650

magnetic drum data-processing machine

manual of operation

MAJOR REVISION (June 1955)

This edition, Form 22-6060-1, obsoletes Form 22-6060-0 and Form 22-6149-0. Significant changes have been made throughout the manual, and this new edition should be reviewed in its entirety.

CONTENTS

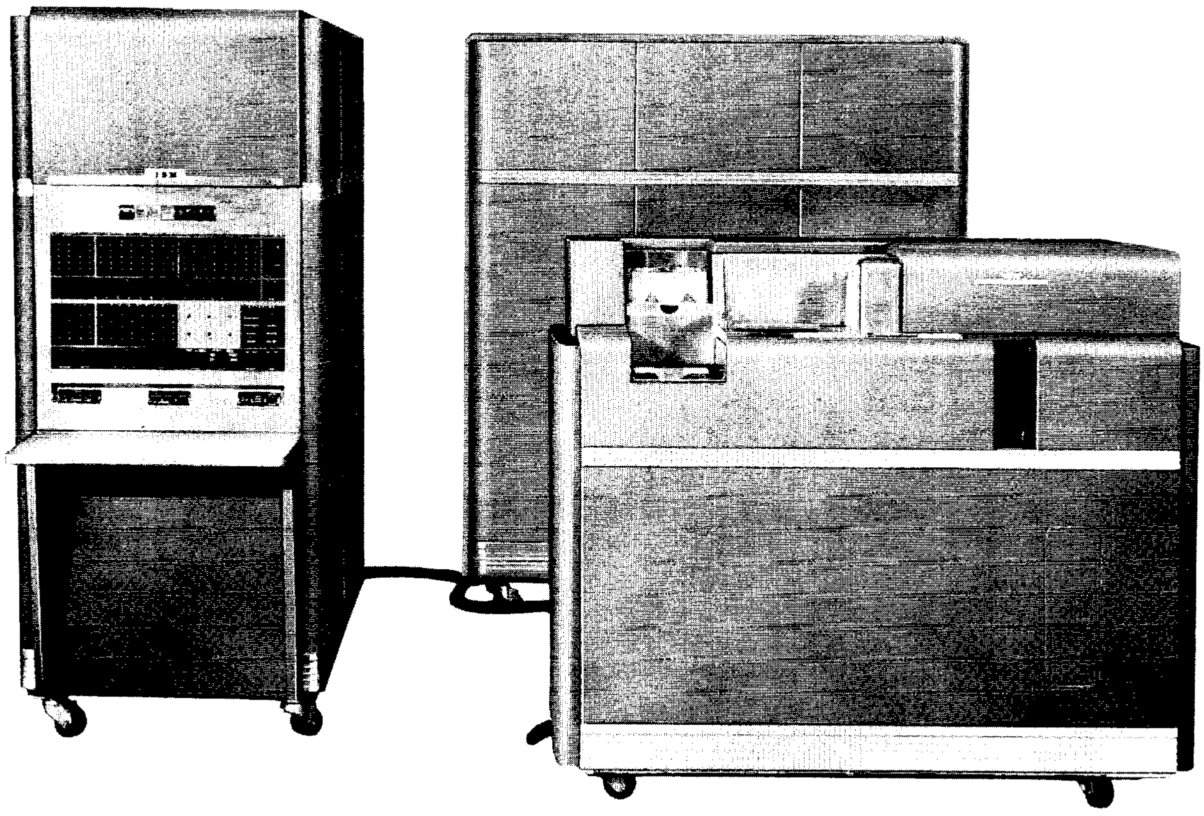


Figure 1. Magnetic Drum Data-Processing Machine

Type 650

MAGNETIC DRUM DATA-PROCESSING MACHINE
Type 650

THE TYPE 650 Magnetic Drum Data-Processing Machine has been developed to offer features heretofore available only in large-scale electronic data-processing machines. The result is an electronic machine intermediate in speed and capacity between the widely accepted IBM Type 604 Electronic Calculator and the more recent IBM electronic data-processing machines. The Type 650 has all the advantages of stored programming, high component reliability, self-checking, automatic operation, compact design, and ease of operation. Because of its great flexibility, the Type 650 serves the needs of accounting, engineering, and scientific computing fields in a most efficient manner.

The Type 650 is a stored-program, modified single-address, numerical, decimal machine. It is available in two models with memory capacities of 10,000 or 20,000 digits. The use of IBM punched cards for both input and output permits easy integration into existing punched-card systems. The flexibility of the Type 650 frequently permits the elimination of clerical operations as well as auxiliary punched-card operations, such as sorting, merging, or gangpunching.

Figure 1 shows the three units that make up the Type 650 Magnetic Drum Data-Processing Machine:

1. Type 533 Read-Punch Unit (right foreground)
2. Type 650 Console Unit (left)
3. Type 655 Power Unit (right rear)

Type 533

Input and output are through the Type 533 Read-Punch unit. A maximum input rate of 200 cards, or 16,000 digits, per minute is possible. Maximum output speed is 100 cards or 8000 digits per minute. A control panel is provided to allow flexibility of input and output.

Type 650

The Type 650 Console Unit contains the calculating units, the control console, and the magnetic drum.

The heart of the Type 650 is the magnetic drum on which may be stored the information necessary to process a problem. The drum is a cobalt-nickel plated cylinder about 4 inches in diameter and 16 inches in length, which revolves at a speed of 12,500 revolutions per minute.

The console of the Type 650 (Figure 2) is the operator's means of simple and immediate communication with the machine. Information can be manually read into internal memory from storage-entry switches or displayed from internal memory in display lights. These functions will be discussed in detail later.

The calculating units will accumulate 10-digit factors to form a 20-digit total, perform 10-digit by 10-digit multiplication to develop a 20-digit product, and divide a 20-digit dividend by a 10-digit divisor to develop a 10-digit quotient and a 10-digit remainder. Complete automatic sign control is provided.

Logical and control-digit tests are provided to determine which path the program will follow.

The machine is capable of performing an automatic table lookup operation. This feature makes practical the solution of problems requiring table references, which occur frequently in both commercial and scientific applications.

Type 655

The Type 655 Power Unit contains the power supplies for all the machine units. It also contains the necessary circuitry to translate the decimal input into the machine code, and the machine code back into decimal for output.

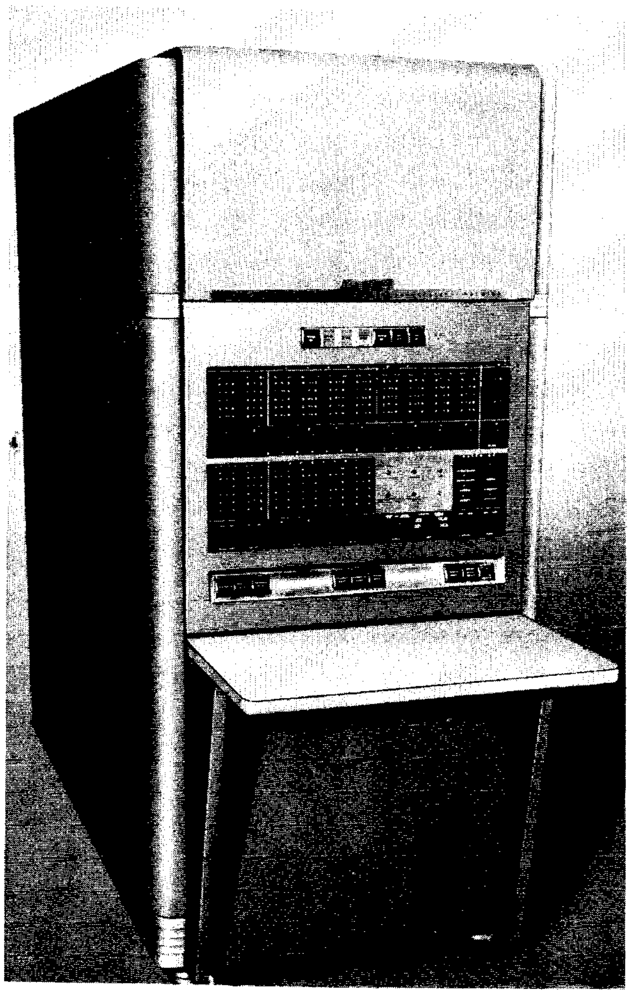


FIGURE 2. TYPE 650 CONSOLE UNIT

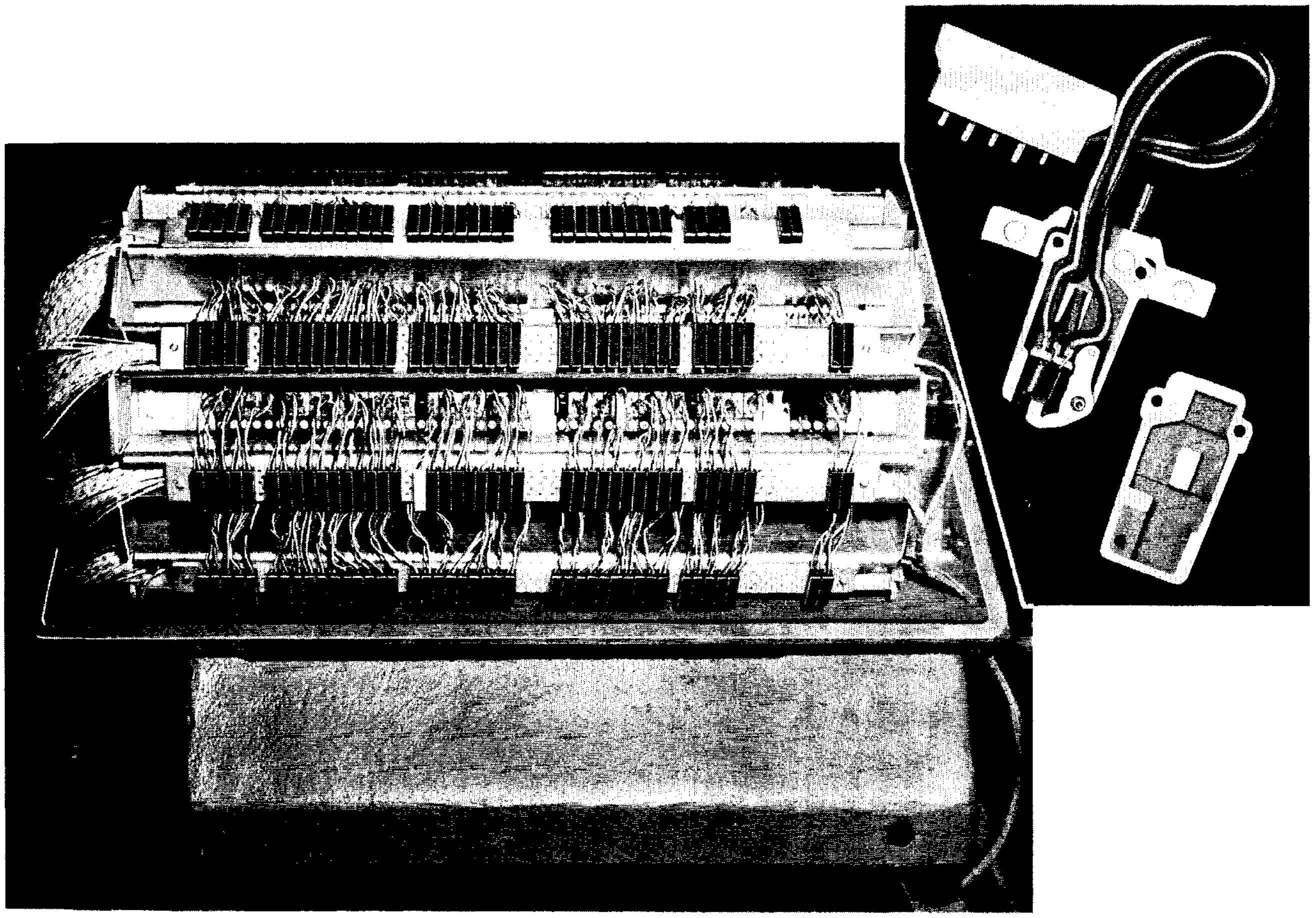


FIGURE 3. MAGNETIC DRUM ASSEMBLY AND READ-WRITE HEAD (INSET)

STORAGE

THE MAGNETIC DRUM provides the general storage or memory for the Type 650. Information is stored on the surface of the drum in the form of magnetic spots. Each digit is identified by a recognizable pattern of these magnetic spots.

Reading and recording the magnetic spots on the drum are accomplished by devices known as read-write heads (Figure 3). These heads are mounted on bars around the drum.

Any magnetized spot recorded on the drum will remain there permanently, or until it is erased by recording another spot in the same location. The machine may be turned off completely without any danger of loss of magnetized spots from the drum surface.

A different method of storage is used in the arithmetical and control units. These units employ condenser storage. A storage of this type is used to provide the speed necessary for calculating. Condenser storage will retain a factor only as long as power is supplied or until another digit is read into the unit. Any information contained in one of the condenser storage units will be lost when the power is turned off.

Information is handled within the Type 650 ten digit positions at a time. Thus, the basic unit of storage contains ten digits and an algebraic sign. Each unit of ten digits and a sign is known as a *word.* The magnetic drum has a capacity for either 1000 or 2000 words (10,000 or 20,000 digits). Several independent factors may be stored within a single ten-digit word.

The algebraic signs of words may be read from punching in the input cards or may be automatically supplied by the machine. Conventional sign indications are 12's for positive factors and 11's for negative factors.

	0		1		2		3		4		5		6		7		8		9	
	00	50	00	50	00	50	00	50	00	50	00	50	00	50	00	50	00	50	00	50
READING	01	51	01	51	01	51	01	51	01	51	01	51	01	51	01	51	01	51	01	51
	02	52	02	52	02	52	02	52	02	52	02	52	02	52	02	52	02	52	02	52
	03	53	03	53	03	53	03	53	03	53	03	53	03	53	03	53	03	53	03	53
	04	54	04	54	04	54	04	54	04	54	04	54	04	54	04	54	04	54	04	54
	05	55	05	55	05	55	05	55	05	55	05	55	05	55	05	55	05	55	05	55
	06	56	06	56	06	56	06	56	06	56	06	56	06	56	06	56	06	56	06	56
	07	57	07	57	07	57	07	57	07	57	07	57	07	57	07	57	07	57	07	57
	08	58	08	58	08	58	08	58	08	58	08	58	08	58	08	58	08	58	08	58
	09	59	09	59	09	59	09	59	09	59	09	59	09	59	09	59	09	59	09	59
	10	60	10	60	10	60	10	60	10	60	10	60	10	60	10	60	10	60	10	60
	11	61	11	61	11	61	11	61	11	61	11	61	11	61	11	61	11	61	11	61
	12	62	12	62	12	62	12	62	12	62	12	62	12	62	12	62	12	62	12	62
	13	63	13	63	13	63	13	63	13	63	13	63	13	63	13	63	13	63	13	63
	14	64	14	64	14	64	14	64	14	64	14	64	14	64	14	64	14	64	14	64
	15	65	15	65	15	65	15	65	15	65	15	65	15	65	15	65	15	65	15	65
	16	66	16	66	16	66	16	66	16	66	16	66	16	66	16	66	16	66	16	66
	17	67	17	67	17	67	17	67	17	67	17	67	17	67	17	67	17	67	17	67
	18	68	18	68	18	68	18	68	18	68	18	68	18	68	18	68	18	68	18	68
	19	69	19	69	19	69	19	69	19	69	19	69	19	69	19	69	19	69	19	69
	20	70	20	70	20	70	20	70	20	70	20	70	20	70	20	70	20	70	20	70
	21	71	21	71	21	71	21	71	21	71	21	71	21	71	21	71	21	71	21	71
	22	72	22	72	22	72	22	72	22	72	22	72	22	72	22	72	22	72	22	72
	23	73	23	73	23	73	23	73	23	73	23	73	23	73	23	73	23	73	23	73
	24	74	24	74	24	74	24	74	24	74	24	74	24	74	24	74	24	74	24	74
	25	75	25	75	25	75	25	75	25	75	25	75	25	75	25	75	25	75	25	75
	26	76	26	76	26	76	26	76	26	76	26	76	26	76	26	76	26	76	26	76
PUNCHING	27	77	27	77	27	77	27	77	27	77	27	77	27	77	27	77	27	77	27	77
	28	78	28	78	28	78	28	78	28	78	28	78	28	78	28	78	28	78	28	78
	29	79	29	79	29	79	29	79	29	79	29	79	29	79	29	79	29	79	29	79
	30	80	30	80	30	80	30	80	30	80	30	80	30	80	30	80	30	80	30	80
	31	81	31	81	31	81	31	81	31	81	31	81	31	81	31	81	31	81	31	81
	32	82	32	82	32	82	32	82	32	82	32	82	32	82	32	82	32	82	32	82
	33	83	33	83	33	83	33	83	33	83	33	83	33	83	33	83	33	83	33	83
	34	84	34	84	34	84	34	84	34	84	34	84	34	84	34	84	34	84	34	84
	35	85	35	85	35	85	35	85	35	85	35	85	35	85	35	85	35	85	35	85
	36	86	36	86	36	86	36	86	36	86	36	86	36	86	36	86	36	86	36	86
	37	87	37	87	37	87	37	87	37	87	37	87	37	87	37	87	37	87	37	87
	38	88	38	88	38	88	38	88	38	88	38	88	38	88	38	88	38	88	38	88
	39	89	39	89	39	89	39	89	39	89	39	89	39	89	39	89	39	89	39	89
	40	90	40	90	40	90	40	90	40	90	40	90	40	90	40	90	40	90	40	90
	41	91	41	91	41	91	41	91	41	91	41	91	41	91	41	91	41	91	41	91
	42	92	42	92	42	92	42	92	42	92	42	92	42	92	42	92	42	92	42	92
	43	93	43	93	43	93	43	93	43	93	43	93	43	93	43	93	43	93	43	93
	44	94	44	94	44	94	44	94	44	94	44	94	44	94	44	94	44	94	44	94
	45	95	45	95	45	95	45	95	45	95	45	95	45	95	45	95	45	95	45	95
	46	96	46	96	46	96	46	96	46	96	46	96	46	96	46	96	46	96	46	96
	47	97	47	97	47	97	47	97	47	97	47	97	47	97	47	97	47	97	47	97
NOT USED FOR TABLES	48	98	48	98	48	98	49	98	48	98	48	98	48	98	48	98	48	98	48	98
	49	99	49	99	49	99	49	99	49	99	49	99	49	99	49	99	49	99	49	99

Figure 4. Magnetic Drum Layout Chart

Location of Factors

It is necessary to have a means of locating each specific word in storage at any time. To make this possible, each segment of the drum and arithmetical units that can contain a ten-digit word is assigned a four-digit location address. Thus, each word has a unique address that is used to locate that word only.

Factors are recorded on the drum in a pattern so that, beginning at a specific point on the drum and going around the surface, one would encounter fifty words before returning to the starting point. Each such strip of fifty word locations comprises what is known as a *band* of general storage. The Type 650 has twenty such bands of storage in the 1000-word model, and forty in the 2000-word model.

Any word location on the drum may be located by selecting the proper band and then the proper word within that band. The drum layout chart (Figure 4) shows how 1000 words of general storage are arranged on the drum surface. The general storage addresses that are used within the Type 650 are 0000-1999 (0000-0999 in the 1000-word machine).

The condenser storage units in the arithmetical circuitry are also addressable. These units and their addresses are:

8001	Distributor
8002	Lower half of the 20-digit accumulator
8003	Upper half of the 20-digit accumulator

The control console has a group of eleven switches (ten for digits and one for a sign) which are addressable from the machine at 8000. The distributor, accumulator, and console switches will each be discussed in later sections of the manual.

Input-Output Buffers

To obtain high-speed operation, input and output factors pass through an intermediate storage area. This intermediate storage is known as *buffer* storage. Both the read and punch buffer storage areas can accommodate 100 digits (10 words). The buffer storage areas are located on a portion of the drum that is not addressable (Figure 5).

The first action of a card-reading operation is to transfer immediately the contents of the read buffer storage to the general storage section of the drum. Then, as the next card is read by the Type 533, the information from that card enters read buffer storage where it is held until the next card read operation is called for.

The punch buffer is used in essentially the reverse of the read buffer. The first action of a punch operation is to transfer immediately the information to be punched from general storage to punch buffer storage.

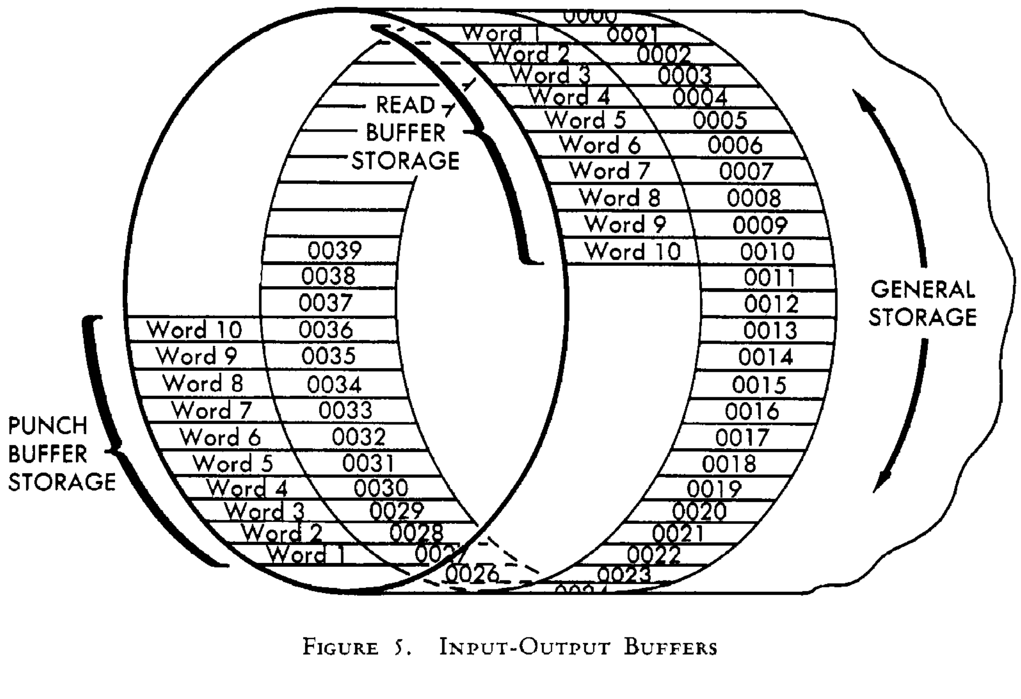


FIGURE 5. INPUT-OUTPUT BUFFERS

The information is then punched into the card from punch buffer storage. Because of this buffer storage arrangement, it is possible for the Type 650 to perform reading, punching, and calculating functions simultaneously.

From the drum layout chart (Figure 4), it may be seen that word locations 1-10 or 51-60 of a band are the only locations that may be used for read input storage. A similar condition exists with respect to punch output storage (locations 27-36 or 77-86 of a band). The reason for the restriction to these locations is their association with the read and punch buffer storage blocks.

INSTRUCTIONS

ALL OPERATIONS performed by the Type 650 are controlled by numerically coded instructions. An instruction consists of ten decimal digits and sign. The ten digits of the instruction are divided into three basic groups (Figure 6):

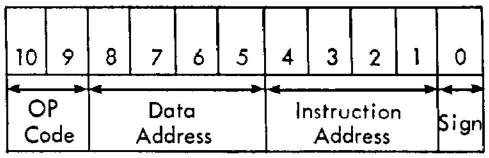


FIGURE 6. TYPE 650 INSTRUCTION GROUPS

1. The operation code, digit positions 10 and 9, designates the operation that the machine is to perform.

2. Digit positions 8-5 designate the data address. Depending upon the operation code of the instruction, the data address can have one of the following meanings:

 a. Location of information to be used in the operation

 b. Location in which information is to be stored by the operation

 c. The number of positions the information in the accumulator should be shifted left or right

 d. The maximum number of positions the information in the accumulator may be shifted on a shift and count operation

 e. Locations into which information is to be read from a card

 f. Locations from which information is to be punched into a card

 g. The location of an alternate instruction

 h. The location of the beginning of a table

3. Positions 4-1 designate the instruction address, which is normally the address of the next instruction to be performed.

The sign associated with an instruction is not considered in the analysis and execution of the instruction but should be present.

Original data and instructions are stored in drum storage locations from punched cards during the machine loading operation. Additional data and instructions may be loaded from cards during the solution of the problem. Because both data and instructions are stored in the same manner, an instruction may be subjected to arithmetical operations and thus altered from its original form. If an instruction is altered in this manner, the sign of the instruction is treated in exactly the same way as the sign of any other number.

As a problem is being solved, the machine refers to a storage location to obtain an instruction. After one operation is performed, the machine obtains another instruction from storage. A meaningful sequence of these instructions is called a *program*. Because the program is stored on the magnetic drum, the Type 650 is a stored-program machine.

Internal Flow (Figure 7)

From the schematic of the internal flow, it can be seen that:

1. The program register can receive instructions directly from general storage.

2. The program register can transmit information directly to the operation and address registers.

3. The distributor can receive information directly from general storage or the accumulator.

4. The distributor can transmit information directly to general storage, the program register, or the accumulator.

5. The accumulator receives information directly from the distributor and indirectly from general storage by way of the distributor.

6. The accumulator transmits information directly to the distributor or program register and indirectly to general storage by way of the distributor.

Instruction Execution

The three-part make-up of the Type 650 instruction provides that, whenever possible, while one operation is being executed, the following instruction is being located.

The internal control section governs the execution

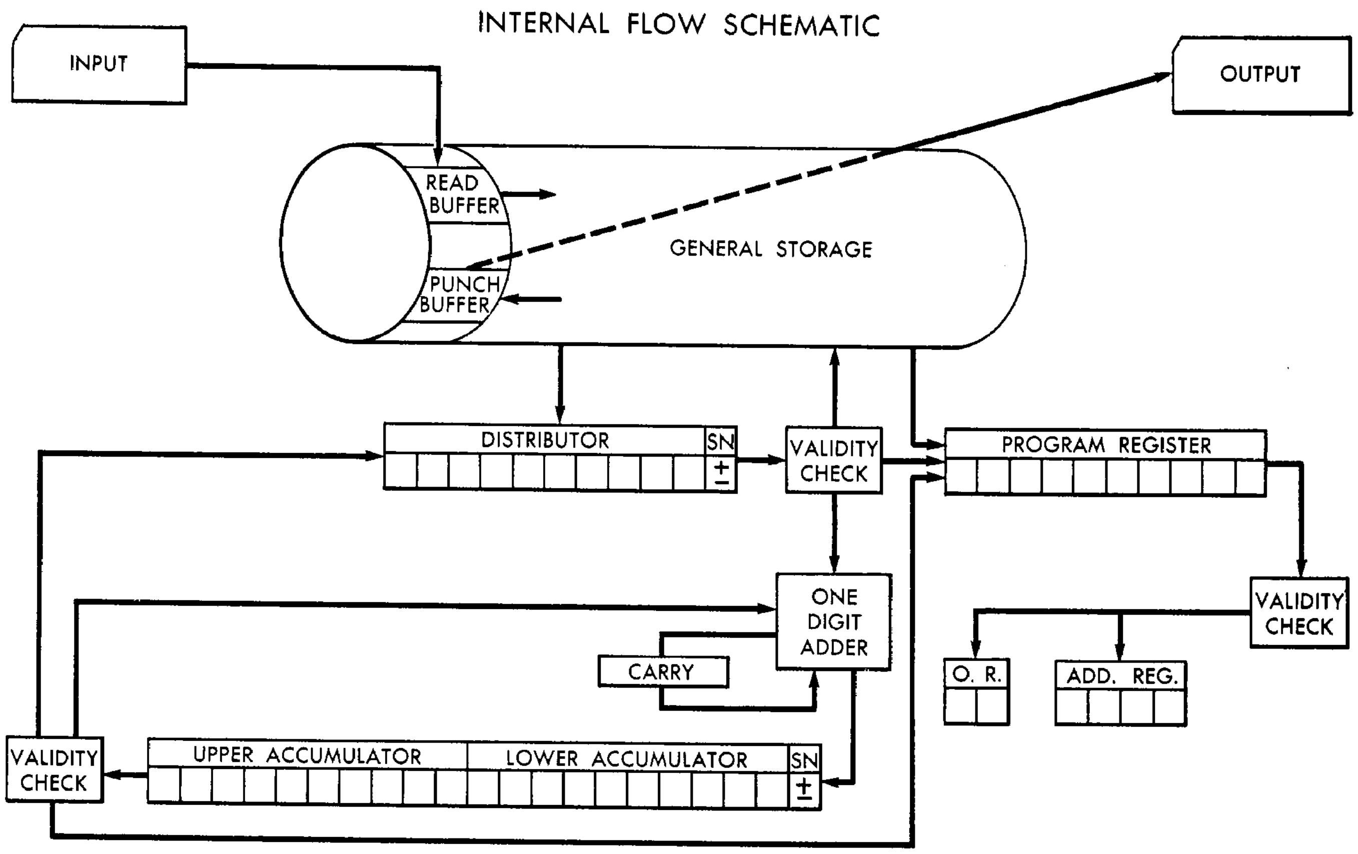


FIGURE 7. INTERNAL FLOW SCHEMATIC

of the instruction. It receives the instruction, interprets the operation code, and sets up the necessary circuitry to accomplish the specified operation. All arithmetical and logical operations of which the machine is capable (addition, subtraction, multiplication, division, transfer of words to and from storage, shifting and logical tests) are performed by the action of the distributor and the accumulator. The functions of these units are controlled by the program, operation, and address registers (Figure 8).

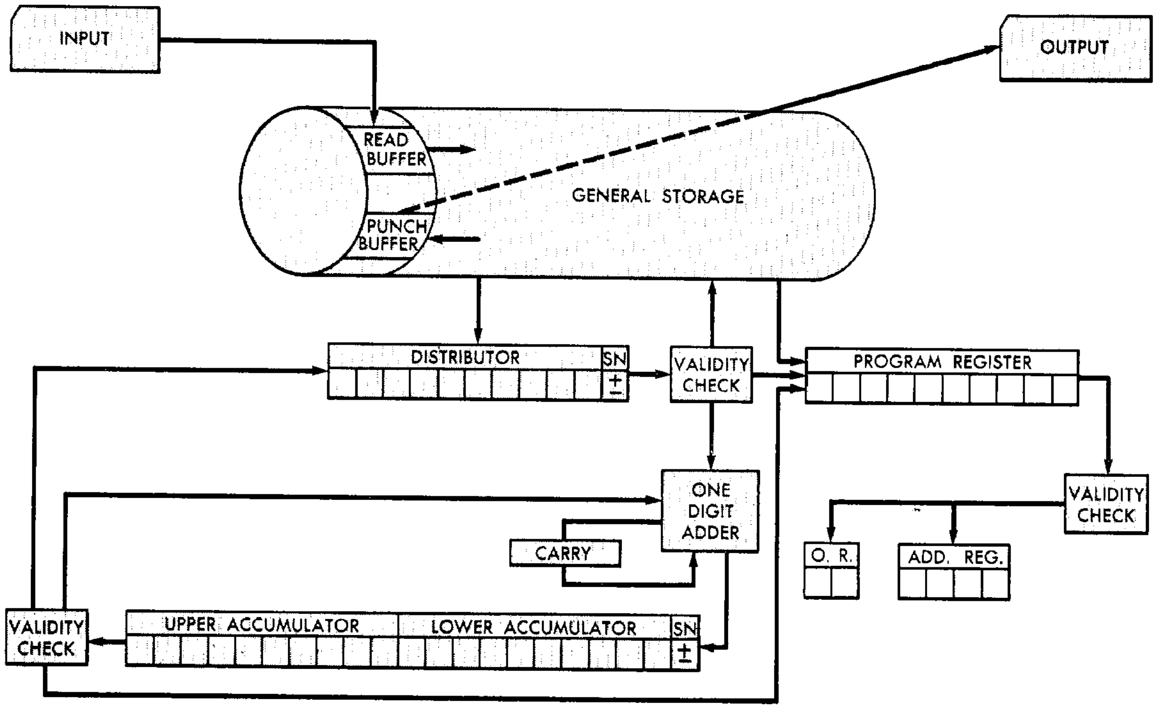
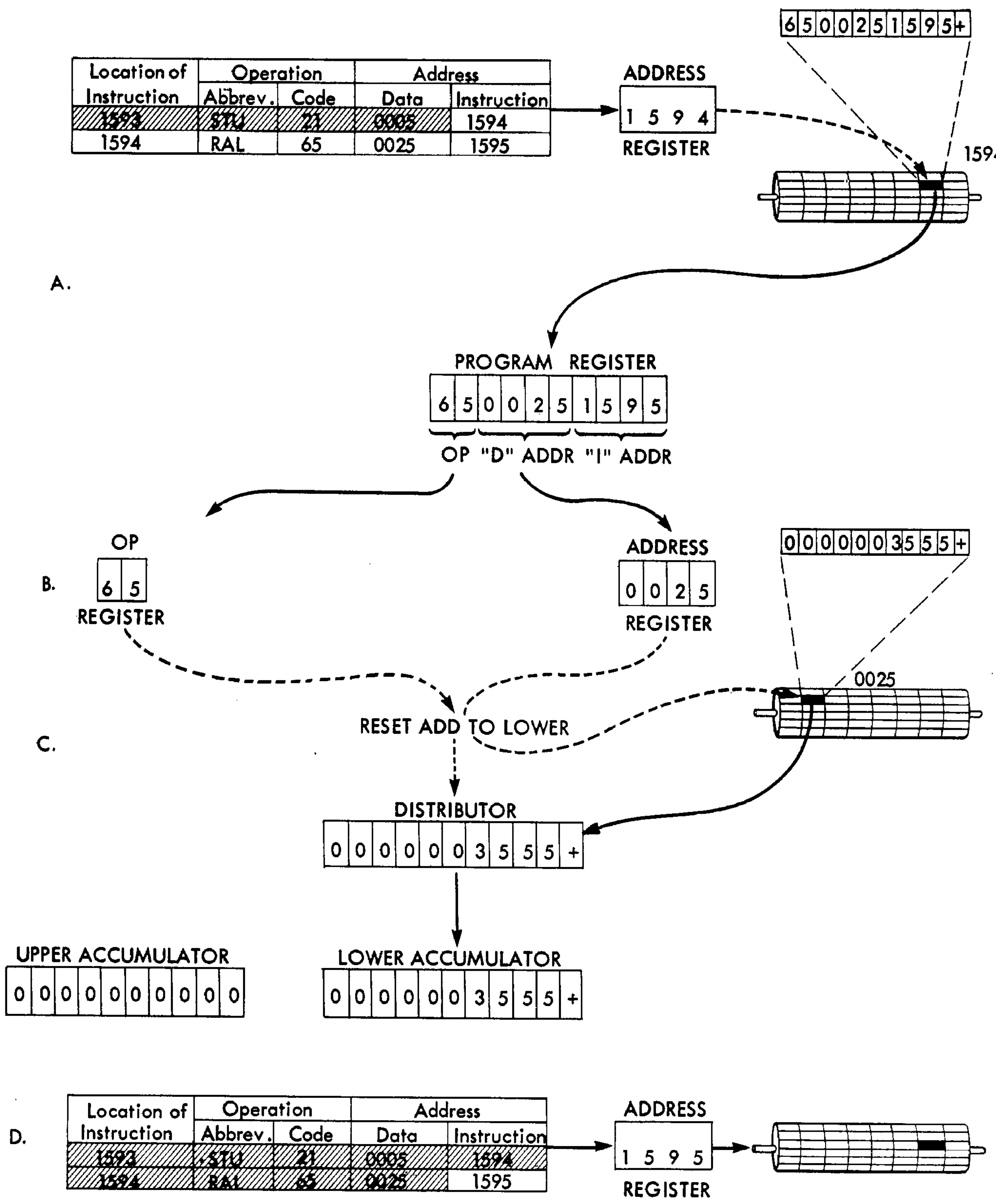


FIGURE 8. INTERNAL FLOW SCHEMATIC



Location of Instruction	Operation		Address	
	Abbrev.	Code	Data	Instruction
1593	STU	21	0005	1594
1594	RAL	65	0025	1595

Location of Instruction	Operation		Address	
	Abbrev.	Code	Data	Instruction
1593	STU	21	0005	1594
1594	RAL	65	0025	1595

Figure 9. Instruction Execution

The program register receives all ten digits of an instruction (Figure 9A). The program register transfers the operation code to the operation register and the data address to the address register for the first half-cycle of the execution of an instruction. This is known as the *data* (D) *half-cycle* (Figure 9B).

The operation code is analyzed in the operation register. The data address is analyzed in the address register. If the operation is arithmetical, the contents of the data address location are read from the D-address location into the distributor (Figure 9C). If the operation is logical, the D-address can specify the number of positions to be shifted, location of a table, branch address, etc., in which cases the previous contents of the distributor will not be altered by the operation. All store accumulator operations, however, will alter the contents of the distributor.

Execution of the instruction is begun once the operation code is interpreted and the distributor loaded (if necessary). The second part or *instruction* (I) *half-cycle* begins as soon as possible. The operation and address registers are reset and the *I* part of the instruction word (positions 4-1) in the program register is transferred to the address register for analysis and selection of the next instruction (Figure 9D).

The next instruction is then read into the program register (Figure 9A). The operation and address registers are reset, and the OP-code and D-address portions of the new instruction word are read out of the program register and into the operation and address registers (Figure 9B). An interlock will prevent any further program advance, except for read or punch operations, until the previous step has finished using the arithmetical units.

In this manner, the machine steps through a program, locating the next instruction, when possible, while executing the current one.

ARITHMETICAL COMPONENTS

ALL OPERATIONS other than read, punch, no-op, and programmed stop that can be performed by the Type 650 make use of one or more of the arithmetical units. There are three basic arithmetical units: the distributor, the accumulator, and the one-digit adder (Figure 10). These units are composed of electronic circuitry and are not a part of drum memory. The distributor and accumulator are controlled by the stored program. The one-digit adder is fully controlled internally, and its operation cannot be programmed.

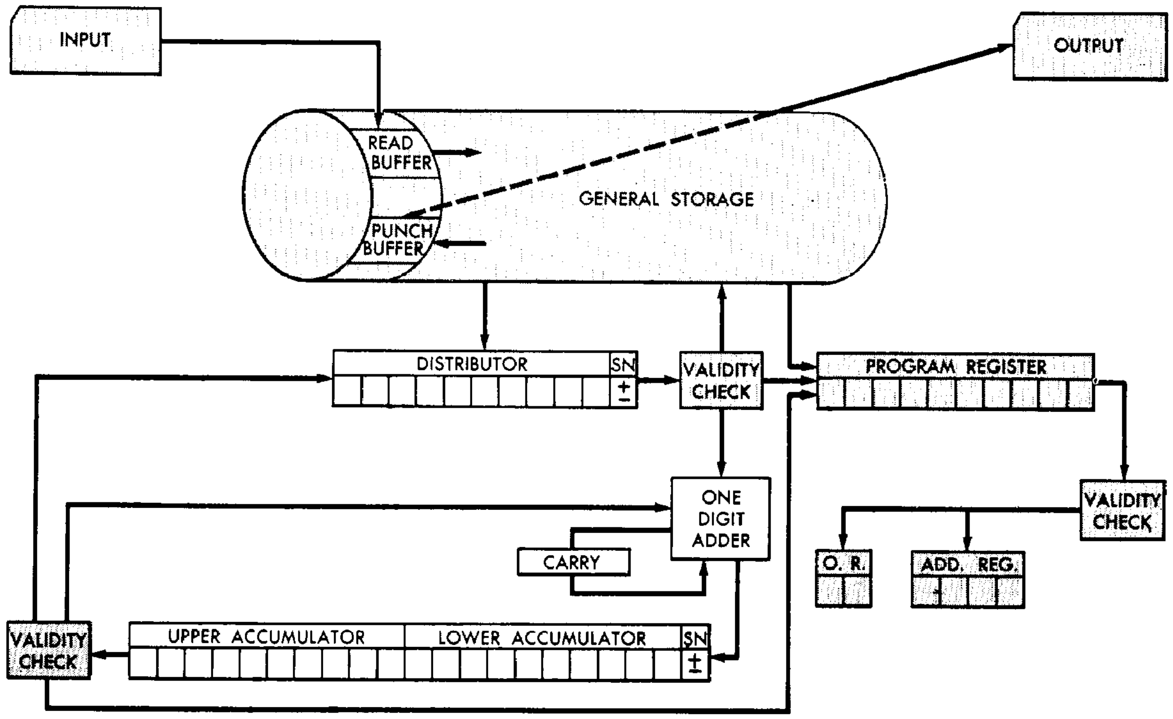


FIGURE 10. ARITHMETICAL COMPONENTS

Distributor

The distributor has a capacity of one word (10 digits and sign) and is addressable at 8001. It acts as an intermediate link between the accumulator and drum memory. Any information transferred between the accumulator and memory, in either direction, must pass through the distributor. Information passed through the distributor remains in the distributor and is available for repeated use until replaced by the entry of new information. The distributor has an extremely low access time and, therefore, provides a word source that can be used to increase the operational speed of the Type 650. The distributor also makes possible increased flexibility in the use of the accumulator. This increased flexibility will be explained in the accumulator discussion that follows.

Accumulator

The accumulator is a storage unit having a capacity of twenty digits and sign. Actual accumulation is accomplished in a one-digit adder, but the results from the adder are stored in the accumulator. Tne twenty positions of the accumulator are divided into two sections, upper and lower. The low-order ten positions comprise the lower half and have an address of 8002. The high-order ten positions comprise the upper half and have an address of 8003.

The accumulator is flexible in control because it is possible to read into and read out of either half. However, the entire accumulator is reset whenever an instruction for either half calls for a reset. It should be noted that there is only one sign for the entire accumulator (except on a division operation where the remainder and quotient have independent signs). In addition and subtraction operations, carries are propagated from the lower to the upper half of the accumulator. Care must be taken to insure that no operations are attempted in one half of the accumulator that would have an undesirable effect on the other half. Such operations would be of the following nature:

1. With an independent factor in each half of the accumulator and the sign of the accumulator plus, a factor larger than the factor within the lower is subtracted from the lower. This will cause a reduction of the units position of the factor in the upper half of the accumulator. The resultant factor in the lower half will be a complement of its actual value and will have a positive sign.

		Accumulator	
Operation		Upper Half	Lower Half
Subtract 6500			
from lower	*before*	0000000123	0000005678+
	after	0000000122	9999999178+

2. With the same condition as in 1, if a factor larger than the factor in the upper half of the accumulator is subtracted from the upper, the factor in the lower half will become a complement and will carry the sign of the accumulator (minus) if stored at this time.

		Accumulator	
Operation		Upper Half	Lower Half
Subtract 500			
from upper	*before*	0000000123	0000005678+
	after	0000000376	9999994322−

In case the capacity of the accumulator is exceeded, an overflow will be sensed. This overflow may be used to stop the machine, or, through programming, may be detected to alter the course of the program.

Because of the distributor, it is possible to perform the following operations on the accumulator:

1. Either half of the accumulator can be added to or subtracted from the other half (Figure 11A).
2. Either half can be added to itself, i.e., multiply by 2 (Figure 11B).
3. Either half can be subtracted from itself (Figure 11C).
4. Either half can be added (or subtracted) into the other half with reset. This is equivalent to a ten-position shift (Figure 11D) .
5. Either half can be added, with reset, into the same half. This is used to clear one half of the accumulator while retaining the contents of the other half (Figure 11E).
6. A factor in the upper accumulator may be squared by a multiply instruction having a data address of 8003 (Figure 11F).

In any of the foregoing operations, the number being moved passes through the distributor and will remain there until replaced by the entry of another factor.

Instruction			8003 Upper Accumulator										8002 Lower Accumulator										8001 Distributor									
OP	Data	Instr.																														
			0	0	0	0	0	0	0	1	2	3	0	0	0	0	0	0	0	4	5	6	0	0	0	0	0	0	0	1	2	3
AU	8002	XXXX	0	0	0	0	0	0	0	5	7	9	0	0	0	0	0	0	0	4	5	6	0	0	0	0	0	0	0	4	5	6

A

Instruction			8003 Upper Accumulator										8002 Lower Accumulator										8001 Distributor									
OP	Data	Instr.																														
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3	4	0	0	0	0	0	0	1	4	6	8
AL	8002	XXXX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	6	8	0	0	0	0	0	0	0	2	3	4

B

Instruction			8003 Upper Accumulator										8002 Lower Accumulator										8001 Distributor									
OP	Data	Instr.																														
			0	0	0	0	0	0	2	3	7	8	0	0	0	0	0	0	1	2	6	9	0	0	0	0	0	0	1	2	6	9
SU	8003	XXXX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	6	9	0	0	0	0	0	0	2	3	7	8

C

Instruction			8003 Upper Accumulator										8002 Lower Accumulator										8001 Distributor									
OP	Data	Instr.																														
			0	0	0	0	1	2	3	4	5	6	0	0	0	2	4	3	2	1	6	0	0	0	0	0	0	0	0	0	0	0
RAL	8003	XXXX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	6	0	0	0	0	1	2	3	4	5	6

D

Instruction			8003 Upper Accumulator										8002 Lower Accumulator										8001 Distributor									
OP	Data	Instr.																														
			0	0	0	0	0	0	2	4	6	8	0	0	0	0	0	3	5	7	9	2	0	0	0	0	0	0	0	1	2	3
RAL	8002	XXXX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	5	7	9	2	0	0	0	0	0	3	5	7	9	2

E

Instruction			8003 Upper Accumulator										8002 Lower Accumulator										8001 Distributor									
OP	Data	Instr.																														
			0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Mult	8003	XXXX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	6	0	0	0	0	0	0	0	0	0	4

F

FIGURE 11

One-Digit Adder

As mentioned in the accumulator description, actual addition or subtraction is accomplished by passing the two factors involved through the one-digit adder (Figure 10). In any operation involving addition, subtraction, multiplication, or division, the factor in the location specified by the data address is automatically placed in the distributor. Then the factor in the distributor and the factor in the accumulator are fed into the adder one digit at a time starting with the low-order digit.

The adder has three inputs: one for the digit from the distributor, one for the digit from the accumulator, and a third for carry input in case the sum of the preceding digits was ten or more. Thus, when two factors are added, the digit from the distributor is added to the digit of equal order from the desired half of the accumulator and, in case a carry occurs, the circuit is set to increase the sum of the next set of digits by one. The signs of both factors are analyzed along with the operation and, if necessary, the individual digits of one or the other of the factors are complemented before they enter the adder. Therefore, the adder accomplishes all arithmetical operations by addition.

BI-QUINARY CODE

THE TYPE 650 translates the decimal-digit representation of the input data into bi-quinary digit representation for internal transmission and use.

The bi-quinary code makes use of seven possible units of information. Two of these units (or bits) are known as the binary portion of the code and have the assigned values of 0 and 5 (Figure 12). The remaining five units are known as the quinary portion of the code and have the assigned values of 0, 1, 2, 3, and 4 (Figure 12).

When this code is used, all digits 0-9 are represented by the combination of one binary unit and one quinary unit of information (Figure 13). Because of this condition, it is possible to check for valid information. This checking is known as *validity checking* and will be discussed under that topic.

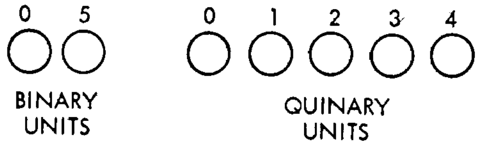


FIGURE 12. BI-QUINARY REPRESENTATION

DIGIT	BINARY	QUINARY
0	0	0
1	0	1
2	0	2
3	0	3
4	0	4
5	5	0
6	5	1
7	5	2
8	5	3
9	5	4

FIGURE 13. BI-QUINARY CODE

Signs of numbers are translated internally into a digit 9 for plus and a digit 8 for minus to enable checking of their validity.

SELF-CHECKING

THE TYPE 650 has been designed with reliability as a paramount consideration. Conservative circuit design and components chosen for their high reliability are employed throughout. The Type 650 also uses built-in self-checking as a supplement, which provides further assurance of accurate results.

Self-checking within the machine is separated into two categories: validity checks and control checks.

Validity Checks

The use of the bi-quinary code within the Type 650 provides a unique method of checking the internal transmission of digits. Checking is desirable to insure that no digits have been affected by failure of any of the transmitting elements.

Because each digit (0-9) is represented by one and only one binary bit and one and only one quinary bit, any transmission error can be detected by checking for any lost or gained bits in each digit position. Thus, validity checking errors are indicated for the following conditions:

1. No binary bit
2. Two binary bits
3. No quinary bit
4. More than one quinary bit
5. No binary and no quinary bits (blank)

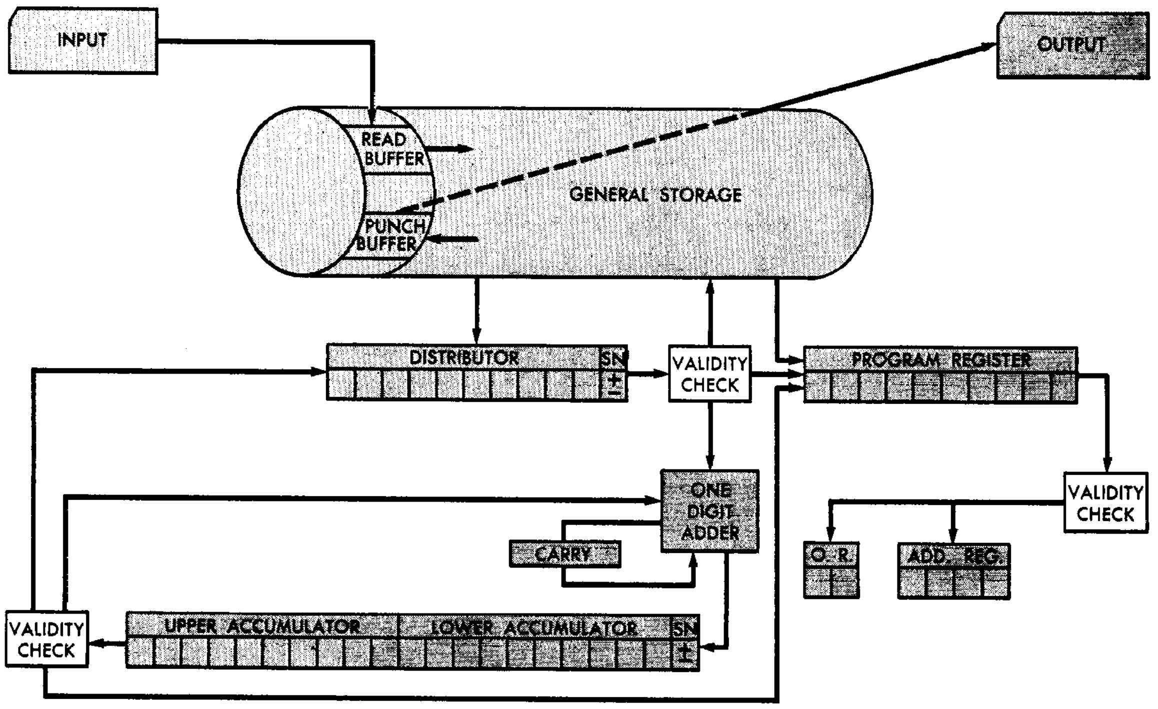


FIGURE 14. VALIDITY CHECK POINTS

It is conceivable that compensating errors might cause the loss of one bit and the simultaneous gain of a different bit. However, even this condition would have to create a valid number. The probability of such a condition ever occurring is extremely remote.

The following units have a validity check point located in their output (Figure 14):

1. Distributor
2. Accumulator
3. Program register

Figure 14 illustrates the validity check points with relation to the flow of information within the Type 650.

Read Checks

Information read into the Type 650 from a punched card undergoes a validity check as soon as it is referred to by the program. This check detects any failures that may have occurred during card reading. Some of these reading failures that are detected and their results are:

1. Failure to read a column for any reason will cause a blank digit position.

2. Shorted reading brushes will cause extra bits to be added to the associated digit positions.

3. Early, late, or crooked card feeding can cause blank digit positions by failing to read a punched hole or extra bits in a digit position by reading punched holes in adjacent columns.

One very important fact should be noted at this time. In order to receive the full benefit of validity checking, 12's must be used for positive signs. If positive signs are not used, the machine assumes that the absence of an 11 punch means a positive sign. It is obvious that a failure to read an existing 11 punch will cause the machine to assume that the factor is positive.

Punch Checks

Output checking is accomplished by the double-punch and blank-column detection device. Any bits lost between the last validity check and the punch operation will cause a blank column. Any bits gained between the last validity check and the punch operation will cause multiple punching of a column.

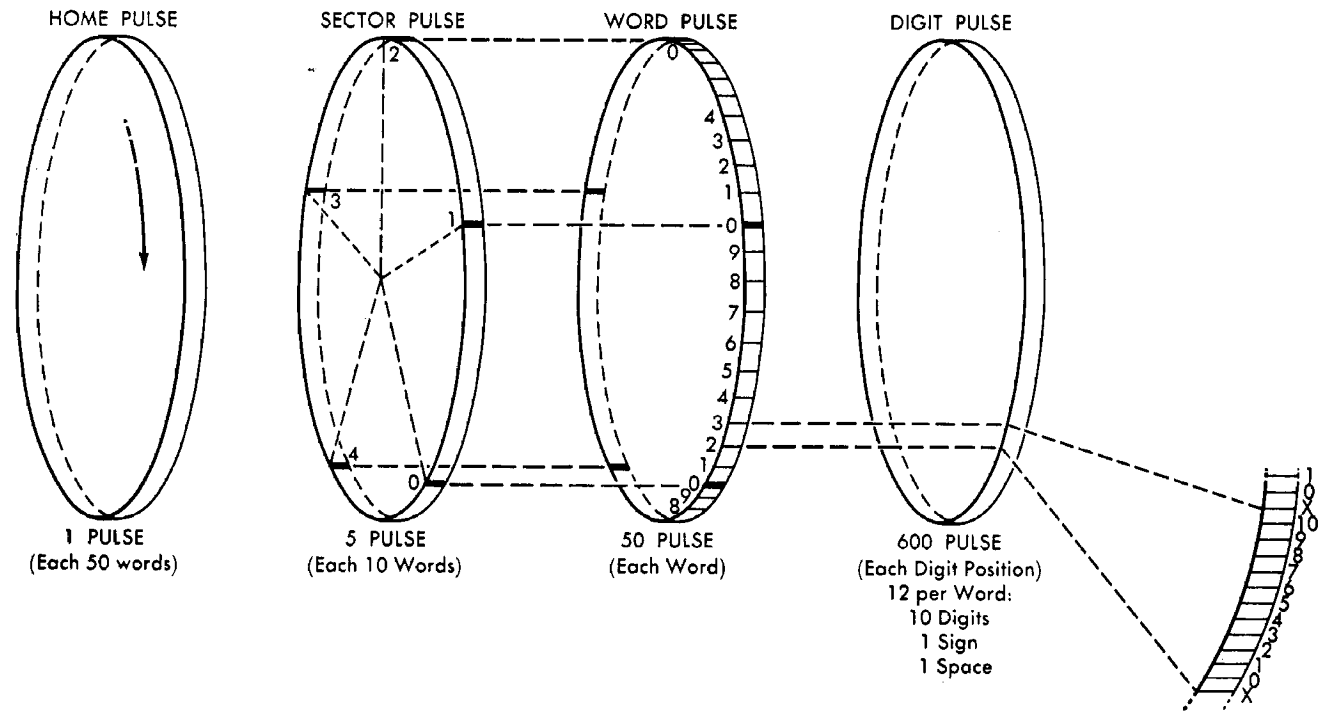


FIGURE 15. TIMING PULSES

Control Checks

Control checks have been incorporated in the machine to detect the following conditions:

1. Addresses other than 0000-1999, 8000, 8001, 8002, and 8003.
2. Operation codes other than those assigned to machine functions.
3. Timing circuitry discrepancies.
4. Accumulator overflows not anticipated in problem processing.

The timing referred to in item 3 is controlled by four tracks of pulses located on the drum (Figure 15). These tracks supply pulses timed to the following:

1. One complete drum revolution
2. One-fifth of a drum revolution (10 words)
3. One word-time
4. One digit-time

Error Correction

When the machine detects an error by means of a validity check or a timing circuit check, it is very possible that this error is a random one that will not occur again, or, at worst, occurs very infrequently. Hence, it is reasonable to assume that the machine will probably perform the problem correctly if given a second chance. Therefore, it will prove profitable in most applications to program the machine to repeat a problem in the case of errors detected by self-checking.

In practice, very large problems will be broken into smaller segments, each segment representing a small part of the whole. In this way, it will not be necessary for the machine to repeat the whole problem, but to repeat only that segment in which the error occurred.

The preceding statements should not be construed to mean that normal accounting and procedural controls might be eliminated or ignored. Such controls should be embodied, if at all possible, in the definition of the problem and should be regarded as part of the general procedure of efficient programming. The method of checking a particular problem is determined by the character of the problem and the degree of confidence in the results the programmer desires.

TYPE 533 INPUT-OUTPUT UNIT

THE INPUT-OUTPUT of the Type 650 Magnetic Drum Data-Processing Machine is by means of IBM punched cards, which are read and punched by the Type 533 (Figure 16). There are two independent card feeds: one for reading cards; the other, for punching cards (Figure 17).

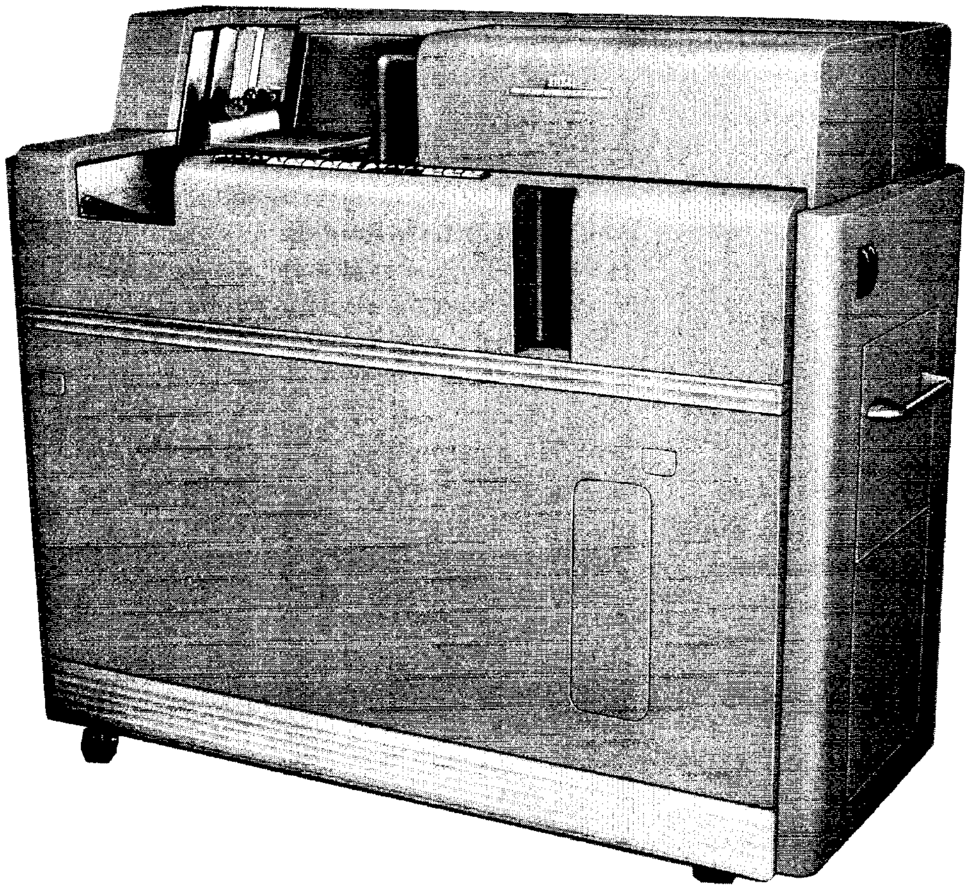


FIGURE 16. TYPE 533 INPUT-OUTPUT UNIT

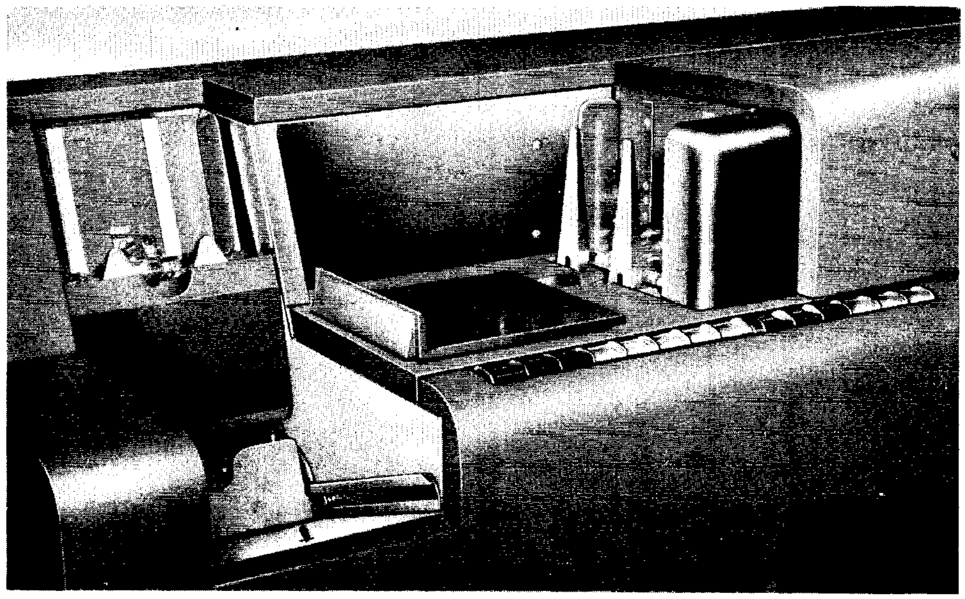


FIGURE 17. READ FEED (LEFT) AND PUNCH FEED (RIGHT)

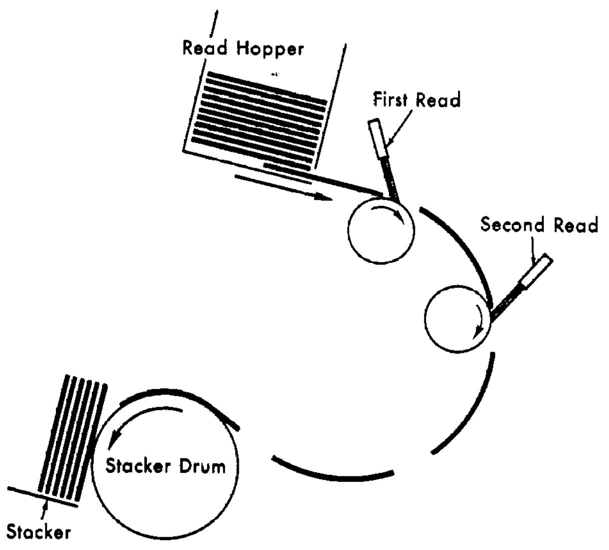


Figure 18. Read Feed Schematic

The speed of both the read and punch feeds can be affected by the amount of time required to perform the instructions between successive read or punch operations. To obtain the highest possible speed, both feeds are driven by multiple-tooth clutches.

Input

The input cards are fed into the card reader, face down, 12 edge first. The card reader (Figure 18) is similar to that on the Type 402 Accounting Machine and operates at a maximum speed of 200 cards per minute. All 80 columns of the card may be read into the Type 650. The card reader has two reading stations at which cards may be read. Normally, the cards are read at the second reading station. The first reading station is usually used to read control information, such as man number, card identification, etc. Group control is performed by internal programming.

Output

The punch feed contains a punch station and a punch brush station (Figure 19). The principle use of the punch brushes is for the detection of double-punched or blank columns in the output. Normal gangpunching from the punch brushes can be accomplished. It is not possible to read into drum storage from the punch brushes.

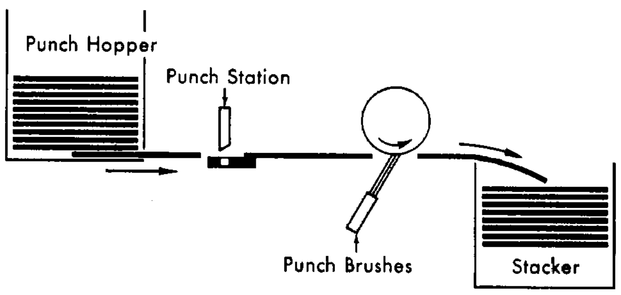


Figure 19. Punch Feed Schematic

The maximum rate of feeding in the punch feed is 100 cards per minute. As in the read feed, cards are fed face down, 12 edge first. The two feeds are entirely independent, and it is not possible to punch information directly from the read feed. Any information from a read card must pass from the reading brushes to the drum before it can be punched.

The control panel receptacle (Figure 20) is mounted in the Type 533. The control panel provides for flexibility of input and output. The operation of the Type 533 input-output unit is under the control of the keys and indicator lights shown in Figure 21.

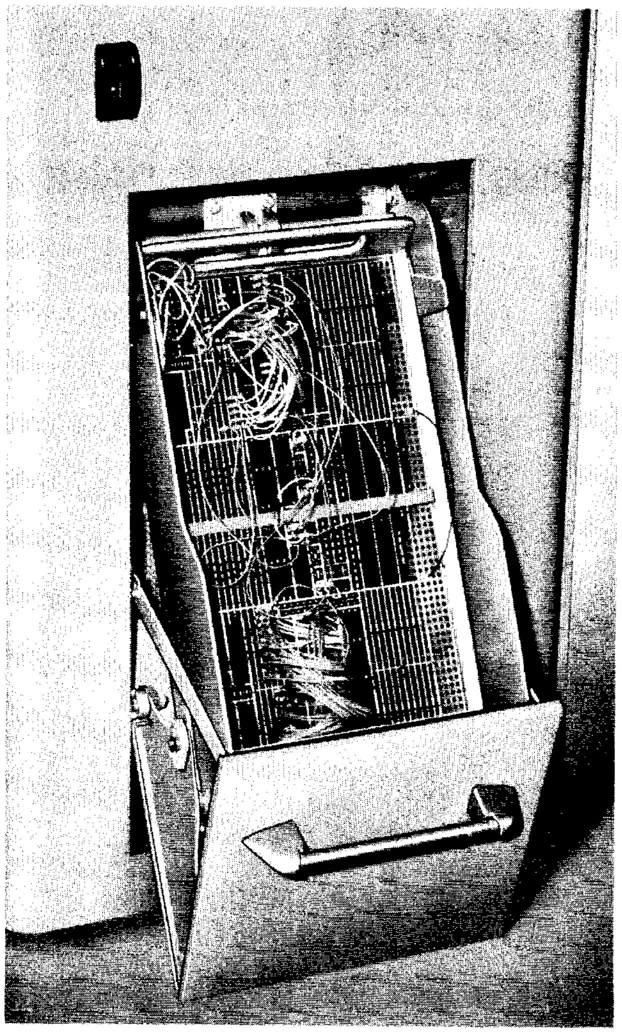


Figure 20. Control Panel

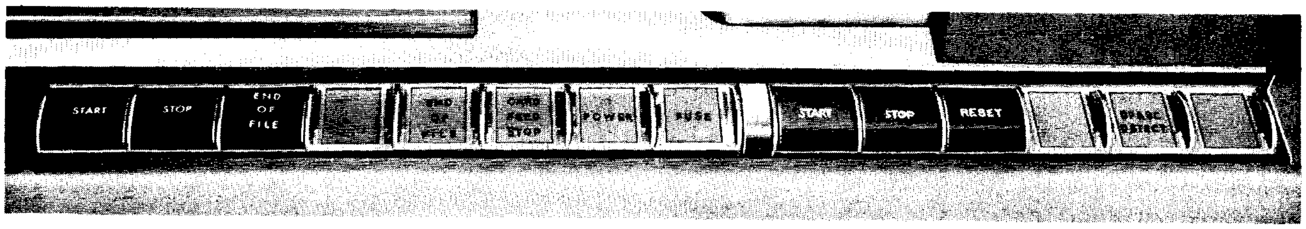


FIGURE 21. TYPE 533 CONTROL KEYS AND INDICATOR LIGHTS

Main-Line Switch

The main-line switch controls the application of power to the Type 533. This switch is located on the right-hand end of the unit.

Power Light

This light will go on when the Type 533 main-line switch is turned on and remains on as long as power is applied.

Fuse Light

This light goes on when a fuse blows in the Type 533.

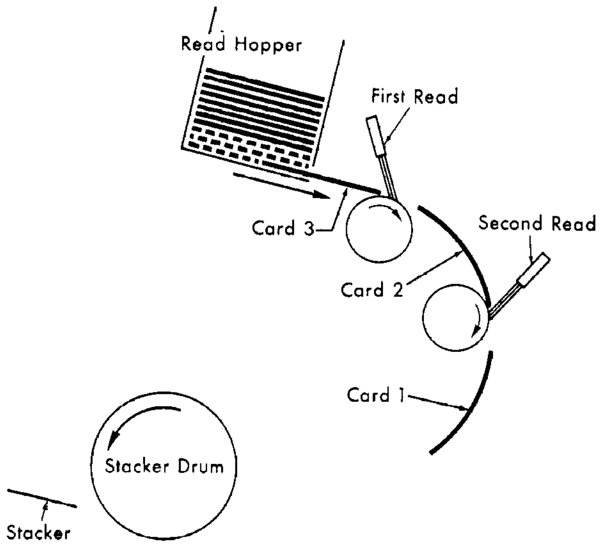


FIGURE 22. READ-START KEY—CONDITION 1

READ-UNIT KEYS AND LIGHTS

Read-Start Key

Depressing the read-start key causes the read feed to operate. The effect of the read-start key is dependent upon the number of cards in the read-feed hopper.

Condition 1. Four or More Cards (Figure 22)

With four or more cards in the read hopper, a single depression of the read-start key will cause three cards to be fed. After the initial run-in, feeding is controlled by the stored program as long as cards remain in the read hopper. With cards in the read hopper and cards at all stations of the read feed (three cards in), depression of the read-start key has no effect.

Condition 2. Three Cards (Figure 23)

With three cards in the read hopper, a single momentary depression of the read-start key will cause all three cards to be fed. Any additional depressions of the read-start key will cause the read feed to run.

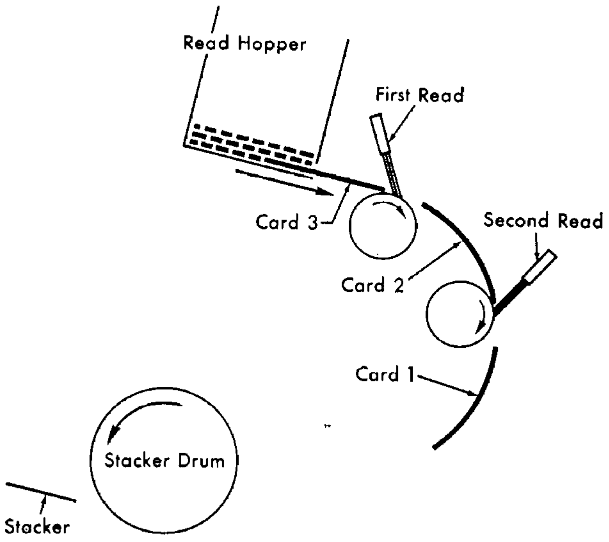


FIGURE 23. READ-START KEY—CONDITION 2

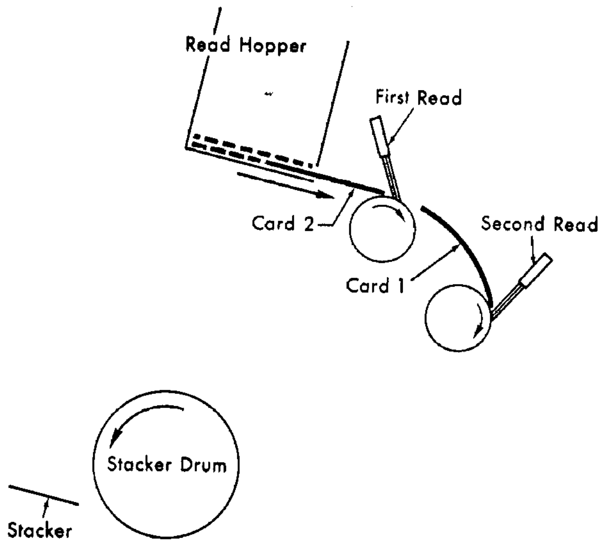


FIGURE 24. READ-START KEY—CONDITION 3

Condition 3. Two Cards (Figure 24)

With two cards in the read hopper, a single momentary depression of the read-start key will cause both cards to be fed. As in condition 2, any additional depressions of the read-start key will cause the read feed to run.

Condition 4. One Card (Figure 25)

With a single card in the read hopper, a single momentary depression of the read-start key will cause only one read-feed cycle, as in conditions 2 and 3 any additional depression of the read-start key will cause the read feed to run.

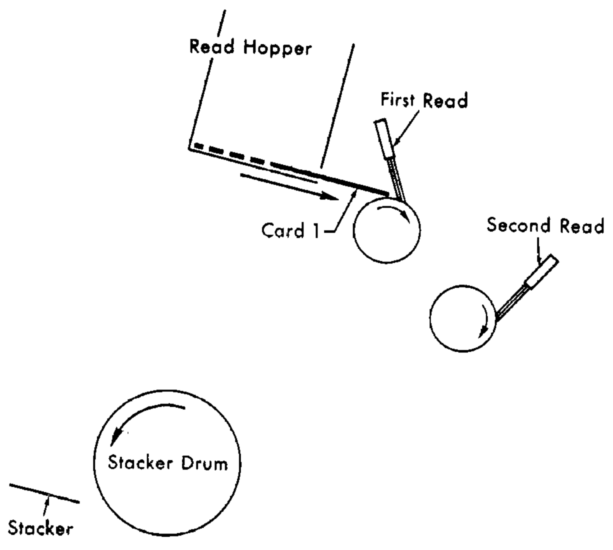


FIGURE 25. READ-START KEY—CONDITION 4

The read-start key also functions as a non-calculate runout key. This fact becomes very important in all conditions just mentioned. Because of the non-calculate runout condition, the last three cards in condition 1 and all cards in conditions 2, 3, and 4 could be passed through the machine without being processed. This would occur if the read-start key were depressed for a sufficient amount of time to cause all the cards in the feed to pass the second reading station.

The methods used to process cards remaining in the card feed are described under *End of File Key.*

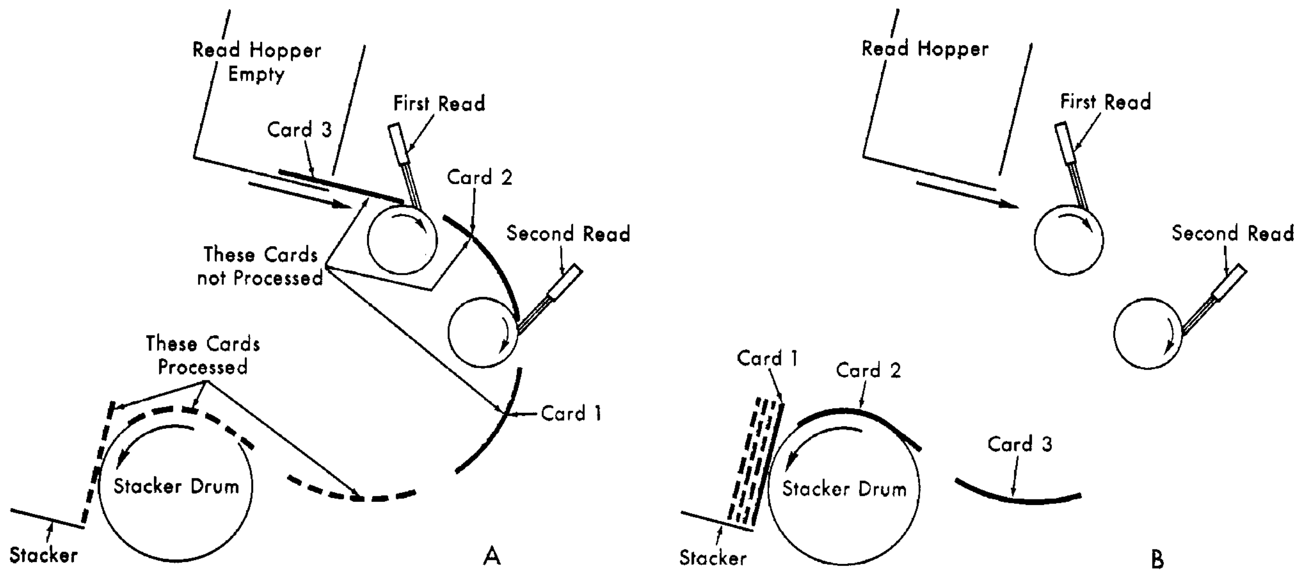


FIGURE 26. END-OF-FILE KEY CONDITION

End-of-File Key

The end-of-file key is effective only when the read hopper is empty, and a card has just passed the second reading station. In order to calculate or process the card that has just passed second reading and any cards that might follow it (maximum of two), the end-of-file key must be depressed. In other words there are three conditions under which the end-of-file key can be used:

1. The hopper becomes empty after feeding three or more cards. The unprocessed cards in the feed are at stations 1, 2, and 3 (Figure 26A).

Depressing the end-of-file key will allow all three cards to be processed (Figure 26B).

2. With only two cards in the read hopper, a single momentary depression of the read-start key will cause the read feed to take two card-feed cycles (Figure 27A).

A second momentary depression of the read-start key will cause the first card to pass the second reading station (Figure 27B).

Depressing the end-of-file key at this time will allow both cards in the feed to be processed (Figure 27C).

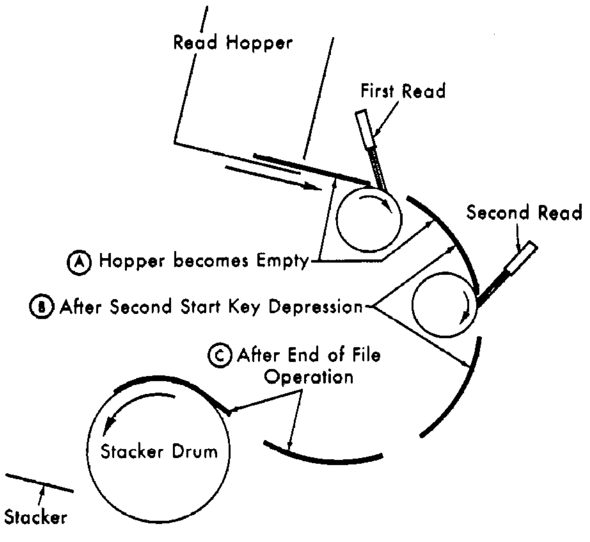


FIGURE 27

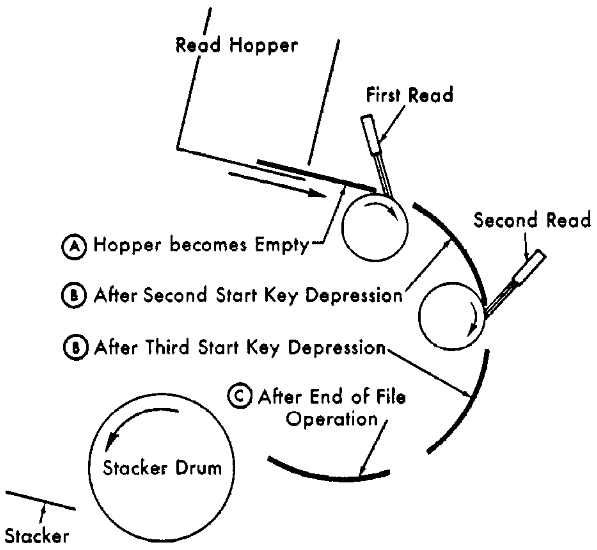

Read Hopper
First Read
Second Read
Ⓐ Hopper becomes Empty
Ⓑ After Second Start Key Depression
Ⓑ After Third Start Key Depression
Ⓒ After End of File Operation
Stacker Drum
Stacker

FIGURE 28

3. With a single card in the read hopper, a momentary depression of the read-start key will cause the read feed to take one card-feed cycle (Figure 28A).

Two additional momentary depressions of the read-start key will move the card past the second reading brushes (Figure 28B).

Depressing the end-of-file key at this time will allow the card to be processed (Figure 28C).

In all of the foregoing cases, the actual processing is under control of the stored program. The stored program must be initiated by the program start key on the control console. The program start key will be discussed in the control console section.

Stop Key

Depressing this key will immediately suspend card feeding in both feeds. However, the stored program continues until such time as a read or punch instruction is called for.

When both feeds are in use, card feeding may be stopped by depressing either the read-stop or punch-stop key. It is necessary to depress both the read-start and punch-start keys in order to resume card feeding.

When only one feed is in use, card feeding may be stopped by depressing either stop key; however, the appropriate start key must be depressed in order to resume feeding.

Unlabeled Light

This light is on whenever the read feed is not in use.

End-of-File Light

This light will go on when the end-of-file button is depressed and remains on until the information from the last card has entered drum storage.

Card-Feed Stop Light

This light will go on when an improper card feed occurs in the read feed. In order to restart, the cards must be removed from the hopper and the feed cleared. The light is turned off by depressing the stop key after the feed has been cleared.

When an improper card feed causes a card-feed-stop condition, the following steps are necessary in order to restart:

1. Remove the cards from the read hopper and depress the read-start key to clear the feed.
2. Depress the read-stop key to turn off the card-feed-stop light.
3. Remove the last two cards from the stacker and place them in front of the file, then restart in the normal manner.

A card-feed-stop on a run-in requires only that the read-stop key be depressed to turn off the card-feed-stop light.

PUNCH-UNIT KEYS AND LIGHTS

Start Key

Depressing this key causes the punch unit to feed cards (Figure 29). With cards in the punch hopper and cards positioned as shown by Card 1 and Card 2, the start key is inoperative. In order to run cards out of the punch unit, the hopper must be empty and the start key depressed. When processing a problem involving card punching, the punch-start key should be depressed and cards positioned as shown, before programming begins.

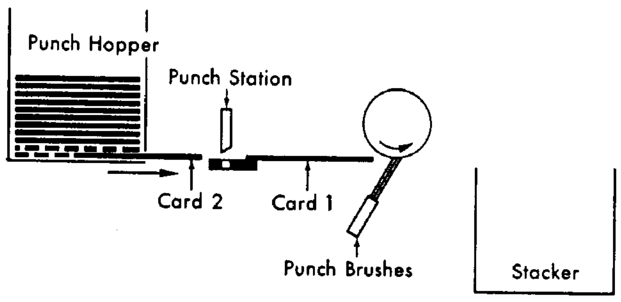


FIGURE 29. CARD POSITION AFTER RUN-IN

Stop Key

Depressing this key will immediately suspend card feeding in both feeds. However, the stored program continues until such time as a read or punch instruction is called for. When both feeds are in use, card feeding may be stopped by depressing either the read-stop or punch-stop key. It is necessary to depress both the read-start and punch-start keys in order to resume card feeding.

When only one feed is in use, card feeding may be stopped by depressing either stop key; however, the appropriate start key must be depressed in order to resume feeding.

If the punch-start key and punch-stop key are depressed at the same time, in an attempt to single-cycle the run-in of the cards in the punch feed, the machine will lock up after feeding one card. It will be necessary to remove the cards from the punch hopper and to run the card out of the punch feed before operation can be resumed.

Reset Key

Depressing this key resets the double-punch and blank-column stop circuit and turns off the double-punch and blank-column (DPBC) light.

Double-Punch and Blank-Column (DPBC) Light

This light comes on when a double-punch or blank-column error is detected. Detection of double-punched columns and blank columns is accomplished by control panel wiring.

Unlabeled Light (Adjacent to Reset Key)

This light is on whenever the punch feed is not in use.

Unlabeled Light (Extreme Right)

This light is not used on the standard machine.

OPERATION CODES

PROGRAMMING

THERE ARE 44 possible operation codes used in programming for the Type 650. In order to simplify the understanding of all operation codes, those operations that function in a similar manner are treated as a group. Thus, when one understands the basic function of one of the group, the others should be equally clear. The explanations are grouped as follows:

1. Read and punch
2. Transfer
3. Add and subtract without reset
4. Add and subtract with reset
5. Add and subtract absolute value
6. Multiply and divide
7. Shifting
8. Branching
9. Table lookup
10. Miscellaneous

One of the most simple problems for the Type 650 would be to read an input card, transfer some or all of the information to punching locations, and punch an output card. Reading and punching are presented first and then transfer of factors in general storage.

Input-Output

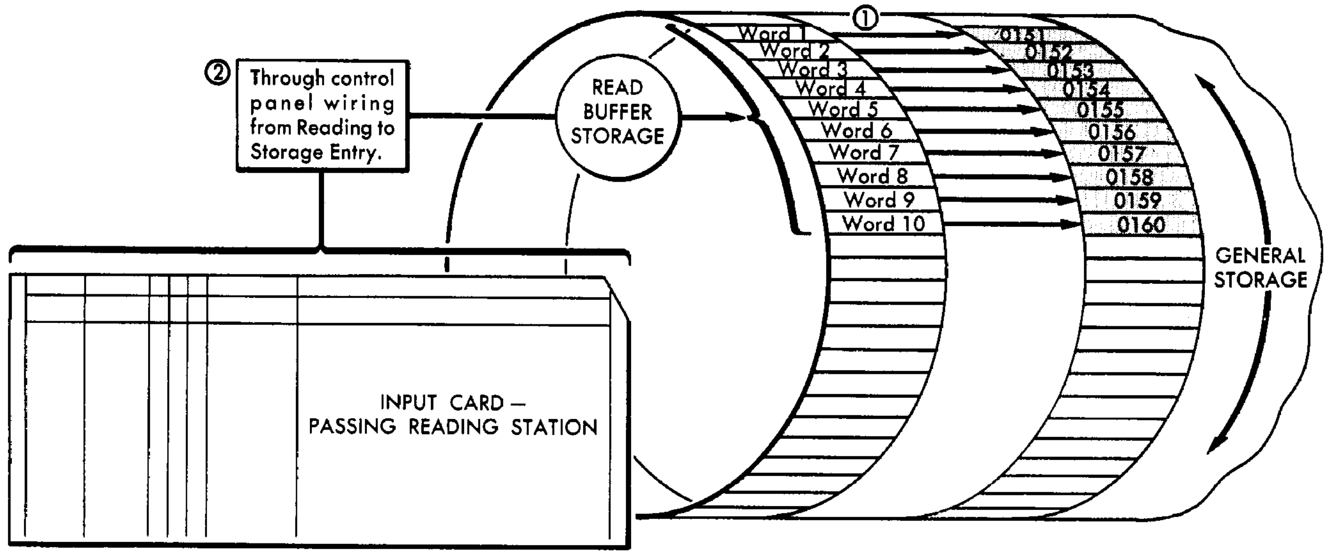
70 RD (Read)

This operation code causes the machine to read cards by a two-step process (Figure 30). First, the contents of the 10 words of read-buffer storage are automatically transferred to one of the 20 (or 40) possible ten-word groups of read general storage. The group selected is determined by the D-address of the READ instruction. Secondly, a card is moved under the reading brushes, and the information read is entered into buffer storage for the next READ instruction. When cards are initially run into the machine, the contents of the first card are read into read-buffer storage. Therefore, the first READ instruction causes the contents of the first card to be transferred to general storage, and the contents of the second card to be read into read-buffer storage.

Of the 300 milliseconds read cycle an average of 270 milliseconds is available for computing because of the read-buffer storage.

Each drum memory band has 10 words that may be used for read storage and any D-address given within that band will cause the card to read into those storage positions. For example, an address of 0663 would cause reading into locations 0651 through 0660. A READ instruction will erase the previous contents of the read storage locations. However, the read storage is available for any purpose during computing.

Special conditions apply when the card being read is identified as a load card. Load cards will be discussed following this section.



INSTRUCTION: 70 0151 0003

STEP ①— Contents of Read Buffer Storage are transferred to Read General Storage Locations 0151-0160 specified by the "D" address of the Read Instruction.

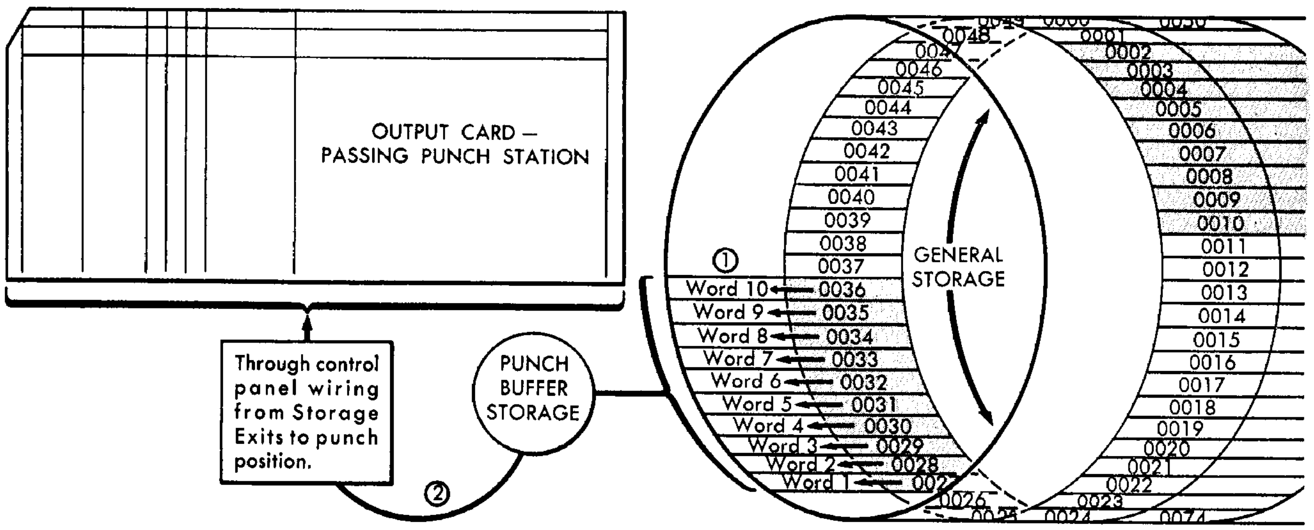
STEP ②— The next card is moved past the Reading Station; and information read is entered, through control panel wiring, into Read Buffer Storage. Control panel wiring allows full flexibility of entry of factors.

FIGURE 30

71 PCH (Punch)

This operation code causes card punching in two steps (Figure 31). First, the contents of one of the 20 (or 40) possible ten-word groups of punch storage are transferred to punch-buffer storage. The group selected is specified by the D-address of the PUNCH instruction. Secondly, the card is punched with the information from buffer storage. Of the 600 milliseconds punch cycle, an average of 565 milliseconds is available for computing because of the punch-buffer storage.

Each drum memory band has 10 words that may be used for punch storage, and any D-address given within that band will cause the card to punch from the punch-storage positions. For example, an address of 0123 will cause punching from locations 0127 through 0136. The data in drum punch storage remains unchanged by the punching operation.



INSTRUCTION: 71 0027 0000

STEP ①— Transfers contents of General Storage Locations 0027-0036 to Punch Buffer Storage.

STEP ②— Transfers contents of Punch Buffer Storage to card at Punching Station through control panel wiring.

FIGURE 31

69 LD (Load Distributor)

This operation code causes the contents of the D-address location of the instruction to be placed in the distributor.

Op Code		Contents of Drum Location	Accumulator Upper	Accumulator Lower	Distributor
69 LD	*before:*	0000123456+	0000000000	0000643217+	0000643217+
	after:	0000123456+	0000000000	0000643217+	0000123456+
		Data Address	**Accumulator Upper**	**Accumulator Lower**	**Distributor**
69 LD	*before:*	8003	1234567890	3838567890+	3838567890+
	after:		1234567890	3838567890+	1234567890+
69 LD	*before:*	8002	1234567890	3838567890–	1234567890–
	after:		1234567890	3838567890–	3838567890–

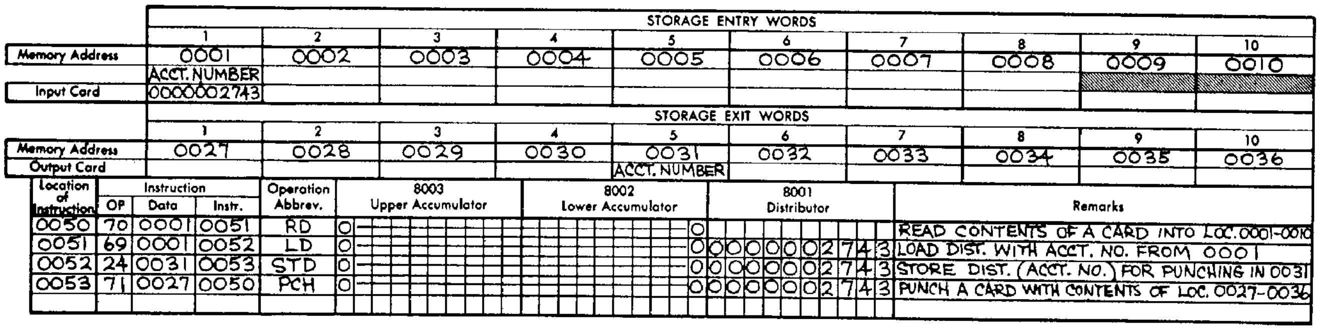
24 STD (Store Distributor)

This operation code causes the contents of the distributor with the distributor sign to be stored in the location specified by the D-address of the instruction. The contents of the distributor remain undisturbed.

It is important to remember that the D-address for *all* store instructions must be 0000-1999. An 8000-series D-address will not be accepted as valid by the machine on any of the store instructions.

Op Code		Contents of Drum Location	Distributor
24 STD	*before:*	0123456789+	0001042666–
	after:	0001042666–	0001042666–

Figure 32 shows a program to accomplish read, transfer, and punch.



	STORAGE ENTRY WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010
	ACCT. NUMBER								▨	▨
Input Card	0000002743									

	STORAGE EXIT WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address	0027	0028	0029	0030	0031	0032	0033	0034	0035	0036
Output Card					ACCT. NUMBER					

Location of Instruction	Instruction OP	Instruction Data	Instruction Instr.	Operation Abbrev.	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
0050	70	0001	0051	RD	0—	—0		READ CONTENTS OF A CARD INTO LOC. 0001-0010
0051	69	0001	0052	LD	0—	—0	0000002743	LOAD DIST. WITH ACCT. NO. FROM 0001
0052	24	0031	0053	STD	0—	—0	0000002743	STORE DIST. (ACCT. NO.) FOR PUNCHING IN 0031
0053	71	0027	0050	PCH	0—	—0	0000002743	PUNCH A CARD WITH CONTENTS OF LOC. 0027-0036

FIGURE 32. POSITION ACCOUNT NUMBER FOR PUNCHING

Addition and Subtraction

In any practical use of the Type 650, there will be a considerable amount of addition and subtraction involved in the solution of a problem. The basic function of the addition and subtraction operations is the same and, for that reason, they have been grouped in the discussion that follows.

The distributor will contain the D-address factor following any addition or subtraction operation.

10 AU (Add to Upper)

This operation code causes the contents of the D-address location to be added to the contents of the upper half of the accumulator. The lower half of the accumulator will remain unaffected unless the addition causes the sign of the accumulator to change, in which case the contents of the lower half of the accumulator will be complemented. Also, the units position of the upper half of the accumulator will be reduced by one.

Op Code		Contents of Drum Location	Accumulator Upper	Accumulator Lower	Distributor
10 AU	*before:*	0012345678+	0000458632	8942711365+	8942711365+
	after:	0012345678+	0012804310	8942711365+	0012345678+
10 AU	*before:*	0012345678+	0000458632	8942711365−	8942711365−
	after:	0012345678+	0011887045	1057288635+	0012345678+
10 AU	*before:*	0012345678+	9989374627	8942711365+	8942711365+
	after:	0012345678+	0001720305 (overflow)	8942711365+	0012345678+
		Data Address	**Accumulator Upper**	**Accumulator Lower**	**Distributor**
10 AU	*before:*	8003	1234567890	3838567890+	3838567890+
	after:		2469135780	3838567890+	1234567890+
10 AU	*before:*	8002	1234567890	3838567890+	1234567890+
	after:		5073135780	3838567890+	3838567890+
10 AU	*before:*	8001	1234567890	3838567890+	1234567890−
	after:		0000000000	3838567890+	1234567890−

15 AL (Add to Lower)

This operation code causes the contents of the D-address location to be added to the contents of the lower half of the accumulator. The contents of the upper half of the accumulator could be affected by carries.

Op Code		Contents of Drum Location	Accumulator Upper	Accumulator Lower	Distributor
15 AL	*before:*	0012345678+	0000000000	9989374627+	9989374627+
	after:	0012345678+	0000000001	0001720305+	0012345678+
15 AL	*before:*	0012345678+	0000000000	9989374627−	9989374627−
	after:	0012345678+	0000000000	9977028949−	0012345678+
15 AL	*before:*	0012345678−	0000000000	9989374627−	9989374627−
	after:	0012345678−	0000000001	0001720305−	0012345678−
		Data Address	**Accumulator Upper**	**Accumulator Lower**	**Distributor**
15 AL	*before:*	8003	9876543210	3838567890+	1234567890+
	after:		987654321(1)	3715111100+	9876543210+
15 AL	*before:*	8002	1234567890	3838567890+	1234567890+
	after:		1234567890	7677135780+	3838567890+
15 AL	*before:*	8001	1234567890	3838567890+	3838567890−
	after:		1234567890	0000000000+	3838567890−

11 SU (Subtract from Upper)

This operation code causes the contents of the D-address location to be subtracted from the contents of the upper half of the accumulator. The contents of the lower half of the accumulator will remain unaffected unless the subtraction causes a change of sign in the accumulator, in which case the contents of the lower half of the accumulator will be complemented. Also, the units position of the upper half of the accumulator will be reduced by one.

Op Code		Contents of Drum Location	Accumulator Upper	Accumulator Lower	Distributor
11 SU	*before:*	0012345678–	0000458632	8942711365–	0000458632–
	after:	0012345678–	0011887045	1057288635+	0012345678–
11 SU	*before:*	0012345678+	0000458632	8942711365–	0000458632–
	after:	0012345678+	0012804310	8942711365–	0012345678+
11 SU	*before:*	0012345678+	9989374627	8942711365–	9989374627–
	after:	0012345678+	0001720305 (overflow)	8942711365–	0012345678+
		Data Address	Accumulator Upper	Accumulator Lower	Distributor
11 SU	*before:*	8003	1234567890	3838567890+	3838567890+
	after:		0000000000	3838567890+	1234567890+
11 SU	*before:*	8002	1234567890	3838567890+	1234567890+
	after:		2603999999	6161432110–	3838567890+
11 SU	*before:*	8001	1234567890	3838567890+	1234567890+
	after:		0000000000	3838567890+	1234567890+

16 SL (Subtract from Lower)

This operation code causes the contents of the D-address location to be subtracted from the contents of the lower half of the accumulator. The contents of the upper half of the accumulator could be affected by carries.

Op Code		Contents of Drum Location	Accumulator Upper	Accumulator Lower	Distributor
16 SL	*before:*	0012345678+	0000000000	9989374627+	9989374627+
	after:	0012345678+	0000000000	9977028949+	0012345678+
16 SL	*before:*	0012345678–	0000000000	9989374627+	9989374627+
	after:	0012345678–	0000000001	0001720305+	0012345678–
16 SL	*before:*	0012345678–	0000000000	9989374627–	9989374627–
	after:	0012345678–	0000000000	9977028949–	0012345678–
		Data Address	Accumulator Upper	Accumulator Lower	Distributor
16 SL	*before:*	8003	9876543210	3838567890+	3838567890+
	after:		98765432(09)	3962024680+	9876543210+
16 SL	*before:*	8002	1234567890	3838567890+	1234567890+
	after:		1234567890	0000000000+	3838567890+
16 SL	*before:*	8001	1234567890	3838567890+	3838567890+
	after:		1234567890	0000000000+	3838567890+

Addition and Subtraction with Reset

The reset-add and reset-subtract operation codes differ in function from the add and subtract operations in that the entire accumulator is reset to zero before the new factor is entered.

60 RAU
(Reset and Add into Upper)

This operation code resets the entire accumulator to plus zero and adds the contents of the D-address location into the upper half of the accumulator.

Op		Contents of	Accumulator		
Code		Drum Location	Upper	Lower	Distributor
60 RAU	*before:*	0012345678+	8942711365	9989374627+	8942711365+
	after:	0012345678+	0012345678	0000000000+	0012345678+
60 RAU	*before:*	0012345678–	8942711365	9989374627+	9989374627+
	after:	0012345678–	0012345678	0000000000–	0012345678–
			Accumulator		
		Data Address	Upper	Lower	Distributor
60 RAU	*before:*	8003	1234567890	3838567890+	3838567890+
	after:		1234567890	0000000000+	1234567890+
60 RAU	*before:*	8002	1234567890	3838567890+	1234567890+
	after:		3838567890	0000000000+	3838567890+
60 RAU	*before:*	8001	1234567890	3838567890+	1234567890–
	after:		1234567890	0000000000–	1234567890–

65 RAL
(Reset and Add into Lower)

This operation code resets the entire accumulator to plus zero and adds the contents of the D-address location into the lower half of the accumulator.

Op		Contents of	Accumulator		
Code		Drum Location	Upper	Lower	Distributor
65 RAL	*before:*	0012345678+	8942711365	9989374627+	8942711365+
	after:	0012345678+	0000000000	0012345678+	0012345678+
65 RAL	*before:*	0012345678–	8942711365	9989374627+	8942711365+
	after:	0012345678–	0000000000	0012345678–	0012345678–
			Accumulator		
		Data Address	Upper	Lower	Distributor
65 RAL	*before:*	8003	1234567890	3838567890+	3838567890+
	after:		0000000000	1234567890+	1234567890+
65 RAL	*before:*	8002	1234567890	3838567890+	3838567890+
	after:		0000000000	3838567890+	3838567890+
65 RAL	*before:*	8001	1234567890	3838567890+	1234567890–
	after:		0000000000	1234567890–	1234567890–

61 RSU
(Reset and Subtract into Upper)

This operation code resets the entire accumulator to plus zero and subtracts the contents of the D-address location into the upper half of the accumulator.

Op Code		Contents of Drum Location	Accumulator Upper	Lower	Distributor
61 RSU	*before:*	0012345678+	8942711365	9989374627+	8942711365+
	after:	0012345678+	0012345678	0000000000–	0012345678+
61 RSU	*before:*	0012345678–	8942711365	9989374627+	8942711365+
	after:	0012345678–	0012345678	0000000000+	0012345678–
		Data Address	Accumulator Upper	Lower	Distributor
61 RSU	*before:*	8003	1234567890	3838567890+	3838567890+
	after:		1234567890	0000000000–	1234567890+
61 RSU	*before:*	8002	1234567890	3838567890+	1234567890+
	after:		3838567890	0000000000–	3838567890+
61 RSU	*before:*	8001	1234567890	3838567890+	1234567890+
	after:		1234567890	0000000000–	1234567890+

66 RSL
(Reset and Subtract into Lower)

This operation code resets the entire accumulator to plus zero and subtracts the contents of the D-address location into the lower half of the accumulator.

Figure 33, which follows the accumulator store operations, shows a program using addition and subtraction.

Op Code		Contents of Drum Location	Accumulator Upper	Lower	Distributor
66 RSL	*before:*	0012345678+	8942711365	9989374627+	8942711365+
	after:	0012345678+	0000000000	0012345678–	0012345678+
66 RSL	*before:*	0012345678–	8942711365	9989374627+	9989374627+
	after:	0012345678–	0000000000	0012345678+	0012345678–
		Data Address	Accumulator Upper	Lower	Distributor
66 RSL	*before:*	8003	1234567890	3838567890+	3838567890+
	after:		0000000000	1234567890–	1234567890+
66 RSL	*before:*	8002	1234567890	3838567890–	3838567890–
	after:		0000000000	3838567890+	3838567890–
66 RSL	*before:*	8001	1234567890	3838567890+	1234567890+
	after:		0000000000	1234567890–	1234567890+

Accumulator Store Operations

There are also store operations for the contents of either half of the accumulator, and two special store instructions for portions of the contents of the lower half of the accumulator.

20 STL (Store Lower in Memory)

This operation code causes the contents of the lower half of the accumulator with the accumulator sign to be stored in the location specified by the D-address of the instruction. The contents of the lower half of the accumulator remain undisturbed.

It is important to remember that the D-address for all store instructions must be 0000-1999. An 8000 series D-address will not be accepted as valid by the machine on any of the store instructions.

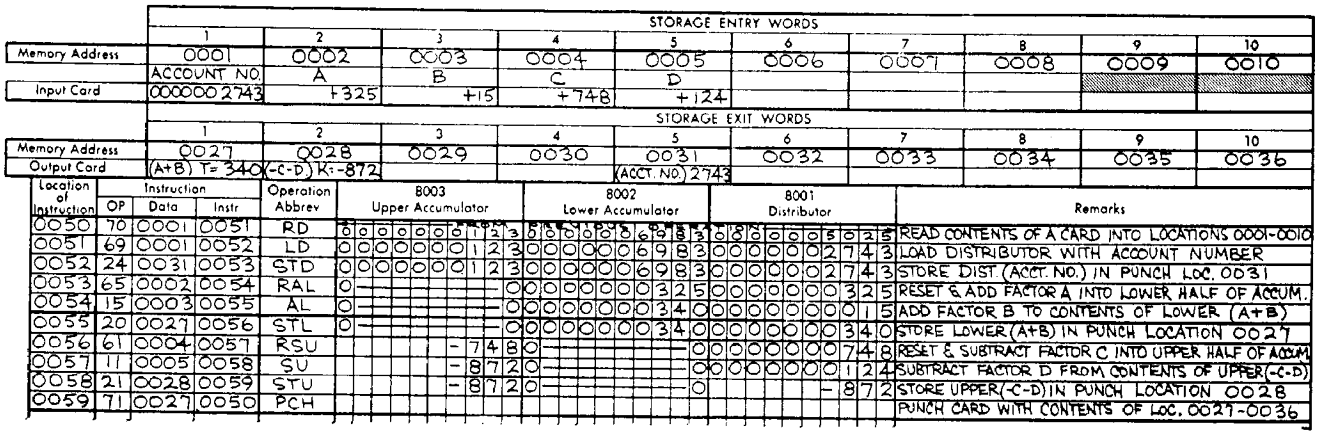
Op Code		Contents of Drum Location	Accumulator Upper	Accumulator Lower	Distributor
20 STL	*before:*	0123456789+	0404532877	0000553882−	0000012998+
	after:	0000553882−	0404532877	0000553882−	0000553882−

21 STU (Store Upper in Memory)

This operation code causes the contents of the upper half of the accumulator with the accumulator sign to be stored in the location specified by the D-address of the instruction. If STU is performed following a division operation, and before another division, multiplication, or reset operation takes place, the contents of the upper accumulator will be stored with the sign of the remainder from the divide operation (OP-CODE 14). The contents of the upper half of the accumulator remain undisturbed.

It is important to remember that the D-address for all store instructions must be 0000-1999. An 8000 series D-address will not be accepted as valid by the machine on any of the store instructions.

Op Code		Contents of Drum Location	Upper	Lower	Distributor
21 STU	*before:*	0123456789+	0404532877	0000553882+	0000012998+
	after:	0404532877+	0404532877	0000553882+	0404532877+
21 STU	*before:*	0123456789+	0404532877	0000553882−	0000012998+
	after:	0404532877−	0404532877	0000553882−	0404532877−
			Remainder	Quotient	
21 STU	*before:*	0136543210+	0006349211−	0014987621+	0006432198+
	after:	0006349211−	0006349211−	0014987621+	0006349211−



	STORAGE ENTRY WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010
	ACCOUNT NO.	A	B	C	D				▨	▨
Input Card	0000002743	+325	+15	+748	+124					

	STORAGE EXIT WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address	0027	0028	0029	0030	0031	0032	0033	0034	0035	0036
Output Card	(A+B) T= 340	(-C-D) K: -872			(ACCT. NO.) 2743					

Location of Instruction	Instruction OP	Instruction Data	Instruction Instr	Operation Abbrev	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
0050	70	0001	0051	RD	0000000123	0000069833	0800005025	READ CONTENTS OF A CARD INTO LOCATIONS 0001-0010
0051	69	0001	0052	LD	0000000123	0000069833	0000002743	LOAD DISTRIBUTOR WITH ACCOUNT NUMBER
0052	24	0031	0053	STD	0000000123	0000069833	0000002743	STORE DIST. (ACCT. NO.) IN PUNCH LOC. 0031
0053	65	0002	0054	RAL	0 — 0	0000000325	0000000325	RESET & ADD FACTOR A INTO LOWER HALF OF ACCUM.
0054	15	0003	0055	AL	0 — 0	0000000340	0000000015	ADD FACTOR B TO CONTENTS OF LOWER (A+B)
0055	20	0027	0056	STL	0 — 0	0000000340	0000000340	STORE LOWER (A+B) IN PUNCH LOCATION 0027
0056	61	0004	0057	RSU	-748	0 — 0	0000000748	RESET & SUBTRACT FACTOR C INTO UPPER HALF OF ACCUM
0057	11	0005	0058	SU	-8720	0 — 0	0000000124	SUBTRACT FACTOR D FROM CONTENTS OF UPPER (-C-D)
0058	21	0028	0059	STU	-8720	0 — 0	-872	STORE UPPER (-C-D) IN PUNCH LOCATION 0028
0059	71	0027	0050	PCH				PUNCH CARD WITH CONTENTS OF LOC. 0027-0036

FIGURE 33. ADDITION, SUBTRACTION, STORING

22 STDA
(Store Lower Data Address)

This operation code causes positions 8-5 of the distributor to be replaced by the contents of the corresponding positions of the lower half of the accumulator. The modified word in the distributor with the sign of the distributor is then stored in the location specified by the D-address of the instruction. The contents of the lower half of the accumulator remain unchanged, and the sign of the accumulator is not transferred to the distributor. The modified word remains in the distributor upon completion of the operation.

It is important to remember that the D-address for all store instructions must be 0000-1999. An 8000 series D-address will not be accepted as valid by the machine on any of the store instructions.

Figure 34 shows how instructions from the program may be modified by addition and by use of the 22 code.

Op Code		Data Address	Location 0100	Accumulator Upper	Accumulator Lower	Distributor
22 STDA	*before:*	0100	69 1946 0600+	0000000000	00 1988 0560 −	69 1950 0600+
	after:		69(1988)0600+	0000000000	00(1988)0560 −	69(1988)0600+

	STORAGE ENTRY WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address										
Input Card									▨	▨

	STORAGE EXIT WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address										
Output Card										

Location of Instruction	Instruction OP	Instruction Data	Instruction Instr.	Operation Abbrev.	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
0100	65	0022	0101	RAL	0————0	6008000104	6008000104	PLACE INSTRUCTION FROM LOC. 0022 IN LOWER
0101	15	1849	0102	AL	0————0	6009500104	0001500000	ADD 0150 TO THE DATA ADDRESS
0102	69	0105	0103	LD	0————0	6009500104	2109490106	LOAD DIST. WITH INSTRUCTION IN LOC. 0105
0103	22	0105	8002	STDA	0————0	6009500104	2109500106	INSERT NEW DATA ADDRESS AND RESTORE IN 0105
8002	60	(0950)	0104	RAU	0000063855	0————0	0000063855	PLACE FACTOR FROM 0950 IN UPPER
0104	10	1850	0105	AU	0000063938	0————0	0000000083	ADD FACTOR FROM 1850 TO UPPER
0105	21	(0950)	0106	STU	0000063938	0————0	0000063938	STORE SUM IN LOCATION 0950

FIGURE 34. INSTRUCTION MODIFICATION

23 STIA
(Store Lower Instruction Address)

This operation code causes positions 4-1 of the distributor to be replaced by the contents of the corresponding positions of the lower half of the accumulator. The modified word in the distributor with the sign of the distributor is then stored in the location specified by the D-address of the instruction. The contents of the lower half of the accumulator remain unchanged, and the sign of the accumulator is not transferred to the distributor. The modified word remains in the distributor upon completion of the operation.

It is important to remember that the D-address for all store instructions must be 0000-1999. An 8000 series D-address will not be accepted as valid by the machine on any of the store instructions.

Figure 35 shows the use of codes 22 and 23 to separate independent factors within a single word.

Op. Code		Data Address	Location 0100	Accumulator Upper	Accumulator Lower	Distributor
23 STIA	*before:*	0100	69 1800 0715 +	0000000000	00 1988 0816 +	69 1800 0750 −
	after:		69 1800(0816) −	0000000000	00 1988(0816)+	69 1800(0816) −

	STORAGE ENTRY WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address	0101									
	A | B | C									
Input Card	12|3456|7890								▨	▨

	STORAGE EXIT WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address										
Output Card										

Location of Instruction	Instruction OP	Instruction Data	Instruction Instr.	Operation Abbrev.	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
0025	65	0101	0026	RAL	0————0	1234567890	1234567890	ADD WORD CONTAINING FACTOR A, B AND C
0026	69	8003	0027	LD	0————0	1234567890	0————0	LOAD DISTRIBUTOR WITH ZEROS
0027	23	0149	0028	STIA	0————0	1234567890	0000007890	SEPARATE AND STORE FACTOR C
0028	16	8001	0029	SL	0————0	1234560000	0000007890	CLEAR C PORTION OF LOWER
0029	69	8003	0030	LD	0————0	1234560000	0————0	LOAD DISTRIBUTOR WITH ZEROS
0030	22	0148	0031	STDA	0————0	1234560000	0034560000	SEPARATE AND STORE FACTOR B
0031	16	8001	0032	SL	0————0	1200000000	0034560000	CLEAR B PORTION OF LOWER
0032	20	0147	0033	STL	0————0	1200000000	1200000000	STORE FACTOR A

FIGURE 35. FACTOR SEPARATION

Addition and Subtraction — Absolute Values

Operation codes are provided that enable the absolute value of the incoming factor to be added or subtracted. These operations ignore the actual algebraic sign of the incoming factor and automatically treat it as a positive factor.

17 AABL (Add Absolute to Lower)

This operation code causes the contents of the D-address location to be added to the contents of the lower half of the accumulator as a positive factor regardless of the actual sign. When the operation is completed, the distributor will contain the D-address factor with its actual sign.

Op. Code		Contents of Drum Location	Accumulator Upper	Lower	Distributor
17 AABL	*before:*	0012345678+	0000000000	9989374627+	9989374627+
	after:	0012345678+	0000000001	0001720305+	0012345678+
17 AABL	*before:*	0012345678–	0000000000	8942711365–	8942711365–
	after:	0012345678–	0000000000	8930365687–	0012345678–
17 AABL	*before:*	0012345678+	0000000000	9989374627–	9989374627–
	after:	0012345678+	0000000000	9977028949–	0012345678+

67 RAABL (Reset and Add Absolute into Lower)

This operation code resets the entire accumulator to zeros and adds the contents of the D-address location into the lower half of the accumulator as a positive factor regardless of its actual sign. When the operation is completed, the distributor will contain the D-address factor with its actual sign.

Op. Code		Contents of Drum Location	Accumulator Upper	Lower	Distributor
67 RAABL	*before:*	0012345678–	8942711365	9989374627–	9989374627–
	after:	0012345678–	0000000000	0012345678+	0012345678–
67 RAABL	*before:*	0012345678+	8942711365	9989374627+	8942711365+
	after:	0012345678+	0000000000	0012345678+	0012345678+

18 SABL (Subtract Absolute from Lower)

This operation code causes the contents of the D-address location to be subtracted from the contents of the lower half of the accumulator as a positive factor regardless of the actual sign. When the operation is completed, the distributor will contain the D-address factor with its actual sign.

Op. Code		Contents of Drum Location	Accumulator Upper	Lower	Distributor
18 SABL	*before:*	0012345678–	0000000000	8942711365–	8942711365–
	after:	0012345678–	0000000000	8955057043–	0012345678–
18 SABL	*before:*	0012345678+	0000000000	8942711365+	8942711365+
	after:	0012345678+	0000000000	8930365687+	0012345678+
18 SABL	*before:*	0012345678+	0000000000	9989374627–	9989374627–
	after:	0012345678+	0000000001	0001720305–	0012345678+

68 RSABL (Reset and Subtract Absolute into Lower)

This operation code resets the entire accumulator to plus zero and subtracts the contents of the D-address location into the lower half of the accumulator as a positive factor, regardless of the actual sign. When the operation is completed, the distributor will contain the D-address factor with its actual sign.

Op. Code		Contents of Drum Location	Accumulator Upper	Lower	Distributor
68 RSABL	*before:*	0012345678+	8942711365	9989374627+	9989374627+
	after:	0012345678+	0000000000	0012345678–	0012345678+
68 RSABL	*before:*	0012345678–	8942711365	9989374627+	9989374627+
	after:	0012345678–	0000000000	0012345678–	0012345678–

Multiplication

Multiplication in the Type 650 is accomplished by repeated addition. The number of additions is controlled in the following manner:

1. The contents of the entire accumulator are shifted one position to the left. This places the high-order digit of the multiplier in a special storage position. The digit in the special position is then analyzed for a zero or nonzero condition. If it is zero, another left shift is initiated. If it is nonzero, an add cycle is set up.

2. The multiplicand is added into the lower half of the accumulator. The digit in the special position is decreased by one each time the multiplicand is added. As soon as the digit becomes zero, the addition operation is stopped, and the contents of the entire accumulator are again shifted to the left to place the next digit of the multiplier in the special position.

Thus, the machine process of multiplication is essentially the same as the paper and pencil process used by most people. The only difference is that the multiplier digits are handled in the reverse order.

```
HAND METHOD               MACHINE METHOD

  1111                            1111
× 111                           × 111
  ----                            ----
  1111                            1111
 1111  shift 1           shift 1   1111
1111   shift 1           shift 1    1111
------                          ------
123321                          123321

 1234                           1234
   23                             23
 ----                           ----
 3702                           1234
2468                            1234
-----                            1234
28382                            1234
                                 1234
                                -----
                                28382
```

19 MULT (Multiply)

This operation code causes the machine to multiply. A 10-digit multiplicand may be multiplied by a 10-digit multiplier to develop a 20-digit product. The multiplier must be placed in the upper accumulator prior to multiplication. The location of the multiplicand is specified by the D-address of the instruction. The product is developed in the accumulator beginning in the low-order position of the lower half of the accumulator and extending to the left into the upper half of the accumulator as required. The multiplier (originally in the upper half of the accumulator) will be lost during multiplication. The multiplicand will remain in the distributor.

The operations necessary to position factors for decimal point alignment and to half-adjust results are covered under *Shifting*.

Op Code		Multiplicand	Accumulator Upper	Accumulator Lower	Distributor
19 MULT	*before:*	0012345678+	8942711365	0000000000+	8942711365+
	after:	0012345678+	0011040383	4959230470+	0012345678+
19 MULT	*before:*	0012345678–	8942711365	0000000000+	8942711365+
	after:	0012345678–	0011040383	4959230470–	0012345678–
19 MULT	*before:*	0012345678–	8942711365	0000000(111)+	8942711365+
	after:	0012345678–	0011040(494)	4959230470–	0012345678–
19 MULT*	*before:*	0012345678–	8942711365	9999999999+	8942711365+
	after:	0012345678–	0011040382	4971576148–	0012345678–
Correct Answer		0012345678–	10011040382	4959230470–	0012345678–

* A carry is propagated into the multiplier to cause an erroneous answer. The carry is also lost in the product as the accumulator has a 20-digit capacity.

If the lower half of the accumulator contains some number at the start of a multiply operation, the absolute value of that number will be added to the absolute value of that portion of the product developed in the upper half of the accumulator. The regular rules of carry and overflow apply in this case. Thus, a sufficiently large number in the lower half of the accumulator could increase the absolute value of the multiplier.

Division

Division in the Type 650 is accomplished by repeated subtraction. The number of subtractions is controlled in the following manner:

1. The contents of the entire accumulator are shifted one position to the left. This places the high-order digit of the dividend in a special storage position and inserts a zero in the units position of the lower half of the accumulator.

2. The ten-digit divisor is subtracted from the contents of the upper half of the accumulator and the special digit position. The subtraction process continues until an overdraw occurs. When the overdraw occurs, the subtraction process stops, and the overdraw is corrected by adding the divisor back into the upper half of the accumulator. Following each subtraction on which an overdraw does not occur, a count of one is added to the units position of the lower half of the accumulator.

3. Following the overdraw correction, the entire accumulator contents are again shifted one position to the left and the process repeats for ten cycles (shifts). The remainder, if any, is the amount left in the upper half of the accumulator after the last overdraw correction.

Thus, the machine process of division is essentially the same as the paper and pencil process of division.

HAND METHOD

```
       1234
23)   28382
      23
       53
       46
        78
        69
         92
         92
```

MACHINE METHOD

```
       1234
23)   28382
     -23
        5
     -23
     -18
     +23
       53
     -23
       30
     -23
        7
     -23
     -16
     +23
        78
      -23
        55
      -23
        32
      -23
         9
      -23
      -14
      +23
         92
       -23
         69
       -23
         46
       -23
         23
       -23
          0
       -23
       -23
       +23
          0
```

14 DIV (Divide)

This operation code causes the machine to divide without resetting the remainder. A 20-digit dividend may be divided by a 10-digit divisor to produce a 10-digit quotient. In order to remain within these limits, the absolute value of the divisor must be *greater than* the absolute value of that portion of the dividend that is in the upper half of the accumulator. The entire dividend is placed in the 20-position accumulator. The location of the divisor is specified by the D-address of the divide instruction. The operations necessary to position the dividend for decimal-point alignment and to half-adjust the results are covered under *Shifting*.

If the machine attempts to develop a quotient of more than 10 digits, a quotient overflow will occur, and the machine will stop. If division by zero is attempted (divisor of zero), a quotient overflow will always occur, and the machine will stop.

At the completion of the divide operation, the divisor is in the distributor; the quotient, with its sign, is in the lower half of the accumulator; and the remainder, with its sign, is in the upper half of the accumulator. The sign of the remainder is the same as the sign of the dividend and will continue to be associated with the upper half of the accumulator until a reset or another divide operation is executed. This is the only condition under which the upper half of the accumulator will acquire an independent sign.

If there is a possibility of a difference in the sign of the quotient and the sign of the remainder, the remainder should be stored in some drum memory location or loaded into the distributor before any arithmetical operation using the remainder is performed. This is necessary to insure that the remainder retains its proper sign.

If a multiply operation is executed any time after a code 14 operation but before a reset operation is given, the sign of the accumulator will be placed in the remainder sign position. A store upper operation would then store the sign of the multiplier, not the sign of the product.

If the branch minus code (46) is used after a divide operation, it is the sign of the accumulator, not the sign of the remainder, that is tested by this code.

Op. Code		Divisor	Accumulator Upper	Accumulator Lower	Distributor
				(dividend)	
14 DIV	*before:*	0000000125+	0000000000	0000004682+	0000004682+
			(remainder)	(quotient)	(divisor)
	after:	0000000125+	0000000057+	0000000037+	0000000125+
				(dividend)	
14 DIV	*before:*	0000000125 –	0000000000	0000004682+	0000004682+
			(remainder)	(quotient)	(divisor)
	after:	0000000125 –	0000000057 (+)	0000000037 (–)	0000000125 –
				(dividend)	
14 DIV	*before:*	0000000125+	0000000000	0000004682 –	0000004682 –
			(remainder)	(quotient)	(divisor)
	after:	0000000125+	0000000057 (–)	0000000037 (–)	0000000125+
				(dividend)	
14 DIV	*before:*	0000000125+	0000000852	1034010200+	0000000852+
					(divisor)
	after:	0000000125+	OVERFLOW		0000000125+

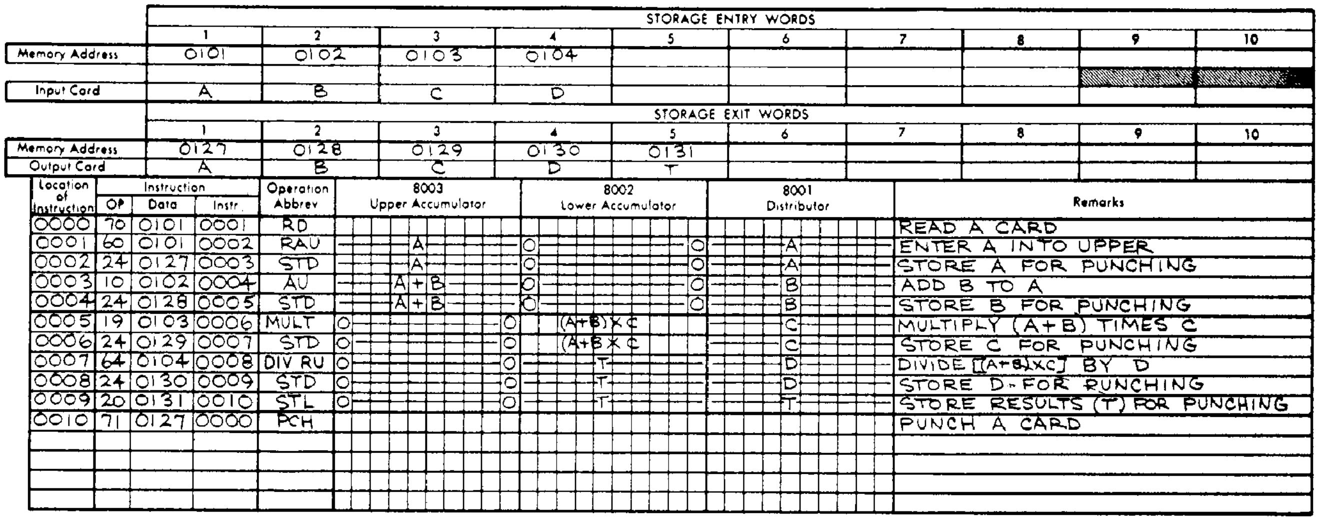
64 DIV RU (Divide and Reset Upper)

This operation code causes the machine to divide as explained under operation code 14 (DIV). However, the upper half of the accumulator containing the remainder with its sign is reset to zeros.

Op. Code		Divisor	Accumulator Upper	Accumulator Lower	Distributor
64 DIVRU	*before:*	0000000125+	0000000000	0000004682+	0000004682+
	after:	0000000125+	0000000000	0000000037+	0000000125+
64 DIVRU	*before:*	0000000125 –	0000000000	0000004682+	0000004682+
	after:	0000000125 –	0000000000	0000000037 –	0000000125 –
64 DIVRU	*before:*	0000000125+	0000000000	0000004682 –	0000004682 –
	after:	0000000125+	0000000000	0000000037 –	0000000125+
64 DIVRU	*before:*	0000000125+	0000000852	1034010200+	0000000852+
	after:	0000000125+	OVERFLOW		0000000125+

Figure 36 shows a program to solve the equation:

$$\frac{(A+B) \times C}{D} = T$$



	STORAGE ENTRY WORDS 1	2	3	4	5	6	7	8	9	10
Memory Address	0101	0102	0103	0104						
Input Card	A	B	C	D						

	STORAGE EXIT WORDS 1	2	3	4	5	6	7	8	9	10
Memory Address	0127	0128	0129	0130	0131					
Output Card	A	B	C	D	T					

Location of Instruction	Instruction OP	Instruction Data	Instruction Instr.	Operation Abbrev	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
0000	70	0101	0001	RD				READ A CARD
0001	60	0101	0002	RAU	A	0 0	A	ENTER A INTO UPPER
0002	24	0127	0003	STD	A	0 0	A	STORE A FOR PUNCHING
0003	10	0102	0004	AU	A+B	0 0	B	ADD B TO A
0004	24	0128	0005	STD	A+B	0 0	B	STORE B FOR PUNCHING
0005	19	0103	0006	MULT	0 0	(A+B)×C	C	MULTIPLY (A+B) TIMES C
0006	24	0129	0007	STD	0 0	(A+B×C	C	STORE C FOR PUNCHING
0007	64	0104	0008	DIV RU	0 0	T	D	DIVIDE [(A+B)×C] BY D
0008	24	0130	0009	STD	0 0	T	D	STORE D FOR PUNCHING
0009	20	0131	0010	STL	0 0	T	T	STORE RESULTS (T) FOR PUNCHING
0010	71	0127	0000	PCH				PUNCH A CARD

FIGURE 36. MULTIPLICATION AND DIVISION

Branching

Branching is the logical means of recognizing one of two possible conditions. A branching instruction consists of the branch code, the data address, and the instruction address:

Branch Code	Data Address	Instruction Address
XX	XXXX	XXXX

When a branching instruction is used, the machine chooses either the data address or the instruction address of the branching instruction as the location of the next instruction to be performed. If the condition of the branch code is fulfilled, the location of the next instruction is specified by the data address of the branching instruction. If the condition of the branch code is not fulfilled, the location of the next instruction is specified by the instruction address of the branching instruction.

44 BRNZU (Branch on Non-Zero in Upper)

This operation code causes the contents of the upper half of the accumulator to be examined for zero. If the contents of the upper half of the accumulator is non-zero, the location of the next instruction to be executed is specified by the D-address. If the content of the upper half of the accumulator is zero, the location of the next instruction to be executed is specified by the I-address. The sign of the accumulator is ignored.

Op Code	Accumulator Upper	Accumulator Lower	Next Instruction
44 BRNZU	0012345678	0008764329+	Take D-address (branch)
	0000000123	0000000000+	Take D-address (branch)
	0012345678	0000000000−	Take D-address (branch)
	0000000000	0012345678+	Take I-address (no branch)
	0000000000	0000000000+	Take I-address (no branch)
	0000000000	0000000000−	Take I-address (no branch)

45 BRNZ (Branch on Non-Zero)

This operation code causes the content of the entire accumulator to be examined for zero. If the content of the accumulator is non-zero, the location of the next instruction to be executed is specified by the D-address. If the contents of the accumulator is zero, the location of the next instruction to be executed is specified by the I-address. The sign of the accumulator is ignored.

Op Code	Accumulator Upper	Accumulator Lower	Next Instruction
45 BRNZ	0000000123	0000000000+	Take D-address (branch)
	0000000000	0000000123+	Take D-address (branch)
	0000000000	0000000123−	Take D-address (branch)
	0000000000	0000000000+	Take I-address (no branch)
	0000000000	0000000000−	Take I-address (no branch)

46 BRMIN (Branch on Minus)

This operation code causes the sign of the accumulator to be examined for minus. If the sign of the accumulator is minus, the location of the next instruction to be executed is specified by the D-address. If the sign of the accumulator is positive, the location of the next instruction to be executed is specified by the I-address. The contents of the accumulator are ignored.

Op Code	Accumulator Upper	Accumulator Lower	Next Instruction
46 BRMIN	0000000000	0000000123−	Take D-address (branch)
	0000000123	0000000000−	Take D-address (branch)
	0000000000	0000000000−	Take D-address (branch)
	0000000000	0000000000+	Take I-address (no branch)
	0000000000	0000000123+	Take I-address (no branch)
	0000000123	0000000000+	Take I-address (no branch)

If branch minus is used after a divide-without-reset operation (14), it is the sign of the accumulator (quotient) not the sign of the remainder that is tested.

It is extremely important, when operation code 46 (BRMIN) is being used, to be aware of the fact that the sign of the accumulator can be minus when its contents are zero. Refer to minus zeros discussion in *Programming and Operating Suggestions.*

47 BROV (Branch on Overflow)

This operation code causes the overflow circuit to be examined to see whether it has been set. If the overflow circuit is set, the location of the next instruction to be executed is specified by the D-address. If the overflow circuit is not set, the location of the next instruction to be executed is specified by the I-address. In order for the machine to branch on overflow, the overflow switch on the control console must be set to SENSE. If the switch is set to STOP, any overflow will cause the machine to stop. The overflow switch is covered in detail in the *Control Console* section. Overflow may occur in one of the following ways:

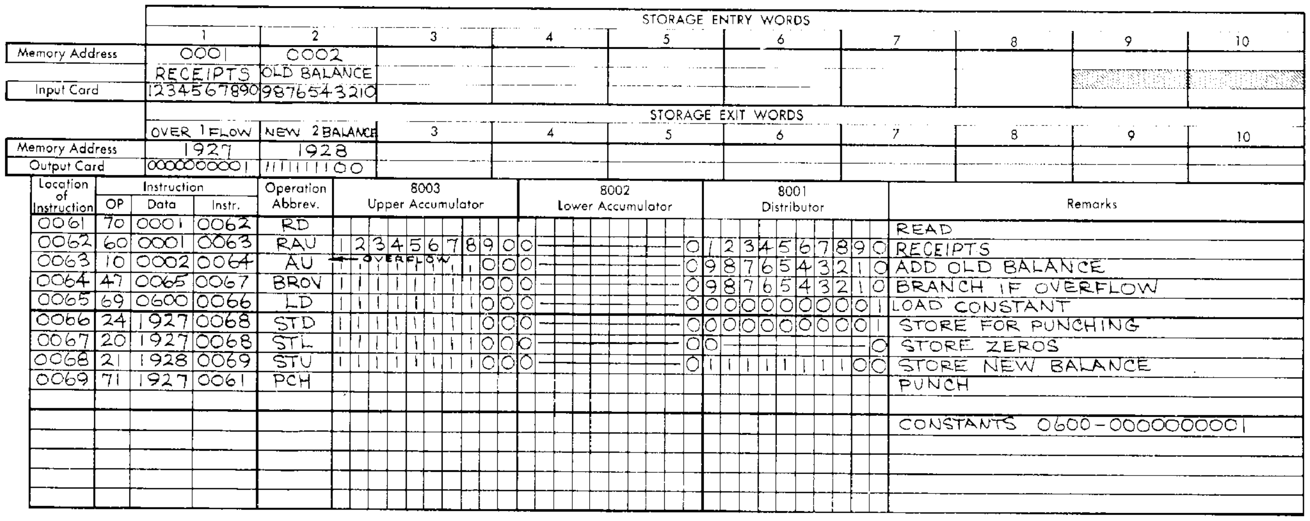
1. An excessive accumulation

2. Multiplication operation in which a sufficiently large factor was in the lower accumulator before multiplication began.

3. Trying to exceed the number of shifts called for in a *shift and count* operation.

4. Trying to develop a quotient of more than ten (10) digits. *A division overflow will always stop the machine.*

Execution of the BROV instruction resets the overflow sense circuit. Another overflow must occur before the circuit is activated again.



	STORAGE ENTRY WORDS 1	2	3	4	5	6	7	8	9	10
Memory Address	0001	0002								
	RECEIPTS	OLD BALANCE								
Input Card	1234567890	9876543210							▨	▨

	STORAGE EXIT WORDS OVER 1 FLOW	NEW 2 BALANCE	3	4	5	6	7	8	9	10
Memory Address	1927	1928								
Output Card	0000000001	1111111100								

Location of Instruction	Instruction OP	Data	Instr.	Operation Abbrev.	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
0061	70	0001	0062	RD				READ
0062	60	0001	0063	RAU	1 2 3 4 5 6 7 8 9 0 0	0	1 2 3 4 5 6 7 8 9 0	RECEIPTS
0063	10	0002	0064	AU	← OVERFLOW 1 1 1 1 1 1 1 1 0 0 0	0	9 8 7 6 5 4 3 2 1 0	ADD OLD BALANCE
0064	47	0065	0067	BROV	1 1 1 1 1 1 1 1 0 0 0	0	9 8 7 6 5 4 3 2 1 0	BRANCH IF OVERFLOW
0065	69	0600	0066	LD	1 1 1 1 1 1 1 1 0 0 0	0	0 0 0 0 0 0 0 0 0 1	LOAD CONSTANT
0066	24	1927	0068	STD	1 1 1 1 1 1 1 1 0 0 0	0	0 0 0 0 0 0 0 0 0 1	STORE FOR PUNCHING
0067	20	1927	0068	STL	1 1 1 1 1 1 1 1 0 0 0	0	0 — 0	STORE ZEROS
0068	21	1928	0069	STU	1 1 1 1 1 1 1 1 0 0 0	0	1 1 1 1 1 1 1 1 0 0	STORE NEW BALANCE
0069	71	1927	0061	PCH				PUNCH
								CONSTANTS 0600-0000000001

FIGURE 37. BRANCHING—ACCUMULATOR OVERFLOW

Figure 37 illustrates the use of code 47 to detect overflows from accumulation.

The instructions in locations 0061 through 0063 cause receipts to be added to old balance; the instruction in location 0064 tests for an overflow. If an overflow occurs, a branch is taken to location 0065, where the constant 0000000001 is loaded into the distributor. The instruction in location 0066 stores the overflow constant for punching. When no overflow is detected, the instruction in location 0067 causes zeros to be stored in punch location 1927. This is necessary because the overflow constant could be carried over from a previous calculation. The instructions in locations 0068 and 0069 cause the new balance to be stored and punched.

90-99 BRD 1-10 (Branch on 8 in Distributor Position 1-10)

This operation code examines a particular digit position in the distributor for the presence of an 8 or 9. Codes 91-99 test positions 1-9, respectively, of the test word; code 90 tests position 10. If an 8 is present, the location of the next instruction to be executed is specified by the D-address. If a 9 is present, the location of the next instruction to be executed is specified by the I-address. The presence of other than an 8 or 9 will stop the machine.

The use of one of the BRD operation codes will usually be for card type identification purposes. The entry of the digit 8 or digit 9 into a word entry will usually be through control panel wiring of a pilot selector controlled by the identifying punch in the card.

OP CODE	DISTRIBUTOR	NEXT INSTRUCTION
90	8000000000	Take D-address (branch)
	9000000000	Take I-address (no branch)
	0980000000	Error Stop
NOTE: The position tested must have either an 8 or 9. Any other digit will cause the machine to stop.		
91	0123456788	Take D-address (branch)
	0123456789	Take I-address (no branch)
92-99	0888888882	Take D-address (branch)
	0999999992	Take I-address (no branch)

Shifting

All of the following shifting operations are accomplished within the accumulator. The contents of the upper and lower halves of the accumulator are regarded as one value and shifted right or left depending on the operation. All vacant positions are filled with zeros. The sign of the accumulator is not affected by a shift operation. The distributor is not affected by a shift operation.

The shifting of all significant digits of a negative value out of the accumulator will result in a minus zero. Shifting after division (before a reset, multipy, or divide) will not change the signs associated with the upper and lower halves of the accumulator and may also result in a minus zero.

In any of the shift operations, the units position of the data address is the only digit analyzed by the machine. However, if for any reason some digits other than zero are placed in the other three positions of the data address, the four digits must be a valid machine address. If an invalid address exists, the machine will stop with a storage selection error indication.

30 SRT (Shift Right)

This operation code causes the contents of the entire accumulator to be shifted right the number of places specified by the units digit of the D-address of the shift instruction. A maximum shift of *nine* positions is possible. A data address with a units digit of zero will result in no shift. All numbers shifted off the right end of the accumulator are lost.

Op Code		Data Address	Accumulator Upper	Accumulator Lower
30 SRT	*before:*	0002	1234567890	1234567890+
	after:		0012345678	9012345678+

31 SRD (Shift and Round)

This operation code causes the contents of the entire accumulator to be shifted right the number of places specified by the units digit of the D-address of the instruction. A 5 is added (– 5 if the accumulator sign is negative) in a blind position to half-adjust the amount in the accumulator. Data addresses with a units digit of 1-9 will cause a shift of 1-9, respectively, with rounding. A data address with a units digit of zero will result in a right shift of 10 with rounding.

Op Code		Data Address	Accumulator Upper	Accumulator Lower
31 SRD	*before:*	0002	1234567890	1234567890+
	after:		0012345678	9012345679+
31 SRD	*before:*	0000	1234567890	1234567890+
	after:		0000000000	1234567890+
31 SRD	*before:*	0005	1234567890	1234567890 –
	after:		0000012345	6789012346 –

35 SLT (Shift Left)

This operation code causes the contents of the entire accumulator to be shifted left the number of places specified by the units digit of the D-address of the instruction. A maximum shift of nine positions is possible. A data address with a units digit of zero will result in no shift. All numbers shifted off the left end of the accumulator are lost. However, the overflow circuit will not be turned on.

Op Code		Data Address	Accumulator Upper	Accumulator Lower
35 SLT	*before:*	0006	1234567890	1234567890+
	after:		7890123456	7890000000+

36 SCT (Shift Left and Count)

This operation code causes

1. The contents of the entire accumulator to be shifted to the left.
2. A count of the number of places shifted to be inserted in the two low-order positions of the accumulator.

The shift and count operation code functions in the following manner. At the beginning of the shift and count operation, the tens complement of the units digit of the data address is placed in the shift counter (zero in the case of a units digit of zero). After each shift is executed, the quantity of one (1) is added to the number in the shift counter. Shifting and counting will continue until some digit other than zero is sensed in the high-order position of the upper half of the accumulator, or until the shift counter attempts to accumulate a total of more than ten.

When shifting and counting is ended by sensing a digit other than zero in the high-order position of the accumulator, the total accumulated in the shift counter is inserted in the two low-order positions of the accumulator, and the machine goes to the next instruction.

When shifting and counting is ended by the shift counter trying to accumulate a total of more than ten, the overflow circuit will be set, the overflow light will come on, and the quantity of ten will be inserted in the two low-order positions of the accumulator. The effect of this overflow condition on machine operation is dependent on the setting of the overflow switch on the console.

If a shift and count operation is called for and no shift takes place (digit other than zero in twentieth position), the two low-order digits of the accumulator contents will be replaced by zeros regardless of the units position of the data address.

If a shift and count operation is called for and only one shift is executed (digit other than zero in the nineteenth position), the units digit of the original accumulator contents will be replaced by zero.

The following are examples of shift left and count with data address whose units digit is zero:

Op Code		Data-Address	Accumulator Upper	Accumulator Lower
		Normal shift of five positions — count of five in units position		
36 SCT	*before:*	0000	0000012345	2222255555+
	after:		1234522222	5555500005+
		Maximum shift of ten positions — count of ten in two low-order positions		
36 SCT	*before:*	0000	0000000000	1234522222+
	after:		1234522222	0000000010+
		Overflow condition — count of ten in two low-order positions		
36 SCT	*before:*	0000	0000000000	0012345222+
	after:		0012345222 (overflow)	0000000010+
		No shift — two low-order digits replaced by zero		
36 SCT	*before:*	0000	1234567890	2222255555+
	after:		1234567890	2222255500+
		One shift — units digit of original contents replaced by zero		
36 SCT	*before:*	0000	0123456789	2222255555+
	after:		1234567892	2222555501+

Examples of shift left and count with data address whose units digit is other than zero:

Op Code		Data-Address	Accumulator Upper	Accumulator Lower
		Normal shift of four positions — count in units position is sum of tens complement of 6 and number of positions shifted.		
36 SCT	*before:*	0006	0000123456	2222255555+
	after:		1234562222	2555550008+
		Maximum shift of 6 — count in low-order digits is sum of tens complement of 6 and number of positions shifted.		
36 SCT	*before:*	0006	0000001234	2222255555+
	after:		1234222225	5555000010+
		Overflow condition — more than six shifts attempted — count of ten in low-order positions.		
36 SCT	*before:*	0006	0000000123	2222255555+
	after:		0123222225 (overflow)	5555000010+
		No shift — two low-order digits replaced by zero.		
36 SCT	*before:*	0006	1234567890	2222255555+
	after:		1234567890	2222255500+
		One shift — units digit of original contents replaced by zero.		
36 SCT	*before:*	0006	0123456789	2222255555+
	after:		1234567892	2222555505+

Table Lookup

The Type 650 is provided with an automatic table lookup operation. This permits a table reference to be performed in a minimum of time and with a minimum of programming effort. Before a TLU operation is performed, careful consideration must be given to the application itself. In some cases optimum results can be obtained from other programming techniques.

In order to understand thoroughly the description of table lookup given below, the terminology used is defined as follows:

Table — An arrangement in condensed form for ready reference of statistics, measures, etc.
Argument — The known reference factor necessary to find a desired item in a table.
Function — The unknown factor or factors within a table associated with a known reference factor (argument).

A table, therefore, will consist of a series of arguments (reference factors) arranged in a sequence of ascending absolute values. Associated with each argument will be one or more functions (results within the table).

Table arguments are stored on the drum in ascending sequence by their absolute values (argument signs are ignored). Therefore, it will be necessary to treat some tables containing arguments with different signs as two different tables. In some cases, only one set of arguments would be stored and two different locations referred to according to the sign of the search argument.

Arguments may be a maximum of 10 digits in size and are stored 48 to a band except for the last band of a table, which may contain less. The last two memory locations in each band (0048, 0049; 0098, 0099; etc.) cannot be used to store table arguments. They may be used to store functions instructions, etc., however. Table arguments should be stored in successive drum locations starting with the first word in a band (0000, 0050, 0100, etc.).

A table does not need to contain all the possible search arguments that will be encountered. A table lookup operation will cause a search until an equal or next higher condition occurs. Thus, when a search argument is not actually in the table, the search will stop on the next higher table argument. Interpolation can be accomplished by programming, if necessary.

The fact that the search stops on a next higher, as well as an equal, condition makes possible, in many cases, the location of both the table argument and the associated function in the same word. The argument must be stored in the most significant positions of the word. For example:

Table			Distributor Argument	
Argument	Function			
00001	00650			
00002	00638	←	00002	00000
00003	00600		Table argument selected as a next higher	

Function values also may be stored a fixed number of locations from their arguments. Thus, having found the location N of the argument, the function is located at N+C where C is the fixed separation of the functions from the arguments.

The D-address of the table lookup instruction is the location of the first argument in the table. If the D-address is other than the first address of a band, the search will begin at the first address of the band, and the number of locations searched to find an equal (or next higher) argument will be added to the D-address. This may be useful for short tables having less than one band of arguments as it saves modification to locate the function address. For example: the instruction 84 0020 xxxx is given for a 20-argument table whose arguments are located 0000-0019. If the argument at 0015 equals the given argument, xx 0035 xxxx will appear in the lower accumulator. If the function is stored 20 locations from the argument, the lower accumulator contains the address of the function with no further modification.

It is possible to store several short tables in one band and use the same D-address. For example: Assume three tables 15 arguments in length each having 5-digit arguments:

Drum Location	Table Arguments	
0000	00000XXXXX	Table 1
0001	00000XXXXX	
│	│	
0014	00000XXXXX	
0015	0000XXXXX0	Table 2
0016	0000XXXXX0	
│	│	
0029	0000XXXXX0	
0030	000XXXXX00	Table 3
0031	000XXXXX00	
│	│	
0044	000XXXXX00	

Searching will begin at 0000. Proper placement of the argument will cause the search to be effective only in the applicable section of the band. When this method is used, it is necessary for the first argument of each table to be greater in value than the last argument of the previous table.

The known argument must be placed in the distributor prior to the search operation. Throughout the table lookup operation, this argument is compared against the table arguments stored on the drum. When an equal argument or next higher (if no equal exists) is located, the address is placed in positions 8-5 of the lower half of the accumulator. As just mentioned, the absolute value (sign ignored) of the distributor argument is compared with the absolute value of the table arguments. When the search is complete, the known argument will be unaltered in the distributor. Positions 10-9 and 4-1 of the lower half of the accumulator will also be unaltered.

It will be necessary to modify the argument address in positions 8-5 of the lower accumulator in order to locate the function except for the two cases mentioned.

1. Argument and function stored in the same word
2. Address of function compiled by giving D-address other than the first in the band.

It is particularly important that an end-of-table—end-of-search indication be incorporated into a table search. Several methods of indicating end of search are possible. The most frequently used method is to insert an argument of 99 . . . . 9 as the last argument of the table to terminate TLU.

If no end-of-table indication is provided, a search argument larger than any of the table arguments will cause the machine to continue searching until it encounters a location that contains a number equal to, or greater than, the search argument. Under this condition, the address of the location on which the search stopped would be used as the address obtained from the TLU operation. If no equal or greater number is encountered, the machine will stop when it attempts to search location 2000 (an invalid address).

If an end-of-table indication is undesirable, it is possible to compare the known argument to the last argument in the table. In this manner, it can be assured that the known argument falls within the table range.

It is possible with the Type 650 to achieve the results of a table lookup operation without actually using the table lookup code (84). This type of operation is feasible if the arguments to be used are four-digit numbers that fall in a block between the limits of 0000 and 1999 or 0000 and 0999 (or can be modified in some manner to meet these conditions).

One application that might lend itself to this method of programming is cost distribution by department number. If it is assumed that department numbers run from 0001 to 0200, it is then possible to assign magnetic drum location 0001 to department 0001, location 0002 to department 0002, etc. When department number which had been read into the machine from a card is used as a data address, it is possible to locate the associated magnetic drum storage location without doing a table lookup.

This is one method of doing table lookup without using code 84. Modifications of this basic approach are possible but are dependent upon the type of information processed.

84 TLU (Table Lookup)

This operation code performs an automatic table lookup using the D-address as the location of the first table argument and the I-address as the address of the next instruction to be executed. The argument for which a search is to be made must be in the distributor. The address of the table argument equal to, or higher than (if no equal exists) the argument given is placed in positions 8-5 of the lower accumulator. The search argument remains, unaltered, in the distributor.

Op Code		Data-Address	Accumulator Upper	Accumulator Lower	Distributor
84 TLU	*before:*	0200	0000000000	65 0000 0554+	8365828300+
	after:		0000000000	65 (0240) 0554+	8365828300+
84 TLU	*before:*	0215	0000012345	65 0000 0554+	8365828300+
	after:		0000012345	65 (0255) 0554+	8365828300+

	Drum Location	Contents	
	0200	8365824300	
	0201	8365824400	
	0239	8365828200	
Drum Location of→	0240	8365828300	←Table Argument Equals
Equal Argument	0241	8365828400	Search Argument
	0247	8365830000	

Miscellaneous

00 No-Op (No operation)

This code performs no operation.

The data address is bypassed, and the machine automatically refers to the location specified by the instruction address of the No-Op instruction. This code may therefore be used for switching the path of the program. It should be noted that the data address must be a valid Type 650 address to prevent a storage selection error.

01 Stop

This operation code causes the program to stop provided the *programmed* switch on the control console is in the STOP position. When the *programmed* switch is in the RUN position, the 01 code will be ignored and treated in the same manner as 00 (NO-OP). The programmed switch is discussed in detail in the *Console Section.*

Program Example

The following is a sample program such as might be used for inventory calculations (Figure 38). It is presented here to display the use of a major portion of the operation codes in an actual program.

	STORAGE ENTRY WORDS 1	2	3	4	5	6	7	8	9	10
Memory Address	1801	1802	1803	1804	1805					
Issues	PART NO.	QUAN. ISSUED		QUAN. ISSUED	IDENT. X					
Input Card	XXXXX00000	XXX0000000		0000000XXX	0000000098					
Receipts	PART NO.	QUAN. RECV'D	UNIT COST	QUAN. RECV'D	IDENT. X					
Input Card	XXXXX00000	XXX0000000	0000000X.XX	0000000XXX	0000000089					
Memory Address										
Output Card										

Location of Instruction	Instruction OP	Data	Instr.	Operation Abbrev.	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
0000	70	1801	0001	RD				
0001	65	0300	0002	RAL	0————0	6000000006	6000000006	ADD CONSTANT INSTRUCTION
0002	69	1801	0003	LD	0————0	6000000006	PART NO. XXXXX00000	LOAD DIST. WITH PART NUMBER
0003	84	0400	0004	TLU	0————0	60XXXX0006	XXXXX00000	SEARCH TABLE STARTING IN LOC. 0400
0004	69	8003	0005	LD	0————0	60XXXX0006	0000000000	LOAD DIST. WITH ZEROS
0005	22	0100	8002	STDA	0————0	60XXXX0006	00XXXX0000	STORE TLU ADDRESS FOR FUTURE USE
8002	60	XXXX	0006	RAU	PART NO. XXXXX00 AVE COST XXX	0————0	PART NO. XXXXX00 AVE COST XXX	ADD TABLE WORD INTO UPPER
0006	11	1801	0007	SU	(SHOULD ZERO) 00XXX	0————0	PART NO. XXXXX00000	SUBTRACT ACTUAL PART NO. FROM TABLE PART NO.
0007	30	0003	0008	SRT	000 (ZERO?) 00	XXX0———0	XXXXX00000	SHIFT AVERAGE UNIT COST INTO LOWER
0008	44	0009	0010	BRNZU	000?????00	XXX0———0	XXXXX00000	BRANCH IF PART NOS. DO NOT AGREE
0009	01	——	——	STOP	000XXXXX00	XXX0———0	XXXXX00000	IMPROPER PART NO. OR IDENTIFYING X PUNCH
0010	35	0003	0011	SLT	0———0XXX	0————0	XXXXX00000	SHIFT AVERAGE UNIT COST TO UPPER
0011	69	1805	0012	LD	0———0XXX	0————0	0———0 9/8 9/8	LOAD DIST. WITH IDENTIFYING EIGHT
0012	91	0014	0013	BRD1	0———0XXX	0————0	0———098	TEST FIRST POS. OF DISTRIBUTOR FOR EIGHT
0013	92	0022	0009	BRD2	0———0XXX	0————0	0———089	TEST SECOND POS. OF DISTRIBUTOR FOR EIGHT
0014	19	1804	0015	MULT	0————0	0000 AMOUNT XXXXXX	0———0 QUAN. XXX	EXTEND QUAN. ISSUED TIMES AVE. UNIT COST
0015	15	1802	0016	AL	0————0	QUAN. XXX 0 AMOUNT XXXXXX	QUAN. XXX 0———0	INSERT QUAN. ISSUED TO LEFT OF AMOUNT
0016	20	0102	0017	STL	0————0	QUAN. XXX 0 AMOUNT XXXXXX	QUAN. XXX 0 AMOUNT XXXXXX	STORE QUAN. ISSUED & AMT. IN TEMP. STORAGE
0017	65	0302	0018	RAL	0————0	1005000019	1005000019	ADD CONSTANT INSTRUCTION
0018	15	0100	8002	AL	0————0	10XXXX0019	00XXXX0000	MODIFY DATA ADDRESS BY TLU ADDRESS
8002	10	XXXX	0019	AU	XXX0XXXXXX	10XXXX0019	XXX0XXXXXX	ADD QUAN. & AMT. FROM TABLE
0019	11	0102	0020	SU	XXX0XXXXXX	10XXXX0019	XXX0XXXXXX	SUBTRACT QUAN. ISSUED & AMOUNT
0020	69	0303	0021	LD	XXX0XXXXXX	10XXXX0019	2100000000	LOAD DIST. WITH CONSTANT INSTRUCTION
0021	22	0103	8001	STDA	XXX0XXXXXX	10XXXX0019	21XXXX0000	INSERT DATA ADDRESS FROM LOWER
8001	21	XXXX	0000	STU	XXX0XXXXXX	10XXXX0019	XXX0XXXXXX	STORE NEW QUAN. & AMT. IN TABLE
0022	60	1803	0023	RAU	0———0 COST XXX	0————0	0———0 COST XXX	ADD RECEIPTS UNIT COST
0023	19	1804	0024	MULT	0————0	0——0 AMOUNT XXXXXX	0———0 QUAN. XXX	EXTEND QUAN. RECEIVED TIMES UNIT COST
0024	15	1802	0025	AL	0————0	QUAN. XXX 0 AMOUNT XXXXXX	QUAN. XXX 0———0	INSERT QUAN. RECEIVED TO LEFT OF AMOUNT
0025	10	0304	0026	AU	1505000027	QUAN. XXX 0 AMOUNT XXXXXX	1505000027	ADD CONSTANT INSTRUCTION TO UPPER
0026	10	0100	8003	AU	15XXXX0027	QUAN. XXX 0 AMOUNT XXXXXX	00XXXX0000	MODIFY DATA ADDRESS BY TLU ADDRESS
8003	15	XXXX	0027	AL	15XXXX0027	QUAN. XXX 0 AMOUNT XXXXXX	QUAN. XXX 0 AMOUNT XXXXXX	ADD TABLE QUAN. & AMT. TO LOWER
0027	10	0305	8003	AU	0500000001 / 20XXXX0028	QUAN. XXX 0 AMOUNT XXXXXX	0500000001	MODIFY INSTRUCTION IN UPPER
8003	20	XXXX	0028	STL	20XXXX0028	QUAN. XXX 0 AMOUNT XXXXXX	QUAN. XXX 0 AMOUNT XXXXXX	STORE PREVIOUS BAL. + RECEIPTS IN TABLE
0028	65	8002	0029	RAL	0————0	QUAN. XXX 0 AMOUNT XXXXXX	QUAN. XXX 0 AMOUNT XXXXXX	CLEAR UPPER
0029	35	0003	0030	SLT	0———0 QUAN. XXX	0 AMOUNT XXXXXX 000	QUAN. XXX 0 AMOUNT XXXXXX	SHIFT QUANTITY TO UPPER
0030	21	0104	0031	STU	0———0 QUAN. XXX	0 AMOUNT XXXXXX 000	0———0 QUAN. XXX	STORE QUANTITY IN TEMP. STORAGE
0031	65	8002	0032	RAL	0————0	0 AMOUNT XXXXXX 000	0 AMOUNT XXXXXX 000	CLEAR UPPER
0032	64	0104	0033	DIVRU	0————0	0000 AVE. COST XXXXXX	0———0 QUAN. XXX	DIVIDE AMOUNT BY QUANTITY
0033	31	0003	0034	SRD	0————0	0———0 AVE. XXX	0———0 QUAN. XXX	SHIFT AND ROUND TO THREE POSITIONS
0034	15	1801	0035	AL	0————0	PART NO. XXXXX 00 COST XXX	PART NO. XXXXX00000	INSERT PART NO. TO LEFT OF AVE. COST
0035	10	0301	0036	AU	2000000000	PART NO. XXXXX 00 AVE. COST XXX	2000000000	ADD CONSTANT INSTRUCTION TO UPPER
0036	10	0100	8003	AU	20XXXX0000	PART NO. XXXXX 00 AVE COST XXX	00XXXX0000	MODIFY DATA ADDRESS BY TLU ADDRESS
8003	20	XXXX	0000	STL	20XXXX0000	PART NO. XXXXX 00 AVE COST XXX	PART NO. XXXXX 00 AVE COST XXX	STORE PART NO. AND NEW AVE. COST IN TABLE

CONSTANTS:
- 0300 — 60 0000 0006
- 0301 — 20 0000 0000
- 0302 — 10 0500 0019
- 0303 — 21 0000 0000
- 0304 — 15 0500 0027
- 0305 — 05 0000 0001

TEMPORARY STORAGE:
- 0100 — TLU ADDRESS
- 0102 — QUAN. & AMT. (ISSUES)
- 0103 — 21XXXX 0000 (NEVER USED)
- 0104 — NEW QUANTITY

FIGURE 38. INVENTORY CALCULATIONS

LOAD CARD

An input card which contains a 12 punch in any specific column may be identified as a *load* card. The identification of a load card is accomplished by control panel wiring as the card passes the first reading station. The control panel wiring is discussed in the control panel section.

As a load card passes the second reading station, control panel wiring for storage entry is completely bypassed. Internal machine wiring automatically reads the entire 80 columns of the card into words 1-8 of read buffer storage. Columns 1-10 enter word 1, columns 11-20 enter word 2, etc. Zeros are entered into words 9 and 10 automatically.

Another important difference in the machine function on a load card is in the analysis of the read instruction. After the operation code has been analyzed and the load card information transferred from buffer to general storage, the machine will obtain its next instruction from the location specified by the D-address instead of the I-address.

Although the primary use of load cards is to load instructions, tables, etc., onto the drum, it is important to realize that a load card may be used at any time. The use of a load-type card during production runs automatically effects a branching function because the address of the next instruction is specified by the D-address. Thus, one type of card identification branching is accomplished with no programming necessary.

Load cards must have all ten columns punched in any words that will be used within the Type 650. In addition, these words must have a 12- or an 11-punch over the units position to indicate a positive or negative sign.

Figure 39 shows how a load card might be used to cause a new deck of inventory status cards to be punched from the results of the calculations in the programming example shown in Figure 38.

A load card, punched with 0000000037 in columns 1-10 and placed behind the last card in the activity deck being processed, would automatically cause the program to branch to location 1801 after the load card has been read. In location 1801 the card has just entered a NO-OP instruction with an I-address of 0037. This sends the program to location 0037 for its next instruction.

	STORAGE ENTRY WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address	1801									
Load Card	0000000037								▨	▨
Input Card										

	STORAGE EXIT WORDS									
	1	2	3	4	5	6	7	8	9	10
Memory Address	1827	1828								
Output Card	PART # & AVE. COST	QUAN. & AMOUNT								

Location of Instruction	Instruction			Operation Abbrev.	8003 Upper Accumulator	8002 Lower Accumulator	8001 Distributor	Remarks
	OP	Data	Instr.					
1801	00	0000	0037	NO-OP				BRANCH ON LOAD CARD
0037	65	0306	8002	RAL	0————0	1004000038	1004000038	RESET ADD CONSTANT
8002	10	XXXX	0038	AU	$\overset{PART}{XXX}\overset{NO.}{XX}00\overset{COST}{XXX}$	10XXXX0038	$\overset{PART}{XXX}\overset{NO.}{XX}00\overset{COST}{XXX}$	ADD SEL. WORD FROM TABLE (PART # & AVE. COST)
0038	44	0039	0053	BRNZU	$\overset{PART}{XXX}\overset{NO.}{XX}00\overset{COST}{XXX}$	10XXXX0038	$\overset{PART}{XXX}\overset{NO.}{XX}00\overset{COST}{XXX}$	BRANCH IF PART NUMBER PRESENT
0039	24	1827	0040	STD	$\overset{PART}{XXX}\overset{NO.}{XX}00\overset{COST}{XXX}$	10XXXX0038	$\overset{PART}{XXX}\overset{NO.}{XX}00\overset{COST}{XXX}$	STORE PART # & AVE. COST FOR PUNCHING
0040	65	8002	0041	RAL	0————0	10XXXX0038	10XXXX0038	CLEAR UPPER
0041	15	0307	8002	AL	0————0	$10\overset{+0}{X}\overset{5}{X}\overset{0}{X}\overset{0}{X}\overset{0}{0}\overset{0}{0}\overset{0}{4}\overset{4}{2}$	0005000004	MODIFY INSTRUCTION IN LOWER
8002	10	$\overset{XXXX}{+0500}$	0042	AU	$\overset{QUAN.}{XXX}0\overset{AMOUNT}{XXXXX}X$	10XXXX0042	$\overset{QUAN.}{XXX}0\overset{AMOUNT}{XXXXX}X$	ADD SEL. WORD FROM TABLE (QUAN. & VALUE)
0042	24	1828	0043	STD	$\overset{QUAN.}{XXX}0\overset{AMOUNT}{XXXXX}X$	10XXXX0042	$\overset{QUAN.}{XXX}0\overset{AMOUNT}{XXXXX}X$	STORE QUAN. & VALUE FOR PUNCHING
0043	71	1827	0044	PCH				PUNCH A CARD
0044	65	8002	0045	RAL	0————0	10XXXX0042	10XXXX0042	CLEAR UPPER
0045	69	8003	0046	LD	0————0	10XXXX0042	0————0	LOAD DISTRIBUTOR WITH ZEROS
0046	22	0103	0047	STDA	0————0	10XXXX0042	00XXXX0000	EXTRACT DATA ADDRESS FROM LOWER
0047	10	8001	0048	AU	00XXXX0000	10XXXX0042	00XXXX0000	ADD DATA ADDRESS INTO UPPER
0048	11	0308	0049	SU	$00\overset{-1}{X}\overset{3}{X}\overset{9}{X}\overset{9}{X}0000$	10XXXX0042	0013990000	SUBTRACT LAST POSSIBLE LOCATION
0049	44	0050	0052	BRNZU	$00\overset{-1}{X}\overset{3}{X}\overset{9}{X}\overset{9}{X}0000$	10XXXX0042	0013990000	BRANCH IF NOT LAST POSSIBLE LOCATION
0050	65	8002	0051	RAL	0————0	10XXXX0042	10XXXX0042	CLEAR UPPER
0051	16	0309	8002	SL	0————0	$10\overset{-0}{X}\overset{4}{X}\overset{9}{X}\overset{9}{X}\overset{0}{0}\overset{0}{0}\overset{0}{3}\overset{4}{8}$	0004990004	CREATE INSTRUCTION TO ADD NEXT PART #
0052	01	0001	—	STOP				STOP-LAST POSSIBLE LOC. OR LAST SIG. PART #
0053	16	0310	0054	SL	0————0	00XXXX0000	1008990038	SUBTRACT LAST LOCATION
0054	45	0055	0052	BRNZ	0————0	00????0000	1008990038	BRANCH IF NOT LAST LOCATION
0055	15	0311	8002	AL	0————0	10XXXX0038	1009000038	ADD CONSTANT TO CREATE NEW INST.
								0306 - 1004000038
								0307 - 0005000004
								0308 - 0013990000
								0309 - 0004990004
								0310 - 1008990038
								0311 - 1009000038

Figure 39. Branching by Using a Load Card

CONSOLE AND ITS OPERATION

The control console (Figure 40) located on the Type 650 Unit contains switches and lights through which the operator may observe and control the course of the problem being solved. The principal use of the console is for detecting errors in new programs and for trouble analysis. The console enables the operator to investigate each step of a program.

Power Control

The bank of buttons across the top of the console is used to control the power to the machine. Therefore, they are separated from the controls used in the actual machine operation (Figure 41).

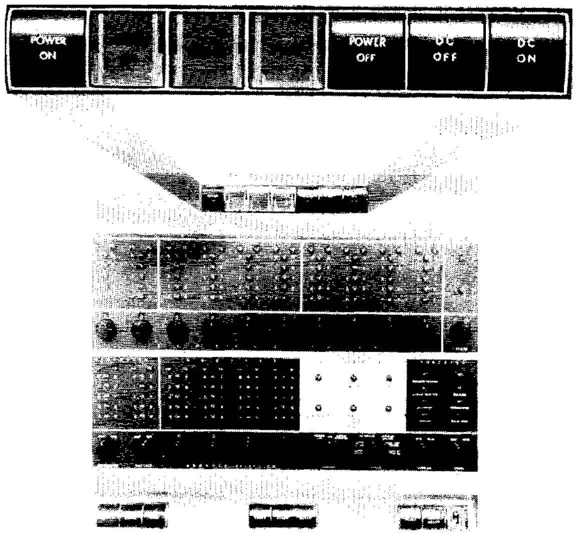


Figure 41

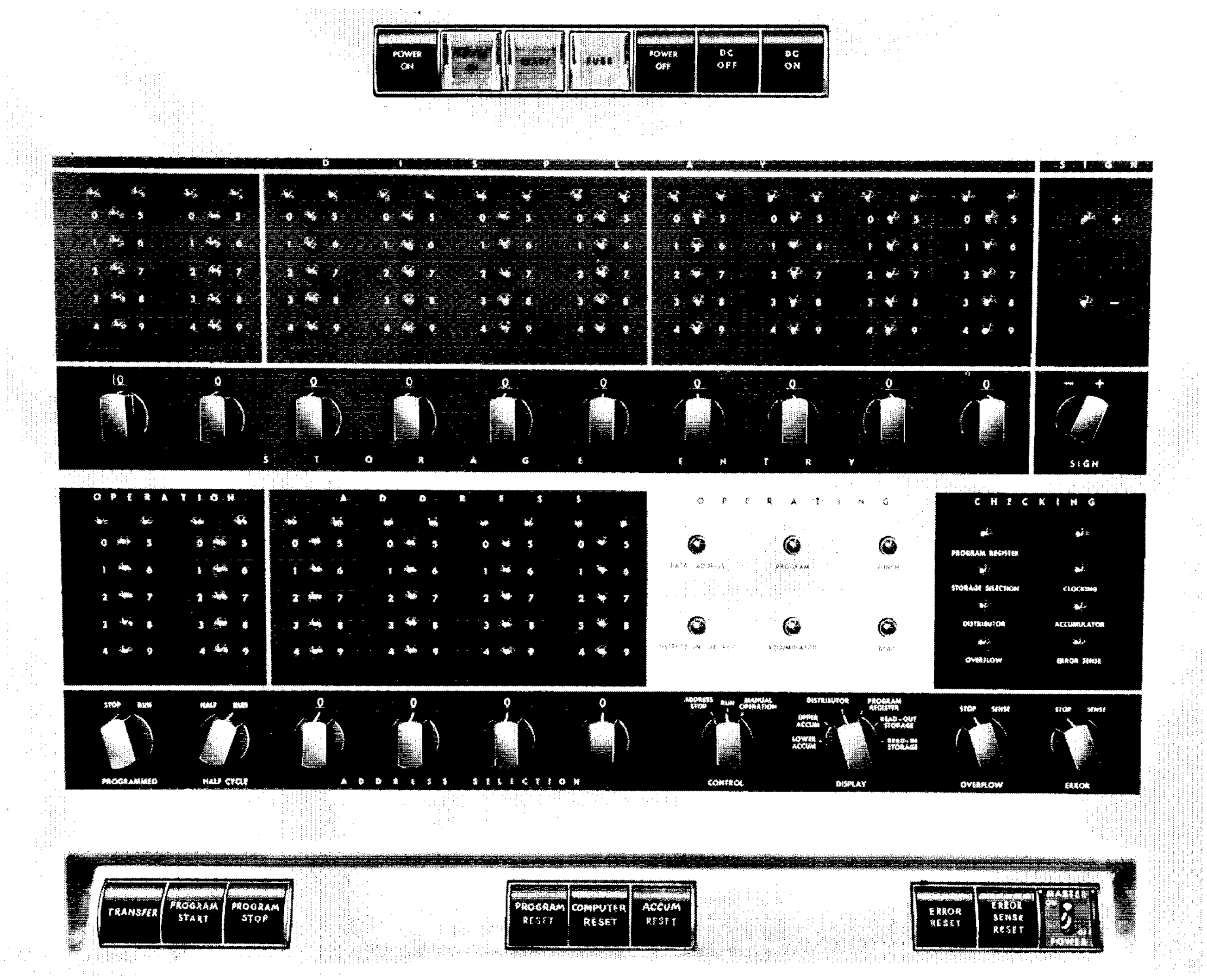


Figure 40. Type 650 Control Console

Power-On Button

Depressing this button has the following effects:

1. AC power is supplied to all parts of the machine.
2. The blower motors and drum drive motor are started.
3. DC power is supplied automatically after approximately a three-minute delay.

Power-On Light

This light goes on when the POWER-ON button is depressed and remains on as long as AC power is supplied to the machine.

Ready Light

This light comes on when DC power is turned on and remains on as long as DC power is supplied to the machine. DC power is automatically turned on approximately three minutes after the POWER-ON button is depressed. However, DC power may be turned off without turning the AC power off (see *DC OFF* and *DC ON*).

Fuse Light

This light indicates that a fuse has been blown in the Type 650 or Type 655.

Power-Off Button

When this button is depressed, the following sequence of events occurs:

1. DC power is turned off.
2. AC power (except that supplied to the blower motors) is turned off.
3. The blower motors are turned off after a five-minute delay.

DC-Off Button

This button is effective only after the POWER-ON button has been depressed, and the delay time for automatically turning on the DC power has elapsed. Depressing the DC-OFF button will cause all DC power to be turned off and the ready light will go out. The DC-OFF button has no effect on the AC power. When this button is used, the machine can be shut down for comparatively short periods (e.g., an hour) without necessitating cooling off and reheating the electronic tubes in the machine.

DC-On Button

The DC-ON button is effective only after the DC power has been turned off by the DC-OFF button. It supplies DC power to the machine components.

Display Lights

There are 10 digit positions and one sign position across the upper portion of the console. Each digit position is represented by seven lights (Figure 42).

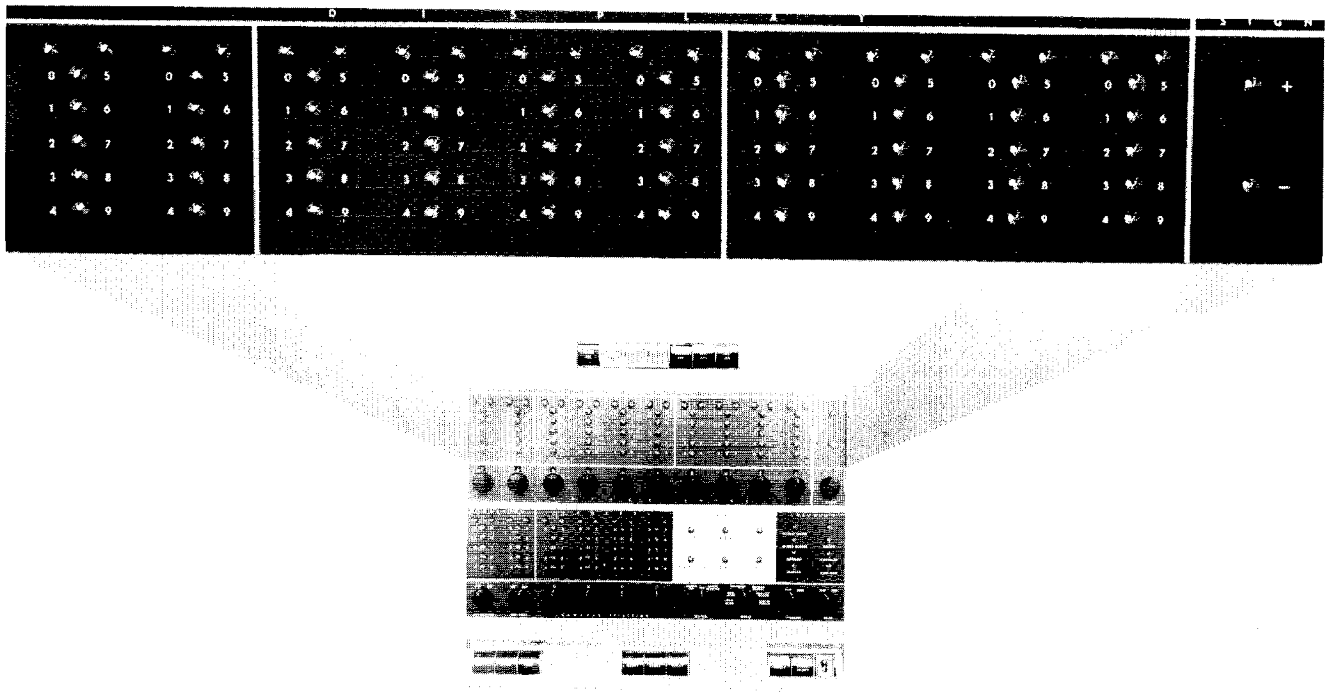


FIGURE 42

The horizontal pairs of lights are known as binary lights and represent the values zero and five. The five vertical lights are known as quinary lights and represent values of 0, 1, 2, 3, 4, or 5, 6, 7, 8, 9, depending upon the binary indication.

The sign of the factor is indicated by one of two sign lights. The upper sign light indicates a plus factor when lit, and the lower sign light indicates a minus factor when lit. Figure 43 shows how digits can be identified. The display lights are used with the display switch to display the contents of a specific location when the machine is stopped.

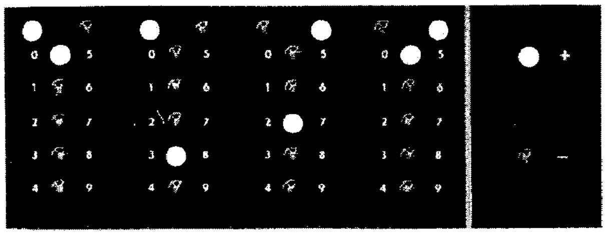


FIGURE 43. NUMBER REPRESENTED 0375+

Storage-Entry Switches

These ten digit switches and one sign switch (Figure 44) can be used in two ways. They can be referred to during the program by using an address of 8000. They can also be used in conjunction with other console buttons and switches to enter information into any drum memory location.

Operation Lights

During the data half-cycle, these two sets of lights (Figure 45) indicate which operation is to be performed. They are blank during the instruction half-cycle.

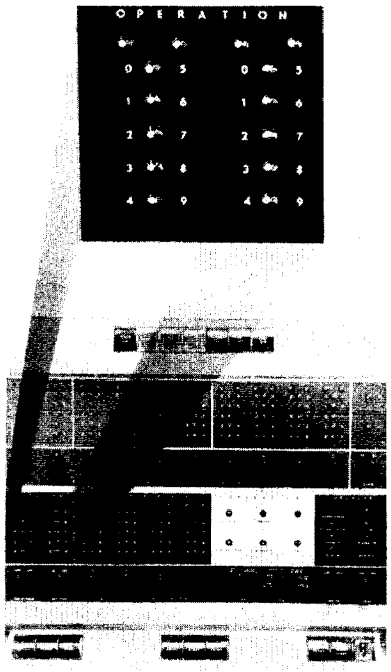


FIGURE 45

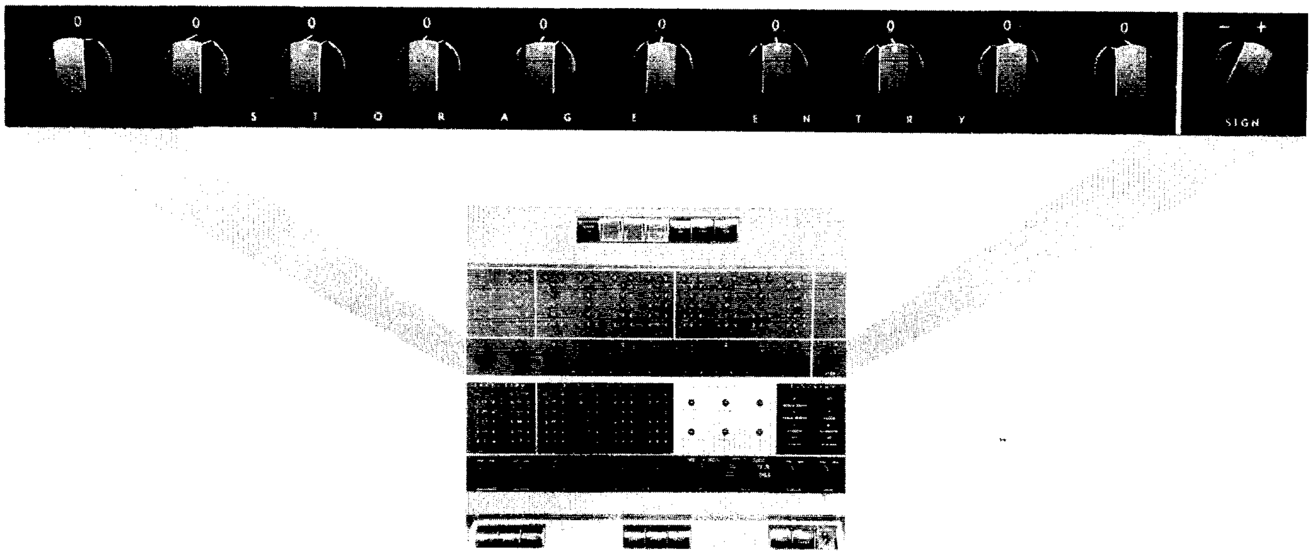


FIGURE 44

Address Lights

These four sets of lights (Figure 46) indicate the contents of the address register. If the data address light is on, the address shown is the data address and the D-half-cycle is ready to be executed. If the instruction address light is on, the address shown is the instruction address and the I-half-cycle is ready to be executed.

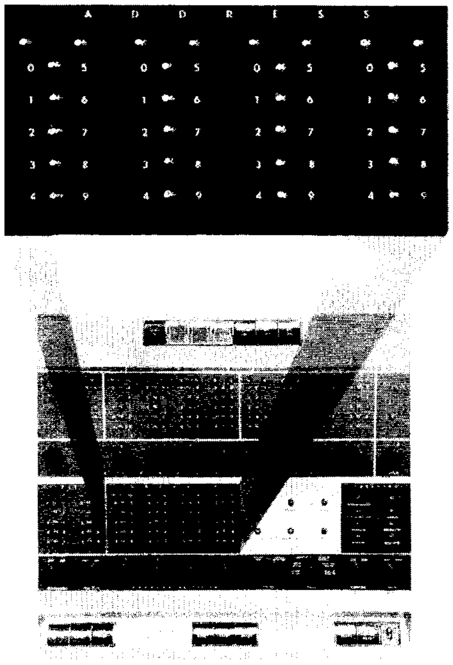


FIGURE 46

Operating Lights

These lights (Figure 47) indicate the operating status of the machine. The data-address and instruction-address lights indicate which half-cycle the control unit is executing. The remaining lights indicate the status of the read, punch, accumulator, and program units.

Data-Address Light

When the machine stops with this light on, the D-half-cycle is ready to be executed.

Instruction-Address light

When the machine stops with this light on, the I-half-cycle is ready to be executed.

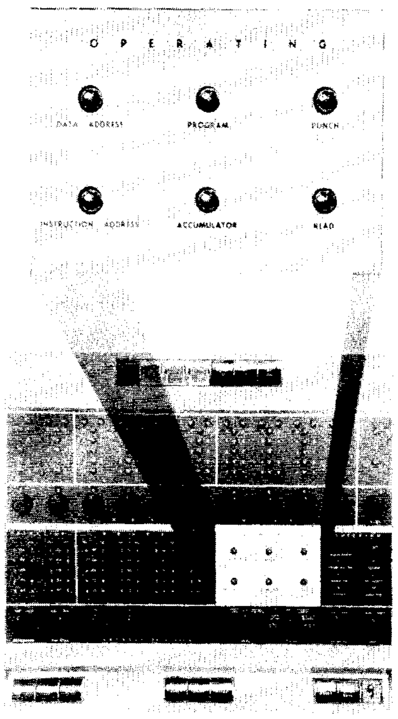


FIGURE 47

Program Light

This light will go on only if the machine stops for a programmed stop, manual stop, or address stop. It will not, however, be on if a manual stop is accomplished during the I-cycle of a READ or PUNCH instruction.

Accumulator Light

This light is on whenever the accumulator is in use.

Punch Light

This light is on during the D-cycle of a PUNCH instruction until the beginning of a punch-feed cycle. If the machine stops when this light is on, one of four conditions is indicated:

1. No cards in the punch hopper
2. A punch-feed failure
3. Cards have not been run into the punch feed
4. Either the read or punch-stop key has been depressed.

Read Light

This light is on during the D-cycle of a READ instruction until the beginning of a read-feed cycle. If the machine stops with this light on, it indicates one of four conditions:

1. No cards in the read hopper
2. A read-feed failure
3. Cards have not been run into the read feed
4. Either the read or punch-stop key has been depressed.

Checking Lights (Figure 48)

Program Register Light

This light indicates the detection of a validity error at the output of the program register.

Storage Selection Light

This light indicates errors of the following type:

1. A store operation with a data address of the 8000 series.
2. An invalid D- or I-address (other than 0000 through 1999 and 8000 through 8003).
3. Information is being written in two or more locations simultaneously.
4. Information is not written in any location on a store operation.
5. Attempted manual entry from storage entry switches to 8001, 8002, or 8003.

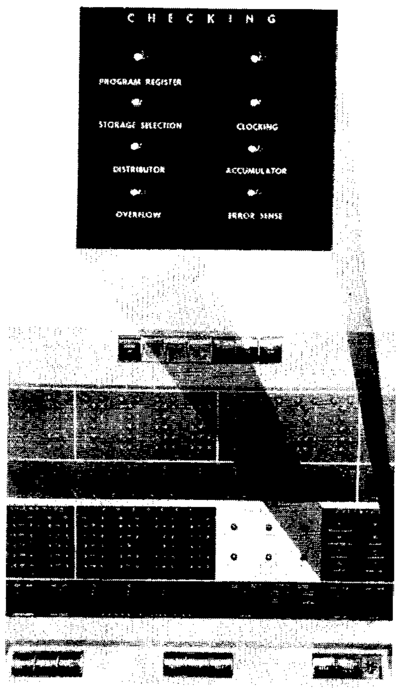


FIGURE 48

Distributor Light

This light will come on if a validity error is detected at the output of the distributor.

Overflow Light

This light will come on if an overflow condition is detected. An overflow condition can be caused by one of the following:

1. An excessive accumulation.
2. Trying to develop a quotient of more than ten (10) digits.
3. Trying to exceed the number of shifts called for in a *shift and count* operation.

Clocking Light

This light will come on if an error is detected in the clocking (timing) circuitry.

Accumulator Light

This light will come on if a validity error is detected at the output of the accumulator.

Error Sense Light

This light is operative when the error sense switch is in the SENSE position and will come on when one of the following errors has been detected:

1. Validity error
   a. Program register
   b. Accumulator
   c. Distributor
2. Clocking (timing) error

This light will remain on until reset by the operator.

Unlabeled Light

The one unlabeled light in the checking lights is completely inoperative and is of no importance to machine control.

Machine Stops (Not Indicated)

There are conditions that will cause the machine to stop with none of the checking lights on. The most common causes of such stops are:

1. A programmed stop (OP CODE 01). See *Operation Code.*

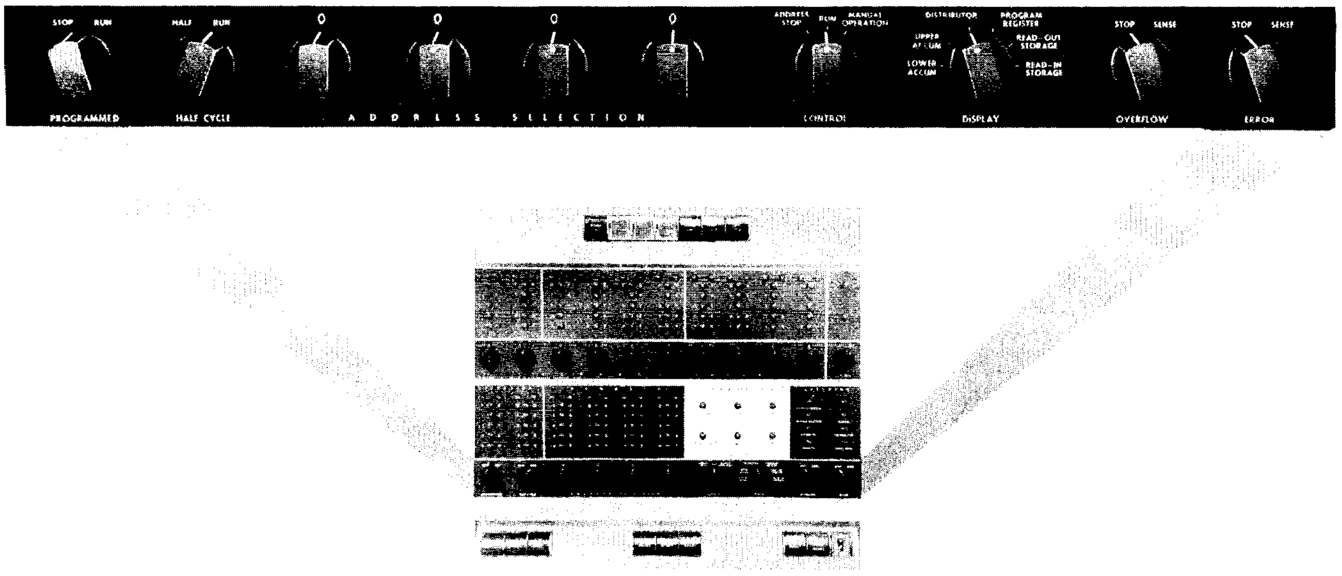


FIGURE 49

2. An impossible operation code in the operation register.
3. An attempted BRD (90-99) operation in which the position tested does not contain either an 8 or 9. See *Operation Code.*
4. A control signal failure.
5. A table lookup failure.

Switch Controls (Figure 49)

Programmed Switch

When this switch is in the RUN position, the program will bypass any operation codes calling for a program stop (code 01). The machine will treat the 01 code as a no-operation code (NO-OP 00).

When this switch is in the STOP position, programming will be stopped whenever a stop code (01) is encountered.

Half-Cycle Switch

When this switch is in the RUN position, the program will proceed automatically once it is started.

When this switch is in the HALF position, the machine will execute one-half of an instruction for each depression of the program start key provided the control switch is in the RUN or ADDRESS STOP positions.

With an operation code in the operation register and a data address in the address register, a depression of the program start key will cause the operation to be performed and place the address of the next instruction in the address register. This is the DATA or D-half-cycle of the instruction. The next depression of the program start key will cause the next instruction to be located and placed in the program, operation, and address registers. This is the INSTRUCTION or I-half-cycle of the instruction.

Address Selection Switches

These four switches are used to set up an address that may be used in one of the following ways:

1. Address at which program is to be stopped.
2. Address to be entered into address register for storage read-in or storage read-out.
3. Address to be entered into address register for starting program.

Control Switch

When this switch is in the ADDRESS STOP position, the program proceeds until the address in the address selection switches is identical to the address in the address register. At this time, the machine stops.

When this switch is in the RUN position, the program proceeds automatically once started.

When this switch is in the MANUAL position, the console may be used to read information into any addressable drum memory location or to display the contents of any addressable drum memory location.

The MANUAL position of this switch is used when entering an address from the address selection switches in order to start the program at a specific address.

Display Switch

When this switch is positioned for lower accumulator, upper accumulator, distributor, or program register, the contents of the corresponding location will be automatically displayed in the display lights.

The contents of any addressable storage location may be examined by use of the READ-OUT STORAGE position of the switch, in conjunction with the address selection switches. Any word may be entered into any drum memory location by use of the READ-IN STORAGE position of the switch, again in conjunction with the address-selection switches. The control switch must be in the MANUAL position to perform either of these operations. The machine will not operate with the control switch in the RUN or ADDRESS-STOP positions if the display switch is in the READ-OUT STORAGE or READ-IN STORAGE positions.

The setting of this switch may not be changed while the program is running. Changing the setting will upset factors in transmission.

Overflow Switch

When the overflow switch is set to STOP, the machine will stop whenever an overflow is detected.

When the overflow switch is set to SENSE, the overflow will be indicated but the machine will not stop *except on a quotient overflow.* The overflow indication may be interrogated by using an operation code (BROV 47) to determine automatically whether an overflow has occurred. The branch-on-overflow operation is discussed in detail in the section *Operation Codes,* page 39.

Error Switch

When the error switch is set to STOP, the machine will stop under any of the following conditions:

1. Validity error
   a. Program register
   b. Accumulator
   c. Distributor
2. Clocking (timing) error

When the error switch is set to SENSE, detection of any of the foregoing errors will have the following effect:

1. The error sense light will come on.
2. The program register, distributor, and accumulator will be reset to zeros.
3. The address register will be reset to 8000, which is the address of the storage entry switches.
4. The next instruction is then taken from the storage entry switches. With the error switch set to SENSE, it is possible to use a correction routine to recalculate part or all of the problem in case an error occurs. The error sense light will remain on to indicate to the operator that an error has occurred.

Key Controls (Figure 50)

Transfer Key

The transfer key functions only when the control switch is in the MANUAL position. When the transfer key is depressed, it will cause the address set in the address-selection switches to be transferred into the address register.

Program Start Key

When the program start key is depressed, the machine will start executing the program with the instruction located at the address specified by the address register.

The program start key is effective in this manner only when the control switch is in the RUN or ADDRESS-STOP positions.

The program start key is also depressed as the final step of a READ-IN or READ-OUT STORAGE operation. Under these conditions, the control switch is set to MANUAL.

Program Stop Key

When the program stop key is depressed, the machine will stop at the completion of the particular half-cycle of the instruction on which it was depressed.

Program Reset Key

When the program reset key is depressed, the program register is reset to zeros. Error circuits that have been activated by a program register validity check, a storage selection error, or a clocking error are reset by the program reset key.

When the control switch is in the MANUAL position, a depression of the program reset key inserts blanks in the operation register and address register.

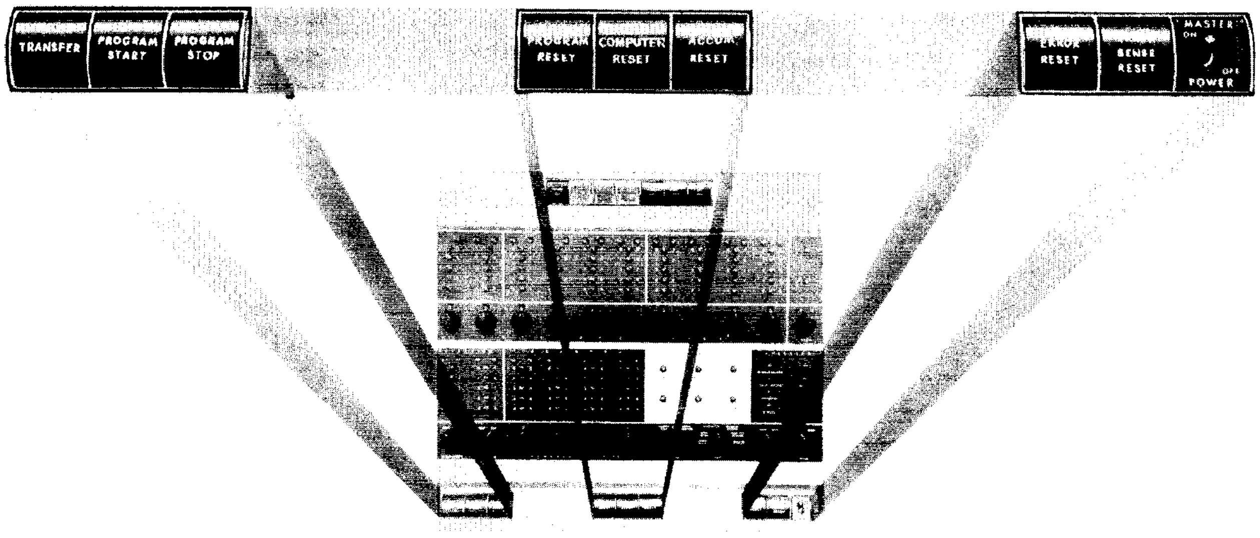


FIGURE 50

When the control switch is in the RUN or ADDRESS-STOP position, a depression of the program-reset key inserts blanks in the operation register and the address of the console switches (8000) in the address register.

Computer-Reset Key

When the computer-reset key is depressed, the program register, the distributor, and the entire accumulator are reset to zeros. This key resets *all* error circuits. It should be noted that the depression of this key will destroy the contents of the distributor, the accumulator, and the program register.

When the control switch is in the MANUAL position, a depression of the computer reset key inserts blanks in the operation register and address register.

When the control switch is in the RUN or ADDRESS-STOP position, a depression of the computer-reset key inserts blanks in the operation register and the address of the console switches (8000) in the address register.

Accumulator-Reset Key

When the accumulator-reset key is depressed, the distributor and the entire accumulator are reset to zeros. Error circuits that have been activated by an overflow condition, a validity check in the accumulator or distributor, a clocking error, or a storage selection error other than that caused by the detection of an invalid address, are reset by the accumulator-reset key. Depression of this key destroys the contents of the distributor and the accumulator. It should be noted that a depression of this key does *not* alter the contents of the program register, operation register, or address register.

Error-Reset Key

When the error reset key is depressed, the error circuits activated by a clocking error or a storage selection error, other than that caused by the detection of an invalid address, are reset.

Errror-Sense Reset Key

When the error-sense reset key is depressed, the error-sense circuit is reset and the error-sense light is turned out. This key is effective only when the error switch is in the SENSE position.

Master Power Switch

The master power switch is for *emergency use only*. When this switch is depressed, all power, including that to the blower motors, is removed from the machine *immediately*.

The depression of this switch also activates a locking device which makes it *impossible* for the operator to turn the machine back on. A customer engineer *must* be called in order to get the machine back in operation.

EXAMPLES OF USE OF THE CONTROL CONSOLE

Display Memory

When checking-out a program, it is often found desirable to examine the contents of a particular storage location. The inspection of a storage location can be accomplished as follows:

1. Stop the program.
2. Set the address of the location to be examined in the address selection switches.
3. Set the control switch to MANUAL.
4. Set the display switch to READ-OUT STORAGE.
5. Depress the program reset key.
6. Depress the transfer key.
7. Depress the program start key.

The contents of the desired storage location will be indicated in the display lights. *Manual read-out is through the distributor, which causes the previous contents of the distributor to be lost.*

When the program is stopped and the display switch is in the distributor, program register, lower accumulator, or upper accumulator positions, the contents of the selected unit are automatically exhibited in the display lights. If the upper accumulator is displayed following a divide without reset operation and before a reset or multiply operation is performed, the sign displayed will be the sign of the remainder.

Enter Information

By means of the control console new instructions and data can be inserted into a program that is already on the magnetic drum. The steps necessary to enter a word manually into some drum memory location are:

1. Stop the program.
2. Set the word to be entered in the storage-entry switches.
3. Set the address of the desired location in the address-selection switches.
4. Set the control switch to MANUAL.
5. Set the display switch to READ-IN STORAGE.
6. Depress the program reset key. The computer-reset key could be used, but the contents of the accumulator would be lost.
7. Depress the transfer key.
8. Depress the program start key.

The word set in the storage entry switches will then enter the location specified in the address selection switches. The word will appear in the display lights. The word that is entered passes through the distributor, replacing whatever may have previously been contained in the distributor. To verify that the manual entry was performed properly, the location in which the word was entered should be inspected. Verification can be accomplished by moving the display switch to READ-OUT STORAGE and performing the following:

1. Depress the program reset key.
2. Depress the transfer key.
3. Depress the program start key.

The address-selection switches and the READ-IN STORAGE position of the display switch *cannot* be used to enter a word directly into the *distributor* or *accumulator* from the storage-entry switches.

A word can be placed in the distributor from the storage-entry switches by entering the desired word into some unused drum memory location using the method described. Because the word entering the drum memory location must pass through the distributor, it is retained in the distributor as desired.

A different method must be used to place a factor in the accumulator. Any method that is used to place a word manually in the accumulator requires at least two steps. One method of placing a word in the accumulator is:

1. Enter the desired factor into some unused drum memory location.
2. The factor in the distributor can then be added to or subtracted from the accumulator contents in the following manner:
   a. Set the desired instruction in the storage-entry switches.
   b. Set the control switch to RUN.
   c. Depress program-reset key.
   d. Depress program-start key.

The I-address of the instruction in the storage-entry switches could be the address of the first instruction to follow.

If it is desirable to inspect the effect of the alteration factor on the contents of the accumulator, the half-cycle switch should be set to HALF.

Another method of placing a word in the accumulator is:

1. Set the desired add or subtract instruction (with a D-address of 8000) in the storage-entry switches.
2. Set control switch to RUN.
3. Set half-cycle switch to HALF.
4. Depress program-start key *one time.*
5. Set desired factor in the storage-entry switches.
6. Depress program-start key.

This will enter the factor into the accumulator; and if the proper I-address was used in the instruction in step 1, the program can continue from this point.

When a factor is entered into drum memory from the console, it may be desirable to save the contents of the accumulator and/or the distributor. The information in the accumulator may be retained by using the program-reset key. The information in the distributor will be lost when the factor being entered passes through. For this reason the contents of the distributor must be noted by the operator and reloaded by console operation.

Half-Cycle

The half-cycle switch is an excellent aid that can be used by the programmer in checking out a new routine. When this switch is set in the HALF position, it can be used in conjunction with other console controls to check the results of each operation before and after execution.

When the half-cycle switch is set in the HALF position, it can be used to check out the program in the following manner. Let us assume that the operation code ADD LOWER (15) is in the operation register, and that 0100 is in the address register. Under these conditions, the operation is ready to be performed. The first depression of the program-start key will cause the machine to perform the operation (D-half-cycle). At this time, it is now possible to use the display switch and check the following units:

1. *Distributor* — this unit would be checked to see that the proper information had been called in from storage (location 0100).
2. *Accumulator* — this unit would be checked to see that the operation had produced the correct result.
3. *Program register* — this unit will still contain the entire instruction.

Simultaneously with the execution of the operation, the operation register is reset to blanks, and the data address in the address register is replaced by the instruction address from the program register.

The second depression of the program-start key will cause the next instruction which is to be executed to enter the program register (I-half-cycle). It is now possible to use the display switch to check the program register. This unit would be checked to see that the next instruction to be executed has been properly chosen and placed in the program register.

The foregoing procedure allows the programmer to check all the individual instructions in a particular block where an error in programming occurs.

Address Stop

It is possible to stop the program at a specific point by using the control switch in conjunction with the address selection switches. This type of stop can be accomplished in the following manner:

1. Set control switch to ADDRESS STOP position.
2. Set desired address in address selection switches.
3. Depress program-start button.

With the switches set in this manner, the machine will stop every time the address in the address register equals the address set in the address selection switches. The stop will occur on both D- and I-addresses.

Depressing the program-start key after such a stop will cause the machine to resume computing from the point at which it was stopped. This feature is useful in program checking when it is desirable to proceed at full speed to a given instruction in the program.

Programmed Stop

The programmed switch makes it possible to ignore stop codes in the program. If the switch is set to STOP, an 01 operation code will stop the machine. If the switch is set to RUN, an 01 code will be treated as an 00 (NO OP) code and the program will proceed. Thus, for testing purposes stop instructions may be inserted at critical points in the program to permit examination of partial results. Later, when the problem is run, these stops may be ignored.

Start Program

Two methods commonly used to start a program are:

Method 1

1. Set 00 0000 xxxx in the storage-entry switches where xxxx is the address of the first instruction to be executed.
2. Depress the program or computer-reset key.
3. Depress the program-start key.

This procedure causes the Type 650 to get its first instruction from the storage-entry switches. This instruction calls for no operation, and its instruction address is the address of the first instruction of the program. The control switch must be in either the RUN or ADDRESS-STOP positions. The display switch must be in either the accumulator, distributor, or program-register position.

Method 2

1. Set the control switch to MANUAL.
2. Set the address of the first instruction to be executed in the address selection switches.
3. Depress the program or computer-reset key.
4. Depress the transfer key.
5. Set the control switch to RUN or ADDRESS STOP.
6. Depress the program start key.

This procedure causes the Type 650 to obtain its first instruction from the location set in the address selection switches. This is the address of the first instruction to be executed.

The starting procedures described are based on the assumption that the Type 533 operator-functions, if any, are also performed.

ALTHOUGH THE control panel is used basically for flexibility of input-output, its use can have a great effect on the programming. For this reason, it is most important to plan the control panel layout as the programming is being done.

Figure 51 shows the control panel layout. The shaded areas represent additional capacity and optional devices.

The timing chart for the control panel impulses is shown in Figure 77.

Input

Basic control panel wiring for entering information into the machine is covered first.

Read Card A, Read Card B, Read Card C (A-D, 1-20; Q-T, 1-20; X-AA, 1-20). The three sets of read hubs are common (Figure 52). They provide convenient exits for wiring the read brushes to each of the three independent sets of storage-entry hubs. The 80 hubs associated with read card A, B, or C are the exits for the impulses read from card columns 1-80 at the second reading station.

Storage Entry A, Storage Entry B, Storage Entry C (E-J, 1-22; K-P, 1-22; AE-AJ, 1-22). Each group of storage-entry hubs (A, B, or C) is independent of the other two (Figure 52). This arrangement permits the entry of information from at least three different card forms without using pilot selectors or co-selectors. Each entry group has words 1-10.

Only one group of storage entry hubs (A, B, or C) can be active during one particular card cycle. STORAGE-ENTRY C hubs are normally active. STORAGE-ENTRY A hubs are made active by impulsing ENTRY A hubs on the previous cycle. STORAGE-ENTRY B hubs are made active by impulsing ENTRY B hubs on the previous cycle.

Each storage-entry word has eleven positions. The extreme right-hand position of each storage-entry word is the sign position (labeled ±). This position will accept *only* 11 impulses for minus, and 12 impulses for plus. The sign hub does not need to be wired when the READ SIGN OVER UNITS (RSU) switch is plugged.

The units position of each storage-entry word will accept all read digit impulses (12-9). The twelve and eleven impulses in this position are used for sign identification when the READ SIGN OVER UNITS (RSU) hubs are wired. Positions two through ten (2-10) of each storage-entry word will accept read digit impulses zero through nine (0-9).

Wiring (Figure 52)

1. Information enters STORAGE-ENTRY A from READ CARD A.
2. Information enters STORAGE-ENTRY B from READ CARD B.
3. Information enters STORAGE-ENTRY C from READ CARD C.

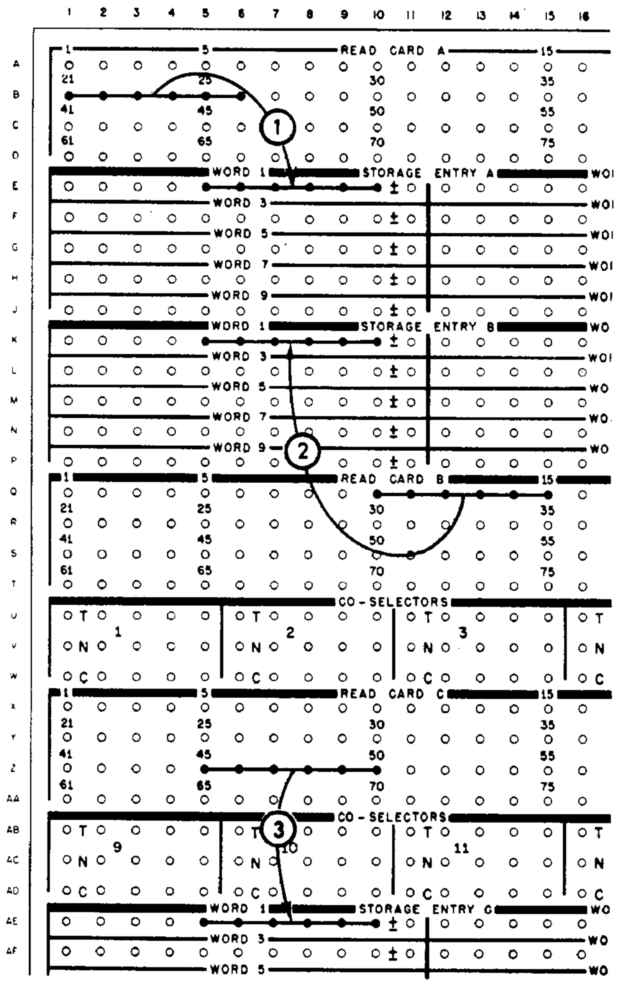


FIGURE 52. READ CARD

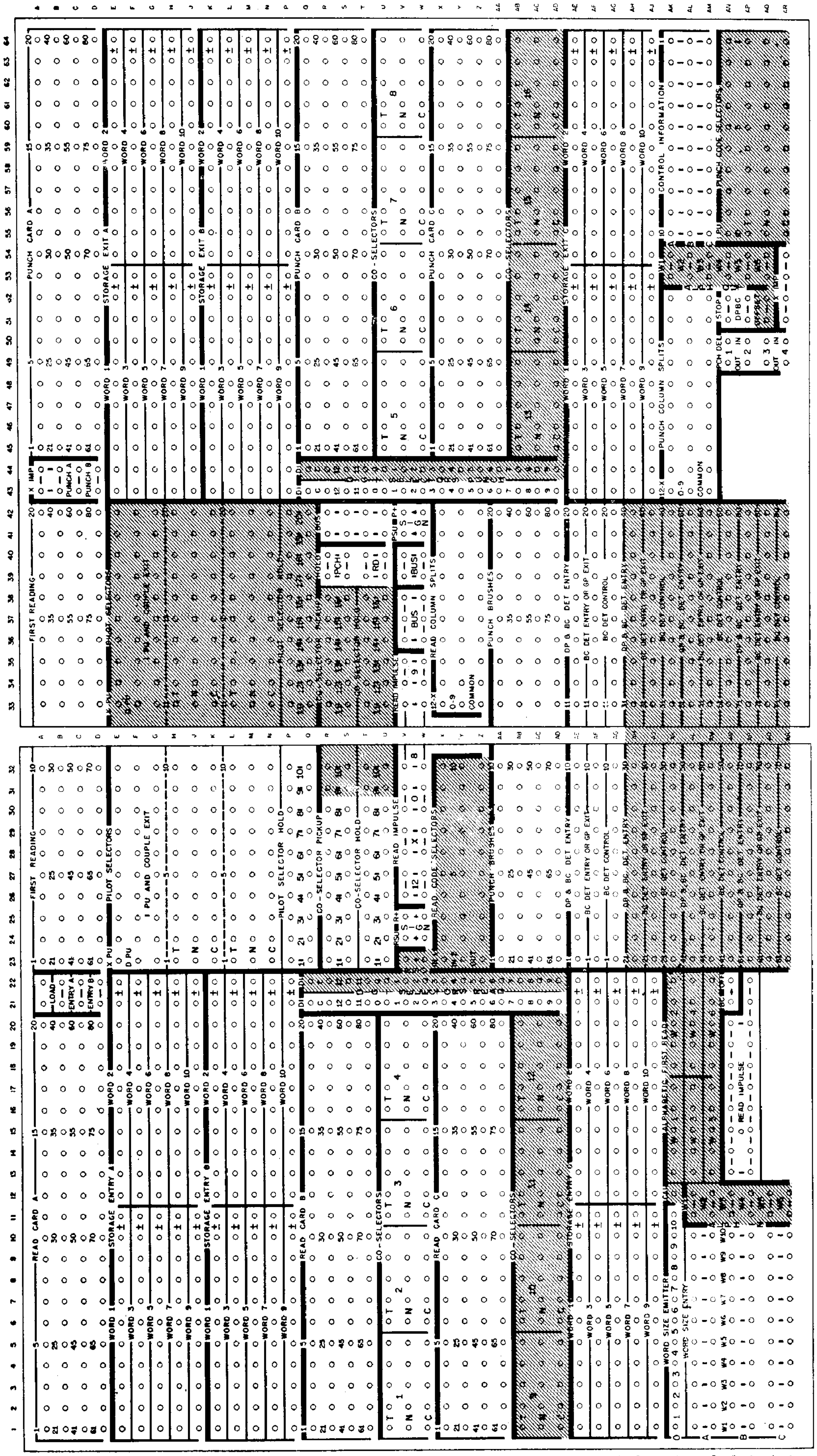


Figure 51. Type 533 Control Panel

The ENTRY A and ENTRY B hubs, mentioned under *Storage Entry,* function as follows:

Entry A, Entry B (C, 21-22; D, 21-22). These hubs will accept all read-digit impulses (Figure 53, wires 2 and 3). An impulse to these hubs will cause either the STORAGE-ENTRY A hubs or the STORAGE-ENTRY B hubs to become active one cycle later. Therefore, these hubs are normally impulsed from first reading. If the ENTRY A and ENTRY B hubs are impulsed simultaneously, only the STORAGE-ENTRY A hubs will be active.

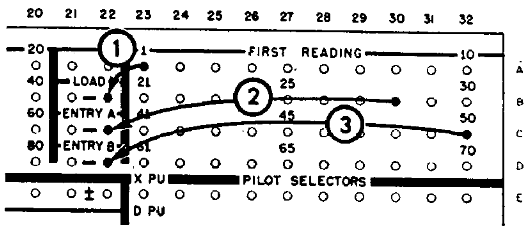


FIGURE 53. READ CARD SELECTION

Load (B, 21-22). The load hubs (Figure 53, wire 1) are receptive to any read 12-impulse. The impulse to this hub causes internal wiring to take precedence over all entry wiring. This internal wiring automatically reads card columns 1-80 into the read storage words 1-8 specified by the D-address of the read instruction. The load hub is effective one read feed cycle after it is impulsed; therefore, it is normally wired from first reading. After the card is read (transferred to general storage), the next instruction will be taken from the location specified by the data address of the READ instruction.

Wiring (Figure 53)

1. A card with a 12-punch in column 1 is identified as a load card. A 12-punch in some column is necessary to identify a load card.
2. Activates STORAGE-ENTRY A.
3. Activates STORAGE-ENTRY B.

First Reading (A-D, 23-42). These hubs (Figure 53) are exits for all read-digit impulses (12-9) and correspond to columns 1-80 of the card being read at the first reading station. They are normally used for control purposes.

Word Sizes

Because the field size of factors being entered from normal cards will vary, the WORD-SIZE EMITTER and WORD-SIZE ENTRY hubs are provided.

Word-size Emitter (AK, 1-11). These hubs, labeled zero through ten (0-10) (Figure 54), are used to fill in zeros automatically to the left of the most significant digit of storage-entry words. It is necessary to enter these zeros in order to satisfy validity checking conditions.

Each of these hubs is labeled to correspond to the number of digits actually being entered into a storage-entry word. For example, a WORD-SIZE EMITTER hub 7 will put zeros into the three high-order positions of a word; WORD-SIZE EMITTER hub 4 will put zeros into the six high-order positions of a word, etc.

When all ten positions of a storage-entry word contain a digit, WORD-SIZE EMITTER hub 10 must be used. WORD-SIZE EMITTER hub 0 can be used when it is desirable to put all zeros in a storage-entry word.

Within an entry group (A, B, or C) that is being used, all word-size entry hubs must be wired to a word-size emitter. Unused words within such an entry group can be wired to any word-size emitter hub.

It is important in wiring the control panel to know the number of digits being read into a storage-entry word. Using a word-size emitter hub less than the actual number of digits will cause zeros to be superimposed over the overlapping digits. Using a word-size emitter hub greater than the actual number of digits will cause blank digit positions in the word.

It is possible to select the word-size emitter impulse by using a pilot selector or co-selector. When word-size emitter impulses are selected, however, it must be remembered that these automatic zeros are the same timewise as zeros read from a card.

Word-Size Entry Hubs (AL-AR, 1-10). These hubs (Figure 54) accept impulses from the word-size emitter hubs. There are three sets of hubs labeled A, B, and C. These hubs are receptive when the corresponding storage-entry hubs are active.

All hubs within a group (A, B, or C) must be wired to a word-size emitter hub when the corresponding group of storage-entry words is being used. If a group of storage-entry words is not being used, no wiring is necessary to the corresponding group of word-size entry hubs.

Wiring (Figure 54)

1. Information enters STORAGE-ENTRY C from READ CARD C.
2. Three zeros filled in to the left of the high-order digit in word 1.
3. Six zeros filled in to the left of the high-order digit in word 2.
4. Ten zeros entered into the unused words of storage group C.

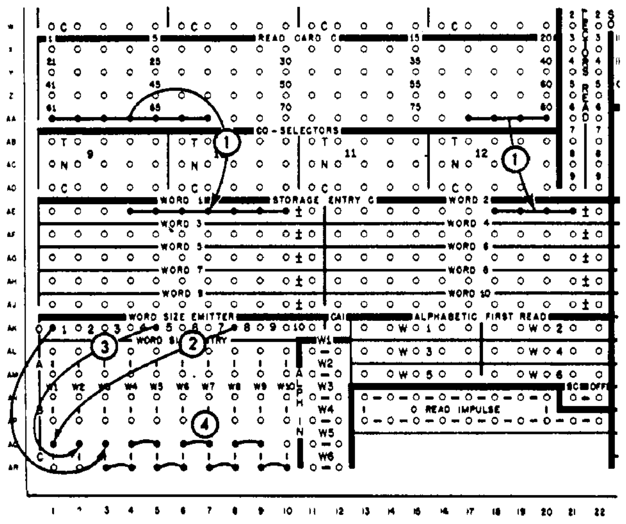


FIGURE 54. WORD SIZE CONTROL

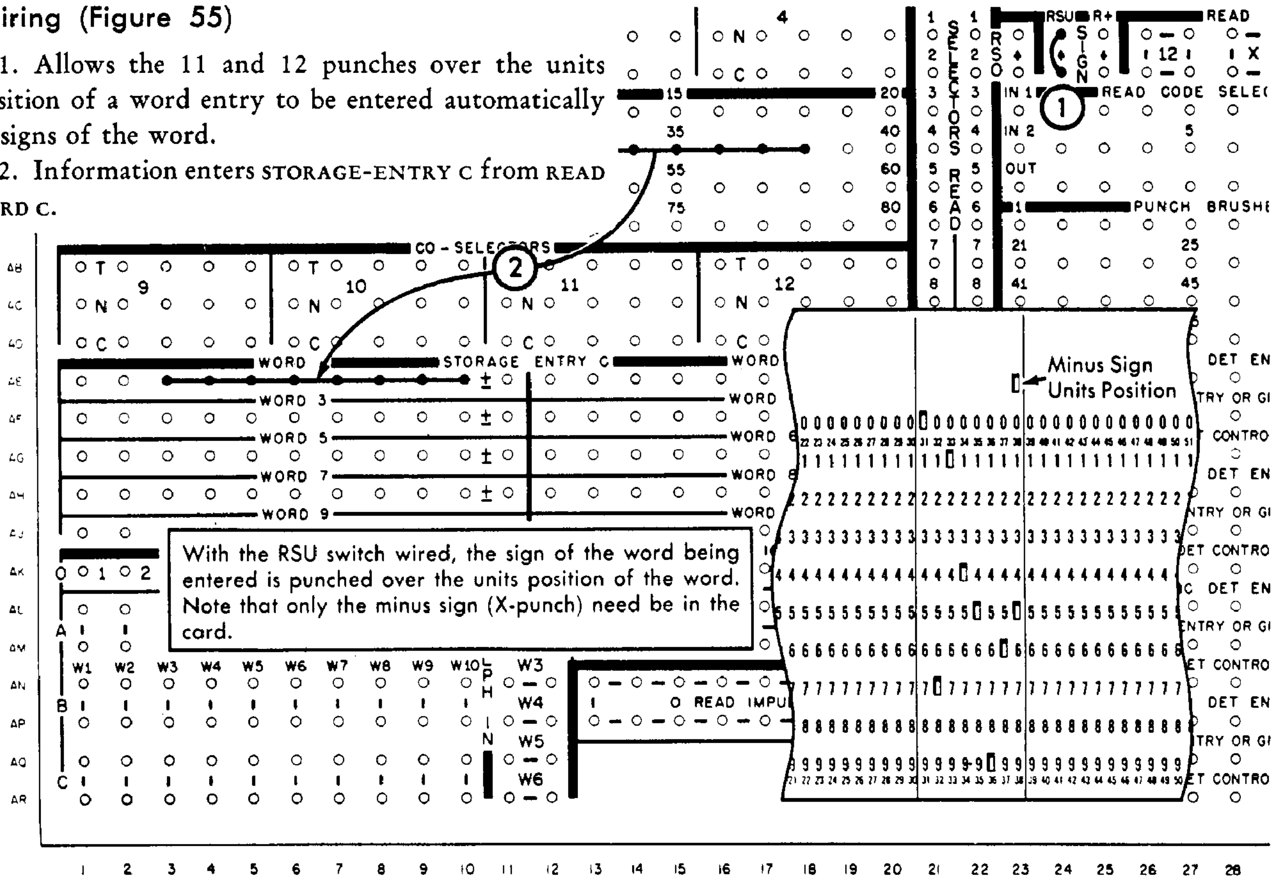
Sign Entry

All factors used within the machine must carry a sign. The entry of signs is controlled by the RSU and R+ switches.

RSU (V-W, 24). When the RSU (read sign over units position) switch is wired (Figures 55 and 56), the units position of each storage entry word is conditioned to accept the sign identification, as well as the units digit of the factor being entered. The RSU switch can be selected. This switch is independent of the R+ (read plus sign) switch.

Wiring (Figure 55)

1. Allows the 11 and 12 punches over the units position of a word entry to be entered automatically as signs of the word.
2. Information enters STORAGE-ENTRY C from READ CARD C.



Minus Sign
Units Position
With the RSU switch wired, the sign of the word being entered is punched over the units position of the word. Note that only the minus sign (X-punch) need be in the card.

FIGURE 55. READ SIGN OVER UNITS POSITION

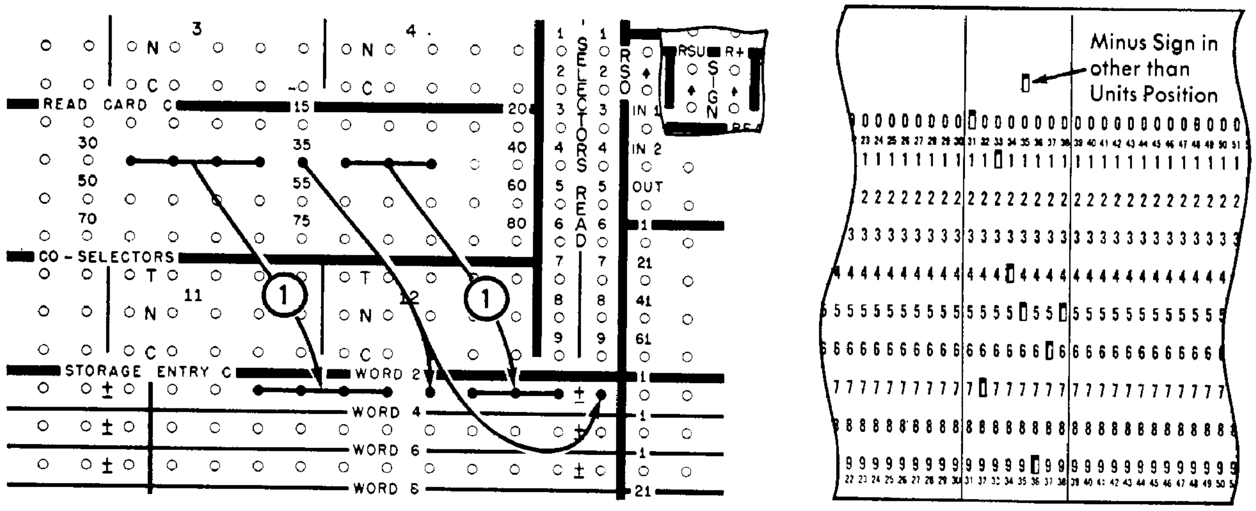


FIGURE 56. SIGN IN OTHER THAN UNITS POSITION

Wiring (Figure 56)

The sign enters the sign entry position of a word if the sign is not over the units position.

1. Normal storage entry wiring.

R+ (V-W, 25). The R+ (read plus sign) switch (Figure 57) is wired if it is desired to recognize twelve punches as positive signs. The R+ switch can be selected. This switch is independent of the RSU (read sign over units) switch. If the R+ switch is not wired, all words that do not receive an 11-impulse will be considered positive.

Wiring (Figure 57)

1. Twelve punches are to be identified as positive signs.

2. This wiring shows how the sign of a factor may be entered into the sign hub if it is not in the units position.

Selection

Pilot selectors and co-selectors are provided to handle the varying conditions of input and output. Each

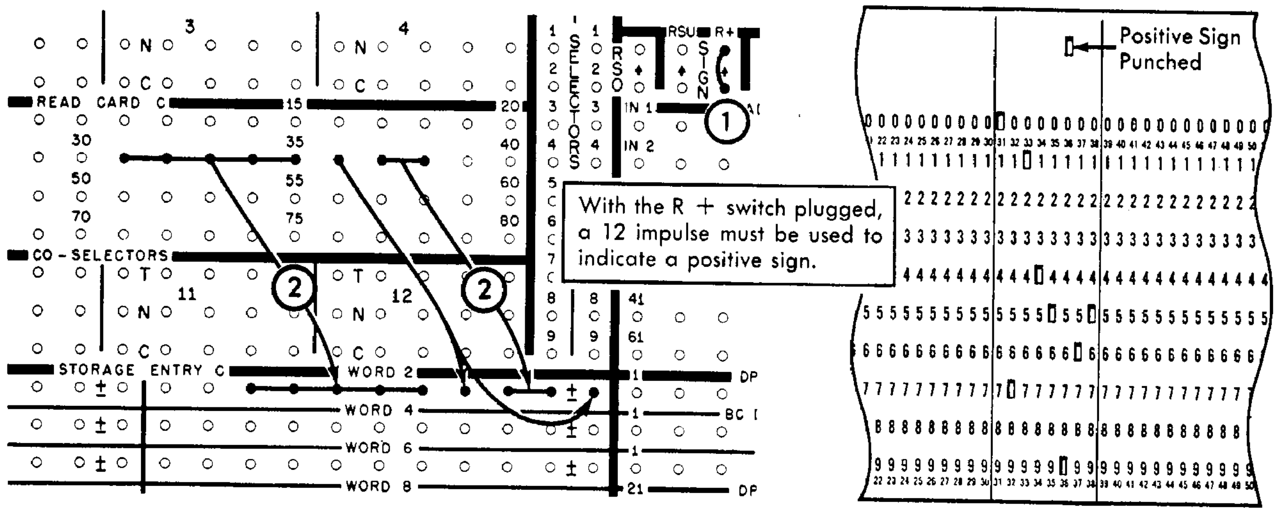


FIGURE 57. READ POSITIVE SIGNS

individual position of a selector is a switch composed of three hubs labeled T, N, and C. The C (Common) hub is always connected to either the N (Normal) or T (Transferred) hub, but never to both. Thus, impulses entering the C hub are selectively available at the N or T hubs depending upon whether the pickup hubs have received an impulse. Conversely, impulses entering either the N or T hubs are available at the C hub under control of the impulses to the pickup hubs.

Both pilot selectors and co-selectors in this machine have hold hubs on the control panel. These hubs must be wired to either the READ or PUNCH HOLD impulses for proper operation of the selectors.

Pilot Selector Pickups (E-G, 23-42). Three types of pilot selector pickups are provided (Figure 58):

1. *XPU.* These hubs will accept any 12 or 11 read impulse and cause the corresponding selectors to transfer on the next read feed cycle. Because there is a one-cycle delay in the transfer of the pilot selectors which is automatically controlled by the read feed, these hubs should not be activated by an impulse timed to the punch feed. The PILOT SELECTOR HOLD hubs must be wired from RD-HOLD when using XPU.

2. *DPU.* These hubs will receive any read-digit impulse (12-9) and cause the corresponding selectors to transfer on the next read feed cycle. Because there is a one-cycle delay in the transfer of the pilot selectors which is automatically controlled by the read feed, these hubs should not be activated by an impulse timed to the punch feed. The PILOT SELECTOR HOLD hubs must be wired from RD-HOLD when using DPU.

3. *IPU and Couple Exit.* These hubs can be either entry hubs or exit hubs.

As entry hubs, they will accept impulses timed to both the read feed and the punch feed. *Impulses timed to the punch feed, which are used to control pilot selectors, must be wired to these hubs.* An impulse entering one of these hubs will cause the associated pilot selector to transfer immediately.

These hubs become exit hubs if either the XPU or DPU hubs are wired. The impulse from these hubs is available at the beginning of the next read-feed cycle after the XPU or DPU hub is impulsed.

The couple exit impulse is usually used to control other pilot selectors and co-selectors.

Pilot Selectors (H-N, 23-42). Ten pilot selectors are standard with the Type 650. Additional pilot selectors are available in groups of five. Each pilot selector has two positions.

Pilot Selector Hold (P-Q, 23-42). These hubs are entries for impulses that will determine how long a pilot selector will remain transferred. Because the pilot selectors can be used with either the read feed or the punch feed, no automatic pilot selector hold impulse is provided. The hold impulse is provided by control panel wiring from either the RD-HOLD or PCH-HOLD hubs depending upon which feed controls the action of the pilot selector.

Read Hold (T-U, 39-40). These hubs emit an impulse which, when wired to CO-SELECTOR HOLD or PILOT SELECTOR HOLD, will cause the co-selector or pilot selector to remain transferred for the duration of the read-feed cycle.

Co-Selector Pickup (R-S, 23-38). These hubs will accept impulses timed to both the read feed and the punch feed. An impulse entering one of these hubs will cause the associated co-selector to transfer immediately.

Co-Selectors (U-W, AB-AD, 1-20 and 45-64). Eight co-selectors are standard with the Type 650. Additional co-selectors are available in groups of four. Each co-selector has five positions.

Co-Selector Hold (T-U, 23-38). These hubs are entries for impulses that will determine how long the co-selector will remain transferred. Because the co-selector can be used with either the read feed or the punch feed, no automatic co-selector hold impulse is provided. The hold impulse is provided by control panel wiring from either the RD HOLD or PCH HOLD hubs depending upon which feed controls the action of the selector.

Wiring (Figure 58)

1. Pilot selector 1 is activated by X or 12 punches in column 24.
2. Co-selector 4 is picked up from couple exit of pilot selector 1.
3. Pilot selector 1 and co-selector 4 remain transferred for the duration of a read feed cycle.
4. Shows how field selection can be accomplished using a co-selector.
5. Shows how column selection can be accomplished using a pilot selector.

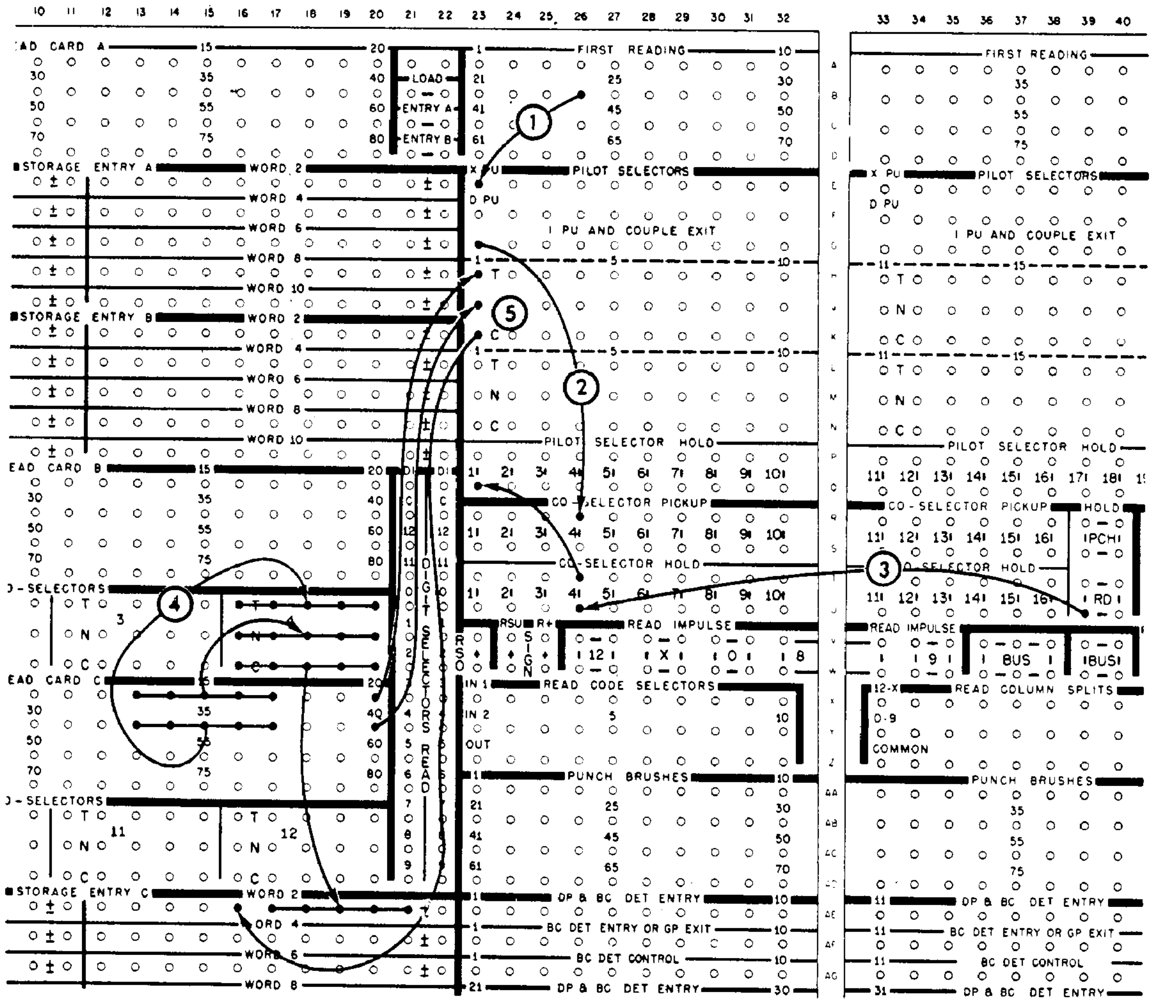


FIGURE 58. FIELD SELECTION AND COLUMN SELECTION

Emitted Impulses (Read)

Factors may be supplied from the control panel for entry into the machine. These emitted factors might be used for such things as filling in zeros to the right of fields wired to storage entry, supplying sign impulses or converting control factors to 8 or 9 for interrogation by the program. The hubs supplying these factors are labeled *Read Impulses.*

There is also a digit selector that can be used as an emitter for all of the read-digit impulses.

Read Impulse (V-W, 26-35; AN, 13-20; AP, 13-22). These hubs (Figure 59) emit read-digit impulses 12, X, 0, 8, and 9.

Wiring (Figure 59)

1. A selected digit impulse enters the units position of word 2.
2. A read digit nine impulse enters the normal side of a pilot selector.
3. A read digit eight impulse enters the transferred side of a pilot selector.

DI (Q, 21). This hub (Figure 60) emits all read-digit impulses (12-9) every read feed cycle.

Digit Selectors-Read (R-AD, 21). One read-digit selector (Figure 60) is standard with the Type 650. One additional read-digit selector or one ½ time emitter may be obtained. Digit selectors are used to select specific digits from a card column or to emit digits on

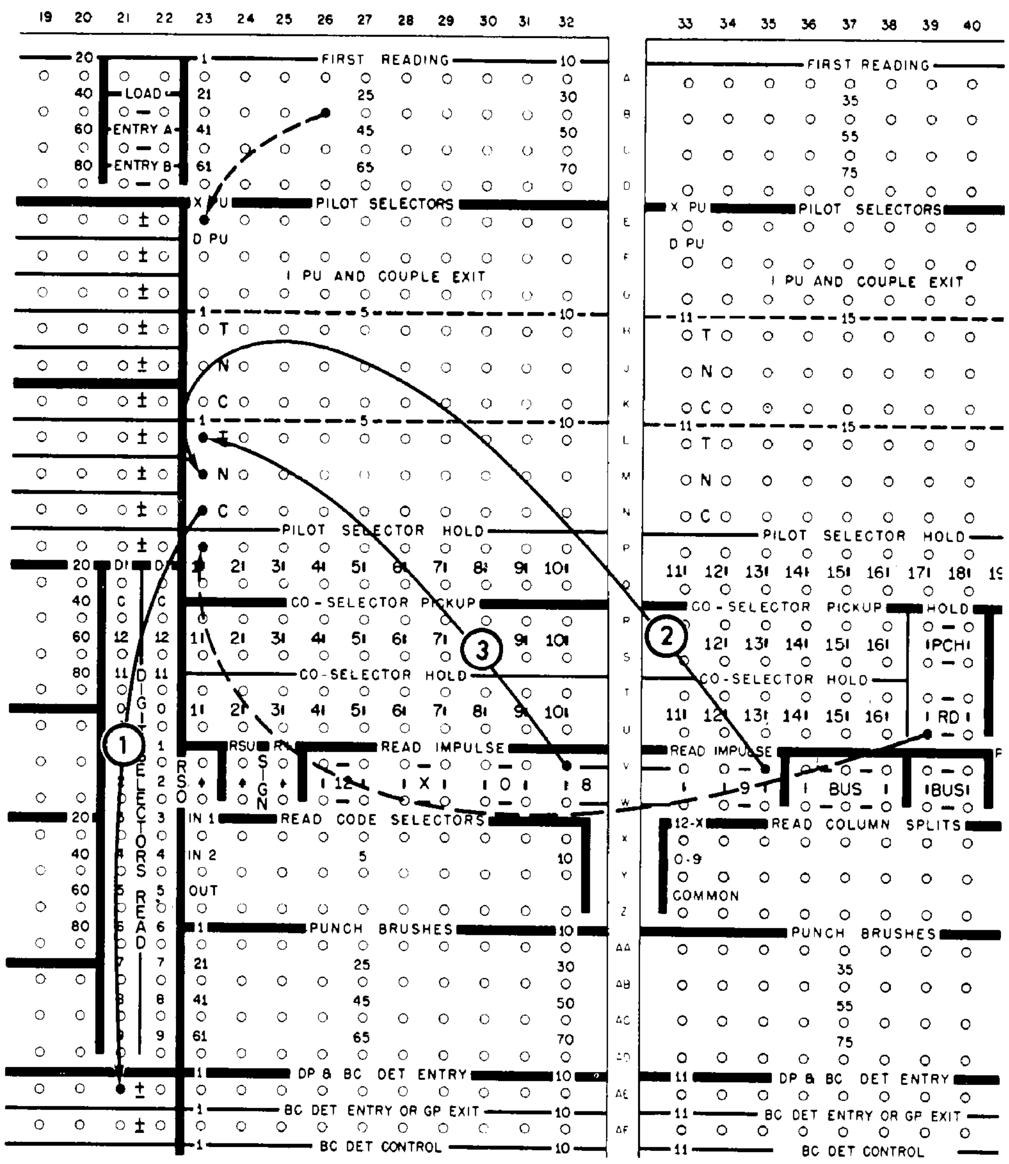


FIGURE 59. READ IMPULSE SELECTION

every read cycle. When a card column is wired to C (Common), a specific digit impulse may be read from the corresponding hub of the digit selector. Thus, each digit that is punched will be available for control purposes. If the DI hub is wired to the C (Common) hub, specific digit impulses are available from each of the exit hubs (digit emitter).

Wiring (Figure 60)

1. Digit impulses available at the read-digit selector.

2. A digit-five impulse being selected using the READ DIGIT SELECTOR.

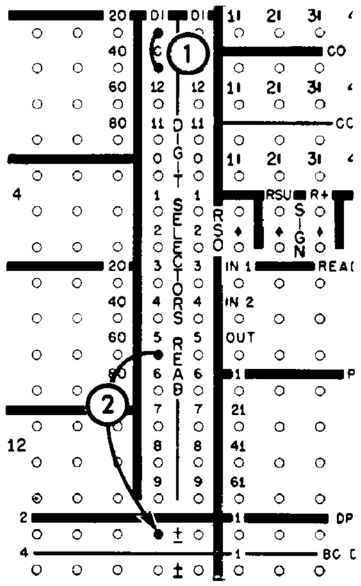


FIGURE 60. DIGIT EMITTING

Column Splitting

The need to separate a column between two digit positions (usually 0 and 11) occurs rather frequently. Read column splits are provided for this purpose.

Read Column Splits (X-Z, 33-42). These hubs (Figure 61) are used to separate 12 and 11 impulses from 0-9 impulses that are read from a card at first or second reading stations.

Wiring (Figure 61)

1. The impulses read in column 80 enter the common of the column split.

2. Digit impulses 0-9 from column 80 are directed to the DPU of pilot selector 9.

3. Digit impulses X-12 from column 80 are directed to the XPU of pilot selector 10.

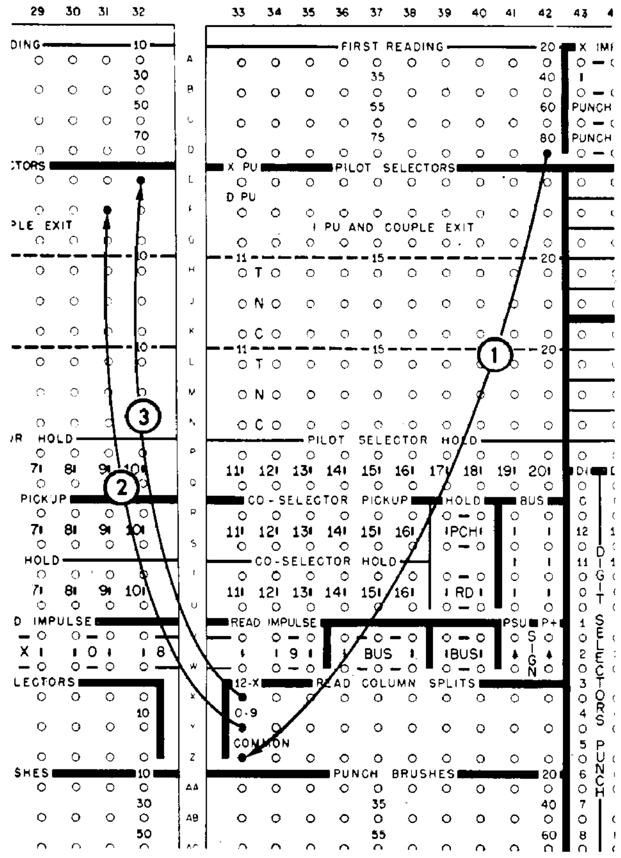


FIGURE 61. COLUMN SPLITTING

Special Devices

Read Code Selectors (X-Z, 23-32), page 101.
RSO (Read Second Only; V-W, 23), page 101.
Alphabetic In (AL-AR, 11-12), page 105.
CAI (AK, 12), page 105.
Alphabetic First Read (AK-AM, 13-22), page 105.

Output

The control panel provides the means for punching the results of the Type 650 calculations into IBM cards for output.

Punch Card A, Punch Card B, Punch Card C (A-D, 45-64; Q-T, 45-64; X-AA, 45-64). The eighty hubs associated with each punch card group (A, B, or C) are entries for punching into the eighty columns of the output card (Figure 62). Each group of punch card hubs (A, B, or C) is independent of the other two. This arrangement permits punching at least three different card forms without using pilot selectors or co-selectors.

Only one group of punch-card hubs (A, B, or C) can be active during one particular card cycle. Punch card C hubs are normally active. PUNCH CARD A hubs are made active by impulsing the PUNCH A hubs. PUNCH CARD B hubs are made active by impulsing the PUNCH B hubs.

Storage Exit A, Storage Exit B, Storage Exit C (E-J, 43-64; K-P, 43-64; AE-AJ, 43-64). The three groups of storage exit hubs are common (Figure 62). They are located to facilitate wiring to PUNCH CARD A, PUNCH CARD B, or PUNCH CARD C. Each group has words 1-10. Each word has 11 exit positions, one for each digit of the word and one for the sign. The action of the sign exit and units position hubs are under control of the PSU (punch sign over units) switch and the P+ (punch plus) switch.

Wiring (Figure 62)

1. Information from STORAGE-EXIT A enters PUNCH CARD A.

2. Information from STORAGE-EXIT B enters PUNCH CARD B.

3. Information from STORAGE-EXIT C enters PUNCH CARD C.

Control Information (AK-AM, 55-64). These hubs (Figure 63) emit an impulse before 12-punch time if there is a digit 8 in the corresponding position of

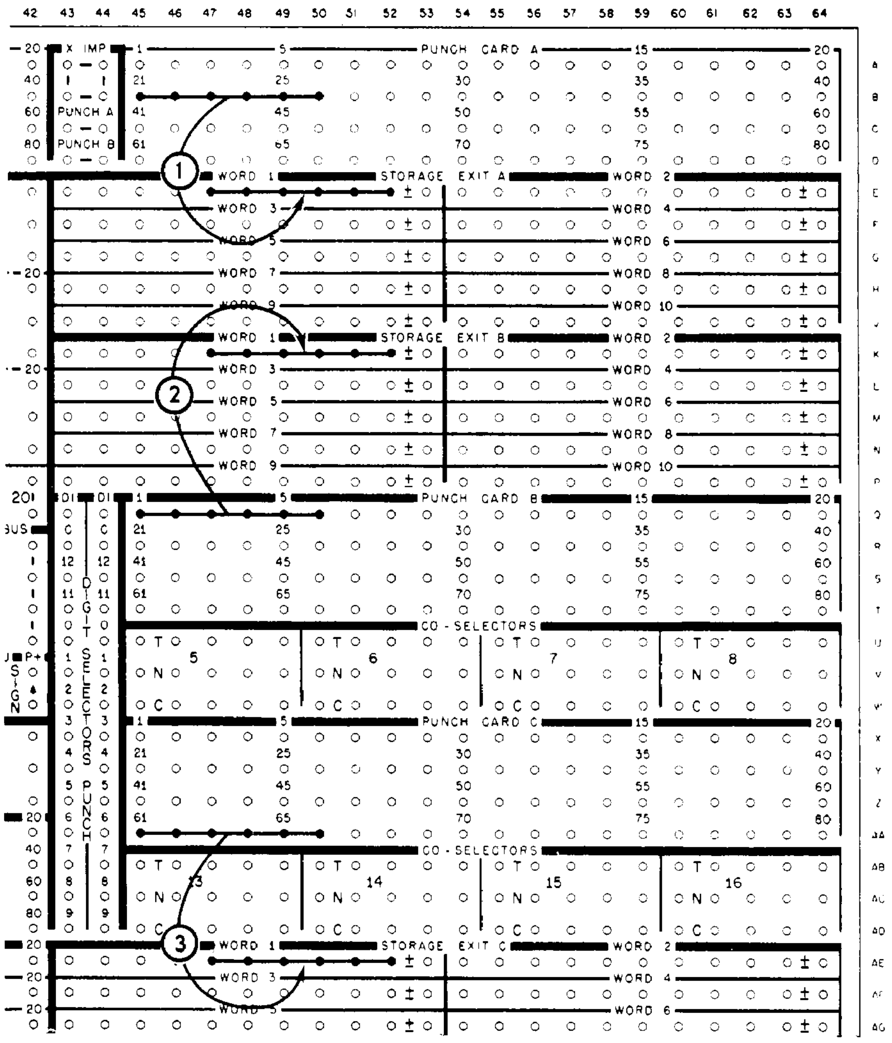


FIGURE 62. PUNCH CARD

storage-exit word 10. If there is any digit other than 8 in a particular position, the corresponding position of control information does not emit any sort of impulse.

Punch A, Punch B (C-D, 43-44). An impulse to PUNCH A or PUNCH B will cause PUNCH CARD A or PUNCH CARD B to become active on the same cycle. These hubs (Figure 63) are normally impulsed from control information. If PUNCH A and PUNCH B are impulsed simultaneously, only PUNCH CARD A will be active.

Wiring (Figure 63)

1. A path is provided for the control information impulse to the PUNCH A hubs.

2. A path is provided for the control information impulse to the PUNCH B hubs.

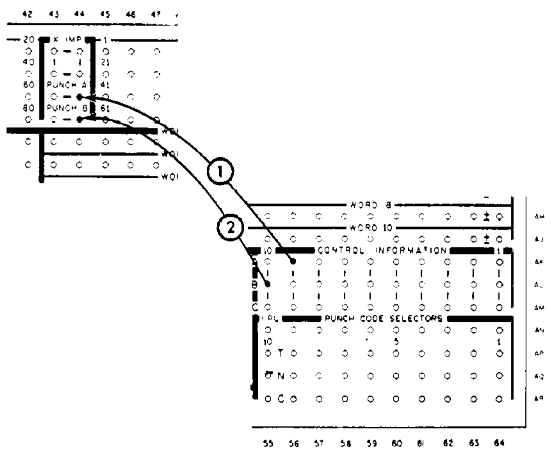


FIGURE 63. PUNCH CARD SELECTION

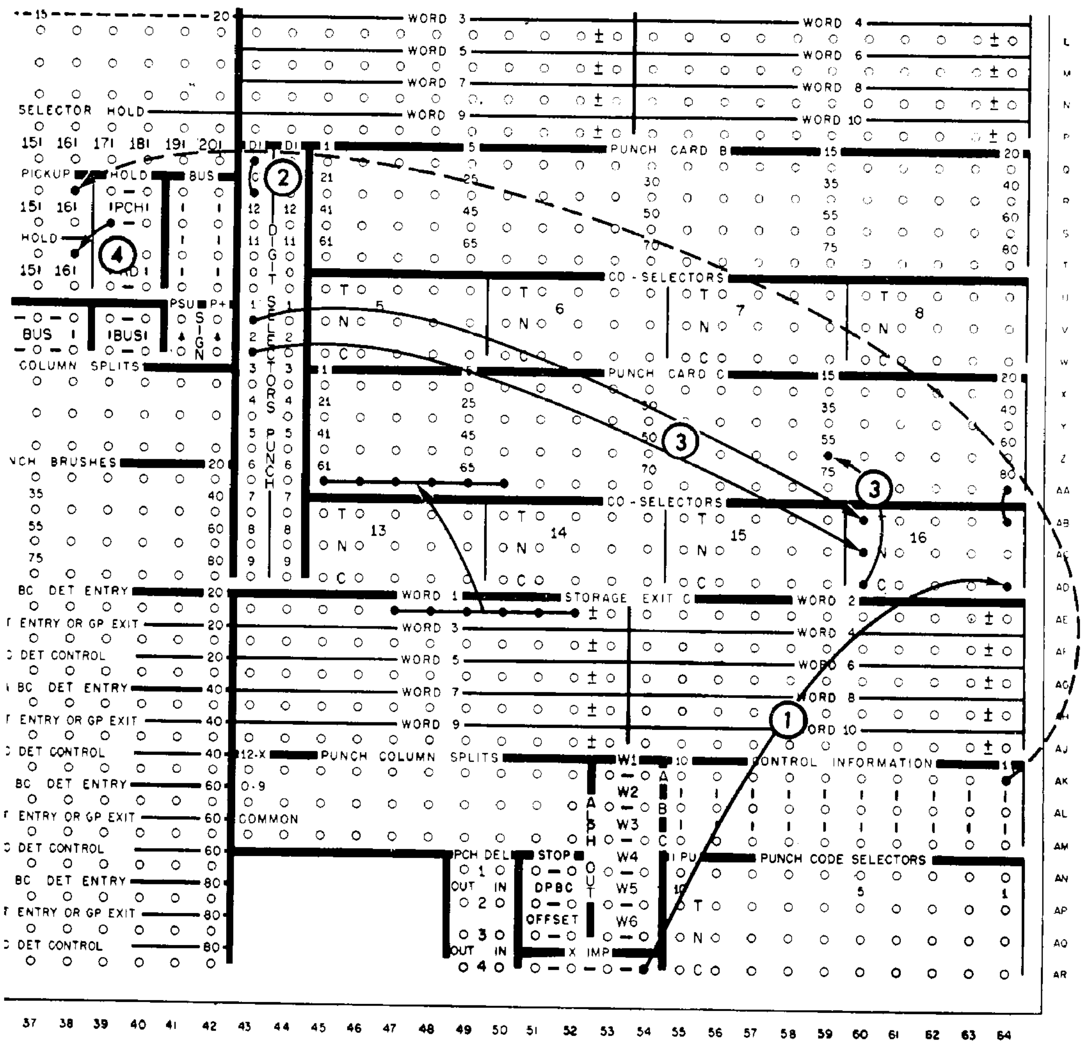


FIGURE 64. CARD IDENTIFICATION AND DIGIT EMITTING

Emitted Punch Impulses

Control panel hubs (Figure 64) are provided for emitting impulses for use while output cards are being punched.

X IMP (A-B, 43-44; AR, 51-54). These hubs emit an X-impulse during each punch feed cycle.

DI (Digit Impulse; Q, 43). This hub emits all punch digit impulses (12-9) every punch feed cycle.

Digit Selector—Punch (R-AD, 43). One punch digit selector is standard with the Type 650. One additional punch digit selector or one ½ time emitter, may be obtained. Punch digit selectors may be used to separate or combine multiple digits. It may also be used as a punch digit emitter by connecting the common to the DI-impulse.

Output Selection

The pickup control of selectors has already been discussed. Because the hold must be timed to the proper feed, punch hold hubs are provided.

Punch delay hubs are provided in case it is desirable to delay the effect of a punch timed impulse one or more punch feed cycles.

Punch Hold (R-S, 39-40). These hubs (Figure 64) emit an impulse which, when wired to a co-selector hold or a pilot selector hold, will cause the selector to remain transferred for the duration of the punch feed cycle.

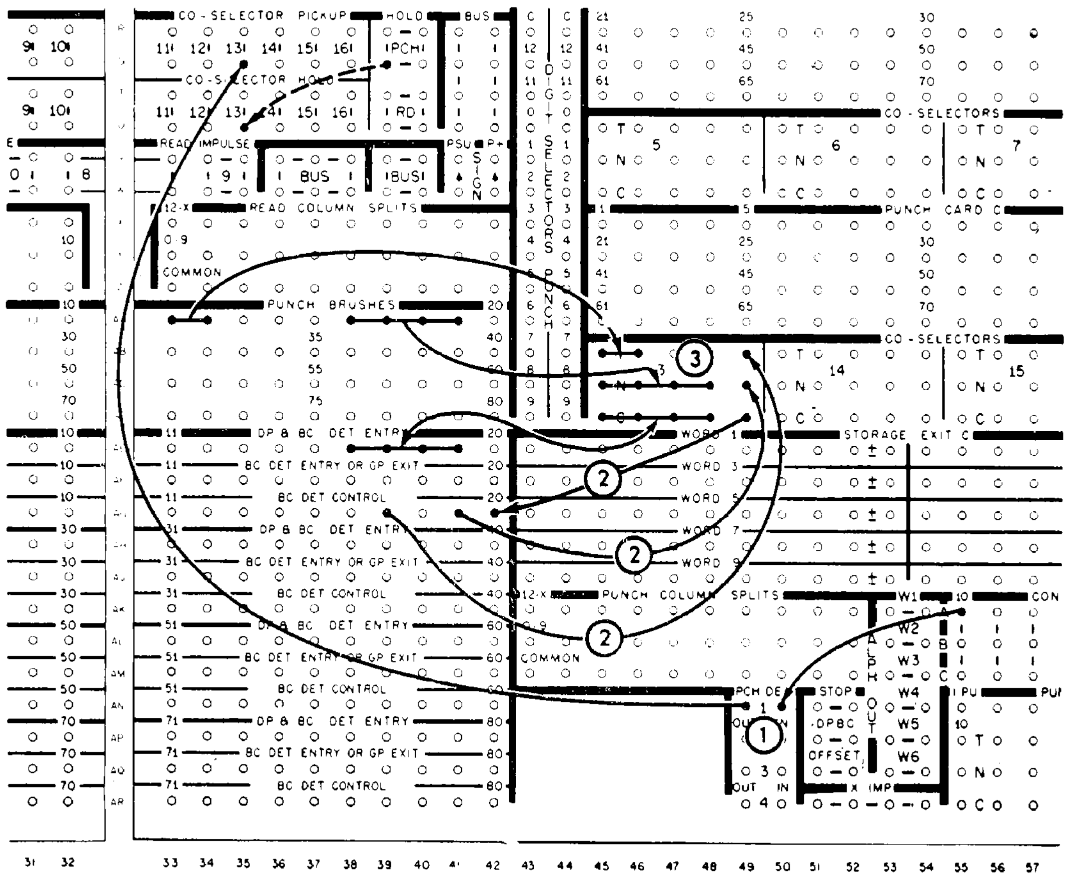


FIGURE 65. FIELD SELECTION USING PUNCH DELAY

Wiring (Figure 64)

1. A path is provided for the X-impulse to column 80 through the transferred points of a co-selector.
2. Punch digit impulses are made available to the punch digit selector.
3. Digit 1 and digit 2 impulses from the digit selector selected by use of a co-selector.
4. Co-selectors remain transferred for the duration of the punch feed cycle.

Punch Delay (AN-AR, 49-50). There are four sets of punch delay hubs (Figure 65) on the control panel. Each set of hubs consists of an IN hub and an OUT hub. The IN hub will accept any impulse timed to the punch feed. On the punch feed cycle after the IN hub is impulsed, the OUT hub will emit an impulse before 12 punch time.

Wiring (Figure 65)

1. The punch-delay unit is activated by an impulse from control information position 10.
2. The selection of blank-column detection control matches the field entering the DP AND BC DET ENTRY hubs.
3. Field selection of information directed to the DP AND BC DET ENTRY.

It is possible by means of control panel wiring to create a two-cycle delay (Figure 66). This can be accomplished by wiring the OUT hub of one delay unit to the IN hub of another delay unit.

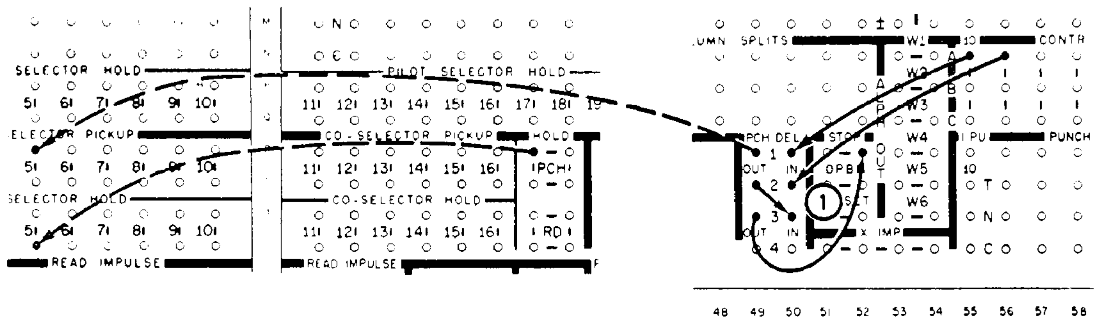
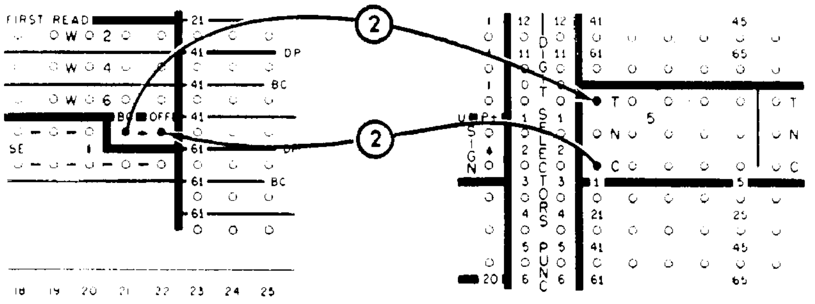



FIGURE 66. PUNCH DELAY AND BC-OFF SELECTION

Wiring (Figure 66)

1. The machine stops two punch-feed cycles after an impulse is emitted from control information position 9.
2. The BC OFF switch selected under control of a control information impulse from position 10.

Output Signs

Punching of signs in output cards is controlled by the PSU and P+ switches.

PSU (Punch Sign over Units; V-W, 41). When this switch is wired (Figure 67), signs will automatically be punched over the units position of a storage-

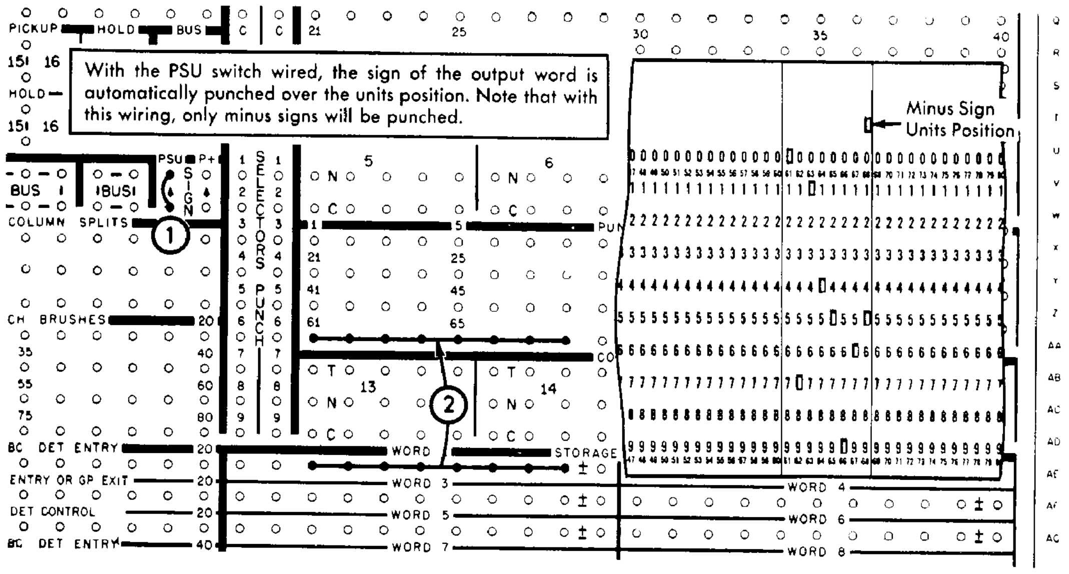

With the PSU switch wired, the sign of the output word is automatically punched over the units position. Note that with this wiring, only minus signs will be punched.
Minus Sign Units Position

FIGURE 67. PUNCH SIGN OVER UNITS POSITION

exit word, and the sign position of each storage-exit word is made inactive. The PSU switch can be selected. This switch is independent of the P+ (punch plus sign) switch.

Wiring (Figure 67)

1. Signs of factors are punched over the units position of the field.

2. Information from STORAGE-EXIT C enters PUNCH CARD C.

P+ (Punch Plus Sign; V-W, 42). The P+ (punch plus sign) switch (Figure 68) is wired if it is desired to punch twelve holes for positive signs. The P+ switch can be selected. This switch is independent of the PSU (punch sign over units) switch.

Wiring (Figure 68)

1. A 12-hole is punched in the card to identify positive factors.

2. Information from STORAGE-EXIT C enters PUNCH CARD C. (Note sign storage exit wiring.)

Punch-column splits are provided. Their function is similar to that of the read-column splits.

Punch-Column Splits (AK-AM, 43-52). These hubs (Figure 69) are used to separate 12- and 11-punch impulses from 0-9 punch impulses.

Punch-Brush Station

A card in the punch feed passes the punch brushes one punch-feed cycle after it has been punched. Information read by the punch brushes may be used for gangpunching and double-punch blank-column detection.

Punch Brushes (AA-AD, 23-42). These hubs (Figure 69) are exits for digit impulses (12-9) read from the output card one cycle after the card is punched.

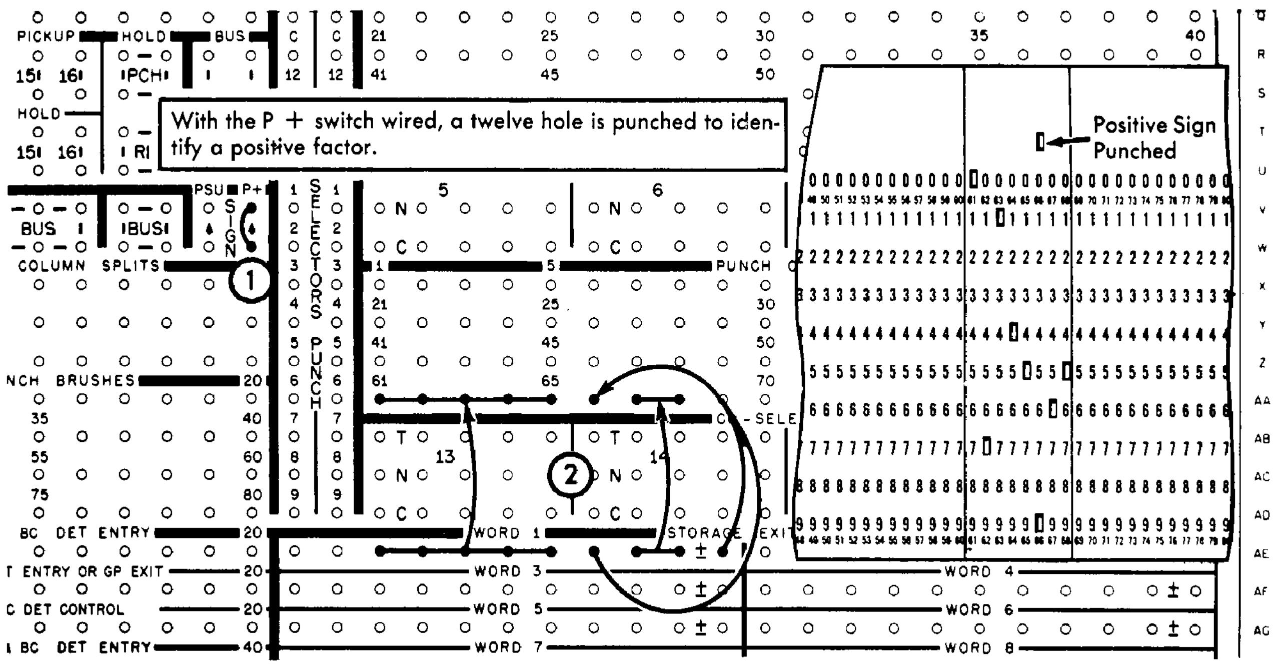


FIGURE 68. PUNCH POSITIVE SIGNS

Wiring (Figure 69)

1. The digit impulse from the punch brushes enters the common of the punch column split.
2. The 12- and 11-impulses from the punch brush are directed to one position of DOUBLE-PUNCH and BLANK-COLUMN-DETECTION ENTRY.
3. The 0-9 impulses from the punch brush are directed to one position of DOUBLE-PUNCH and BLANK-COLUMN-DETECTION ENTRY.

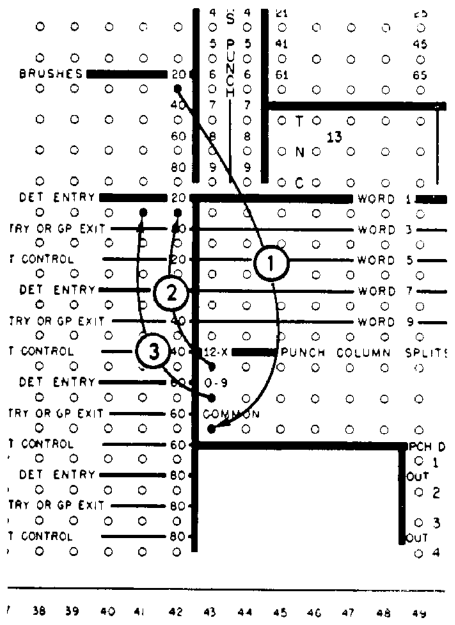


FIGURE 69. COLUMN SPLITTING

Double-Punch Blank-Column Detection (DPBC)

The Type 650 is equipped with 20 positions of DPBC detection as a standard feature. Additional positions are available in groups of ten, maximum of six additional groups.

BC OFF (Blank column off; AN, 21-22). When this switch is wired, blank columns will not be detected by the DPBC device. The BC OFF switch may be selected (Figure 66).

DP & BC DET ENTRY (Double-punch & blank-column detection entry; AE, AH, AL, AP, 23-42). These hubs (Figure 70) are entries for the digit impulses from the punch brushes. They are used to detect double-punched or blank columns in the output cards.

Wiring (Figure 70)

1. Twenty positions are checked for double punches and blank columns.

BC DET ENTRY or GP EXIT (Blank-column detection entry or gangpunch exit; AF, AJ, AM, AQ, 23-42). These hubs may be either entries or exits (Figure 71).

As entries, they are used to detect blank columns in the output cards without double-punch detection.

As exits, they are wired to PUNCH CARD A, B, or C for gangpunching. When used as gangpunch exits, only the first digit of any column wired to a DP and BC DET ENTRY hub will be available at the associated GP EXIT hub.

Wiring (Figure 71)

1. Positions 1-10 are checked for double punches and blank columns; the same columns are also gangpunched.

2. Positions 11-20 are checked for blank columns only.

BC DET CONTROL (Blank-column detection control; AG, AK, AN, AR, 23-42). These hubs (Figure 72) provide a means for control of blank-column detection. All positions of DPBC are internally connected for blank-column detection. It is possible to bypass any particular position or positions that are not to be checked for blanks by control panel wiring in the following manner:

The blank-column-detection control hub for the last position, in which blanks are to be checked, is connected by a control panel wire to the blank-column-detection control hub preceding the next position which is to be checked for blanks.

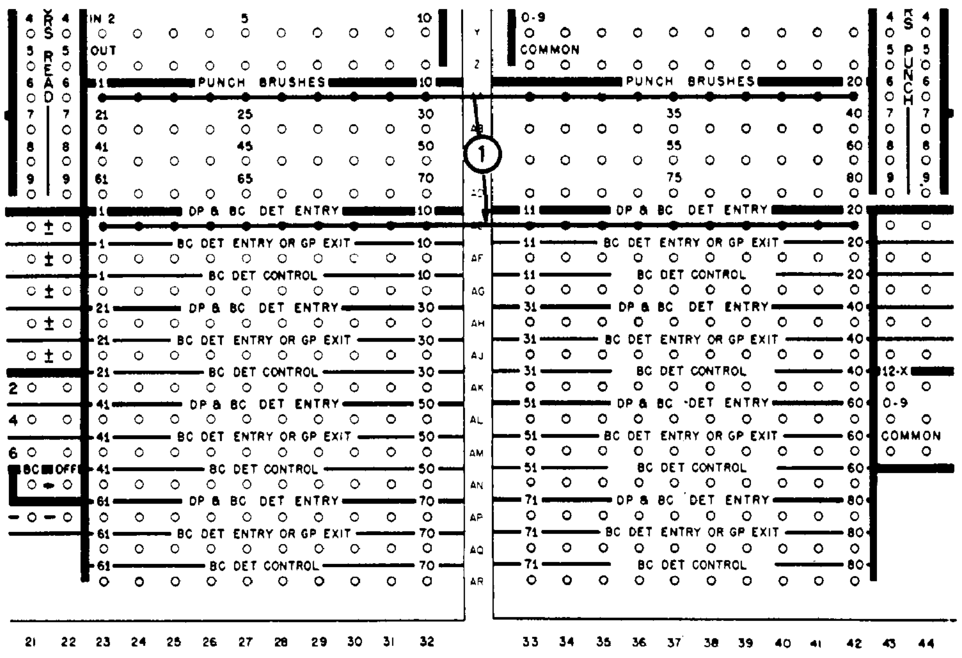


FIGURE 70. DOUBLE-PUNCH AND BLANK-COLUMN DETECTION

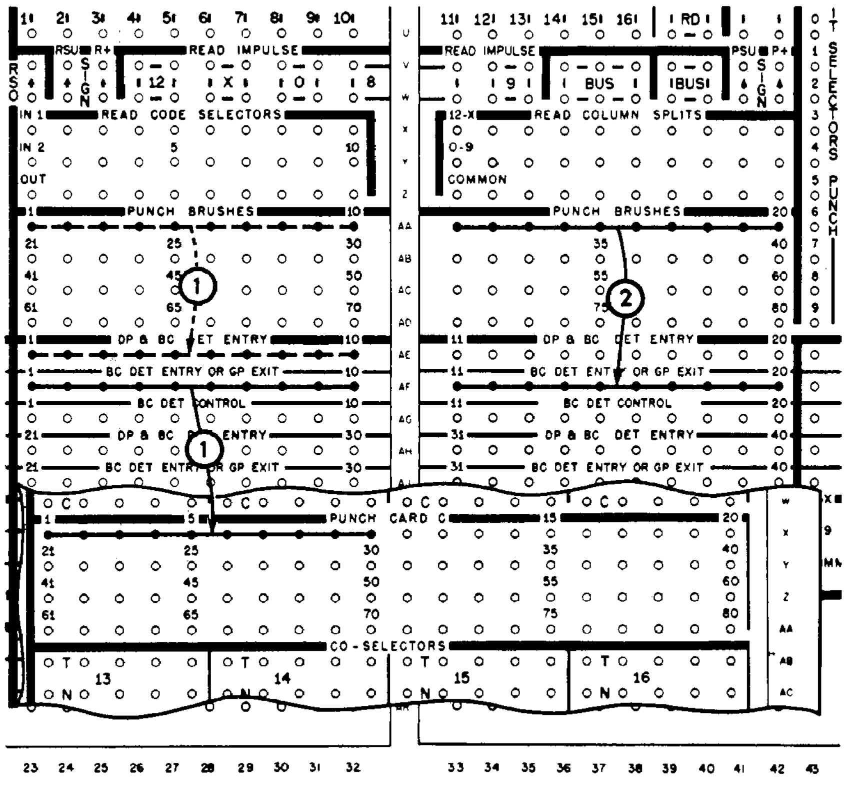


FIGURE 71. GANGPUNCHING AND BLANK-COLUMN DETECTION ONLY

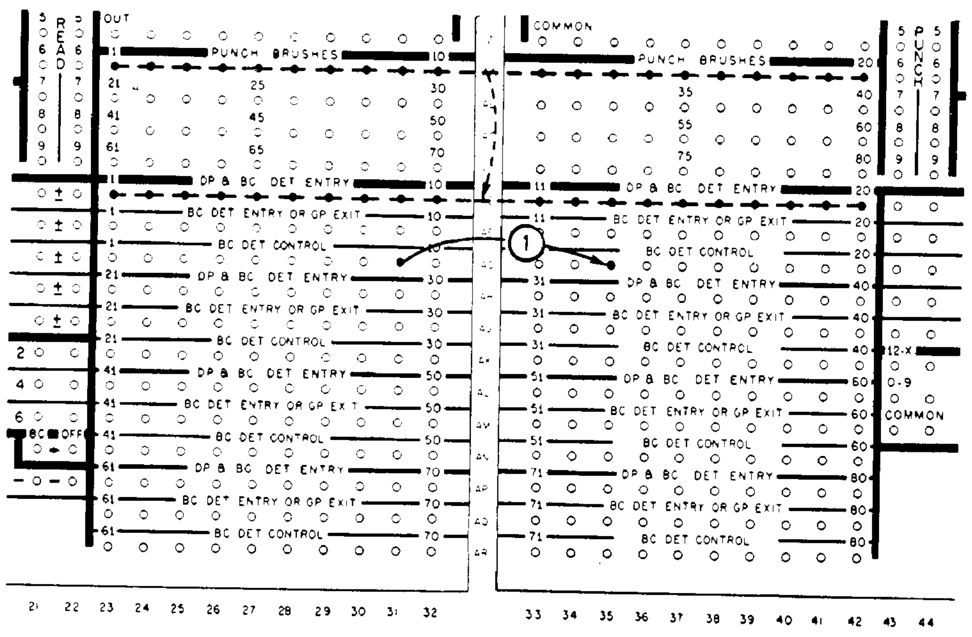


FIGURE 72. BLANK-COLUMN DETECTION CONTROL SELECTION

Wiring (Figure 72)

1. Columns 10-13 are checked for double punches only.

In case it is necessary to bypass checking for blanks in the last position (position 20 for standard machine), the blank-column-detection control hub for the *last position in which blanks are to be checked* must be connected by a control panel wire to the blank-column-detection control hub of the last DPBC position (Figure 73).

Wiring (Figure 73)

1. The last ten positions of blank-column detection are bypassed.

In case it is desirable to bypass checking for blanks in DPBC position one, it is necessary to wire from the exit hub of the BC OFF switch (AN, 21) to the blank-column-detection control hub preceding the first position that is to be checked for blanks (Figure 74).

The wiring to bypass blank-column checking may be selected.

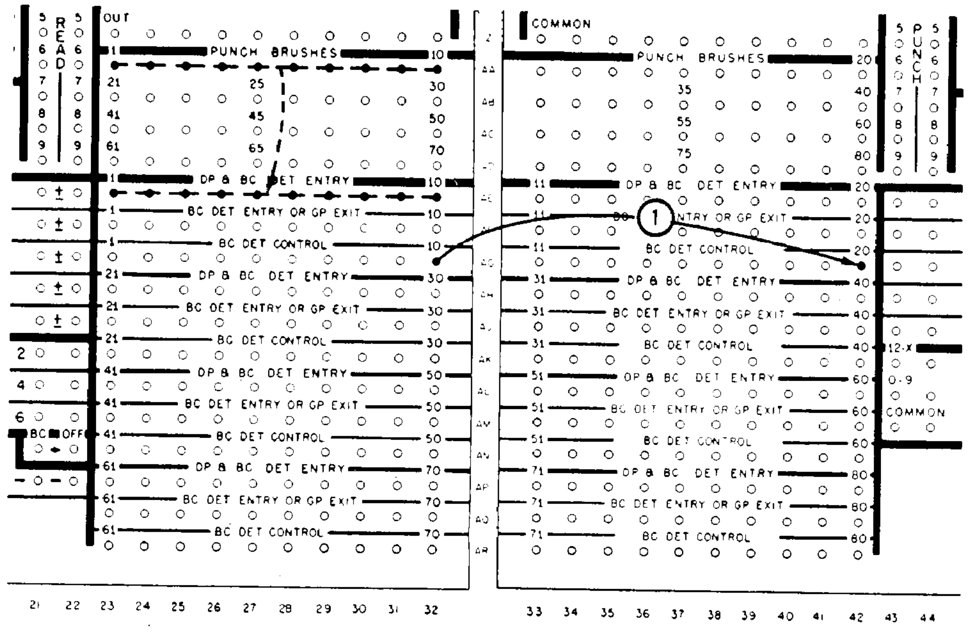


FIGURE 73. BLANK-COLUMN DETECTION CONTROL SELECTION

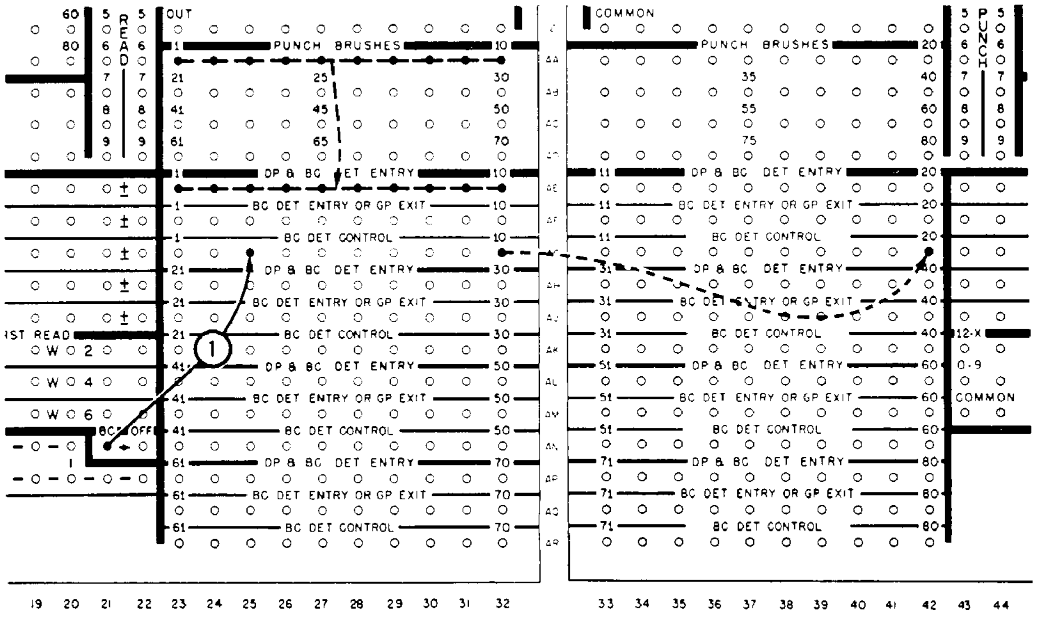


FIGURE 74. BLANK-COLUMN DETECTION CONTROL SELECTION

Wiring (Figure 74)

1. The first three positions of blank-column detection are bypassed.

DPBC (AP, 51-52). These hubs (Figure 75) emit an impulse one punch-feed cycle following the detection of a double-punch or blank-column by the DPBC device. If the DPBC impulse is wired to STOP, the punch feed will stop with the error card as the last card in the punch stacker.

STOP (AN, 51-52). These entry hubs (Figure 75) accept impulses to stop the punch feed.

Wiring (Figure 75)

1. A double-punch or blank-column indication stops the machine.

BUS (R-U, 41-42; V-W, 36-38 and 39-40). Three sets of bus hubs are provided on the control panel. All hubs that are connected by lines on the panel are common. They are provided to minimize the need for split wires.

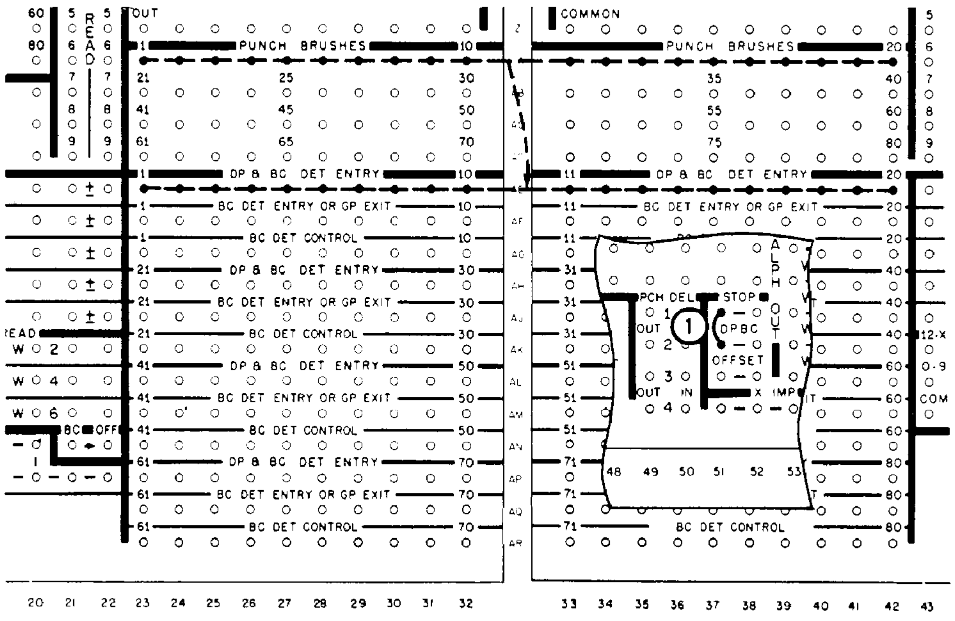


FIGURE 75. MACHINE STOP

Special Devices

Offset (AQ, 51-52), page 104.
Punch Code Selectors (AN-AR, 55-64), page 102.
Alpha Out (AK-AQ, 53-54), page 105.

Control Panel Example — Inventory

Figure 76 shows the control panel wiring for the problem in programming examples shown in Figures 38 and 39.

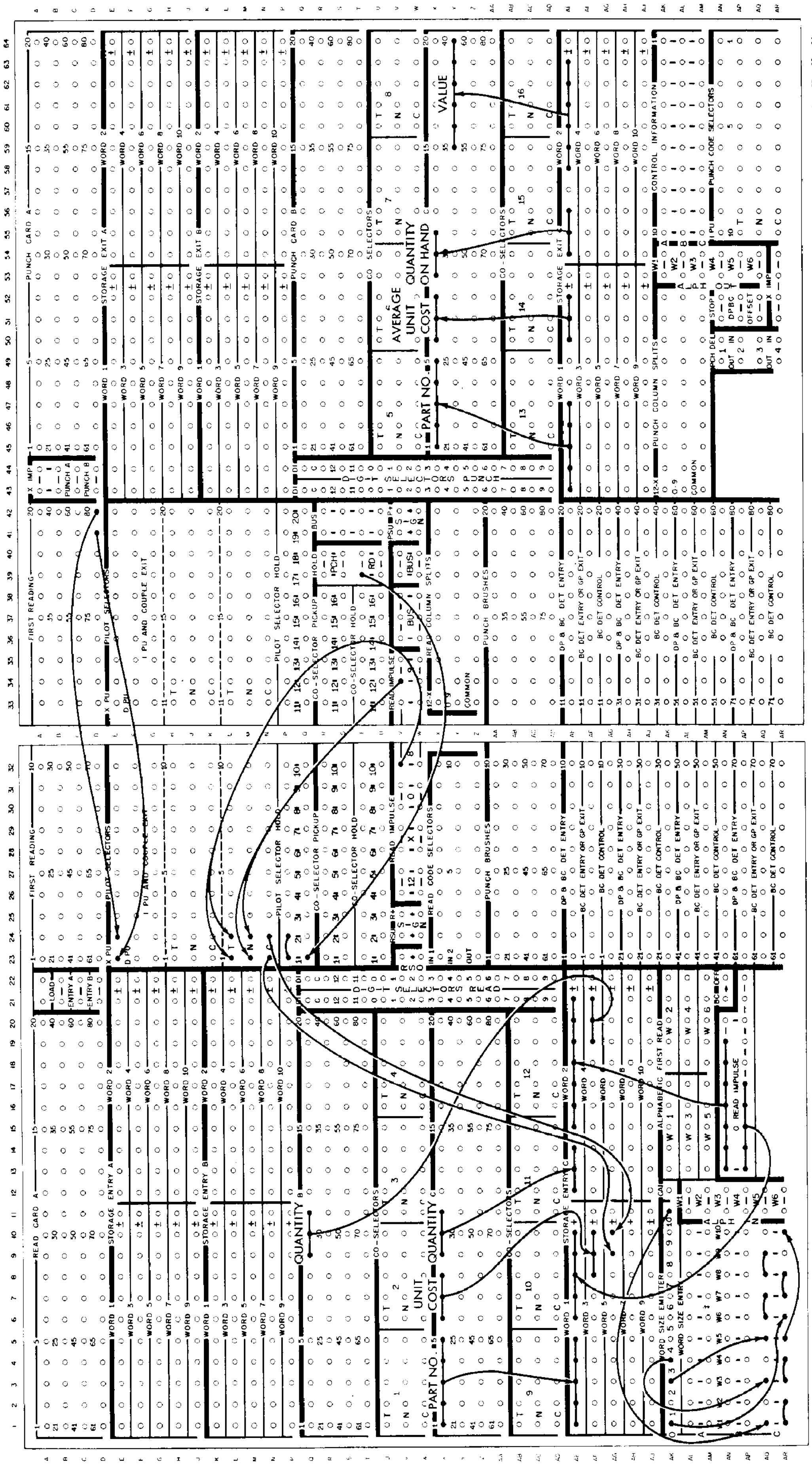
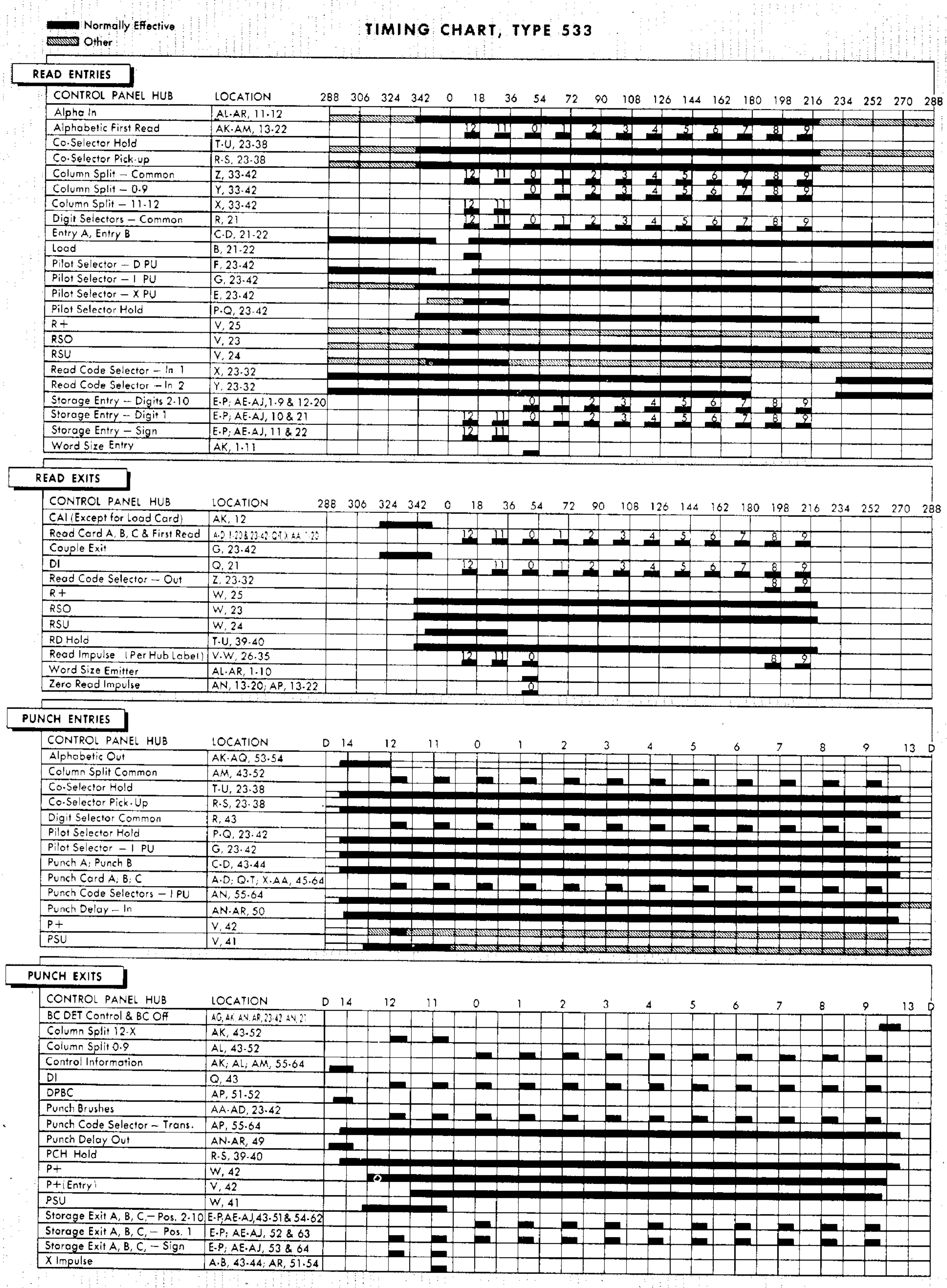


Figure 76. Inventory Control



READ ENTRIES

CONTROL PANEL HUB	LOCATION
Alpha In	AL-AR, 11-12
Alphabetic First Read	AK-AM, 13-22
Co-Selector Hold	T-U, 23-38
Co-Selector Pick-up	R-S, 23-38
Column Split — Common	Z, 33-42
Column Split — 0-9	Y, 33-42
Column Split — 11-12	X, 33-42
Digit Selectors — Common	R, 21
Entry A, Entry B	C-D, 21-22
Load	B, 21-22
Pilot Selector — D PU	F, 23-42
Pilot Selector — I PU	G, 23-42
Pilot Selector — X PU	E, 23-42
Pilot Selector Hold	P-Q, 23-42
R+	V, 25
RSO	V, 23
RSU	V, 24
Read Code Selector — In 1	X, 23-32
Read Code Selector — In 2	Y, 23-32
Storage Entry — Digits 2-10	E-P; AE-AJ, 1-9 & 12-20
Storage Entry — Digit 1	E-P; AE-AJ, 10 & 21
Storage Entry — Sign	E-P; AE-AJ, 11 & 22
Word Size Entry	AK, 1-11

READ EXITS

CONTROL PANEL HUB	LOCATION
CAI (Except for Load Card)	AK, 12
Read Card A, B, C & First Read	A-D, 1-20 & 23-42; Q-T; X-AA, 1-20
Couple Exit	G, 23-42
DI	Q, 21
Read Code Selector — Out	Z, 23-32
R+	W, 25
RSO	W, 23
RSU	W, 24
RD Hold	T-U, 39-40
Read Impulse (Per Hub Label)	V-W, 26-35
Word Size Emitter	AL-AR, 1-10
Zero Read Impulse	AN, 13-20; AP, 13-22

PUNCH ENTRIES

CONTROL PANEL HUB	LOCATION
Alphabetic Out	AK-AQ, 53-54
Column Split Common	AM, 43-52
Co-Selector Hold	T-U, 23-38
Co-Selector Pick-Up	R-S, 23-38
Digit Selector Common	R, 43
Pilot Selector Hold	P-Q, 23-42
Pilot Selector — I PU	G, 23-42
Punch A; Punch B	C-D, 43-44
Punch Card A; B; C	A-D; Q-T; X-AA, 45-64
Punch Code Selectors — I PU	AN, 55-64
Punch Delay — In	AN-AR, 50
P+	V, 42
PSU	V, 41

PUNCH EXITS

CONTROL PANEL HUB	LOCATION
BC DET Control & BC Off	AG, AK, AN, AR, 23-42 AN, 21
Column Split 12-X	AK, 43-52
Column Split 0-9	AL, 43-52
Control Information	AK; AL; AM, 55-64
DI	Q, 43
DPBC	AP, 51-52
Punch Brushes	AA-AD, 23-42
Punch Code Selector — Trans.	AP, 55-64
Punch Delay Out	AN-AR, 49
PCH Hold	R-S, 39-40
P+	W, 42
P+ (Entry)	V, 42
PSU	W, 41
Storage Exit A, B, C, — Pos. 2-10	E-P, AE-AJ, 43-51 & 54-62
Storage Exit A, B, C, — Pos. 1	E-P; AE-AJ, 52 & 63
Storage Exit A, B, C, — Sign	E-P; AE-AJ, 53 & 64
X Impulse	A-B, 43-44; AR, 51-54

FIGURE 77. TYPE 533 IMPULSE TIMING CHART

OPTIMUM PROGRAMMING

THE TYPE 650 has been designed with ease of programming as one of the primary considerations. As a result, the programmer is not burdened with timing restrictions that must be considered throughout the planning of a program. Interlocks within the machine make it impossible to violate timing conditions and cause the machine to give erroneous results. It is important, however, for the programmer to realize that the simplest method of programming (using sequential drum locations for succeeding instructions) causes the machine to waste a large amount of time in nonproductive waiting or searching.

In many cases it will be desirable to increase the productive work performed by the machine in the time available between successive read and/or punch operations. Proper recognition and analysis of the sequence of events occurring within the machine will indicate how one might significantly increase the effective speed by the proper location of data and instructions on the drum surface. Optimum programming is the technique by which data and instructions are located in such a manner as to minimize or eliminate, if possible, nonproductive waiting or searching time.

The basic time element of interest to the programmer is the time required to read one word. This is known as a *word time.* Because there are fifty words in each band of general storage, a word time is equal to 1/50 of the time required for a drum revolution. The drum revolves at 12,500 revolutions per minute; therefore, a word time is equal to .096 milliseconds. In order to establish a set of rules for optimum programming, each operation has been analyzed to determine the number of word times required for its interpretation and execution.

From a timing viewpoint, there are forty drum locations (one per band, assuming a 2000-word machine), which are equivalent. Therefore, only the angular location or the position between the first and fiftieth locations in a band must be considered in optimum programming. The band makes no difference. For example, any of the addresses 0003, 0053, 0103, 0153, ... , 0953, 1003, 1053, ... etc., along the length of the drum could be used without any loss of time.

Because the accumulator contains twenty positions, it takes two word times to read its contents. The internal timing synchronizes the upper half of the accumulator (8003) to *odd* word times (odd locations), and the lower half of the accumulator (8002) to *even* word times (even locations). For this reason, when an operation requires the use of the accumulator, it may be necessary to wait for an even word time, because the entire accumulator must be read starting with the lower half.

The D and I address system used in the Type 650 makes possible optimum programming, because it allows data and instructions to be placed in any location on the drum. As just mentioned, once an optimum location has been determined, there are thirty-nine additional locations that are equally accessible with respect to time.

Another factor of importance is that the I-address of many instructions can be optimum within a range of locations. This is because of the 650's ability to overlap arithmetical execution with the search for the next instruction. In most cases, it will not be practical to attempt to optimum-program such operations as multiplication, division, and table look-up due to the wide variation of factors.

Sequence Chart

The sequence chart (Figure 78) shows the steps taken in the interpretation and execution of the various types of operations and the word times required for each step. In several cases the sequence chart branches into two parallel paths; this is to show that two functions are being performed simultaneously. One of the parallel branches represents the arithmetical process, and the other, the process of obtaining the next instruction.

Analyzing the steps carried out by the machine in the performance of any operation, one can see that in every case certain fundamental word times are required. Begin consideration of each instruction at the time the instruction has been located. Starting at this point, the first word time of every operation is used to transfer the instruction from its memory location into the program register. The next word time is used to transfer the operation code to the

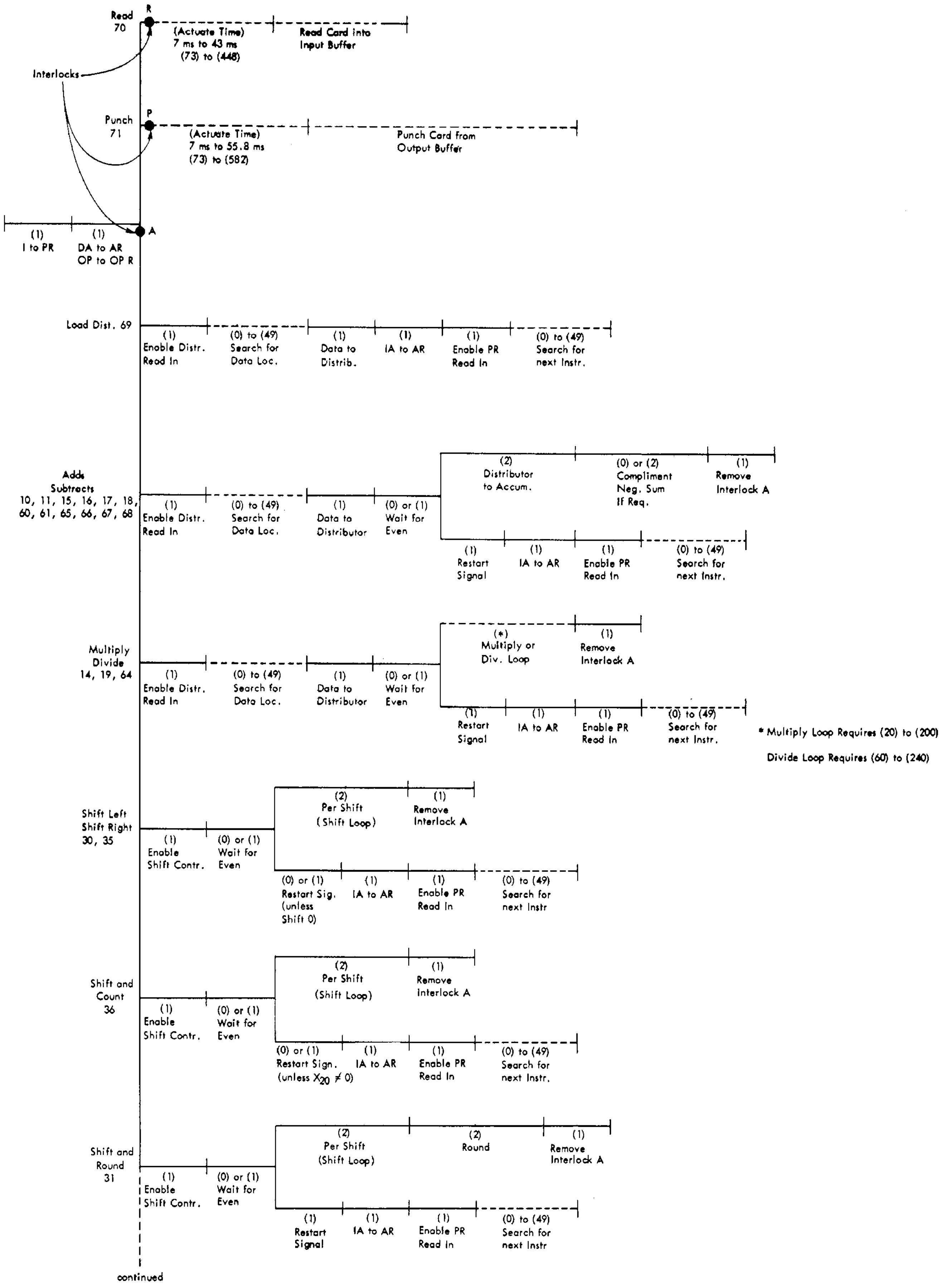
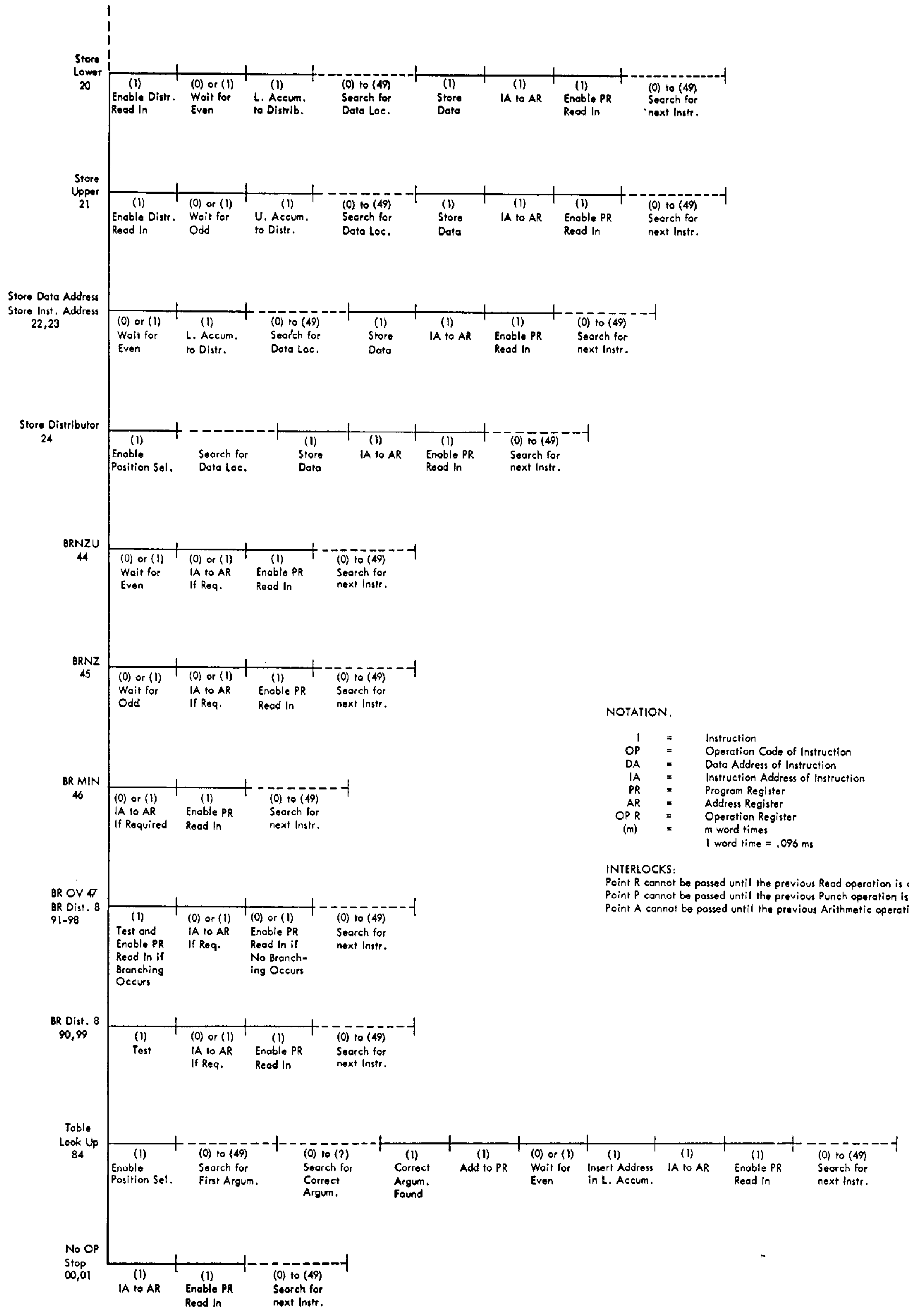

Read 70
R
(Actuate Time) 7 ms to 43 ms (73) to (448)
Read Card into Input Buffer
Interlocks
Punch 71
P
(Actuate Time) 7 ms to 55.8 ms (73) to (582)
Punch Card from Output Buffer
(1) I to PR
(1) DA to AR OP to OP R
A
Load Dist. 69
(1) Enable Distr. Read In
(0) to (49) Search for Data Loc.
(1) Data to Distrib.
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
Adds Subtracts 10, 11, 15, 16, 17, 18, 60, 61, 65, 66, 67, 68
(1) Enable Distr. Read In
(0) to (49) Search for Data Loc.
(1) Data to Distributor
(0) or (1) Wait for Even
(2) Distributor to Accum.
(0) or (2) Compliment Neg. Sum If Req.
(1) Remove Interlock A
(1) Restart Signal
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
Multiply Divide 14, 19, 64
(1) Enable Distr. Read In
(0) to (49) Search for Data Loc.
(1) Data to Distributor
(0) or (1) Wait for Even
(*) Multiply or Div. Loop
(1) Remove Interlock A
(1) Restart Signal
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
* Multiply Loop Requires (20) to (200)
Divide Loop Requires (60) to (240)
Shift Left Shift Right 30, 35
(1) Enable Shift Contr.
(0) or (1) Wait for Even
(2) Per Shift (Shift Loop)
(1) Remove Interlock A
(0) or (1) Restart Sig. (unless Shift 0)
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr
Shift and Count 36
(1) Enable Shift Contr.
(0) or (1) Wait for Even
(2) Per Shift (Shift Loop)
(1) Remove Interlock A
(0) or (1) Restart Sign. (unless $X_{20} \neq 0$)
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
Shift and Round 31
(1) Enable Shift Contr.
(0) or (1) Wait for Even
(2) Per Shift (Shift Loop)
(2) Round
(1) Remove Interlock A
(1) Restart Signal
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr
continued

FIGURE 78. TYPE 650 SEQUENCE CHART FOR OPTIMUM PROGRAMMING


Store Lower 20
(1) Enable Distr. Read In
(0) or (1) Wait for Even
(1) L. Accum. to Distrib.
(0) to (49) Search for Data Loc.
(1) Store Data
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
Store Upper 21
(1) Enable Distr. Read In
(0) or (1) Wait for Odd
(1) U. Accum. to Distr.
(0) to (49) Search for Data Loc.
(1) Store Data
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
Store Data Address Store Inst. Address 22,23
(0) or (1) Wait for Even
(1) L. Accum. to Distr.
(0) to (49) Search for Data Loc.
(1) Store Data
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
Store Distributor 24
(1) Enable Position Sel.
Search for Data Loc.
(1) Store Data
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
BRNZU 44
(0) or (1) Wait for Even
(0) or (1) IA to AR If Req.
(1) Enable PR Read In
(0) to (49) Search for next Instr.
BRNZ 45
(0) or (1) Wait for Odd
(0) or (1) IA to AR If Req.
(1) Enable PR Read In
(0) to (49) Search for next Instr.
BR MIN 46
(0) or (1) IA to AR If Required
(1) Enable PR Read In
(0) to (49) Search for next Instr.
BR OV 47 BR Dist. 8 91-98
(1) Test and Enable PR Read In if Branching Occurs
(0) or (1) IA to AR If Req.
(0) or (1) Enable PR Read In if No Branching Occurs
(0) to (49) Search for next Instr.
BR Dist. 8 90,99
(1) Test
(0) or (1) IA to AR If Req.
(1) Enable PR Read In
(0) to (49) Search for next Instr.
Table Look Up 84
(1) Enable Position Sel.
(0) to (49) Search for First Argum.
(0) to (?) Search for Correct Argum.
(1) Correct Argum. Found
(1) Add to PR
(0) or (1) Wait for Even
(1) Insert Address in L. Accum.
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.
No OP Stop 00,01
(1) IA to AR
(1) Enable PR Read In
(0) to (49) Search for next Instr.

NOTATION.

I	=	Instruction
OP	=	Operation Code of Instruction
DA	=	Data Address of Instruction
IA	=	Instruction Address of Instruction
PR	=	Program Register
AR	=	Address Register
OP R	=	Operation Register
(m)	=	m word times
		1 word time = .096 ms

INTERLOCKS:
Point R cannot be passed until the previous Read operation is complete
Point P cannot be passed until the previous Punch operation is complete
Point A cannot be passed until the previous Arithmetic operation is complete

FIGURE 78. TYPE 650 SEQUENCE CHART (CONTINUED)

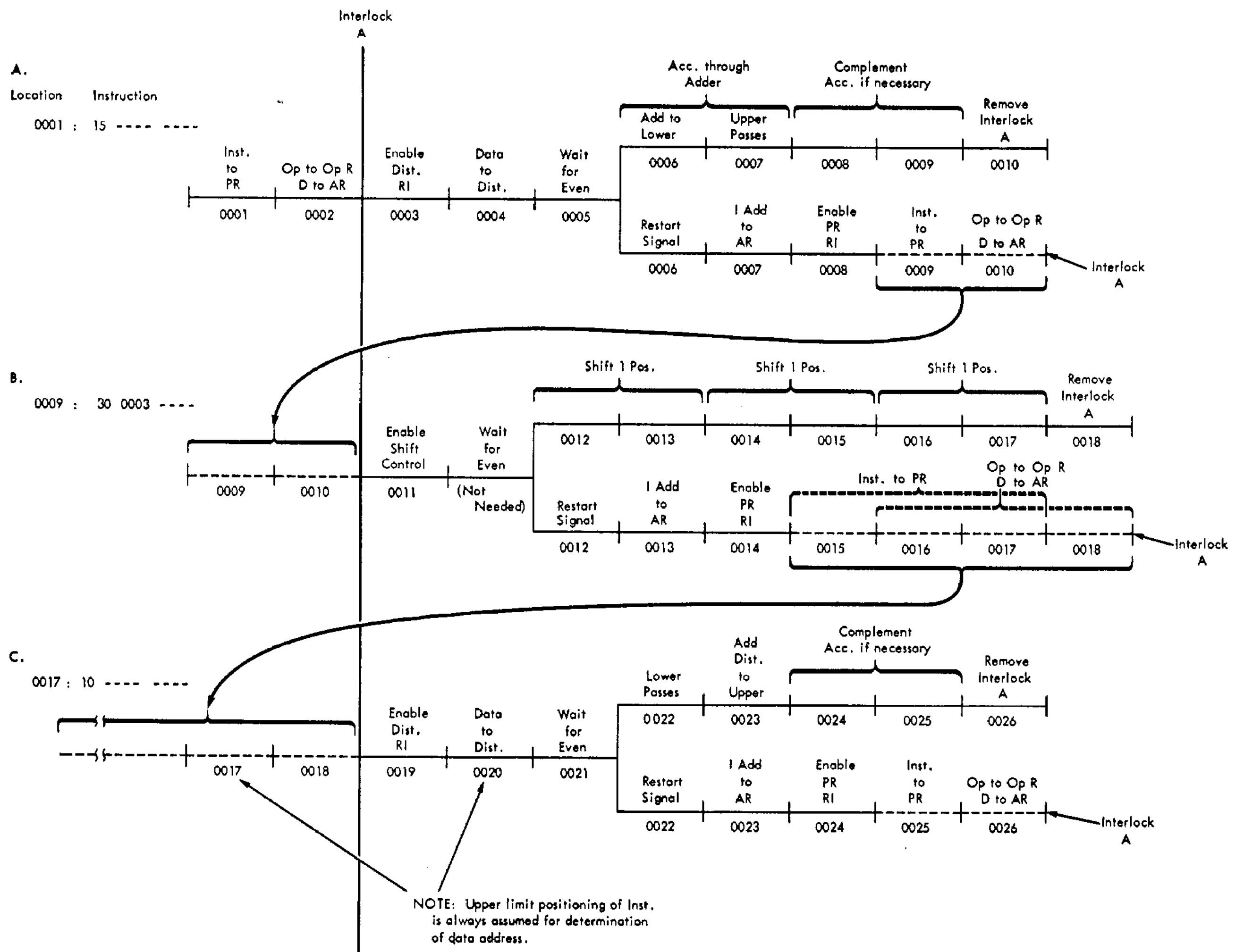


Figure 79. Sequence Chart

operation register, and the data address, to the address register. The steps performed during the word times beyond this point will depend upon the particular instruction being executed.

Consider, for example, an instruction calling for an add lower (15) operation. The sequence chart for this particular operation is shown in Figure 79A. Assume that the add lower instruction is in location 0001. Then, as location 0001 passes the reading heads, the instruction is read into the program register (1 word time). As location 0002 passes the reading heads, the operation code (15) is transferred to the operation register, and the data address (XXXX) is transferred to the address register. As location 0003 passes the reading heads, the control circuits necessary to perform addition are set up, and the distributor is signaled to read in. When the location specified by the data address passes the reading heads, its contents will be read into the distributor. Therefore, if the data address of the add lower instruction had been 0004, the search time for the factor to be added would be zero. Thus, location 0004, or any of the 39 other locations having the same angular position, would be the optimum location for data in this example.

Assuming the data is in location 0004, the factor to be added will enter the distributor as location 0004 passes the reading heads. The machine is now ready to add the factor in the distributor to the contents of the lower half of the accumulator. Because location 0005 is ready to pass the reading heads, the machine is starting an odd word time.

The accumulator contents can only be read into the one- digit adder beginning with an even word time. For this reason, nothing is accomplished as location 0005 passes the reading heads. As location 0006 passes the reading heads, the contents of the distributor are added into the lower half of the accumulator. Simultaneously with the actual addition operation, a restart signal is given to the program control system. During the time location 0007 is passing the reading heads, the contents of the upper half of the accumulator are passing through the adder, and the instruction address is transferred to the address register.

If no complementing of the accumulator contents is necessary, the arithmetical interlock A is removed as location 0008 passes the reading heads, and the program register is readied to read in the next instruction. Therefore, if the instruction address of the add lower instruction had been 0009, the search time for the next instruction would be zero. Thus, location 0009, or any of the other 39 locations having the same angular position, would be the optimum location for the next instruction in this example.

If it is necessary to complement the accumulator contents, the over-all timing is not affected. Under this condition, the simultaneous operation continues while the next instruction is read into the program register, and the operation code and data address are transferred into the operation and address registers. Thus, the next instruction in process reaches the interlock point just as the interlock is removed.

The foregoing add operation and the location of the next instruction require eight word times (.768 milliseconds) for execution when optimum programming is used. The same two functions could require as much as 106 word times (10.176 milliseconds) if the data and next instruction were placed in the worst possible locations. Of course, the factors would rarely fall in such poor relationship to each other; but when sequential locations for instructions are used, it can easily be seen that each operation and location of the next instruction must take at least 51 word times. For this reason, it is easy to gain speed increases of 3 to 1 or better by the use of rough optimum programming. Such significant speed increases will often be well worth the small amount of additional effort required.

Referring to the sequence chart again, one can see that multiplication, division, and shifting operations may vary in the number of word times required for the arithmetical function. These operations offer latitude for positioning the succeeding instruction depending upon the time required for the multiply, divide, or shift function.

To continue with the example, assume that the instruction found in location 0009 (Figure 79B) calls for a shift right of three positions. The instruction is read into the program register as location 0009 passes the reading heads. During the time location 0010 passes the reading heads, the operation code and data address are transferred to the operation and address registers. Because no drum data location is used in a shift operation, the only concern of the programmer is in positioning the following instruction.

As location 0011 passes the reading heads, the shift control is readied. Accumulator read-out can begin immediately, because location 0012 (even word time) is ready to pass the reading heads. Parallel operations begin at this point. During word time 0012, actual shifting is begun, and the restart signal is given to the control unit. The next word time (0013) is used to complete a shift of one position and to transfer the instruction address to the address register. During word time 0014, the second position of shifting is begun, and the program register is signaled to receive the next instruction.

At this point the latitude for positioning the succeeding instruction enters the picture. The next instruction could be in location 0015, which would be the lower limit for optimum location. However, the shifting operation is not completed until word 0017, and interlock A is not removed until 0018. Therefore, the instruction could be placed in location 0016 or 0017 and still have the new operation code and data address in the operation and address registers by the end of word time 0018 when the interlock is removed. Thus, location 0017 is the upper limit for optimum location of the instruction. Any of the three locations 0015, 0016, or 0017 will give zero access time for obtaining the instruction.

The data and instruction addresses, 0020 and 0025, respectively, (Figure 79C) of the instruction which follow are chosen as if the upper limit had been used. This is done because the arithmetical function determines when the operation can continue beyond interlock A.

LOCATION OF INSTRUCTION	OPERATION		ADDRESS	
	ABBRV.	CODE	DATA	INSTRUCTION
0001	AL	15	?	?
?	SRT	30	0003	?
?	AU	10	?	?

LOCATION OF INSTRUCTION	OPERATION		ADDRESS	
	ABBRV.	CODE	DATA	INSTRUCTION
0001	AL	15	0004	0009
0009	SRT	30	0003	0015
0015	AU	10	0020	0025

FIGURE 80. OPTIMUM PROGRAMMING

Thus, the original skeleton program can be filled in as shown in Figure 80.

Read or punch operations do not require the use of any of the arithmetical units; therefore, interlock A is bypassed by read or punch instructions. Calculating is delayed only until a signal is received from the read or punch feed to indicate that the feed has started to move. Once this signal is received, calculating can continue. The interlocks on a read or punch operation affect only succeeding read or punch operations. Interlock R (read) prevents a second read operation from starting until the current read operation is completed. Interlock P functions in the same manner with respect to punch operations.

Optimum Programming Rules

A set of rules has been developed for optimum programming based on the analysis shown in the sequence chart. These rules have been arranged in table form for easy reference. The table is divided into several sections, each of which covers operations that use similar rules.

One section (Figure 81) covers all the addition, subtraction, multiplication, division, load distributor, and store operations.

For any of the foregoing operations, the table is used in the following manner:

1. If the location (n) of the instruction is even, the number to be added to the location (n) to determine the data address (d) is found in the column headed *n even* and opposite the operation specified by the instruction.

2. When the data address (d) has been determined, the number to be added to the data address (d) to determine the instruction address (i) may be found opposite the operation in the proper column headed *d even* or *d odd.*

3. When the instruction address (i) has been determined, the location (n) of the next instruction has also been determined.

Thus, with an operation code 15 in location 0001 (n), the data address (d) equals n + 3, or 0004. The instruction address (i) equals d + 5, or 0009. These are exactly the same as the addresses determined in the example using the sequence chart.

Another section of the table covers all shift operations (Figure 82).

CODE

n. Location of instruction	Σm. Sum of multiplier digits
d. Data address of instruction	Σq. Sum of quotient digits
i. Instruction address of instruction	

Operation	n even d = n+	n odd d = n+	d even i = d+	d odd i = d+
Add, subtract, etc., 10, 11, 15, 16, 17, 18, 60, 61, 65, 66, 67, 68	3	3	5	4
Multiply, 19	3	3	21+2Σm	20+2Σm
Divide, 14, 64	3	3	61+2Σq	60+2Σq
Store Lower Accumulator, 20	5	4	3	3
Store Upper Accumulator, 21	4	5	3	3
Load Dist., Store Dist., 69,24	3	3	3	3
Store D Address, Store I Address, 22, 23	3	4	3	3

FIGURE 81. OPTIMUM PROGRAMMING RULES

CODE
i_l: lower limit for i
i_u: upper limit for i

Shift Operations	No. of Positions	n even $i_l = n+$	n even $i_u = n+$	n odd $i_l = n+$	n odd $i_u = n+$
Shift Right, 30; Shift Left, 35; Shift and Count, 36	(0)	6	6	5	5
	(1)	7	7	6	6
	(2)	7	7	6	6
	(3)	7	9	6	8
	(4)	7	11	6	10
	(5)	7	13	6	12
	(6)	7	15	6	14
	(7)	7	17	6	16
	(8)	7	19	6	18
	(9)	7	21	6	20
Shift and Count Only, 36	(10)	7	23	6	22
Shift and Round, 31	(1)	7	7	6	6
	(2)	7	9	6	8
	(3)	7	11	6	10
	(4)	7	13	6	12
	(5)	7	15	6	14
	(6)	7	17	6	16
	(7)	7	19	6	18
	(8)	7	21	6	20
	(9)	7	23	6	22
	(10)	7	25	6	24

FIGURE 82. OPTIMUM PROGRAMMING RULES

The table shows the word times to be added to the location of the shift instruction to determine the instruction address. Both the lower (i_l) and the upper (i_u) limits are stated. With a shift right operation code and a data address of 0003, the instruction address (i_l: lower limit) equals n + 6, or 0015. The upper limit (i_u) equals n + 8, or 0017. These are exactly the same as the address limits determined in the example using the sequence chart.

Two additional sections of the table give the information for determining d and i for the various branching operations and for table lookup (Figure 83).

a: Location of argument found by TLU operation.

Operation	n even $d = n+$	n odd $d = n+$	a even $i = a+$	a odd $i = a+$
Table Look-up, 84	3	3	5	6

Operation	n even $d = n+$	n odd $d = n+$	n even $i = n+$	n odd $i = n+$
Branch Non-Zero Upper, 44	3	4	4	5
Branch Non-Zero, 45	4	3	5	4
Branch Minus, 46	3	3	4	4
Branch Overflow, Branch Dist. 8, 47, 91-98	3	3	5	5
Branch Dist. 8, 90, 99	4	4	5	5
No Operation, Stop, 00, 01	—	—	4	4

FIGURE 83. OPTIMUM PROGRAMMING RULES

800X Addresses

Because the storage-entry switches (8000), distributor (8001), lower accumulator (8002), and upper accumulator (8003) are addressable, special consideration must be given to the cases where one of these locations is addressed. The storage-entry switches and distributor are always immediately accessible. However, the lower accumulator can be read into or out of only during an even word time. The upper accumulator can be read into or out of only during an odd word time. Thus, for purposes of optimum programming, the storage-entry switches and the distributor may be treated as being equivalent to any address, the lower accumulator as equivalent to an even address, and the upper accumulator as equivalent to an odd address.

If, for example, a reset-add upper operation (or any add-type operation) has a data address of 8000 or 8001, these addresses can be treated as being equivalent to n + 3 and the instruction address of that instruction can be determined accordingly. If a data address of 8002 were used, and the instruction were in an even location, the optimum location for the datum for that instruction would be n + 3, which is odd in this case. Therefore, an extra cycle must be taken to wait for an even location so that the lower accumulator may be read out. Because the effective data address is then even, the rule i = d + 5 must be used to determine the location of the next instruction. Similar analyses may be made for each of the other cases.

The following table (Figure 84) gives the rules for determining the instruction address of an instruction in cases in which a data address of 800X is used:

Because the accumulator may be in use when an instruction address of 8002 or 8003 is given, it would seem possible to take the next instruction from the accumulator before the arithmetic operation was completed. For example, if an 8002 instruction address were used on a multiply operation, one of the partial products might be taken as the next instruction rather than the final product. For this reason, an additional interlock is provided to prevent the next instruction from being taken from an 800X address until completion of the arithmetic portion of the operation. This prevents such an error from occurring.

The following table (Figure 85) gives the rules for determining the effective instruction address in those cases in which an instruction address of 800X is used.

Operation	Data Address	n even i = n+	n odd i = n+
Add, Subtract, etc., (10, 11, 15, 16, 17, 18, 60, 61, 65, 66, 67, 68)	8000	7	8
	8001	7	8
	8002	9	8
	8003	7	8
Load Distributor, 69	8000	6	6
	8001	6	6
	8002	7	6
	8003	6	7

FIGURE 84. OPTIMUM PROGRAMMING RULES

Operation	Inst. Address	d even i~d+	d odd i~d+
Add, Subtract, etc. (10, 11, 15, 16, 17, 18, 60, 61, 65, 66,67, 68) No Complement Cycle Required	8000	5	4
	8001	5	4
	8002	6	5
	8003	5	4
Add, Subtract, etc. (10, 11, 15, 16, 17, 18, 60, 61, 65, 66, 67, 68) Complement Cycle Required	8000	7	6
	8001	7	6
	8002	8	7
	8003	7	6
Load Distributor, 69	8000	3	3
	8001	3	3
	8002	4	3
	8003	3	4

~ means "equivalent to"

FIGURE 85. OPTIMUM PROGRAMMING RULES

By application of the principles used in determining the above-mentioned rules it is possible to determine equivalent addresses for any other desired case. It is necessary to remember only that an 8002 address is equivalent to an even drum address, and 8003 is equivalent to an odd drum address.

Consider the instruction 10 0643 8000. Because 0643 is odd, the rule in d + 4 must be used. Therefore, the instruction at 8000 can be programmed as though it were located in location 0647.

A wallet-sized card (Form 22-6219) is available which shows the tables for optimum programming and a complete list of the operation codes.

Figure 86 shows a portion of an actual payroll problem. This has been included to help show how optimum programming might be used.

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS	LOCATION OF INSTRUCTION	OPERATION		ADDRESS	
	ABBRV.	CODE	DATA	INSTRUCTION			ABBRV.	CODE	DATA	INSTRUCTION
	Sequential							Optimum		
0013	RAL	65	1815	0014	Add Constant Instruction	0013	RAL	65	1916	0021
0014	LD	69	1802	0015	Load Dist. with Man No.	0021	LD	69	1802	0005
0015	TLU	84	0600	0016	On Man No. Table	0005	TLU	84	0600	0006
0016	LD	69	1812	0017	Load Dist. with Zeros	0006	LD	69	8003	0012
0017	STDA	22	0449	8002	Store Selected Data Address	0012	STDA	22	0515	8002
(8002	RAL	65	xxxx	0018)	Add Selected Man No. and Assoc. Data	(8002	RAL	65	xxxx	0025)
0018	SRT	30	0005	0019	Shift Off Dept. No.	0025	SRT	30	0005	0037
0019	SLT	35	0005	0020	Shift Selected Man No.	0037	SLT	35	0005	0049
0020	SL	16	1802	0021	Sub. Actual Man No.	0049	SL	16	1802	0007
0021	BRNZ	45	0022	0023	Branch if not in Table	0007	BRNZ	45	0010	0011
0022	STOP	01	0004	8000	No Master for this Man No.	0010	STOP	01	0004	8000
0023	RAL	65	1816	0024	Add Constant Instruction	0011	RAL	65	1914	0019
0024	AL	15	0449	8002	Insert Data Address	0019	AL	15	0022	8002
(8002	RAU	60	xxxx	0025)	Add Selected Data to Upper	(8002	RAU	60	xxxx	0035)
0025	SLT	35	0001	0026	Shift off Tax Cl.	0035	SLT	35	0001	0041
0026	SRT	30	0001	0027	Shift Rate to Right	0041	SRT	30	0001	0047
0027	STU	21	0498	0028	Store Stand. Rate	0047	STU	21	0516	0017
0028	SU	11	1805	0029	Subt. Job Rate	0017	SU	11	1805	0009
0029	BRMIN	46	0031	0030	Branch if Job Rate Greater	0009	BRMIN	46	0062	0063
0030	LD	69	0498	0032	Load Dist. with Selected Rate	0063	LD	69	0516	0069
0031	LD	69	1805	0032	Load Dist. with Sel. Rate	0062	LD	69	1805	0069
0032	STD	24	0499	0033	Store Selected Rate	0069	STD	24	0522	0075
0033	RAU	60	0499	0034	Load Upper with Sel. Rate	0075	RAU	60	8001	0031
0034	MULT	19	1803	0035	Reg. Hrs. X Rate	0031	MULT	19	1803	0014
0035	SRD	31	0002	0036	Shift 2 and Round	0014	SRD	31	0002	0071
0036	STL	20	0598	0037	Store Job Card Reg. Earn.	0071	STL	20	0526	0029
0037	SLT	35	0005	0038	Shift Left 5 Pos.	0029	SLT	35	0005	0036
0038	AU	10	1822	0039	Add Constant Instruction	0036	AU	10	1939	0043
0039	AU	10	0449	8003	Modify Data Address	0043	AU	10	0515	8003
(8003	AL	15	xxxx	0040)	Accumulate Reg. Earnings	(8003	AL	15	xxxx	0026)
0040	STL	20	0549	0041	Store Word 3 in Temp. Storage	0026	STL	20	0531	0034
0041	RAU	60	0499	0042	Load Upper with Sel. Rate	0034	RAU	60	0522	0077
0042	MULT	19	1804	0043	Sel. Rate X O T Hrs.	0077	MULT	19	1804	0064
0043	SRD	31	0002	0044	Shift and Round 2 Pos.	0064	SRD	31	0002	0070
0044	STL	20	0599	0045	Store O T Earn for Job Cost Dist.	0070	STL	20	0525	0078
0045	AL	15	0549	0046	Create New Word 3	0078	AL	15	0531	0085
0046	AU	10	1817	0047	Add Constant Instruction	0085	AU	10	1938	0093
0047	AU	10	0449	8003	Modify Data Address	0093	AU	10	0515	8003
(8003	STL	20	xxxx	0054)	Store Word 3 in Table	(8003	STL	20	xxxx	0054)

FIGURE 86

PROBLEM: Square Root

$a = \sqrt{A}$, $0 < A < 1$

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV.	CODE	DATA	INSTRUCTION	
		SEQUENTIAL PROGRAMMING REQUIRES .321 SEC.			
0102	RAL	65	0200	0103	
0103	AU	10	8001	0104	$a_0 = (A + 1)\ /2$
0104	MULT	19	0201	0105	
0105	ST U	21	0202	0106	
0106	RAU	60	0200	0107	
0107	MULT	19	0201	0108	
0108	DIV RU	64	0202	0109	$a_{n+1} = (a_n + A/a_n)\ /2$
0109	AU	10	0200	0110	
0110	MULT	19	0202	0111	
0111	ST U	21	0203	0112	
0112	SU	11	0202	0113	Test $a_{n+1} - a_n$. If $a_{n+1} - a_n \geq 0$,
0113	BRNZU	44	0114	0101	transfer back to main routine (0101).
0114	BR MIN	46	0115	0101	
0115	LD	69	0203	0116	$a_{n+1} - a_n < 0$, thus, prepare to repeat
0116	ST D	24	0202	0106	iteration
0200:	1/2	.5000000000			
0201:	A				
0202:	a_n				
0203:	Temporary Storage				
0101:	Contains next instruction in main routine				
- - - - - -	- - - - - -	- - - -	- - - - -	- - - - -	- - - - - - - - - - - - - - - -
		OPTIMUM PROGRAMMING REQUIRES .152 SEC.			
0139	RAL	65	0142	0148	
0148	AU	10	8001	0105	$a_0 = (A + 1)\ /2$
0105	MULT	19	0108	0189	
0189	ST U	21	0144	0147	
0147	RAU	60	0100	0155	
0155	MULT	19	0108	0140	
0140	DIV RU	64	0144	0134	$a_{n+1} = (a_n + A/a_n)\ /2$
0134	AU	10	0137	0141	
0141	MULT	19	0144	0125	
0125	ST U	21	0130	0133	
0133	SU	11	0144	0120	Test $a_{n+1} - a_n$. If $a_{n+1} - a_n \geq 0$,
0120	BRNZU	44	0124	0128	transfer back to main routine (0128).
0124	BRMIN	46	0127	0128	
0127	LD	69	0130	0135	$a_{n+1} - a_n < 0$, thus, prepare to
0135	ST D	24	0144	0147	repeat iteration.
0142:	1/2 =	.5000000000			
0108:	A				
0144:	a_n				
0100:	1/2 =	.5000000000			
0137:	1/2 =	.5000000000			
0130:	Temporary Storage				
0128:	Contains next instruction in main routine				

FIGURE 87. SQUARE ROOT

SQUARE ROOT (Figure 87)

When programmed sequentially, this routine requires approximately .321 seconds (assuming five iterations). The same routine is shown programmed optimumly. The time required is reduced to .152 seconds. Thus, the increase in speed is a factor of 2.1 to 1. Note, however, that the optimumly programmed routine requires two additional storage locations so that the constant ½ will always be located optimumly. Experience has shown that the gain realized may be a factor of less than 2 to 1 to a factor as large as 6 or 7 to 1.

Techniques for Using Optimum Programming

The gains obtained by optimum programming will depend upon the particular problem considered and the skill of the programmer. Every instruction cannot be optimum, because conflict will occasionally exist between instructions. For example, data placed optimumly for a store operation may not be optimumly located for a later add operation. Furthermore, an instruction that is preceded by several branch instructions cannot usually be optimumly located for each of these branch instructions.

Techniques for efficient use of optimum programming will be developed through experience. It is too lengthy a task to explore thoroughly the techniques for its use in this manual. However, certain general principles can be applied to any program one may wish to consider. Some of these principles will be indicated in the following paragraphs.

The first consideration that must be made is: When should optimum programming be used? Any problem in which the speed of input or output is appreciably reduced because of lengthy calculations can justifiably be programmed optimumly.

Once it has been determined that optimum programming is necessary, there are two ways in which it can be done. One could simply program the problem optimumly by straightforward application of the rules derived previously without consideration of the program as a whole. Such a procedure may result in data and instructions being poorly located for the latter part of the problem. This type of programming can be done quickly and with very little more difficulty than if the instructions were programmed in sequence. Even though it is done roughly, it will usually result in a significant increase in the over-all speed, and will be well worth the small amount of additional effort required.

The second type of optimum programming is where the programming is done elegantly. This type of programming requires a greater amount of work and thought on the part of the programmer. Instead of simply programming each step optimumly as it occurs, the programmer must think ahead to see how this might possibly affect later steps that will use the same data or perhaps branch to the same instruction. Many possibilities exist when programming in this manner. One can see after he has completed the problem how a simple re-arrangement of the beginning may improve the latter part of the problem. It may be necessary to reprogram the same problem several times in order to obtain the most efficient program. Obviously such a procedure would require much more time than *sequential* programming or the *rough* optimum programming just described.

Elegant optimum programming should be used only when it is desirable to use the Type 650 in the most efficient manner possible. The types of problems for which this is necessary are those that must be done repeatedly on a mass basis and that realize reduced input-output speeds. It will not always be necessary to program the entire problem this way but only those segments or sub-routines that are most frequently used. For example, sub-routines, such as floating decimal operations, square root, sine, and cosine evaluations, which occur frequently in technical applications, should be programmed to function in the most efficient manner possible. Similar applications that will occur in commercial problems are extension from gross to net in payroll calculation, insurance dividend calculation, and bill computation in public utility customer accounting.

Computing During Input and Output Operations

At a card-punching rate of 100 cards per minute, up to 544 milliseconds are available for computing. This is approximately 5600 word times, or 110 drum revolutions. At a card reading speed of 200 cards per minute, up to 257 milliseconds are available for computing. This is approximately 2700 word times, or 54 drum revolutions.

LOADING ROUTINES

One of the first problems that arise in using a stored program machine is how to originally enter the instructions, constants, tables, etc., necessary to solve a problem. The flexibility of the input system of the Type 650 gives the programmer a wide range of possibilities. The routines that follow are merely three of the many approaches that might be used to solve the loading problem.

One Word Per Card

This routine enters one instruction (or word) from a card identified as a load card. Each card also contains two instructions that are used with the storage-entry switches (8000) to form the entire program necessary to load the instruction or word in the desired location. There is ample room in the card for indicative information and description.

Because each card is an entity, the cards may be arranged in any desired order, dependent, of course, on the extent of the indicative information. The cards can be sorted, collated, or listed in any desired manner for analyzing the program.

Because of its simplicity and flexibility, this method is suggested as the basic means for loading programs that are to be tested. Once a program has been tested and proved correct, it is a simple matter to convert the program to some multiple instruction per card form to save time in reloading. The original single-item cards also provide the easiest method of creating reproductions of the program through simple listings.

A standard card form (853862X) is available. This card is designed for keypunching directly from the planning chart (Form 22-6151). It provides space for the two instruction words needed in the loading program, ten columns of indicative information, the word to be loaded, the alphabetical abbreviation for the operation and descriptive remarks.

A sample program to load a single word per card is shown in Figure 88.

Once the read-in locations are chosen, the D-address of the read instruction is determined. It is the location into which the load distributor instruction from the load card has been read.

The D-address of the 69 instruction is the address of the location into which the word to be stored on the drum has been read. The I-address of this instruction is the address of the location into which the store distributor instruction has been read.

The D-address of the 24 instruction is the location into which the word is to be stored. The I-address is always 8000.

Multiple Words Per Card

It may be desirable after a program has been proved to reduce the number of cards necessary to load the program. This can be accomplished by placing several instructions in each load card. A card could contain up to eight instructions. Because it will usually be desirable to have some means of identification punched into all program load cards, something less than eight instructions per card is recommended.

An important factor to note is that any loading method using more than two instructions per card will require the storage of a loading routine on the drum.

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV.	CODE	DATA	INSTRUCTION	
8000	RD	70	1951	i	Set on Console. See Note #1.
1951	LD	69	1954	1953	Standard code in load card.
1953	STD	24	XXXX	8000	This word combines a standard store
					instruction with a variable data
					address. The data address is the
					location in which the instruction or
					factor is to be stored.
NOTE #1-- i is variable representing the address of the first instruction					
of the normal program.					

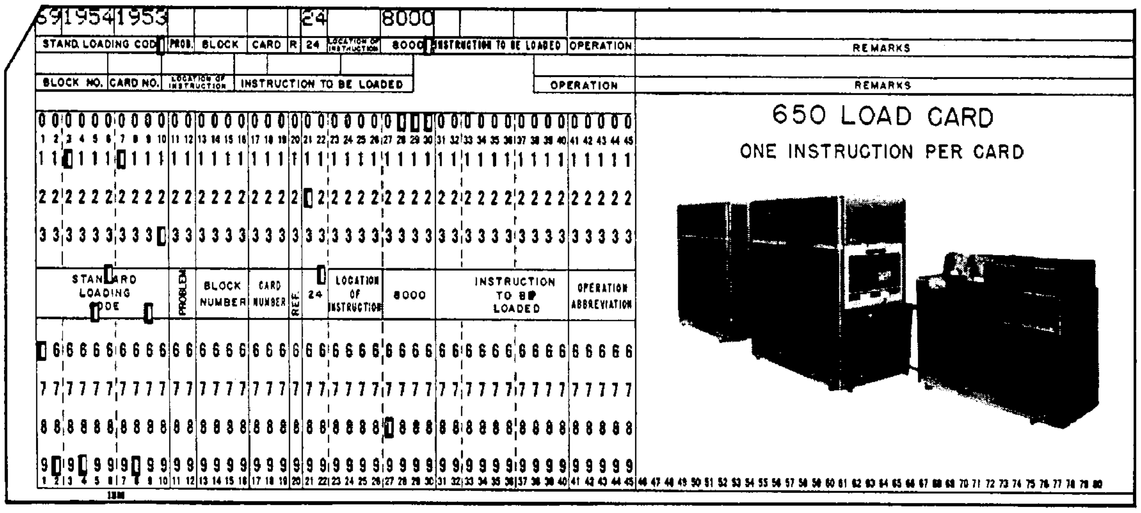


FIGURE 88

Four Words per Card

This routine enters four words from a card. In addition to the four words, the card contains the addresses of the locations into which the instructions are to be stored. The loading routine consists of nine instructions, five of which are stored on the drum, and four that are entered from each card as it is read. The program consists of a read instruction followed by four sets of load distributor-store distributor instructions. The four store distributor instructions are punched into the cards with the data address of each as the location into which the adjacent word is to be loaded.

The five instructions to be stored on the drum may be loaded from single-item load cards. A sixth card could then be used to effect the switch into the four-per-card routine just loaded.

After all the load cards have been processed, it is necessary to transfer to the main program just loaded. This can be accomplished in one of the following ways:

1. If the first card following the program deck is not a load card, the switch to the main program could be accomplished by using the I-address of the read instruction.
2. Probably a more useful method of switching to the main program is to change the I-address of the last store distributor instruction in the last program load card. The I-address used would simply be the location of the first instruction in the main program.

By proper choice of the I-address of the last store distributor instruction in any load card, cards containing one, two, or three instructions can also

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV.	CODE	DATA	INSTRUCTION	
1999	RD	70	1995	0000	Note I
1995	LD	69	1952	1951	Note I
1951	STD	24	xxxx	1996	Note II
1996	LD	69	1954	1953	Note I
1953	STD	24	xxxx	1997	Note II
1997	LD	69	1956	1955	Note I
1955	STD	24	xxxx	1998	Note II
1998	LD	69	1958	1957	Note I
1957	STD	24	xxxx	1999	Note II
					Note I: These instructions kept on the drum.
					Note II: These instructions are read into the 650 on each load card. The data addresses specify the location into which each of the four pieces of information is to be stored.

FIGURE 89

be processed by the four-per-card loading routine.

Figure 89 is an example of a routine for loading four instructions per card using locations 1951 to 1960 for read storage.

SEVEN WORDS PER CARD

This routine is used to load up to seven words from a single load card. The eighth word is used as a control word and is punched with the address into which the first word is to be entered and the number of words to be entered from that particular card. Because only one address is in each card, the words from each card are loaded into consecutive drum locations.

Figure 90 shows a routine to load up to seven instructions per card using the control word layout just mentioned.

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV.	CODE	DATA	INSTRUCTION	
1988	RD	70	1994	m	(Transfer control to location m if no
1994	RAL	65	1951	1979	load card)
1979	LD	69	1982	1986	
1986	ST DA	22	1982	1989	
1989	SLT	35	0004	1981	Prepare to store starting with
1981	AL	15	8001	1990	location f.
1990	ST DA	22	1978	1984	
1984	RAL	65	1991	1992	
1992	AU	10	1982	8002	
8002	LD	69	(1952)	8003	Store word i in location f + i - 1
8003	ST D	24	(f)	1977	
1977	AU	10	1980	1985	Modify instructions in preparation to
1985	AL	15	8001	1993	store word i + 1.
1993	SU	11	1978	1983	Test to see if word n - 1 has been
1983	BRNZU	44	1987	1988	stored. If so, read the next card.
1987	AU	10	8001	8002	Prepare to store word i + 1.
1982	ST D	24	0000	1977	
1991	LD	69	1952	8003	Constants
1980		00	0001	0000	
1978	ST D	24	f + n	1977	Temporary Storage
1951		00	f	000n	Control word punched in load card
					f is address where first word on card
					is to be entered. n is the number of
					words to be entered.

FIGURE 90

PROGRAMMING AND OPERATING SUGGESTIONS

Drum Clearing

The entire 2000 drum memory locations, or any portion thereof, can be cleared to zeros in less than twenty seconds. Clearing the drum memory to zeros before loading a program offers the following operating advantage:

If a stop instruction (operation 01) is stored in location 0000 during the loading of the program, a failure to load all the instructions of the program will cause the machine to stop. The stop will occur as soon as the first missing instruction is referred to once the program has been started. There will be zeros in any memory location where an instruction is missing. When this location is referred to, the zeros in positions 10 and 9 become a NO-OP code (00), and the machine will go immediately to the I-address of the instruction (positions 4-1). Because the I-address portion is also zeros, the address of the next instruction is 0000. Location 0000 contains a stop code and the machine will stop at that point if the programmed switch is set to STOP.

If a program load card were missing and the drum had not been set to zero, the location or locations that should have been loaded from the missing card might contain an instruction from a previously loaded program. When this instruction is encountered, it would be performed and the machine would then refer to the location specified by the I-address. This would probably upset the entire program.

There are, of course, other methods of insuring that none of the program load cards is missing. Two such methods are:

1. The load cards could be sequence checked.
2. A card count could be carried and checked by the operator against the known number of load cards.

Both of the foregoing checks would require programming during the loading routine.

Clear Entire Drum

Figure 91 shows a routine that can be used to clear effectively the entire drum to minus zeros. All in-

Location of Instruction	Instruction			Operation Abbrev.
	OP	Data	Instr.	
8000	70	0004	XXXX	RD
0004	61	0008	0007	RSU
0007	69	0006	0005	LD
0005	24	0000	8003	STD
8003	20	XXXX	0003	STL
0003	10	0001	8003	AU

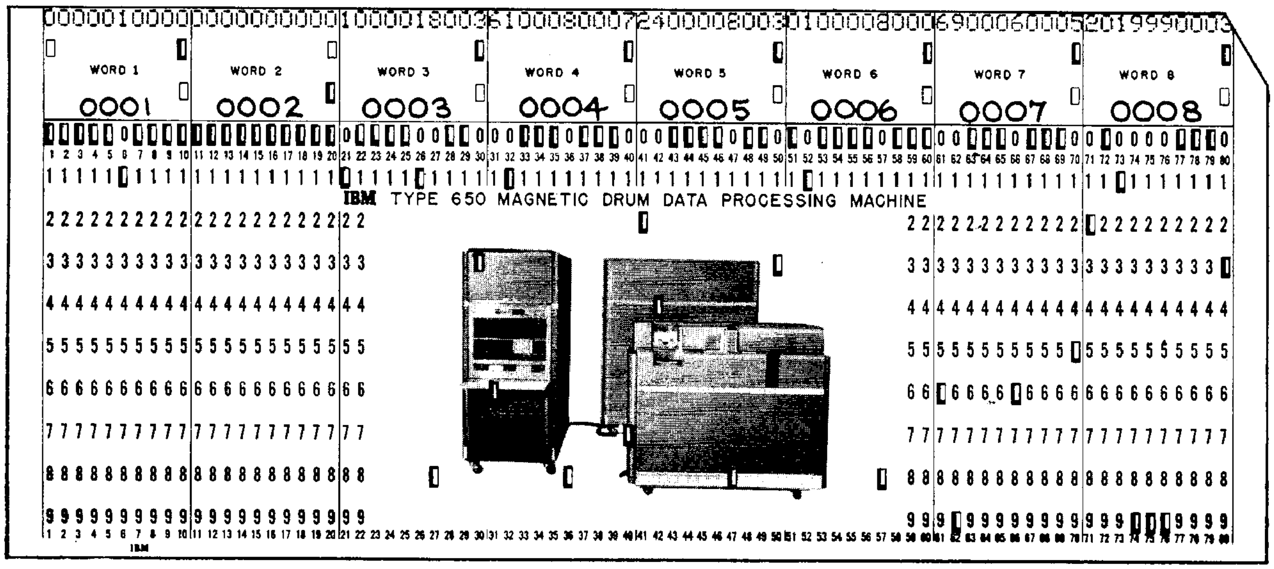


FIGURE 91. DRUM ZEROING ROUTINE

structions for this routine are contained in one load card. Minus zeros are used to differentiate between a location that was never loaded and a location that might be cleared to plus zero by a store instruction in the program.

This routine will clear all drum locations to zeros with the exceptions of location 0000, which will contain the constant 01 0000 8000, and location 0001, which will contain the constant 00 0001 0000.

The routine was set up in this manner so that any reference to location 0000 will cause the machine to stop. With the constant 00 0001 0000 in location 0001, any reference to this location will also cause the machine to stop (NO-OP and go to location 0000).

Clearing Between Limits

Figure 92 shows a drum-clearing routine that will clear the drum to zero between specified limits. The entire routine may be punched into a single card. The card is identified as a load card. This could be accomplished by using any of the eight sign punches as shown in the sample or by punching any other desired 12 in the card.

The limits are specified by the data address of words 2 and 6 (the lower limit in word 2 and the upper limit in word 6). In the example which follows, the limits are 0500 and 1999.

Location of Instruction	Instruction			Operation Abbrev.
	OP	Data	Instr.	
8000	70	0001	XXXX	RD
0001	60	0002	8003	RAU
8003	20	0500	0003	STL
0003	10	0004	0005	AU
0005	11	0006	0007	SU
0007	44	0008	8000	BRNZU
0008	10	8001	8003	AU

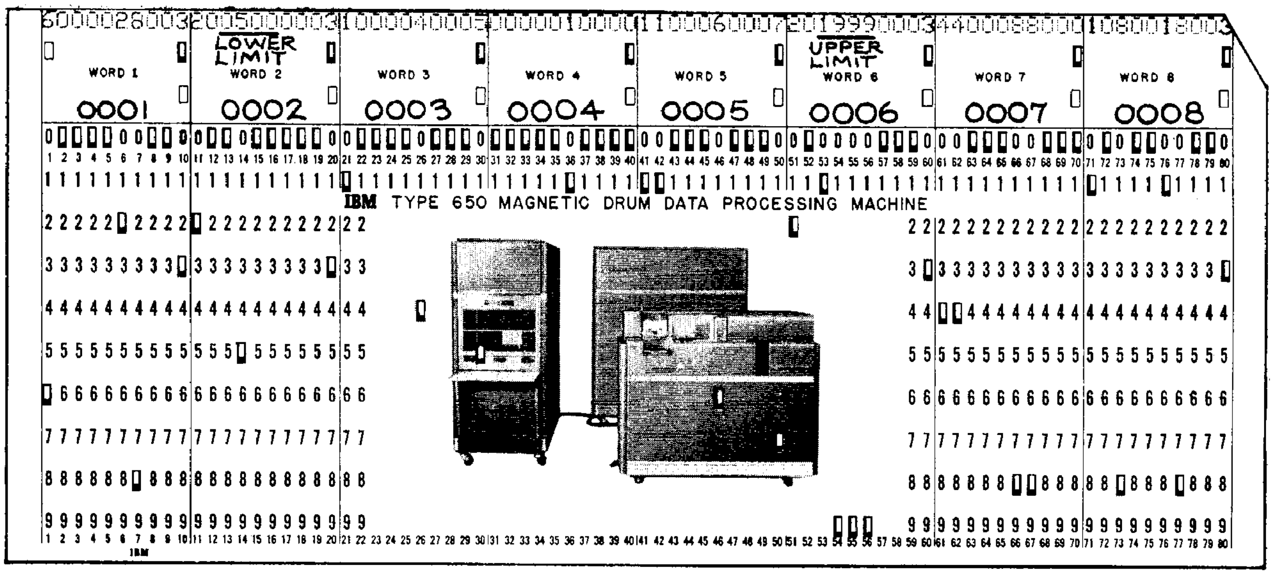


FIGURE 92. ZEROING DRUM BETWEEN LIMITS

Minus Zero

During the course of a problem, a word containing ten zeros with an associated sign of minus (a minus zero) may be developed in the upper half or lower half of the accumulator. The minus zero that is developed through programming can be stored in any drum memory location as a minus zero.

The problem of bringing the contents of a drum memory location that could be minus zero into either half of the accumulator with a resultant accumulator sign of minus for purposes of branch minus can be accomplished by the following method:

1. Accumulator must contain minus zero before operation begins.

2. Operation must be add upper (10) or add lower (15) from drum memory location containing the minus zero.

3. Accumulator results will be minus zero.

It is recognized that the foregoing method of bringing a minus zero from a drum memory location into the accumulator is rather cumbersome. A simpler method that will bring a minus zero into the accumulator is as follows:

1. Multiply the minus zero of the drum memory location by a plus (+) 1.

2. The result in the accumulator will be minus zero.

Because bringing a minus zero from drum memory into the accumulator with a resultant sign of minus is rather complicated, the most logical approach is to branch minus (46) when the minus zero is originally developed.

Branching

Because of its flexibility the Type 650 offers the programmer a wide choice of techniques with respect to branching on zero and minus values. *Loop* (series of instructions) routines require that some means of branching or instruction modification be utilized in order to determine whether the loop has been completed. When programming a routine of this type, be careful that the *end-of-loop* test does not miss by 1. For example, a loop is to be executed 300 times. If a constant of 300 is used in conjunction with a zero test (to determine the end of loop) and the test is made prior to executing the loop, the routine will be executed only 299 times.

The alternative here is to increase the constant to 301 or to make the end-of-loop test after the instructions within the loop have been executed. The following examples offer several approaches to this problem.

Figures 93, 94, 95 and 96 illustrate a *table-punch-out* routine. Each program performs the same operation; they differ only in testing and branching techniques.

The table occupies drum locations 0300 through 0625, exclusive of locations XX48, XX49, and XX98, XX99; therefore, provisions are made to effect skipping the last two locations of each band. Contained in each table location is part number and quantity arranged as follows:

PART NO.	QUANTITY
XXXXX	XXXXX

None of the examples is singled out as being the best method; they serve merely to illustrate various branching techniques.

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV.	CODE	DATA	INSTRUCTION	
1950	RAU	60	1972	1951	"D" Address = location to be punched
1951	SLT	35	0004	1952	Drop op. code & 2 high order positions
					of "D" address.
1952	SU	11	1970	1953	Test for location xx48.
1953	BRNZU	44	1954	(1965)	Branch if not location xx48
1954	SU	11	1971	1955	Test for location xx98
1955	BRNZU	44	1956	(1965)	Branch if not location xx98
1956	RAL	65	1972	8001	"D" address = location to be punched
8001	LD	69	xxxx	1957	Part # and quantity
1957	STD	24	0927	1958	Store for punching
1958	PCH	71	0927	1959	Punch
1959	AU	10	1973	1960	Add limit to upper accumulator
1960	SU	11	8002	1961	Subtract this location
1961	BRNZU	44	1963	1962	Branch if not last location
1962	STOP	01	0000	8000	Stop
1963	AL	15	1974	1964	Add 1 to "D" address
1964	STL	20	1972	1950	Store
1965	RAL	65	1972	1966	"D" address = xx48 or xx98
1966	AL	15	1975	1964	Add 2 to "D" address to
					increase to xx00 or xx50
					1970 - 48 1957 0000 - Constant
					1971 - 50 0000 0000 - Constant
					1972 - 69 xxxx 1957 - "N" Location
					1973 - 69 0625 1957 - Last Location
					1974 - 00 0001 0000 - Constant
					1975 - 00 0002 0000 - Constant

FIGURE 93. BRANCHING—NON-ZERO UPPER

Figure 93. The instructions in 1950 through 1955 test for locations XX48 and XX98 of a band. These instructions are repeated in each example and will not be discussed in the succeeding examples.

The instructions in 1956 through 1958 load, store, and punch a table value. The D-address of the instruction in location 1972 will be 0300 initially; thereafter, it will be increased until it has progressed to 0625.

The instructions in 1959 and 1960 set up 8003 for testing. The instruction in 1961 tests 8003 for a non-zero condition. If the last table location has been punched, 8003 will contain zeros, and the stop instruction in 1962 will be called for. If 8003 is non-zero, the instructions in 1963 and 1964 will add 1 to the D-address and store the modified instruction in 1972.

The instructions in 1965 and 1966 increase the D-address (table location) by 2. These instructions are called for when location XX48 or XX98 has been reached.

Figure 94. The instructions in 1956 and 1957 load and store the table values. The instruction in 1958 subtracts the high limit. Note that the limit, contained in 1973, is 0626 because a minus branch is utilized. If the accumulator remains plus, the last location has been punched; therefore, 8002 is restored to its original value by adding back the contents of 1973. The instructions in 1961 and 1962 increase the D-address by 1 and store the modified instruction back into 1972.

The STOP instruction in 1964 is called for when the accumulator becomes plus. The instructions in 1965 through 1967 are called for if location XX48 or XX98 has been reached.

Figure 95. The instructions in 1956 and 1957 set up 8002 for testing. Note that 0626 (location 1973) is used for the high limit (last location) because punching takes place after testing. If the accumulator is non-zero, the last location has not been punched; therefore, the instruction in 1959 adds back the contents of 1973 in order to restore 8002 to its original value. The instructions in 8002 and 1960 load and store the contents of a table location. The instruction in 1961 increases the D-address by 1. Note that the modified instruction is not stored back into 1972 until punching has taken place; this allows the same STORE LOWER instruction to be used when the D-address has been increased by 2 as a result of the instructions in 1965 and 1966.

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV	CODE	DATA	INSTRUCTION	
1950	RAU	60	1972	1951	"D" address = Location to be punched
1951	SLT	35	0004	1952	Drop "Op Code" & 2 high order positions
					of "D" address
1952	SU	11	1970	1953	Test for location xx48
1953	BRNZU	44	1954	(1965)	Branch if not location xx48
1954	SU	11	1971	1955	Test for location xx98
1955	BRNZU	44	1956	(1965)	Branch if not location xx98
1956	RAL	65	1972	8001	"D" address = Location to be punched
8001	LD	69	xxxx	1957	Load values to be punched
1957	STD	24	0927	1958	Store values to be punched
1958	SL	16	1973	1959	Subtract high limit plus 1
1959	BRMIN	46	1960	(1964)	Branch if not high limit plus 1
1960	AL	15	1973	1961	Restore
1961	AL	15	1974	1962	Add 1 to "D" address
1962	STL	20	1972	1963	Store next location to be punched
1963	PCH	71	0927	1950	Punch
1964	STOP	01	0000	8000	Stop
1965	RAL	65	1972	1966	"D" Address = xx48, xx98
1966	AL	15	1975	1967	Add 2 - increase to xx00, xx50
1967	STL	20	1972	1950	Store next location to be punched
					1970 = 48 1957 0000
					1971 = 50 0000 0000
					1972 = 69 xxxx 1957
					1973 = 69 0626 1957
					1974 = 00 0001 0000
					1975 = 00 0002 0000

FIGURE 94. BRANCHING—MINUS

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV.	CODE	DATA	INSTRUCTION	
1950	RAU	60	1972	1951	"D" address = Location to be punched
1951	SLT	35	0004	1952	Drop "Op Code" & 2 high order positions
					of "D" address
1952	SU	11	1970	1953	Test for location xx48
1953	BRNZU	44	1954	(1965)	Branch if not location xx48
1954	SU	11	1971	1955	Test for location xx98
1955	BRNZU	44	1956	(1965)	Branch if not location xx98
1956	RAL	65	1972	1957	"D" address = location to be punched
1957	SL	16	1973	1958	Subtract high limit plus 1
1958	BRNZ	45	1959	(1964)	Branch if not high limit plus 1
1959	AL	15	1973	8002	Restore
8002	LD	69	xxxx	1960	Load values to be punched
1960	STD	24	0927	1961	Store values to be punched
1961	AL	15	1974	1962	Add 1 to "D" address
1962	PCH	71	0927	1963	Punch
1963	STL	20	1972	1950	Store next location to be punched
1964	STOP	01	0000	8000	Stop
1965	RAL	65	1972	1966	"D" address = xx48 or xx98
1966	AL	15	1975	1963	Add 2 to "D" address to increase
					to xxoo or xx50
					1970 = 48 1960 0000
					1971 = 50 0000 0000
					1972 = 69 xxxx 1960
					1973 = 69 0626 1960
					1974 = 00 0001 0000
					1975 = 00 0002 0000

FIGURE 95. BRANCHING—NON-ZERO

LOCATION OF INSTRUCTION	OPERATION		ADDRESS		REMARKS
	ABBRV.	CODE	DATA	INSTRUCTION	
1950	RAU	60	1972	1951	"D" address = Location to be punched
1951	SLT	35	0004	1952	Drop Op-Code & 2 high positions of
					"D" address
1952	SU	11	1970	1953	Test for location xx48
1953	BRNZU	44	1954	(1966)	Branch if not location xx48
1954	SU	11	1971	1955	Test for location xx98
1955	BRNZU	44	1956	(1966)	Branch if not location xx98
1956	RSL	66	1972	8001	"D" address = location to be punched
8001	LD	69	xxxx	1957	Part # & quantity to distributor
1957	STD	24	0927	1958	Store for punching
1958	PCH	71	0927	1959	Punch
1959	SL	16	1974	1960	Increase "D" address by 1
1960	AL	15	1973	1961	Add last location
1961	BRMIN	46	1962	1963	Branch if last location punched
1962	STOP	01	0000	8000	Stop
1963	SL	16	1973	1964	Restore to original value
1964	RAABL	67	8002	1965	Change sign to plus
1965	STL	20	1972	1950	Store
1966	RAL	65	1972	1967	"D" address = xx48 or xx98
1967	AL	15	1975	1964	Add 2 to "D" address to increase to
					xx00 or xx50
					1970: 48 1957 0000
					1971: 50 0000 0000
					1972: 69 xxxx 1957
					1973: 69 0625 1957
					1974: 00 0001 0000
					1975: 00 0002 0000

FIGURE 96. BRANCHING—MINUS

Figure 96. The instructions in 1956, 8001, 1957, and 1958 load, store, and punch a table location. Note that the contents of 1972 are placed in 8002 as a negative quantity. This allows a minus branch to be utilized without increasing the high limit (last location) by 1 as was necessary in Figure 94. The instruction in 1959 increases the D-address by 1, the instruction in 1960 adds the high limit (last location) to 8002. If the accumulator becomes plus, the last location has not been punched; if it remains minus, 8002 is restored to its original value by the instruction in 1963. The instruction in 1965 stores the modified instruction back into 1972.

Sub-Routines

A sub-routine is a compact, short program designed to accomplish one particular job such as square root, part number comparison, rate computation. Usually these routines will be part of a library of programs built up at each Type 650 installation. Thus, a programmer will often be able to assemble a number of sub-routines along with some original programming to plan his problem.

Careful consideration should be given to the following points by the programmer before employing a sub-routine:

1. Ease in programming
2. Speed of programming
3. Calculating time
4. Storage space consumed

If a library of sub-routines is to be really valuable, time and effort should be exerted to insure they are written for a minimum of running time and storage space. Those routines used often should be optimumly programmed for several storage areas for convenient use by the programmer.

There are many variations for entering and leaving sub-routines. The programmer must decide which best suits his needs. A few examples are offered here, assuming the sub-routine starts at location 0650 and exits from 0672. The simplest type of entry and exit is:

Example: xx xxxx 0650 (address of first instruction in sequence)

This assumes that the location of any necessary data is fixed as well as the location of the next instruction to be executed following the sub-routine.

In order to provide a flexible exit point, the following example requires the address of the first instruction after the sub-routine to be stored from the distributor by the sub-routine.

```
0200   LD    0203   0650  (0203 : xx xxxx 0201)
0650   STD   0672   0675
```

In the previous example no attempt to vary the location of the factors referred to by the sub-routine was made. A sub-routine might be written so that a factor (x) needed in the sub-routine could be in any location for the general routine. The following example shows the factor (x) being placed in the lower half of the accumulator immediately before entry into the sub-routine.

```
0200   RAL   Loc. x   0201
0201   LD     0204    0650  (0204 : xx xxxx 0202)
0650   STD    0622    0675
```

If two factors (x and y) are required and they have the same sign, the following can be done:

```
0200   RAU   Loc. x   0201
0201   AL    Loc. y   0202
0202   LD    0205     0650   (0205 : xx xxxx 0203)
0650   STD   0672     0675
```

In order to get a variety of sign combinations, one factor could be placed in the distributor, and the next instruction (contents of 0205), in the upper accumulator, etc.

If multiple factors are required by the sub-routine, these may be moved to a block of locations always used by the sub-routine.

There are two basic methods for handling results computed by sub-routines. The first method leaves the result in the accumulator and/or the distributor for immediate access upon return to the main program. The second method stores the results in temporary storage locations fixed by the sub-routine.

The coding of sub-routines within storage blocks makes for easier and neater programming. It is better to sacrifice optimum programming somewhat in order to achieve a program in one consecutive block rather than have blank spaces within a sub-routine.

Initialization

One of the first things a program should do is insure that the following preparation or initialization steps are performed:

1. All locations used for accumulation are set to zero.
2. All variable addresses are set to the proper initial value.
3. All switches within the program are set to the initial condition.

It should be remembered that any error routine written for a problem must make provision for any necessary initialization. In the case of errors in procedure (cards out of sequence, punch errors in input, etc.) the programmer will need to make provisions to perform initialization either through programming or reloading of factors necessary to restart with the proper conditions set.

Error Sense — Error Correction

When random errors occur, it is more efficient for the machine to repeat the calculation a specified number of times rather than require manual operations by the operator. In addition to the speed superiority of programmed corrections over manual corrections, experience indicates a far greater reliability on automatic procedures.

Error correction may be accomplished, in some cases, by merely returning to the first step (other than a read) in the program. In other cases, it will be desirable to block the problem into breaking points. A compact error routine can then be incorporated in the program to restart the program automatically at the last correct breaking point. The correct restart point is determined by altering the error routine as each block is successfully completed.

When an error correction routine is being used the error switch is set to SENSE. If an error is detected, the machine automatically refers to the storage entry switches (8000). The instruction in 8000 should direct the program to the error correction routine.

To simplify the problems that might be encountered in writing an error, correction routine for problems involving intermittent accumulation of factors, the following technique is suggested as part of the main program.

When the new total has been stored and while the total is still in the accumulator, the factor just written should be loaded back into the distributor from the drum location. If the factor has been improperly written, the factor loaded from the drum will not pass the validity check in the distributor. When this method is used, the error switch must be set to STOP. Thus, the operator can obtain the correct factor from the accumulator for storage on the drum.

Program Testing

In order to verify that a program is correctly written to accomplish all the specifications set down in the problem definition, a test case is run. This test should include precalculated *partial* and *final* results for all logical branches in the program. If this method is used, the time required to test and correct a program will be greatly reduced.

Assuming that a program contains some errors, corrections can be made by using any of the following three methods:

1. Console operation
2. Memory punch-out
3. Tracing

Console operation is unquestionably the simplest and fastest method for locating errors in a program. When the address-stop feature is used in conjunction with the precalculated partial results, it is a simple matter to locate the point at which an error occurs. The display switch permits examination of the accumulator, distributor, and program register without disturbing their contents. The contents of any drum location may be displayed. The storage entry switches provide a means for entering factors into storage.

Memory punch-out can be used to punch the contents of drum locations along with the location address. It is possible to specify limits for the area to be punched. In order to speed the operation, several locations are punched in each card. Memory punch-out requires that a punch program be stored in the machine.

Tracing makes use of a program that monitors the program being tested. As each step of the program is executed, a card is punched that contains the following information:

1. Location of the instruction executed
2. The instruction
3. Contents of the accumulator and distributor
4. Condition of the overflow indicator (if desired)

The output cards can then be listed to examine the programming. Tracing is slow and expensive and is not recommended except as a last resort when the Type 650 is being used.

The same test case used to verify the program provides an excellent test deck for future use. The test deck can be used to verify machine setup when a run is being started; it also can be a great aid in machine trouble analysis.

Program Planning Aids

A booklet, *Program Planning Aids* (Form 22-6220), contains several hints and suggestions aimed at helping the programmer, as well as information on block diagramming, use of the planning chart, single instruction loading card, and preliminary program testing and analysis.

Read Code Selectors and Punch Code Selectors

Read code selectors and punch code selectors are optional features on the Type 650. They can be obtained in groups of five read code and five punch code selectors (maximum two groups). Read or punch code selectors are not furnished separately.

Read Code Selectors (X-Z; 23-32). Each read code selector has three associated control panel hubs. These hubs are:

In 1. This hub is normally wired from first reading and accepts digit impulses 12 through 7.

In 2. This hub is normally wired from READ CARD A, READ CARD B, or READ CARD C, and accepts digit impulses 12 through 7.

Out. This is an exit hub. If both the IN 1 and the IN 2 hubs receive an impulse, the OUT hub will emit an 8 impulse. If neither the IN 1 hub nor the IN 2 hub receives an impulse, the OUT hub will emit a 9 impulse. If only one of the IN hubs receives an impulse, *no* impulse will be available at the OUT hub.

Associated with the read code selectors is a switch called READ SECOND ONLY (RSO). When this switch is wired, both IN hubs may be wired from the same reading station (Read Card A, B or C, or First Reading).

Because the OUT hub of a read code selector emits an 8 or 9 impulse, the selector can be readily used to enter 8's and 9's into storage entry for X and No-X conditions.

Figures 97 and 98 show control panel wiring for two uses of read code selectors.

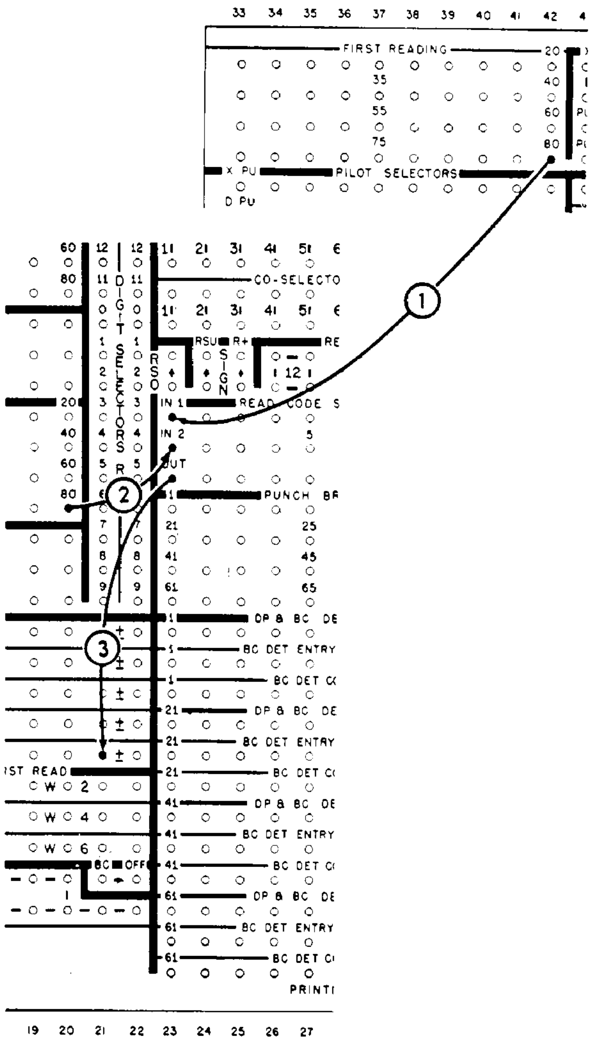


FIGURE 97. READ CODE SELECTORS

Wiring (Figure 97)

1. Any digit impulse (12 through 7) read at first reading column 80 enters the IN 1 hub of the READ CODE SELECTOR.
2. Any digit impulse (12 through 7) read at second reading column 80 enters the IN 2 hub of the READ CODE SELECTOR.
3. Either an 8 impulse, a 9 impulse, or a blank enters the units position of word 10 depending on how the IN hubs have been conditioned.

Wiring (Figure 98)

1. Both IN hubs of the READ CODE SELECTORS are activated from the same set of reading brushes.
2. Any digit impulse (12 through 7) read at second reading column 80 enters both the IN 1 and IN 2 hubs of the READ CODE SELECTORS.
3. Either an 8 impulse, or a 9 impulse, enters the units position of word 10 depending on how the IN hubs have been conditioned.

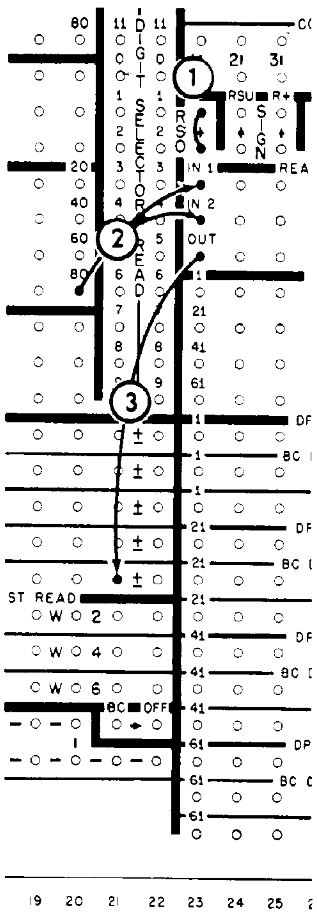


FIGURE 98. READ CODE SELECTORS

Punch Code Selectors (AN-AR; 55-65). The punch code selectors available for the Type 650 are single-position selectors with one pickup hub. These selectors may be controlled by any impulse timed to the punch feed. The selector transfers immediately when impulsed. Figures 99 and 100 show control panel wiring for two uses of the punch code selectors.

Wiring (Figure 99)

1. The punch code selectors are activated by a control information impulse resulting from an eight in the units position of word 10.

2, 3. An X is punched in column 80 of the output card when the punch code selector is transferred.

Wiring (Figure 100)

1. Punch code selector 10 transfers when an impulse is emitted from control information position 10.

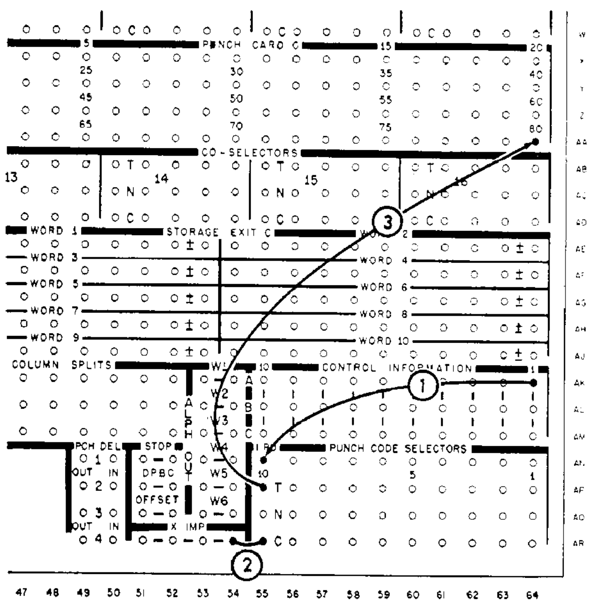


FIGURE 99. PUNCH CODE SELECTORS

2. A digit 1 enters the transferred side of punch code selector 10.

3. A digit 2 enters the normal side of punch code selector 10.

4. A digit 1 or a digit 2 is punched in column 80 of an output card depending upon the transferred or normal condition of punch code selector 10.

Half-Time Emitter

Half-time emitters are available for both the read and punch feeds. Either a half-time emitter or

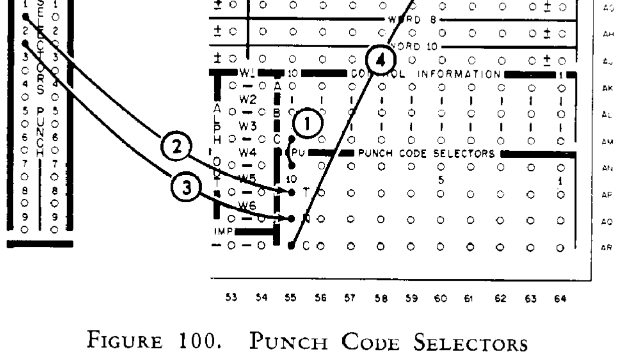


FIGURE 100. PUNCH CODE SELECTORS

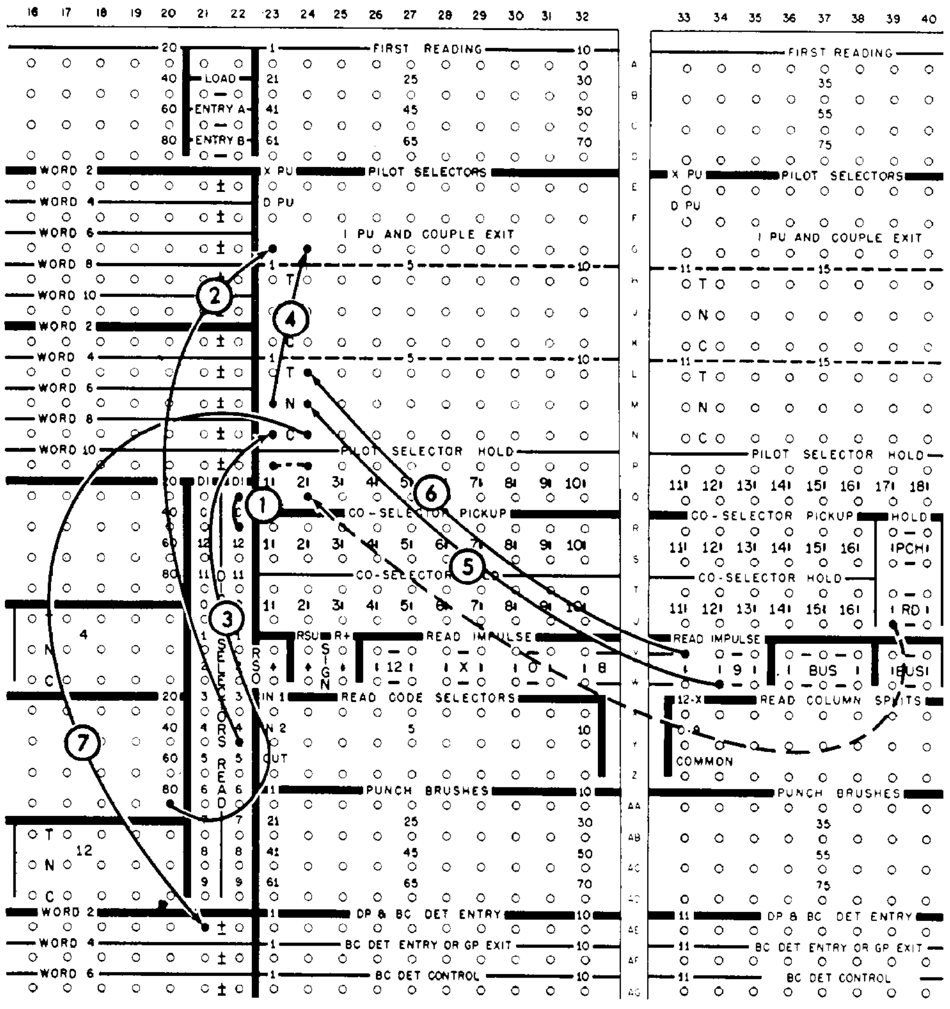


FIGURE 101. HALF-TIME EMITTING

an additional digit selector, not both, can be installed for each feed. The control panel location for half-time emitters is the same as the additional digit-selector location. It is possible to install a half-time emitter on one feed and an additional digit selector on the other.

The half-time emitter is similar to the digit emitter, except that the impulses occur at half-time after the digits 12-9. These impulses are normally used to cause selectors to transfer between two specific digits.

The control panel locations for the half-time emitters are:

Read (Q-AD, 22)
Punch (Q-AD, 44)

The half-time impulses are available at the DI hubs (Q, 22) or (Q, 44).

An example of the use of the half-time emitter is shown in Figure 101.

Wiring (Figure 101)

1. Half-time READ DIGIT impulses supplied to the half-time emitter.
2. A half-after-digit-four impulse picks up pilot selector 1 every read feed cycle.
3. The digit impulses read in column 80 enter the common of pilot selector 1.
4. Any digit impulses (12 through 4) read in column 80 pick pilot selector 2.
5. A digit eight enters word two if any digit 12 through 4 is read in column 80.
6. A digit nine enters word two if any digit 5 through 9 or blanks are read in column 80.
7. The common of pilot selector 2 connected to the units position of word 2.

OFFSET STACKING DEVICE

An offset stacking device is available for installation on the punch stacker. This device is used to locate specified cards by offsetting them in the punch stacker 3/8" toward the front of the machine. The operation of this device is controlled by control panel wiring to the offset hubs (AQ, 51-52).

Examples of control panel wiring to control the offset stacking device are shown in Figures 102 and 103.

Wiring (Figure 102)

1. The card in which a double-punched or blank-column error was detected is offset in the punch stacker.
2. The digit impulses read in column 80 by the punch brush enter the common of the punch column splits.
3. The 11 or 12 impulses read in column 80 activate the punch delay device.
4. The offset stacking device is activated one cycle after an 11 or 12 card identification punch has been read in column 80 by the punch brushes. The identified card will be offset in the punch stacker.

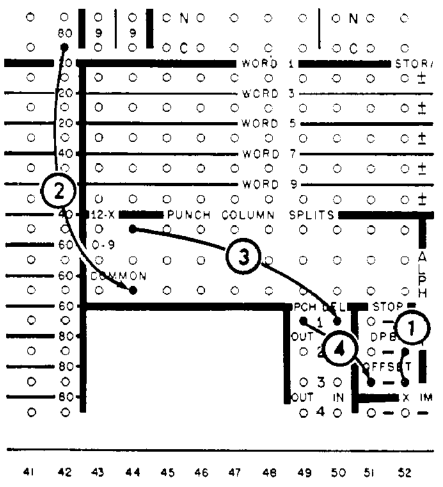


Figure 102. Offset Stacking

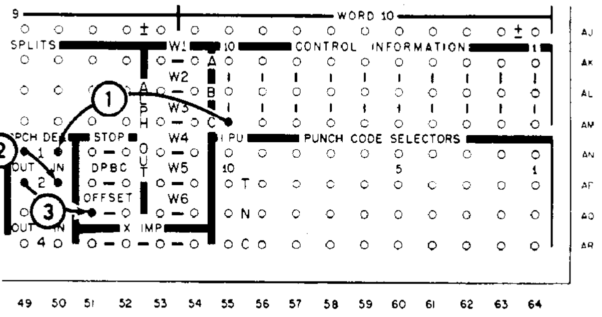


Figure 103. Offset Stacking

Wiring (Figure 103)

1. The punch delay device is activated by the control information impulse when an eight is in the tenth position of word 10.
2. This wire causes a second cycle delay.
3. The offset stacking device is activated two cycles after the control information impulse identifies the card to be offset. The identified card will be offset in the punch stacker.

ALPHABETIC DEVICE

The alphabetic device provides a maximum input and output of 30 alphabetic positions distributed in the first 6 storage entry or storage exit words. The device will handle all 26 alphabetic characters plus the 10 numerical digits. Special characters are excluded.

Transfer of pure alphabetic or alphabetic-numerical information from read input to punch output is possible. The alphabetic input may be used logically in the program for table look-up arguments and other operations as required. Those entry words, designated by control panel wiring as alphabetic, have their entry information translated in the entry scanning circuits into a 10-digit numerical word. From that point on, the translated word is handled internally as any other numerical word.

The coded alphabetic-numerical information assumes the following sequence relationship internally:

Character	Card Code		Type 650 Code
Blank		=	00
A	12-1	=	61
B	12-2	=	62
C	12-3	=	63
D	12-4	=	64
E	12-5	=	65
F	12-6	=	66
G	12-7	=	67
H	12-8	=	68
I	12-9	=	69
J	11-1	=	71
K	11-2	=	72
L	11-3	=	73
M	11-4	=	74
N	11-5	=	75
O	11-6	=	76
P	11-7	=	77
Q	11-8	=	78
R	11-9	=	79
S	0-2	=	82
T	0-3	=	83
U	0-4	=	84
V	0-5	=	85
W	0-6	=	86
X	0-7	=	87
Y	0-8	=	88
Z	0-9	=	89
Zero	0	=	90
1	1	=	91
2	2	=	92
3	3	=	93
4	4	=	94
5	5	=	95
6	6	=	96
7	7	=	97
8	8	=	98
9	9	=	99

When a word is alphabetically coded, the number of digits involved is doubled. For example, a part number 5A72R is recoded as 9561979279. If more than five characters are involved, and the alphabetic portion is interspersed at random, at least two alphabetic words must be used. Alphabetic words are automatically read into the machine as positive numbers.

The device is flexible from a capacity standpoint. Any number of input words from 0 to 6 may be designated as alphabetic by selective control panel wiring. Similarly, any number of output words from 0 to 6 may be made alphabetic. The output words may be controlled by either internal programming or control panel wiring. The same words need not be designated as alphabetic on both input and output. The alphabetic designation may vary from output card to output card as required by control panel selection. Because the zone portions of the alphabetic characters are read at first reading, it is impossible to use field selection on alphabetic fields in the normal manner. Alphabetic field selection may be accomplished as shown on pages 108 and 109.

Alphabetic Input Hubs

Alphabetic in entry hubs *(AL-AR, 11-12)* receive impulses to set up the corresponding storage entry word to receive alphabetic information. These hubs may be constantly impulsed from the CONSTANT ALPHABETIC IMPULSE (CAI) exit hub or selectively from the COUPLE EXIT of a pilot selector.

CAI exit hub *(AK, 12)* provides a constant alphabetic impulse except during load cards. This may be used to impulse the desired ALPHABETIC IN hubs.

Alphabetic first read entries *(AK-AM, 13-22)* are normally wired from first reading. The alphabetic columns to be read from the card are entered directly as required.

The normal STORAGE ENTRY hubs of the alphabetic words must be impulsed from second reading. The card columns to be entered are wired to the five low-order digits of the storage entry. Each alphabetic or numerical character is internally translated to a standard two-digit number for use in the computer.

NOTE: When the word in question is alphabetic, the word size entry is always wired for word size 10.

Alphabetic Output Hubs

Alphabetic out entry hubs *(AK-AQ, 53-54)* receive impulses to convert the corresponding storage exit words to alphabetic output. These entry hubs normally receive impulses from CONTROL INFORMATION when selective control is required. If constant alphabetic output is required, an impulse is entered from the PSU exit, V-41.

The normal STORAGE EXIT hubs are wired to the selected punching positions, using the five low-order exits of the alphabetic word.

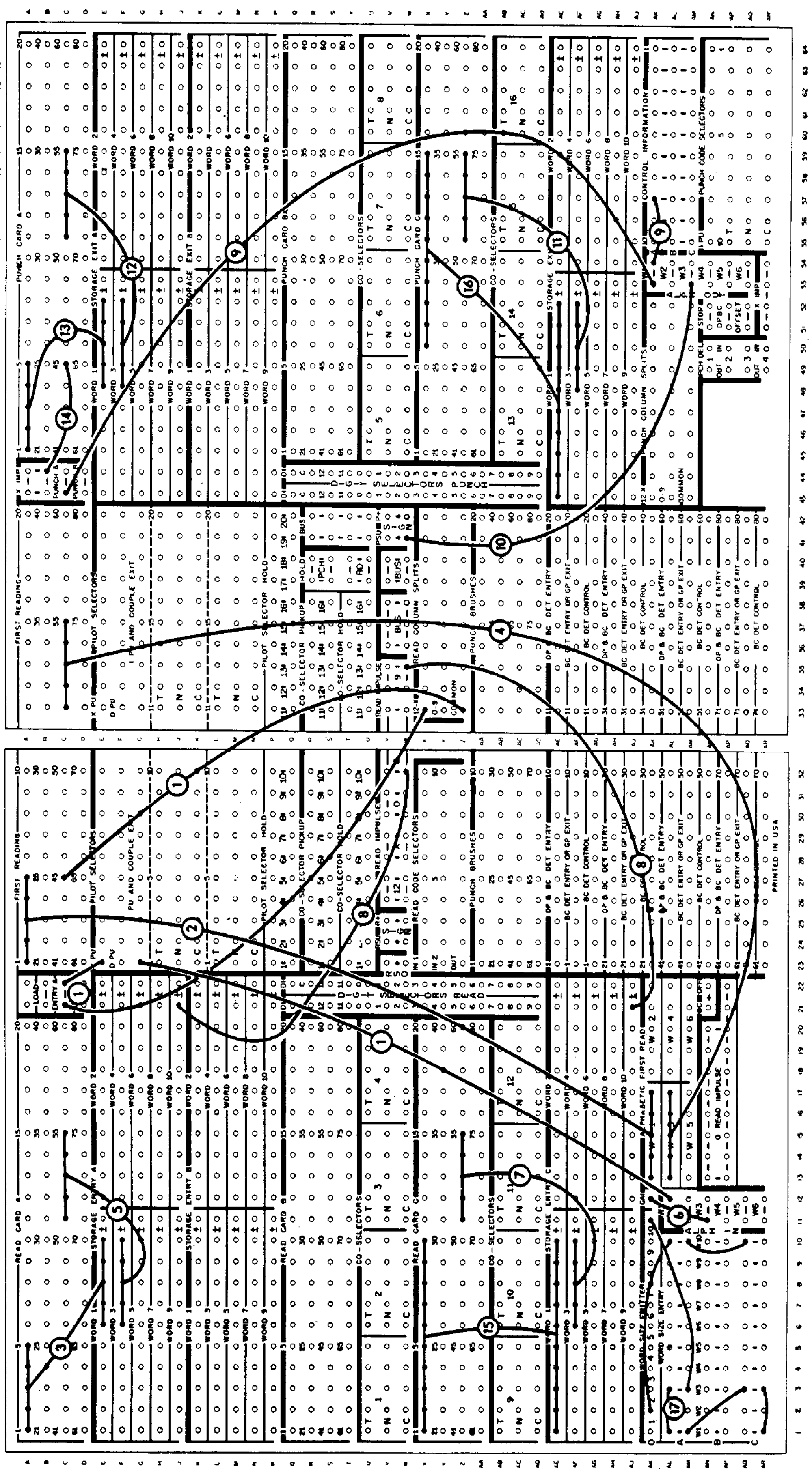

READ CARD A
FIRST READING
PILOT SELECTORS
PU AND COUPLE EXIT
STORAGE ENTRY A
STORAGE ENTRY B
READ CARD B
CO-SELECTORS
READ CARD C
CO-SELECTOR PICKUP
PILOT SELECTOR HOLD
CO-SELECTOR HOLD
READ IMPULSE
READ CODE SELECTORS
PUNCH BRUSHES
STORAGE ENTRY C
DP & BC DET ENTRY
BC DET ENTRY OR GP EXIT
BC DET CONTROL
WORD SIZE EMITTER
WORD SIZE ENTRY
ALPHABETIC FIRST READ
READ IMPULSE
PUNCH CARD A
STORAGE EXIT A
STORAGE EXIT B
PUNCH CARD B
PUNCH CARD C
STORAGE EXIT C
PUNCH COLUMN SPLITS
CONTROL INFORMATION
PUNCH CODE SELECTORS
PRINTED IN USA

FIGURE 104

SAMPLE CONTROL PANEL WIRING

Problem

Card columns 51-55 are alphabetic and are to be reproduced in the same columns of all output cards from all input cards.

Card columns 1-5 contain alphabetic information only in X-45 cards, and this is to be carried through into output cards identified by an X-45.

Card columns 1-10 contain numerical information in NX-45 cards. This is to be entered into the computer, and the results are to be punched in columns 6-15 of NX-45 output cards.

Wiring (Figure 104)

1. X only in column 45. The X energizes ENTRY A and continues on to pick pilot selector 1. The IPU (couple exit) is used to impulse ALPHABETIC IN WORD 1. Word 1 is alphabetic only on ENTRY A cards.

2. First reading, columns 1-5, contains alphabetic information and are wired to ALPHABETIC FIRST WORD 1. These wires are used to condition WORD 1 to receive alphabetic information.

3. READ CARD A, columns 1-5, contains the alphabetic information and are wired to the low-order positions of STORAGE ENTRY A, WORD 1.

4. First reading, columns 51-55, contains alphabetic information and are wired to ALPHABETIC FIRST READ, WORD 3. These wires condition word 3 to receive alphabetic information.

5. READ CARD A, columns 51-55, contains alphabetic information and are wired to the low-order positions of STORAGE ENTRY A, WORD 3.

6. CONSTANT ALPHABETIC IMPULSE (CAI) is wired to ALPHABETIC IN WORD 3 to cause word 3 to be alphabetic for all cards.

7. READ CARD C, columns 51-55, is wired to STORAGE ENTRY C, WORD 3, low-order positions. This provides alphabetic input for word 3 for entry C cards.

8. An *8* is emitted into position of word 10, entry A for identification. Likewise, a *9* is wired into the same position of entry C for the normal indication.

9. A CONTROL INFORMATION impulse is emitted from digit 8 to identify PUNCH A CARDS. This is also used to cause STORAGE EXIT WORD 1 to become alphabetic for PUNCH A CARDS.

10. The PSU EXIT hub emits an early timed impulse to ALPHABETIC OUT WORD 3 to cause word 3 exit to be alphabetic for all cards.

11. STORAGE EXIT C, WORD 3, five low-order positions are wired to PUNCH CARD C, columns 51-55 to PUNCH CARD C alphabetic output.

12. STORAGE EXIT A, WORD 3, five low-order positions are wired to PUNCH CARD A, columns 51-55 to PUNCH CARD A alphabetic output.

13. STORAGE EXIT A, WORD 1, five low-order positions are wired to PUNCH CARD A, columns 1-5 to PUNCH CARD A alphabetic output.

14. An X-impulse is emitted into column 45, PUNCH CARD A for identification.

15. Ten digits of numerical data are read into word 1 on entry level C. Level C is identified by NX-45 cards.

16. Ten digits of numerical data are punched out of word 1 on exit level C. Level C output is also identified by NX-45 cards.

17. WORD SIZE ENTRY wiring for those words described here is shown. No other word size entry wiring is illustrated. Note that all these words are wired to emitter 10. The alphabetic words are five characters each, but the recoding expands them to ten digits.

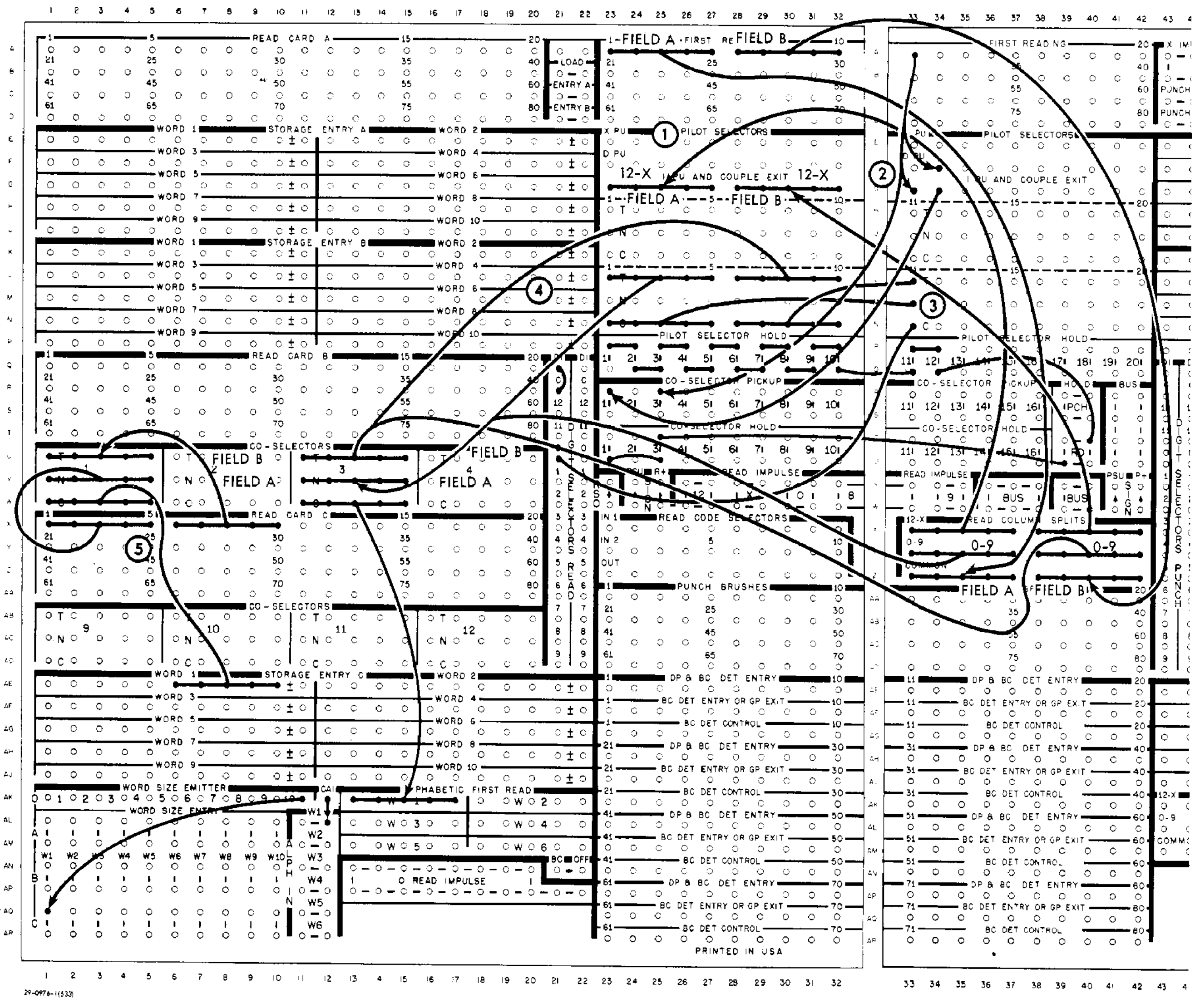


FIGURE 105. ALPHABETIC FIELD SELECTION — METHOD 2D ONLY

Alphabetic Selection (Without Special Characters)

Several types of alphabetic selection can be performed. Control panel diagrams are not included for items 1, 2a, 2b, and 2c.

1. Alphabetic field for X-cards; numeric field for NX-cards. A pilot selector is picked by the X at first reading. The pilot selector couple exit is wired to the ALPHA IN hub for the correct word. The alphabetic first read wiring is ignored on the NX-cards. Because of this, the numeric field may be in different columns than the alphabetic and only the storage entry wiring from second reading needs to be selected.

2. Alphabetic Field Selection.

METHOD A. Alphabetic fields can be readily selected based upon an identifying punch in the preceding card. Both the alphabetic first read and storage entry wiring must be selected.

METHOD B. Results equivalent to alphabetic field selection can be obtained by reading both fields all the time, for example, field A on word 1; field B on word 2, and then selecting the correct field by programming.

METHOD C. If both fields are entirely alphabetic (every column has an alphabetic character), the alphabetic first read wiring does not have to be selected, but can be wired directly from either of the two fields. The storage entry wiring from second reading (read card C) has to be selected in the normal manner.

NOTE: This is possible since the alphabetic first read hubs identify a card column as having a zone (12, 11, 0) and/or a numeric (1-9). It does not identify the particular zone or the particular numeric value. Blanks are identified by the absence of the zone and numeric.

METHOD D. Completely flexible alphabetic field selection is possible within the selector capacity of the machine. The card for which field B is to be selected (field A is the normal field) must be identified by either a 12 or an 11 punch. The method used depends upon the fact noted above and operates as follows:

1. Pilot selectors are transferred whenever an X or 12 occurs on either of the two fields.

2. A pilot selector (No. 11) is transferred by the control X or 12.

3. A digit emitter zero is selected to test either the field A pilot selectors or the field B pilot selectors.

4. The output from the selectors (a zero) substitutes for the corresponding 12 or X-impulse and is wired into ALPHABETIC FIRST READ along with the correct numeric (1-9) for the selected field.

5. The storage entry wiring is selected in the normal manner.

SPECIAL CHARACTER DEVICE

A SPECIAL CHARACTER device is available for the Type 650. This device is available in two different groups and is capable of handling special characters. The number of special characters that this device will handle will be dependent on the group specified.

The available groups and their capacities are:

1. Group I will handle eleven and twelve punches only.

2. Group II will handle all Type 407 special characters including eleven and twelve punches only.

NOTE: The foregoing discussion of alphabetic selection does not apply when the special character device is in use.

Each character that is to be handled by the special character device is automatically recoded into a two-digit number. The sequence of the special character code and the standard alphabetic code is the same as that of the Type 89 Alphabetic Collator; for example, blank, special characters, alphabetic, and numeric from the lowest to the highest order. The numerical representations of the complete code are:

Character	Card Code		Type 650 Code	
Blank		=	00	Standard Alphabet Special Characters Group I, II
.	12-3-8	=	18	Special Characters Group II
⌑	12-4-8	=	19	
&	12	=	20	Special Character Group I, II
$	11-3-8	=	28	Special Characters Group II
*	11-4-8	=	29	
-	11	=	30	Special Character Group I, II
/	0-1	=	31	Special Characters Group II
,	0-3-8	=	38	
%	0-4-8	=	39	
#	3-8	=	48	
@	4-8	=	49	
A	12-1	=	61	Standard Alphabet
B	12-2	=	62	
C	12-3	=	63	
D	12-4	=	64	
E	12-5	=	65	
F	12-6	=	66	
G	12-7	=	67	
H	12-8	=	68	
I	12-9	=	69	
J	11-1	=	71	
K	11-2	=	72	
L	11-3	=	73	
M	11-4	=	74	
N	11-5	=	75	
O	11-6	=	76	
P	11-7	=	77	
Q	11-8	=	78	
R	11-9	=	79	
S	0-2	=	82	
T	0-3	=	83	
U	0-4	=	84	
V	0-5	=	85	
W	0-6	=	86	
X	0-7	=	87	
Y	0-8	=	88	
Z	0-9	=	89	
Zero	0	=	90	
1	1	=	91	
2	2	=	92	
3	3	=	93	
4	4	=	94	
5	5	=	95	
6	6	=	96	
7	7	=	97	
8	8	=	98	
9	9	=	99	

INDEX

INDEX (Continued)

OPERATION CODES

IBM

FORM 22-6060-1 (6-56:15M-VO)