.
su x	The sign of the number in the D register is reversed.
ada x	The sign of the number in the D register is set to plus (0).
sua x	The sign of the number in the D register is set to minus (1).

A detailed discussion of the mechanism of addition may be of interest, and is given below.

Two toggles are used in conjunction with the adder during the addition cycle, the sign toggle and the complement toggle. The function of these toggles will be understood from the following description. After the number in the D register has the appropriate sign, the machine executes the following sequence of operations to perform the addition:

1. Compare the signs of A and D. If they are different, set the complement toggle; if they are the same, do not.

Copyright 1954
by ElectroData Corporation

2. Transfer the sign of D to the sign toggle.

3. Clear the signs of A and D.

4. Inspect the complement toggle. If it is set, replace A, including sign, with the nines complement plus $1 \cdot 10^{-10}$. If it is not set, do not change A.

5. Add A and D including sign position.

6. If the result has in the sign position

   0, transfer the number in the sign toggle to the sign of A and addition is complete.

   1, set the overflow toggle and transfer sign toggle to the sign of A and addition is complete.

   (These two cases take two word-times.)

   9, recomplement A, reverse the state of the sign toggle, and add the cleared D register to A. (The last case takes four word-times.)

Example:

The A register contains 1.2222222222, the command is "subtract," and the number going into the D register is 0.5555555555. The subtract command changes the D register to 1.5555555555. Since ASG and DSG are now the same, the complement toggle is not set. Since DSG is 1, the sign toggle is set to 1. In the registers, both 1's (ASG and DSG) are cleared. After the add operation, the A register contains 0.7777777777. Since the sign toggle contains 1, the A register is changed to 1.7777777777.

The A register contains 0.2222222222, the command is "add," and the number going into the D register is 0.8888888888. DSG is not changed for an add command. Since ASG and DSG are the same, the complement toggle is not set,

Copyright 1954
by ElectroData Corporation

and since DSG is 0, the sign toggle is set to 0. ASG and DSG are cleared, which in this case produces no change. After the add operation, the A register contains 1.1111111110. Since DSG is 1 at the end of the add operation, the overflow toggle is set. Then ASG is replaced by the content of the sign toggle, in this case 0. If the sign toggle had held a 1, ASG would have remained 1.

The A register contains 1.8888888888, the command is "add absolute," and the number going into the D register is 1.2222222222. "Add absolute" changes the D register to 0.2222222222. Since ASG and DSG are now different, the complement toggle is set to 1. Since DSG is 0, the sign toggle is set to 0. Since the complement toggle is set, the A register is changed during the addition, digit by digit, so that it is effectively the nines complement plus $1 \cdot 10^{-10}$, or 9.1111111112. After the add operation, the A register contains 9.3333333334. The sign toggle is reversed to 1. Since the complement toggle remains on, the second add operation, initiated by the 9 in ASG, complements the A register digit by digit, adding $1 \cdot 10^{-10}$. The D register is zero after the first operation, since its right shift is non-circulating. After the second add operation, the A register contains 0.6666666666. Since the sign toggle contains 1, the A register is changed to 1.6666666666.

With this information it is easy for the reader to check the sign-of-zero listing given earlier.

Multiplication

Multiply and Hold(mh x 60)Multiply the number in D by the number in the A register; and hold the 20-digit product in the A and R registers. In the multiply-hold command, R is considered to be an extension of A. Before

Copyright 1954
by ElectroData Corporation

any multiplication command the R register is automatically cleared.

Multiply and Round Off (mr x 70) Multiply the number in the D register by the number in A and round off. If the first digit of R is 5 or greater, increase the absolute **value** of the number in the A register by 10^{-10}. If the first digit of R is less than 5, do not change A. In either case the R register is cleared after the command has been executed. There is no possibility of overflow upon execution of a multiply and round off command.

Round Off

Round Off(**ro** 23) If the first digit of R is 5 or greater, add 10^{-10} to the absolute value of the number in the A register and clear R. If the first digit of R is less than 5, clear R. Overflow is possible if A contains all 9's and the first digit of R is 5 or greater.

Division

Divide(**div** x 61) Divide the number in the A and R registers by the number in memory location x. In this case as well as in the result of the mh command, the R register is considered as an extension of A. If the number in A and R is less in absolute value than the number in x, the division is performed; the A register will contain the 10-digit quotient, the R register the true remainder after the divide command has been executed. If, however, the number in A and R is greater in absolute value or equal to the number in the D register, the overflow toggle is set; A and R are cleared.

The setting of the overflow toggle indicates only that the numerator

Copyright 1954
by ElectroData Corporation

in the proposed division was greater than or equal to the denominator in absolute value, indicating that the quotient is greater than or equal to 1. Since the magnitude of the overflow is not known as it is in addition, both registers are cleared. With proper scaling such overflows can be avoided. Unlike the multiply commands, a divide command does not cause the preliminary clearing of the R register. Therefore, when the R register contains a number of no importance to a division, the coder should clear R before executing the divide order.

Clear R

Clear R(cl R 33) Clear the R register.

Unit Adjust

Unit Adjust(ua 06) Set the "1" toggle in the first position of the A register.

This command effectively increases an even digit in A1, making it odd, and leaves an odd digit unchanged. This command is normally used following a right shift to compensate for overflow on an addition. It allows the coder to compensate for the overflow without testing for sign.

A1 before ua	A1 after ua
0	1
1	1
2	3
3	3
4	5
5	5
6	7
7	7
8	9
9	9

Copyright 1954
by ElectroData Corporation

Storage Commands

The results of arithmetic operations are stored in the A register. The transfer of these results to memory is effected by two commands which do not affect the R register. They are:

To Memory and Hold (**tmh x 12**) Store the number in the A register in memory location x, and retain the number in the A register.

To Memory and Clear (**tmc x 02**) Store the number in the A register in memory location x, and clear the A register, including the sign position.

Shift Commands

For scaling, command modification, interpretation, etc., it is sometimes necessary to relocate the digits in the A and R registers. This can be done with shift commands. Shift commands are necessary for transferring information from the A to the R register. During many operations the R register is not disturbed and it may be used for temporary storage by shifting (A) into R. The shifting commands along with the extract command can be used to divide a word into component parts. For example, the mantissa and exponent index of a word in floating-point computation must be separated before arithmetic operations are performed with that word. Shift commands make it a simple matter to multiply or divide by powers of ten. Special shift commands have been incorporated for the normalization of numbers and the computation of exponent indices for floating-point operation.

Shift Right (**sr n 13**) Shift the number in the A and R registers, not including sign, n places to the right (n is interpreted mod 20). The n least

Copyright 1954
by ElectroData Corporation

significant digits are lost. The first n significant digits become zero. In this command as well as the following command, the number of shifts (n) is indicated by the address part of the command word.

Shift Left(**sl n 14**) Circulate the number in the A and R registers, not including sign, n places to the left (n is interpreted mod 20). The n most significant digits in A become the n least significant digits in R.

Logical Cycle Left(**lcl n 01**) Circulate the A register, including the sign digit, n+1 places to the left, where n is interpreted mod 20. The R register is not affected by this command.

Special Left Shift(**spl y 15**) If the number in the A register is not zero, shift the A and R registers to the left until the first non-zero digit in A is in the first position of the A register. Change of control does not take place. If the A register contains zero, replace the contents of the A register by the contents of the R register and change control to memory location y.

This command normalizes the number in A and R. It is frequently used in floating-point operations. For further discussion of floating-point arithmetic see Section 6.

Since it is important to know the number of shifts that occur as a result of the special left shift, a count of these shifts is stored in the special counter.

Special Counter Commands

The special counter is a four-toggle binary counter recycling on 15. It is used in the machine for three purposes:

Copyright 1954
by ElectroData Corporation

1. To count the shifts during multiplication
2. To count the shifts during division
3. To count the shifts necessary to normalize a number with **a special left.** The number in the special counter after the completion of a multiply or divide command is 9 (except when division sets the overflow toggle). After a special left (spl y) command the special counter shows the number of zeros in A to the left of the first non-zero digit; in the case where A = 0, the special counter counts to 10. The number in the special counter is retained until a multiply, divide, or another special left has been executed. In all of these cases it is cleared before the new number is added to it. Information may be obtained from the special counter by means of two commands.

Add Special Counter(**spc+ 16**) Add (algebraically) the number in the special counter scaled 10^{-10}, to the number in the A register.

Subtract Special Counter(**spc- 17**) Subtract (algebraically) the number in the special counter scaled 10^{-10}, from the number in the A register.

As in the addition commands, there is a possibility of overflow. Also, the special counter can contain a forbidden combination. In such a case, the execution of a spc$\pm$ command will signal the forbidden combination alarm, and the computer halts.

Extract Command (**ex x 63**) The extract command as it is normally used will be considered first. In this case, memory location x will contain a combination of 0's and 1's. This number is referred to as an extract combination.

Copyright 1954
by ElectroData Corporation

For the digits of the extract combination that are zeros, replace the corresponding digits in the A register by zeros. For the digits of the extract combination that are ones, leave the corresponding digits in the A register unchanged. For example, if the A register reads

$$(A) = -6754902387$$

and the extract combination in memory location x reads

$$(x) = +0001111010,$$

then the number in the A register after the execution of the extract command is

$$(A) = +0004902080.$$

If the extract combination is not restricted to 0 and 1, the execution of the command is an addition according to the following rule:

If an extract digit is even, then the digit corresponding to it in the A register is replaced by the extract digit.

If an extract digit is odd, then the digit corresponding to it in the A register is added to the extract digit minus one.

The result replaces the A register digits; i.e., $(A_1)_a = (A_1)_b + (x_1) -1$. If this sum is greater than or equal to 10, a carry occurs, which will be added to the A register digit to the left of the digits whose "addition" resulted in a carry. It is possible that certain extract combinations can cause the overflow toggle to be set, e.g.

$$(A)_b = +.7963828465$$

$$(x) = -.7589910674$$

$$(A)_a = 1.4392620724$$

Copyright 1954
by ElectroData Corporation

The extract combination containing only 0's and 1's is merely a special case of the generalized extract combination, and follows the same rule of "addition."

<u>Loop Transfer Commands</u>

<u>Block Transfer into a Loop (bl(4)<---x 34)</u> Block transfer into the 4000 loop the 20 words at memory location x through memory location x + 19, the main memory and loop addresses corresponding modulo 20. The commands <u>bl(5)<---x</u>, <u>bl(6)<---x</u>, <u>bl(7)<---x</u>, are executed in the same way as the above command, the 20 words being blocked into the 5000, 6000, and 7000 loops respectively. Because it is not possible to block transfer from one loop to another, all blocking commands are interpreted modulo 4000. Information transferred from main memory into a high-speed loop remains in main memory.

<u>Block Transfer into Main Memory (bl(4)--->x 24)</u>Block transfer into 20 main memory positions starting at memory location x the words in the 4000 loop. The words will go into main memory locations corresponding modulo 20 to their loop addresses. bl(5)--->x, bl(6)--->x, bl(7)--->x, commands are defined as above with the exception that we substitute 5000, 6000, and 7000, respectively, for 4000. After execution of a block transfer command, the information is retained in the loop until new information is **written** in.

<u>Change Control Commands</u>

<u>Unconditional Change of Control(cu y 20)</u>Change control unconditionally to memory location y. This command replaces the contents of the control counter with the address part of the cu command.

<u>Conditional Change of Control(cc y 28)</u>If the overflow toggle is set, change

Copyright 1954
by ElectroData Corporation

control to y and reset the overflow toggle to 0. If the overflow toggle is not set, ignore this command.

<u>Control Block Transfer</u>

The two commands listed under this section and two of the commands in the following sections are a combination of change of control commands and block transfer.

<u>Block and Unconditional Change of Control (cub y 30)</u> Block transfer into the 7000 loop the words at 20 consecutive locations starting at main memory location y, and change control to the image y in the 7000 loop; y is interpreted modulo 4000. It is important to note that the words occupy locations in the 7000 loop which are congruent modulo 20 to their previous main memory addresses. Change of control is effected by replacing the first two digits in the address part of the command word by 70 and placing this in the control counter. This command therefore automatically changes control to the 7000 loop.

<u>Block and Conditional Transfer of Control (ccb y 38)</u> If the overflow toggle is set, this command acts in the same way as the cub y command described in the previous paragraph. If the overflow toggle is not set, this command is ignored.

Since the control counter performs the operation of counting up one only after the command word has been fetched, it can be seen readily that the cub command must replace the first two digits by 70, not merely replace the first digit by 7. If, say, a cub 2999 were executed and the first digit only were replaced by 7, one command execution would cause a

Copyright 1954
by ElectroData Corporation

control change from the 7000 loop into main memory.

<u>Control-Record Commands</u>

<u>Unconditional Change of Control and Record (cuR y 21)</u> Clear the R register, replace its four most significant digits with the contents of the control counter, and change control unconditionally to memory location y. This command records the address from which a departure into a subroutine has been made, and makes it possible to provide in advance for a control change to the next address in the main routine.

<u>Conditional Change of Control and Record (ccR y 29)</u> If the overflow toggle is set, clear the R register, replace its four most significant digits with the contents of the control counter, and change control to memory location y. If the overflow toggle is not set, ignore this command.

<u>Block, Unconditional Change of Control and Record (cubR y 31)</u> Clear the R register and replace its four most significant digits with the contents of the control counter; block transfer the words in memory location y and the following 19 locations into the 7000 loop and change control unconditionally to the number corresponding to y in the 7000 loop.

<u>Block, Conditional Transfer of Control and Record (ccbR y 39)</u> If the overflow toggle is set, effect a cubR y command. If the overflow toggle is not set, ignore this command.

The record commands are used for subroutine entry and exit. For further details see Section 6.

Note that all block transfer commands are interpreted modulo 4000. The y's in the commands that perform a blocking operation are interpreted

Copyright 1954
by ElectroData Corporation

modulo 4000, while those in other commands are interpreted modulo 8000. These commands do not disturb the contents of any register other than the C register.

Zero Check (z y 04) If the A register does not contain ±0 before the execution of this command, change control to memory location y. If the A register contains ±0, ignore this command (-0 is changed to +0).

Sign Compare (sgc x 73) If the sign of the A register is the same as the sign of the word in memory location x, do not set the overflow toggle. If the sign of the A register is not the same as the sign of the word in memory location x, set the overflow toggle.

The sign compare command is usually followed by a cc, ccR, ccb, or ccbR command. If it is not, the setting of the overflow toggle will cause the machine to stop. It is only because of the sgc x command that the programmer need know the sign of zero.

B-Register Commands

Replace B (Bx 72) Set the B register to the last four digits in main memory location x. The sign of the number in x has no bearing on the setting of B.

Increase B (B+ 32) Add 1 to the contents of the B register.
If the B register before the execution of a B+ command contains 9999, it will contain 0000 after the execution of the command.

B to A (B-->A) Clear the A register and replace its four least significant digits with the contents of the B register. The B register is not changed by this command.

Decrease B (B- y 22) If the B register before the execution of this

Copyright 1954
By ElectroData Corporation

command did not contain zero, subtract 1 from the contents of the B register and change control to memory location y. If the B register before the execution of this command contained zero, ignore this command. In the latter case the command leaves the B register set to 9999.

Stop (s 08) Halt the machine. The operator can continue his program by pressing the **CONTINUOUS button.**

Paper Tape and Keyboard Input Commands

The INPUT SELECTOR switch may be set to OPTICAL READER, MECHANICAL READER (on the Flexowriter) or KEYBOARD.

Input (in x 00) Start reading the first word on the tape or received from the keyboard into memory location x. Read following words into the next consecutive locations. (See Special Use of the Sign Column)

Single Digit Add Input (dA 10) Add to the contents of the A register in the A10 position the positive value of the number punched on the keyboard or read by the mechanical reader. When the command is executed, the machine will stop until a digit has been entered, after which it resumes operation.

Special Use of the Sign Column Words containing numbers other than 0 or 1 in the sign column are used for control of the input device and for B modification of words during input. A 2 or 3 placed in the sign column will cause the address part of the word to be modified by the B register as the word is read into memory. In memory, the word will have a positive sign if there was a 2 in the sign column, a negative sign if there was a 3 in the sign column. In both cases, the word will be stored with the contents of the B register added to its address part.

Copyright 1954
by ElectroData Corporation

This is summarized in the following table:

Sign Column	B Modification Before Execution	B Modification On Input	Comments
0	no	no	
1	yes	no	
2	no	yes	Sign appears in memory as 0.
3	yes	yes	Sign appears in memory as 1.

A tape command is a command which appears only on the tape and does not enter the memory of the computer. It is used to control the input and is read directly into the command register and executed. This type of command is indicated by the presence of a 4, 5, 6, or 7 in the sign column. The effect of each of these numbers with cu or STOP appears in the table below. With the "in" command, a 6 or 7 in the sign column does not stop the input device.

Sign Column	Tape Stops or Keyboard is Deactivated	B Modification Before Execution
4	no	no
5	no	yes
6	yes	no
7	yes	yes

The commands most often used with a tape control digit in the sign column are cu, cub, in, and stop.

Flexowriter and Paper Tape Output

There are two printout commands. With either of them, the second digit of the address part is a coded format instruction to the Flexowriter, Flexowriter

Copyright 1954
by ElectroData Corporation

punch or console punch. (The two punches carry along the format instruction as an information digit for the typewriter control. It does not affect format until the printing operation is carried out.) These instructions, which are sensed and carried out before any other part of the command is executed, are as follows:

Second Address-Digit	Instruction
0	None.
1	Feed out one character-space of blank tape (10 to the inch). (This instruction has no effect on the Flexowriter.)
2	Print decimal point and suppress sign digit.
3	Suppress sign digit and substitute a space for sign digit when the word is printed.
4	Translate alphanumerically, two A-register digits per alphanumeric character.
5	Actuate carriage return.
6	Actuate tab key.
7	Stop printout. Idle the computer if any printout command comes up before the typewriter-control RESET is pressed.
8	Actuate space bar.
9	None.

Print Out (po n, 03) This command prints out the sign and n digits of the A register (unless the sign has been suppressed by a 2 or a 3 format instruction), n being interpreted modulo 20, since a tens digit must be sensed to print out 10 digits. The R register is neither printed out nor

Copyright 1954
by ElectroData Corporation

shifted. The A register, including the sign, is circulated n + 1 places left. (This operation differs from that of the shift commands in that they do not shift the sign.)

The format-instruction digits make it possible to control format completely by proper construction of the computer command word. It is also possible to control format by means of cascaded counters in the typewriter control which may be set for WORDS PER LINE, LINES PER GROUP, and GROUPS PER PAGE, each setting from 1 to 20. (See Section 8.)

The po n command for any one word may inhibit the advancement of the words-per-line counter by using an 8 as the first address-digit; this in effect cuts out the external format control. The format control will resume its count and will actuate the tab, carriage return, and page stop according to the settings, at the end of the next printout command which does not have an 8 as the first address-digit.

Example:

Print out 50 words, beginning with the word in 2062 and ending with the word in 2013. Set the WORDS PER LINE selector at 5, the LINES PER GROUP selector at 10. The code for this is as follows:

Storage Address of Command	Command	Number
7054	B 7059 (0 0000 72 7059)	
7055	1 ca 2013 (1 0000 64 2013)	
7056	po 0010 (0 0000 03 0010)	
7057	B- 7055 (0 0000 22 7055)	
7058	STOP (0 0000 08 0000)	

Copyright 1954
by ElectroData Corporation

Storage Address of Command (continued)	Command	Number
7059		0 0000 00 0049

If the printout command in 7056 had been written 0 0000 03 8010, the format selector switches would have been inoperative and provision would have had to be made for internal format control.

Print Out (po f 07) The po f command does nothing but carry out the format instruction contained in the second address-digit. (The first address-digit is immaterial to this command, since po f cannot advance the words-per-line counter.) The second address-digit most frequently contains format-instructions 1, 5, 6, 7, or 8. It should be noted that format instructions 2, 3, and 4 will affect the next po n command; 2 and 3 will suppress the sign and 4 will cause alphanumeric printout in the next po n command.

Punch Out (po n 03; po f 07) The console punch uses the same two commands as the Flexowriter and typewriter control with the difference that the OUTPUT SELECTOR is set to TAPE. The console punch has no connection with the typewriter control except that its output tape may be an input to various arrangements of the typewriter control patch panel. (See Section 8.)

Alphanumeric Code Typewriter action corresponding to the various two-digit combinations read out from the A register is given on the following page. Alphanumeric information comes out on the console punch as pairs of decimal digits. The format instruction (4) accompanies the information so that when the tape is read for printing later it will be translated by the typewriter control into alphanumeric Flexowriter action.

Copyright 1954
by ElectroData Corporation

Typewriter Action L.C.	U.C.	Alphanumeric Code
a	A	20
b	B	61
c	C	62
d	D	63
e	E	64
f	F	65
g	G	66
h	H	67
i	I	70
j	J	71
k	K	72
l	L	36
m	M	73
n	N	74
o	O	75
p	P	76
q	Q	77
r	R	21
t	T	23
u	U	52
v	V	53
w	W	54
x	X	55
y	Y	56
z	Z	57

Typewriter Action L.C.	U.C.	Alphanumeric Code
0	)	40
1	1/2	41
2	&	42
3	/	43
4	$	44
5	%	45
6	?	46
7	!	47
8	*	50
9	(	51
+	=	24
-	_	25
;	:	26
,	,	31
.	.	32
'	'	33
lower case		27
upper case		30
space		34
color shift		35
ignore		00
back space		01
tab		04
carriage return		05
stop		07

Alphanumeric Code	Typewriter Action L.C.	U.C.
00	ignore	
01	backspace	
04	tab	
05	carriage return	
07	stop	
20	a	A
21	r	R
23	t	T
24	+	=
25	-	_
26	;	:
27	lower case	
30	upper case	
31	,	,
32	.	.
33	'	'

Alphanumeric Code	Typewriter Action L.C.	U.C.
50	8	*
51	9	(
52	u	U
53	v	V
54	w	W
55	x	X
56	y	Y
57	z	Z
61	b	B
62	c	C
63	d	D
64	e	E
65	f	F
66	g	G
67	h	H
70	i	I

Copyright 1954
by ElectroData Corporation

Alphanumeric (cont.) Code	Typewriter Action L.C.	U.C.	Alphanumeric Code	Typewriter Action L.C.	U.C.
34	space		71	j	J
35	color shift		72	k	K
36	l	L	73	m	M
40	0	)	74	n	N
41	1	1/2	75	o	O
42	2	&	76	p	P
43	3	/	77	q	Q
44	4	$			
45	5	%			
46	6	?			
47	7	!			

Since it takes two digits to represent one alphabetic character, a po 0406, written 0.0000030406, will print out three alphabetic characters from the A register.

Example: If (A) = +.7065464646, and the command mentioned in the above paragraph is executed, the following characters will be printed out: if6.

Punched Card Commands

Card Input ((1000-m) ci x 44) Read in m cards starting at memory location x. If x is less than 8000, the address of the command is interpreted modulo 4000 (words are read into main memory). If x is greater than or equal to 8000, the cards are read into the 4000 loop, the starting location in the loop corresponding modulo 20 to x. The three most significant digits of the command word are 1000-m. When the command is fetched, it is placed in both the A register and the C register. This means that any useful information in A before the execution of this command must be stored. After the execution of the command, the A and C registers will contain the original command modified so that the address part holds the address of the first word on the last card and the first three digits are zero. This command

Copyright 1954
by ElectroData Corporation

also destroys the previous contents of the 4000 loop. After its execution, the 4000 loop will contain the last 20 words read in from the cards. Card-input information is always read into the 4000 loop. It is then block transferred into main memory, if the address of the command word is less than 8000. If the address is greater than or equal to 8000, it remains in the 4000 loop. This means that if k represents the number of words per card, km + 20 - k locations have been written in main memory starting at location x, provided that the address of the command word is less than 8000. The number of words per card, k, is set on an input selector switch on the converter unit where $1 \leq k \geq 8$.

<u>Card or Tabulator Output ((1000-m) co x 54)</u> Punch out m cards, starting with the word at memory location x. If x is less than 8000, this command is interpreted modulo 4000. If x is greater than or equal to 8000, the information is punched directly from the 4000 loop, starting at the position corresponding modulo 20 to x. The three most significant digits of the command word contain 1000-m.

The A register is cleared upon the execution of this command and will contain the command with the number of words read in added to the address part of the command word.

Example: If k is the number of words per card, and the command to be executed is 0.7473 54 2011, then 253 cards will be punched out, k words per card, starting at memory location 2011. If we let k = 2, then 1494 words will have been read out after the command has been executed. The A register will appear as 0 0003 54 3505. The six most significant digits

Copyright 1954
by ElectroData Corporation

of the C register will be 543505. The 4000 loop will contain the block of 20 words starting at x+k(m-1), or in this case at 3503. k is set on a selector switch.

For tabulator output the command is the same except that the words "line" and "printed" are substituted for the words "card" and "punched" respectively.

Copyright 1954
by ElectroData Corporation

TIMING

The basic consideration governing timing is that orders and operands can be obtained from or returned to memory only at those instants when the designated cell passes under a reading or writing head. Thus, in carrying out a program stored in main memory, the actual time required to carry out each order is usually irrelevant, because this time is nearly always small compared with the time required to bring the next order under the read head. A long execution time is in this case compensated for by a subsequent short wait. However, with the program in the high-speed loops, timing becomes more significant and it is for this reason that the following description has been included.

As previously defined, a sector is 1/200 part of the drum and consists of 20 memory locations, whose addresses are congruent modulo 200, on the 20 bands of the drum. See illustration on drum, Section 5. It will be convenient to number these sectors from 0 to 199, with each sector number being congruent modulo 200 to numbers of cells in the sector. The time required for one sector to pass under the read-write heads is called a word-time and is taken as the fundamental timing unit for the computer.

We can calculate the word-time in seconds since we know that the drum makes approximately 3570 revolutions per minute, or 59½ revolutions per second. Time required for one revolution is thus approximately .0168 seconds. One word-time would be 1/200 of this, or about 84 microseconds. The access time to any cell on the drum may vary from 0 to 199 word-times. For example, if the computer starts to look for cell 3136 at the beginning of sector 136, it will not have

Copyright 1954
by ElectroData Corporation

to search, but can immediately start reading or writing. The same applies to cells 2336, 2536, 2736 or any other cell in sector 136. On the other hand, if the computer starts to look for cell 1432 at the beginning of sector 33, it will have to search for 199 word-times. The average access time to any cell in main memory will be halfway between these extremes—that is, 100 word-times, or approximately 8.5 milliseconds.

However, since numbers and commands are usually stored in the high-speed loops during computation, a more realistic access time for the computer would be one obtained for the loops.

A number may be read from a high-speed loop at any time when a sector which is congruent to it modulo 20 is under the read head. For example, the number in cell 6013 may be read from the drum during sector 13, 53, ... 173, or 193. Since the number may be read every tenth part of a drum revolution, the maximum access time for the loops would be 1.7 milliseconds and the average access time approximately .85 milliseconds.

We define the <u>timing cycle</u> for commands as the interval between the appearance of a command location, c, under the reading head, and the appearance of the location of the next command to be executed, c', under the reading head. A timing cycle will be divided into four phases as illustrated below. Each step has been assigned a symbol to simplify description of examples given later.

Symbol	Phase Description
	At beginning of cycle drum is in position to read command into D.
α	Computer reads command into D register, modifies it, if necessary, as it passes through adder and then places it in C register. Here the command is interpreted and conditions are set up to

Copyright 1954
by ElectroData Corporation

execute the command or search for the operand. (Minimum time for α is four word-times.)

t_x	Time taken to do actual search for operand, (x).
β	Time taken to execute command and set up conditions to search for next command.
$t_{c'}$	Actual time spent in search for command c'.

Note that the drum is rotating throughout the whole timing cycle.

The total number of word times, T_c, necessary to execute the timing cycle for c will accordingly be given by

$T_c = \alpha + t_x + \beta + t_{c'}$, where

α is a function of the command at c.

β is a function of both the nature of the command and the magnitude of the operands.

t_x depends on the location x, but clearly is zero for those choices of the command at c which do not involve consulting memory for an operand, (x).

For convenience of description, orders are grouped into four classes.

Class I Orders

Orders in Class I have no operand, (x), therefore $t_x = 0$. In most cases, $\alpha + \beta$ is less than 21 word-times; however, this is the minimum time necessary to read successive commands from high-speed loops. For main memory, the figure is 201 word-times. In the following table T_c is given directly where possible or as a combination of $(\alpha + \beta)$ and $t_{c'}$.

Copyright 1954
by ElectroData Corporation

Order	T_c Loop	From Main Memory	Comment
ua	21	201	
B --> A	21	201	
sr n	21	201	
sl n	21	201	
spc +	21	201	
spc -	21	201	
ro	21	201	
B+	21	201	
cl R	21	201	
po f	21	201	
lcl n	21	201	If $n \leq 12$.
	41	201	If $n > 12$.
z y	21	201	If $A = 0$.
	$(9 + t_{c'})$		If $A \neq 0$.
B- y	21	201	If $(B)_b = 0$.
	$(9 + t_{c'})$		If $(B)_b \neq 0$.
cu x	$(8 + t_{c'})$		
cuR x	$(8 + t_{c'})$		
cc x	$(8 + t_{c'})$		If O.F.T.
	21	201	If no O.F.T.
ccR x	$(8 + t_{c'})$		If O.F.T.
	21	201	If no O.F.T.

Copyright 1954
by ElectroData Corporation

Order	T_c From Loop	T_c From Main Memory	Comment - continued	
spl y	21	201	If $n \leq 6$	n refers to number of places
	41	201	If $6 < n \leq 9$	before first non-zero digit
	$(28 + t_{c'})$		If $n = 10$	in A register.

Examples of Class I Orders:

1. From the instant a cl R order is under the read head in 6005, how many word times will it be before the next order can be read in? The answer is 21 word times, since even though the order does not take that long to execute, the drum must turn one-tenth of a revolution plus one word-time before the read head will be over cell 6006 again.
2. If $A \neq 0$ and we execute a z 3150 from cell 3167, what is T_c? Since $\alpha + \beta = 9$, we are ready to fetch the command from 3150 when the read head is over $(3167 + 9) = 3176$. It will be necessary to wait until drum has turned from this point to 3350 (3350 = 3150 modulo 200), or a period of 174 word-times. T_c is then $174 + 9 = 183$ word-times.

Class II Orders

Orders in Class II require a search for the operand (x). Because t_x and $t_{c'}$ vary depending on location of a particular command or operand, T_c cannot be given directly, but must be calculated for each separate case. $\alpha = 5$ for all Class II orders. β is given by the following table.

Copyright 1954
by ElectroData Corporation

$T_c = 5 + t_x + \beta + t_{c'}.$

Order	β	Comments
tmc x	4	
tmh x	4	
bl(4)->	23	
bl(5)->	23	
bl(6)->	23	
bl(7)->	23	
bl(4)<-	23	
bl(5)<-	23	
bl(6)<-	23	
bl(7)<-	23	
mh x	16 + 2n*	n* = sum of digits of A.
mr x	17 + 2n*	
div x	55 + 2n**	β = 27 if $\lvert(A)\rvert \geq \lvert(x)\rvert$ n**= sum of digits of quotient
ex x	6	
ca x	6	
cs x	6	
caa x	6	
csa x	6	
Bx	6	
sgc x	5	
ad x	6	β = 8 if $\lvert(A)_b\rvert > \lvert(x)\rvert$ and A-SG $\neq$ D-SG.

Copyright 1954
by ElectroData Corporation

Order		Comments - continued
su x	6	$\beta = 8$ if $\lvert(A)_b\rvert > \lvert(x)\rvert$ and A-SG $\neq$ D-SG. (See ADDER OPERATION, Section 3.)
ada x	6	
sua x	6	

* Taking 45 as an average value of n, $\beta = 106$ for mh x and 107 for mr x.

** Taking 45 as an average value of n, $\beta = 90 + 55 = 145$ for div x.

Examples of Class II Orders:

1. Compute the timing of tmc 6003, assuming command is located in 5013.

Phase	No. Word Times Elapsed	Position of Read-Write Heads over 6000 loop	over 5000 loop
START	0	6013	5013
α	5	6018	5018
t_x	5	$6023 \equiv 6003$ (mod 20)	5023
β	4		5027
$t_{c'}$	7		$5034 \equiv 5014$ (mod 20)

This makes a total of 21 word-times.

Class III Orders

Orders in Class III are similar to Class II orders except that $\alpha = 6$ instead of 5. [illegible]

cub x	23
cubR x	23
ccb x	23
ccbR x	23

These values are true if O.F.T. is on. If O.F.T. is not on, T_c = 21 word-times if c' is executed from high-speed loop and 201 if from main memory.

Copyright 1954
by ElectroData Corporation

Examples of Class III Orders:

1. How many word-times are necessary to execute a cub 1240 from 4032?

   Answer: If we assume sector 32 is under the read heads,

Phase	No. Word-Times Elapsed	Position of Read-Write Head	
		over 4000 loop	main memory
START		4032	1232
α	6	4038	1238
t_x	2	4040	1240
β	23	4063≡4023	1263
$t_{c'}$	17	7040	

Total, $T_{c'}$, = 48 word-times.

2. When executing the following sequence of commands, what cells in the 2600-2700 band may xxxx be and still cause no time to be wasted?

5008	caa	2600
5009	ada	xxxx
5010	mh	2773

   Answer: Assuming we start at sector 0188:

Phase	No. Word-Times Elapsed	Position of Reading Head	
		over 5000 loop	over 2600 band (or 2400)
START	0	5008	2588
α	5	5013	2593
t_x	7	5020≡5000*	2600
β	6	5006	2606
$t_{c'}$	3	5009	2609
α	5	5014	2614

*modulo 201

Copyright 1954
by ElectroData Corporation

At this point the computer starts searching for xxxx. The next command, c', will be under the read heads at 5030≡5010 mod 20. However, to allow the maximum time for β', the operand at xxxx must be found before 5022. Thus, in order not to waste time, xxxx should be between 2614 and 2622.

Class IV Orders

Drum timing is not a pertinent factor in Class IV orders since time per command is much larger than any drum time.

Order	Time
in x	500 decimal digits per second using photoelectric reader
	10 decimal digits per second using mechanical reader
po n	10 digits per second, Flexowriter printout or punch
	15 digits per second, console punch
STOP	Determined by operator
dA	Determined by operator
ci x	100, 150, or 200 cards (or lines) per minute depending on the
co x	IBM machine being employed.

See Section 8 for time in terms of input-output media.

Information from the above tables can be used to find the speed of subroutines when needed. However, for general program timing, it is more practical to have an average time figure for each order.

The average timing, based upon use of the high-speed memory for storage of commands and operands, including access time for procurement of the command and operand, and the actual operation time itself, is given in the following table:

Copyright 1954
by ElectroData Corporation

Operation	Average time per operation (rounded to tenth of millisecond).	Comments
ca x, caa x, csa x, cs x, ex x, B x, ad x, ada x, su x, sua x, tmh x, tmc x, sgc x	2.5	
mh x, mr x	10.8	
div x	14.0	
bl(n)->x, bl(n)<-x	11.5	n = 4,5,6,7 x is a main memory address
cu x, cuR x, cc x, spl y, B- x, ccR x, z y, clR, ro, ua, sl n, sr n, B+, B->A, spc±, lcl n.	1.8	
cubR x, cub x,	12.3	x is a main memory address
ccb x, ccbR x	12.3 1.8	If O.F.T. set If O.F.T. not set x is a main memory address

Each reference to main memory, other than those mentioned in the comments, increases the corresponding average t_x or t_c, by 90 word-times, or 7.6 milliseconds.

Copyright 1954
by ElectroData Corporation

CODING METHODS

Once a coder has grasped the basic concepts involved in the function of an internally programmed computer, the translation of mathematical symbols into the coded instructions for the computer is rather elementary. Elementary, that is, if we refer to the writing of a code that will cause the computer to solve a stated problem. It is not so simple a matter to write a polished code -- one that conserves machine operation time and storage. The process of coding for the ElectroData computer will lead the coder by necessity and intuition to coding methods and schemes which will help him polish his codes. This section introduces coding methods which are standard practice and are useful no matter what type of problem or computer application is being coded. It is hoped that this exposition will leave the coder's intuition and initiative free to develop coding methods not so well known.

Use of basic arithmetic commands in a computer with an internally stored program is very similar to the use of a desk calculator. To add two numbers, the accumulator register is cleared and the augend is put into it. Next, the addend is added to it and the result appears in the accumulator.

Similarly, in the ElectroData computer, two numbers are added by giving a "clear and add" command which clears the A register and puts the first number into it, and then an "add" command which adds the second number to what is already in the A register, the sum appearing in the A register.

The other arithmetic commands, such as subtract, multiply, and divide are executed similarly. One operand is put in the A register by one command or a

Copyright 1954
by ElectroData Corporation

series of commands, and the operation is performed by a second command which contains the address of the second operand.

Cycling

The internally-stored program computer exhibits its greatest usefulness in the fact that the program itself may be altered by suitable commands.

As a means of illustrating this, consider the problem of summing a series of numbers. Assume the numbers are stored in memory locations 0100 to 0149. With no altering of commands, the program would appear as follows:

Storage Address	Command
0500	ca 0100
0501	ad 0101
0502	ad 0102
0503	ad 0103
.	.
.	.
.	.
0549	ad 0149

Note: In this and the following example assume that the final sum is less than 1.

This coding is plainly very tedious to write and wasteful of space. If it were necessary to sum 2000 words in this way, half the total memory would be used for data and the other half for commands. Clearly a method is needed whereby the series of ad x commands is replaced by one such command used many times, the address being varied. The process of repeating a command or series of commands, altering the address, and then going back to the beginning, is called cycling.

In order to cycle the above example three things are necessary:

1. Alter the address of the ad x command.

2. Provide a means for returning to execute the command just altered.

3. Set up a means for stopping after cycling 50 times.

Copyright 1954
by ElectroData Corporation

For step 1, the A register must be used. However, it is now occupied by the partial sum we are computing. It will thus be necessary to store this partial sum while the ad x command is altered.

Before discussing points 2 and 3 let us write out the new cycling program, using some of the cells previous to 0500 as storage for constants.

```
                0498       00 00000000       Storage cell for sum.
                0499       0 0000 00 0001    Constant

   Start --     0500       ca 0100
                0501       ad [0101]
                0502       tmc 0498          Store sum temporarily
                0503       ca  0501          (A)a = ad x command
                0504       ad  0499          Add "1" to address of ad x
                0505       tmc 0501          Store altered ad x
                0506       ca  0498          (A)a = sum stored temporarily
                0507       cu  0501          Return to add next number
```

(The bracket around the address part of the command in cell 0501 is a convention used to indicate that that particular address is being altered by the program.)

While the above program will accomplish steps 1 and 2, it will proceed indefinitely: there is as yet no provision for ending the cycling. Obviously, what is needed is a method for counting the number of times the set of commands has been executed and stopping the computation when this reaches 49 (there are only 49 add ends left after the initial ca 0100.

This process of counting the number of passes through a series of commands and making a choice based on this count is known as tallying. One method of tallying on the ElectroData computer is to use the first three digits of the command word as follows:

Place 1000 minus n, the number of passes, in the first three digits and then

Copyright 1954
by ElectroData Corporation

add 1 to the third digit on each pass. On the nth pass, the A register will overflow. For example, for seven passes through an <u>ad</u> command this method would use + 9930 74 [0000]. This would be altered by adding +0010000001 to give +9940 74 0001 . The process would be continued for each pass until the one just before the last, when the command would appear as: + 9990 74 [0006].

The routine below provides for stopping the computer when the tally causes overflow. Cell 0508 could just as well have contained a cu command to return to a main routine.

	Cell Number	Spares	Order	Address	
	0498	0000	00	0000	Storage for sum
	0499	0010	00	0001	
START ---	0500		ca	0498	
	0501	9500	ad	[0100]	
	0502		tmc	0498	
	0503		ca	0501	
	0504		ad	0499	
	0505		cc	0508	
	0506		tmc	0501	
	0507		cu	0500	
	0508		STOP		

Note that the addition of the first number is here rendered like all subsequent additions by starting with zero as the first addend, and the code thus further compressed.

Cell 0505 is known as a <u>branch point</u> since at this point the program may take one of two paths depending upon whether or not the overflow toggle is set.

Operation of the C Register

To explain the above commands more fully it is necessary to consider the operation of the C register.

The first six decades of the C register are used to hold the command while

Copyright 1954
by ElectroData Corporation

it is being interpreted and executed. The last four decades, called the control counter, contain the address of the command to be executed next. The C register goes through two phases of operation for each command in the program. These are:

1. Fetch cycle: Command in cell designated by control counter is brought from drum and placed in first six decades of C register. Number in control counter is increased by one.

2. Execute cycle: The command, which is now in first six decades of C register, is interpreted and executed.

The computer will alternate between these two cycles during calculation.

The following table illustrates this operation of the C register during the running of the routine developed earlier.

Command from Cell Number	End of Fetch or Execute	C REGISTER READING			Comments
		Order	Address	Control Counter	
START				0500	
0500	F	ca	0498	0501	
	E	ca	0498	0501	$(A)_a=(0498)=0$ initially
0501	F	ad	0100	0502	
	E	ad	0100	0502	$(A)_a=(0498)+(0100)$
0502	F	tmc	0498	0503	
	E	tmc	0498	0503	
0503	F	ca	0501	0504	
	E	ca	0501	0504	$(A)_a=+9500\ 74\ 0100$
0504	F	ad	0499	0505	
	E	ad	0499	0505	$(A)_a=+9510\ 74\ 0101$
0505	F	cc	0508	0506	
	E	cc	0508	0506	Command ignored since O.F.T. not set.
0506	F	tmc	0501	0507	
	E	tmc	0501	0507	
0507	F	cu	0500	0508	
	E	cu	0500	0500	Address part placed in command counter.
0500	F	ca	0498	0500	
	E	ca	0498	0501	
0501	F	ad	0101	0501	
	E	ad	0101	0502	$(A)_a=(0498)+(0100)+(0101)$

Copyright 1954
by ElectroData Corporation

This sequence will continue in the C register until the last time through, when the following changes will occur beginning with cell 0503.

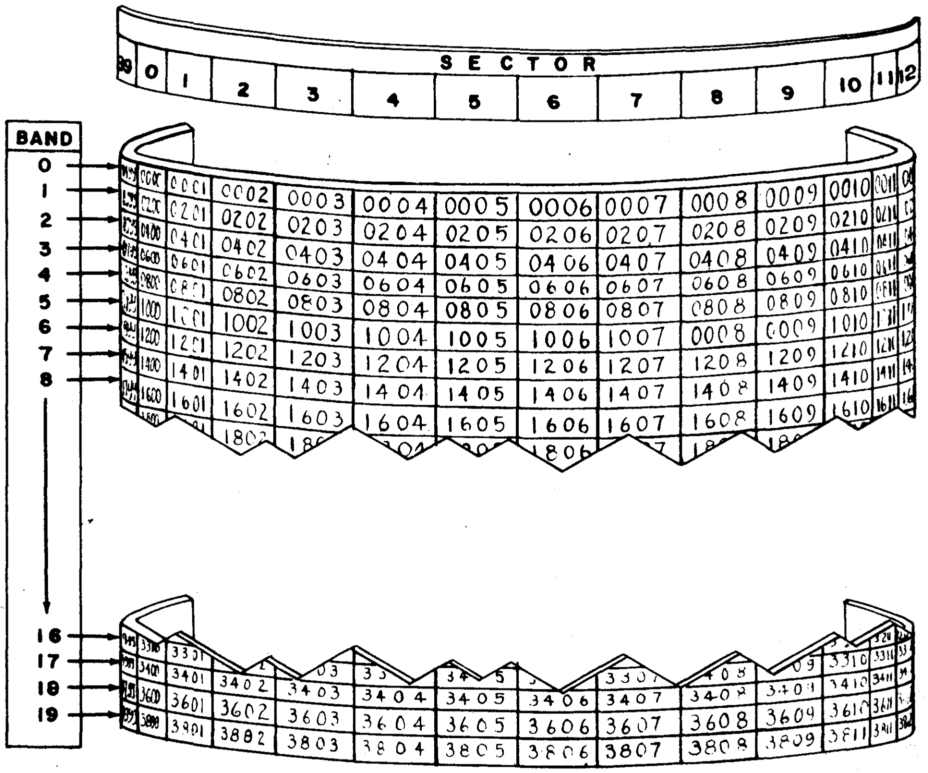
0503	F	ca	0501	0504	
	E	ca	0501	0504	$(A)_a$=+9990 74 0149
0504	F	ad	0499	0505	
	E	ad	0499	0505	$(A)_a$=+0000 74 0150; O.F.T. set
0505	F	cc	0508	0506	
	E	cc	0508	0508	Address part placed in command register.
0508	F	STOP	0000	0509	
	E	STOP	0000	0509	Computer Stops; $(0498)_a$= sum of 50 numbers.

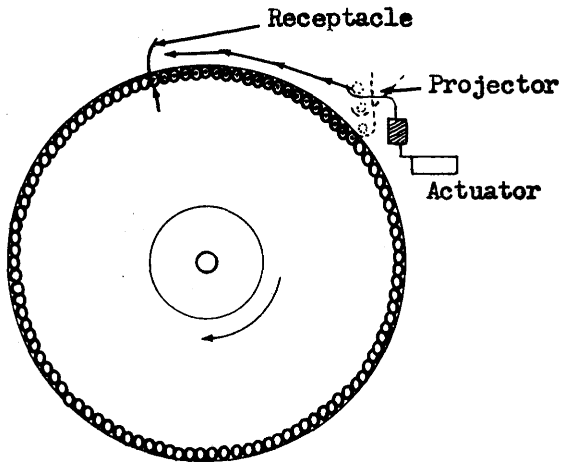
Note that on all commands other than cu commands and cc commands accompanied by an overflow, the C register has the same appearance in both execute and fetch cycles. The distinguishing mark is the setting of the timing toggle. If the FETCH light is on, the next operation the machine will perform is to fetch the word at the address designated by the control counter. If the EXECUTE light is on, the next operation the machine will perform is to execute the command in the first six decades of the C register.

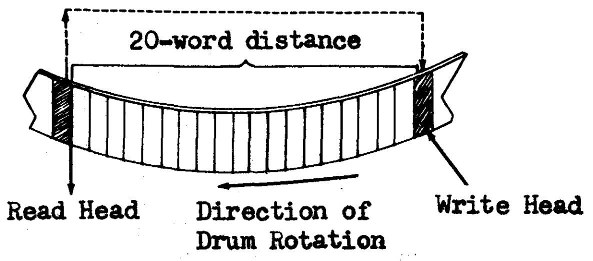
The computer recognizes a word as a command only when it is brought into the C register, and only a command whose address is brought into the control counter will be drawn into the C register. An address can enter the control counter via a change of control order (cu, cc, etc.) owing to its following a previous command that is not a change of control order. Commands which are altered in the A register are treated the same as data during this alteration. It is important to note also that if a part of the data should be brought into the C register, due to coding error, it would be treated as a command.

A more detailed discussion of C-register operation will follow after the use

Copyright 1954
by ElectroData Corporation







Copyright 1954 by ElectroData Corporation

of the quick-access loops has been described.

<u>Pre-storage and Entry Point</u>

In programming a long problem it is often necessary to repeat a previous section of a routine, This may be part of a cycling routine or simply a repetition of the section as a check, or a method of correcting errors. For example, when checking out a routine, the coder frequently finds that he needs to go back through part of the routine in order to locate a coding error. It is convenient not to have to read the routine in again.

In the example given previously, after all or part of the routine has been carried out, cell 0501 has been changed from its original value of +9510 74 0101 to any one of the alterations up to +0000 74 0150.

In order to do this section over, it is necessary first to <u>restore</u> cell 0501 to its original value. A convenient way of doing this is to pre-store the original content of the cell at a second memory location. Then by a short series of steps stored ahead of the main routine, cell 0501 can be restored. A prestorage routine for the example is:

```
Entry point ---      0494   ca            0497 }
                     0495   tmc           0501 } Pre-storage steps
                     0496   cu            0500 }
                            ________________
                     0497   +9500 74 0100
                     0498   [+0000000000]
                     0499   +0010000001
START ---            0500   ca            0498
                     0501   9500 ad [0100]  etc.
```

Whenever it is necessary to go through this section of the routine again, control is changed to cell 0494, where the original content of 0501 is restored before the section of the routine beginning with (0500)is carried out -- hence the name pre-storage.

Copyright 1954
by ElectroData Corporation

Cell 0494 is called an entry point to the routine since from here it can always be entered and executed in its original form.

Operational Features of Quick-Access Memory

Before discussing coding methods permitted and necessitated by the quick-access loops, it may be worth while to review the operational characteristics of this section of memory.

Opposite is a schematic representation of main memory storage in the ElectroData computer. Note that each word can be identified by a physical location on the drum surface. This location is specified by the band number and the sector number. There are 20 bands, numbered 0 through 19, each of which contains 200 word spaces designated as sectors 0 through 199. An assembly of four magnetic heads, each capable of both reading and writing, is positioned over each band. To read or write a given word in a band may require one complete drum revolution before the word is in position under the read-write head. One revolution of the drum requires approximately 17 milliseconds; hence, the average access time for a word in main memory is approximately 8.5 milliseconds.

Since each quick-access loop contains 20 words, it may be considered as one band with the same information repeated ten times.

Unlike the main memory bands, which have one read-write assembly per band, each quick-access loop has a reading assembly and a writing assembly spaced 20 words apart. (See opposite) Also, instead of reading or writing only when an impulse is received from the computer, these heads read and write continuously. As a word comes under the read head, it is rewritten by the write head.

Copyright 1954
by ElectroData Corporation

It may be of some help to think of the loops in terms of the following analogy:

Imagine that the space between sectors in a band are small pockets, each of which will hold one steel ball. Two hundred pockets would then be formed. Suppose further, a mechanism is designed which will pick up a steel ball projectile and throw it to a receptacle twenty pockets away. (See facing page 8.) The 200 pockets around the circumference of the drum form an endless conveyer belt for steel balls, traveling between the projector and the receptacle. Once 20 balls are in the system, they will travel on the drum from the receptacle to the projector and back again in an endless cycle. If it is necessary to replace one of the balls with a new one, from outside the system, the projector is not actuated when that ball is ready for projection and the new ball is dropped in the receptacle from outside to take its place. Replacement of all 20 balls may be accomplished by inhibiting the projection for 20 balls and, at the same time transferring, in proper time sequence, 20 different balls into the receptacle. If the main memory of the drum is imagined constructed in the same manner -- that is, of 20 bands of 200 pockets each, the reader can carry this analogy through to include the block transfer of 20 balls from and to main memory and the transfer of one ball (word) from the A register to the loops.

The foregoing analog implies that entire words (one steel ball) are transferred from the read head back to the write head of a quick-access loop. This, of course, is an oversimplification. The transfer is actually piecemeal, digit by digit, with twelve transfers per word (space, sign, and 10 digits).

The implication that when blocks of 20 words are transferred from main memory to the loops the words are lost in main memory, as would be physically the case

Copyright 1954
by ElectroData Corporation

with the balls, is erroneous. A block transfer of 20 words from the main memory to the loops leaves the 20 words in main memory unchanged. A block transfer of 20 words from a loop to main memory does not in any way alter the information in the loops. Reading information from memory does not alter the information. Information is lost only by writing new information over it.

The same sector numbers are used for addressing individual words in main memory and in the quick-access loops. One drum revolution after a word is written in a loop, it is stored not only in the location where it was originally recorded, but, also, assuming that no block transfer to the loop or single-word transfer from the A register intervenes, in nine other locations whose addresses are congruent modulo 20 to that of the original cell. More important, the word is available to the read head every 20 word-times. A word written at sector 18 will be written, after one "circulation" of the loop, at sector 38, and again at 58, 78, 98, 118, 138, 158, 178, and 198. After one revolution it is again written at sector 18.

To identify a word within a loop, regardless of its original position on the drum, sector addresses are interpreted modulo 20, that is, sectors 18, 38, 58, 78, 98, 118, 138, 158, 178, and 198, all congruent modulo 20, are used to identify the eighteenth word in the loop no matter which of the above sectors the word will be read from during a particular drum revolution. Likewise, all addresses that refer to the loop are reduced modulo 20. The first digit of a four-digit loop address identifies the loop. Thus, any address of the form 7xxx refers to loop 7 (L_7); 6xxx to loop 6(L_6); 5xxx to loop 5(L_5); 4xxx to loop 4(L_4). The last three digits of any loop address are interpreted modulo 20 to identify one of the 20 words

Copyright 1954
by ElectroData Corporation

within a loop. For example: the address 4378 is interpreted as 4018; 7642 as 7002, 6473 as 6013, 5099 as 5019, etc. There are thus 1000 addresses, 7000-7999, that may be used to address the 20 words in L_7; 6000-6999 for L_6; 5000-5999 for L_5; 4000-4999 for L_4.

On the transfer of 20 words from main memory to one of the loops, each individual word of the twenty is transferred from its sector position to the same sector position in the loop. For example, a word stored at 3016, (sector 16) will be transferred to 4016 on a block transfer from main memory to the 4000 loop. The transfer will not take place until sector 016 is under the read head of main memory. For this reason the loop address of each word written in a loop via a block transfer will correspond modulo 20 to the main memory address from which it is read. The following table illustrates the address, sector, and modulo 20 correspondence of words in a 20-word block transfer from main memory positions 2168-2187 to L(5) and a 20-word block transfer from L(6) to main memory positions 3142-3161.

Command: bl(5)<---2168 Command: bl(6)--->3142

Word in main memory location	Transferred to loop location	Sector at which reading-writing operation takes place	Address Mod 20 correspondence	Word in loop position	Transferred to main memory position	Sector at which reading-writing operation takes place	Mod 20 address correspondence
2168	5008	168	8	6002	3142	142	2
2169	5009	169	9	6003	3143	143	3
2170	5010	170	10	6004	3149	144	4
2171	5011	171	11	6005	3145	145	5
2172	5012	172	12	6006	3146	146	6
2173	5013	173	13	6007	3147	147	7
2174	5014	174	14	6008	3148	148	8
2175	5015	175	15	6009	3149	149	9
2176	5016	176	16	6010	3150	150	10
2177	5017	177	17	6011	3151	151	11
2178	5018	178	18	6012	3152	152	12
2179	5019	179	19	6013	3153	153	13
2180	5000	180	0	6014	3154	154	14
2181	5001	181	1	6015	3155	155	15
2182	5002	182	2	6016	3156	156	16
2183	5003	183	3	6017	3157	157	17
2184	5004	184	4	6018	3158	158	18
2185	5005	185	5	6019	3159	159	19
2186	5006	186	6	6000	3160	160	0
2187	5007	187	7	6001	3161	161	1

Copyright 1954
by ElectroData Corporation

Quick-access memory cells, as well as main memory, may be referred to by the addresses of input commands, so that information may be read directly into the loops without the necessity of block transfer from main memory.

The operation of the control counter in connection with the modulo-20 interpretation of loop addresses should be noted. The control counter may advance by 1 for each command executed from loops, even though the commands are repeated. For example, because of modular correspondence, cell 5020 is the same as 5000, and if no command is used which resets the control counter, the same set of 20 commands will be executed three times as the control counter advances from 5000 to 5059 (which calls for the command in 5019).

This feature of control counter behavior gives three opportunities to use 40 consecutive quick-access commands without providing a control change: from 4980 through 5019, from 5980 through 6019, and from 6980 through 7019. Except when the control counter provides an automatic change from the last twenty addresses in one thousand-series to the first twenty of the next thousand, the only way to change control from one loop to another is to provide a command for that purpose.

When programmed properly, the quick-access loops obviate the need for restoring to its original form, information which may have been altered during computation. This is illustrated by the program below.

```
         4 in                 C Register           A Register

2309     bl(4)<-2309          34 2309 2310
2310     cu      4311         20 4311 4311
2311     ca      4316         64 4316 4312         +.9324937021
2312     mr      4317         70 4317 4313         -.6529690681
2313     tmh     4317         12 4317 4314
2314     z       4311         04 4311 4315
2315     STOP                 08 0000 4316
2316     +.9324937021
2317     -.7002396548
       6 cu      2309
```

As this program is executed, the 20 words in 2309 through 2328 are block transferred to 4309 through 4328 (interpreted as 4009 through 4008). Since the B register is not used, the address parts of the commands following the first one must be loop addresses. After the execution of the tmh in 4313 (originally 2313), cell 4317 contains -.6529690681. Cell 2317, however, still contains the original value -.7002396548. This program, then, including the initial block transfer, may be repeated without going through the steps of prestoring initial values.

Branching Commands

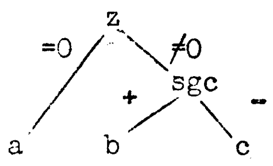
In the program above, the 04 originally stored at 2314 is a branching command. The others are B- y, spl y, and cc and its variants. (The command sgc is not a branching command, but, like arithmetic overflow, provides the condition for the branching of cc.) Every time the content of the A register is not zero, the 04 command is converted automatically within the computer to a control change; the 4311 address is shifted into the control counter, displacing 4315, and the program returns to 4311. On each repetition the magnitude of the product from the mr in 4312 becomes smaller; when it is small enough to fall within the R register, leaving the A register zero, the 04 will be unchanged and the control counter will continue to read 4315. All branching commands are skipped if the condition for change of control is not met, and if the condition is met, all of them accomplish the change by converting the contents of the order register (C1 and C2) to a cc command (28) and momentarily setting the overflow toggle. A branching command allows the computer to choose which of two paths it should take in order to find its next command, depending on conditions arising from the execution of the program.

Copyright 1954
by ElectroData Corporation

Multiple Branches

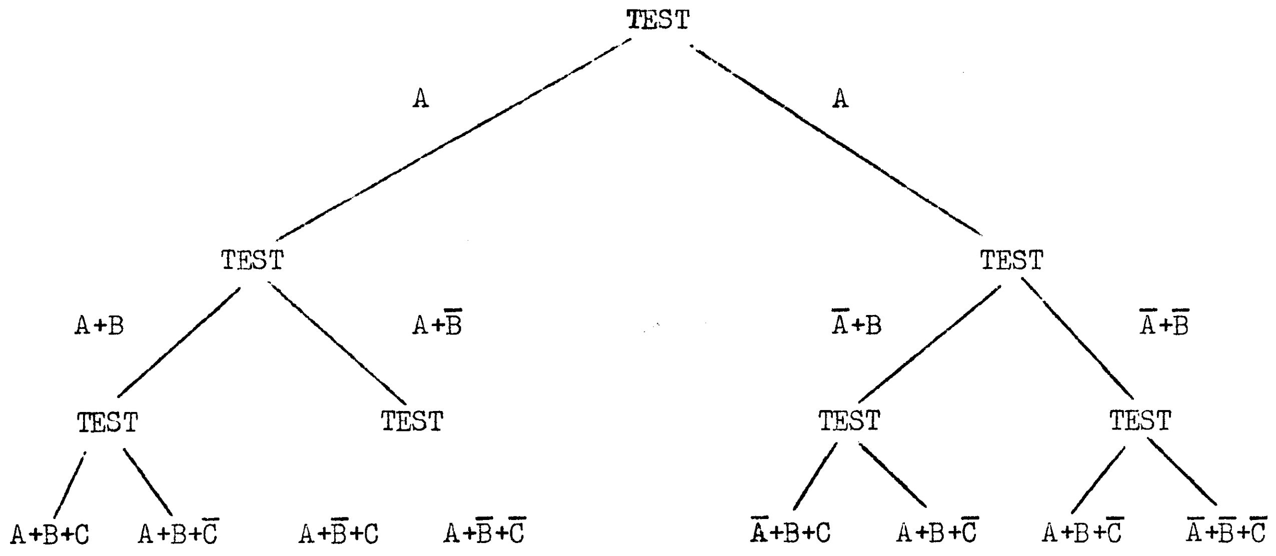
So far, only two choices on a test have been considered. It is possible, by using a series of branching commands, to have an unlimited number of choices. For example, the following series of commands branches to b, c, or a, depending on whether the number in the A register is positive, negative, or zero. The diagram at the right is a symbolic outline of the program.

```
6002    z      6004
6003    cu     (a)
6004    sgc    6002
6005    cc     (c)
6006    cu     (b)
```



With n levels of branching it is possible to work out a program which contains 2^n possible choices as illustrated below:

A bar over the letter indicates that the condition is <u>not</u> true. The plus sign indicates "and."



In this diagram there are $2^3 = 8$ possibilities. This process could be continued indefinitely, but need not be symmetrical.

Copyright 1954
by ElectroData Corporation

Use of the B Register

The most important uses of the B register are address modification, and, since the B register can be counted down, tallying a cycle while the addresses of commands within the cycle are altered by the addition of a different number each time. Modification of a command word by addition of the B register, including decimal carries, takes place after the word is read from the drum, as it is being transferred from the D register through the adder to the C register. The word remains unchanged in the quick-access or main memory cell from which it was read. The signal for address modification by the B register is a 1 in the sign position of the command word stored on the drum. Several examples of B-register modification are given below:

Word on Drum	Content of B Register	Word in C Register
0 0000 64 2349	0794	64 2349 ----
1 0000 64 2349	0794	64 3143 ----
1 0000 64 2349	9043	65 1392 ----
1 0000 64 2349	0000	64 2349 ----

Note that the addition of the contents of the B register in the third example above changes the instruction from "clear and add" to "clear and subtract."

The following code exemplifies the use of the B register as a command modifier and tally.

Problem: Count all words with negative sign stored in 3016 through 3999; store the resulting count in 2001.

```
Storage
Address         Command              Effect                C Register
               4 in 0060
0060              B 7069            Load B                 72 7069 7061
0061           - ca 3016                                   64 3999 7062
0062            sgc 7061          Test for negative        73 7061 7063
0063             cc 7067     Omit count if positive        28 7067 7067
```

Copyright 1954
by ElectroData Corporation

Storage Address - cont.	Command	Effect	C Register
0064	ca 7070		64 7070 7065
0065	ad 7071	Add 1 to count if negative	74 7071 7066
0066	tmh 7070		12 7070 7067
0067	B- 7061	Tally and recycle	22 7061 7061
0068	tmc 2001	Store count	02 2001 7069
0069	STOP 0983		08 0983 7070
0070	[0 0000 00 0000]	Loop count storage	
0071	0 0000 00 0001		
	6 cub 0060		

Note that B is loaded with 0983, the address part of the command in 7069. (This command was put into 7069 from 0069 by the cub 0060.) The C-register content shown for the command in 7063 (28 7067 7067) is what would be there if the number in 3999 were positive; that is, the overflow toggle would be set, there would be no addition to the count, and control would be changed to 7067 (the address tally for the 3016-3999 series). If the number in 3999 were negative, the sign comparison would show no difference, the cc (28) would be changed to a cu (20) and control would be changed to 7064, which would show in the control counter. This and the two succeeding commands would store a count of 1 for a negative number in 3999.

The B- 7061 counts B down to 0982 and changes control to 7061. On this execution the command in 7061 brings its operand from 3998, since the content of the B register is changed. When the B register is counted down to 0, the preceding operand will have been brought from 3016 and control will be changed to 7068, which will store the count in 2001. The computer stops on the next command.

B-register address modification used with the B- command is the most frequently used tallying device. To go through n cycles of a program, set B to n - 1 and let the address part of the B- be the address of the first command in the cycle.

Copyright 1954
by ElectroData Corporation

Problem: Add +.0000000500 to the numbers stored in 0400 through 0599. Store the new numbers consecutively in 0600 through 0799, so that (0600) = (0400) +.0000000500, (0601) = (0401) +.0000000500, etc. Assume that there will be no overflow.

```
          4  in        1098
  1098       bl(6) ←—  1100)     Block the program into the 6000 loop and
  1099       cu        6100)     change control to the loop.
  1100       B         6115      Set B at 19.
  1101       bl(4) ←— [0400]     Block 20 numbers into L4.
  1102      -ca        4400)
  1103       ad        6116)     Compute and temporarily store the 20 new
  1104      -tmh       5000)     numbers.
  1105       B¯        6102)
  1106       bl(5) —→[0600]      Block the new numbers to storage.
  1107       ca        6101)
  1108       ad        6117)     Alter the blocking command so that the next
  1109       cc        6115)     20 numbers will be brought into the loop.
  1110       tmc       6101)     Simultaneously tally: if there is overflow,
                                 branch. The problem is finished.
  1111       ca        6106)
  1112       ad        6117)     Alter the blocking command which stores the
  1113       tmc       6106)     new numbers.
  1114       cu        6100      Change control to the beginning for the
                                 computation of 20 new numbers.
  1115       STOP      0019      Stop the computer.
  1116        +.0000000500
  1117       +.01000 00020
          6  cu        1098
```

In this example the command in cell 1101, bl(4) 0400, is written as +.9000 34 0400. Therefore after the command has been altered ten times there will be an overflow, indicating that the final block of twenty numbers has been computed.

This example illustrates a frequent use of the loops, that is, as a storage place for numbers when they are being used in computation. In spite of the extra blocking commands which are needed, there is a considerable saving of time in being able to execute the entire program from the high-speed loops.

After the program has been blocked into the 6000 loop, all the loop addresses differ from the corresponding main-memory addresses by only the first digit:

Copyright 1954
by ElectroData Corporation

1115 becomes 6115 rather than 6015. Although this is not necessary, in view of the modulo 20 correspondence of loop addresses, it simplifies the identification of the commands being executed.

Cycles Within Cycles

A long cycling program **may** contain one or more small cycles within a larger cycle.

Problem: Find the sum of the individual digits of the numbers in 0100 through 0199 and store the sums in corresponding locations from 0200 to 0299.

Storing the B register before each small cycle will make it possible to use the B register again in the small cycle.

```
                       4 in 0000

            0000     B      7015     Load B initially.
  ┌─────────
  │         0001    -ca     0100     (A)a = first argument.
  │         0002     sr     0010
  │ LARGE   0003     B->A   0009)
  │ CYCLE   0004     tmc    7016)    Store B.
  │         0005     tmc    7017     (7017)a = zero.
  │         0006     B      7003     Load B for small cycle.
  │       ┌ 0007     sl     0001)
  │ SMALL │ 0008     ad     7017)    Sum and store individual digits.
  │ CYCLE │ 0009     tmc    7017)
  │       └ 0010     B-     7007
  │         0011     B      7016     Load B for large cycle.
  │         0012     ca     7017
  │         0013    -tmc    0200     Store sum.
  └──────── 0014     B-     7001
            0015     STOP   0099
            0016                     B storage
            0017                     Sum storage
                       6  cub 0000
```

Step 0005, starting from an A register cleared by the previous order, is merely a convenient method for prestoring a zero.

Copyright 1954
by ElectroData Corporation

It is sometimes not desirable to make the address parts of command words specific. This circumstance may arise in the coding of subroutines or when several sections of a large problem are coded separately and it is not known in advance just where in memory a given section will be placed. The commands may be coded with relative addresses: that is, the addresses within the words are assigned on the assumption that this section of coding will be stored in consecutive cells beginning with 0000 or with any appropriately chosen cell.

Such relative codes may be stored anywhere on the drum, and on input the B register may be used to adjust the address parts of words to accord with the locations at which the commands in the codes are stored. This is done by means of a 2 or a 3 in the sign column. The 2 adds the B register to the command and is cleared before the word is put away, so that the word has 0 in the sign column. The 3 adds the B register and is changed to a 1 as the word is put away.

Since the address parts of commands in subroutines are usually relative to the location of words within the subroutine, it is not necessary to modify such words by the B register on input unless the address part is to refer to main memory during the execution of the subroutine.

The following is an example of coding with relative addresses.

Problem: Print out a block of 20 numbers starting at any memory location. Assume that it is desirable to be able to place the program for this printout in any convenient memory location. (A program of this type is sometimes useful in checking a routine.)

(1) Set B manually at 1120, assumed to be the first of 13 vacant cells where the program will be placed.

(2) Set R1, R2, R3, R4 manually at 2250, assumed to be the address of the first of the 20 words which are to be printed out.

Copyright 1954
by ElectroData Corporation

(3) Read in the following program punched on tape.

```
     Program on Tape                       Program as it appears in Memory

       5 in      0000

0000   2 ca      0011              1120   ca     1131
0001     sl      0004              1121   sl     0004   Forms a 1 ca 2250
                                                        command in cell 1124.
0002   2 tmc     0004              1122   tmc    1124
0003   2 B       0007              1123   B      1127   Sets B initially at 0.
0004   - ca      0000              1124   - ca   0000
0005     po      0010              1125     po   0010   Prints out the 20 words.
0006     B+      0000              1126     B+   0000
0007     B—>A    0000              1127     B→A  0000
0008   2 su      0012              1128     su   1132
0009   2 z       0004              1129     z    1124
0010     STOP    0000              1130    STOP  0000   Stops the computer.
0011   1 0000 00 0064              1131   1 0000 00 0064
0012     0000 00 0020              1132     0000 00 0020

       7 cu      0000
```

The 5 in the sign column will start the tape feeding in and the "in" address will have the contents of the B register added to it. The 7 in the sign column will stop the tape and will start the execution at the memory location of the first word in the program.

A relatively coded program can be placed in any memory location automatically, without a manual setting of the B register before read-in.

For illustration, assume that, as in the example above, the program is to be placed in consecutive memory locations beginning with cell 1120.

Place the following commands ahead of the relatively coded program.

```
     4 in     1120       Starts reading the tape into cells beginning with
                         location 1120.

1120   B      1121       Sets B to the content  of cell 1121, which is 1120.

1121   in     1120       Reads the relatively coded program into cells
                         beginning with 1120.

     4 cu     1120       Executes the B setting before reading in the program.
```

Copyright 1954
by ElectroData Corporation

Relatively coded program follows after six inches of blank tape, which allows time to execute the two commands.

The program blocks over the commands which were used to set B for the read-in operation, so that it is not necessary to use two additional memory locations. The program appears in memory exactly as it did in the preceding example.

It is important to note that a relatively coded program which is to be blocked into a loop must be placed in memory in consecutive locations beginning with one which corresponds to 0 modulo 20. If this were not the case, the references within the loop would not correspond to the correct addresses. Another example of relative coding is given in Section 6.

Methods of Checking Programs

After a program has been coded and punched on tape or cards, it must be given a trial run on the computer. During this trial run, the programmer checks his routine for any errors in coding, scaling, or punching.

Points in the routine at which the operator can compare computer results with precalculated or predetermined data are called check points. If the coding is in error, the operator would like to retrace the computational path up to the point of error from a portion of the code that he knows to be correct.

BREAKPOINT Switch

Check points can be easily identified by a system of breakpoints, or predetermined automatic stops. The positions of the BREAKPOINT switch are OFF, 4, 2, and 1. The 4, 2, and 1 positions refer to binary components of the decimal digit used for the breakpoint code (in the column to the left of the two order digits). The breakpoint digit will cause a stop if the switch is set for a binary component which it contains. The computer stops after executing the

Copyright 1954
by ElectroData Corporation

command containing the breakpoint code. Shown below are the binary representations of 1, 3, and 7 as they would appear in the breakpoint digit:

```
o  o  o          o = off
o  o  o          c = on
o  ●  ●
●  ●  ●
1  3  7
```

With the switch at 4, a breakpoint 7 will cause a stop but 3 or 1 will not.
With the switch at 2, a breakpoint 7 or 3 will cause a stop but 1 will not.
With the switch at 1, a breakpoint 7, 3, or 1 will cause a stop.

Switch Setting	Corresponding Breakpoint Digits for stop
4	4, 5, 6, 7
2	2, 3, 6, 7
1	1, 3, 5, 7, 9

The stratification of stops made available by the BREAKPOINT switch makes it possible for an operator to run a properly coded routine first on a 4 setting of the switch to determine in which major portion of the code an error lies, and then, by changing the switch setting, to narrow the range of points bounding the error. In the following table, the commands of the routine are omitted, leaving only the arrangement of breakpoint digits. Between any two breakpoint digits there might be only a few commands or a long section of coding.

```
7                  1
3                  1
1                  1
7                  3 ←second indication of error
3                  7←first indication of error
1                  3
3                  7
←exact location of error
```

On the first time through the program, with a 4 switch setting, the computer stops only on the 7's; the operator discovers an error at the third 7 breakpoint. The error lies between it and the second 7 breakpoint. Setting the switch at 2

Copyright 1954
by ElectroData Corporation

and sending the program back to the second 7 breakpoint, the operator narrows the field to the section of coding between the third and fourth 3 breakpoints. Sending the program back to the third 3 breakpoint, the operator finds that the error is between the third 3 breakpoint and the third 1 breakpoint. Sending the program once more back to the third 3 breakpoint and proceeding by STEP operation, the operator determines the exact location of the error.

Correcting Errors

Once an error has been found, the operator may correct it in the memory and continue, or he may correct it on tape or cards and then reload the program. If correcting the error involves only a change in an order or an address, a trailer tape (or set of cards) is added to the input, on which a few commands are used to record the correct information over what is already on the drum.

If correcting the error involves inserting missing instructions, the coder may store in the cell just preceding the one where the omitted command(s) should be, a cu which will change control to a group of unused cells. Into these cells he puts the command replaced by the cu, the omitted command(s), and a cu back to the program. Renumbering the commands after inserting missing instructions involves a good deal of alteration of addresses and is usually not done unless the error is discovered before the problem is run on the computer.

The following program, for example, will probably result in error since the "divide" command has not been preceded by a cl R. It could be a small part of a much larger program.

```
                                Correction 1              Correction 2

4340    mh 5000                 4340     mh 5000          4340       mh 5000
4341   tmc 5005                 4341    tmc 5005          4341      tmc 5005
4342  - ca 1000                 4342  - ca 1000           4342     - ca 1000
```

Copyright 1954
by ElectroData Corporation

```
4343     su 4348
4344    div 4349
4345 -  tmc 2000
4346     B- 4342
4347    cub 0360
4348    operand
4349    operand
```

Correction 1 - cont.

```
4343      cu 4350
4344     div 4349
4345  -  tmc 2000
4346      B- 4342
4347     cub 0360
4348     operand
4349     operand
4350      su 4349
4351    cl R
4352      cu 4344
```

Correction 2

```
4343      su 4349
4344    cl R
4345     div 4350
4345  -  tmc 2000
4347      B- 4342
4348     cub 0360
4349     operand
4350     operand
```

SKIP Switch

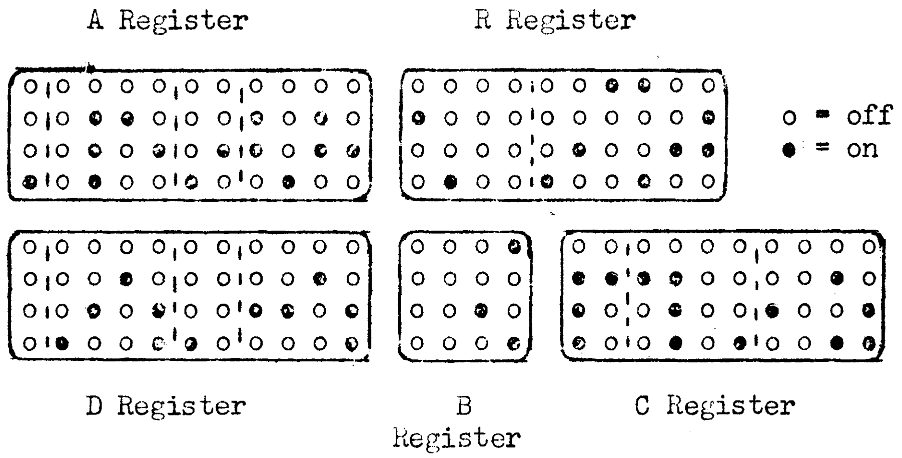
If the SKIP switch is turned on, any command with an 8 in the column to the left of the two order digits will be skipped. This is a useful device to incorporate in a program which might have several alternate paths. For example, a section of coding which treats data in one way may be preceded by a cu which, if executed would change control to another section of coding which would treat the data in a different way. With an 8 in the fourth digit position of the cu command, the position of the SKIP switch determines which of the two paths is taken.

The SKIP switch can be useful also in checking a routine. During the checking procedure, it may be desirable to print out information somewhere in the middle of the routine. With an 8 in the fourth digit of the printout commands and the SKIP switch turned on, the printout will be eliminated on the final run of the problem.

CONTROL PANELS, KEYBOARD, AND PAPER TAPE EQUIPMENT

Register Displays

The contents of all registers are shown in arrays of neon bulbs on the control console and on the supervisory control panel (on the front of the computer). Each vertical column of neon bulbs is a decimal digit whose value is the sum of the values of the lighted bulbs. Reading from top to bottom the bulbs represent 8, 4, 2, and 1. No value higher than 9 normally appears in any register. The register display arrangement on the console is shown below:



Only the A and D registers have algebraic signs (0 = +, 1 = -). Other digits in all registers are designated by the letter of the register and the number of the digit position from left to right, not including sign--for example, A5, R10, B4, C6. The signs are labeled ASG and DSG. The three parts of the C register separated by dotted lines are the order register (C1, C2), the address register (C3, C4, C5, C6), and the control counter (C7, C8, C9, C10). The register contents shown are:

```
-.0742126262 2100128926
+.1243102243 0029 7447012053
```

Copyright 1954
by ElectroData Corporation

During STEP operation this oonfiguration would show that the number in D, brought from cell 4701, will be added (74) to the number in the A register.

FETCH Light and EXECUTE Light

The timing toggle determines the basic timing cycle of the computer, which alternates between the fetching of a command (from the drum to the C register) and the execution of the command fetched. The two neon bulbs labeled FETCH and EXECUTE on the console panel show the state of the timing toggle. Since the toggle is triggered to the opposite state at the beginning, not at the end, of each phase of the timing cycle, the lighted bulb shows during STEP operation which of the two phases will be performed next. This means that the FETCH light is on while the "execute" phase proceeds and the EXECUTE light is on while the "fetch" phase proceeds.

CLEAR Button

Pressing the CLEAR button sets every digit of every register (including the sign digits of the A and D registers) to zero. It should be noted that this puts the command "in 0000" (00 0000 0000) into the C register. The CLEAR button will not affect any information on the drum.

INPUT Selector

The INPUT selector has three positions, MECHANICAL READER, OPTICAL READER, and KEYBOARD. The mechanical reader, attached to the Flexowriter, reads into the computer from paper tape (subject to the wiring of the patch panel on the back of the typewriter control) at about 12 digits per second. The optical reader, a photoelectric device in the console, reads into the computer from paper tape at about 540 characters per second.

Copyright 1954
by ElectroData Corporation

The keyboard contains a kay for each decimal digit and a "finish" key. Pressing the finish key completes the entry of a word into the computer from the keyboard. The keyboard is generally used to read in a few words at a time, as for inserting corrections or inserting or deleting breakpoints.

OPERATION Buttons

Pressing the CONTINUOUS button throws the timing toggle to EXECUTE and starts continuous operation of the computer. If the registers have been cleared, the first command executed will be the "in 0000" order in the C register, and the machine will start to read in from the device selected by the INPUT switch.

Pressing the STOP button stops the computer.

Pressing the STEP button with the machine stopped puts the program through the computer one step at a time. The button is pressed once for the "fetch" and once for the "execute" phase of each command. Pressing the STEP button while the computer is in continuous operation will produce confusion and will garble information in the arithmetic registers; the STOP button should be pressed before step operation is begun.

OPERATION Lights

Associated with the OPERATION buttons are two OPERATION lights. The CONTINUOUS light indicates continuous operation. The NOT READY light indicates that one or more of the switches on the computer switch panel are not in NORMAL position.

BREAKPOINT Switch

The use of the BREAKPOINT switch is described in Section 5. When the switch is not in the OFF position, the computer will stop after executing a command

Copyright 1954
by ElectroData Corporation

whose fourth digit contains a binary 1 in the row corresponding to the switch setting--4, 2, or 1. If the BREAKPOINT switch is in the OFF position, the computer ignores breakpoint codes. Pressing the CONTINUOUS button will start the computer after a breakpoint stop.

Switch Setting	Breakpoint Digits Which Stop Computer
4	4, 5, 6, 7
2	2, 3, 6, 7
1	1, 3, 5, 7, 9
OFF	None

SKIP Switch

The use of the SKIP switch is described in Section 5. When the SKIP switch is in the ON position, the computer will skip any command having an 8 or 9 in the fourth-digit position (which is also the breakpoint position). It should be noted that if there is a 9 in the fourth-digit position, the computer will stop at that command, but will not execute it.

When the SKIP switch is in the OFF position the computer will ignore skip codes.

COMPUTER STOP Lights

The COMPUTER STOP lights are labeled IDLE, FC/SA, CONTROL, and BKPT (BREAKPOINT).

The IDLE light is on when the computer is in step operation and will go on if any malfunction should interrupt the fetch/execute cycle.

The CONTROL light is on when the computer is stopped by the STOP button or by a programmed stop (08).

The FC/SA light will go on for a "forbidden combination" (register digit greater than 9 in value) appearing in any one of a number of strategic locations,

Copyright 1954
by ElectroData Corporation

or for a "sector alarm"--that is, if the sector counter should fail to contain 000 at the instant of the origin pulse on the drum. (Such failure would mean that reading and writing operations were out of synchronism, which would result in garbled information.)

The BREAKPOINT light is on whenever the computer has made a breakpoint stop.

AUDIBLE ALARM Switch

When the AUDIBLE ALARM switch is on, a buzzer will signal if any of the COMPUTER STOP lights or the OVERFLOW light goes on. The buzzer is disabled if the switch is off.

OVERFLOW Light

The computer stops and the OVERFLOW light goes on if an addition, a subtraction, or a division results in arithmetic overflow (result equal to or greater than 1.0000000000 in absolute value) and is not followed by a conditional change of control. The fractional sum or difference remains in the A register after an overflow addition or subtraction, but the A and R registers are cleared after an overflow division and the dividend is lost.

The computer stops and the OVERFLOW light goes on if a sign comparison discovers different signs and is not followed by a conditional change of control.

As has been noted in Section 5, the overflow toggle is set and the OVERFLOW light goes on momentarily in branching commands, but the cc is built into those commands and the computer does not stop.

The overflow toggle can be set to zero and the OVERFLOW light turned off by pressing the RESET button located below the OVERFLOW light.

Copyright 1954
by ElectroData Corporation

OUTPUT Selector

The OUTPUT selector may be left in the OFF position if no printout commands are used. If a printout command comes up for execution while the selector is in the OFF position the computer will stop and the IDLE alarm light will go on.

If the OUTPUT selector is in PAGE position, output will go to the typewriter, subject to the patch panel connections on the back of the typewriter control. It should be noted that failure to turn on power at the typewriter when the selector is in PAGE position can result in loss of information. In such a case, the A-register contents will be shifted as usual and the computer will continue as if it had printed out.

If the OUTPUT selector is in TAPE position, printout commands will actuate the motorized paper tape punch on the control console.

If the computer idles because the selector is in OFF position at the time of a printout command, the STOP button should be pressed; otherwise, moving the selector may introduce transient pulses into the computer. With the computer stopped, the operator may set the OUTPUT selector, change control to a location from which the printout command can be repeated, and then press the CONTINUOUS button.

P.O. SUPPRESS (Printout Suppress) Switch

The PRINTOUT SUPPRESS switch, when on, will cause the computer to ignore printout commands and the machine will operate normally otherwise. If the switch is off, printout commands will be executed.

Supervisory Control Panel

The supervisory control panel on the front of the computer contains all switches, lights, and buttons appearing on the console panel and additional

Copyright 1954
by ElectroData Corporation

controls and indicators which are helpful in computer maintenance.

Banks of push buttons corresponding to the banks of neon-bulb indicators for the register toggles make it possible to set each register toggle to 1 or 0. Pressing one of the buttons will set the corresponding toggle to 1 and light the neon bulb. Pressing one of the buttons and the ZERO button at the same time (the toggle push button should be released first) will set the corresponding toggle to 0 and turn off the neon bulb.

A CLEAR button associated with each register will set all toggles in the register to 0 and turn off all neon bulbs.

Computer Switch Panel

The computer switch panel, below the supervisory control panel, is used primarily for maintenance. If any of its switches (except PRINTOUT SUPPRESS and SKIP, which can be overridden at the console panel) is off the NORMAL position, the NOT READY light goes on. The computer panel switches that an operator may have occasion to use are described in the following paragraphs.

AUTOMATIC ROLLBACK Switch In NORMAL position this switch has no effect. When the switch is off NORMAL, the computer will call for input, after any stop, from the device indicated by the INPUT selector.

TAPE SUM Switch In NORMAL position this switch has no effect. When the switch is off NORMAL, all information coming from the keyboard, the mechanical tape reader, or the photoelectric tape reader, both numbers and commands, is summed in the A register as decimal numbers (absolute value). The information does not go to the drum or to the C register. The fractional part of the sum appears in the A register when the input stops.

Copyright 1954
by ElectroData Corporation

The tape sum operation is started by setting the INPUT selector, clearing the registers, and, with the TAPE SUM switch off NORMAL, pressing the STEP button. Tape commands (words with 4, 5, 6, or 7 in the sign column) appearing in the input will not be included in the summation: words with 4 or 5 in the sign column will be ignored, and words with 6 or 7 in the sign column will stop the input operation, making it necessary to press the STEP button again to continue the summation.

IGNORE FC & SECTOR ALARM Switch When this switch is off the NORMAL position, the computer will continue operating sequentially despite overflow; cc, ccb, ccbR, and ccR commands will be treated as if no overflow had occurred. When the switch is in the NORMAL position, arithmetic overflow will cause the machine to stop unless a cc, ccb, ccbR, or ccR command follows the command that led to the overflow.

FILL MEMORY Switch, CIRCULATE D Switch, and RECYCLE Switch When these switches are off the NORMAL position, any number of memory locations can be filled with a given quantity. The quantity is placed in the D register from the supervisory control panel, the control counter set to the address of the first location to be filled, input is set to KEYBOARD, and the computer put into continuous operation. The quantity in the D register will be read into consecutive memory locations, the control counter counting up 1 each time. The operation may be stopped by pressing the STOP button.

DECIMAL CORRECT SUPPRESS Switch When this switch and the IGNORE FC & SECTOR ALARM switch are off the NORMAL position, correction of the binary sum appearing in the adder is suppressed, and the addition of two positive or two negative quantities

Copyright 1954
by ElectroData Corporation

will treat the digits as hexadecimal digits with hexadecimal carries, the result appearing in the A register as hexadecimal digits. If after the adjustment of D-sign for subtraction and for absolute-value commands the signs of A and D are different, the A register will be complemented on 10 as in normal operation and the resulting digits will be added hexadecimally to the contents of the D register.

Control Settings for Normal Operation

Most codes can be read into the computer with the following control settings:

OUTPUT Selector	PAGE
P.O. SUPPRESS Switch	OFF
SKIP Switch	OFF
BREAKPOINT Switch	As desired
INPUT Selector	OPTICAL READER
Computer Switch Panel	All switches at NORMAL
Patch Panel (On Typewriter Control)	FROM CONSOLE TO FORMAT/FROM FORMAT TO FLEXOWRITER
Flexowriter	ON
Format Control Panel (On Typewriter Control)	Press RESET; other settings as desired

Pressing the CLEAR button and then the CONTINUOUS button will start input and put the computer in continuous operation.

Starting and Shutting Down the Computer System

Various power controls of primary interest to the engineer must be in normal condition before the system can be operated from the power control panel. These controls and their locations and conditions are:

Three-phase disconnect switch and filament power disconnect switch (near motor-generator)	Closed
Circuit-breakers (near power control)	On
DC LOCKOUT switches (one on power control panel, one on computer switch panel, and one on marginal voltage check panel at rear center of computer)	Off
Overload breakers marked M. G. OUTPUT CONTROL -160, +200, +250 and MASTER CONTROL 115 V.A.C. (inside door at left of power control cabinet)	ON
Filament variable transformer (handwheel inside door at left of power control cabinet)	Turned clockwise to the stop

On the power control panel are ON buttons (black) and OFF buttons (red) for the blowers, tube filaments, drum motor, motor-generator, and d-c voltages, arranged from left to right. At the top of the panel eight computer voltages are indicated, each with a voltmeter (upper) and an ammeter (lower) above it. The voltmeters must be checked when the motor-generator is turned on. Any variance of a meter reading from the correct voltage indicated on the panel should be brought to the attention of the maintenance engineer.

Starting the System

Press the BLOWERS ON button.
Press the FILAMENT ON button.
Turn the filament transformer handwheel slowly counter clockwise until the meter reads 230 volts. This should take at least 30 seconds.

Copyright 1954
by ElectroData Corporation

Starting the System - continued

Press the FILAMENT FAIL RESET button.
Press the DRUM ON button.
Wait five minutes for tube temperatures to stabilize.
Press the MOTOR GENERATOR ON button. An alarm will sound to remind the operator to check the voltmeters.
Press the DC FAIL RESET button to turn off the alarm.
Press the DC ON button.

Shutting Down the System

Press the DC OFF button.
Press the MOTOR GENERATOR OFF button.
Press the DRUM OFF button.
Turn the filament transformer handwheel slowly clockwise until it hits the stop. This should take at least 30 seconds.
Press the FILAMENT OFF button.
Press the BLOWERS OFF button.

Input from Keyboard

With the registers cleared, the C register contains an "in" command (00 0000 0000). If the INPUT selector is set at keyboard, pressing the STEP or the CONTINUOUS button will initiate the process of reading keyboard words into consecutive locations starting with 0000.

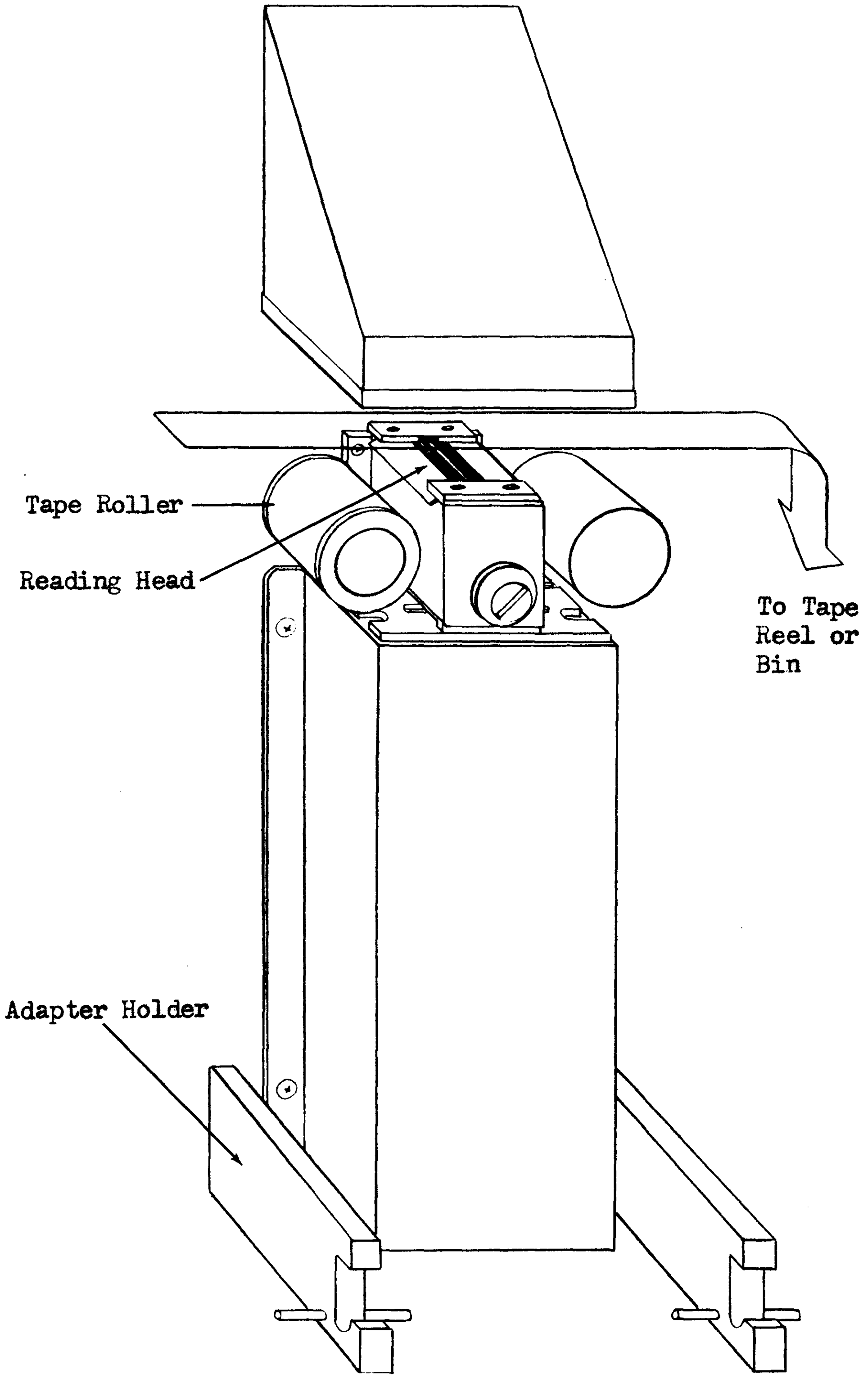
This state of affairs is altered only by some keyboard word (it may be the first) containing a 4, 5, 6, or 7 in the sign column. (Such words are usually called "tape commands," but their transfer to the C register for immediate execution takes place also if they come from the keyboard.) It may be well to recall here that 6 or 7 stops input while 4 or 5 does not, and that 5 or 7 adds the B register while 4 or 6 does not.

The operations for which keyboard input will most often be used are:

To put a word into a memory cell or a number of words into consecutive memory cells.

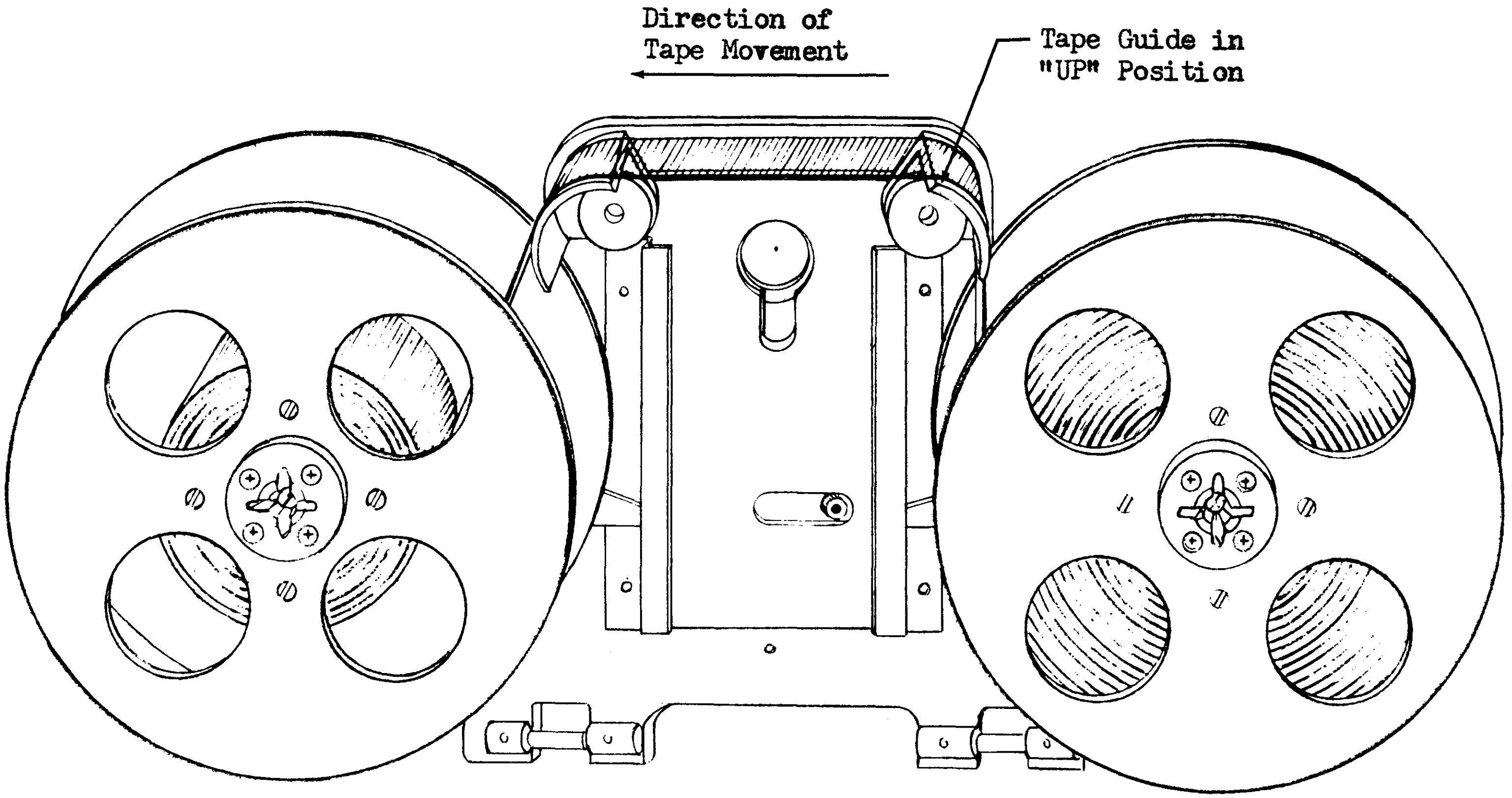
To execute a command or commands from the keyboard (usually "input," "clear and add," or "change control" command).

Copyright 1954
by ElectroData Corporation



Copyright 1954 by ElectroData Corporation

Copyright 1954 by ElectroData Corporation



Back of Reel Adapter with reels in place, ready for placing in holder.

Examples: (It is assumed that registers are cleared and INPUT is set to KEYBOARD)

To put one or more words into memory:

Press the CONTINUOUS button.
Type on the keyboard n 0000 00 xxxx f, where n is 4 or 5, xxxx is the desired storage address, and F is the F key. Pressing the F key puts the desired input order into the C register, replacing the 0000 address.
Type the word which is to be put into memory. Pressing the F key puts this word into cell xxxx (or xxxx + (B) if the sign digit of the input command was 5).
Words subsequently typed on the keyboard will be stored in consecutive memory cells until a sign-column instruction diverts the input to the C register.

To execute one or more commands from keyboard:

Press the STEP button.
Type on the keyboard n 0000 xx xxxx F, where n is 6 or 7 (4 or 5 if more than one command is to be executed) and xxxx are the address digits (xxxx + (B) if the sign digit of the input command was 5 or 7). Pressing the F key puts this command word into the C register.
Press the STEP button again. This executes the command.

To start executing a routine stored in memory:

Press the CLEAR button.
Press the STEP or the CONTINUOUS button.
Type on the keyboard a "change control" command (cu) with 6 or 7 as the sign digit.

Input from Punched Paper Tape

Press the CLEAR button.

Set the INPUT selector to MECHANICAL READER or OPTICAL READER.

Press the CONTINUOUS button.

Loading Tape into the Photoelectric Reader

The following two pages show the front of the photoelectric reader chassis and one of the two assemblies designed to adapt it to different lengths of tape,

Copyright 1954
by ElectroData Corporation

in that order. Hinge-pins on both assemblies fit into brackets at the bottom of the chassis. The larger assembly, with reels, is designed for tapes containing roughly 1000 words or more, and the smaller assembly for shorter tapes, which are sometimes spliced into loops. Either assembly is conveniently handled by a knob which serves to lock it in place on the reader. The tape moves from left to right of an observer facing the reader, with the "C" punches (accompanying the decimal digits) on the side away from the machine.

To use the loop adapter:

Put the tape in position over the read head.
Put the hinge-pins in the brackets.
Turn the knob counter clockwise and push the adapter forward over the tape.
Turn the knob clockwise, which locks the adapter in place.
Give a light tug on the tape on the left side to take out the slack.

To use the reel adapter:

Put the tape in the slot on the spool of the take-up reel and give the reel one or two turns.
Turn the knob counter-clockwise and put both reels on the adapters, fitting the reel lockpins in place and leading the tape over the raised tape guides.
Put the hinge-pins in the brackets.
Push the adapter forward and turn the knob clockwise, which locks the adapter in place, lowers the tape guides, and allows the tape to rest on the read head.

Copyright 1954
by ElectroData Corporation

DESCRIPTION AND USE OF EQUIPMENT EXTERNAL TO THE COMPUTER

Provision has been made in the ElectroData computer for the attachment of a number of pieces of auxiliary equipment which may be directly controlled by the computer. This equipment is of two kinds: input-output equipment, and auxiliary storage equipment. In addition, various auxiliary equipment not directly under control of the computer is involved in preparation and processing of data and instructions to be entered in the computer. This section describes and explains the use of standard auxiliary equipment.

Systems

The input and output equipment attached to any particular computer depends on the application of the computer. ElectroData computers accept information from:

Punched paper tape

Punched cards (IBM)

Decimal Keyboard

ElectroData computers read out to:

Punched paper tape

Punched cards (IBM)

Flexowriter or IBM Tabulator

Console register display

The two principal combinations used are:

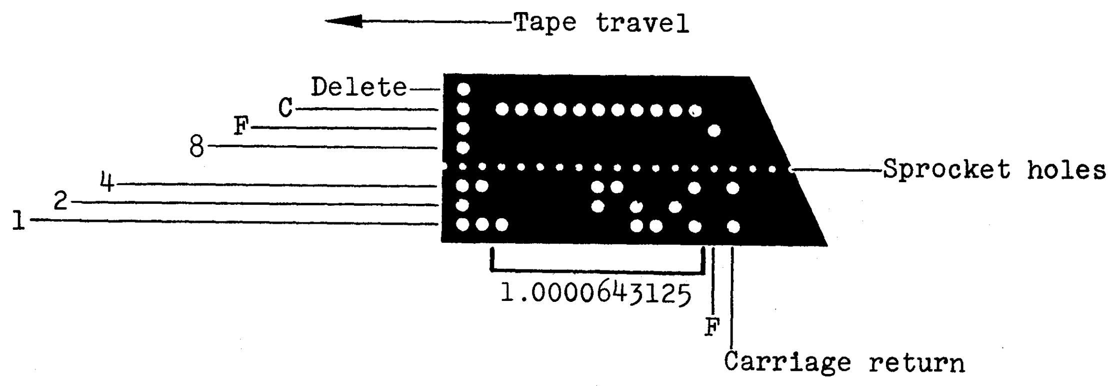
Computer, console, keyboard, photoelectric tape reader, paper tape perforator, Flexowriter tabulator.

Copyright 1954
by ElectroData Corporation

Computer, console, keyboard, converter unit, IBM units for card reading and punching, IBM tabulator.

These two systems will be referred to as the punched paper tape system and the punched card system.

Punched Paper Tape System Information to be read into the computer is punched on paper tape in serial decimal digits. Each word on the tape occupies thirteen decimal-digit positions: one for the algebraic sign; ten for digits of the word; one for the finish punch; and one for a carriage-return character, which is ignored by the computer.



Copyright 1954
by ElectroData Corporation

The illustration on the preceding page shows the upper side of the tape as it goes into the photoelectric tape reader or the Flexowriter.

The four lower channels contain the decimal digit value in binary code. The fifth channel contains the F (or "finish") punch which signals the end of an 11-digit word. The sixth channel contains the C (or "clock") punches which indicate to the computer that the content of the lower four channels at the same time is to be accepted as a binary-coded digit. The lower four channels will be read into the computer only when accompanied by a C punch.

The seventh channel is a "delete" punch indicating that the other channels are to be ignored when read by a tape preparation unit. This punch is an instruction to the Flexowriter only, and is not sensed by the photoelectric reader. A tape containing an error which is indicated for deletion by the seventh channel will print correctly on a tape preparation unit, but should not be read into the computer, since the error will not only remain, but will add an extra digit to the word.

Tape Preparation:

The Flexowriter and the ElectroData Tape Preparation Unit can be used to prepare punched tape for the computer. Numerical data and numerical equivalents of all symbolic codes, are punched on the tape in the order in which they are to be entered into the computer. It is advisable to leave about 10 inches blank at the beginning of the tape. This is to allow the photocells to warm up and the tape drive to come up to speed. Also, it can be helpful to have some blank tape at the end of the program and at several intervals throughout the program. This

Copyright 1954
by ElectroData Corporation

will facilitate making corrections or insertions if they are necessary.

Flexowriter

The Flexowriter is an electric typewriter with an integral punch for paper tape and a reader for punched paper tape. It is provided with an ElectroData 13-key numerical keyboard in addition to its standard keyboard. It will transfer information as follows:

From	To
Either of the keyboards or Its punched tape reader	Page by printing and Paper tape by punching

Blank tape is fed to the Flexowriter punching unit from a reel. A prepunched tape is inserted into the read unit in preparation for a reading operation. As modified by ElectroData, the Flexowriter will punch binary code when the decimal-digit keys are actuated.

In addition to the format controls of a regular typewriter, the controls of the Flexowriter include an ON-OFF switch and the following six keys:

START READ When depressed, this tab causes the paper tape to move through the reader and the information on the tape to be printed on the Flexowriter.

STOP READ When depressed, this tab stops the tape from moving through the reader.

PUNCH ON When depressed, this tab causes information typed on the keyboard or obtained from the read unit to be punched into tape.

TAPE FEED When depressed, this tab causes blank tape to be fed through the punching unit if the PUNCH ON tab is also depressed.

Copyright 1954
by ElectroData Corporation

CODE DELETE When pressed this tab causes the Flexowriter to punch the seventh channel of the tape. This punch causes the character containing it to be omitted during a subsequent reading operation in the read unit.

STOP CODE Punches holes in the lower three channels of tape. The tape will stop at this point when read by the Flexowriter and will move forward again when the START READ tab is depressed.

If the PUNCH ON tab is depressed and the START READ tab is depressed, then information passing through the reader will be punched on a new tape by the punch, with the exception that blank tape and columns where a "delete" hole is punched in the seventh channel will not be reproduced by the punch.

A 13-key decimal-keyboard may be attached to Flexowriters which are used for tape preparation. When punching numerically coded tapes it is more convenient to use this attachment than to use the typewriter keyboard. In addition to the digits 0 through 9, there is a key which will punch four successive zeros and also a key which produces a finish punch and a carriage-return punch.

Whenever tape is being punched by the Flexowriter, or read through the reader, the Flexowriter, in addition, will type the information it receives. The proof sheet thus produced is an aid in detecting errors punched on the tape. An error is deleted moving the erroneous character to the position directly under the punch dies and pressing the CODE DELETE tab. It is necessary to press the delete tab for each character in order to delete an entire word. If an error is discovered immediately

Copyright 1954
by ElectroData Corporation

after the word is punched, the word may be immediately deleted and the correct word punched before continuing to the next word. If the error is not discovered until later, it is necessary, when reproducing the tape, to stop the read-in of the old tape at the word to be deleted and manually punch-in the correct information via the typewriter or keyboard. The START READ tab is used to resume tape reproduction. The deleted word will not appear on the new tape.

To operate the Flexowriter when preparing a tape for use on the computer:

1. Put the ON-OFF switch at ON.
2. Depress the PUNCH-ON tab.
3. Feed out 8 to 10 inches of blank tape by pressing the TAPE FEED tab.
4. Punch out the tape by using the numerical keyboard. At the end of each word in the program (sign plus ten digits), press the F key, which produces a finish punch indicating the end of the word to the computer and actuating the carriage return on the Flexowriter. The typewritten copy of the program will thus appear as a single column of words.
5. After finishing the program, feed out more blank tape.
6. Check the printed copy of the program against the original document.

To reproduce the tape:

1. Place the leading end of the tape in the Flexowriter reader.
2. Depress the PUNCH ON tab.
3. Feed out blank tape on the new tape by pressing the TAPE FEED tab.
4. Press the START READ tab. This will feed the original tape through the reader and punch a reproduction on the new tape.

Copyright 1954
by ElectroData Corporation

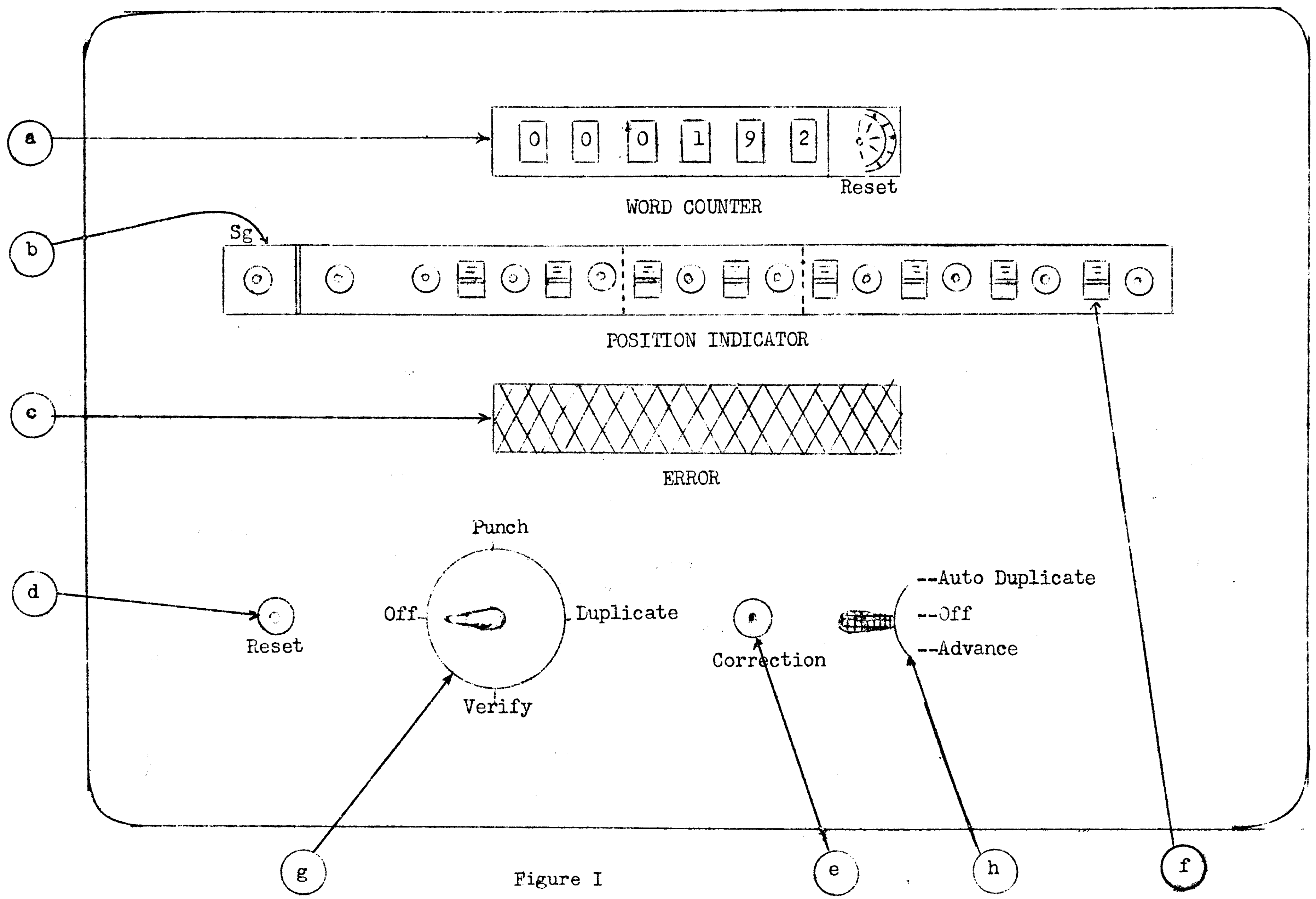

a
0 0 0 1 9 2
Reset
WORD COUNTER
b
Sg
POSITION INDICATOR
c
ERROR
d
Reset
Punch
Off
Duplicate
Verify
Correction
--Auto Duplicate
--Off
--Advance
g
e
h
f

Figure I

5. At the end of the program on the original tape, press the STOP READ tab.

6. Feed out blank tape at the end of the new tape.

7. Remove both tapes from the Flexowriter.

ElectroData Paper Tape Preparation and Verification Unit

Unlike the Flexowriter, this unit has been designed for a "word" as opposed to a "character" mode of operation. This means that the working unit for this equipment is one word: 11 digits, F punch and carriage-return punch. A description of various components of this system follows:

The system consists of the following units:

a. A Commercial Controls Corporation motorized paper tape reader.

b. A Commercial Controls Corporation motorized paper tape punch.

c. An ElectroData or IBM numerical keyboard.

d. An ElectroData paper tape preparation and verification control unit. (Hereafter referred to as Control Unit)

The system will perform the following functions:

a. Punching of characters entered in the keyboard or by a position counter (see "d" of Section 4) into paper tape.

b. Punching of characters entered in the keyboard or by the position counter into paper tape, with simultaneous verification against characters read from a prepunched control tape. The verification is of the "serial" type; i.e., a character entered in the keyboard or by the position counter is checked against the character on the control tape before punching.

c. Duplication of punched paper tapes.

The paper tape reader, the paper tape punch, and the keyboard plug into the control unit. Power for the entire system is supplied by the control unit.

The front of the control unit cabinet serves as a control panel for the

Copyright 1954
by ElectroData Corporation

system. The following controls and indicators for the system are on the control panel. (See Figure I)

a. Word Counter.

This is a six-digit manual-reset counter. The counter is pulsed when an F hole is punched by the paper tape punch.

b. Position Indicator.

This is a series of 11 neon lamps that correspond to the 11 digits of a word; sign and digits 1 through 10. The light indicates the next digit position to be punched. If no light is on, the F or carriage-return position of a word is indicated.

c. Error Light.

This is a red light that glows if a character entered at the keyboard or position counter and a character entered at the reader during the verification operation do not correspond.

d. RESET Button

The digits punched are counted by a position counter. The positions of this counter correspond to the position indicator lights. This counter controls the automatic insertion of F and CR punches (finish and carriage return) after every 11 digits punched. The reset button resets this counter to the "home" position, i.e. in position for punching the sign digit of the next word. The reset condition is indicated on the control panel by the SG neon.

e. CORRECTION Button

The verification of one character is realized by storing, in relay storage, the code for one character read by the reader and comparing this character via coincidence circuits with a character entered from the keyboard or position counter. No character may be punched during the verification operation unless this coincidence exists. Hence, if the character read by the reader is incorrect, some means of storing the correct character in the relays must be employed. The correction button performs this function by activating a relay switching circuit that connects the keyboard with the storage relays. The relay is dropped out after one, and only one, digit has been inserted from the keyboard.

Copyright 1954
by ElectroData Corporation

f. Zero Tab Set Switches.

These switches control the automatic punching of zeros. (See Keyboard Control keys)

g. Function Selector Switch.

This selector switch may be set to one of 4 positions:

OFF

When the selector is in OFF position no power is on the control unit; in any other position power is on.

PUNCH

When the selector is in PUNCH position the keyboard is connected directly to the punch. The punching is under control of the position counter for automatic insertion of F and carriage-return punches.

DUPLICATE

When the selector is in DUPLICATE position the reader and punch are directly connected so that the tape in the reader may be duplicated. The leader, trailer, and all blank parts of the tape in the reader are duplicated on the tape in the punch.

VERIFY

When the selector is in VERIFY position the keyboard and position counter are connected so that the character entered by the keyboard or by the position counter is checked against the character entered in the storage relays from the control tape in the reader. If the two coincide, the character is punched. If they do not, punching is halted and the ERROR light is turned on.

h. Word Advance Key and Duplication Control Switch.

This is a three-position toggle that has one switch position, an off position, and a momentary contact key position.

ADVANCE

The depression of the toggle to this position causes the tape in the reader to be advanced past the next carriage-return punch. This advancement is effected regardless of the position indicator

Copyright 1954
by ElectroData Corporation

setting. If tape is in the reader it can be advanced by this button during any operation except automatic duplication.

With the selector set at VERIFY, characters advanced through the reader will not be stored in the storage relays. The first digit after the carriage-return punch is stored in the relays, available for verification purposes, and the position counter is reset. With the selector set at DUPLICATE, the characters read during the advancement will be duplicated by the punch. With the selector set at PUNCH, the advancement of a tape in the reader will have no effect on the operation of the punch or the position counter.

AUTO DUPLICATE

When the duplication control switch is in this position the tape in the reader will be duplicated if the function selector switch is set at DUPLICATE. It has no effect if the function selector switch is set to any other position.

OFF

This is the normal position of the duplication control switch. The OFF position can be used to interrupt automatic duplication.

5. The following control keys are located on the keyboard.

a. ZERO TAB

The depression of this key initiates the emission of zeros for both punching and verifying operations. Zeros are punched until the end of the word, as defined by the position counter, is reached, or until a zero tab stop is encountered, or, in the case of verification, if an error is detected. One or more tab stops may be set. The number of zeros punched will depend on the tab stop settings and on the position of the word to be punched when the ZERO TAB key is depressed. Tabbing to the end of a word causes the 11 zeros to be punched and the F and carriage-return punches to be inserted at the end of the word.

b. ZERO FINISH

The depression of this key causes zeros to be emitted for punching and verifying operations. This key can be used to emit a maximum of 11 zeros (one word) or a minimum of 1 zero. The word, as defined by the position indicator, will be filled with zeros on the right and an F and carriage-return character punched. The ZERO TAB stops do not affect this operation. If

Copyright 1954
by ElectroData Corporation

an error is detected on verification, the emission will halt and the position indicator will indicate the position of error.

6. The keyboard is disconnected when the function selector is set at DUPLICATE.

The use of this unit for tape preparation is described below:

<u>Punching</u>

To prepare the system for punching, the control panel is set up as follows:

1. Reset the WORD COUNTER to zero.

2. Turn the function selector switch to PUNCH.

3. Make sure the POSITION INDICATOR is reset. (The sign position indicator light on.) If necessary depress the RESET button.

4. Set the ZERO TAB stops as indicated by the nature of the information to be punched. If the data consists of coded commands, a tab stop before the first order-digit is probably most useful. For other data, the pattern of zeros will determine the tab stop settings. Tab stop settings may be altered during the punching operation as the nature of the zero pattern in the data warrants.

5. Start punching the program via the keyboard. After 11 digits are punched, the F and carriage-return punches are automatically produced and the word counter advances. This can be used as a partial check on punching. If a punching error is observed, the ZERO FINISH key should be pressed and the word in error repunched.

   The ZERO TAB key and ZERO FINISH key may be used wherever they are appropriate for speeding up the punching operation.

   After the original document is transcribed, note on the original program the number in the word counter, the correct number of words, if known, the number of error words on the tape, and any other information that might help the verifier operator to interpret errors.

   If there is an interruption in the punching operation, the number in the word counter and the lighted neon lamp in the position indicator will aid in resuming at the proper digit. If there is any doubt of the correctness of a word punched, assume an error.

Copyright 1954
by ElectroData Corporation

Fill the word out with zeros and repunch.

After the tape has been punched, have it verified, preferably by another person.

Verification

Some of the errors encountered in paper tape preparation are listed below:

1. Misinterpretation of the data in the original document by the punch operator, resulting in erroneous or transposed digits.
2. Skipping of words by the punch operator.
3. Inadvertent repetition or insertion of extra words by the punch operator.
4. Malfunction of the tape preparation unit, which may cause any one of the types of errors listed above and, in addition, may result in addition or deletion of single characters.

The ElectroData paper tape preparation and verification system is designed to detect these errors.

1. Insert the tape punched from the original document into the reader. This is referred to as the "control tape".
2. Reset the word counter.
3. Turn the function selector switch to VERIFY.
4. Press the RESET button. This causes the tape to advance to the first character punched. This character is transferred to the storage relays. The leader of the control tape will be duplicated by the punch.
5. Enter the data via the keyboard from the original document which was used in preparing the control tape. As each character is entered at the keyboard or by the position counter (F or carriage-return punch), it is checked against the character in the storage relays.

   If coincidence exists, the character is punched and the next character in the control tape is automatically read into the storage relays. F characters punched during the verifying operation are accompanied

Copyright 1954
by ElectroData Corporation

by a "1" punch. This identifies verified words (tapes) so that they may easily be recognized by operators, coders, and clerks. When a verified tape is printed on a Flexowriter an "r" will be printed after each word; unverified words will be followed by an "a".

All the data is punched on the new tape according to the same procedures which were used on the control tape, until the program is finished or an error is detected.

An error is detected when the character in the storage relay and the character entered from the keyboard or the position counter do not coincide. When an error is sensed, the ERROR light goes on and the punching stops. The ERROR light will be turned off when coincidence is realized. The procedure for turning the ERROR light off, classified by types of error, will usually be one of the following.

Digit in Error. Look at the position indicator to tell which digit is in error, read this character from the original document, and enter it in the keyboard. If this does not turn the light off, an error exists on the control tape. To correct the error, depress the CORRECTION button and enter the correct digit at the keyboard. The digit is transferred to the storage relays but is not punched. Then enter the correct digit at the keyboard again. If the same digit is entered both times, it will be punched, the control tape will advance one position, and the ERROR light will be turned off. If the two digits do not coincide, the ERROR light will stay on as the second digit is entered, and the same procedure must be followed again.

Word in Error. If a word is discovered to be in error, and the position indicator is set at the SG position, press the ADVANCE button. This skips the word and continues with verification. If part of the verification of the word is completed before it is discovered that the entire word is misplaced, the control tape must be advanced to the corresponding position of the next word by using the manual knob on the reader. The

Copyright 1954
by ElectroData Corporation

first digit after the manual advancement has been made must be entered by using the procedure described under DIGIT IN ERROR, above, since a new digit from the control tape will not be transferred into the storage relays until the next digit is punched.

Omission of words. Occasionally the punch operator will fail to punch a word or, rarely, an entire group of words. To make sure that all of the omitted words carry the "verified" punch((a 1 (one) punch accompanying the F punch)) after they are inserted, one of the following procedures must be followed:

a. To insert a small number of words:

1. Using the manual knob on the reader, back up the control tape n full words, where n is the number of words to be inserted. The reader will be at the sign position of the nth word. If the number of words missing is not known, back up to the sign position of the previous word on the control tape before inserting each word.

2. Follow the procedure for correcting digits in error, described earlier, for each digit of each word missing. Zeros must be entered singly from the keyboard rather than with the ZERO FINISH or ZERO TAB keys. After the 11 digits of a word have been entered in this fashion the position counter will be at F and the F and carriage-return characters emitted by the counter should correspond with the F and carriage-return punches on the control tape, since the control tape advances one digit position for each digit punched during the insertion. If the tapes do not correspond, the ERROR light will go on at the F and carriage-return position-indication and the punching of these characters may be forced by simply pressing the CORRECTION button.

After the word or words have been inserted, the position indicator will be at SG. If the operator has backed up the control tape correctly the verification may proceed without adjustment at the end of the insertion, since the sign digit of the first word after the insertion will be in the storage relays. If the backing up was not done properly and the control tape is not aligned with the original document, the operator may back up the tape to the sign position of the word just previous to the word where verification is to start and press the ADVANCE key. This will cause the sign digit of the first word to be

Copyright 1954
by ElectroData Corporation

read to the storage relays. The operator may elect to align the control tape by hand and insert the first digit using the correction procedure already described. The verification may proceed in normal form after the first digit is entered in this way.

<u>The position indicator, the location (within the word) of the digit in the reader, and the location (within the word) of the digit being read from the original document must correspond for correct verification.</u>

To insert groups of words:

1. Remove the tape from the punch (do not tear punched tape from the reel supply in the punch) and insert a piece of tape long enough to punch the deleted group.

2. Transfer the function selector switch to PUNCH and punch the deleted group of words on a "stub" tape. (NOTE: The WORD COUNTER will be off by the number of words punched during this operation.)

3. Remove the control tape from the reader, marking the next word on the tape to be verified, and enter the control stub tape for the deleted words.

4. Replace the original tape in the punch and switch back to VERIFY. Verify the deleted information, replace the original control tape in the reader, and continue.

<u>Machine Malfunction during the Punching Operation.</u> The deletion of a binary digit by the punch will be detected and corrected as described in DIGIT IN ERROR. The deletion or addition of an entire character by the punch will necessitate realignment as described in WORD IN ERROR.

The F and carriage-return punches may be forced by depressing the CORRECTION button when the ERROR light is on and the position counter is at one of these positions.

After the tape is completed, the verifier operator should check the number in the word counter against the number of words on the original document as a final check on the preparation. <u>Each word on every tape prepared</u>

Copyright 1954
by ElectroData Corporation

and returned must be accompanied by the 1 (one) punch indication under F punch.

The leader, all blank areas, and the trailer of the control tape will be reproduced on the verified tape.

Duplication

To set up the system for the duplication of master tapes:

1. Place the master tape in the reader at the very beginning of the leader.
2. Set the duplication control switch to OFF.
3. Reset the WORD COUNTER.
4. Set the function selector switch to DUPLICATE.
5. Throw the duplication control switch to AUTO DUPLICATE.

The operation is halted when the master tape runs out of the reader, or when the duplication control switch is turned to OFF. The leader, all blank areas, and the trailer of the master tape will be reproduced on the copy.

The copied tape is exactly like the master with the exception that all duplicated tapes will automatically be punched with a 2 punch accompanying the F and 1 punch of each word. This serves to identify copied tapes so that they will not be confused with verified master tapes. This combination prints as an "s" on the Flexowriter.

After a tape is duplicated, the number in the WORD COUNTER must agree with the pre-recorded number of words on the master tape. The sensing of the end of the master tape locks up the system. In order to continue operation the RESET button must be pressed.

Copyright 1954
by ElectroData Corporation

Duplication will often be used to correct, add to, or delete from, master tapes.

To correct a tape:

1. Watch the WORD COUNTER until it approaches the desired number, and then turn the duplication control switch to OFF.

2. Advance the duplication a word at a time until the word to be corrected is next to be read.

3. Turn the function selector switch to PUNCH and enter the correction. The ADVANCE key is depressed once to by-pass the incorrect word on the master tape. The function selector is turned to DUPLICATE, the duplication control switch is set to AUTO DUPLICATE and the operation continues.

To add to a tape:

Follow the same procedure as for corrections but do not advance the master tape after the punching of the additional words.

To delete from a tape:

1. Use the WORD COUNTER to locate the words to be deleted.

2. Turn the function selector switch to PUNCH and use the ADVANCE key to move the master tape past the word or words to be deleted.

3. Continue duplicating.

   Since the POSITION INDICATOR and position counter do not operate during duplication, the POSITION INDICATOR should be inspected before punching corrections or additions to make sure it is reset. If it is not, depress the RESET button.

<u>Use of punched paper tape as output</u>

Computer output to the punched paper tape system may be either tape from the console tape punch or printed page on the Flexowriter. The OUTPUT selector on the console may be set to TAPE, PAGE, or OFF. Format control

Copyright 1954
by ElectroData Corporation

is available for both types of output, either coded on the tape or carried out in the format of the printed page. This format control may be exercised either by internal programming or externally by setting switches on the typewriter control.

Tape output can be in several different forms depending upon the future use of the tape.

With the output selector switch set at PAGE and with the normal patch-panel wiring; i.e. FROM CONSOLE TO FORMAT, FROM FORMAT TO FLEXOWRITER, the information will be printed by the Flexowriter. If the Flexowriter PUNCH ON key is depressed, the information being printed will be punched on tape simultaneously.

If it is desired to have the output on tape without a typewritten record, setting the output selector switch to TAPE will punch out the information by means of the computer console punch. With a change in the patch-panel on the back of the typewriter control so that it reads FROM READER TO FORMAT, FROM FORMAT TO FLEXOWRITER, the Flexowriter will be operating independently of the computer. If the end of the tape being punched by the console punch is fed through the Flexowriter reader, the information on it will then be printed by the Flexowriter. Since the console punch operates at almost twice the speed of the Flexowriter, the punching operation will end before the printing is finished and while computations on the computer go on -- the effect being that of a buffer.

During the running of a problem, minor changes on the tape can be made without using the tape preparation units. Any holes which should not

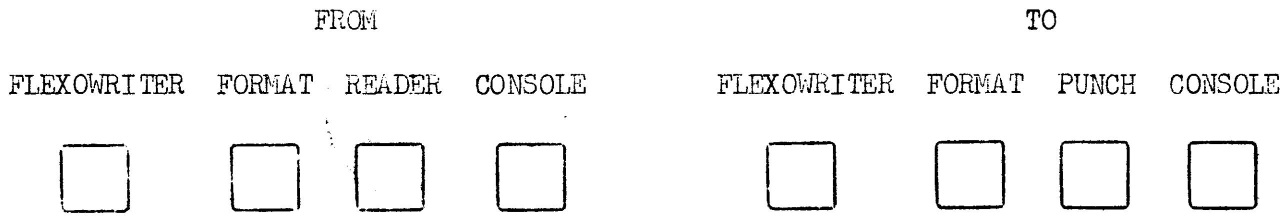
Copyright 1954
by ElectroData Corporation

have been punched can be covered by opaque tape so that the light from the photoelectric reader will not pass through them. New holes may be punched on the tape by a hand punch which will punch one hole at a time. The original tape can be cut into two parts, a section of tape inserted, with careful alignment, and the ends of the tape spliced with opaque tape.

<u>Typewriter Control</u>

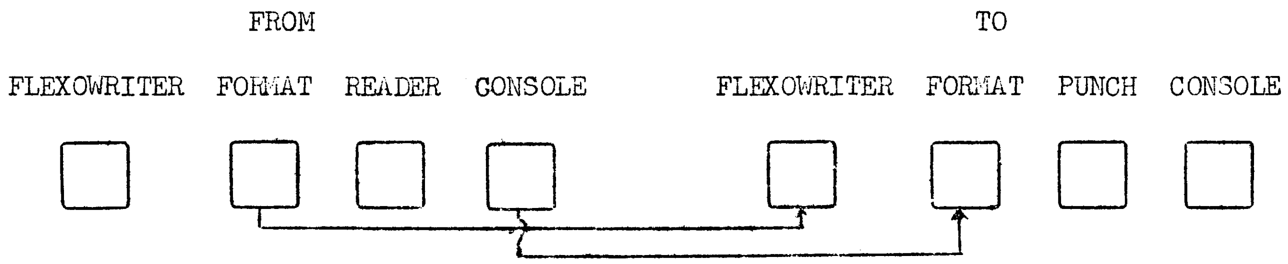
The typewriter control has sometimes been called the "format control" or simply the "format." It translates computer information for the Flexowriter, and, by means of switches and controls, governs page format when this is not determined within the computer by the printout command itself. The Flexowriter is mounted on the typewriter control.

<u>Patch Panel</u> The patch panel on the back of the typewriter control contains eight plugs, as shown below:



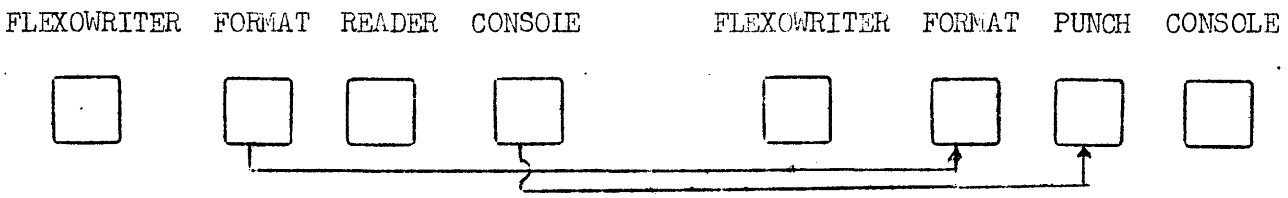
The FROM plugs refer to sources of information and the TO plugs refer to devices to which information is being sent. Since all Flexowriter and punched-tape information going to or from the computer passes through the console, the FROM CONSOLE plug refers to information in machine language (binary-coded decimal). The PUNCH plug refers to the Flexowriter punch only; the console punch puts out only untranslated binary-coded decimal information. Normal wiring for Flexowriter printout is

Copyright 1954
by ElectroData Corporation



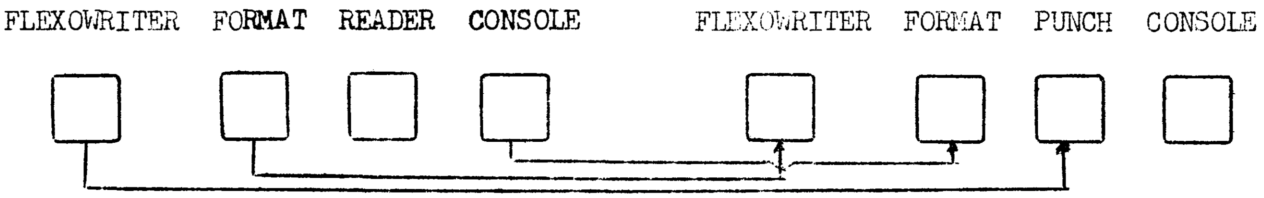
With this arrangement, information goes from the console to the typewriter control and, from the typewriter control, translated, to the Flexowriter, where it is printed out.

Other useful wirings are:



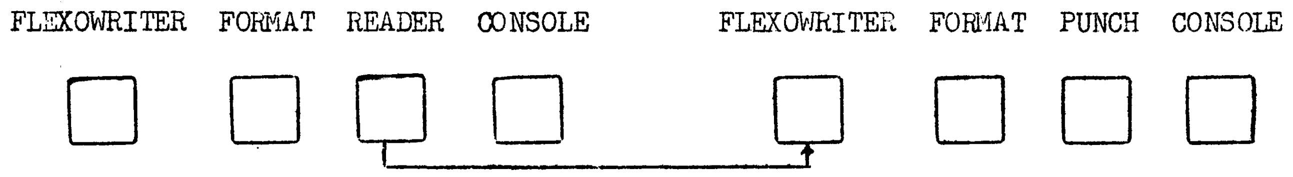
With this arrangement, information from the computer is translated and punched on the Flexowriter punch. Included are translated format instructions from the computer and those determined by settings of the switches on the front of the typewriter control panel.

The two preceding arrangements may be combined as

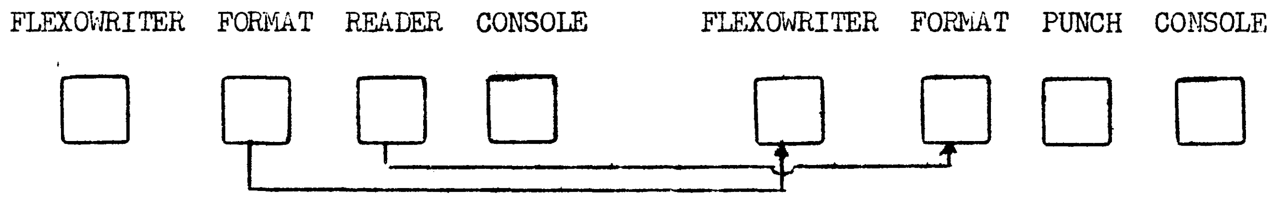


With the arrangement above, information may be printed and punched at the same time if the Flexowriter PUNCH ON tab is depressed. Blank tape can be generated by pressing the Flexowriter TAPE FEED tab. Tape produced by any of the foregoing arrangements can be printed with

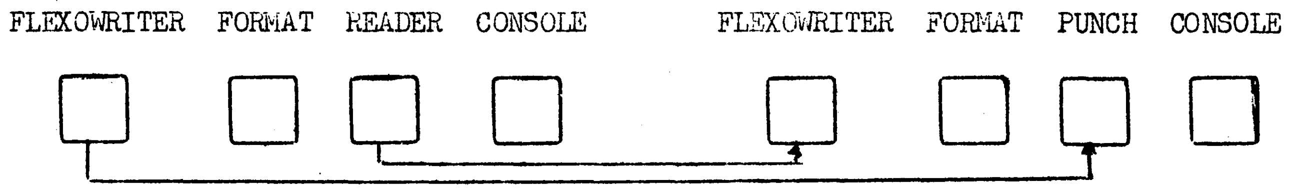
Copyright 1954
by ElectroData Corporation



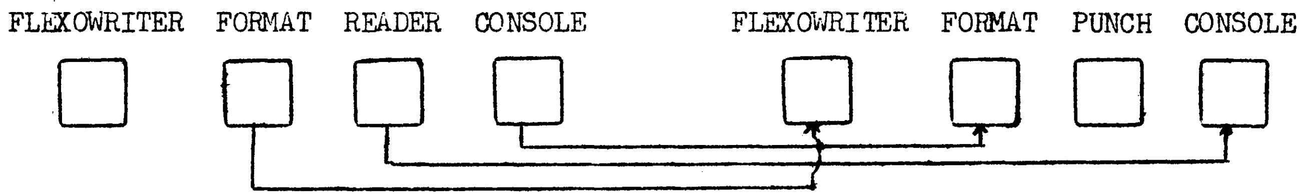
Tape from the console punch can be printed with



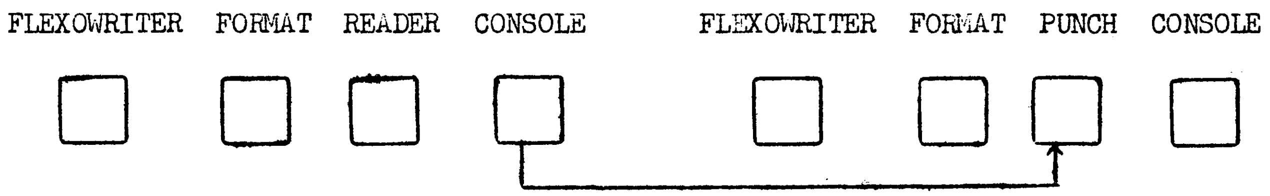
Format instructions punched on tape are executed and format may be controlled by the settings of switches on the typewriter control.



The arrangement above reads translated tape, prints, and reproduces tape simultaneously.



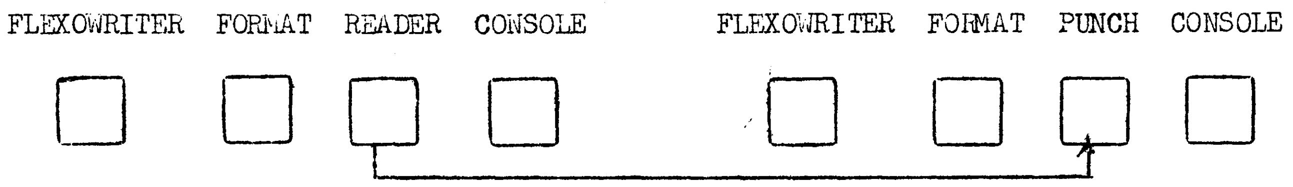
The arrangement above reads information into the computer from the Flexowriter reader and prints out on the Flexowriter. INPUT SELECTOR is set at MECH READER, OUTPUT SELECTOR is set at PAGE.



The arrangement above will produce tape containing untranslated word information and untranslated format instructions which will be carried out when the tape

Copyright 1954
by ElectroData Corporation

is read through the FORMAT plug to FLEXOWRITER and/or PUNCH. Tape from the preceding arrangement is identical with that produced by the console punch.



The arrangement above is the only one which will reproduce a tape exactly, except for "delete" holes. Format instructions, including the 1 for feeding out one character-space of blank tape, appear as numerical information. (The tape in this case contains a 1 punch meaningless to the computer and the typewriter control since it is unaccompanied by a C pulse.) Lengths of unpunched tape going into the reader will be matched by equal lengths of output tape unpunched. While this reading and punching operation is in progress, the printing mechanism of the Flexowriter can be used, with appropriate patch panel connections, for computer output.

Typewriter Control Panel

WORDS/LINE Switch The WORDS PER LINE switch enables the operator to set the number of words per line, from 1 to 20.

LINES/GROUP Switch and Grouping Switch A group is a number of lines that are equally spaced. Between groups the spacing is doubled. The LINES PER GROUP switch sets the number of lines per group, from 1 to 20. The spacing into groups is in effect when the GROUPING switch is ON. If the GROUPING switch is OFF, then all lines are equally spaced.

GROUPS/PAGE Switch and AUTO STOP Switch Any number of groups from 1 to 20 constitutes a page. The GROUP/PAGE switch sets the number of groups. At the

Copyright 1954
by ElectroData Corporation

end of a page, the computer will stop if the AUTO STOP switch is ON. This is so even if the GROUPING switch is OFF.

Using the following notation:

L = LINES/GROUP switch setting

G = GROUPS/PAGE switch setting

C = number of lines advanced by carriage return, 1 to 3, set on Flexowriter carriage (6 lines = 1 inch)

= 1 if GROUPING switch is ON
  0 if GROUPING switch is OFF

Then N, the number of lines of printed information per page is given by

$$N = L \times G$$

and I, the number of inches to a page is given by

$$I = \frac{C}{6}\ (L \times G) + (G - 1)$$

If the AUTO STOP switch is off, then the computer will not stop at the end of a page.

<u>RESET Button and Indicator Light</u> If the computer stops at the end of a page, the operator can tear off the page and resume the computation by pressing the RESET button, which sets all format counters to zero. The RESET state is indicated by the flowing of the RESET indicator light. If the operator desires to type a heading on the new page manually, the computer should be stopped with the STOP button. After the heading is typed, computation can be resumed by pressing the RESET button and the CONTINUOUS button.

<u>Zero Suppress</u> If the ZERO SUPPRESS switch is in the ON position, the first digit to be printed after the sign will be the first non-zero digit. If the

Copyright 1954
by ElectroData Corporation

ZERO SUPPRESS switch is OFF, leading zeros are not suppressed.

However, if the printout command calls for a decimal point to replace the sign, there is no ZERO SUPPRESS for that printout.

SPACE-OFF-TAB Switch The setting of the SPACE-OFF-TAB switch determines the horizontal spacing between printouts. With the switch at SPACE, the Flexowriter will insert a typewriter space between printouts. With the switch at OFF, there will be no spacing between printouts. With the setting at TAB, the spacing between printouts will be determined by the operator's tab settings.

IBM CARD SYSTEM

Card Preparation An installation using IBM input-output will have auxiliary equipment for card preparation.

The ElectroData punched card system will read or punch up to eight words per card, each word containing algebraic sign and up to 10 digits. The sign of a word is indicated by an x punch over a suitable column of the field.

The ElectroData punched card converter takes information from cards and transforms it to information used by the computer. Similarly, it will take information from the computer and convert it to a form suitable for punching on cards.

Of the pieces of IBM equipment available for reading and punching, - any reproducing punch or summary punch may be used, such as the IBM 513, 514, 516, 517, 519, 523 - a combination of any two can be used as a card reader and card punch. The IBM 528 may be used as a combination card reader and card punch.

Information may be printed out on an IBM tabulator.

The commands referring to IBM equipment are the same for all units.

Copyright 1954
by ElectroData Corporation

The front panel of the ElectroData punched card converter contains two selector switches - one for input and one for output. The switches determine the number of words, from one to eight, read from each card, punched on each card, or printed on each line by the tabulator.

The plugboards on the IBM input-output devices are wired to allow information to pass between them and the converter. Suggested wiring for each type of punched card machine is indicated by ElectroData Corporation.

Input From Punched Cards

1. Place cards to be read into hopper of IBM card reader.
2. Set ON-OFF switch on reader to ON and press START button to allow first card to reach station just prior to reading stations.
3. Set input dial on converter for the number of words per card.
4. If cards are to be punched or results printed during computation, set output dial on converter for the number of words per card or per line.
5. Put into the A register by means of the keyboard or the supervisory control panel the word +mmm044xxxx, where mmm is 1000 minus the number of cards to be read and xxxx is the first computer memory location to be filled by input data.
6. Put into the C register the word 44 xxxx yyyy, where xxxx is the first memory location to be filled by input data and yyyy is the address of the next command to be executed.
7. Use other switches on the console the same as for other commands. (Note that the setting of the INPUT selector is immaterial, since it is effective only on an "in" command.)
8. To start read-in of cards, press CONTINUOUS button on console.

Output From Punched Card Machines

Two forms of output from punched card machines are possible - punched cards or printed lines. The same command is used for both. To punch, plug the cable

Copyright 1954
by ElectroData Corporation

connector from the punch unit into the WRITE output assembly of the converter. To print, plug the cable connector from the tabulator into the WRITE output of the converter.

Setting of the output switch on the converter determines the number of words per card to be punched or words per line to be printed.

Time

The amount of time required to read information into or out of the computer is dependent upon the type of equipment used. Below is a summary of the time required for input or output for each system.

Input Medium	Reader	Speed	Comments
Perforated paper tape	ElectroData Photoelectric reader	540 decimal characters per second.	The entire 4000 word main storage can be filled in about 1.5 minutes.
	Flexowriter mechanical reader	10 decimal characters per second.	
Punched IBM cards	513, 514 517, 523, etc.	100 cards per minute (150 decimal characters per second maximum.	Each card may contain eight ten digit words.
	528	200 cards per minute (300 decimal characters per second minimum.)	

Output Form	Punch or Tabulator	Speed	Comments
Typewritten page	Flexowriter with ElectroData code	9 decimal characters per second.	Can control format by coding or by external switches.
Perforated paper tape	Flexowriter punch or console punch	20 decimal characters per second.	Gain over Flexowriter typeout partly because of omission of spaces and carriage returns.

Copyright 1954
by ElectroData Corporation

Output Form	Punch or Tabulator	Speed	Comments
Punched cards	IBM 513, 514, 517, 523, 528, etc.	100 cards per minute or up to 150 decimal characters per second.	Up to 8 words per card.
Tabulated Page	IBM 402, 407, etc. Tabulators	100 or 150 lines per minute	Up to 8 numeric words per line. Can be wired for alphabet printout.
	416	150 lines per minute.	Numeric only.

Copyright 1954
by ElectroData Corporation

ELECTRODATA CORPORATION FACILITIES

ElectroData Corporation recognizes that as a manufacturer of computers and auxiliary equipment, it has responsibilities to its customers, to organizations interested in computing equipment, and to the public in general. To fulfill these responsibilities, the company sponsors certain educational and professional activities. This section of the manual describes these activities so that purchasers of ElectroData equipment may know about services that they may draw on.

CODING ASSISTANCE

Coding Course

A detailed course in coding for the ElectroData computer is regularly scheduled, generally twice a month. The size of the class is held to between six and ten registrants, so that individual instruction can be given. The content of the course is mainly instruction in how to use the machine code, and in some of the programming techniques that will be of use to the potential coder or programmer. The course also gives the student an opportunity to follow through the complete procedure of working up a code, preparing the paper tape and cards for the computer, operating the machine, performing machine checking, and finally seeing the results emitted in one or more of the output forms.

Sample Coding

ElectroData maintains a staff of mathematicians trained to analyze, program, and code problems of diverse origin, and solve these problems on the computer. They will solve sample problems for any organization interested in acquiring a machine, to demonstrate that the machine is capable of performing the work

Copyright 1954
by ElectroData Corporation

required. ElectroData mathematicians set up an initial code and tape for standard problems that the organization may have. This serves to indicate to the user of computing equipment what is involved in an operation of this type and allows him to estimate the time, the personnel, and the operations involved. Organizations which use ElectroData coding facilities on this basis may elect to do the analysis or to code the problem, or perhaps participate in the final machine solution. In this case, the ElectroData staff will give assistance and will verify codes or other work. They will also indicate how the work may be improved to increase efficiency, but they will not act upon these suggestions unless they are approved by those who submit the problems.

Field Coding

The purchaser of an ElectroData system may often want to have initial codes worked up by a trained staff, or to have trained personnel temporarily assigned to his staff. This service is available at a reasonable fee.

TRAINING

Personnel Reserve

ElectroData offers scholarship awards to universities to finance the training of students, usually graduates, in the use of ElectroData equipment. In a summer session, the students work with the ElectroData organization and learn analysis, and coding and operating techniques. Upon completing this summer work, they have an excellent background to serve as key members of computing organizations built around the ElectroData computer. The names of students who have completed such training are available to organizations using ElectroData equipment.

Copyright 1954
by ElectroData Corporation

INFORMATION DISSEMINATION

Field Conferences

ElectroData mathematicians are in attendance at many professional conferences where they will gladly supply information to anyone interested in any aspect of the company's activities. Research and technical papers on subjects related to automatic computation are presented by the ElectroData staff throughout the country.

Sales Representation

ElectroData's trained mathematicians supplement the services of the sales force by giving advice on detailed and complex mathematical questions which may arise. By mail or personal visit, ElectroData's mathematical staff is available to answer such questions. It has also available a roster of consultants who will be of service on exceptional problems.

Computing

In the diversified work of the ElectroData Computing Center, it is necessary to formulate many varied routines, subroutines, and types of higher-order coding. ElectroData has a considerable length of experience since it has had an operating installation since May, 1953. The aggregate of ElectroData's experience, resources, training, and publications is available to interested organizations.

Symposia

To develop information and facilitate its exchange, ElectroData will periodically hold symposia for users of its equipment. It has been found in similar symposia for various types of equipment produced by ElectroData's parent company, Consolidated Engineering Corporation, that the information gained by the

Copyright 1954
by ElectroData Corporation

participants has proved extremely valuable to their operations.

CONTRACT COMPUTING

Sample problems that can be handled at ElectroData for potential customers were referred to in the preceding paragraphs. In many cases, an organization would prefer to have partial use of a piece of equipment over an extended period of time to insure that the equipment will be successfully used by their installation. Also, an organization with a relatively small but increasing work load may want partial use of a computer for the transition period before they are able to keep one busy full time. To satisfy this need and desire, the staff of ElectroData, with their own personnel or in conjunction with the personnel of the customer organization, will do the work on a contract basis.

Copyright 1954
by ElectroData Corporation

TECHNICAL BULLETIN 3040

ELECTRODATA CORPORATION

AN AFFILIATE OF CONSOLIDATED ENGINEERING CORPORATION

717 N. Lake Avenue, Pasadena 6, California

ELECTRONIC DATA PROCESSING EQUIPMENT FOR SCIENCE, INDUSTRY AND COMMERCE