*Equipment Series*

Bendix

# G-20

# *General Reference Manual*

# *G-20*

# *General Reference Manual*

# REVISION NOTICE

This manual is a major revision of the G–20 GENERAL REFERENCE MANUAL dated 25 November 1960. Two new accessories are described in this edition, the PT–10 Paper Tape Station and the DC–11 Data Communicator. The latter device provides for **two** additional input/output channels that communicate with Core Memory.

The Magnetic Tape, MT–10, has been modified to increase its effective speed, to provide a continuous read mode, and to remove certain programming restrictions.

The Central Processor mnemonic operation codes adopted by the G–20 users are contained in this edition. Cross reference to the older mnemonic codes, if required, can be made through the octal hardware codes.

This revised manual contains an entirely new and improved treatment of the subject of timing.

Discussion of programming has been deleted; separate publications are available for the various G–20 programming systems.

In addition to the above specific changes, the text has been updated and rewritten.

**1 May, 1962**

CAP

# PREFACE

This manual presents a concise and exhaustive treatment of the G–20 System, and is oriented to the advanced user or system programmer.

For normal application programming the user should refer to the *programming* manual relating to the particular service routine or system program he is using. This source will give all the information ordinarily required for such purposes.

For maintenance instructions the *maintenance* manuals for the G–20 System should be consulted.

# TABLE OF CONTENTS

PAGE

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF EXAMPLES

# CHAPTER 1
# INTRODUCTION

The G–20 Data Processing System [Figure 1.0–1] contains a high-speed Central Processor and a complete set of input/output accessories matched to the Central Processor. Depending upon the number and variety of accessories, the system is well suited for either scientific computing or business data processing.

A G–20 System may have from 4,096 to 32,768 words of core memory, with a 6–microsecond read/write cycle. Each word contains 32 bits plus a parity bit. The Arithmetic Unit carries out all operations in parallel, in 50–bit double precision floating point octal arithmetic. There are 42 bits in the mantissa, 6 bits in the octal exponent, and 2 sign bits. Quantities can be stored automatically in 29–bit single precision form [one word] or 50–bit double precision form [two words], corresponding to 7 octals or 14 octals in the mantissa. Storage of numbers in the "pickapoint" mode, and integer storage, are provided automatically. In the pickapoint mode the octal point is placed at any desired fixed location in the word. Addressing is flexible in that unlimited cascading of addressing is permitted. There are 104 opcodes including 8 address preparation codes, 56 arithmetic and logic operations, 32 block arithmetic and logic operations, two single character input/output operations, and 6 block input/output operations. In block operations the block can be of any size.

Sixty-three locations in core memory are used as Index registers. There are also several addressable hardware registers used for various purposes in the machine. A real time clock is provided, operating in increments of one second. Representative single address execution times for the Central Processor including command access and operand access are as follows:

**TABLE 1.0–1 EXECUTION TIMES INCLUDING ACCESS, MICROSECONDS**

| | |
|---|---|
| Add/Subtract | 12 |
| Multiply | 42 |
| Divide | 81 |
| Shift (1-13 octals) | 26 |
| Store | 14 |
| Indexed command | 6 additional |
| Decrement and test index | 19 |
| Transfer control | 6 |
| Block add | 8 per word |
| Table lookup | 8 per word |

For communication with external accessories the Central Processor packs and unpacks computer words automatically into 6–bit or 8–bit characters and transmits or receives them at rates up to 200,000 characters per second.

A high-speed Magnetic Tape system reads or writes at 120,000 characters per second. With appropriate service routines the Magnetic Tape system operates as a completely addressable storage medium. If a block length of 512 words is selected, a 3,600 foot reel of one-inch tape stores 8 million words. Unlimited rewrite over a block is permitted. A Paper Tape Unit punches out 110 characters per second and reads forward and backward at 500 characters per second. A Card-Printer Coupler connects the G–20 to conventional 80–column card equipment, or to unbuffered Line Printers, providing up to 1,500 lines per minute. A buffered Line Printer prints up to 1,000 lines per minute. The G–20 console contains an 88–character typewriter whose full alphabet can be handled by the computer. The console also contains a set of indicator lights which can be used to inform the operator of the status of the program.

A Control Buffer available for the G–20 System stores 4,096 characters in its core memory, and can handle off-line operations such as tape to printer without intervention from the G–20. A Data Communicator provides two scatter-read, gather-write, input/output channels that are independent of the Arithmetic Unit.

All of the units of the G–20 System communicate in a common language of 10-bit characters, each consisting of eight numerical bits, a data flag, and a parity bit. Thus, the G–20 System can handle alphabets having up to 256 characters. Bendix Extended Hollerith is a 256–character code containing as a subset all of the punch configurations that are permitted in ordinary Hollerith Code. There are four circuits on which the G–20 can receive interrupts, and two circuits on which it can transmit interrupts. Connection of various units to these circuits is made at installation according to the desires of the user. The interrupt feature and the Data Communicator make possible maximum utilization of the arithmetic capacity of the G–20 Central Processor.

# CHAPTER 2
# CENTRAL PROCESSOR

The G–20 is a single-address computer with magnetic core storage and parallel arithmetic. Timing is synchronous with 1–microsecond clock cycle. A memory word contains 32 bits plus a parity bit; maximum core memory consists of 32,768 words. Operations are performed in 50-bit floating point octal arithmetic with a 42-bit mantissa, a 6-bit exponent and 2 sign bits.

Two forms of floating point storage are available: 7 octal digit mantissa [one word], and the full 14-digit mantissa [two words] of the Arithmetic Unit. A fixed point mode of operation is available using storage of 9 octal digits. Six-bit or 8-bit alphanumeric characters can also be handled.

Sixty-three core locations are available as Index registers and the command structure permits addressing of almost unlimited flexibility. Thirty-two of the commands operate automatically on a block of any length. Input/output can be time-shared with computation.

## SECTION 2.1 HARDWARE ORGANIZATION

The hardware organization is discussed below under three general headings, viz., physical characteristics, information flow, and arithmetic.

**2.1–1 Physical Characteristics** The Central Processor, Figure 2.1–1, contains about 5,150 transistors and 30,700 diodes, in addition to one or two modules of core memory containing 130,000 cores each. [Additional core memory is in separate cabinets.] Components are fixed to printed circuit cards mounted on strips known as master boards. The strips are arranged on hinged panels which swing out to make all parts of the unit easily accessible. It is 64 inches wide, 28 inches deep, and 64 inches high, and weighs 2,500 pounds, approximately. According to whether the Central Processor contains 4,096 or 8,192 words of core, it requires 1.9 KVA or 2.1 KVA of 115 or 230-volt, single-phase, 60-cycle power.

The G–20 Central Processor may contain one or two memory modules of 4,096 words each. For systems requiring larger memories, up to six memory modules can be added external to the Central Processor. Each word in core storage is composed of 32 magnetic cores [plus one extra core which is used for parity checking and is not available to the programmer]. A group of 4,096 magnetic cores assembled in a 64 by 64 array is known as a memory plane, and 33 such planes with associated circuitry are combined in a memory module. A word takes one of its binary digits from each of the planes. The core memory is entered by means of a destructive "read" requiring 3 clock cycles followed by a "restore" requiring 3 clock cycles, a total of 6 clock cycles. A word read from memory is available for use at the end of the third clock cycle. Computation then begins, provided that the next operation required is not another memory access. In the latter case computation does not begin until after the "restore" is completed.

**2.1–2 Information Flow** A simplified block diagram of the G–20 Central Processor is shown in Figure 2.1–2. The major varieties of information flow can be described in terms of this diagram.

In the reading of a command from memory the number in the Next Command word register, NC, is transferred to the Memory Address register, MA. At the same time the NC register is incremented in preparation for the next order. The quantity in memory in the location given by the number in MA is read into the Memory Buffer, MB. Since this act destroys the word in core, the quantity in MB is immediately copied back into memory as well as transferred to Command Decoding, CD. Using the terminology that (XY) means

FIGURE 2.1-1 G-20 CENTRAL PROCESSOR

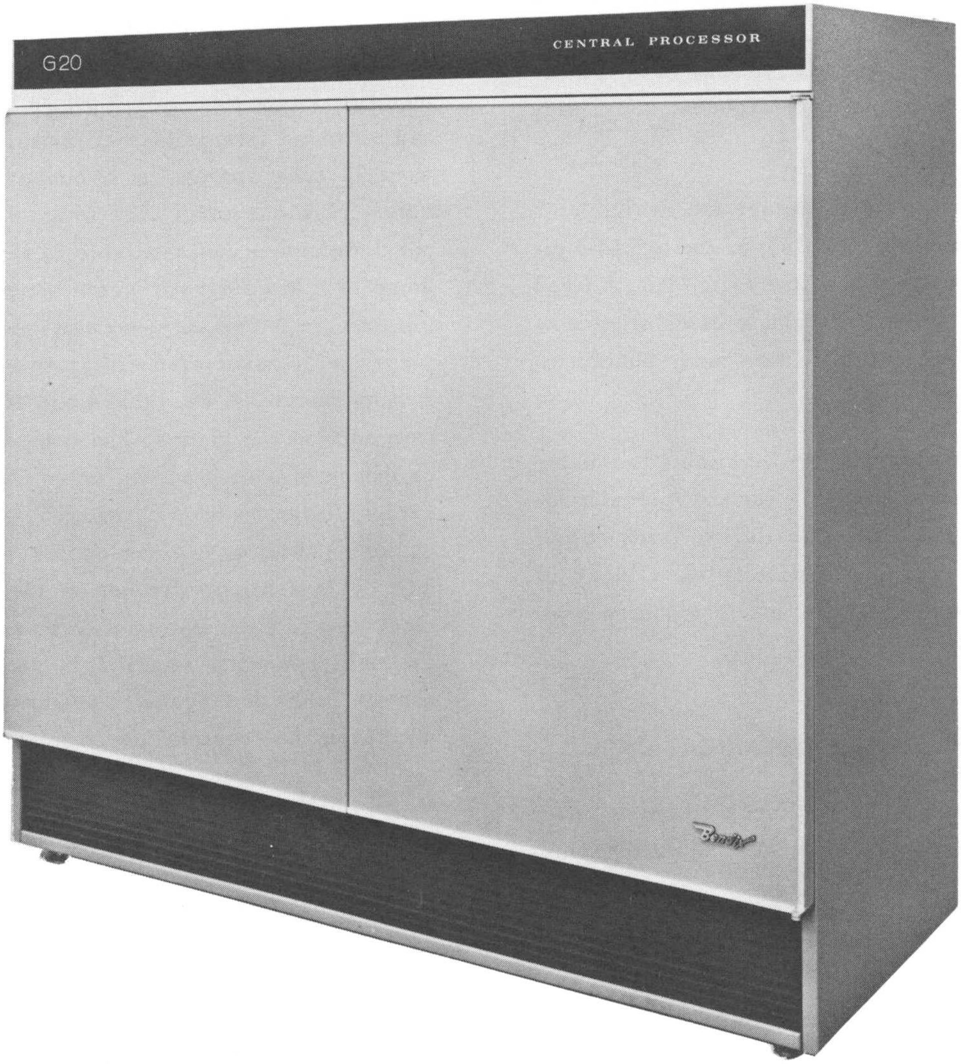
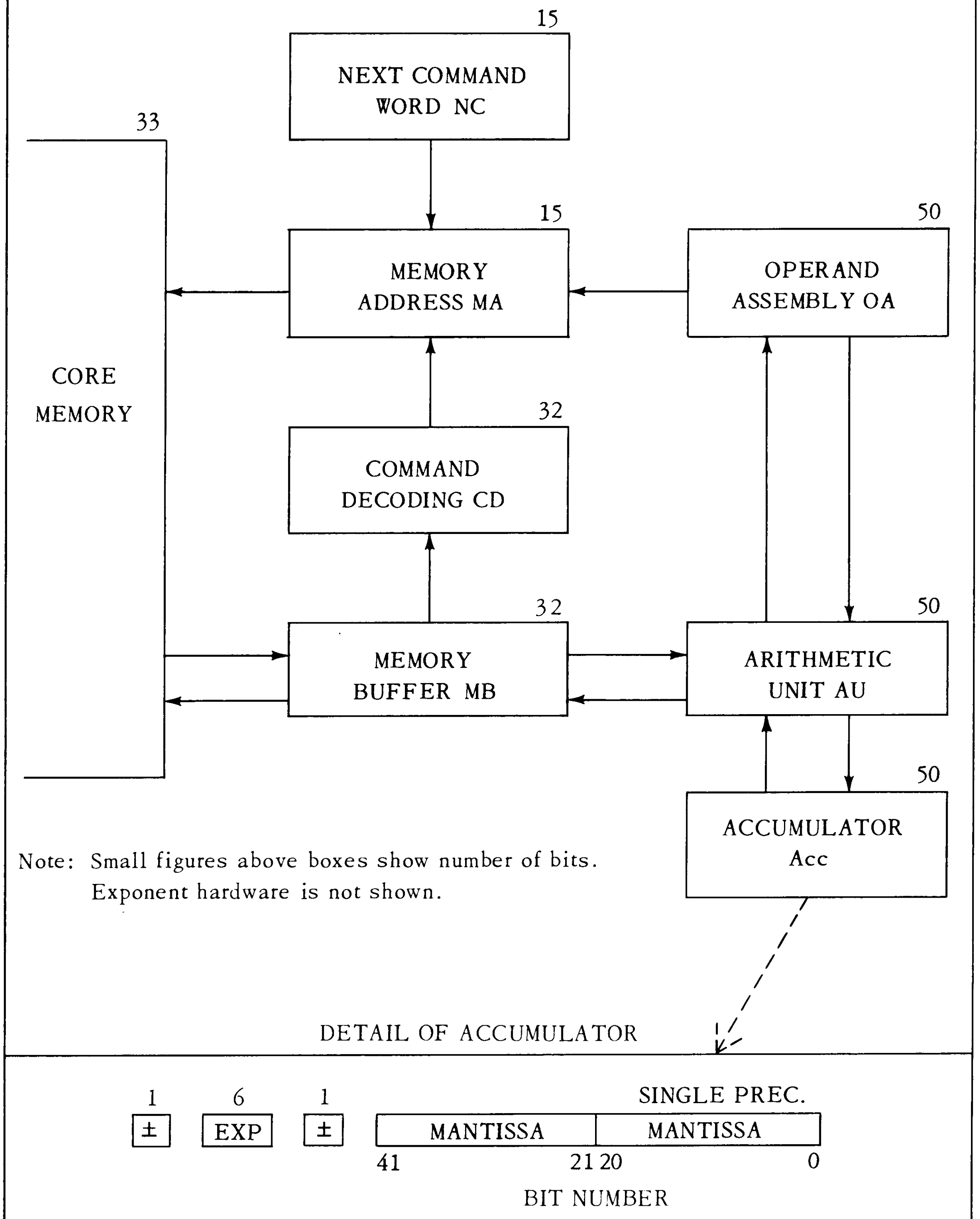


FIGURE 2.1-2 SIMPLIFIED BLOCK DIAGRAM G-20 CENTRAL PROCESSOR



Note: Small figures above boxes show number of bits.
Exponent hardware is not shown.

"contents of register XY," (57) means "contents of memory location 57," and the arrow, →, indicates "replaces," these operations can be given by the following expressions:

**FIGURE 2.1-3 SIMPLIFIED READ COMMAND SEQUENCE**

| OPERATION | COMMENTS |
|---|---|
| (NC) → (MA) | Next command word address to MA. |
| (NC) + 1 → (NC) | Increment Next Command register. |
| ((MA)) → (MB) | Read the command to MB. |
| (MB) → ((MA)) | Restore the memory. |
| (MB) → (CD) | Read the command to CD. |

When the command has been decoded, it will either state that some operand is to be assembled in the Operand Assembly register, OA, or it will call for an arithmetic or logic operation. The latter will usually involve the operand at the location given by the base and index addresses of the command, combined with the contents of the Accumulator, and the result placed back in the Accumulator. If the command called for the contents of base address A and index address I to be added and left in OA, the following sequence of events would occur:

**FIGURE 2.1–4 SIMPLIFIED ADDRESS ASSEMBLY SEQUENCE**

| OPERATION | COMMENTS |
|---|---|
| A → (MA) | Base address to MA. |
| ((MA)) → (MB) | Fetch contents of base address. |
| (MB) → ((MA)) | Restore memory. |
| (MB) → (AU) | Original contents of A is now in Arithmetic Unit. |
| I → (MA) | Index address to MA. |
| ((MA)) → (MB) | Fetch contents of index address. |
| (MB) → ((MA)) | Restore memory. |
| (AU) + (MB) → (AU) | Add index to base address. |
| (AU) → (OA) | Desired operand building block is now in OA. |

When the appropriate operand address has been constructed in OA, it is transferred to MA and used to fetch the operand from memory.

In the performance of a typical arithmetic operation the following steps occur:

**FIGURE 2.1–5 SIMPLIFIED ARITHMETIC OPERATION SEQUENCE**

| OPERATION | COMMENTS |
|---|---|
| (OA) → (MA) | Operand address to MA. |
| ((MA)) → (MB) | Fetch operand. |
| (MB) → ((MA)) | Restore memory. |
| (Acc) $\odot$ (MB) → (Acc) | Answer is now in Accumulator, Acc. |

In the last step $\odot$ is taken to mean any arithmetic or logic operation. When the number in the Accumulator register is to be stored, it is read from Acc to AU to MB to memory.

These few examples by no means exhaust the operations of the G–20, but do indicate the principal paths used in executing commands. Sections 2.3 and 2.4 list all of the methods by which addresses can be constructed and all of the operations which can be performed.

**2.1–3 Arithmetic** The G–20 performs all arithmetic in integer floating point octal form. A number has the form of a positive or negative octal integer multiplied by a positive or negative integral power of 8. Hardware representation is three binary digits for each octal digit of the number. A floating point number is said to be octal-or binary-normalized if its leftmost octal or binary digit is non-zero.

The command structure makes available automatic adjustment of the exponent to a preassigned value, or to zero, to facilitate operations in fixed point or integer form. The Arithmetic Unit performs all arithmetic to 14 octals precision and the numbers can be so stored. It is also possible to store automatically the most significant 7 octals, a fixed exponent number of 9 octals, an integer of 9 octals, or an integer of 7 octals. Address arithmetic is performed by the same unit but is automatically truncated at 5 octals. A non-biasing roundoff rule is applied automatically except in division and "integer" operations, where truncation is used instead.

Thirty-two block commands are provided for repeatedly executing add/subtract, logic, or test operations until a test fails, the end of the block is encountered, or a data flag causes an interrupt. Two flags in each word provide [if enabled] automatic interrupt at flagged commands, flagged data words, or flagged logic words.

A number in the G–20 Arithmetic Unit has the form:

$$\pm \text{ NNNNNNNNNNNNNNN. x } 8^{\pm NN}$$

where N is an octal digit. There can be any number of leading zeros. In case it is not desired to store the full precision of the number, single precision storage is available in the form:

$$\pm \text{ NNNNNNN. x } 8^{\pm NN}$$

Pickapoint storage is also available in the form:

$$\pm \text{ NNNNNNNNN. x } 8^{\pm (PE)}$$

where (PE) is the fixed pickapoint exponent held in register PE and need not be stored with each number. Any single precision number fetched from memory while the pickapoint flip-flop is on will have appended to it the stored exponent (PE). Likewise, during pickapoint storage the number is first shifted so that its exponent equals (PE), and then 9 octals are stored. Integers can be stored in the forms:

± NNNNNNN.<br>
or ± NNNNNNNNN.

according to whether the pickapoint flip-flop is off or on. The equivalent decimal ranges are as indicated in Table 2.1–1.

The G–20 usually rounds answers where required; in a few cases it truncates. Truncation, whether at one end or both, merely discards the unused digits. Round-off where called for is always accomplished as follows: if the discarded portion is less than one-half, it is ignored; if it is greater than one-half, the least significant digit saved is increased to the next higher unit; if it is exactly one-half, the last digit saved is rounded to the nearer odd value. This is analogous to the rule in decimal arithmetic which rounds to the nearer even digit. Its purpose is to eliminate bias and to reduce "noise" in the case of repeated rounding of the same number.

An exponent overflow in the G–20 results in an automatic interrupt request. An exponent underflow will be corrected if possible by shifting the mantissa. When this fails, the number will be replaced by zero, i.e., positive zero exponent and positive zero mantissa.

**TABLE 2.1-1 G-20 NUMBER RANGES**

| | |
|---|---|
| FLOATING POINT MODE | |
| Double Precision | ±4,000,000,000,000. x $10^{\pm 56}$ |
| Single precision | ± 2,000,000. x $10^{\pm 56}$ |
| Integer | ± 2,000,000. |
| PICKAPOINT MODE | |
| Pickapoint | ±130,000,000. [Note] |
| Integer | ±130,000,000. |

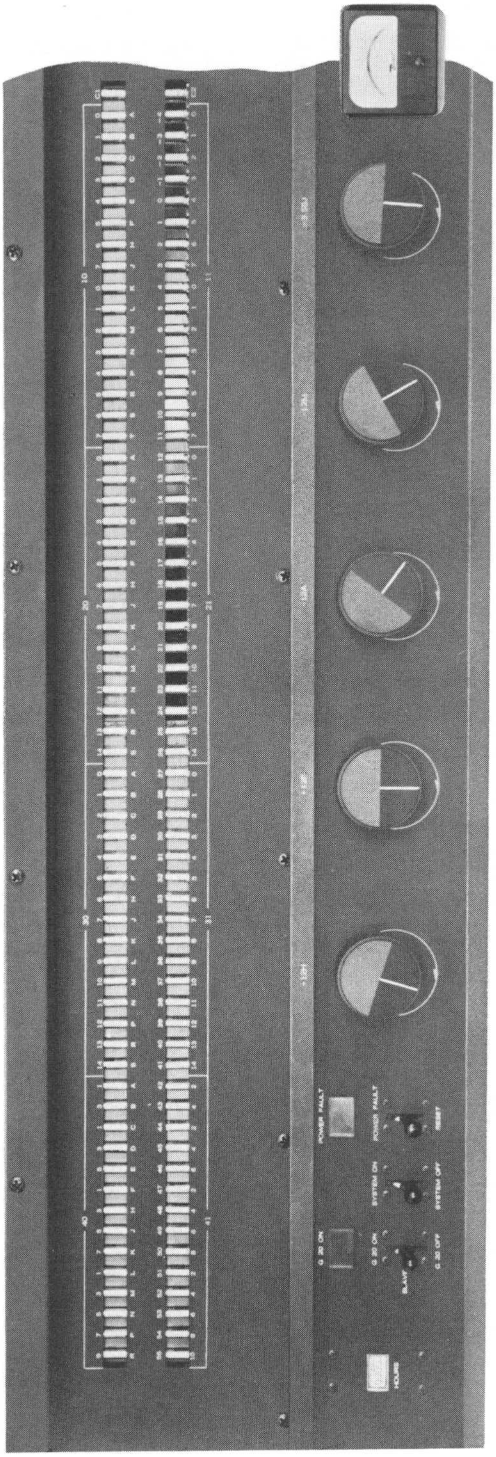
Note: Times the fixed pickapoint exponent whose range is equivalent to $10^{\pm 56}$.

**2.1–4 Central Processor Controls** The controls of the G–20 Central Processor which are of interest to the programmer are located on the control panel shown in Figure 2.1–6, and Figure 2.1–7.

On the lower part of the left side of the control panel are the power controls and indicators. The G–20 ON/SLAVE/OFF lever switch places the Central Processor unconditionally ON, or under control of the SYSTEM power signal line, or unconditionally OFF, respectively. The G–20 ON light shows when the Central Processor is on. The momentary SYSTEM switch can supply a power on or power off pulse to all equipment on the G–20 Communication System via the SYSTEM power signal line. In the event of a power fault, the POWER FAULT light will come on and a tone signal will sound [However, see below]. When the fault has been corrected, operation of the POWER FAULT RESET lever switch will shut off the POWER FAULT light. The five power supply voltage adjustment knobs are not normally of interest to the programmer.

On the lower part of the right side of the control panel are several controls used by the programmer. The INITIAL LOAD lever switch and INITIAL LOAD

## FIGURE 2.1–6 CENTRAL PROCESSOR CONTROL PANEL, LEFT SIDE



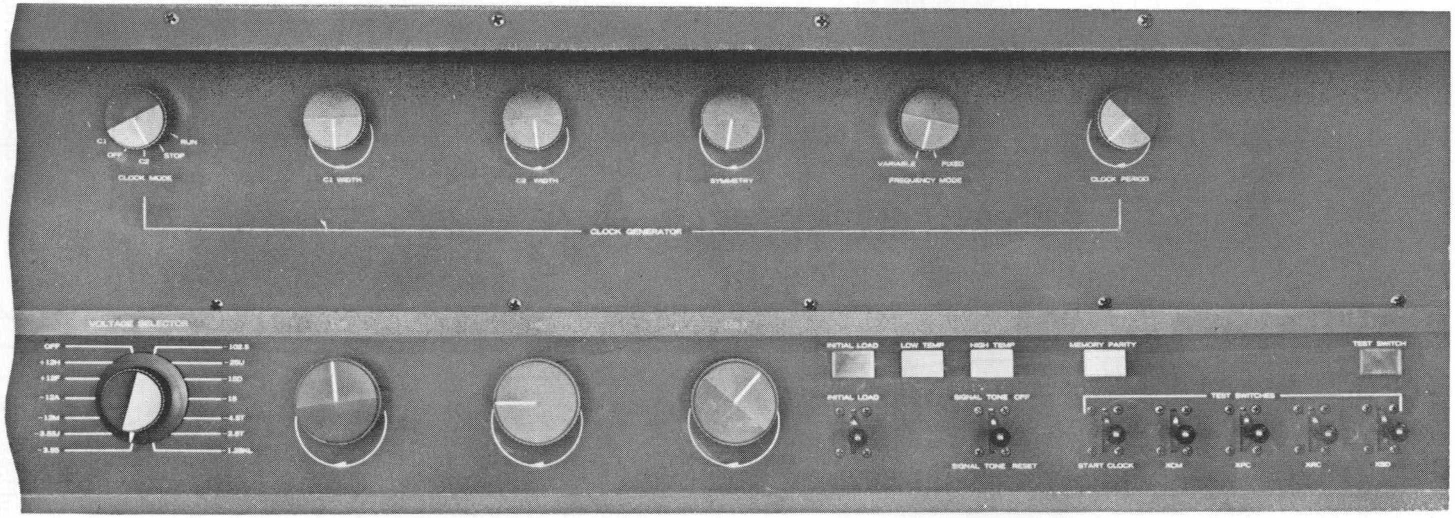


FIGURE 2.1–7 CENTRAL PROCESSOR CONTROL PANEL, RIGHT SIDE

light are used in starting up the computer. If the temperature of the core stack is outside of operating limits, the computer will display a HIGH TEMP or LOW TEMP light, and sound a tone signal [unless in TONE OFF position]. Memory parity error will halt the computer, light the PARITY light, and turn on the tone signal [unless the SIGNAL TONE lever switch is in the TONE OFF position]. Following a parity error, the initial load procedure must be executed. When a fault that has caused a tone signal has been corrected, the SIGNAL TONE lever switch must be operated momentarily to the TONE RESET position to turn off the tone signal. The START CLOCK lever switch is used in conjunction with initial loading. For details of initial loading see the section on the PT–10, PC–10, or CB–11. The balance of the controls on this section of the panel are not used by the programmer.

The upper portion of the left side of the control panel displays the contents of selected registers by means of glow lamps. This information is in binary form.

The upper right portion of the control panel is not used by the programmer.

## SECTION 2.2 ADDRESS FACILITIES

The G–20 provides for flexible addressing and indexing without disturbing the contents of the Accumulator. A string of numbers and/or addresses is built up in a register known as the Operand Assembly register. This string can be used as an operand, or an address, or as the address of the first term of a new string. The process can be continued indefinitely. This makes possible operands of, but not limited to, the forms:

1. ±A±B±. . ±(I)±(J)± . .
2. (± A ± B ± . . ± (I) ± (J) ± . . )
3. ( ± A ± B ± . . ± (I) ± (J) ± . .) ±D±E± . .
   ± (K) ± (L) ± . .
4. (( ± A ± B ± . . ± (I) ± (J) ± . . ) ±D±E± . .
   ± (K) ± (L) ± . . )

where A, B, D, E, etc., are base addresses, I, J, K, L, etc., are index addresses, and ( ), contents of.

The way in which this is accomplished is explained in more detail in the following paragraphs. Some elementary examples are also given.

**2.2–1 Addressing Modes** A command word in the G–20 [see Figure 2.2–1] carries, in addition to the operation code and two flag bits, a 2-bit mode code and two addresses: a base address, A, of 15 bits and an index address, I, of 6 bits. The mode bits specify the way in which A and I are to be interpreted to form an operand or an operand address. The information can be summarized as shown in Table 2.2–1, which indicates the manner in which the operand designator, X, is formed from the previous contents [if any] of the Operand Assembly register combined with the two addresses.

**FIGURE 2.2-1 COMMAND WORD**

| 2 | 2 | 7 | 6 | 15 |
|---|---|---|---|---|
| FF | | OP | I | A |
| 31 | ↑<br>MODE | | | 0 |

**TABLE 2.2-1 ADDRESSING MODES**

| MODE | ACTION |
|---|---|
| 0 | (OA) + A + (I) = X |
| 1 | (OA) + (A) + (I) = X |
| 2 | ((OA) + A + (I)) = X |
| 3 | ((OA) + (A) + (I)) = X |

where OA = Operand Assembly register
X = operand designator
A = base address
I = index address
( ) = contents of

Note: A few commands cannot be indexed. In particular the commands which operate on a bus register or an index register use the I field of the command word to specify the register to be operated upon. Address arithmetic can however be used to give an equivalent result.

The hardware executes the equations of Table 2.2-1 from left to right, and from inside outward. Intermediate results are held in OA.

In coding, the convention has been established that the presence or absence of a single symbol following the opcode distinguishes a 0 or 1 address mode from a 2 or 3 address mode. Class 1 opcodes, in which the operand designator signifies an operand, i.e., preparation, arithmetic, logic, index, bus register and single-character transmission commands, use a minus to indicate modes 0 or 1 and no symbol for modes 2 or 3. Class 2 opcodes, in which the operand designator is meaningful only as a location, i.e., store, control, repeat, and block input/output commands, use no symbol for modes 0 and 1, and an asterisk for modes 2 and 3. There is one special case, opcode SKP, which uses an operand to determine a location. Opcode SKP is coded as Class 1. Selection of this method for coding will result in a large fraction of the commands in an application program having no "mode flag."

**2.2–2 Simple Addressing** In simple addressing with single word commands (OA) is zero at the start of the operation and no index is used. The coding is indicated in the tables below. In all of the following tables OP is an arbitrary opcode.

**TABLE 2.2-2 SIMPLE ADDRESSING, CLASS 1 OPCODES**

| MODE | CODING | | ACTUAL OPERAND |
|---|---|---|---|
| 0 | OP– | A | A |
| 1 | OP– | (A) | (A) See note |
| 2 | OP | A | (A) See note |
| 3 | OP | ( A) | ((A)) |

Note: Modes 1 and 2 do not always give the same result, as some logic codes use logic format access in mode 2, and number format access in mode 1.

**TABLE 2.2-3 SIMPLE ADDRESSING, CLASS 2 OPCODES**

| MODE | CODING | | LOCATION USED |
|---|---|---|---|
| 0 | OP | A | A |
| 1 | OP | (A) | (A) |
| 2 | OP* | A | (A) |
| 3 | OP* | (A) | ((A)) |

**2.2–3 Addressing With One Index** Because the I field contains six bits, it can address 63 of the locations in core memory, which can thus be used as Index registers. [Location 0 is not an Index register and a zero I address will be ignored by the machine. See Section 2.3–6.]

For operations with one index, the possible forms of addressing and their coding are as follows:

**TABLE 2.2-4 SINGLE INDEX ADDRESSING, CLASS 1 OPCODES**

| MODE | CODING | | ACTUAL OPERAND |
|---|---|---|---|
| 0 | OP– | A + (I) | A + (I) |
| 1 | OP– | (A) + (I) | (A) + (I ) |
| 2 | OP | A + (I) | (A + (I)) |
| 3 | OP | (A) + (I) | ((A) + (I)) |

**TABLE 2.2-5 SINGLE INDEX ADDRESSING, CLASS 2 OPCODES**

| MODE | CODING | | LOCATION USED |
|---|---|---|---|
| 0 | OP | A + (I) | A + (I) |
| 1 | OP | (A) + (I) | (A) + (I) |
| 2 | OP* | A + (I) | (A + (I)) |
| 3 | OP* | (A) + (I) | ((A) + (I)) |

Since this form of addressing is limited to 63 indices, problems requiring a greater number of indices should make use of unrestricted general addressing described in Section 2.2–4.

**2.2–4 Address Arithmetic** Address arithmetic in the G–20 is facilitated by the preparation opcodes. Through the use of these codes the following can be accomplished:

1. An operand can be used as an address.
2. An arithmetic or logic result standing in the Accumulator register can be used as an address.
3. The contents of the Accumulator can be combined with an operand using the add/subtract hardware and the result then used as an address.
4. The address can be repeatedly operated upon by the addressing modes and the add/subtract hardware as in 3 above.

The intermediate results are always held in OA and need not be stored in core memory with resulting loss of time.

Table 2.2–6 summarizes the preparation opcodes. In this table, X represents the operand designator resulting from the combination of A, I, and the previous contents of OA according to Table 2.2–1. The arrow signifies "replaces". Acc is the Accumulator register.

## TABLE 2.2-6 ADDRESS PREPARATION OPERATIONS [OA COMMANDS]

| OPCODE | NAME | OPERATION |
|---|---|---|
| OCA | Clear and add [answer to OA] | $X \rightarrow (OA)$ |
| OCS | Clear and subtract [answer to OA] | $-X \rightarrow (OA)$ |
| OAD | Add [answer to OA] | $X + (Acc) \rightarrow (OA)$ |
| OSU | Subtract [answer to OA] | $-X + (Acc) \rightarrow (OA)$ |
| OAN | Add and negate [answer to OA] | $-[X + (Acc)] \rightarrow (OA)$ |
| OSN | Subtract and negate [answer to OA] | $-[-X + (Acc)] \rightarrow (OA)$ |
| OAA | Add and take absolute value [answer to OA] | $\lvert X + (Acc)\rvert \rightarrow (OA)$ |
| OSA | Subtract and take absolute value [answer to OA] | $\lvert -X + (Acc)\rvert \rightarrow (OA)$ |

Note: The Accumulator is not disturbed in these opcodes. These commands plus one other, ERO, taken up under register opcodes, Section 2.3-9, give all the means available for dealing with (OA). These are Class 1 opcodes. The hardware executes the operations from left to right and from inside outward.

In order to determine the result of an OA or address preparation opcode, compute the operand designator, X, according to Table 2.2–1, and then perform the operation shown in Table 2.2–6. Thus, for the command:

$$\text{OAD} \qquad A + (I)$$

the mode is 2, as there is no minus sign following the opcode. Therefore, the operand designator according to Table 2.2–1 is:

$$X = ((OA) + A + (I)).$$

The operation indicated in Table 2.2–6 is now performed using this value of X. The over-all result is:

$$((OA) + A + (I)) + (Acc) \rightarrow (OA).$$

Again, the command:

$$\text{OSN}- \qquad (A) + (I),$$

results in:

$$-[-[(OA) + (A) + (I)] + (Acc)] \rightarrow (OA).$$

Numerous other possibilities will present themselves. The programmer should become familiar with them.

NUMBER FORMAT. In most instances a number called from memory is interpreted as a single or double precision floating point number, or as a pickapoint number, according to the length flag on the number and the state of the pickapoint flip-flop. The mantissa digits are placed in the Accumulator, and the exponent digits in an exponent register. For pickapoint numbers the exponent is taken from PE. This is referred to as a "number format" access. Numbers written in the command [address mode 0] will be supplied with positive sign and zero exponent.

LOGIC FORMAT. The exceptions to number format access are modes 2 and 3 with logic opcodes or logic tests. In these cases the final memory access is in logic format, which means that the bits just as they appear in memory are placed in the right end of the Accumulator; positive sign and zero exponent are supplied by the hardware.

A number to be used as an address will be shifted to zero exponent and truncated so as to save the least significant 15 bits. The Operand Assembly register is cleared after all operations except preparation opcodes and ERO, extract register into OA, where the result is to be saved for subsequent manipulations. An attempt to use a negative or other nonexistent address will result in an interrupt request.

## SECTION 2.3 OPERATION CODES

The operation codes of the G–20 include arithmetic operations, arithmetic tests, logic operations, logic tests, as well as store, register, index, transfer control, and single character output commands. There are also block arithmetic commands and block input/output commands. A block command occupies two words in memory.

All arithmetic commands are carried out in double precision floating point form [14-octal mantissa plus 2-octal exponent]. The distinction between double precision and single precision exists only in store commands. Logic operations are carried out to 32–bit precision and the other bits, 41–32, of the Accumulator are cleared. Arithmetic test and logic test commands, except when repeated for a block, leave the Accumulator undisturbed. The indicated operations are performed from left to right and from inside outward.

The mnemonic forms and processing rules for the various opcodes are outlined below by categories. The symbols used in these tables are defined as follows:

| | |
|---|---|
| → | replaces |
| ( ) | contents of |
| \| \| | absolute value |
| $\wedge$ | logical intersection, extraction, Boolean AND [1–bit in answer wherever 1–bit appears in both operands]. |
| $\vee$ | logical union, Boolean OR [1–bit in answer wherever 1–bit appears in either operand]. |
| $\overline{A}$ | logical negation of A [exchanges 0's for 1's and 1's for 0's]. |
| EXP | Exponent of (Acc). |
| PE | Pickapoint Exponent register. |
| NC | Next Command address register. |
| $_m[ABC]_n$ | bits m through n of expression ABC. |
| * | multiply |

**TABLE 2.3-1 ADD/SUBTRACT OPERATIONS**

| OPCODE | NAME | OPERATION |
|---|---|---|
| CLA | Clear and add | X → (Acc) |
| CLS | Clear and subtract | −X → (Acc) |
| ADD | Add | X + (Acc) → (Acc) |
| SUB | Subtract | −X + (Acc) → (Acc) |
| ADN | Add and negate | −[X + (Acc)] → (Acc) |
| SUN | Subtract and negate | −[− X + (Acc)] → (Acc) |
| ADA | Add and take absolute value | \|X + (Acc)\| → (Acc) |
| SUA | Subtract and take absolute value | \|−X + (Acc)\| → (Acc) |

**TABLE 2.3–2 ADD/SUBTRACT TESTS**

| OPCODE | NAME | CRITERION |
|---|---|---|
| FOP | If operand plus | X > 0 |
| FOM | If operand minus | −X > 0 |
| FGO | If (Acc) greater than operand | −X + (Acc) > 0 |
| FLO | If (Acc) less than operand | − [− X + (Acc)] > 0 |
| FUO | If (Acc) unequal to operand | \|− X + (Acc)\| > 0 |
| FSP | If sum plus | X + (Acc) > 0 |
| FSM | If sum minus | − [ X + (Acc)] > 0 |
| FSN | If sum non-zero | \| X + (Acc)\| > 0 |

Take next command if test is satisfied; otherwise skip one command word, that is, increment (NC) before continuing.

### 2.3–1 Add/Subtract Operations and Tests

Add/Subtract opcodes are shown in Table 2.3–1 and add/subtract tests, in Table 2.3–2. These are Class 1 codes.

Add/subtract operations and add/subtract tests are performed in floating point arithmetic. Left or right shifts are performed only when needed to equalize exponents or prevent overflow or underflow. Equalization of exponents is always first attempted by left shift of the number with larger exponent. Only when this fails to be sufficient is the other number shifted right as far as required and rounded according to the round-off rule. The discarded digits are examined and enough information remembered to make the correct round-off after shifting is completed. If, after the addition, a right shift is required to prevent mantissa overflow, the answer is rounded according to the same rule.

Because the hardware picks up (I) in number format, it is possible to make use of the address modes to perform additional varieties of arithmetic by regarding (I) as an operand rather than as an address modifier. Thus using mode 1 [or mode 0] we can add two numbers to (OA) or (Acc) in a single command:

OCA– $(A) + (I)$,

which gives

$$(OA) + (A) + (I) \rightarrow (OA)$$

CLA– $(B) + (J)$,

which gives

$$(OA) + (B) + (J) + (Acc) \rightarrow (Acc)$$

for a net result:

$$(OA) + (A) + (I) + (B) + (J) + (Acc) \rightarrow (Acc)$$

Summation of a few quantities by this method may be preferable to a repeat command.

Add/subtract tests follow the same rules as add/subtract operations except that the Accumulator is not disturbed.

### 2.3–2 Logic Operations and Tests

Tables 2.3–3 and 2.3–4 show the logic operations and logic tests of the G–20. These are Class 1 codes.

Logic opcodes and logic test opcodes draw words from memory in the *number format* except for the final access in modes 2 and 3 which use *logic format.* A logic format word is supplied with a zero exponent, making it in effect an integer. Bits 41–32 of the Accumulator are cleared in logic operations. In logic tests the Accumulator is undisturbed.

In all logic opcodes except ADL and SUL, all operands are shifted to zero exponent and truncated. The operation is then performed and the result of the operation is truncated at both ends to 32 bits. In ADL and SUL floating point arithmetic is performed and the answer is then shifted to zero exponent and truncated. It is possible to perform limited integer arithmetic to 32 bits by using opcodes CAL, ADL, SUL and STL. An overflow will, however, cause an erroneous answer with no warning. [For STL see Section 2.3–5.]

**TABLE 2.3–3 LOGIC OPERATIONS**

| OPCODE | NAME | OPERATION |
|---|---|---|
| CAL | Clear and add logic word | ${}_{31}[\ X\ ]_0 \rightarrow (Acc)$ |
| CCL | Clear and add complement of logic word | ${}_{31}[\ \bar{X}\ ]_0 \rightarrow (Acc)$ |
| ADL | Add logic word | ${}_{31}[\ X + (Acc)]_0 \rightarrow (Acc)$ |
| SUL | Subtract logic word | ${}_{31}[-X + (Acc)]_0 \rightarrow (Acc)$ |
| EXL | Extract with logic word | ${}_{31}[\ X \wedge (Acc)]_0 \rightarrow (Acc)$ |
| ECL | Extract with complement of logic word | ${}_{31}[\ \bar{X} \wedge (Acc)]_0 \rightarrow (Acc)$ |
| UNL | Unite with logic word | ${}_{31}[\ X \vee (Acc)]_0 \rightarrow (Acc)$ |
| UCL | Unite with complement of logic word | ${}_{31}[\ \bar{X} \vee (Acc)]_0 \rightarrow (Acc)$ |

Note: See Section 2.3 for meaning of symbols. $0 \rightarrow {}_{41}[(Acc)]_{32}$ for all of these codes.

**TABLE 2.3–4 LOGIC TESTS**

| OPCODE | NAME | CRITERION |
|---|---|---|
| IOZ | If operand zero | ${}_{31}[\ X\ ]_0 \quad = 0$ |
| ICZ | If complement of operand zero | ${}_{31}[\ \overline{X}\ ]_0 \quad = 0$ |
| ISN | If sum non–zero | ${}_{31}\|\ X + (Acc)\ \|_0 \quad > 0$ |
| IUO | If unequal to operand | ${}_{31}\|-X + (Acc)\ \|_0 \quad > 0$ |
| IEZ | If extraction zero | ${}_{31}[\ X \wedge (Acc)\ ]_0 \quad = 0$ |
| IEC | If extraction with complement zero | ${}_{31}[\ \overline{X} \wedge (Acc)\ ]_0 \quad = 0$ |
| IUZ | If union zero | ${}_{31}[\ X \vee (Acc)\ ]_0 \quad = 0$ |
| IUC | If union with complement zero | ${}_{31}[\ \overline{X} \vee (Acc)\ ]_0 \quad = 0$ |

If test is satisfied, take next command; otherwise skip one command word, that is, increment (NC) before continuing.

Note: See Section 2.3 above for meaning of symbols.

In ADL and SUL, the final sign of (Acc) depends on the result of the operation. In the other cases the sign of the Accumulator does not depend on the operation. In CAL and CCL, the final sign of the (Acc) is the original sign of X for modes 0 and 1, and always positive for modes 2 and 3. In all other cases the previous sign of (Acc) is retained in the answer.

In logic tests except ISN and IUO, all operands are reduced to zero exponent, the indicated quantity is formed, truncated at both ends, and compared with zero. If the quantity is zero, the next command is taken in sequence; otherwise, one command word is skipped. In ISN and IUO the operand is added to, or subtracted from, (Acc) in floating point form, and the answer is reduced to zero exponent, truncated at both ends, and compared with zero. The next command is taken if the magnitude of the result is greater than zero, and one command word is skipped if the result is equal to zero.

**2.3–3 Repeated Commands** It is possible to execute a block of add/subtract operations, add/subtract tests, logic operations, or logic tests. The mnemonic opcode for such a block command contains an R [for repeat] as its first letter. [Because of mode considerations the "R" does not mean "Repeat this opcode".] In the convention established for coding, the operand address is followed by the block length set off by a comma, as shown below:

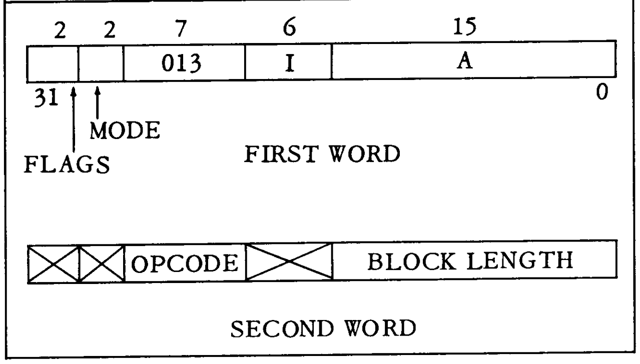
```
RADD        A+ (I), B
```

which states: repeat add for block length B the numbers starting at location A + (I). Repeat codes are Class 2. [Simple arithmetic and logic opcodes are Class 1.]

A repeated command is always terminated if the block length is reached. A block of zero is forbidden. A repeated test command may be terminated sooner by the failure of an operand to meet the criterion tested for. Such an operand would cause a command word to be skipped if a simple [i.e., not repeated] test command were being executed. A repeated command can also be terminated by a logic or data flag if enabled, or by an overflow, or illegal code or address [see Section 2.4].

Because of the additional information which it must contain, a repeat command occupies two machine words rather than the usual one. This fact should be taken into account in reckoning the number of words occupied by a program. The composition of the two-word command is shown in Figure 2.3–1. A repeat command does *not* signify that the simple command obtained by deleting the "R" is to be executed repeatedly.

**FIGURE 2.3–1 REPEAT COMMAND FORMAT**



REPEATED ADD/SUBTRACT OPERATIONS, TABLE 2.3–5. The commands RCLA and RCLS, since they clear the Accumulator, are useful only for bringing into the Accumulator the first flagged operand in a block. If there is no flagged operand in the block, the last operand in the block will be left in the Accumulator at the conclusion of the operation.

The other repeated add/subtract commands give the results in Table 2.3–5. Repeat add/subtract commands work for mixed single and double precision operands. The limit applies to operands, not to words. See also Section 2.4–3.

REPEATED ARITHMETIC TESTS, TABLE 2.3–6. Ordinarily the testing will continue until terminated by a failure to meet the criterion, *a data flag*, or the set upper limit. There will be standing in the Accumulator an address one greater than that for which the test failed. Both single and double precision operands are handled. If the repeated arithmetic test locates an operand failing the test criterion, computation is resumed after skipping one command word by incrementing (NC). If an enabled *data flag* or the end of the block is reached before such an operand has been found, computation continues in sequence. A flag causes an interrupt request. When an interrupt request terminates the operation, computation resumes at (NC) or (NC) + 1 according to whether or not the criterion was met. See Section 2.4–3.

REPEATED LOGIC OPERATIONS, TABLE 2.3–7. In repeated logic operations the operands are always fetched from memory in *logic format.*

The logic commands which clear the Accumulator, RCAL and RCCL, are mainly useful for picking up the first flagged operand or its complement. The command RADL [or RSUL] adds [or subtracts] each operand to the contents of the Accumulator, rounding if necessary, after which the partial answer

**TABLE 2.3–5 REPEATED ADD/SUBTRACT OPERATIONS**

| OPCODE | NAME | OPERATION |
|---|---|---|
| RCLA | Repeat clear and add | $Q_i \rightarrow (\text{Acc})$ |
| RCLS | Repeat clear and subtract | $-Q_i \rightarrow (\text{Acc})$ |
| RADD | Repeat add | $(\text{Acc}) + Q_1 + Q_2 + \ldots + Q_n \rightarrow (\text{Acc})$ |
| RSUB | Repeat subtract | $(\text{Acc}) - Q_1 - Q_2 - \ldots - Q_n \rightarrow (\text{Acc})$ |
| RADN | Repeat add and negate | $-[-\ldots-[-[(\text{Acc}) + Q_1] + Q_2] + \ldots + Q_n] \rightarrow (\text{Acc})$ |
| RSUN | Repeat subtract and negate | $-[-\ldots-[-[(\text{Acc}) - Q_1] - Q_2] - \ldots - Q_n] \rightarrow (\text{Acc})$ |
| RADA | Repeat add and take absolute value | $\vert \ldots \vert \vert (\text{Acc}) + Q_1 \vert + Q_2 \vert + \ldots + Q_n \vert \rightarrow (\text{Acc})$ |
| RSUA | Repeat subtract and take absolute value | $\vert \ldots \vert \vert (\text{Acc}) - Q_1 \vert - Q_2 \vert - \ldots - Q_n \vert \rightarrow (\text{Acc})$ |

Note: See Section 2.3 above for meaning of symbols. In repeat commands the starting location is determined from X, the operand designator. The individual operands, $Q_i$, of the block, are floating point single or double precision quantities.

### TABLE 2.3–6 REPEATED ARITHMETIC TESTS

| OPCODE | NAME | CRITERION |
|---|---|---|
| RFOP | Repeat if operand plus | $Q_i > 0$ |
| RFOM | Repeat if operand minus | $-Q_i > 0$ |
| RFGO | Repeat if (Acc) greater than operand | $(Acc) - Q_i > 0$ |
| RFLO | Repeat if (Acc) less than operand | $-[(Acc) - Q_i] > 0$ |
| RFUO | Repeat if (Acc) unequal to operand | $\lvert(Acc) - Q_i\rvert > 0$ |
| RFSP | Repeat if sum plus | $(Acc) + Q_i > 0$ |
| RFSM | Repeat if sum minus | $-[(Acc) - Q_i] > 0$ |
| RFSN | Repeat if sum non-zero | $\lvert(Acc) + Q_i\rvert > 0$ |

Note: See Section 2.3 for meaning of symbols. The starting location is determined from X. The operands $Q_i$ are single or double precision floating point quantities.

is reduced to zero exponent and truncated to 32 bits. The operation is terminated by flag or block limit.

Commands REXL, RECL, RUNL, and RUCL give the logical union or intersection of the contents of the Accumulator and all elements in the block [or their complements]. The operation is terminated by a *logic flag* or block limit. See also Section 2.4–3.

### TABLE 2.3–7 REPEATED LOGIC OPERATIONS

| OPCODE | NAME | OPERATION |
|---|---|---|
| RCAL | Repeat clear and add logic word | $L_i \rightarrow (Acc)$ |
| RCCL | Repeat clear and add complement of logic word | $\bar{L}_i \rightarrow (Acc)$ |
| RADL | Repeat add logic word | ${}_{31}[\cdots{}_{31}[{}_{31}[(Acc) + L_1]_0 + L_2]_0 + \dots + L_n]_0 \rightarrow (Acc)$ |
| RSUL | Repeat subtract logic word | ${}_{31}[\cdots{}_{31}[{}_{31}[(Acc) - L_1]_0 - L_2]_0 - \dots - L_n]_0 \rightarrow (Acc)$ |
| REXL | Repeat extract with logic word | ${}_{31}[(Acc) \wedge L_1 \wedge L_2 \wedge \dots \wedge L_n]_0 \rightarrow (Acc)$ |
| RECL | Repeat extract with complement of logic word | ${}_{31}[(Acc) \wedge \bar{L}_1 \wedge \bar{L}_2 \wedge \dots \wedge \bar{L}_n]_0 \rightarrow (Acc)$ |
| RUNL | Repeat unite with logic word | ${}_{31}[(Acc) \vee L_1 \vee L_2 \vee \dots \vee L_n]_0 \rightarrow (Acc)$ |
| RUCL | Repeat unite with complement of logic word | ${}_{31}[(Acc) \vee \bar{L}_1 \vee \bar{L}_2 \vee \dots \vee \bar{L}_n]_0 \rightarrow (Acc)$ |

Note: See Section 2.3 for meaning of symbols. The starting location is determined from X. The operands $L_i$ are logic words.

## TABLE 2.3–8 REPEATED LOGIC TESTS

| OPCODE | NAME | CRITERION |
|---|---|---|
| RIOZ | Repeat if operand zero | $L_i = 0$ |
| RICZ | Repeat if complement zero | $\bar{L}_i = 0$ |
| RIUO | Repeat if unequal to operand | ${}_{31}\lvert(\text{Acc}) - L_i\rvert_0 > 0$ |
| RISN | Repeat if sum non-zero | ${}_{31}\lvert(\text{Acc}) + L_i\rvert_0 > 0$ |
| RIEZ | Repeat if extraction zero | ${}_{31}[(\text{Acc}) \wedge L_i]_0 = 0$ |
| RIEC | Repeat if extraction with complement zero | ${}_{31}[(\text{Acc}) \wedge \bar{L}_i]_0 = 0$ |
| RIUZ | Repeat if union zero | ${}_{31}[(\text{Acc}) \vee L_i]_0 = 0$ |
| RIUC | Repeat if union with complement zero | ${}_{31}[(\text{Acc}) \vee \bar{L}_i]_0 = 0$ |

Note: See Section 2.3 for meaning of symbols. The starting location is determined from the operand designator, X. The operands $L_i$ are logic words.

REPEATED LOGIC TESTS, Table 2.3–8. These tests continue as long as operands are found which satisfy the criterion. When an operand fails the criterion, computation is resumed after skipping one command word. If no operand is found which fails the criterion, the sequence will be terminated by a *logic flag* or end of block. If termination was by end of block, computation will resume in sequence, from which point a jump can be executed. If the operation is terminated by a flag, an interrupt will be requested; computation will resume at (NC) or (NC) + 1 according to whether or not the criterion was met. See Section 2.4–3.

These opcodes operate on single-word operands only and on the least significant 32 bits of the Accumulator. The result is reduced to zero exponent and truncated on both ends before the test is performed.

At the conclusion of the operation there will be standing in the Accumulator an address one greater than that for which the test failed.

**2.3–4 Multiply, Divide, Shift** All multiplication, division, and shifting in the G–20 is performed by means of the three commands shown in Table 2.3–9. These are Class 1 codes.

## TABLE 2.3–9 MULTIPLY/DIVIDE

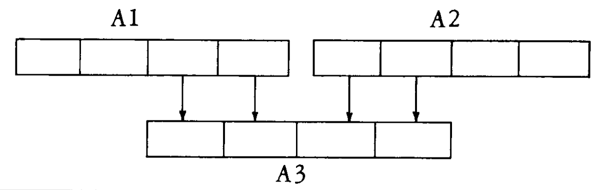
| OPCODE | NAME | OPERATION |
|---|---|---|
| MPY | Multiply | (Acc) * X → (Acc) |
| DIV | Divide | (Acc) / X → (Acc) |
| RDV | Reverse Divide | X / (Acc) → (Acc) |

MULTIPLICATION. A floating point multiplication is performed using (Acc) as the multiplicand and X as the multiplier. The multiplicand is first binary-normalized. Two 14–octal mantissas are multiplied together to produce a 14–octal product. Rounding is used if necessary to keep the product within 14 octals. See Section 2.1–3. A right shift of the product mantissa will be made if necessary to prevent exponent underflow, but otherwise it will be normalized. A product of magnitude smaller than $8^{-63}$ [$\sim 10^{-57}$] will be replaced by zero. Exponent overflow causes an interrupt request. Multiplication terminates when the non-zero bits of the multiplier have all been used.

DIVISION. Division of either variety takes the floating point quotient of a 14–octal dividend by a 14–octal divisor and always produces a 14–octal result. Both operands are first binary-normalized. The quotient is truncated and the remainder is lost. The mantissa of the quotient will be shifted right if required to prevent exponent underflow, but otherwise will be normalized. A quotient of magnitude smaller than $8^{-63}$ [ $\sim 10^{-57}$ ] will be replaced by a zero. Exponent overflow will cause an interrupt request, as will an attempted division by zero. Use of a zero dividend will merely result in a zero answer.

SHIFTING. Shifting logic words in the G–20 is accomplished by taking advantage of the hardware property which can automatically shift a number left or right so as to reduce its exponent to zero, truncating if necessary. Shifting is thus a matter of adjusting the magnitude by multiplying by 1, 2, or 4 * $8^{\pm NN}$ followed by a reduction to zero exponent. All logic operations reduce exponents to zero and truncate to 32 bits. For example, to shift a logic word eight bits to the left, one multiplies the number by $2^8=4 * 8^2$. The multiplication will leave the mantissa in the Accumulator, octal-normalized, with the appropriate exponent. Any logic operation on the product causes an automatic reduction to zero exponent and returns the mantissa to the desired form. Since a logic operation frequently follows the shifting, an extra command is usually not needed. An example is a shift followed by a logic store or a shift followed by a logical union. In case no logic operation follows the shift, it is necessary to insert a dummy operation such as unite with zero or logic add to zero.

**FIGURE 2.3–2 SHIFT ILLUSTRATION**



Thus, to combine the last two characters of word A1 and the first two characters of word A2 and store the result in A3, as in Figure 2.3–2, one can write:

| OPCODE | ADDRESS | REMARKS |
|---|---|---|
| CAL | A1 | Bring in A1. |
| MPY | L16 | Adjust magnitude [L16 contains $2 * 8^5 = 2^{16}$]. |
| STL | A3 | Reduce to zero exponent, truncate and store. [The number will still be normalized in Acc]. |
| CAL | A2 | Bring in A2. |
| MPY | R16 | Adjust magnitude [R16 contains $4 * 8^{-6} = 2^{-16}$]. |
| UNL | A3 | Reduce to zero exponent, truncate and unite with A3. |
| STL | A3 | Store. |

To store the second character of A1 in the least significant end of A2, one can write:

| OPCODE | ADDRESS | REMARKS |
|---|---|---|
| CAL | A1 | Bring in the word. |
| MPY – | 256 | Adjust magnitude by $2^8 = 256$. |
| ADL – | 0 | Shift left, truncating leftmost 8 bits. |
| MPY | R24 | Adjust magnitude [R24 contains $8^{-8} = 2^{-24}$]. |
| STL | A2 | Shift to zero exponent, truncate at the right and store. |

Certain types of shifts of *floating point* numbers can be accomplished as follows: A number can be converted to a fixed [but not smaller] exponent by adding and subtracting any normalized number having the desired exponent. The number being shifted will be rounded by the rounding rule.

A number can be normalized [leading octal digit non-zero] by multiplying by unity.

**2.3–5 Storage Operations and Formats** As noted in Table 2.3–10, there are four storage commands: store logic word, store double precision, store single precision, and store integer, plus a special command which will clear a memory location by storing a zero in it. In store single precision and store integer, the

**TABLE 2.3–10 STORE OPERATIONS**

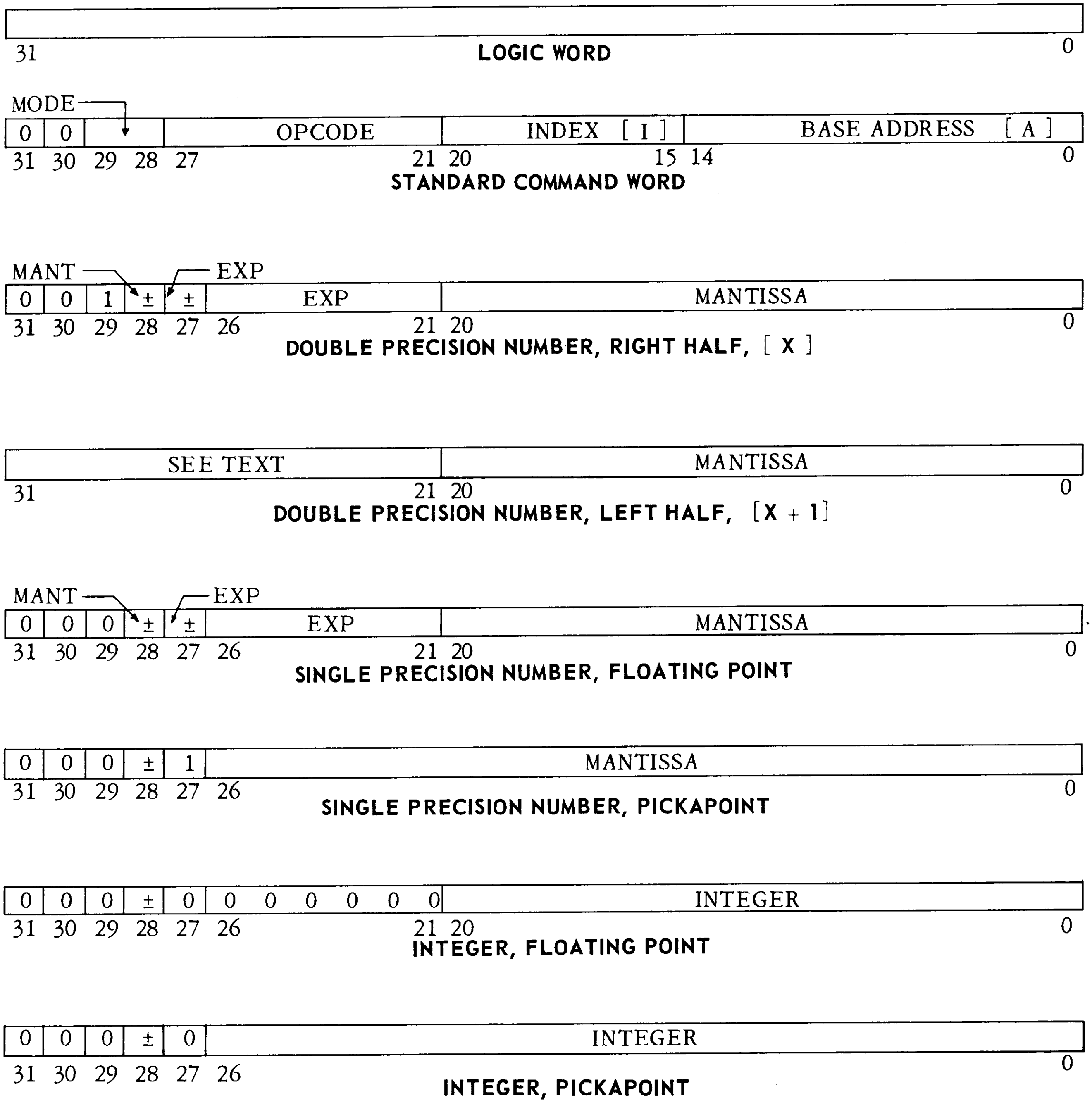
| OPCODE | NAME | OPERATION |
|---|---|---|
| STL | Store logic word | $0 \to \text{EXP}$ |
| | | ${}_{31}(\text{Acc})_0 \to (X)$ |
| STD | Store double precision [1] | ${}_{20}(\text{Acc})_0 \to (X)$ |
| | | ${}_{41}(\text{Acc})_{21} \to {}_{20}(X+1)_0$ |
| STS | Store single precision [1] | |
| | [a] floating point mode[2] | ${}_{41}(\text{Acc})_{21} \to {}_{20}(X)_0$ |
| | [b] pickapoint mode | $(\text{PE}) \to \text{EXP}$ |
| | | ${}_{41}(\text{Acc})_{15} \to {}_{26}(X)_0$ |
| STI | Store integer [1] | |
| | [a] floating point mode | $0 \to \text{EXP}$ |
| | | ${}_{20}(\text{Acc})_0 \to {}_{20}(X)_0$ |
| | [b] pickapoint mode | $0 \to \text{EXP}$ |
| | | ${}_{26}(\text{Acc})_0 \to {}_{26}(X)_0$ |
| STZ | Store zero | $0 \to {}_{31}(X)_0$ |

Note: See Section 2.3 above for meaning of symbols. When the quantity following → is an exponent, a corresponding mantissa shift is implied.

[1] See also Figure 2.3–3 for sign and exponent.

[2] (Acc) is first octal-normalized to the left.

FIGURE 2.3–3 STORAGE FORMATS



LOGIC WORD

STANDARD COMMAND WORD

DOUBLE PRECISION NUMBER, RIGHT HALF, [ X ]

DOUBLE PRECISION NUMBER, LEFT HALF, [X + 1]

SINGLE PRECISION NUMBER, FLOATING POINT

SINGLE PRECISION NUMBER, PICKAPOINT

INTEGER, FLOATING POINT

INTEGER, PICKAPOINT

action depends upon the state of the pickapoint flip-flop [bit 8 of register CE]. This gives, in effect, six varieties of storage plus store zero. The Accumulator is not disturbed in any storage command. The formats are summarized in Figure 2.3–3. The operand designator X is shifted to zero exponent and truncated to 5 octals before being used as an address. Store commands follow the usual rules for indexing. Note that store commands are Class 2 codes. See Section 2.2–1.

STORE LOGIC WORD. The number in the Accumulator is shifted to zero exponent and truncated. The least significant 32 bits are stored. The sign of the Accumulator is ignored. This command is the only one which will store a flag.

STORE DOUBLE PRECISION. The number in the Accumulator is stored in floating point format. The less significant half with sign and exponent is stored in location X with flags 30 and 31 cleared. The more significant half of the mantissa is stored in bits 0-20 of location X + 1. Bits 21-31 of location X + 1 receive a copy of bits 21-31 of location X. Note the marker in bit 29 of location X. ·When the word is fetched in number format, this bit tells the hardware to fetch the contents of location X + 1 also. On double precision *access*, bits 21-31 of location X + 1 are ignored.

STORE SINGLE PRECISION. When the pickapoint flip-flop is on [pickapoint mode], this command produces a pickapoint storage of 9 octals. The number is first shifted until its exponent equals the pickapoint exponent [held in PE] and it is rounded by the rounding rule [see Section 2.1-3]. It is then stored in pickapoint format. Flags 30 and 31 are cleared. During exponent adjustment, an attempt to generate a number of more than 9 octals results in an interrupt request.

When the pickapoint flip-flop is off [floating point mode], this command produces floating point storage of 7 octals. The number in the Accumulator is shifted and rounded to 7 octals and stored in the floating point mode. An exponent overflow during shift results in an interrupt request.

STORE INTEGER. In integer storage the number is always truncated and overflows are ignored. With pickapoint flip-flop on, the number in the Accumulator is shifted to zero exponent and the least significant 9 octals stored. Bits 30 and 31 are cleared. With pickapoint flip-flop off, the number in the Accumulator is shifted to zero exponent and the least significant 7 octals stored. Bits 30 and 31 are cleared.

STORE ZERO. In store zero all 32 bits of the specified location are cleared to zero.

**2.3-6 Index Operations** The operations involving setting and incrementing Index registers, as shown in Table 2.3-11, do not permit indexing of the command itself because the index address in the command is used to specify the Index register on which the incrementation, etc., is to be performed.

**TABLE 2.3–11 INDEX**

| OPCODE | NAME | OPERATION |
|---|---|---|
| LXP | Load index plus | X → (I) |
| LXM | Load index minus | –X → (I) |
| ADX | Add to index | X +(I) → (I) |
| SUX | Subtract from index | –X +(I) → (I) |

In the following opcodes, if the final value of the index is not zero, take the next command; if the final value of the index is zero, skip one command word, that is, (NC) + 1 → (NC).

| OPCODE | NAME | OPERATION |
|---|---|---|
| XPT | Load index plus & test | X → (I) |
| XMT | Load index minus & test | –X → (I) |
| AXT | Add to index & test | X +(I) → (I) |
| SXT | Subtract from index & test | –X +(I) → (I) |

Note: See Section 2.3 above for meaning of symbols. Result of these opcodes is stored as a single precision integer in I. These opcodes belong to Class 1.

The number format is used in all accesses, and floating point arithmetic is performed. Integer storage of the index is performed by the hardware. The number stored will be truncated [on both ends] to 9 octals if the computer is in pickapoint mode [i.e., pickapoint flip-flop on], or 7 octals if it is in floating point mode. The location 0 can be addressed with the index opcodes. Index opcodes do not disturb the Accumulator.

Index-and-test opcodes permit branching of the program depending on the contents of the Index register at the conclusion of the index command. If the index is non-zero, the next command is taken in sequence, from which point a transfer can be written returning to the beginning of the loop. If the index is zero, one command word is skipped, i.e., (NC) + 1 → (NC).

**2.3-7 Control Operations** The control opcodes are shown in Table 2.3-12.

**TABLE 2.3–12 TRANSFER OF CONTROL**

| OPCODE | NAME | OPERATION |
|---|---|---|
| TRA | Transfer [to location X] | $_{14}[X]_0 \rightarrow (NC)$ |
| TRE | Transfer and enable interrupts | $_{14}[X]_0 \rightarrow (NC)$<br>See Text |
| SKP | Skip [X words] | $_{14}[X + (NC)]_0 \rightarrow (NC)$ |
| TRM | Transfer [to location X + 1] and mark [in location X] | $(NC) \rightarrow (X)$<br>$_{14}[X + 1]_0 \rightarrow (NC)$ |

In transfers and skips, TRA, TRE, and SKP, control is transferred to the indicated location. Opcode TRE permits one command to be executed and then turns on the master interrupt control flip-flop, CE0, which enables interrupts. This feature facilitates executing a program one command at a time. However, a TRE which transfers control to itself will hang up the machine.

All memory accesses for control commands are in number format. The operand designator, X, is shifted to zero exponent before performing the arithmetic, if any, and the result is truncated [at both ends] to 5 octals before the transfer of control. In coding, bear in mind that control opcodes except SKP belong to Class 2 [see Section 2.2-1]. Both positive and negative operands are permitted in a skip command, but the final address must be positive and must not cause a round-off due to overflow. If either of these conditions is violated, there will be an interrupt. A note of caution: command SKP advances the command word counter by X; thus, it skips X words in the object program, not X words in the source program.

The transfer and mark command, TRM, places the address of the next command in location X and transfers control to location X + 1. The same comments as on TRA, TRE, and SKP above apply to address computation.

**2.3-8 Input/Output Operations** Input/output opcodes of the G-20 Central Processor are of two varieties, single character transmission codes, and block operation codes. Table 2.3–13 indicates the action of the single character codes. Each of these causes a single character to be transmitted on the communication line, flagged as data or command [see Figure 3.4-1]. The flag is considered as the most significant bit of the resulting 9-bit character. The flag bit is "1" for a data character. These are Class 1 opcodes.

**TABLE 2.3–13 SINGLE CHARACTER OUTPUT**

| OPCODE | NAME | OPERATION |
|---|---|---|
| TLC | Transmit Line Command | $_7[X]_0 \rightarrow$ Command Character |
| TDC | Transmit Data Character | $_7[X]_0 \rightarrow$ Data Character |

Standard addressing and indexing, and number format access are used with these commands. The number X is reduced to zero exponent and truncated at both ends so as to save the least significant 8 bits to be sent on the line. The hardware supplies the flag bit and the parity bit to complete the character [see also Section 3.3-2].

The G–20 possesses 6 block input/output opcodes which permit transmission of data, transmission of commands, and reception of data in either 6-bit or 8-bit characters. Packing and unpacking are accomplished automatically by the hardware. One G–20 word comprises either one 8-bit and four 6-bit characters, or four 8-bit characters. Coding is as follows:

BTD8 A + (I), B

which states: transmit as data in 8-bit characters, the B words beginning with location A + (I). A block command occupies two words in core and this should be allowed for in calculating memory space. The format of a block input/output command is shown in Figure 2.3–4. Table 2.3–14 lists the operation codes.

The block input/output codes use standard addressing and indexing but are Class 2 opcodes [see Section 2.2–1]. Accesses are in logic format and words can be unpacked [and packed] as 8-bit or 6-bit characters.

A zero block length is forbidden. See Section 3.3–3.

## TABLE 2.3–14 BLOCK INPUT/OUTPUT

| OPCODE | NAME | OPERATION |
|---|---|---|
| BTD8<br>[OP $=100_2$] | Block transmit data<br>8 –bit characters | send<br>$31^{(L)}24;\ 23^{(L)}16;\ 15^{(L)}8;\ 7^{(L)}0$<br>$31^{(L+1)}{}_{24}$; etc. Data Flag = 1 |
| BTC8<br>[OP $= 110_2$] | Block transmit commands<br>8 –bit characters | send<br>$31^{(L)}24;\ 23^{(L)}16;\ 15^{(L)}8;\ 7^{(L)}0$<br>$31^{(L+1)}{}_{24}$; etc. Data Flag = 0 |
| BRD8<br>[OP $= 101_2$] | Block receive data<br>8 –bit characters | replace<br>$31^{(L)}24;\ 23^{(L)}16;\ 15^{(L)}8;\ 7^{(L)}0$<br>$31^{(L+1)}{}_{24}$; etc. |
| BTD6<br>[OP $= 000_2$] | Block transmit data<br>6 –bit characters | send<br>$31^{(L)}24;\ 23^{(L)}18;\ 17^{(L)}12;\ 11^{(L)}6;\ 5^{(L)}0$<br>$31^{(L+1)}{}_{24}$; etc. Data Flag = 1 |
| BTC6<br>[OP $= 010_2$] | Block transmit commands<br>6 –bit characters | send<br>$31^{(L)}24;\ 23^{(L)}18;\ 17^{(L)}12;\ 11^{(L)}6;\ 5^{(L)}0$<br>$31^{(L+1)}{}_{24}$; etc. Data Flag = 0 |
| BRD6<br>[OP $= 001_2$] | Block receive data<br>6 –bit characters | replace<br>$31^{(L)}24;\ 23^{(L)}18;\ 17^{(L)}12;\ 11^{(L)}6;\ 5^{(L)}0$<br>$31^{(L+1)}{}_{24}$; etc. |

Note: The normal mode of operation of the G–20 hardware is the 8–bit mode, and only in the 8–bit mode is it possible to make use of the full line command list when sending commands in blocks. The 6–bit mode is included because it permits some reduction in alphanumeric data space, if alphanumeric data are limited to a 63–character alphabet. Standard scientific programs use 8–bit mode exclusively. The starting location is determined from X, the operand designator. The operands $L_i$ are accessed as logic words. For location of the OP bits see Figure 2.3–4.

## FIGURE 2.3–4 FORMAT OF BLOCK INPUT/OUTPUT COMMAND

| 2 | 2 | 7 | 6 | 15 |
|---|---|---|---|---|
| F | M | 033 | INDEX | BASE ADDRESS |

FIRST WORD

| 4 | 3 | 2 | 8 | 15 |
|---|---|---|---|---|
|  | OP |  | CCC | BLOCK LENGTH |

SECOND WORD

Note: The part of the second word marked CCC becomes the first line signal. In BTC8 and BTC6 it can be any line command. In all other block commands it must be SDT [$010_8$], and becomes the signal which starts data transmission. For OP see Table 2.3–14.

## FIGURE 2.3–5 LAYOUT OF BUS REGISTERS

CONTROL/ENABLE
CE

| BIT | CE |
|---|---|
| 14 | Command flag 31 |
| 13 | Command flag 30 |
| 12 | Logic flag 31 |
| 11 | Logic flag 30 |
| 10 | Data flag 31 |
| 9 | Data flag 30 |
| 8 | Pickapoint |
| 7 | Clock |
| 6 | — — — — — — — — |
| 5 | Tone signal |
| 4 | Send interrupt |
| 3 | Send interrupt |
| 2 | — — — — — — — |
| 1 | — — — — — — — |
| 0 | Master interrupt control |

INTERRUPT REQUEST
IR

| BIT | IR |
|---|---|
| 14 | Command flag 31 |
| 13 | Command flag 30 |
| 12 | Logic flag 31 |
| 11 | Logic flag 30 |
| 10 | Data flag 31 |
| 9 | Data flag 30 |
| 8 | Exponent overflow |
| 7 | Clock |
| 6 | — — — — — — — |
| 5 | — — — — — — — |
| 4 | Receive interrupt |
| 3 | Receive interrupt |
| 2 | Receive interrupt |
| 1 | Receive interrupt |
| 0 | Illegal command |

LINE RESPONSE
LRE

| BIT | LRE |
|---|---|
| 14 | ↑ Incomplete Word Code, bits 14-12 ↓ |
| 13 | |
| 12 | |
| 11 | No response |
| 10 | "REQ" response |
| 9 | Parity error |
| 8 | Most sig. digit |
| 7 | ↑ |
| 6 | Store response to LC, bits 8-0 |
| 5 | |
| 4 | |
| 3 | |
| 2 | |
| 1 | ↓ |
| 0 | Least sig. digit |

NEXT COMMAND ADDRESS REGISTER
NC

| 15 Bits |
|---|

14 0

PICKAPOINT EXPONENT
REGISTER PE

| 7 Bits |
|---|

6 0

REGISTER CODES
(I FIELD OF COMMAND)

| Alpha | Octal |
|---|---|
| CE | 01 |
| IR | 03 |
| LRE | 02 |
| NC | 00 |
| PE | 04 |

INCOMPLETE WORD
CODE [See Section 3.3–3]

| Characters Remaining | Code |
|---|---|
| 1 | 6 |
| 2 | 2 |
| 3 | 3 |
| 4 | 1 |
| 5 | 0 |

**2.3–9 Bus Register Operations** The hardware registers of the G-20 available to the programmer are the following:

| Alpha | Name | Length |
|---|---|---|
| NC | Next Command word address | 15 bits |
| PE | Pickapoint Exponent | 7 bits |
| CE | Control and Enable | 15 bits |
| IR | Interrupt Request | 15 bits |
| LRE | Line Response | 15 bits |

The register opcodes are listed in Table 2.3-15. The bus registers are addressed using the I field of the command and, hence, these opcodes cannot be indexed.

**TABLE 2.3–15 REGISTER OPERATIONS**

| OPCODE | NAME | OPERATION |
|---|---|---|
| LDR | Load Register: | |
| | CE, IR, LRE, | $_{14}X_0 \rightarrow$ (Reg. I) |
| | PE | $_6X_0 \rightarrow$ (Reg. I) |
| EXR | Extract Register I into itself | $_{14}[(\text{Reg. I}) \wedge X]_0 \rightarrow$ (Reg. I) |
| ERO | Extract Register I into OA | $_{14}[(\text{Reg. I}) \wedge X]_0 \rightarrow$ ( OA ) |
| ERA | Extract Register I into Acc | $_{14}[(\text{Reg. I}) \wedge X]_0 \rightarrow$ (Acc) |

All four register codes work for CE, IR, LRE. Register PE can be loaded only, using LDR. In this operation, bit 6 of X becomes the sign of PE and bits 5-0 become the magnitude of PE. A negative zero in PE will cause errors and is therefore forbidden.

Register NC can be read only, using ERO or ERA. [The control commands give methods for loading and incrementing NC.] In a register command the index address specifies the particular register and the base address specifies the number or extractor to be used. Number format is used for all accesses, and the number X is shifted to zero exponent and truncated before the operation. Register codes are Class 1.

The layout of registers in the G-20 is shown in Figure 2.3-5.

PICKAPOINT FLIP-FLOP. Note that the pickapoint flip-flop is contained in register CE position 8. When this flip-flop is set, the computer is in the pickapoint mode for storage and access.

TONE SIGNAL. A tone signal is provided, controlled by CE5. When this bit is loaded, the tone is turned on, and when the bit is cleared, the tone goes off.

## SECTION 2.4 INTERRUPTS

There are two classes of interrupt requests in the G–20: automatic interrupt requests and enabled interrupt requests. Automatic interrupt requests can be generated internally by an illegal opcode or address, or by exponent overflow. Automatic interrupt requests are also generated externally by signals from the peripheral equipment. Enabled interrupt requests can originate only internally from flags in command, logic, or data words, or from the 1–second real time clock. As the name implies, these interrupt requests can occur only if they are enabled by the setting of certain flip-flops.

It is important to distinguish between interrupt requests and actual interrupts. An actual interrupt cannot occur unless the master interrupt control is on. An interrupt request once generated is stored and can cause an interrupt at any future time that the master interrupt control is turned on, unless the request register has deliberately been cleared by the program before the master interrupt control is turned on.

Master interrupt control is bit 0 of register CE. [See Figure 2.3–5.] All interrupt requests are stored in register IR. Automatic interrupt requests appear in the locations marked Exponent Overflow IR8, Illegal Command IR0, and Receive Interrupt IR1–4. After being set by the occurrence of a fault or of a line interrupt signal, these flip-flops remain set until cleared by a register command.

Bits 7 and 9–14 of register IR store enabled interrupt requests. None of these bits can be set by a clock pulse or flag unless its enabling flip-flop in register CE has first been turned on by command. These flip-flops remain set until reset by command.

When an interrupt of either type occurs, the program in progress is halted at the completion of the current command. [However, see also Section 2.4–3.] The hardware causes automatic interrupt mark transfer to core location 64; that is, CE0 is turned off, the contents of the Next Command word counter are stored in location 64, and control is transferred to location 65. Note that when an interrupt occurs on a transfer of control command, the address stored in 64 is the one to which control was to be transferred. There is normally in 65 and subsequent locations a routine known as the Interrupt Service Routine, or ISR whose task it is to "secure" the current program, storing any pertinent data, and then "service" the interrupt.

Servicing the interrupt consists of the determination of which bit [or bits] is set in IR. This in turn is followed for an internal interrupt by exit to an appropriate subroutine to take whatever action the programmer has called for in the event of that particular interrupt.

An external interrupt will result from one [or more] of the Receive Interrupt bits being set. Each of these signifies an interrupt request originating on one of the four interrupt lines. The ISR will determine the particular unit which interrupted by calling the units known to be on that line and asking each if it interrupted. External interrupts are taken up in more detail elsewhere in this manual.

**2.4–1 Fault Interrupts** Any negative address, or any address greater than the maximum number of storage locations available in the particular installation will cause an automatic interrupt request. If during the computation of the final address in an SKP command there is a carry-out calling for a roundoff, an automatic interrupt request will also be generated. A non-existent opcode or an illegal register address, i.e., an address incompatible with the opcode, or a bus register address greater than 4, will also cause an automatic interrupt request. Any of these faults will also set IR0.

If at the conclusion of the command there has been an exponent overflow, an automatic interrupt request is generated. An interrupt request is also generated if one tries to store an otherwise legal number with a pickapoint exponent that is too small and thus causes mantissa overflow. These faults will set IR8.

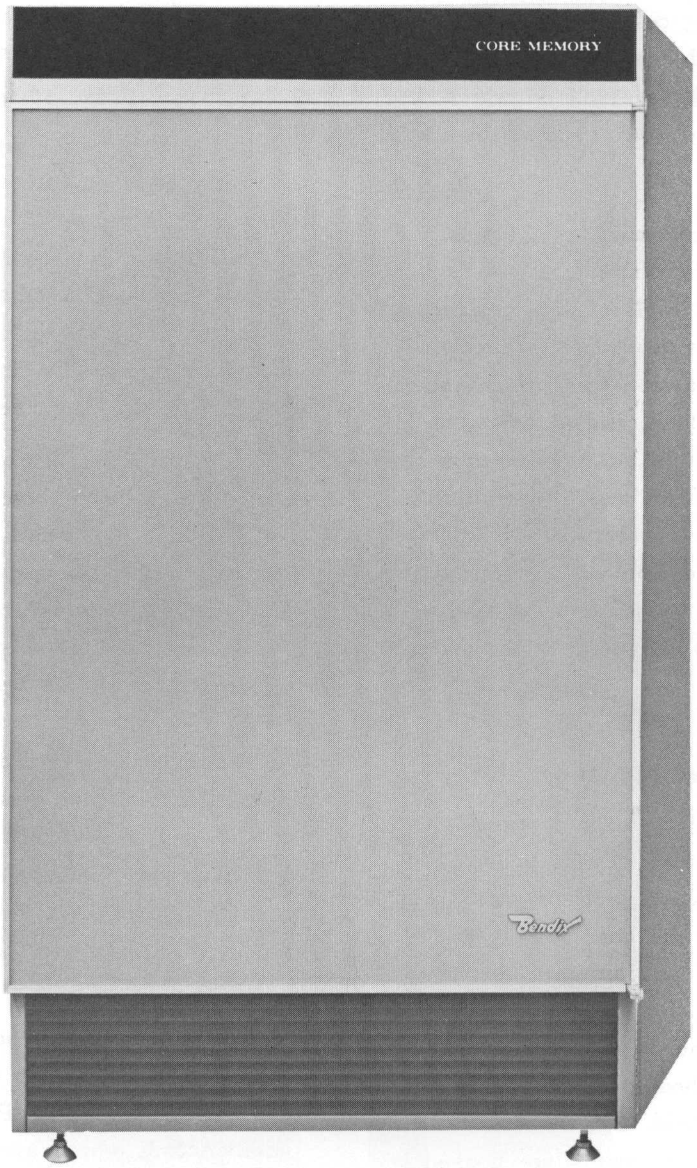
Memory parity failure will unconditionally halt the G–20 and cause a parity failure signal to be displayed. Since memory parity is odd, a completely cleared location will show incorrect parity. A program to write zeros in the memory following an unexpected shut down will obviate this problem.

**2.4–2 Enabled Interrupts** The interrupts which require both the setting of an enable flip-flop and the occurrence of a particular interrupt condition before the computer will be interrupted are flag interrupts and real time clock interrupts. In order to cause the computer to halt at particular words, two independent flags are provided in bits 31 and 30. When a word is picked up during computation, the computer decides from the context whether it is a command, data, or logic word. It then checks whether the word has a flag, and if it does, whether the corresponding enabling flip-flop in register CE is set. If so, the appropriate bit in IR is set and computation is interrupted at the conclusion of the opcode if CE0 was set. [However, see also Section 2.4–3.] The same applies to interrupts from the real time clock, which is actually a 1–second timer slaved to the power line. If its interrupts are appropriately recorded, this timer can serve as an interval timer, or as an actual clock after being set to the correct time.

**2.4–3 Interrupt Processing** After any illegal command word or overflow there will be an automatic interrupt request. Except in repeat commands, the other types of interrupt requests can occur only at the conclusion of the command, that is, after OA has been cleared. An OA or an ERO command is not a completed command for interrupt purposes. The first word of a two-word repeat opcode or a two-word input/output opcode is not treated as a completed command for interrupt purposes. Where computation is resumed after an interrupt, at least one command in the main program will ordinarily be executed before another interruption is permitted, even though not all of the interrupt requests have been cleared. An interrupt on a TRE command is inhibited until the completion of the following command. See Section 2.3–7.

Repeat commands are handled somewhat differently from all other commands from the point of view of

FIGURE 2.5–1 AUXILIARY CORE MEMORY, MM–10



terrupts because an enabled flag encountered in the operands [or in the assembly of X] will terminate the sequence. Data and logic interrupt request indicators must therefore be off before a repeat command is executed. Likewise, the illegal address indicator must be off. An overflow occurring during the command will cause an automatic interrupt request and termination of the sequence.

When a repeated arithmetic or logic command is terminated, by an enabled flag or otherwise, there remains in the Accumulator what was in it at the completion of processing of the operand on which the sequence was interrupted, i.e., the result of the operation thus far.

When a repeated test is terminated, an address one greater than that at which the test halted will be left in the Accumulator. The NC register will then be incremented if termination was caused by failure of the test, or left alone if termination was for any other cause, e.g., flag, end of block, illegal command. If the master interrupt control is on, the interrupt register will then be examined and if an interrupt request exists, the current contents of NC will be marked in location 64 and control transferred to location 65. If no interrupt exists, computation will be resumed at the location indicated by the current value of (NC).

## SECTION 2.5 CORE MEMORY, MM–10

A minimum G–20 Central Processor contains 4,096 words of core memory. Memory may be added in increments of 4,096 words, up to a total capacity of 32,768 words. While all of the core memory is conceptually and electrically an integral part of the Central Processor, not all of this memory is in the main cabinet containing the logic. Systems having 4,096 or 8,192 words can house all core memory in the Central Processor cabinet. For systems with a larger memory, 4,096 or 8,192 words are housed in the Central Processor cabinet, and the balance in external cabinets of 4,096 words, MM–10 (A) or 8,192 words, MM–10 (B). External memory is limited to a total of 3 cabinets in any combination. Each word in an external Core Memory module is directly addressable, the same as words in internal memory. A complete memory cycle for a word in external memory is 6 clock cycles, the same as for a word in the internal memory. There is no distinction in coding for use of external memory locations.

A cabinet 64 inches high, 36 inches wide, and 28 inches deep, as shown in Figures 2.5–1 and 2.5–2 will house an external Core Memory module of 4,096 or 8,192 words. The weights and power drains are shown below.

| | Approx. Weight | Approx. Power |
|---|---|---|
| MM–10 (A) 4,096 words | 1,265 lb. | 1.0 KVA |
| MM–10 (B) 8,192 words | 1,300 lb. | 1.5 KVA |

Power required is 115-volt, 60-cycle, single phase.

### 2.5–1 Comparison of Internal and External Memory

As stated above, there need be no distinction between internal and external memory from the standpoint of Central Processor coding. However, from the standpoint of total execution time and from the standpoint of programming for input/output simultaneous with computing [using the Data Communicator] there are two significant differences.

While the complete memory cycle from start of access to start of next access is 6 microseconds for both internal and external memory, the time from the start of access to *availability* of the word is different. This time is 3 microseconds for internal memory and 4 microseconds for external memory. Any access to external memory ties up the external memory communication lines for 2 microseconds. Following any access to external memory by one control device, another control device can start an access to another cabinet in 2 microseconds. When a Central Processor has *written* in external memory, it can start an access in another cabinet in 3 microseconds. For all other situations the effective memory cycle is the basic 6 microseconds.

The differences mentioned above result from the following facts: Transfer of a word from an external cabinet to the Central Processor requires about one microsecond, which accounts for the difference in availability. There is no difference in cycle time because an external module has its own Memory Buffer register, and can start restoring the word before it has reached the Central Processor. For *reading* from either internal or external memory the time until another memory cycle can begin is always 6 microseconds.

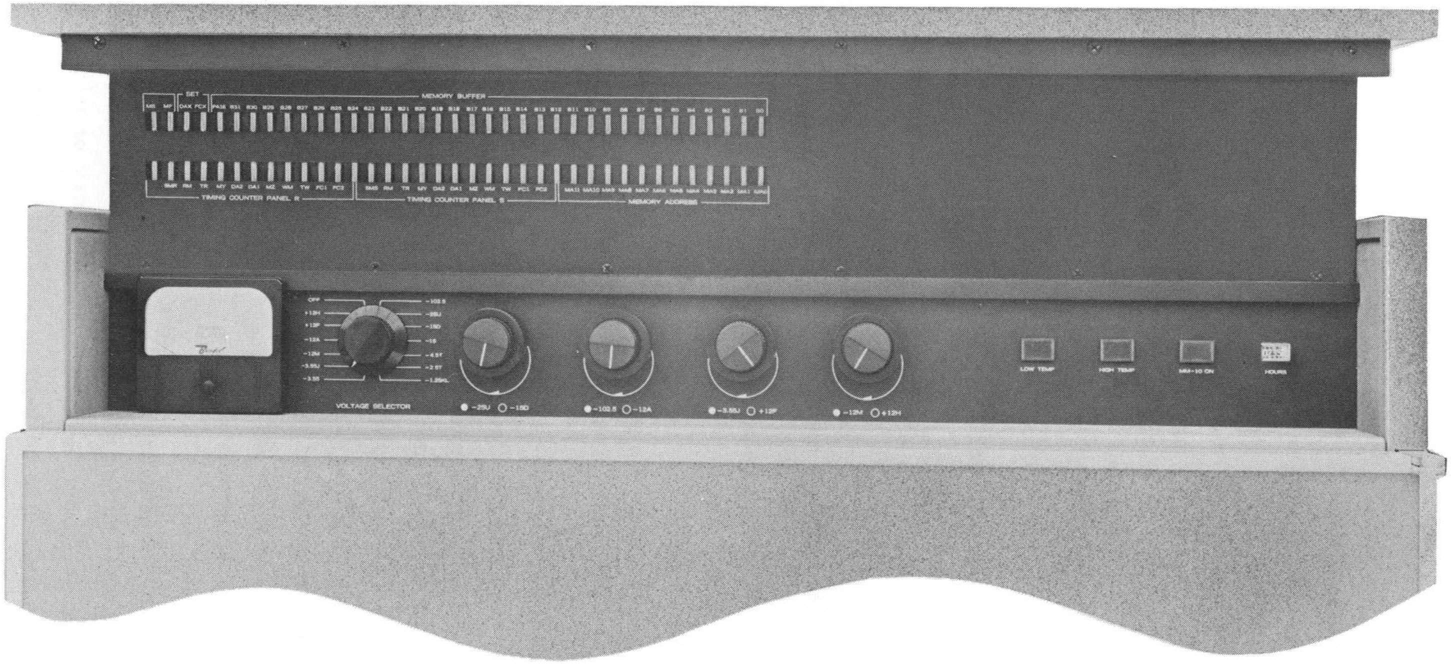


FIGURE 2.5–2 CORE MEMORY CONTROL PANEL

# CHAPTER 3

# THE G-20 COMMUNICATION SYSTEM

Much of the flexibility of the G–20 stems from its communication system. All units in the system communicate in a common language of 10-bit characters over a "party line". The device controlling a data transfer need not be the transmitter or receiver of the data. Data transfers can be of three varieties: direct, meaning that the core memory either furnishes or receives the data; buffered, meaning that the G–20 establishes the communication but the data transfer is between a buffer and some other accessory; and off-line, which means that a Control Buffer entirely disconnected from the system establishes and participates in a data transfer with another accessory. Direct data transfers can be handled through the Central Processor hardware or through the DC–11. For an off-line operation, the instructions to be loaded into the buffer and executed must previously have been recorded on magnetic tape, paper tape, or cards.

## SECTION 3.1 SYSTEM ORGANIZATION AND HARDWARE

All devices in a G–20 System [with the exception of auxiliary Core Memory] are designed to operate on a standard form of communication line and to use a common language. There can be more than one line in a system and command-controlled switches are provided to transfer devices from one line to another. To fix ideas, Figure 3.1–1 shows a simple G–20 System with a single line to which all devices are permanently connected. Figure 3.1–2 shows a medium system with 16,000 words and a DC–11. Figure 3.1–3 shows a large system. The system of Figure 3.1–1 must time-share in order to carry out more than one task at a time. The system of Figure 3.3–2, however, can carry out several tasks simultaneously. The system of Figure 3.1–3 can be operated as one large system or several small systems, thus affording great flexibility.

**3.1–1 Communication Hardware** The communication trunk such as shown as a single heavy line on Figure 3.1–1 consists of 24 identical circuits of 100-ohm impedance in two cables distributed as follows:

| | | |
|---|---|---|
| Data lines | 10 | Main Cable |
| Request line | 1 | |
| Spare | 1 | |
| Interrupt lines | 6 | Auxiliary Cable |
| System on/off | 1 | |
| Spare | 5 | |

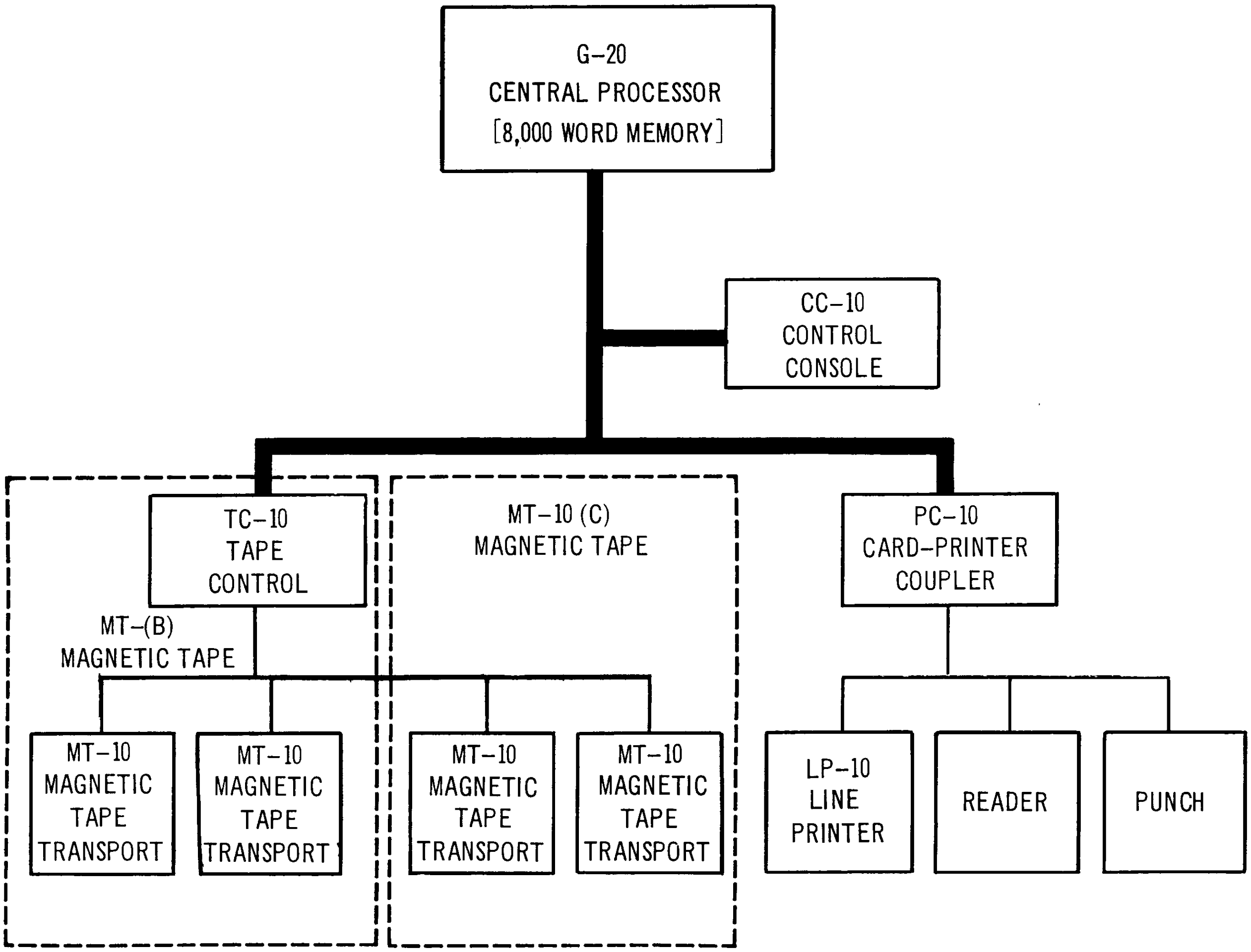
For pin assignments, see Appendix V.

A communication line may be up to 1,700 feet long less 35 feet for each line coupler. Line couplers may not exceed 38 and must be spaced at least 10 feet. There are some additional restrictions for high speed Magnetic Tape, MT–10.

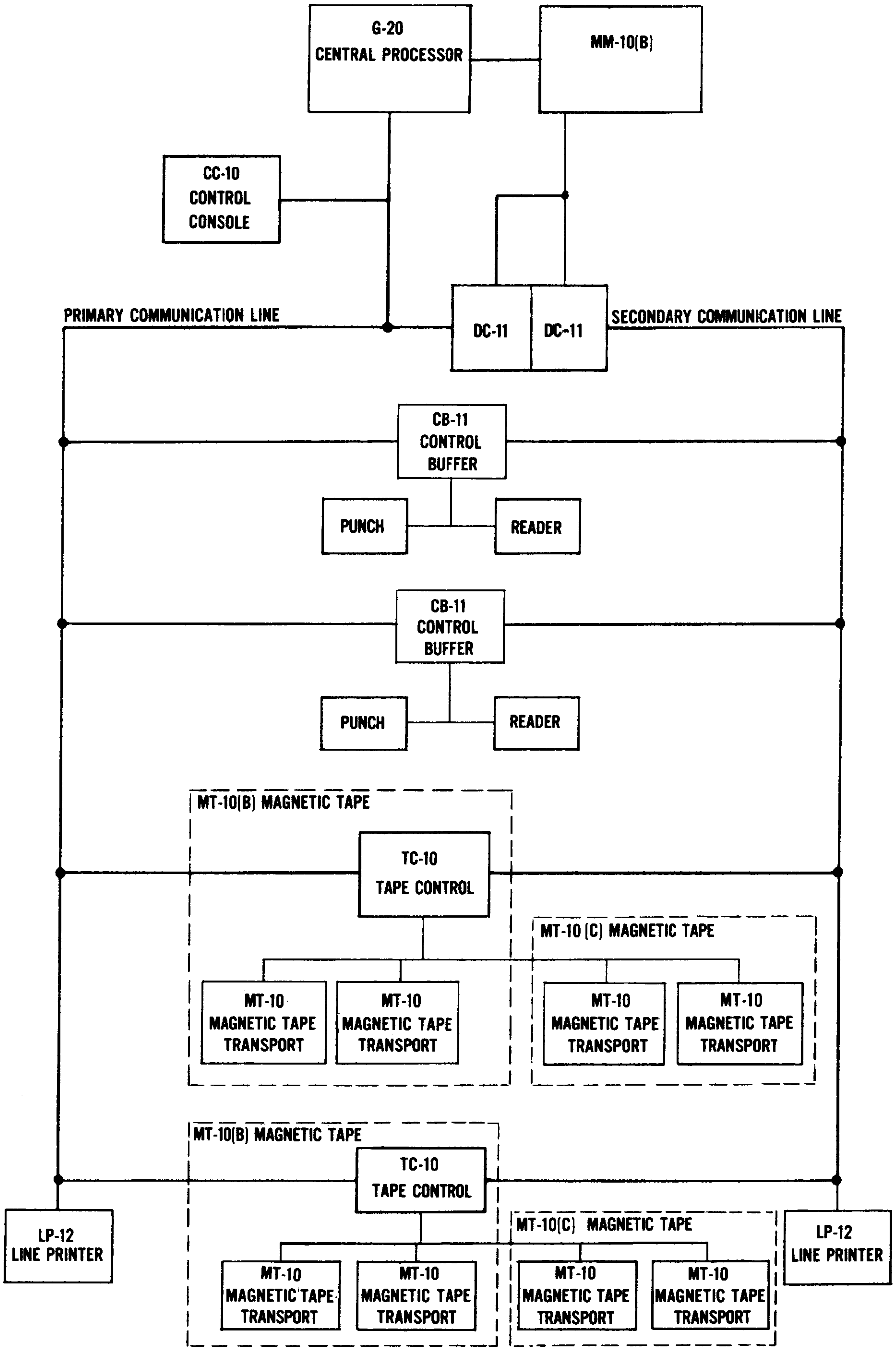
System on/off is a manually applied 24-volt DC signal of positive or negative polarity for setting or resetting locking relays, but all other intelligence carried is in the form of 8-volt pulses of fractional microsecond duration. The data lines carry 10-bit characters consisting of 8 information bits, a data flag, and an even parity bit. In expressing a character in octal notation the flag is taken to be the most significant bit. Command characters are 001-377. Data characters are 400-777. The interrupt lines carry interrupt pulses from accessories to the G-20 or DC-11 or between G-20's and DC-11's, and the request line is used by any receiving device to request its next character from the transmitting device.

Because the various units differ in internal circuit details, line couplers provide matching to the communication line. The line switching function, when

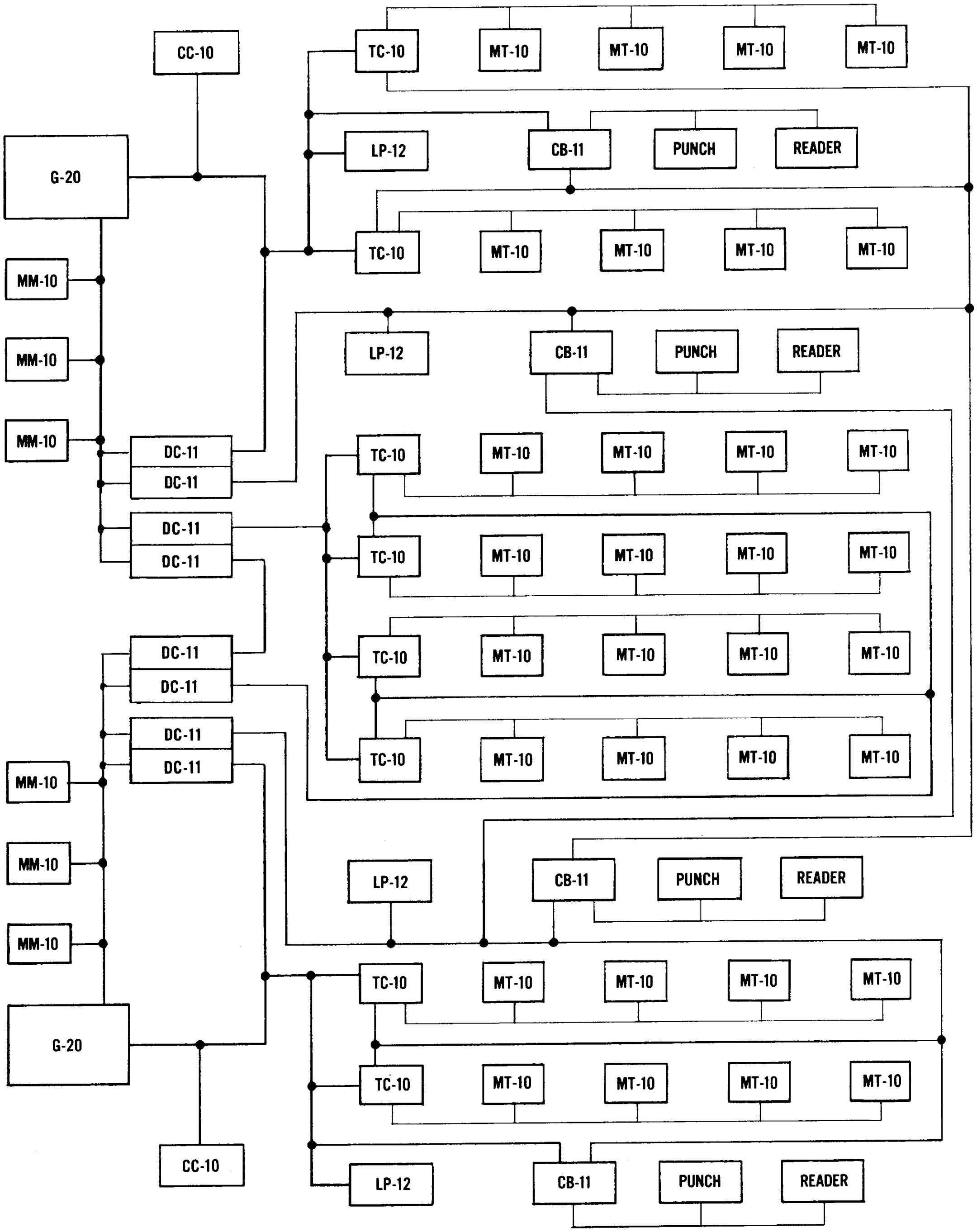
FIGURE 3.1-1 A BASIC G-20 SYSTEM



## FIGURE 3.1–2 A MEDIUM SIZE G–20 SYSTEM



## FIGURE 3.1–3 A LARGE SIZE G–20 SYSTEM



provided, is built into the line coupler. All wires of the Main Cable are switched. In the Auxiliary Cable only the interrupt line used by the device is switched.

## SECTION 3.2 PRINCIPLES OF OPERATION

The data lines of a communication trunk act as a party line for transmission of data or command characters either singly or in blocks. Therefore, it is a rule that only one conversation is permitted on the line. Each transmission of a character is followed by an echo, in the form of a reply, or a request for next character. This serves to confirm that information has been received and serves also to limit the data rate to what the slower device can handle.

Each character on the data lines carries an even parity bit which is used to verify that there has not been a dropped bit. The character consisting of all zeros is not used.

Signals on the data lines must conform to certain rules to avoid loss of data. The interval between two successive characters must be 2.5 microseconds or greater; a character on the data line must not follow a request on the REQ line more closely than .7 microsecond; a request on the REQ line must not follow a character on the data line more closely than .9 microsecond.

Interrupts are not required to conform to any rules. The simultaneous occurrence of two or more interrupts on one line causes no difficulty as the source of an interrupt is always determined by interrogating all units capable of interrupting.

The data lines are used for all transmission of data and command characters between units, both in single characters and in blocks.

## SECTION 3.3 OPERATION

The operation of any line in the Communication System is always under the control of a device capable of storing and executing a program and of issuing commands to the participating units. The control device may also participate in the transfer of information as well as controlling. Units which can be controlling devices are Central Processors, DC–11's and Control Buffers.

In order to initiate any action, the controlling device calls the unit which is to take the action, using a unique command signal which is the call number for that unit. The unit answers if it is on-line and places itself in readiness to answer queries and receive commands. If the action to be taken involves only the unit being instructed [e.g., search magnetic tape] the action begins as soon as the control device has completed the instructions necessary. In the case that two units are involved, the first unit is completely instructed and then the second unit is called and instructed. The controlling unit then gives a start signal, and the action which has been set up is carried out. At the conclusion of a data transfer not involving the controlling device, one of the units involved may signal the controlling device by sending an interrupt. This provides a way for the controlling device to know that the line was free following a data transfer. Any interrogations on the line would interfere with the data transfer. In order to permit a workable system, a number of rules and conventions have been established as outlined in succeeding paragraphs.

**3.3–1 Operating States** To clarify the discussion of the Communication System a number of operating states have been defined covering all those aspects of unit behavior significant to the programmer or operator. Not all units can occupy all of the states listed. For more detail on this, see the discussion under each individual accessory.

The discussion below treats the signals directed to accessories and the replies they give. Actually, many accessories do not communicate directly, but through a control unit of some sort. When a tape unit is called, it is actually the tape control unit which answers. From the programming standpoint these answers can be taken to have been given by the unit itself.

In order to make clear the discussion of states below, several names of command signals have been introduced even though the line vocabulary has not been taken up.

| | | |
|---|---|---|
| GRN | = | Green, program continues |
| RED | = | Program branches |
| SDT | = | Start data transfer |

OUT = Go out of service
END = End of block
ERR = Error

OUT OF SERVICE. When OUT of Service a unit is effectively disconnected, and can only be put back on-line by manual operation of the ON-LINE button. Such action causes the unit to go into the STANDBY state. Any unit may be called and directed by command to go OUT of Service.

STANDBY. A unit in the STANDBY state examines and rejects call signals until it hears its own unique call with correct parity, whereupon it answers GRN and enters the CALLED state.

CALLED. A unit in the CALLED state is able to answer all queries and execute all commands meaningful to it in this state. Undefined characters, or characters with incorrect parity are usually ignored.

INSTRUCTED. A unit is in the INSTRUCTED state when it has been given the complete set of instructions required to establish a data transfer but has not yet been told by the control unit to start data transfer. A unit which has been set up to receive will, upon hearing SDT, enter the MESSAGE state and send REQ. A unit which has been set up to transmit will, upon hearing SDT, enter the MESSAGE state but give no response to the SDT. A unit in the INSTRUCTED state upon hearing its own call will answer GRN and return to CALLED. No commands except SDT and its own call are meaningful to a unit in the INSTRUCTED state.

BUSY. When a unit is executing an operation other than a block data transfer, or is interlocked, it is said to be BUSY. BUSY has two substates. The first is BUSY-ALERT in which the unit will answer queries but will not execute commands except OUT. When a unit goes to a BUSY condition it usually enters BUSY-ALERT. Upon hearing a call directed to another unit it will go to BUSY-QUIET without answering. In BUSY-QUIET it can hear only its own call, which it will answer and go to BUSY-ALERT. Upon completing an operation, a BUSY-QUIET unit goes to STANDBY and may send an interrupt, and a BUSY-ALERT unit goes to CALLED and may send an interrupt.

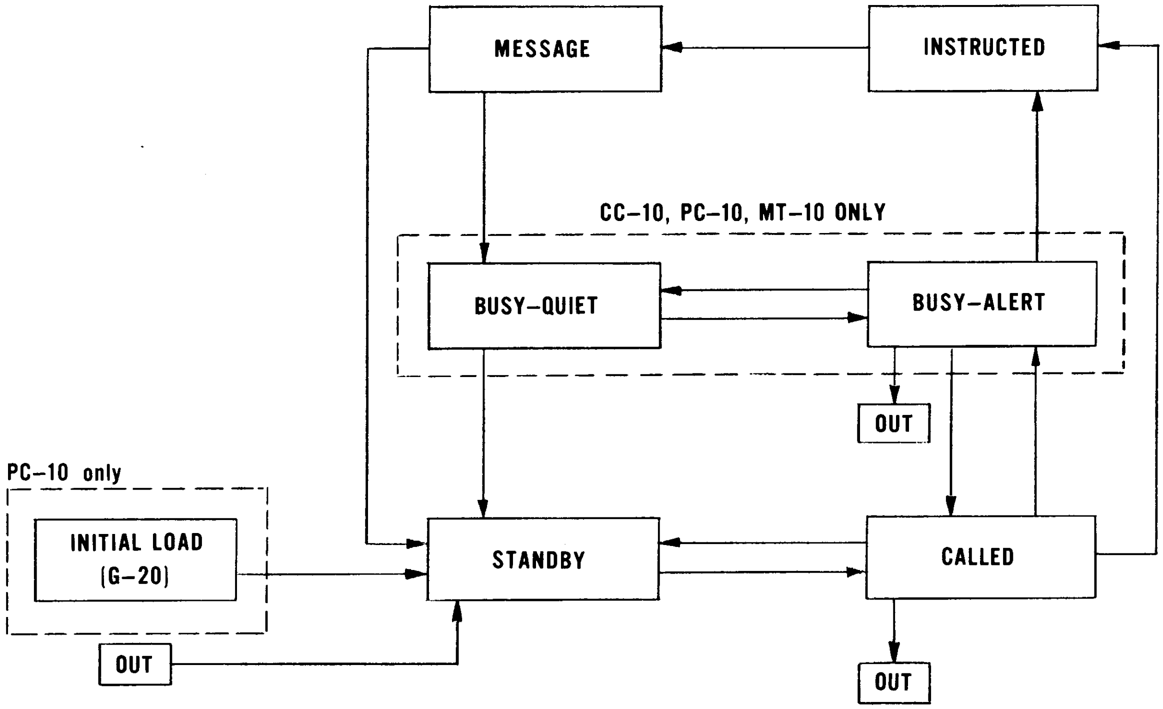
MESSAGE. A unit engaged in block communication is said to be in the MESSAGE state. All other units are forbidden to use the data lines during this time. When the message has been completed, the transmitter sends END or ERR and returns to STANDBY. The END or ERR signal returns the receiver to STANDBY. If the G–20 is the receiver, it may terminate the block by sending END or ERR. At the end of a transfer in which a buffer is in control of the line, the buffer will signal the G–20 with an interrupt to signify that the line is free again. See Figure 3.3–1.

CONTROL STATES. The CB–11 and DC–11 possess certain control states in addition to the above. See the sections dealing with these units.

**3.3–2 Single Character Communication** The single character codes TLC and TDC were mentioned in Section 2.3–8. These codes serve to transmit as a command or data character the least significant eight bits of the word which is the operand of the TLC or TDC code. The usually anticipated replies are GRN, which calls for a continuation of the program, and RED which calls for branching of the program. Therefore, a GRN response will cause the program to continue after skipping one command word. A response other than GRN will be stored in the Line Response register, LRE [see Section 2.3–9], and the next command will be taken in sequence. This command should ordinarily call for a transfer of control. The programmer can determine from interrogating LRE with command ERO or ERA what action he wishes to take. The program will also take the next command in sequence if the response was REQ, or contained a parity error, or if there was no response. In these cases flip-flops LRE–10, LRE–9, or LRE–11 will be set. Again the programmer can decide after examining register LRE the course of action he wishes to take. In the case of no response the computer will wait one second and then set the no response flip-flop, LRE–11, and proceed with the program.

**3.3–3 Block Communication** Block communication can be of three varieties. In all cases, one of the devices is operating at its own speed and the other device is required to keep in step. Only "control" devices possess this capability of keeping in step.

FIGURE 3.3–1 G–20 ACCESSORY OPERATING STATES



Note: Control states of the CB–11 and DC–11 are omitted.

Therefore, a control device [e.g., Central Processor, CB–11, DC–11] must be one of the parties to any block communication.

In block transmission of commands the Central Processor is operating in a block mode, and is frequently sending a group of calls and queries. The device or devices receiving the commands are usually in either STANDBY, CALLED, or BUSY states: The rate at which the block transmission proceeds depends on the rate at which the devices called or interrogated answer the command signals. Technically, there may be many short conversations taking place successively during one block.

The other two block communication situations involve movement of blocks of data between two devices and two only; there is only a single conversation taking place after the start signal has been given. In Transmit Data and Receive Data the characters alternate with REQ signals. A block communication is usually terminated by an END code which is sent by the transmitting device.

TRANSMIT COMMANDS. The commands BTC8 and BTC6 enable the G–20 to instruct a peripheral device completely in a single G–20 command or to send a long list of queries to the units on a line in order to determine which one originated an interrupt signal. They may be used for other purposes where command TLC would take too much time or waste too much core space for its operands. The first line command to be sent, i.e., the call signal to the first unit, is packed in the second word of the command in bits 15–22. The balance of the block of line commands is stored four to a word. The action of BTC8 and BTC6 is similar to that for command TLC with some additions: when a situation occurs which calls for the program to branch, i.e., no response, or parity error, or any response other than GRN, or the block called for would extend beyond the end of memory, the block transmission will terminate. Bits 14–12 of register LRE will contain the "incomplete word code" and bits 8–0 of register LRE will contain the non–GRN response, if any. The incomplete word code is as shown in Figure 2.3–5. If memory overflow was the cause of termination, IR0 will be set. Also, the address of the last word operated on will be saved. The Accumulator will contain an address greater by two than the last word processed. When a block transmission of commands results in GRN replies to each character in the block, computation will be resumed after skipping one command word. When the block is terminated by a non–GRN response, computation will be resumed with the next command, which will ordinarily be a transfer of control.

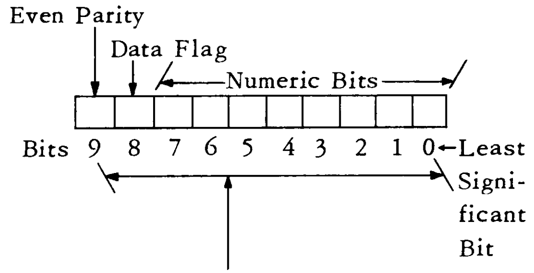
TRANSMIT DATA. In a transmit data operation, BTD8 or BTD6, the G–20 will send SDT and then send characters in response to REQ signals. At the completion of the assigned block, the G–20 will send END and resume computation after skipping one command word. At any time that the response is other than REQ, or is delayed more than one second, or there is a memory overflow, the sequence will be terminated. An ERR code will be transmitted, the appropriate flip-flops will be set indicating the cause of trouble, and the non–GRN response, the word address, and the incomplete word code will be preserved. This information will be exactly as shown under *Transmit Commands.* Computation will be resumed with the next command which will ordinarily be a transfer of control.

RECEIVE DATA. In receiving data, BRD8 or BRD6, the G–20 will send SDT and then will transmit REQ signals and receive data characters, packing them in words in the indicated block. When the required number have been received, the computer will send END and resume computation after skipping one command word. Any response other than a data character will terminate the sequence; a parity error, or memory overflow, or a delay of more than one second will also terminate the sequence. The computer will send ERR and set the appropriate error flip-flops. The Accumulator will contain an address one higher than the last one operated upon. A partially completed word will have its least significant characters filled in with zeros. Computation will be resumed in sequence from which point a transfer can be executed. Register LRE 8–0 will show a non–GRN response.

## SECTION 3.4 LINE COMMUNICATION CODES

Except for interrupt and request signals, all communication between elements of a G–20 System takes place one character at a time using an alphabet of 511 characters, each containing eight numeric bits, a data flag, and an even parity bit. Alphabets of fewer than eight bits are handled by filling in zeros for one or more of the eight information bits. When the G—20 transmits a 6—bit character the hardware supplies two zero bits. If a 5—bit character is received from a nonstandard paper tape, the hardware supplies three zero bits. The bits that are filled in with zeros are those at the most significant end of the character. The location of the data flag bit and parity bit is the same in all characters. The 511 characters of the G—20 communication alphabet are numbered in octal 001 through 777. The character 000 represents the absence of a signal. Figure 3.4—1 summarizes these facts.

**FIGURE 3.4–1 FORMAT OF LINE SIGNAL**



These nine bits are referred to as an octal character of three digits. In 6—bit data transmission bits 6 and 7 may be non-zero only in the first character of a word.

Henceforth, we shall ignore the parity bit and discuss only bits 8—0 which we shall refer to as an octal number of three digits. Table 3.4—1 indicates the manner in which the allowable 511 characters are assigned by general categories. Excluding calling signals and command numerics, there is a maximum of 63 command characters available for instructing or querying an accessory device.

Since only one device is instructed at a time in the G—20 System, it is perfectly possible and feasible for a character to have different meanings to different devices. This is actually necessary in some cases because there are more than 63 meaningful queries, actions, and responses which can be given by units in the system. Accordingly, a certain octal command character may have several "names" or mnemonic designations, depending on the accessory to which the command is to be directed. Appendix III lists the octal line signals and the alpha designations which have been attached to them for various accessories.

**TABLE 3.4–1**
**CLASSIFICATION OF LINE SIGNALS**

| OCTAL NUMBERS | CLASSIFICATION |
|---|---|
| 001—057 | General commands |
| 060—077 | Query commands |
| 100—177 | Command numerics |
| 200—377 | Calling signals |
| 400—777 | Data characters |

# CHAPTER 4
# THE G-20 ACCESSORY EQUIPMENT

The accessory equipment available for use in a G-20 System includes:

Control Console, CC-10
Paper Tape Station, PT-10
Magnetic Tape, MT-10
Card-Printer Coupler, PC-10
Printers LP-10, LP-11
[unbuffered]
Control Buffer, CB-11
Printer, LP-12
[buffered]

The auxiliary Core Memory Module, MM-10 was described in Chapter 2, and the Data Communicator, DC-11, is described in Chapter 5.

Control of the G-20 is exercised through the Control Console, CC-10, which is an input/output typewriter with added logic and controls. There can be several consoles in a G-20 System and they can be located remote from the Central Processor.

For those users having special requirements, the Paper Tape Station, PT-10, has been designed for maximum flexibility in adapting to existing paper tape formats. By program control it can read or punch any 5, 6, 7, or 8-level paper tape code.

High speed, high density Magnetic Tapes are provided, two transports to a cabinet and four transports to one tape control unit.

Card and printer operations can be handled by the Card-Printer Coupler, PC-10, which couples standard card-handling equipment or Bendix Line Printers, LP-10 and LP-11 to the G-20 System.

The Control Buffer, CB-11, can also couple to card or printer equipment and contains, in addition, a 4,096 character core memory and some data-editing ability. The CB-11 can handle off-line data manipulations.

The completely buffered printer, LP-12, requires no coupling device and operates directly on a communication line.

The principal limitations on the maximum number of accessories in a G-20 System are the number of available addresses for accessories, namely, 128, and the line loading caused by line couplers which connect the accessories to the communication line. Each accessory or control or coupling unit connected to the communication line requires one line coupler. The number of line couplers on one communication line is never allowed to exceed 38 because of loading considerations.

## SECTION 4.1–CONTROL CONSOLE, CC-10

The Control Console, CC-10, is intended primarily as a control and monitor device, although it can also be used as an input/output device. It consists of an electric typewriter with associated logic, several control switches, a set of six program-controlled lights, and a tone speaker. Thus, the Control Console provides for the following functions:

Control of the G-20 System
Input of messages, including printed record
Output of messages
Display of visual information on program lights
Production of audible signals for warning and monitor purposes

The CC-10 is housed in a desk-type cabinet, 64 inches wide, 28 inches deep, and 30 inches high. It draws 0.6 KVA at 115 volts, 60 cycles. The console weighs 500 pounds. A general view of the CC-10 appears in Figure 4.1-1.

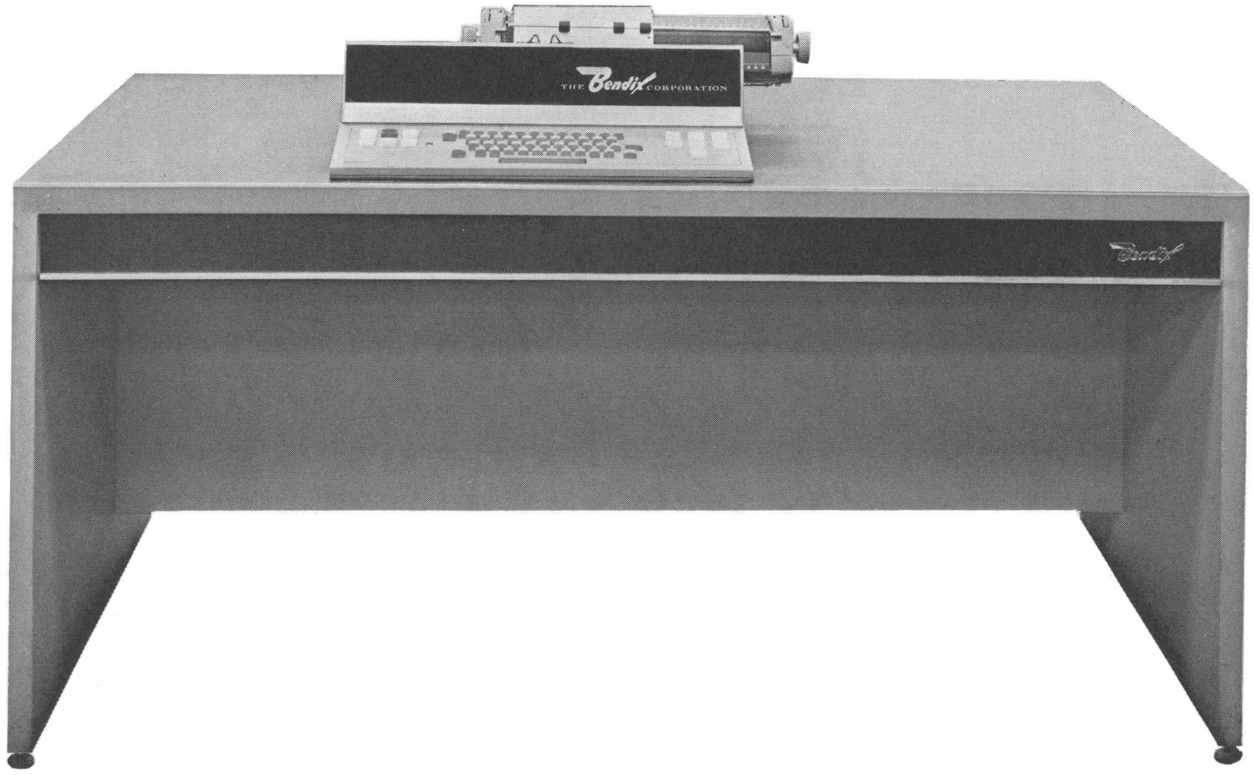


FIGURE 4.1–1 CONTROL CONSOLE, CC–10

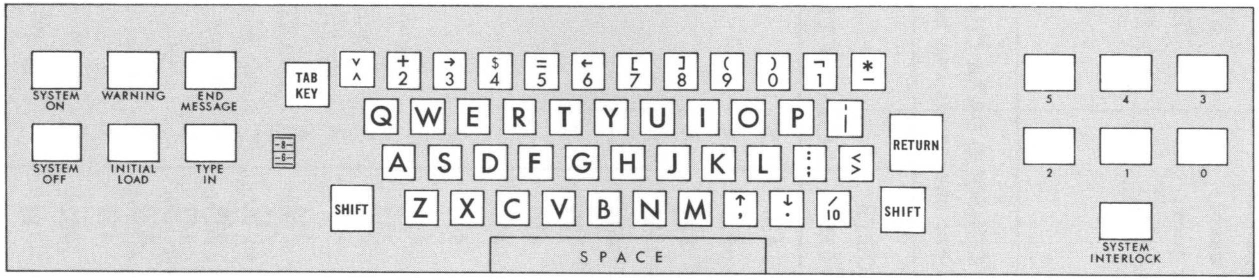


FIGURE 4.1–2 CONTROL CONSOLE KEYBOARD

The CC-10 typewriter sends and receives data in the in the single character mode only, and interrupts the G-20 program briefly for the transmission or reception of the electrical signal corresponding to each character. The CC-10 can also answer calls and queries coming from the G-20 in single character form or block form. Figure 4.1-2 shows the typewriter keyboard, which prints 88 characters including upper and lower case letters, figures, and special symbols [see Table 4.1-1]. This alphabet includes a usable 62-character subset of ALGOL, the same set provided on the Line Printers LP-10 and LP-11, but lacking the symbol "≠". [The combination ¬ = can be substituted.]

**4.1–1 Hardware of the CC–10** The following controls are provided for operating the G–20 System from the console station. See Figure 4.1–2 which shows the control panel.

SYSTEM ON
SYSTEM OFF
SYSTEM INTERLOCK
TYPE IN
END MESSAGE

**TABLE 4.1–1 CONSOLE DATA CHARACTER CODES**

[Except as noted otherwise these characters can be both transmitted and received.]

| OCTAL CODE | CHARACTER | OCTAL CODE | CHARACTER | OCTAL CODE | CHARACTER |
|---|---|---|---|---|---|
| 400 | [space] | 430 | X | 460 | = |
| 401 | A | 431 | Y | 461 | ∨ |
| 402 | B | 432 | Z | 462 | not used |
| 403 | C | 433 | \| | 463 | ∧ |
| 404 | D | 434 | ← | 464 | < |
| 405 | E | 435 | → | 465 | $ |
| 406 | F | 436 | ¬ | 466 | > |
| 407 | G | 437 | , | 467 | ; |
| 410 | H | 440 | 0 | 470 | ( |
| 411 | I | 441 | 1 | 471 | [ |
| 412 | J | 442 | 2 | 472 | ] |
| 413 | K | 443 | 3 | 473 | ) |
| 414 | L | 444 | 4 | 474 | ↓ |
| 415 | M | 445 | 5 | 475 | ↑ |
| 416 | N | 446 | 6 | 476 | : |
| 417 | O | 447 | 7 | 477 | ' |
| 420 | P | 450 | 8 | 500 | [error] |
| 421 | Q | 451 | 9 | 501 | a |
| 422 | R | 452 | $_{10}$ | 502 | b |
| 423 | S | 453 | . | 503 | c |
| 424 | T | 454 | + | 504 | d |
| 425 | U | 455 | – | 505 | e |
| 426 | V | 456 | * | 506 | f |
| 427 | W | 457 | / | 507 | g |

**TABLE 4.1-1 CONSOLE DATA CHARACTER CODES (Continued)**

| OCTAL CODE | CHARACTER | OCTAL CODE | CHARACTER |
|---|---|---|---|
| 510 | h | 560 | TAB |
| 511 | i | 561 | CARRIAGE RETURN |
| 512 | j | 562 | SET TAB STOP [REC only] |
| 513 | k | 563 | CLEAR TAB STOP [REC only] |
| 514 | l | 564 | not used |
| 515 | m | 565 | not used |
| 516 | n | 566 | BACKSPACE [REC only] |
| 517 | o | 567 | UNLOCK KEYBOARD [REC only] |
| 520 | p | 570 | END OF MESSAGE [TRANS only] |
| 521 | q | 571–777 | not used |
| 522 | r | | |
| 523 | s | | |
| 524 | t | | |
| 525 | u | | |
| 526 | v | | |
| 527 | w | | |
| 530 | x | | |
| 531 | y | | |
| 532 | z | | |
| 533–557 | not used | | |

Note: Character 567, UNLOCK KEYBOARD, must not be given if keyboard is already unlocked. In the event that 567 is sent when the keyboard is unlocked, the console logic must be reset using SYSTEM INTERLOCK.

The SYSTEM ON and SYSTEM OFF switches control the power to the entire G-20 System. To shut the system down, SYSTEM OFF and SYSTEM INTERLOCK must be operated simultaneously. SYSTEM INTERLOCK is also used separately to reset the console logic and place the console on-line. TYPE IN and END MESSAGE are used to request type-in and to signify completion of type-in, respectively.

For installations with multiple consoles, the SYSTEM ON, SYSTEM OFF controls can be disabled for consoles other than the master console.

Several indicators, some of which are combined with switches, convey visible or audible signals to the operator. The SYSTEM ON indicator is on when power is on. The INITIAL LOAD indicator is on during initial loading. The TYPE IN indicator is on whenever the typewriter keyboard is unlocked. An operator WARNING light comes on to indicate power failure, abnormal temperature, memory parity failure, or that a test switch has been left on. The tone speaker can be energized by program [see Section 2.3-9] and serves as an additional operator warning signal. The tone speaker can be disabled by shutting off the tone generator at the G-20.

A set of six program lights can be selectively lit by sending a command numeric to the console. Each of the last six bits of the command corresponds to one of the lights. The lights remain lit until another command numeric is sent to specify a new configuration. The bulbs may be tested by manually depressing the light buttons. The programmer may make various uses of the program lights, especially during debugging.

The Control Console is designed to control the G-20. It is not normally used off-line.

**4.1–2 Principles Of Operation** The switches and indicators of the CC-10 are, in general, self-explanatory and do not incorporate any unusual or unfamiliar features. The typewriter, on the other hand, has been designed to conform to a few special principles, since it constitutes a device which must be used by both the operator and the computer. Without proper attention to design details, ambiguity could exist as to whether the operator or the computer has control of the keyboard at a given moment. Also, the typewriter is so designed that any operation executed by the operator will be evidenced by a signal to the computer. The typewriter, considered as a terminal device, can operate in several of the states described in Section 3.3–1.

To deal with the possible ambiguity mentioned above, it is established as an operating principle that the keyboard always remains locked except during operator type-in. While the keyboard is locked, the computer has control of the typewriter and none of the keys can be operated. The operator requests control by operating the TYPE IN switch which interrupts the G-20. At the end of the current command, the keyboard may be unlocked by program control, whereupon the operator begins typing in his message. To conclude his message and cause it to be processed, he must operate the END MESSAGE switch which automatically locks the keyboard. Thus, the keyboard can be unlocked only by the computer, and locked only by the operator. The program should be so written that the computer will not attempt to use the typewriter while the keyboard is unlocked.

A further design feature of the typewriter pursuant to its dual use as an input and output device is that certain actions [in addition to UNLOCK KEYBOARD] are available only to the computer. These are SET TAB, CLEAR TAB, and BACKSPACE. These features permit complete control of format by program and obviate the possible ambiguity in the record inherent in use of BACKSPACE by the operator. There is no provision for verifying SET TAB and CLEAR TAB. Except for the keyboard lock mentioned above, all other typewriter operations are available to both the operator and the computer.

The margins are deliberately not made adjustable to avoid a possible programming restriction. With adjustable margins, the range of adjustment is limited; thus, certain formats can only be achieved by the use of TAB operations. For uniformity, all left margin adjustments are made by tabulating the appropriate amount from the left limit stop. An indicator is provided which is automatically set when the right limit stop is reached.

Two kinds of interrupts are used in the operation of the typewriter as follows: depression of TYPE IN generates an interrupt and sets the interrupt flip-flop NT2. After securing the program, the routine will acknowledge the operator's signal and unlock the keyboard for him. The routine can, of course, refuse recognition to the operator if the program is using the console in question. Each time the operator depresses a key, a character is printed and its code equivalent is sent to a holding register. Mechanical completion of the stroke sets flip-flop NT1 whose change of state generates an interrupt. [Because the NT1 and NT2 signals are on two different interrupt lines, the G-20 can distinguish between them.] The computer may, within the next 20 milliseconds, interrogate the holding register to determine the character. Longer delay may entail loss of the character. The character code is sent to the holding register whether the key was depressed by the operator or by a solenoid from computer signal. In the latter case, the character in the holding register can be used for verification, if desired. The G-20 program interrogates the interrupt flip-flops to determine their state. Note that failure to reset NT1 can cause NT1 interrupts to be inhibited.

Those states taken up in Secion 3.3-1 which apply to the typewriter are OUT of Service, STANDBY, CALLED, BUSY-QUIET, and BUSY-ALERT.

Table 4.1–2 lists the line signals which can be transmitted or received by a console. Command signals are checked for parity by the console, but data characters are not.

Table 4.1-3 lists some other signals and switch operations significant to the operation of the CC-10.

## TABLE 4.1–2 CONSOLE COMMAND CODES

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 002 | GRN | Green. A response the console gives to line commands. Causes the G–20 program to skip one command word.<br>Timing 8.4 microseconds. |
| 003 | RED | Red. A response the console gives to line commands. Causes the G–20 program to take the next command in sequence.<br>Timing 8.4 microseconds. |
| 011 | OUT | Out. Console in CALLED or BUSY–ALERT state answers GRN, disconnects itself from the communication line, and enters OUT of Service state.<br>Timing 8.4 microseconds. |
| 016 | TRA | Transmit. Console in CALLED state delivers to the line the contents of the holding register which will contain the code for the last character sent by the operator or the last action taken by command from the G–20. The latter is used for verification purposes.<br>Timing 8.4 microseconds. |
| 060 | QRD | Query Ready. Console in CALLED state answers GRN if ready to transmit or receive, not at right margin, and has no mechanical operation in process. Otherwise answers RED. In BUSY–ALERT state answers RED.<br>Timing 8.4 microseconds. |
| 062 | QIL | Query no Interlock. Console in CALLED or BUSY–ALERT state answers GRN unless right margin interlock is open. Otherwise answers RED.<br>Timing 8.4 microseconds. |
| 066 | QN2 | Query no NT2 Interrupt. Console in CALLED or BUSY–ALERT state resets the NT2 flip-flop. Answers GRN if the flip-flop was reset, and RED if flip-flop was set.<br>Timing 8.4 microseconds. |
| 067 | QN1 | Query no NT1 Interrupt. Console in CALLED or BUSY–ALERT state resets the NT1 flip-flop. Answers GRN if the flip-flop was reset, and RED if flip-flop was set.<br>Timing 8.4 microseconds. |

| TABLE 4.1-2 CONSOLE COMMAND CODES (Continued) | | |
|---|---|---|
| OCTAL CODE | ALPHA CODE | DESCRIPTION |
| 1nn | 1nn | A command numeric. Console in CALLED or BUSY-ALERT state interprets "nn" as a 6-bit binary number for setting the six program lights so that 1 = ON, 0 = OFF. The left bit sets #1 light, etc. For signal 112, nn = 12 = 001010; the lights are set: off, off, on, off, on, off. Response is GRN.<br>Timing 8.4 microseconds. |
| 200 + uuu | CALs | Call to the console itself. CC-10 in CALLED state stays in CALLED state. CC-10 in STANDBY state enters CALLED. CC-10 in BUSY-QUIET state enters BUSY-ALERT. Answers GRN.<br>Timing 8.4 microseconds. |
| 200 + uuu | CALo | Call to some other unit. CC-10 in CALLED state goes to STANDBY. CC-10 in BUSY-ALERT goes to BUSY-QUIET. No answer.<br>Timing 8.4 microseconds. |
| 400 - 532 | ddd | Printing characters. Characters 462 and 500 are not used. CC-10 goes to BUSY-ALERT. Upon completion sends interrupt and goes to CALLED. |
| 533 - 557 | none | Not used. |
| 560 - 570 | none | Control characters. Console enters BUSY-ALERT and performs the operation. Upon completion goes to CALLED. Numbers 564 and 565 are not used. |
| 400 - 577 | none | Any undefined data character in this range will be acknowledged with a GRN reply. The Error code 500 will be placed in the holding register. |
| 600 - 777 | none | These codes are not recognizable by the CC-10; it will not respond in any manner to them. |

| TABLE 4.1-3 CONSOLE SIGNALS, MISCELLANEOUS | | |
|---|---|---|
| LINE SIGNAL | NAME | DESCRIPTION |
| Interrupt [NT1] | Operation Complete | In going from reset to set, NT1 sends an interrupt. Completion of any mechanical operation sets NT1. Characters can now be transmitted or received. If in BUSY-ALERT, goes to CALLED. If in BUSY-QUIET, goes to STANDBY. |
| Interrupt [NT2] | Type in | Caused by operator depressing TYPE IN switch. Requests control of the typewriter. |
| None | Key Depressed | Caused by operator depressing typewriter key. Causes character to be sent to holding register. |
| Interrupt [NT1] | End Message | Caused by operator depressing END MESSAGE switch. Locks keyboard, sends interrupt, and sends 570 to holding register. [570 sent to console will not lock keyboard.] |
| None | On-Line | Depression of SYSTEM INTERLOCK places console on-line in STANDBY state. |

Examples 4.1–1, 4.1–2 and 4.1–3 show the sequence of line signals for typical console operations.

**EXAMPLE 4.1–1 SEQUENCE OF LINE SIGNALS – TYPE-OUT WITHOUT VERIFICATION**

| G–20 | CC–10 | REMARKS |
|---|---|---|
| CALs | GRN | Console enters CALLED state. |
| QRD | GRN | Test ready. |
| QN1 | RED | Resets NT1 flip-flop, answer ignored. |
|  | GRN |  |
| ddd | GRN | Data character. Console enters BUSY-ALERT state, and types character. |
|  | (Types) |  |
| [A] | INT | Interrupt [NT1] at completion of type cycle. Console enters CALLED state. |
|  | [NT1] |  |
| CALs | GRN | G–20 services the interrupt. Console enters CALLED state. |
| QN1 | RED | NT1 flip-flop is reset. |
| QRD | GRN | Test ready. |
| ddd | GRN | Data character. |
|  |  | Return to [A]. |
|  |  | When the type-out is complete, the typewriter remains in the CALLED state until some other unit is called, at which time it returns to STANDBY. The NT1 flip-flop will remain set until reset by program. |

**EXAMPLE 4.1–2 SEQUENCE OF LINE SIGNALS – TYPE-OUT WITH VERIFICATION**

| G–20 | CC–10 | REMARKS |
|---|---|---|
| CALs | GRN | Console enters CALLED state. |
| QRD | GRN | Test ready. |
| QN1 | RED | Resets NT1 flip-flop. Answer ignored. |
|  | GRN |  |
| ddd | GRN | Data character. Enters BUSY-ALERT state and types the character. |
|  | (Types) |  |
| [A] | INT | Interrupts at completion of type cycle. Enters CALLED state. |
|  | [NTI] |  |
| CALs | GRN | G–20 services the interrupt. Console enters CALLED state. |
| QN1 | RED | Query Interrupt. Interrupt flip-flop is reset. |
| TRA | ddd | G–20 requests and receives verification. |
| QRD | GRN | Test ready. |
| ddd | GRN | G–20 sends next character. |
|  |  | Return to [A]. |
|  |  | When the type-out is completed, the typewriter remains in the CALLED state until some other unit is called, at which time it returns to STANDBY. NT1 flip-flop remains set until reset by program. |

## EXAMPLE 4.1–3 SEQUENCE OF LINE SIGNALS – CONSOLE INPUT

| G–20 | CC–10 | REMARKS |
|---|---|---|
| | INT<br>[NT2] | TYPE IN switch depressed by operator. |
| CALs | GRN | Console enters CALLED state. |
| QN1 | { RED<br>GRN | Reset NT1 flip-flop; answer ignored. |
| QN2 | RED | NT2 flip-flop is reset. |
| 567 | GRN | Unlock keyboard. Console enters BUSY-ALERT as the mechanical unlocking proceeds, then goes to CALLED. |
| | INT<br>[NT1] | At end of unlock cycle, console interrupts. TYPE IN button lights to inform operator. |
| CALs | GRN | Console enters CALLED state. |
| QN1 | RED | NT1 flip-flop is reset. |
| [A] | | (Operator types) |
| | INT<br>[NT1] | At completion of type cycle, console interrupts. |
| CALs | GRN | Console enters CALLED state. |
| QN1 | RED | Interrupt flip-flop is reset. |
| TRA | ddd | G–20 requests and receives the character. |
| | | Return to [A] unless END MESSAGE switch is depressed. |
| | INT<br>[NT1] | End Message sends interrupt and locks keyboard. |
| CALs | GRN | Process interrupt. |
| QN1 | RED | Resets interrupt flip-flop. |
| TRA | 570 | End of Message character sent. |

## SECTION 4.2 – PAPER TAPE STATION, PT–10

The PT–10 Paper Tape Station provides the G–20 user with the facility for reading or punching any 5, 6, 7, or 8–level paper tape of standard dimensions. Reading is bidirectional at 250 or 500 characters per second, and punching is at 110 characters per second. The PT–10, Figure 4.2–1, is 64 inches high, 34 inches wide, 29 inches deep and weighs 1000 pounds, approximately. It draws 1 KVA of 115-volt, 60-cycle, single-phase power.

**4.2–1 Organization Of The PT–10** As shown in Figure 4.2–2 the punch and read stations of the PT–10 share a line coupler and associated command decoding but are otherwise distinct units and have distinct calling signals. The PT–10 accepts G–20 System standard form line commands and checks them for even parity. Parity of data characters is also checked. Except for the replies RED, GRN, END, ERR, STP all characters delivered by the PT–10 to a communication line have the standard format for data characters. The characters RED, GRN, END, ERR, STP have the standard format for command characters.

The feed and takeup reels of the reader are controlled by servos to maintain proper slack in the tape. Tape reels up to 8 inches in diameter can be handled at high speed, 500 characters or 50 inches per second. Reels up to 10 1/2 inches in diameter can be handled at low speed, 250 characters or 25 inches per second. Operation at 500 characters per second of reels larger than 8 inches may result in tape breakage. Paper tape and some other materials can be punched.

FIGURE 4.2–1 PAPER TAPE STATION, PT–10

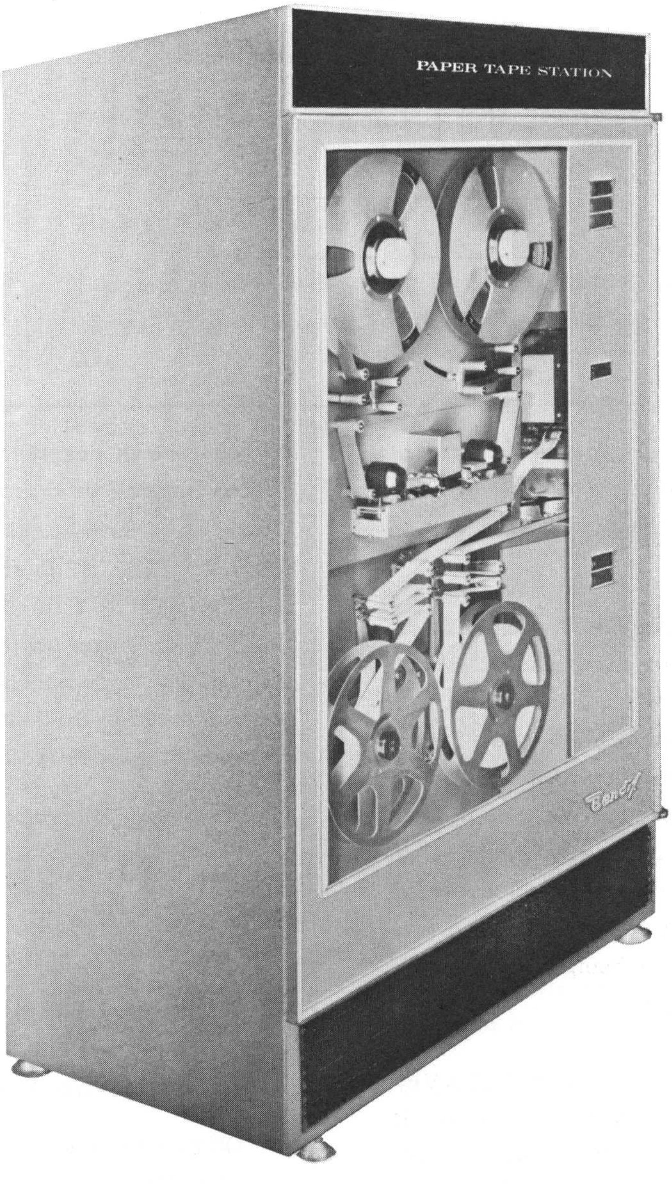
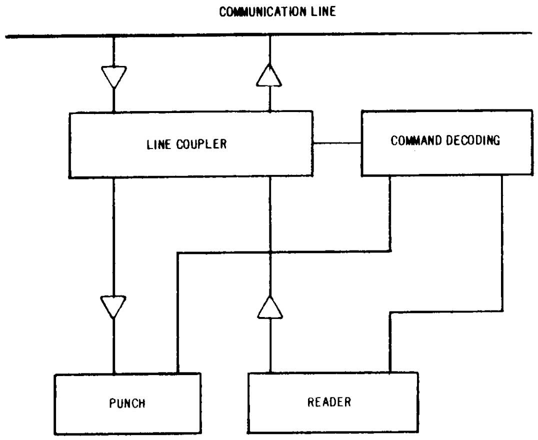


FIGURE 4.2-2 SIMPLIFIED BLOCK DIAGRAM OF PT-10 PAPER TAPE STATION



All forms of tape of less than 40 per cent light transmission can be read. Interlocks are provided for stopping either the punch or the reader when out of tape.

Binary information goes from the G–20 memory to paper tape [and vice versa] unchanged except for the rules of the communication line requiring a data flag in bit 8 and an even parity bit in bit 9. See Figure 3.4–1. Longitudinal or lateral checks are performed by program in the G–20. Four or five tape spaces correspond to one G–20 word in the block modes. In the single character interrupt mode a character being punched originates from the least significant 8 bits of a G–20 word, and a character being read is stored in register LRE. See Chapter 3.

**4.2–2 Controls of the PT–10** The ON/SLAVE/OFF Power switch of the PT–10 turns it on or off unconditionally, or places it under control of the System Power signal line. The INITIAL LOAD switch starts initial loading of the G–20 from Paper Tape. The punch and reader line switches place the units ON-LINE or OUT of Service. A LEADER button causes leader to be run out as desired. The number of levels to be read is set by the READ LEVEL switch to 5, 6, 7, or 8. FAST/SLOW provides 500/250 characters per second reading speed, and REWIND winds all of the tape back onto the feed reel. The HALT switch has two positions, PROGRAM and END PUNCH. In the former position, the tape is halted only by a line command STP [050] or END [004]. In the latter position, it is halted also when it reads an end code punched in the tape. The end code is established by the setting of a group of END CODE switches to the desired octal code.

The following indicators are provided on the PT–10:

INITIAL LOAD
INTERLOCK OPEN
PUNCH ON-LINE
PUNCH OUT of Service
READER ON-LINE
READER OUT of Service

**4.2–3 Operating States** Both the punch and the reader can occupy the following states with respect to the communication line:

OUT of Service
STANDBY
CALLED
BUSY-ALERT
BUSY-QUIET
INSTRUCTED
MESSAGE

These states conform to the rules outlined in Chapter 3.

**4.2–4 Punching** Table 4.2–1 lists the line signals applicable to punching.

**TABLE 4.2-1 PT-10 PUNCH COMMAND CODES**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 002 | GRN | A reply to calls and queries. Causes G-20 program to continue. |
| 003 | RED | A reply to queries. Causes G-20 program to branch. |
| 004 | END | End of data block. Unit in MESSAGE state goes to STANDBY. No reply.<br>Timing 5 microseconds. |
| 005 | ERR | End of data block containing an Error. Unit in MESSAGE state goes to STANDBY. No reply.<br>Timing 5 microseconds.<br>The punch station will send ERR in lieu of REQ if tape supply is interlocked, or if a character with incorrect parity is received in the MESSAGE state.<br>Timing 5 microseconds. |
| 010 | SDT | Start Data Transfer. Unit in INSTRUCTED state will enter MESSAGE state and send REQ.<br>Timing 5 microseconds. |
| 011 | OUT | Go OUT of Service. Unit in CALLED or BUSY-ALERT state will reply GRN, stop its motor [if operating] and disconnect itself from the line.<br>Timing 5 microseconds. |
| 013 | BLK | Block mode. Unit in CALLED state enters INSTRUCTED state and answers GRN. This command should be preceded by QRD.<br>Timing 5 microseconds. |
| 050 | STP | Stop. Unit in CALLED state replies GRN and stops punch motor. The punch motor should be stopped when it is not needed.<br>Timing for reply 5 microseconds. |
| 056 | STT | Start. Unit in CALLED state answers GRN, and starts punch motor. When up to speed, sends an interrupt.<br>Timing for reply 5 microseconds.<br>Timing for interrupt 4 seconds max. |

**TABLE 4.2–1 PT–10 PUNCH COMMAND CODES (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 060 | QRD | Query Ready. Unit in CALLED state replies GRN if motor is up to speed and there are no open interlocks. Replies RED otherwise. Unit in BUSY–ALERT replies RED.<br>Timing 5 microseconds. |
| 062 | QIL | Query no Interlock. Unit in CALLED or BUSY–ALERT replies GRN if there are no open interlocks. Otherwise answers RED.<br>Timing 5 microseconds. |
| 067 | QIN | Query no Interrupt. Unit in CALLED or BUSY–ALERT replies GRN if there has been no interrupt since the last QIN. Replies RED if there has been such an interrupt. Resets interrupt indicator.<br>Timing 5 microseconds. |
| 200 +<br>uuu | CALs | Call to the unit itself. Unit in STANDBY goes to CALLED. Unit in BUSY–QUIET goes to BUSY-ALERT. Unit in INSTRUCTED goes to CALLED. Reply GRN.<br>Timing 5 microseconds. |
| 200 +<br>uuu | CALo | Call to some other unit. Unit in CALLED goes to STANDBY. Unit in BUSY–ALERT goes to BUSY–QUIET.<br>Timing 5 microseconds. |
| 400 +<br>ddd | ddd | A data character. [$0 \leq ddd \leq 377.$] Unit in CALLED state replies GRN, goes to BUSY–ALERT, punches a data character, and sends an interrupt. It returns to CALLED or STANDBY. Unit in MESSAGE state punches the character and replies with an REQ. If the tape supply is interlocked, responds with an ERR.<br>Timing for GRN reply 5 microseconds.<br>Timing for REQ or interrupt 3.5 to 15 milliseconds. |
| none | INT | An interrupt request sent by the PT–10 to signal completion of a task, [e.g., start motor or punch a character]. |
| none | REQ | A request for a data character sent on the REQ line. If tape feed is interlocked, PT–10 will send ERR instead of REQ. |

A typical sequence of line signals for punching in the interrupt mode is shown below.

**EXAMPLE 4.2–1**
**SEQUENCE OF LINE SIGNALS – PT–10 PUNCH, INTERRUPT MODE**

| CONTROL DEVICE | PT–10 | REMARKS |
|---|---|---|
| CALs | GRN | Punch enters CALLED state. |
| STT | GRN | Punch motor starts. |
| . . . . | . . . . | . . . . |
|  | INT | Punch motor is up to speed, sends interrupt. |
| [A]CALs | GRN | Enters CALLED state. |
| QIN | RED | Source of interrupt is identified. |
| QRD | GRN | Unit is ready to punch. |
| ddd | GRN | Unit enters BUSY–ALERT. |
| . . . . | . . . . | . . . . |
|  | INT | Punching complete. Unit goes from BUSY-ALERT to CALLED [or from BUSY-QUIET to STANDBY] and sends an interrupt. Returns to [A] for subsequent characters. |
| CALs | GRN | At completion, punch motor should be stopped by command. |
| QIN | RED |  |
| STP | GRN |  |

A typical sequence for punching in the block mode is given below.

**EXAMPLE 4.2–2**
**SEQUENCE OF LINE SIGNALS – PT–10 PUNCH, BLOCK MODE**

| CONTROL DEVICE | PT–10 | REMARKS |
|---|---|---|
| CALs | GRN | Punch enters CALLED state. |
| STT | GRN | Punch motor starts. |
| . . . . | . . . . | . . . . |
|  | INT | Motor is up to speed. Sends interrupt. |
| CALs | GRN |  |
| QIN | RED | Interrupt is identified. |
| QRD | GRN | Unit is ready to punch. |
| BLK | GRN | Unit enters INSTRUCTED state. |
| SDT | REQ | Enters MESSAGE state, requests character. |
| ddd | REQ | First character sent. |
| . . . . | . . . . | . . . . |
| ddd | REQ | Last character. |
| END |  | Unit enters STANDBY state. |
| CALs | GRN | At completion, punch motor should be stopped by command. |
| STP | GRN |  |

**4.2–5 Reading** Line signals applicable to the PT–10 read station are shown in Table 4.2–2.

## TABLE 4.2-2 PT-10 READ COMMAND CODES

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 002 | GRN | A reply the unit gives to calls and queries. |
| 003 | RED | A reply the unit gives to queries. |
| 004 | END | End. Unit in MESSAGE state goes to STANDBY. Tape is stopped. No reply.<br>Timing 5 microseconds. |
| 005 | ERR | End of block containing an Error. Otherwise same as END.<br>ERR is sent in reply to TRA if tape is interlocked.<br>ERR is sent in reply to REQ if tape is interlocked. |
| 010 | SDT | Start Data Transfer. Unit in INSTRUCTED state goes to MESSAGE state. The pinch roller on reader is energized.<br>Timing 5 microseconds. |
| 011 | OUT | Go OUT of Service. Unit in CALLED or BUSY–ALERT answers GRN, disconnects itself from the line and stops the tape.<br>Timing 5 microseconds. |
| 013 | BLK | Block mode. Unit in CALLED state will enter INSTRUCTED state. Must be preceded by QRD. Answers GRN.<br>Timing 5 microseconds. |
| 016 | TRA | Transmit. Unit in BUSY–ALERT state sends a data character from register. However, if unit is interlocked it will send ERR in answer to TRA and go to CALLED. If it reaches an end code while in the end punch mode, will send STP, stop the tape and go to CALLED.<br>Timing 5 microseconds. |
| 044 | FWD | Forward. Unit in CALLED state replies GRN. Prepares to go forward.<br>Timing 5 microseconds. |
| 045 | BAC | Backward. Unit in CALLED state replies GRN. Prepares to go backward.<br>Timing 5 microseconds. |
| 050 | STP | Stop. Unit in BUSY–ALERT. Stops tape motion. The unit replies GRN and enters the CALLED state. This instruction must be sent within 1.5 milliseconds after the interrupt was sent, when operating at 500 characters per second, or within 3.0 milliseconds, when operating at 250 characters per second. See TRA [016] . |

| TABLE 4.2-2 PT-10 READ COMMAND CODES (Continued) | | |
|---|---|---|
| OCTAL CODE | ALPHA CODE | DESCRIPTION |
| 056 | STT | Start. Unit in CALLED state energizes pinch roller and begins to read in the previously specified direction. Replies GRN. Enters BUSY–ALERT state. When a character has been read into the register the unit sends an interrupt and continues reading the next character.<br>Timing 5 microseconds to GRN, 5 milliseconds to Interrupt. If the direction is opposite to that of the last operation, add 200 milliseconds max. |
| 060 | QRD | Query Ready. Unit in BUSY–ALERT state replies GRN if there is a character in the register waiting to be transmitted and there are no interlocks. Otherwise answers RED.<br>Unit in CALLED state answers GRN if there are no interlocks. Otherwise RED.<br>Timing 5 microseconds. |
| 062 | QIL | Query no Interlock. Unit in CALLED or BUSY–ALERT state answers GRN if there are no interlocks. Otherwise RED.<br>Timing 5 microseconds. |
| 067 | QIN | Query no Interrupt. Unit in CALLED or BUSY–ALERT state answers GRN if there has been no interrupt since last QIN. Otherwise RED. Resets interrupt indicator.<br>Timing 5 microseconds. |
| 200 +<br>uuu | CALs | Call to the unit itself. Unit in STANDBY goes to CALLED. Unit in BUSY–QUIET goes to BUSY–ALERT. Unit in INSTRUCTED goes to CALLED. Answers GRN.<br>Timing 5 microseconds. |
| 200 +<br>uuu | CALo | Call to some other unit. Unit in CALLED goes to STANDBY. Unit in BUSY–ALERT goes to BUSY–QUIET.<br>No answer.<br>Timing 5 microseconds. |
| 400 +<br>ddd | ddd | A data character. [0 ≤ ddd ≤ 377.] PT–10 in MESSAGE state sends a data character for each REQ. In BUSY–ALERT state sends a data character in response to TRA. |
| none | INT | An interrupt sent by the PT–10 to signify that a character is ready to be transmitted. |
| none | REQ | A request for data character sent on the request line. PT–10 in MESSAGE state will reply with data character or with END or ERR. The first REQ following SDT causes the first non-zero character to be read from the tape. |

Examples are given of the sequence of line signals for:

Read: Interrupt Mode
Read: Block Mode
Read: Initial Load G–20

These sequences are self-explanatory except for initial load G–20. For initial loading G–20 at turn-on:

1. Turn on System power. Wait for INITIAL LOAD light at console or G–20 control panel.
2. Mount tape in reader.
3. Operate INITIAL LOAD control on PT–10.

For initial loading after power turn-on:

1. Make sure all peripheral equipment is in STANDBY state by operating ŻERO MACHINE switches.
2. Operate XCM on, INITIAL LOAD, XCM off, and START CLOCK switches at G–20.
3. Mount tape in reader.
4. Operate INITIAL LOAD control on PT–10.

In either of the above cases the PT–10 will begin loading with the first non-zero character encountered on the paper tape.

**EXAMPLE 4.2–3 SEQUENCE OF LINE SIGNALS – PT–10 READ, INTERRUPT MODE**

| CONTROL DEVICE | PT–10 | REMARKS |
|---|---|---|
| CALs | GRN | Unit enters CALLED state. |
| QRD | GRN | No interlocks. |
| FWD | GRN | Specifies direction. |
| [BAC] | | |
| STT | GRN | Unit begins to read character. Enters BUSY-ALERT. Start is delayed .2 seconds if direction is opposite to that of last operation. |
| . . . . | . . . . | In the event of a CALo, the PT–10 returns to BUSY–QUIET. |
| [A] | INT | Character is now in register. |
| CALs | GRN | Enters BUSY–ALERT, if not already in this state. |
| QIN | RED | Identifies interrupt. |
| QRD | GRN | Character is ready to be sent. |
| TRA | ddd | Character goes on the line. Program returns to [A]. |
| TRA | STP | When in end punch mode, PT–10 sends STP in lieu of data character. |
| or | | |
| STP | . . . . | When in program halt mode, control device may send STP in lieu of TRA. |

## EXAMPLE 4.2–4 SEQUENCE OF LINE SIGNALS – PT–10 READ, BLOCK MODE

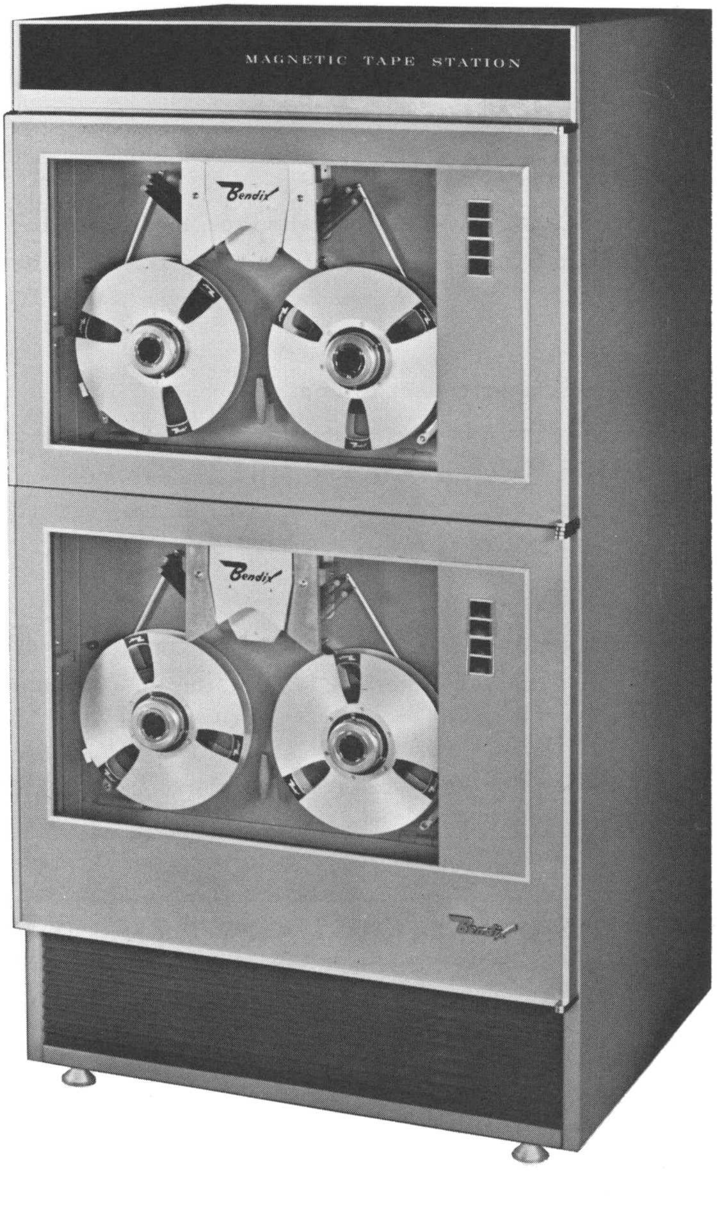
| CONTROL DEVICE | PT–10 | *First non-zero character encountered.<br>REMARKS |
|---|---|---|
| CALs | GRN | Unit enters CALLED state. |
| QRD | GRN | No interlocks. |
| FWD | GRN | Specifies direction. |
| [BAC] | | |
| BLK | GRN | Enters INSTRUCTED state. |
| SDT | No Answer | Enters MESSAGE state, starts tape, reads character to register.<br>[Start is delayed .2 seconds if direction is opposite to the last operation]. |
| REQ | . . . . | . . . . |
| . . . . | ddd* | First character sent on line. |
| REQ | . . . . | . . . . |
| . . . . | ddd | Next data character. |
| . . . . | . . . . | . . . . |
| REQ | END<br>[ERR] | End of block. In the end punch mode, the PT–10 can terminate the block with END. If an interlock is open, the PT–10 can terminate the block with ERR. |
| or<br>END<br>[ERR] | | The control device can terminate the block with END or ERR. No answer from PT–10. |

## EXAMPLE 4.2–5 SEQUENCE OF LINE SIGNALS – PT–10 READ, INITIAL LOAD G–20

| PT–10 | G–20 | REMARKS |
|---|---|---|
| | | System power on, G–20 and PT–10 INITIAL LOAD switches operated. PT–10 in MESSAGE state. |
| ddd* | REQ | ddd $\rightarrow_{31} (64)_{24}$ in G–20 |
| ddd | REQ | ddd $\rightarrow_{23} (64)_{16}$ in G–20 |
| . . . . | . . . . | . . . . |
| ddd | REQ | Last character |
| END | | Block terminated by end flag in bit 31 of last word on tape. PT–10 goes to STANDBY. G–20 begins executing at location 65 = $101_8$. |

*First non-zero character encountered.

FIGURE 4.3–1 MAGNETIC TAPE MODULE, MT–10



## SECTION 4.3 MAGNETIC TAPE, MT–10*

The MT–10 Magnetic Tape provides a high density dual diversity recording of 8–bit [or 6–bit] characters with lateral parity check, on 1–inch wide mylar tape. The transfer rate reading or writing is 120,000 characters per second at a tape speed of 110 inches per second. For continous reading or writing the interblock time is 6 milliseconds. Deletion or rewrite of a single block is possible. Double speed, non-stop positioning any number of blocks in either direction is provided.

A magnetic tape group consists of one tape control and from one to four tape transports. The tape transports are connected through the tape control to the communication line and thence to other equipment in the system. Only one of the transports of a group may be reading, writing, or deleting at a given moment, but there is no restriction on searching or rewinding. The maximum number of tape transports in a G–20 System is limited only by the 128 distinct calls available.

The tape transports are physically arranged in modules 64 inches high, 34 inches wide, and 28 inches deep. See Figure 4.3–1. Such a module may be equipped with a tape control and one tape transport, MT–10 (A); a tape control and two tape transports, MT–10 (B); or two tape transports and no control, MT–10 (C). Two modules belonging to the same tape group must be kept adjacent. The cable between the tape group and other devices with which it communicates can be up to 750 feet in length.

The weights and power drains for the three types of modules are given below:

| | MT–10 (A) | MT–10 (B) | MT–10 (C) |
|---|---|---|---|
| Weight in pounds | 750 | 1,000 | 900 |
| Power in KVA, 115-volt, 60-cycle | 3.0 | 4.5 | 3.0 |

**4.3–1 Principles of Operation** The MT–10 records each bit of information in two physically separated tracks on magnetic tape. If either of these two bits is read back correctly, the information will be recovered. This is called a dual diversity recording because the information for each data bit is recorded in two physically separated places.

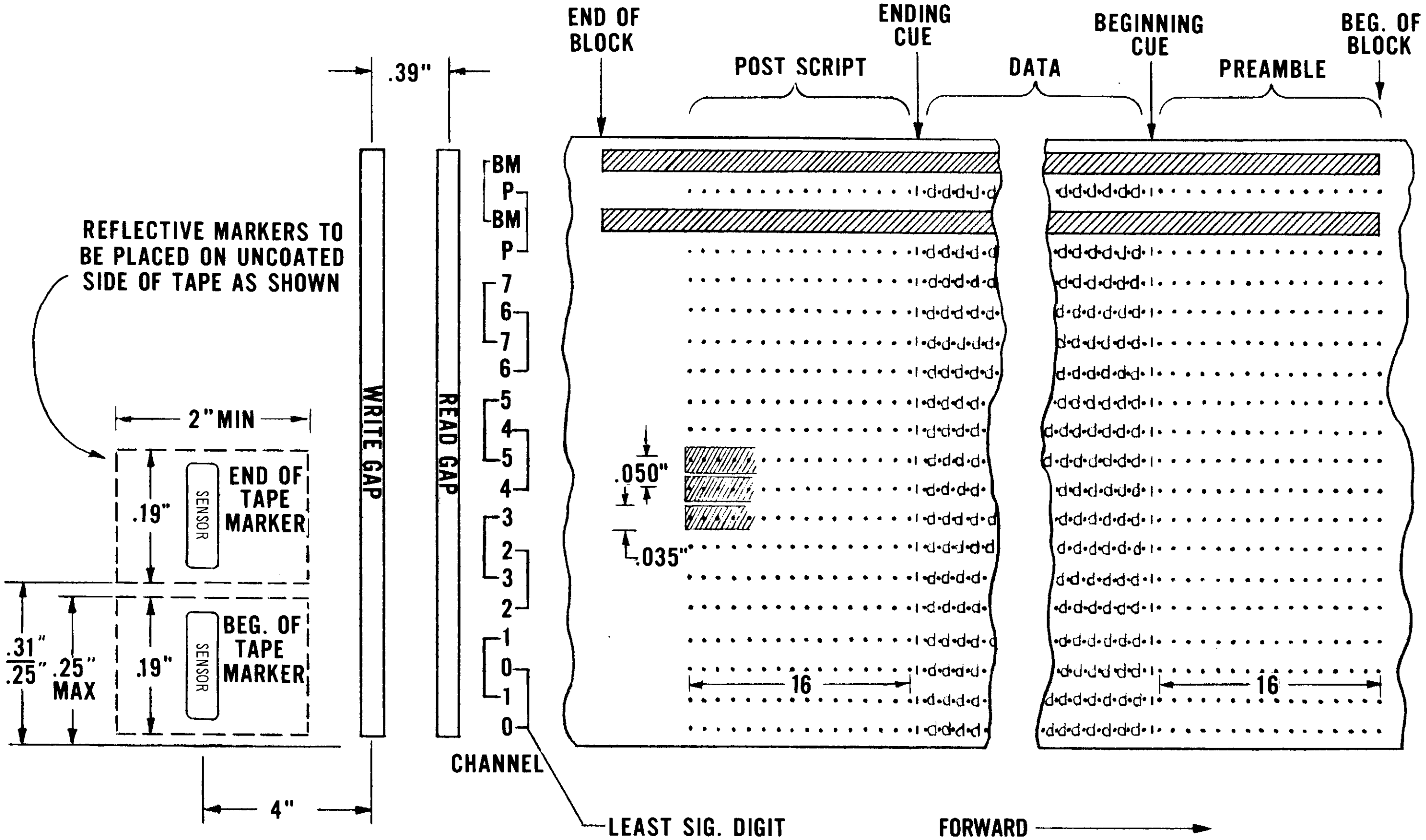
A 12–bit block counter in each tape transport permits high speed non-stop positioning of any tape without requiring attention from the tape control. This operation is called "slewing".

FORMAT OF A TAPE BLOCK. Figure 4.3–2 indicates schematically the appearance of a tape block as viewed from the oxide-coated side of the tape. A 32–bit G–20 word is written on tape as four 8–bit characters, or as one 8–bit character and four 6–bit characters, the latter containing two leading zeros. Each character is accompanied by an even parity bit. The most significant character of the word is recorded first. The data flag which accompanies a data character on the Bendix communication line is not recorded on magnetic tape. When a character is read from tape, the data flag for communication line use is supplied by the hardware. As shown in Figure 4.3–2, the eight data bit channels and the parity bit channel are each recorded in two tracks on the tape. This is a clocked recording; a clock pulse [shown as a dot] precedes each character. The block mark channel contains a non-return-to-zero recording [shown as a heavy line when on].

The information of a block is preceded by a preamble of 16 blank characters and a "cue" character of all 1's. This latter is a special character in that it has incorrect parity and is used for control purposes. The information is followed by a cue character and a postscript of 16 blank characters. The balance of the space in the data tracks out to the end of the block [indicated by the block mark track] is unrecorded. The preamble, postscript, and cue characters are supplied by the hardware.

BLOCK DIAGRAM. The simplified block diagram of the tape system in Figure 4.3–3 indicates the manner in which the system operates. During receipt of line commands, the tape control decodes the command, interrogates its registers if necessary, sends the required response, and sends signals to the appropriate tape transport if required. The tape control recognizes its own call signal and those of its 1–4 associated tape transports as assigned at installation. It also can determine which transports are idle, slewing, rewinding, or interlocked [i.e., OUT of Service] and can answer queries on these points. Any communication with the tape control is forbidden while a tape

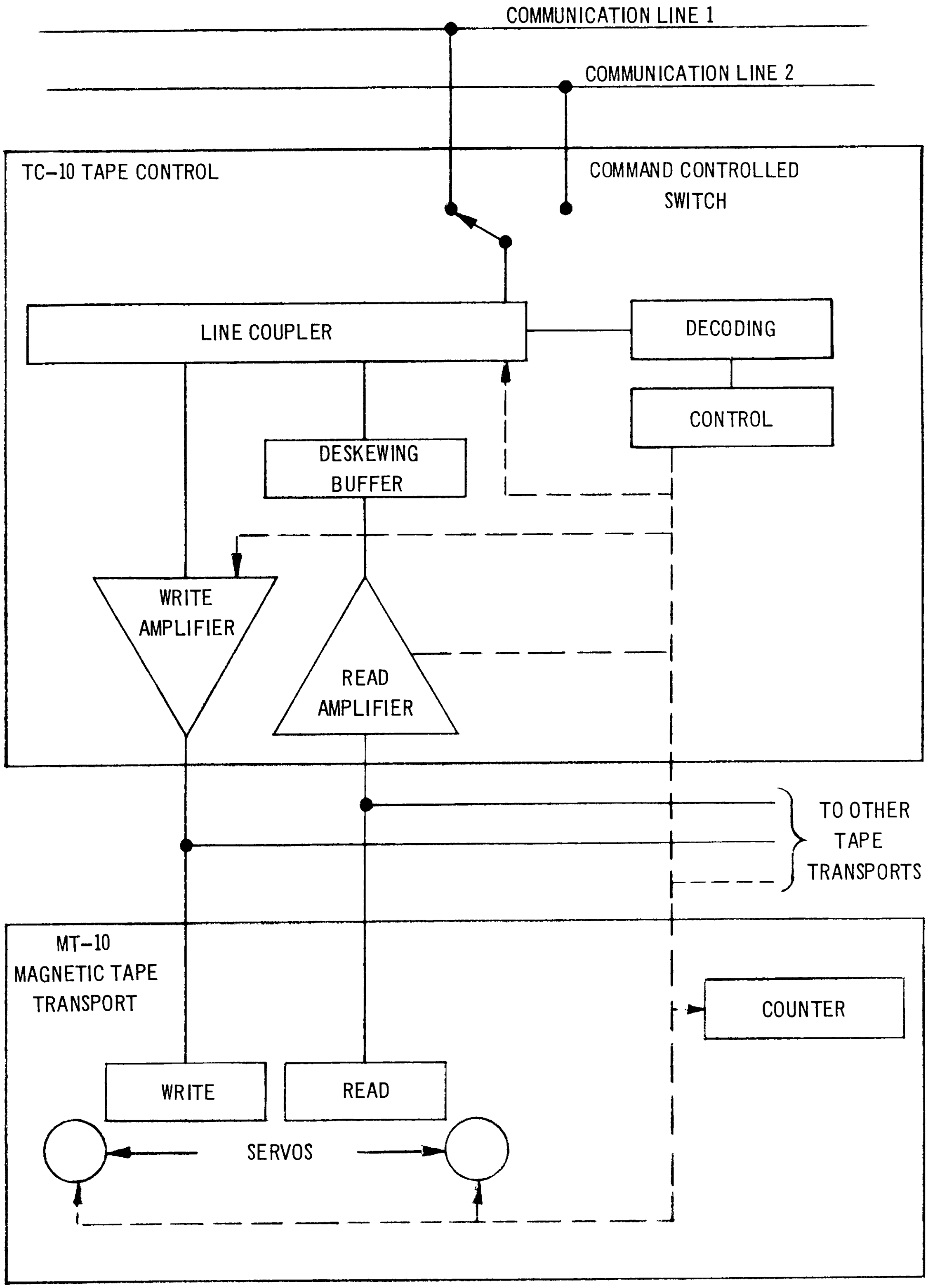
*Description applies to MT–10 model 2. For MT–10 model 1, see Bulletin BEB–001.



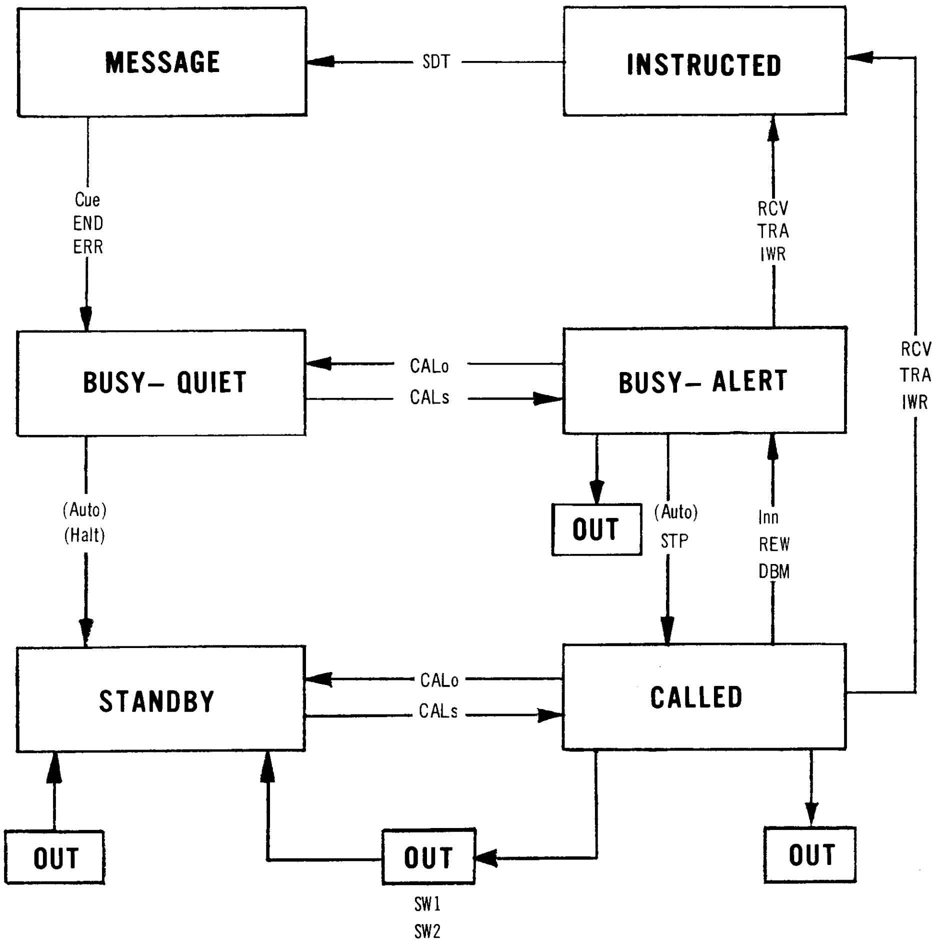
Note: ·=Clock Mark
i=Cue Character Bit
d=Data Bit
P=Parity Channels
BM=Block Mark Channels

FIGURE 4.3–2 LAYOUT OF INFORMATION ON MT–10 MAGNETIC TAPE

FIGURE 4.3-3 SIMPLIFIED BLOCK DIAGRAM OF MT-10 MAGNETIC TAPE SYSTEM



## FIGURE 4.3–4 MAGNETIC TAPE OPERATING STATES



transport of the group is reading or writing. For systems in which an MT—10 must operate on two different communication lines, Bendix supplies a command-controlled switch in the tape control unit.

During recording of data, the tape control receives the characters from the line and delivers write signals to the tape transport. During reading of data, the tape control unit performs a "de-skewing" operation necessitated by the high density of the recording. Because the bits making up one character are in different tracks and may not come under the read head at the same instant, de-skewing buffers are necessary. The buffer associated with each channel is capable of storing up to six bits. After a time delay sufficient to take care of the worst skew, the first recorded bit in each buffer is read out and these bits constitute the character transmitted.

**4.3—2 Operating States** The tape control handles all of the communication for a tape group and is able by means of the answers it gives to calls and queries, to reflect the operating states of the transports in the group. Each tape transport has its own unique call signal and calls are ordinarily addressed to an individual transport. The tape control has a call, but this is used only for special purposes. Slewing and rewinding are initiated with the aid of the tape control, but, completed without its assistance. In general, other communication with a tape group is forbidden whenever any one of the tape transports is reading, writing, or deleting. A transport must occupy one of the following states:

| | | |
|---|---|---|
| OUT of Service | STANDBY | CALLED |
| | BUSY-QUIET | BUSY-ALERT |
| | MESSAGE | INSTRUCTED |

These states are the same as those mentioned in Chapter 3 with minor exceptions. A transport occupying one of the states in the second column has its "command gate" closed; it cannot be commanded or queried. For states in the third column the "command gate" is open; commands and queries can be handled. See also Figure 4.3—4.

OUT of Service. A tape transport in this state is effectively disconnected. If not interlocked, it can be placed in service by manually depressing the ON—LINE [green] button. A transport can be taken OUT of Service by the OUT button or the OUT command. A transport that is interlocked is OUT of Service. There will be an interlock condition if the front door is open, the load handle is in the loading position, or a tension arm is fully extended. The tape control will not answer if a transport's number is called while that transport is OUT of Service.

STANDBY. A tape transport which is on-line and not engaged in an operation [including start/stop] and has not heard its call is in the STANDBY state. In this state it responds only to its own call number with correct parity. It answers GRN and enters the CALLED state.

CALLED. A tape transport which is not engaged in an operation and which has answered a call, is in the CALLED state. It will answer calls and commands. A call to any other unit will cause the transport to return to STANDBY.

INSTRUCTED. A tape transport which has been called and instructed for reading or writing a block of data, but has not received SDT, start data transfer, is in the INSTRUCTED state. If it should now hear its own call it will return to BUSY-ALERT.

BUSY-QUIET. A tape transport that is terminating a block communication, or that is slewing, rewinding, or deleting and has its command gate closed, is in the BUSY-QUIET state. In this state as in STANDBY it ignores all signals except its own call, which takes it to BUSY-ALERT. Upon completing the operation a transport in BUSY-QUIET goes to STANDBY.

BUSY-ALERT. A tape transport that is terminating a block communication, or that is slewing, rewinding or deleting and has its command gate open, is in the BUSY-ALERT state. In this state it can answer calls and queries, but cannot execute general commands. A transport in BUSY-ALERT responds RED to query ready, QRD.

When any other unit is called, a transport in BUSY-ALERT goes to BUSY-QUIET and gives no answer. When a transport is deleting it answers calls and queries in the normal manner. Calls to other transports of the group will not be answered until the delete operation has been completed because delete operations use the circuitry of the tape control. Upon completing the operation a unit in BUSY-ALERT enters CALLED.

MESSAGE. During read or write, a tape transport is in the MESSAGE state from the time it receives SDT until the tape control has again become available for communication following the completion of the data transfer. Communication with any transports of the tape group is forbidden while a transport is in the MESSAGE state. Any communication on the line presently being used by the tape group is also forbidden.

**4.3–3 Special Features Of The MT–10** Any character read or written on the MT–10 is checked for parity. In order to lessen the likelihood of records being damaged due to programming errors or equipment malfunction, the MT–10 incorporates a write enable ring on tape reel. There is also an automatic halt on unrecorded tape, and two sensor channels either of which will halt the tape at a reflective marker. The latter are also useful for locating particular points such as end of file or end of record. A circuitry reset is provided for use in the event that one of the safety devices has halted the tape to prevent damage.

PARITY CHECK. As noted in Figure 4.3–2, a point on the tape passes the read head .39 inch after it passes the write head while moving forward. The read head is operative during both read and write operations. All information picked up by the read head during either type operation is checked for parity. In the event of a parity failure, the parity error flip-flop is set. When a parity error is detected during reading, the erroneous character is replaced by an error code.

CIRCUITRY RESET. Certain situations can arise, usually as a consequence of programming errors, in which the logical circuitry is ''hung up'' due to an arbitrary halt executed by one of the protective devices of the MT–10. In order to reset the circuitry to the STANDBY state, a special command is provided. This command takes the form of a calling signal directed to the control unit of the affected tape group.

WRITE ENABLE RING. The write enable ring is provided for protecting records from unintentional destruction. Unless the ring is present, the tape cannot be initially written, rewritten, or deleted. An attempt to execute one of these operations will be unsuccessful and the tape will answer RED. Before normal operation can be resumed the tape transport must be called to return the unit to the CALLED state, or the tape control can be called to reset the circuitry.

UNRECORDED TAPE HALT. In order to prevent tape run-away, unrecorded tape halt circuitry stops the mechanism whenever 1.4 ± .2 seconds have elapsed with tape in motion and no block marks have been detected. This is equal to about 150 inches of tape at low speed or about 300 inches of tape at high speed. Unrecorded tape halt is effective during any operation except rewind. When the tape has been stopped by the unrecorded tape detector during reading, writing or deleting, the MT–10 circuitry should be reset by sending QUT to the transport, and calling the tape control. No further tape motion is possible until the circuitry has been reset. To reset circuitry after an unrecorded tape halt occurring during slew it is necessary to send QUT, but not necessary to call the tape control.

REFLECTIVE SENSORS. In order to identify special points on magnetic tape such as beginning and end, strips of reflective material may be placed on the back of the tape. The reflective areas are detected by sensors located 4 inches from the read/write heads and having the dimensions shown in Figure 4.3–2. The sensor nearest the edge of the tape is designated as the beginning of tape sensor, and the other as the the end of tape sensor. At any time that a reflective area or a clear tape leader is entered upon from a non-reflective area, the tape will be halted automatically within 1/4 inch. Entering a non-reflective area from a reflective area will not halt the tape. If tape halt occurs during initial writing, the block will be garbled and should be deleted. If tape halt occurs during read or delete, it is evidence of a programming error. In either event the MT–10 circuitry should first be reset by calling the tape control. Moving the tape before the circuitry is reset will result in garbling the tape. Whenever the tape is halted by either or both of these sensors, the corresponding indicators will be set. These indicators are interrogated by the queries QBT and QET.

COMMUNICATION LINE SWITCH. When provided with a command-controlled switch, the tape control can be switched between two lines designated as line 1 and line 2. Switching may be accomplished by calling any of the transports in the group and giving the command to switch. When the circuitry is reset using the ZERO MACHINE switch, the tape control is placed on line 1.

**4.3–4 Functions Of The MT–10 Magnetic Tape** The MT–10 can position itself forward or backward a specified number of blocks without stopping. It can write a block complete with block marks [initial write], write in a block designated by existing block marks [rewrite], or read any block. These operations take place in the forward direction only and do not require halting between blocks. Tape need not be erased before initial write. Tape can be deleted, both data and block mark channel, in single blocks only, and in the backward direction only.

One possible method of using the MT–10 is to set up an addressable tape library with fixed blocks, *rewriting* the information when required. An entire reel of tape is initially recorded in blocks of standard size and any block which fails the write parity check is deleted. All blocks are then permanently numbered. Block length on initial write should be 5% longer than nominal to allow for normal variations in recording density.

Another method of using the MT–10 is to *initial write* at all times. Information is revised or updated by reproducing an entire file on another reel. When the information on a reel is no longer of value, the tape may be reused merely by initial writing over the old information. In using this method bear in mind that the new writing may begin or end in the middle of one of the old blocks.

It may be helpful to refer to Figure 4.3–4 in tracing the changes of state which the tape undergoes during normal operation.

SLEW AND REWIND. Hardware is provided in the MT–10 system to permit high-speed non-stop tape positioning a specified number of blocks. Except for the commands necessary to initiate the operation, this tape positioning can take place simultaneously with reading or writing on another tape transport of the same group. A 12–stage counter in each tape transport enables the tape to count forward or backward any number of blocks up to 4,096. Slewing 4,096 or more blocks requires a repetition of the slew command. In order to implement this system, the current tape address of each transport may be stored by the service routine, which then handles the addressing arithmetic.

Slewing is accomplished by giving [one or] two numeric commands [1nn]. The direction of slew will depend on the state of the direction flip-flop, which is set or reset by BAC or FWD. [The commands IWR, RCV, TRA leave this flip-flop in the forward [reset] state]. The first numeric sets the most significant 6 bits of the counter and starts the tape moving. The last 6 bits may be set by sending a second numeric, which if used should be supplied within 500 microseconds. As soon as the contents of the counter is non-zero the tape goes from CALLED to BUSY-ALERT and begins movement. Upon receiving the first numeric, the transport in the CALLED state enters BUSY-ALERT. If some other unit is called before completion of slew, the transport which is slewing enters BUSY-QUIET. At completion of slew, the transport if in BUSY-ALERT returns to CALLED, and if in BUSY-QUIET, returns to STANDBY. In either case the tape sends an interrupt at completion of slew. Slew takes place at a speed dependent on the number of blocks to be slewed. If more than 511 blocks are to be slewed, slew begins at high speed [220 inches per second]. The last 511 blocks are traversed at low speed [110 inches per second]. A slew of 511 or fewer blocks takes place at low speed.

Rewind of the MT–10 is accomplished by giving the command rewind [REW]. The tape enters BUSY-ALERT and moves backward at high speed. If some other unit is called, the transport which is rewinding enters BUSY-QUIET. Upon encountering a beginning or end of tape reflective marker, or a transparent tape leader, the tape halts, and after a 1.5 second delay, sends an interrupt. For programming purposes the tape is regarded as still slewing until the end of the 1.5 second delay. The reason for the delay is to permit recovery of the servo circuitry which was halted at high speed. The transport if in BUSY-ALERT then returns to CALLED, and if in BUSY-QUIET returns to STANDBY.

READ TAPE. Reading is accomplished in the MT–10 by giving the command transmit [TRA], which places the MT–10 in the INSTRUCTED state. This must be followed not more than 2.9 milliseconds later by start data transfer [SDT] and a request for character [REQ]. SDT places the unit in the MESSAGE state. The controlling device of the G–20 System may carry out any internal commands or instruct other equipment outside the tape group while the MT–10 is in the INSTRUCTED state. When the cue character of the preamble [Figure 4.3–2] has been detected by the read head, the next character, which is the first character of the block, is transmitted on the line. Subsequent characters are sent at intervals of approximately 8.3 microseconds, provided the necessary REQ signals have been received. Upon encountering the cue character of the postscript, the MT–10 sends an END code and enters BUSY-QUIET.

If a new call and read command are received after the END code but not later than .9 millisecond after the end of block mark is detected there will be no tape halt, and reading will continue as soon as the starting cue character of the next block is reached. The time between the end of block mark and the starting cue character is 5.5 to 6.3 milliseconds. This time may be used for computation or instructing other equipment.

If the new call and read commands are not received within 1.0 millisecond after END code, the tape will return to STANDBY and halt. The tape control unit will become available for use by other tapes of the group not later than 1.3 milliseconds after the end of block mark.

The parity of each character read is checked before the character is transmitted. Any time that a parity error is detected the erroneous character is replaced by an error code [ERR]. Transmission of data halts, but tape motion continues to the end of the block. The error flip-flop is set.

It is possible to terminate the transmission of a block before all the recorded information has been read. The receiving device can terminate transmission by end code [END] or an error code [ERR] at any time. In the event that the receiving device sends an end or error code before the end of the recording has been reached, transmission is halted but the parity check for the balance of the block is completed and the error flip-flop set if there was a parity error, even in the part of the block not transmitted.

INITIAL WRITING AND REWRITING. Before any writing can take place on the MT–10, the write enable ring must be in place on the back side of the feed reel [left]. The command initial write [IWR] or write [RCV] causes a tape transport in the CALLED state to enter the INSTRUCTED state and begin movement. Either write command must be followed in not more than 8.4 milliseconds by start data transfer [SDT]. In the event of initial write [which may take place over old information] the tape transport erases for .64 inch beyond the end of the last previous block and lays down a starting block mark. It records also a preamble and a starting cue character, and sends a request [REQ]. The transmitting unit must supply the first character within 5 microseconds. Additional REQ signals will be sent at 8.3 microsecond intervals until an end code [END] or an error code [ERR] is received by the tape transport. At this time the MT–10 will go to BUSY-QUIET, record a stopping cue character, a postscript, and an end of block mark.

The tape continues movement until the end of the recording has passed under the read heads [approximately 3.5 milliseconds]. An interrupt is sent as the read head reads the cue character. The parity of all characters is checked by the read head, and if any errors are found, the error flip-flop is set. Unless this flip-flop is in the reset state the tape will show RED to a query ready [QRD]. Interrogation QER resets the flip-flop.

Nine-tenths of a millisecond after the read head has read the end of block, the tape transport will go to STANDBY and halt if a new call and initial write command have not been received since the end code. In the event that the new call and initial write are received within 1.0 millisecond from the interrupt, writing can continue. The SDT signal for the new block must be received within 1.7 milliseconds after the read head reads the end block mark of the last block. In continuous initial write, a total of 5.5 milliseconds is available between blocks for computation or instructing other units.

Rewriting is similar to initial writing except for minor points. The tape transport in this case senses the existing block mark instead of writing a block mark. The recording of characters follows the same pattern except that the new block may now be shorter [but not longer] than the old block. Following the END or ERR code the transport goes to BUSY-QUIET and a stopping cue character and a postscript are recorded, and the balance of the block, out to the existing end of block mark, is erased. As in initial write, the error flip-flop is set if any parity errors are detected. The error flip-flop is also set if the block being recorded will not fit in the existing block. The excess characters are lost. Halting is similar to initial write.

DELETE. This command is provided for use in tape checking and permits block marks and data to be erased from a defective section of tape. The write enable ring must be present to permit deletion of a block. If it is absent, the unit will respond RED to delete [DBM]. Delete causes the tape to enter the BUSY-ALERT state and move backward erasing all marks in both the block mark channel and the data channels for the duration of the block. Calls to other transports of a tape group will not be answered until a delete operation in progress is completed.

When an initial write operation has been stopped *during data transfer* by an unrecorded tape halt or a reflective marker, the garbled block must be deleted as follows: (1) Call the tape control which resets the circuitry, (2) Using any suitable timing program slew the tape forward for 6.0 to 6.2 milliseconds [to reposition the heads], (3) Give a delete command.

**4.3–5 Manual Controls And Indicators** There are three push buttons on the MT–10 transport of interest to the user: OUT of Service [red], ON-LINE [green] and REWIND [white]. On the tape control there is an ON/SLAVE/OFF switch, a POWER ON light [red], and a ZERO MACHINE lever switch. These are under the flip-top on the MT–10 (A) and MT–10 (B).

Pressing the OUT button places the unit OUT of Service and illuminates the button. All operations will halt. If tape motion was involved at the time of pressing the button, the ZERO MACHINE switch must be operated, or the tape control called, before any further operations are executed in the tape group.

Provided that there is no interlock condition, pressing the ON-LINE button places the tape transport in the STANDBY state and illuminates the ON-LINE button.

If the unit is OUT of Service and there is no interlock condition, pressing the REWIND button will cause the tape to rewind to a reflective marker [or to clear leader].

A WRITE ENABLE lamp [yellow] is illuminated if the write enable ring is present on the feed reel.

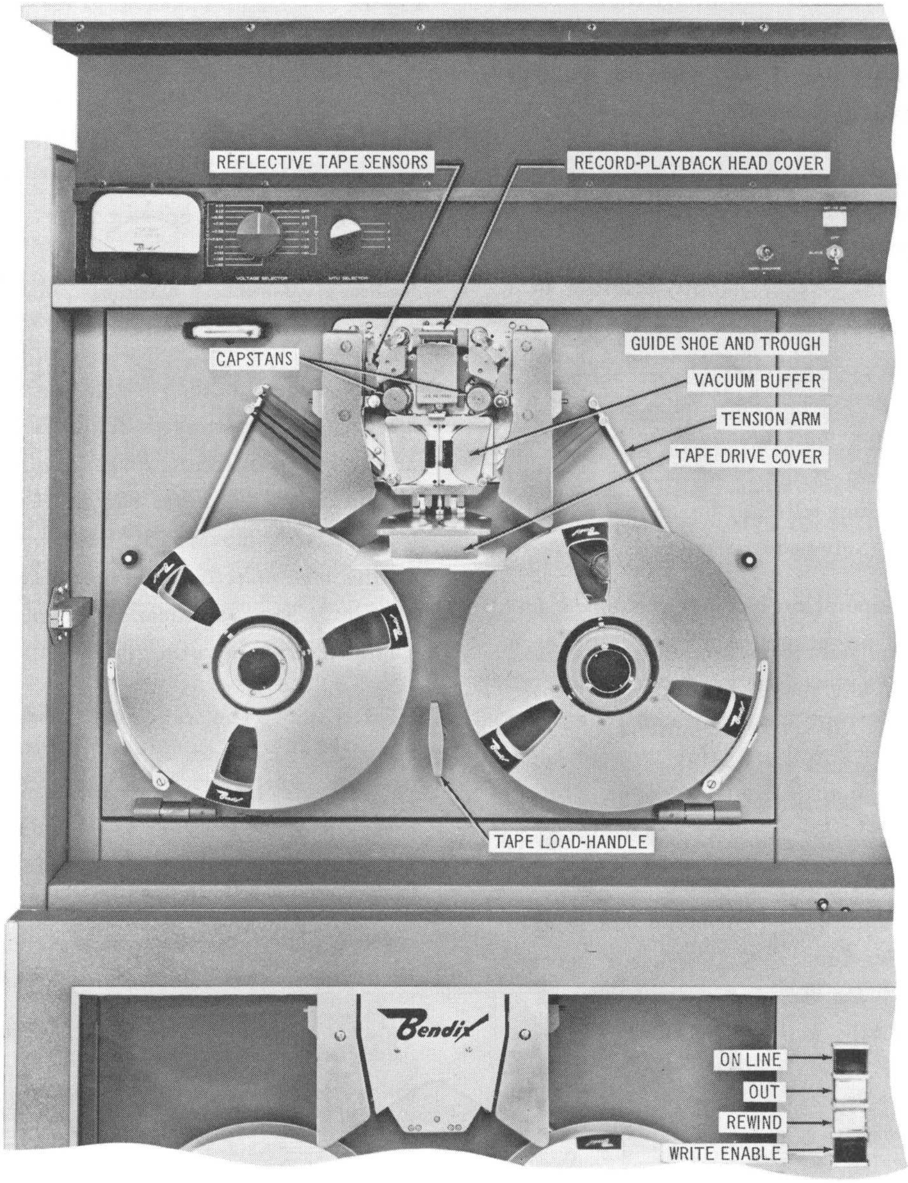
Power to the G–20 system is remotely controllable by means of a 24–volt D.C. signal sent on the communication line. The three positions of the ON/SLAVE/OFF power switch make the Magnetic Tape unconditionally on, or slave and dependent on the 24 volt D.C. signal, or unconditionally off, respectively. Upon application of power the tape transport enters the STANDBY state provided that there are no open interlocks. A 5–second warm-up period is required during which the unit is not operable.

**4.3–6 Tape Loading** In order to take maximum advantage of the design features built into the MT–10 and to avoid difficulty, users should follow the recommendations given here. The oxide-coated tape supplied for use with the MT–10 has characteristics matched to the mechanical and electronic design of the tape handler. The performance of the system cannot be guaranteed if any other tape is used.

LEADERS. The tape supplied for the MT–10 has clear plastic leader approximately 5 feet long at each end. Figure 4.3–6 shows the relative locations of the reels, leader, reflective sensors, and read/write heads when the tape is rewound to the physical end of the oxide. Note that the sensors which sense reflective markers [and will also sense clear leader] are four inches into the oxide-coated tape. In order to avoid any tape damage or defect that may have been created in affixing the reflective marker, it is recommended that the first actual tape block be placed approximately 80 inches beyond the reflective marker. For proper operation it is also necessary that the area from beginning of tape marker to the first block be free of block marks.

CASE 1. To start a new or degaussed reel which has a reflective marker 4–8 inches from the end, rewind, slew forward 80 inches using a timing program, give a stop command, and begin initial write.

FIGURE 4.3-5 CONTROL PANEL AND DETAII.S OF THE MT-10 (B OR C).



CASE 2. In the case of an undegaussed reel having a reflective marker, degauss and proceed as in Case 1, or proceed as follows: Rewind, slew forward one block [which should be about 80 inches], check for unrecorded tape halt, and begin initial write. If the tape halts in substantially less than 80 inches, remove the marker and proceed as in Case 3. If the unrecorded tape sensor halts the tape, proceed as for new tape.

CASE 3. For a tape that is not degaussed and has no marker, attach a marker, degauss, and proceed as for new tape, or proceed as follows: Manually position the tape so that the heads are on the beginning of the coated portion of the tape, initial write for approximately 80 inches, delete all but the last block written, affix a reflective marker 4–8 inches from the end, and proceed as in Case 2.

LOADING. In order to load the MT–10 proceed as follows: See Figure 4.3–7.

1. Select a reel of rewound tape and check that it has a beginning of tape reflective marker 4–8 inches from the end.
2. If the tape is to be written on or deleted, affix a write enable ring to the reel.
3. Mount the rewound reel on the left pivot so the free end of tape hangs on the right side of the reel.
4. Mount an empty reel on the right pivot.
5. Retract the tension arms by turning the tape load handle 180 degrees clockwise.
6. Lower the tape drive cover, and raise the record-playback head cover. Thread the tape as shown in Figure 4.3–7. Manually wind leader onto take-up reel until beginning of tape marker is past sensor. Friction will hold the tape to the coated hub. Do not use any other fastening.
7. Release the tension arms by turning the tape load handle counter clockwise. Close the recore-playback head cover and the tape drive cover.
8. Close door.
9. Press the ON-LINE button.

The first command to the tape should be backward slew or rewind.

**4.3–7 Programming Precautions** Certain types of programming errors should be guarded against in programming for the MT–10.

In a slew command the second numeric if sent, must follow the first within .5 millisecond. Both numerics should be transmitted from the G–20 in the same block command to avoid the possibility of an interrupt coming between them.

Caution should be exercised in the use of repeated rewind commands. A rewind command when the tape is at the first reflective marker will rewind to clear leader, and a second rewind command will strip the tape from the reel. [Interlock protection will take the tape OUT of Service.]

In the execution of a read or write command, it is essential that SDT and REQ signals reach the tape unit before the tape has reached the position to read or write. If the SDT and/or REQ are not received, data transmission is inhibited with no warning to the programmer. In the case of read, the tape will move to the end of block and halt, but no data will be transmitted. In the case of initial write, or write, a zero length block will be laid down complete with preamble and postscript.

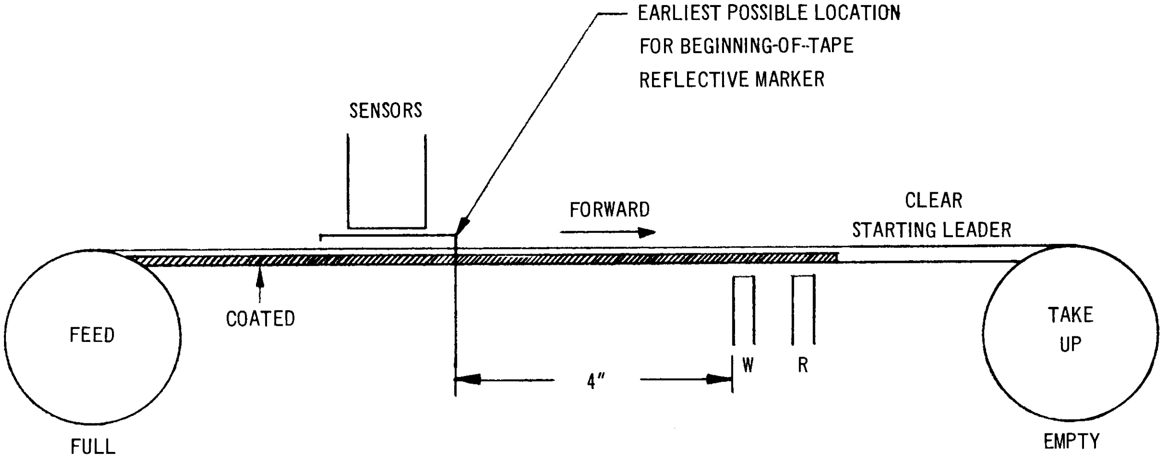
When a tape transport has been instructed and is awaiting SDT, a call to any tape of the group will inhibit transmission in the same way as failure to receive SDT.

In a rewrite operation if the data being recorded cannot all fit in the block being written in, the excess characters will be lost and the error flip-flop set.

An undefined command to the tape may cause it to "hang up". It can be recovered by operating the ZERO MACHINE switch.

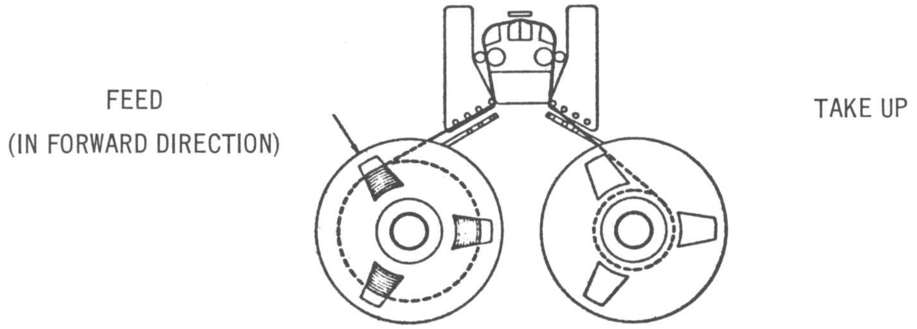
INTERRUPTS. Since the handling of interrupts is an important part of programming for complex systems, the situations in which interrupts are sent are reiterated here.

## FIGURE 4.3-6 LOCATION OF TAPE MARKERS

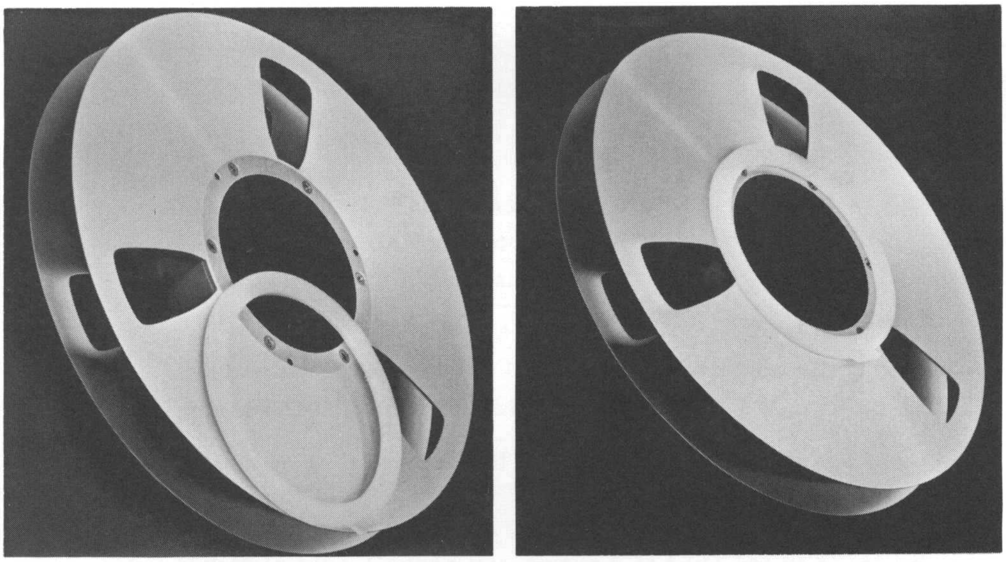


## FIGURE 4.3–7 LOADING THE MT–10

TAPE THREADING



"A" WIND (TAPE WOUND ON REEL WITH OXIDE SURFACE FACING IN)



WRITE ENABLE RING

1. At the endof a write operation as the stopping cue character passes the read head at completion of write check.
2. At the completion of a slew or rewind operation [unless the unit is OUT of Service at completion].
3. Whenever the tape is halted by a reflective marker or the unrecorded tape detector.
4. At the completion of a delete operation.

When a parity error is detected in a write check, the interrupt is sent immediately, not at the end of the block.

When information being rewritten cannot all fit in the assigned block space, the interrupt is sent at the time the end of the assigned block space is reached.

In the circumstance that the tape is halted from high speed, the interrupt is delayed 1.5 seconds to permit servo recovery.

During a continue read or write operation no calls should be directed to other transports of the group, even by the interrupt service routine. Failure to observe this precaution will result in stop/start operation. See CALs in command list.

**4.3–8 Command List** Table 4.3–2 lists the line signals which can be transmitted or received by a Magnetic Tape.

## TABLE 4.3–2 MAGNETIC TAPE COMMAND CODES

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 002 | GRN | Green. This is a reply the tape transport gives to commands and queries. |
| 003 | RED | This is a reply the tape transport gives to commands and queries. |
| 004 | END | End of block. Tape transport in transmit MESSAGE state sends END as last character of block and enters BUSY-QUIET state.<br>Control device can send END to terminate reading before end of recording is reached.<br>Control device sends END at end of write. Tape sends interrupt.<br>END never requires a reply. |
| 005 | ERR | Error. Sent by tape transport in lieu of a character with incorrect parity detected during reading. Tape goes from MESSAGE state to BUSY-QUIET and sends interrupt.<br>Control device can send ERR to terminate reading before end of block is reached.<br>Nine-tenths of a millisecond after the end of the block, tape will return to STANDBY and halt if new call and read command have not been received.<br>ERR never requires a reply. |
| 006 | SW1 | Switch to Line 1. Tape transport in CALLED [or BUSY-ALERT] state and on Line 2 answers GRN and enters STANDBY [or BUSY-QUIET. ] Entire tape group switches to Line 1, and is OUT of Service for 5–7 milliseconds during switching.<br>If group is already on Line 1, no reply or switching will occur and unit will enter STANDBY if in CALLED, or BUSY-QUIET if in BUSY-ALERT.<br>At turn-on or at operation of ZERO MACHINE switch, the line switch is connected to Line 1. |
| 007 | SW2 | Switch to Line 2. Tape transport in CALLED [or BUSY-ALERT] state and on Line 1 answers GRN and enters STANDBY [or BUSY-QUIET]. Entire tape group switches to Line 2, and is OUT of Service for 5–7 milliseconds during switching.<br>If group is already on Line 2, no reply or action will occur but unit will go to STANDBY if in CALLED, or BUSY-QUIET if in BUSY-ALERT. |
| 010 | SDT | Start Data Transfer. Tape transport in INSTRUCTED state enters MESSAGE state. If instructed to receive, sends REQ. If instructed to transmit, gives no reply. The line will be tied up for the duration of the block. |

| TABLE 4.3–2 MAGNETIC TAPE COMMAND CODES (Continued) | | |
|---|---|---|
| OCTAL CODE | ALPHA CODE | DESCRIPTION |
| 011 | OUT | Go OUT of Service. Tape transport in CALLED or BUSY-ALERT answers GRN and disconnects itself from the line. A slew or rewind in progress will be completed. OUT should not be given if any other operation is in progress in the group. Garbling of records will result. The tape cannot be returned to service until the ON-LINE button is depressed. |
| 014 | RCV | Receive. A command to write. Tape transport in CALLED state answers GRN, enters INSTRUCTED state and begins movement if write enable ring is present. Interrupts when finished.<br>If ring is absent, unit will answer RED and enter INSTRUCTED state, but will not move. A call to the transport will return it to CALLED state.<br>After a write operation the direction flip-flop is left in the reset [forward] state. |
| 016 | TRA | Transmit. A command to read. Tape transport in CALLED state answers GRN, enters INSTRUCTED state and begins movement.<br>After a read operation the direction flip-flop is left in the reset [forward] state. |
| 044 | FWD | Forward. Tape transport in CALLED state answers GRN and resets the direction flip-flop. A slew executed now will be in the forward direction. Any other action will be unaffected. |
| 045 | BAC | Backward. Tape transport in CALLED state answers GRN and sets the direction flip-flop. A slew command executed now will be in the backward direction. Any other action will be unaffected. |
| 046 | IWR | Initial Write. Write and supply block marks. A tape transport in CALLED state enters INSTRUCTED state and begins movement if write enable ring is present. Interrupts when finished. If ring is absent, unit will answer RED and enter INSTRUCTED state, but will not move. A call to the transport will return it to CALLED state.<br>After an initial write operation the direction flip-flop is left in the reset [forward] state. |
| 047 | REW | Rewind. Tape transport in CALLED state answers GRN, enters BUSY-ALERT and moves backward at high speed. Tape will halt if a beginning or end of tape marker is reached or if clear leader is reached. One and one half seconds after halt the tape returns to CALLED if in BUSY-ALERT, or to STANDBY if in BUSY-QUIET, and sends an interrupt. |

**TABLE 4.3-2 MAGNETIC TAPE COMMAND CODES (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 050 | STP | Stop. Tape transport in CALLED or BUSY-ALERT answers GRN and halts unconditionally.<br>If slewing at low speed, returns to CALLED in 8 milliseconds and sends interrupt. If slewing at high speed or rewinding, returns to CALLED.in 1.5 seconds and sends interrupt.<br>In all other cases returns to CALLED immediately. The circuitry should be reset with CALc.<br>If on unrecorded tape, next action can be IWR. Otherwise, next action must be slew or rewind. |
| 056 | DBM | Delete. Tape transport in CALLED state with write enable ring present answers GRN, enters BUSY-ALERT and deletes data and block marks for one block in the backward direction. Upon completion, sends interrupt and enters CALLED state.<br>During delete the other tapes of the group are not available. If one of them is called during this period, the reply will be delayed until completion of the delete.<br>A tape in CALLED state with write enable ring absent replies RED to DBM.<br>After a delete operation the direction flip-flop is left in the reset [forward] state. |
| 060 | QRD | Query Ready. Tape transport in CALLED state answers GRN if ready, RED if not ready. Tape in BUSY-ALERT answers RED. Tape is not ready if any tape motion is in progress, or if it has been halted by a reflective marker, or if the unrecorded tape flip-flop is set, or the error flip-flop is set. Most operations are impossible if tape is not ready; an exception is continuous read or write. Also TRA, RCV, DBM or slew [but not IWR] can be executed when on a reflective marker. |
| 061 | QER | Query no parity Error. In most cases tape transport in CALLED or BUSY-ALERT state answers GRN if the error flip-flop is in the reset state and RED if the flip-flop is in the set state. QER resets the flip-flop.<br>In the particular case that the tape is traversing the balance of a block the reading of which was terminated by command, a RED answer is immediate if the flip-flop is in the set state, but a GRN answer is deferred until completion of the block and signifies that the *entire* block is free of error. |
| 063 | QUT | Query not on Unrecorded Tape. Tape transport in CALLED or BUSY-ALERT answers GRN if unrecorded tape flip-flop is in reset state, and RED if flip-flop is in set state. QUT resets the flip-flop. |

## TABLE 4.3-2 MAGNETIC TAPE COMMAND CODES (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 064 | QBT | Query not at Beginning of Tape. Tape transport in CALLED or BUSY-ALERT answers GRN if not stopped by beginning of tape sensor; otherwise, answers RED. |
| 065 | QET | Query not at End of Tape. Tape transport in CALLED or BUSY-ALERT answers GRN if not stopped by end of tape sensor; otherwise, answers RED. |
| 066 | QSL | Query not Slewing. Tape transport in CALLED or BUSY-ALERT answers GRN if not slewing or rewinding; otherwise, answers RED. |
| 067 | QIN | Query no Interrupt. Tape transport in CALLED or BUSY-ALERT answers GRN if interrupt flip-flop is in reset state; otherwise, answers RED. QIN resets the flip-flop. |
| 0XX | None | Any undefined command. May cause a tape transport in CALLED or BUSY-ALERT to "hang up". ZERO MACHINE switch can be operated to reset circuitry. |
| 1nn | 1nn | Command numeric. Used for slew operations. Tape transport in CALLED state answers GRN. Direction of slew will depend on state of direction flip-flop. See FWD, BAC.<br>The digits nn are loaded into the most significant 6 places of the block counter. When the contents of the counter are non-zero, the tape goes from CALLED to BUSY-ALERT and begins tape movement. If nn $\geq 10_8$ slew will start at high speed. If nn $= 01_8$ slew will start at low speed.<br>A second numeric following within .5 millisecond sets the right 6 bits of the block counter. Each block traversed decrements the block counter by one unit. When the counter contents is $< 1000_8$, transport assumes low speed. Tape halts when counter has been decremented to zero, and an interrupt is sent. [For a retrofitted model 1 tape transport the second numeric must be sent even if it is zero.] |
| 200 + uuu | CALs | A call to the tape transport itself. [$0 \leq uuu \leq 177_8$]. Tape transport in STANDBY [or CALLED] answers GRN and enters CALLED. Transport in BUSY-QUIET or BUSY-ALERT answers GRN and enters BUSY-ALERT. If the tape control is occupied due to a read, write, or delete operation in another transport of the group, the answer to the call is delayed until the operation is finished. |
| 200 + uuu | CALc | A call to a tape control. [$0 \leq uuu \leq 177_8$]. Answers GRN. Any transports of the group in BUSY-ALERT or INSTRUCTED will go to BUSY-QUIET. Any transports in CALLED will go to STANDBY. This command must not be directed to any group one of whose transports is in MESSAGE state. |

## TABLE 4.3-2 MAGNETIC TAPE COMMAND CODES (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 200 + uuu | CALo | A call to some other unit. $[0 \le uuu \le 177_8]$. No answer. Transport in CALLED enters STANDBY, transport in BUSY-ALERT enters BUSY-QUIET. A call must not be directed to a transport if another transport of the same group is in INSTRUCTED; the INSTRUCTED transport will go to BUSY-QUIET. |
| 400 + ddd | ddd | A data character. $[0 \le ddd \le 377_8]$. Sent by a transport in transmit MESSAGE state, in response to REQ. When received by a transport in receive MESSAGE state, is recorded and causes an REQ to be sent. |
| None | REQ | A request for a character. Sent on the request line by the device acting as receiver during block transmission or reception in MESSAGE state. An REQ is usually.answered by ddd, but can be answered by END or ERR when a block is terminated. |
| None | INT | An interrupt request sent by the MT-10 on the interrupt line. Used in some cases to signify that an operation is complete, or that there has been a tape halt. |

The following examples show a possible sequence of line signals for Slew, Read, Delete, and starting a new reel of tape. In the following examples an * indicates the portion of the Interrupt Service Routine used to establish the source of interrupts.

## EXAMPLE 4.3-1 SEQUENCE OF LINE SIGNALS - SLEW MAGNETIC TAPE

| G-20 | MT-10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport is available. |
| FWD<br>BAC | GRN | Sets direction flip-flop for forward or backward, omit if slewing forward. |
| 1nn | GRN | Sets left 6 bits of block counter. Transport enters BUSY-ALERT state. Begins motion as soon as counter contents are non-zero. |
| 1nn | GRN | Sets right 6 bits of block counter if required. |
| . . . . | . . . . | Tape continues slewing. Each block traversed decrements block counter. |
|  | INT | Slewing complete. |

## EXAMPLE 4.3-2 SEQUENCE OF LINE SIGNALS – WRITE ONE BLOCK OF MAGNETIC TAPE

| G-20 | MT-10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport is available. |
| IWR<br>RCV | GRN | Enters INSTRUCTED state and begins motion. |
| . . . . | . . . . | A maximum of 8.4 milliseconds delay. |
| SDT | REQ | Transport enters MESSAGE state. |
| ddd | REQ | Data Transfer |
| . . . . | . . . . | |
| ddd | REQ | |
| END | . . . . | Block terminated. Transport enters BUSY-QUIET, writes end of block mark if initial writing, or continues to end of block mark if rewriting. |
| . . . . | . . . . | Approximately 3.5 milliseconds. |
| | INT | Cue character read. |
| . . . . | . . . . | Transport waits until tape motion stops and enters STANDBY. |
| * CALs | GRN | Enters CALLED state. |
| * QIN | RED | Interrupt identified. |
| QER | GRN | No parity error on previous operation. |

## EXAMPLE 4.3-3 SEQUENCE OF LINE SIGNALS – CONTINUOUS WRITE OR REWRITE ON MAGNETIC TAPE

| G-20 | MT-10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport is available. |
| IWR<br>RCV | GRN | Enters INSTRUCTED state and begins motion. |
| . . . . | . . . . | Maximum of 8.4 milliseconds. |
| [A] SDT | REQ | Enters MESSAGE state. |
| ddd | REQ | Data Transfer. |
| . . . . | . . . . | |
| ddd | REQ | |
| END | | Block terminated. Enters BUSY-QUIET, writes end of block if initial writing, or continues to end of block mark if rewriting. |
| . . . . | . . . . | Approximately 3.5 milliseconds. |
| | INT | Cue character read. |
| . . . . | . . . . | . . . . |
| * CALs | GRN | Enters BUSY-ALERT state. |
| * QIN | RED | Interrupt identified. |
| QER | GRN | No parity error on previous operation. |
| . . . . | . . . . | Maximum of 1.0 milliseconds since interrupt. |
| IWR<br>RCV | GRN | Transport enters INSTRUCTED state. |
| . . . . | . . . . | Maximum of 5.5 milliseconds since interrupt. Continue from [A]. |

### EXAMPLE 4.3–4 SEQUENCE OF LINE SIGNALS – READ PARTIAL BLOCK ON MAGNETIC TAPE

| G–20 | MT–10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport available. |
| TRA | GRN | Enters INSTRUCTED state and begins motion. |
| . . . . | . . . . | 2.9 milliseconds maximum. |
| SDT | | Enters MESSAGE state. |
| REQ | ddd | |
| . . . . | . . . . | Data Transfer. |
| REQ | ddd | |
| END | | Terminates transmission. Transport enters BUSY-QUIET. |

The fastest return to the beginning of the block is given by the following commands:

| | | |
|---|---|---|
| CAL | GRN | Enters BUSY-ALERT state. |
| STP | GRN | Stop tape motion. |
| CALc | GRN | Resets circuitry. |
| . . . . | . . . . | . . . . |
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | |
| BAC | GRN | Returns to beginning of block. |
| 100 | GRN | |
| 101 | GRN | |
| | INT | Operation complete. |

### EXAMPLE 4.3–5 SEQUENCE OF LINE SIGNALS – READ ONE BLOCK OF MAGNETIC TAPE

| G–20 | MT–10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport is available. |
| TRA | GRN | Enters INSTRUCTED state and begins motion. |
| . . . . | . . . . | 2.9 milliseconds maximum. |
| SDT | | Enters MESSAGE state. |
| REQ | ddd | |
| . . . . | . . . . | Data Transfer. |
| REQ | ddd | |
| REQ | END | Transmission terminated. Transport remains in BUSY state until tape motion stops, then enters STANDBY state. |

## EXAMPLE 4.3-6 SEQUENCE OF LINE SIGNALS – CONTINUOUS READ MAGNETIC TAPE

| G-20 | MT-10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport available. |
| TRA | GRN | Enters INSTRUCTED state. |
| . . . . | . . . . | Maximum of 2.9 milliseconds delay. |
| [A] SDT | | Enters MESSAGE state. |
| REQ | ddd | Data Transfer. |
| . . . . | . . . . | |
| REQ | ddd | |
| END | [no reply] | Transmission terminated. Transport goes to BUSY-QUIET. |
| CALs | GRN | Transport enters BUSY-ALERT. |
| QER | GRN | No parity error. |
| TRA | GRN | Enters INSTRUCTED state without halting tape. |
| . . . . | . . . . | G-20 may return to other program for 5.5 milliseconds maximum.<br>Return to [A]. |

## EXAMPLE 4.3-7 SEQUENCE OF LINE SIGNALS – DELETE A BLOCK OF MAGNETIC TAPE

| G-20 | MT-10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport is available. |
| DBM | GRN | Delete previous block. |
| | INT | Delete operation complete. |

## EXAMPLE 4.3–8 SEQUENCE OF LINE SIGNALS – START NEW OR DEGAUSSED TAPE WITH REFLECTIVE MARKER IN PLACE [CASE 1]

| G–20 | MT–10 | REMARKS |
|---|---|---|
| CALs | GRN | Enters CALLED state. |
| QRD | GRN | Transport is available. |
| BAC | GRN | Backward. |
| 101 | GRN | Enters BUSY-ALERT. Starts backward at 110 inches/second. |
| [100 | GRN] | [Required on retrofitted Model 1 MT–10]. |
|  | INT | Has reached reflective marker, goes to CALLED state. |
| * CALs | GRN | Identifies interrupt. |
| * QIN | RED |  |
| QBT | RED | On beginning of tape. |
| FWD | GRN | Slew forward. |
| 110 | GRN | Enters BUSY-ALERT. Starts forward at 220 inches/second. |
| [100 | GRN] | [Required on retrofitted Model 1MT–10]. |
| . . . . | . . . . | Timing program for .35 seconds. |
| STP | INT | Transport is approximately 80 inches beyond reflective marker. |
| * CALs | GRN | Interrupt identified. |
| * QIN | RED |  |
| QRD | GRN | Transport available. |
| IWR | GRN | Begins writing dummy block. |

## EXAMPLE 4.3–9 SEQUENCE OF LINE SIGNALS – REUSE TAPE WITH REFLECTIVE MARKER IN PLACE [CASE 2]

| G–20 | MT–10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport enters CALLED state. |
| QRD | GRN | Transport is available. |
| BAC | GRN | Slew backward. |
| 101 | GRN | Enters BUSY-ALERT. Starts backward at 110 inches/second. |
| [100 | GRN] | [Required on retrofitted model 1]. |
|  | INT | On reflective marker. |
| * CALs | GRN | Identifies interrupt. |
| * QIN | RED |  |
| QBT | RED | On beginning of tape. |
| FWD | GRN | Forward slew one block. |
| 100 | GRN |  |
| 101 | GRN |  |
|  | INT | Slew complete. If substantially less than 80 inches, rewind, remove marker and proceed as Case 3. |
| * CALs | GRN | Identifies interrupt. |
| * QIN | GRN |  |
| QRD | GRN | May proceed as desired. |
|  | RED | Not available. |
| QUT | RED | Rewind and proceed as in Case 1. |

## EXAMPLE 4.3-10 SEQUENCE OF LINE SIGNALS — REUSE TAPE WITH NO REFLECTIVE MARKER [CASE 3]

Position tape so write head is on beginning of coated portion of tape.

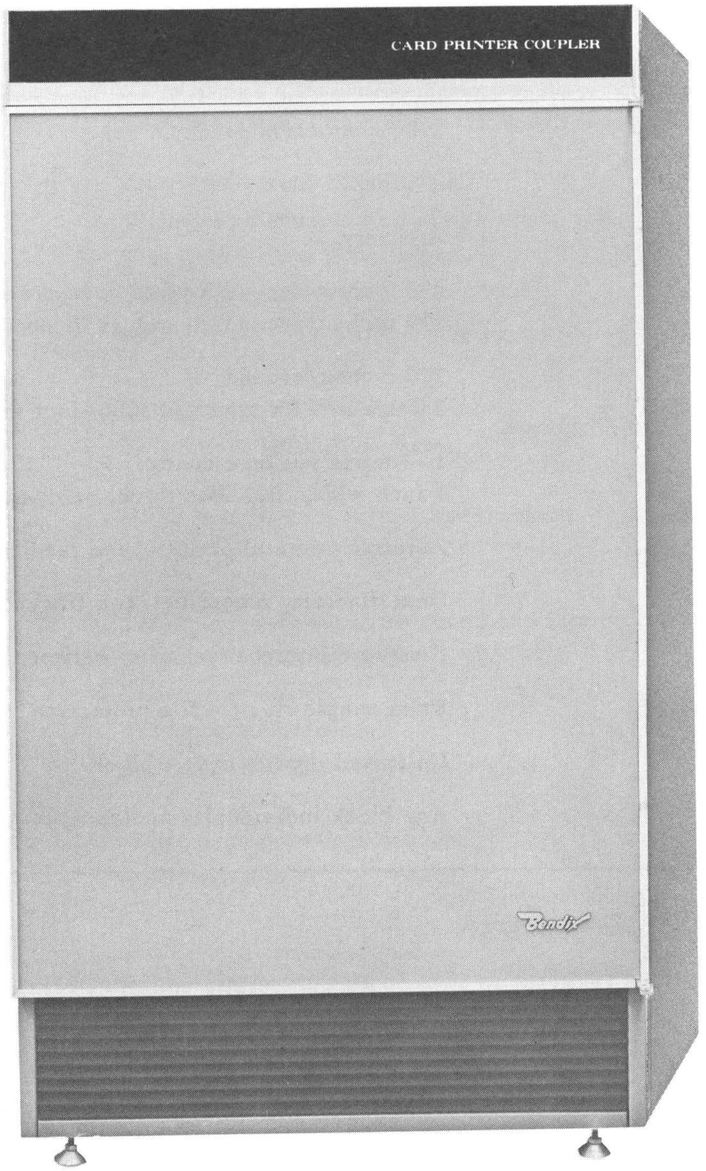
| G-20 | MT-10 | REMARKS |
|---|---|---|
| CALs | GRN | Transport in CALLED state. |
| QRD | GRN | Transport available. |
| [A] IWR | GRN | Initial Write. Transport enters INSTRUCTED state and starts tape motion. |
| | [RED | No write enable ring.] |
| SDT | REQ | Transport enters MESSAGE state. |
| ddd | REQ | G-20 dumps 4,096-word memory in 6-bit characters into a single block on Magnetic Tape. One memory dump will use approximately 19.3 inches of tape. |
| . . . . | . . . . | |
| ddd | REQ | |
| END | [no reply] | Transport enters BUSY-QUIET. |
| | INT | Transport interrupts at end of write check. |
| * CALs | GRN | Transport enters CALLED state. |
| * QIN | RED | Source of interrupt identified. |
| | | Repeat from [A] 3 times making a total of 4 blocks, or 80 inches. |
| IWR | GRN | Unit enters INSTRUCTED state. |
| SDT | REQ | Unit enters MESSAGE state. |
| ddd | REQ | G-20 lays down a normal length block. This will become a "dummy" block. |
| . . . . | . . . . | |
| ddd | REQ | |
| END | [no reply] | |
| | INT | Transport performs write check and interrupts. |
| * CALs | GRN | |
| * QIN | RED | Interrupt identified. |
| QRD | GRN | Transport available. |
| BAC | GRN | Sets direction flip-flop to backward. |
| 100 | GRN | |
| 101 | GRN | Slews back past dummy block. |
| | INT | Slew complete. |
| * CALs | GRN | |
| * QIN | RED | Source of interrupt identified. |
| [B] DBM | GRN | Delete Block. Transport goes to BUSY-ALERT state.<br>[G-20 may compute or communicate with units outside of tape group.] |
| | INT | Transport has deleted block. |
| * CALs | GRN | Transport enters CALLED state. |
| * QIN | RED | Source of interrupt identified. |
| QRD | GRN | Return to [B]. |
| | RED | |
| QBT | RED | Beginning of tape. Affix reflective marker 4 inches from end of transparent leader and proceed as in Case 2. |
| CALc | GRN | Clear circuitry. Halted by reflective marker. |

### 4.3–9 Summary Of Magnetic Tape Characteristics

**TABLE 4.3–2 MT–10 MAGNETIC TAPE CHARACTERISTICS**

| | |
|---|---|
| Read/write rate | 120,000 8–bit characters per second.<br>110 inches per second, forward only. |
| Interblock time | 6.2 milliseconds [compute 5.5 maximum].<br>For continuous read or write. |
| Start/Stop | |
| Start read | 4.1 milliseconds [compute 2.9 maximum]. |
| Stop read | 1.2 milliseconds [compute all]. |
| Start write | 8.0 milliseconds [compute 5.4 maximum]. |
| Stop write | 4.7 milliseconds [compute all]. |
| Recording density | 1100 characters per inch. |
| Interblock gap | .64 inch. |
| Block length | Arbitrary. |
| Search [non-stop] | 220 inches/second forward or backward. Last 511 blocks 110 inches/second. Search is independent of tape control. |
| Rewind | 220 inches/second.<br>3.3 minutes for standard 3600–foot reel. |
| Control | 1–4 tapes per tape control. |
| Tape | 1 inch wide, .001 inch thick, oxide-coated. |
| Other features | Automatic lateral parity check reading and writing.<br>Dual diversity recording [two tracks for each channel].<br>Complete interchangeability between units.<br>Write enable ring for file protection.<br>Unlimited rewrite over a block.<br>Any block individually addressable. |

FIGURE 4.4–1 CARD-PRINTER COUPLER, PC–10



## SECTION 4.4–CARD–PRINTER COUPLER, PC–10

The Card–Printer Coupler, PC–10, permits the reading of the cards on an IBM 088, 085, 514 or 528. It permits the punching of cards in Hollerith on an IBM 544, 528, 514 or 523 and row binary punching on IBM 544. It permits high-speed line printing on the Bendix LP–10 or LP–11, as well as low-speed line printing on an IBM 407. The PC–10 also permits the G–20 Computer to be initial-loaded using punched cards with one of the above-mentioned card readers. Any proposed use of the PC–10 with an accessory other than those listed, should be checked with the Bendix Applications Engineer before proceeding. The Card–Printer Coupler enables the following functions to be performed: read cards, verify reading, punch cards, verify punching, line print. The punch channel can also be used for tabulating. The unit is housed in a cabinet 64 inches high, 34 inches wide, 28 inches deep, and weighs 800 pounds. Power required is 1.5 KVA of 115-volt AC, 50 or 60 cycles, single-phase. See Figure 4.4–1.

**4.4–1 Organization Of The PC–10** As may be noted from the block diagram, Figure 4.4–2, the PC–10 consists of a central unit containing a line register and a distributor with the necessary controls and amplifiers, plus 5 adapters, each of which couples the PC–10 for one of the 5 functions.

No other function can be performed by the PC–10 during high-speed line printing, but time-sharing is possible on some card operations. However, at any given moment the PC–10 is handling one function only. It is possible to perform two or more card operations simultaneously provided that the time required to process a row of information for each function successively is appreciably less than the allowed processing interval for the function having the shorter available processing interval.

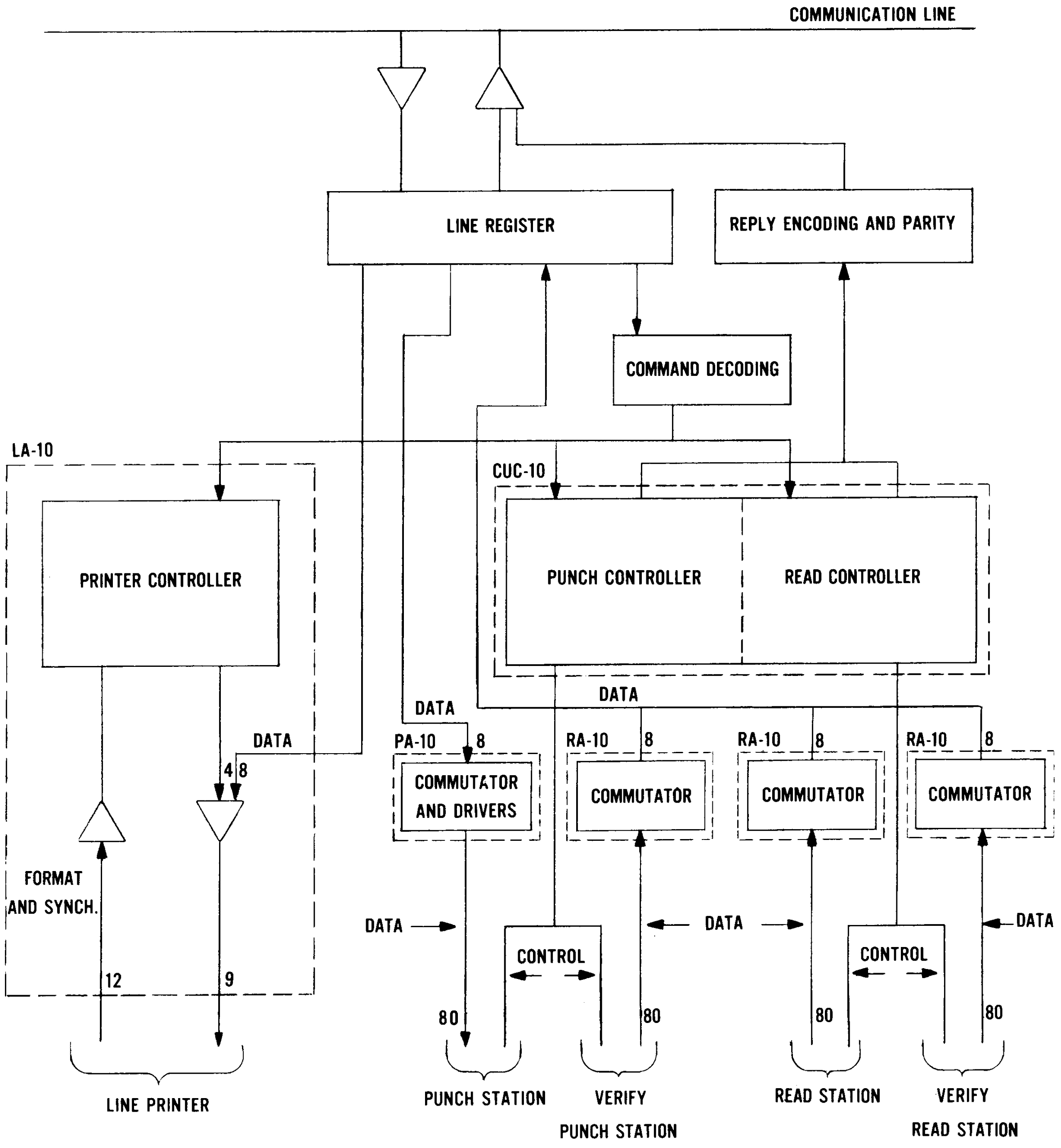
All data transfer to and from the PC–10 takes places in a row-wise fashion. That is, 12 bursts of information must be exchanged between a card reader and a PC–10, or 63 bursts of information between a high-speed Line Printer and a PC–10, etc., and any data to be put out via the PC–10 must be arranged in this row binary fashion for transmission to the PC–10. Likewise, information read in through the PC–10 may need to be translated before it can be used elsewhere in the system. Reading or punching of row binary cards requires no translation. All other operations on the PC–10 will require conversion to be made to or from row binary.

The commutators shown in the block diagram perform the function of packing 8–bit characters from the G–20 communication line into a row of the appropriate length for use with the card equipment and for unpacking the same row for the reverse operation. In card punching, for example, ten characters are required for a full row on the card. This will require three G–20 words, the last of which carries two characters which are ignored by the PC–10. The PC–10 is able to sense completion of one row of card information during reading, but has no other counting ability.

The PC–10 can exist in the following states: OUT of Service, STANDBY, CALLED, INSTRUCTED, or MESSAGE, described in Chapter 3, plus one special state. This state is called INITIAL LOAD and is a modified MESSAGE state permitting the initial loading of the computer's memory via cards. A manually operated switch places the PC–10 in the INITIAL LOAD state. A PC–10 which is on-line will unconditionally answer GRN to any of the 5 calls assigned to it and will remember the number by which it was called in order to address future instructions to the proper unit. Most commands to the PC–10 are addressable to a particular unit, while certain other commands are handled by common hardware. There is a single ERROR flip-flop and a single interrupt flip-flop for the PC–10. Also a single command clears both of them. Other indicators refer to a particular unit.

**4.4–2 Card Reading** Card reading on the PC–10 is in the row binary mode. Each time the card is in a position where one of the rows can be read by the brushes, an interrupt signal will be sent to the G–20. At this time the G–20 will process the interrupt and, upon determining that it is the read station attached to the PC–10 which requires service, will direct the PC–10 to deliver the block of data to the computer. The information from columns 1–8 becomes the first

FIGURE 4.4–2 SIMPLIFIED BLOCK DIAGRAM, CARD-PRINTER COUPLER, PC–10



Note: Figures indicate number of bits

character transmitted, column 1 being the most significant digit, etc. The G–20 must either keep track of the rows or query each time to determine whether the last row of the card had been reached.

When the PC–10 is placed in the INITIAL LOAD state, one entire card is read into the G–20 starting in location 64 and running through location 93. The manner in which the information from the card is loaded into the G–20 is shown in Figure 4.4–3. The computer will then begin executing the command in location 65. Since one card can not contain more than 30 words, the initial loading program must be so constructed as to permit the starting of a routine to read in further information, using not more than 30 words. When the first card has been read in, the PC–10 will revert to the STANDBY state.

**4.4–3 Card Punching** As with card reading, card punching in the PC–10 is done in row binary only. An interrupt is sent to the G–20 each time the card is in the punch position. After processing the interrupt, the G–20 will send to the PC–10, 1–4 words of binary information for one row.

The first word will be placed in columns 1–32, the second word in columns 33–64. The third word sent will have its first two characters placed in columns 65–80, and the balance of the transmission will be ignored. If more than four words are sent, the fifth word will be stored on top of the first [actually united] causing garbled information. The transmission may consist of one word, or two words, if it is not desired to punch all of the card. Thus, one may punch the first 32 columns, the first 64 columns, or the entire card.

**4.4–4 Line Printing** Before line printing can be accomplished with the PC–10, the information to be printed must be formed into a print image. The number of rows in the image will depend on the characters to be printed, e.g., 32 rows for numeric information. The number of columns required will equal the number of characters in the image. This image will contain in each column position of its nth row a "1" wherever the nth character on the print roll is to appear on the copy at that column. One row of this image will be read into the printer at each print roll position 0–62 of the print roll. A space is produced in the copy by having an entire blank column in the print image. [Information must not be sent to the LP–10 or LP–11 corresponding to "space", row 65 of these printers.] The first character transmitted governs the operation of the first 8 print hammers at the left, etc. Information need not be delivered for blank columns at the right of the format. The printing sequence for each line of information is initiated by an index pulse interrupt which occurs at print roll position 63.

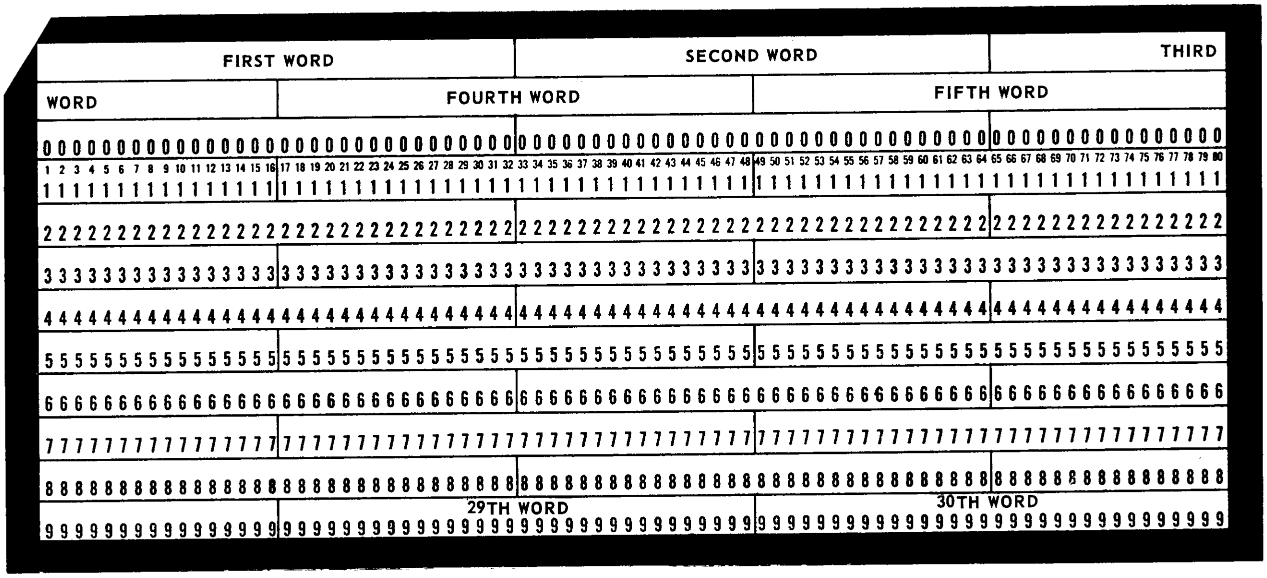
Paper advance in the printer can be accomplished a line at a time by command, or it can be controlled by a paper tape format loop. Thus, it is possible to program arbitrary formats as well as to use any of the eight formats on the format loop.

Line printing can be done on the LP–10 and LP–11 in three modes: alphanumeric, numeric, and alphabetic. The print roll is laid out with letters in odd row positions and numbers in even row positions, and the printer can be operated printing odd rows only, even rows only, or all rows. To select one of these modes, the program must give a command to print alphanumeric, print numeric, or print alphabetic. The printing mode will remain as set until changed by a subsequent command, or until a card operation is begun.

Because the PC–10 is merely a coupler and does not contain data-editing capability, the Central Processor must construct the printer image and transmit it a line at a time to the PC–10. For this reason, the maximum printing speed available using the PC–10 can in some instances be limited by the operations taking place in the G–20. For full alphanumeric printing, the printer is delayed after each line while the Central Processor constructs the printer image for the next line. The over-all speed for alphanumeric printing is approximately 450 lines per minute for either the LP–10 [72 columns] or the LP–11 [120 columns].

In numeric printing on the LP–10 or LP–11, the characters to be used occupy only about 50 percent of the circumference of the print roll; the unused time

## FIGURE 4.4-3 FORMAT OF INITIAL LOAD CARD



is available for editing. With the LP–10, numeric printing can take place at 900 lines per minute, and with the LP–11, at 1,000 lines per minute. With the more elaborate CB–11 accessory, or the buffered printer LP–12, which remove much of the burden from the Central Processor, line printing can be accomplished at speeds limited only by the printer.

In printing with the IBM 407 accounting machine, the punch channel of the PC–10 is used to couple to the tabulator. The information to be printed must be formed into a tabulator image in Hollerith code as this is information required for the 407. The tabulator image is read into the tabulator in row binary fashion at the appropriate point in the mechanical cycle of the tabulator. This point is signalled by an interrupt to the computer.

**4.4–5 Special Functions** The PC–10 can energize certain leads with card machine [SW +] voltage under program control to permit selectors to be picked up for special functions. There are two such leads to the punch feed station available [in parallel] in the PUNCH cable and the PUNCH VERIFY cable of the PC–10, and two leads similarly available in the READ and READ VERIFY cables. These leads are ordinarily used to control alternate stackers and are labeled AS1 and AS2. When not being used for this function they can be used to energize any hub on the control panel of the card machine. At the time of the line command the lead is energized and remains so for the duration of the card cycle.

Ordinarily, the digit impulse hub is connected directly to the PC–10 cable and causes the PC–10 to send an interrupt signal to the G–20 each time a card is in position to be processed. It is possible to make the sending of the interrupt dependent on information sensed in the card at the X brush station which is ahead of the first regular station. This may be the case if, for example, one wishes to process only certain cards. The digit impulse should go to the PC–10 cable via a selector made up by the information read at the X station.

**4.4–6 Errors And Interlocks** A parity check is performed on all characters received. A command character with wrong parity is ignored. A data character with wrong parity is processed, but also causes the Error flip-flop to be set.

When a row of a print image or a card image has not been fully processed in the time available, the Incomplete Transmission flip-flop and Error flip-flop are set.

In card reading, the time available is the time during which the EMITTER digit line is high. In card punching, the time available is established by the PC–10 as the next 4 milliseconds following the start of the EMITTER digit impulse. In line printing with the full alphabet, the time available is the time between two successive CHARACTER pulses from the printer. In printing alternate characters, such as numeric only, the time available is the interval from one CHARACTER pulse to the second following CHARACTER pulse.

An interlock flip-flop in the PC–10 is in the reset condition when processing can proceed, and set otherwise. This flip-flop is set after each card and is reset for the next card provided no interlock is open in the card unit. Conditions which cause an open interlock are: power off, out of cards, stacker full, card jam.

When a continue feed command cannot be executed because of an open interlock [i.e., card jam, empty hopper, full stacker] an interrupt will be sent at the end of the current card cycle. Therefore, at a first row interrupt, the program should test for interlock with QIL before processing the card.

**4.4–7 Initial Loading Of The G–20** Initial load of the G–20 by means of the PC–10 and a card reader can be accomplished either at system power turn-on or after the system is on. For initial load in conjunction with turn-on:

1. Turn on system power and card equipment power, and wait for INITIAL LOAD light at console [or G–20 control panel].

2. Place the start up deck in the card reader, press the START button and wait for PC–10 ON light.

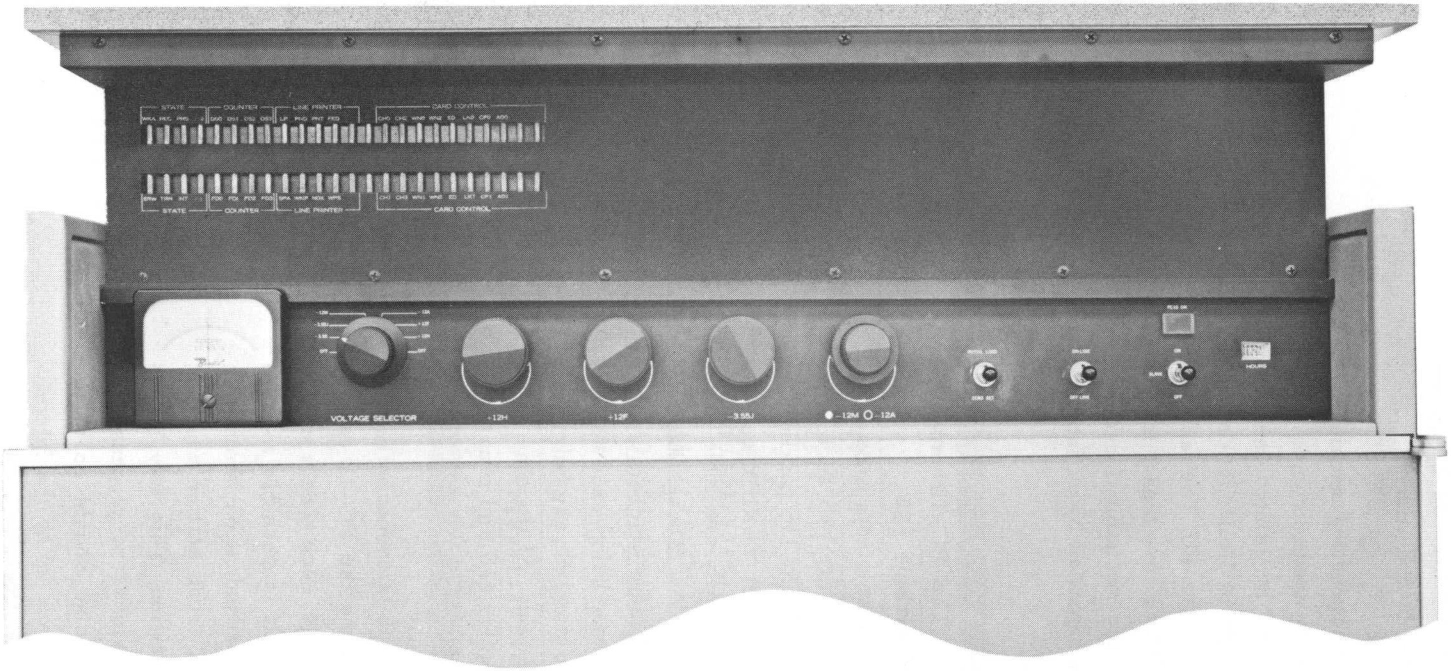


FIGURE 4.4-4 CARD-PRINTER COUPLER CONTROL PANEL

3. Operate the INITIAL LOAD switch on the PC–10.Loading of the program will now begin.

When the power is already on:

1. With XCM at hold, actuate the INITIAL LOAD switch of the G–20, return XCM to normal and operate START CLOCK switch.
2. Make sure that all units are at STANDBY. If necessary, operate ZERO MACHINE switches on MT–10's and SYSTEM INTERLOCK on console.
3. Place the start up deck in the card reader and press the START button.
4. Operate the INITIAL LOAD switch on the PC–10. Loading of the program will now begin.

**4.4–8 Controls** The PC–10 control panel, Figure 4.4–4, contains a Power switch with three positions: ON, SLAVE, and OFF. In the SLAVE position, the power to the unit is under control of the System Power signal line. ON and OFF are unconditional. A two-position switch places the unit ON-LINE or OUT of Service. No signals are sent to the communication line during transition to or from the OUT position. The INITIAL LOAD/ZERO SET switch when in the ZERO SET position places the PC–10 in STANDBY without going through the turn-on cycle. The INITIAL LOAD position is used to start the G–20 System from the PC–10 as described above.

The PC–10 also has an operating time meter,voltage controls, a voltmeter, a power-on light, as well as a service panel behind the flip-up lid.The Service panel contains test indicator lights connected to various points in the logic for test and trouble-shooting purposes.

**4.4–9 Operation Codes Of The PC–10** Programming of the PC–10 is accomplished using the command codes shown in Table 4.4–1. It should be clear from the foregoing that all communication addressed to a terminal unit is handled by the PC–10. However, from the programmer's standpoint communications may be considered to be taking place directly with the terminal accessory. Descriptions of the line codes are framed with this consideration in mind.

The PC–10 will answer any code which it understands even though the action requested is not meaningful to the operation in progress. In this case no action will occur. The PC–10 will, for example, answer GRN to 055 received during card read, but will take no other action.

Queries pertaining to card operations usually apply to both stations of a particular feed, so that in verification following reading or verification following punching, certain queries need not be repeated.

Examples 4.4–1, 4.4–2, 4.4–3, and 4.4–4 indicate the sequence of line signals which appear during normal card reading, initial loading, card punching, and line printing.

## TABLE 4.4–1 CARD–PRINTER COUPLER COMMAND CODES

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 001 | CLQ | Clear Query flip-flops. PC–10 in CALLED state answers GRN and clears common Error and Interrupt flip-flops. Clears Incomplete Transmission flip-flop of unit called. |
| 002 | GRN | Green. The response the PC–10 gives to all five of its calls. Also a possible response to a query. |
| 003 | RED | Red. A possible response to a query. |
| 004 | END | A PC–10 in MESSAGE state upon hearing END will enter STANDBY state and give no reply. The Ready flip-flop of the unit called goes to not ready [Set].<br>A PC–10 in receive MESSAGE state during card reading sends END in response to the 11th REQ and enters STANDBY state. The reader goes to not ready. PC–10 in receive MESSAGE state during INITIAL LOAD sends END in response to the 120th REQ and enters STANDBY state. Reader goes to not ready. |
| 005 | ERR | End of block containing an Error. PC–10 in MESSAGE state enters STANDBY state and does not reply. Addressed unit goes not ready. |
| 010 | SDT | Start Data Transfer. PC–10 in INSTRUCTED state enters receive MESSAGE and sends REQ or enters transmit MESSAGE state and listens for REQ. |
| 014 | RCV | Receive. PC–10 in CALLED state enters INSTRUCTED state, and answers GRN. [PC–10 now can hear only SDT or CALs.] Not applicable to reader. |
| 016 | TRA | Transmit. PC–10 in CALLED state enters INSTRUCTED state, and answers GRN. [PC–10 can now hear only SDT or CALs.] Applies to reader only. |
| 050 | PAN | Print Alphanumeric. PC–10 in CALLED state answers GRN and prepares to print all rows. Applies to LP–10, LP–11 only. [When turned on the equipment assumes an indeterminate state. Any 050, 051, or 052 directed to a card station sets the print circuitry to the alphanumeric state]. |
| 051 | PNU | Print Numeric. PC–10 in CALLED state answers GRN and prepares to print even rows only. Applies to LP–10, LP–11 only. [See note under PAN]. |
| 051 | AS1 | Alternate Stacker 1. PC–10 in CALLED state answers GRN and uses alternate stacker 1 for the next card cycle of the card feed addressed. This command must be given for each card cycle which is to use alternate stacker 1. |

## TABLE 4.4-1 CARD-PRINTER COUPLER COMMAND CODES (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 052 | PAB | Print Alphabetic. PC-10 in CALLED state answers GRN and prepares to print odd rows only. Applies to LP-10, LP-11 only. [See note under PAN]. |
| 052 | AS2 | Alternate Stacker 2. PC-10 in CALLED state answers GRN and uses alternate stacker 2 for the next card cycle of the card feed addressed. This command must be given for each card cycle which is to use alternate stacker 2. |
| 054 | CFD | Continue Feed. PC-10 in CALLED state will answer GRN and continue feed of cards or paper in the addressed feed. In order to feed the next card without losing time this signal must be sent after the last card row but not later than T milliseconds after the last row. See Table 6.5-3, and also Section 4.4-6. For line printing, CFD following STT will cause an advance of one additional line. Continuous advance will result if CFD is given before completion of each upspace. An interrupt will be sent at the completion of each upspace except the last. Upspace cycle is 6.5 milliseconds for LP-11 and 8.2 milliseconds for LP-10. |
| 055 | SIN | Send Index Interrupt. PC-10 in CALLED state answers GRN and causes an interrupt to be sent the next time row 63 of the print roll comes into print position after paper feed stops. |
| 056 | STT | Start Feed. PC-10 in CALLED state will answer GRN and feed first card of the card feed addressed [CFD required to continue feed]. For line printing, this command upspaces one line unless followed by CFD or by a command numeric, 10n, where 0≤n≤7. In this case, upspacing continues until stopped by the next hole in channel "n" of the format loop. |
| 060 | QRD | Query Ready. A PC-10 in CALLED state answers GRN if a card row or print row is in position for processing in the unit whose number was called. Otherwise, RED. |
| 061 | QER | Query no Error. PC-10 in CALLED state answers GRN if there have been no parity errors, or incomplete transmission errors, since the last CLQ. Replies RED otherwise. The Error flip-flop is not reset by QER. [Use CLQ]. |
| 062 | QIL | Query no Interlock. PC-10 in CALLED state answers GRN if card feed addressed is not stopped due to any card machine interlock being open. Reply is RED otherwise. The Interlock flip-flop is reset by manual removal of the interlock condition, but is not reset by QIL. |

**TABLE 4.4–1 CARD–PRINTER COUPLER COMMAND CODES (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 063 | QCT | Query Complete Transmission. PC–10 in CALLED state answers GRN if data were completely processed in time allowed in both stations of the feed addressed. Reply is RED otherwise. The Incomplete Transmission flip-flop is not reset by QCT. [Use CLQ]. |
| 065 | QIP | Query Index Pulse. PC–10 in CALLED state answers GRN during the interval between the printer index pulse position 63 and printer position 65, provided an SIN has been received before QIP and provided paper feed is stopped. Reply is RED at other times. |
| 065 | QLR | Query not Last Row. PC–10 in CALLED state answers GRN if card feed addressed is not processing last row. Otherwise, RED. |
| 067 | QIN | Query no Interrupt. PC–10 in CALLED state answers GRN if PC–10 has not transmitted an interrupt request since the last CLQ was received. Reply is RED otherwise. The Interrupt flip-flop is not reset by the QIN. [Use CLQ]. |
| 10n | 10n | Select format channel n [0 ≤ n ≤ 7] . This command received by PC–10 in CALLED state following STT will cause printer paper feed to halt at the next hole in format channel n. Reply is GRN. |
| 200+<br>uuu | CALs | Call to the PC–10 itself. PC–10 in STANDBY [or INSTRUCTED] enters CALLED state. Reply is GRN. |
| 200 +<br>uuu | CALo | Call to other unit. PC–10 in CALLED state enters STANDBY. No reply is transmitted. |
| 400+<br>ddd | ddd | Data character. [0 ≤ ddd ≤ 377.] The data character is processed provided the PC–10 is in the receive MESSAGE state. Reply is REQ. |
| none | REQ | Request for character, sent on REQ line. A PC–10 in transmit MESSAGE state upon hearing REQ will send next data character; at end of data block will send END and enter STANDBY state. A PC–10 in receive MESSAGE state will send REQ in reply to SDT and after each data character. |
| none | INT | An interrupt sent on the interrupt line at the time that the interrupt flip-flop is being set by the rise of the emitter signal in card operations, or by the arrival of character time 63 in printer operations. The sending of an interrupt is always conditioned on the flip-flop being in the reset state at the time the interrupt is to be sent. In printer operations it is also conditioned upon SIN having been sent previously. See CLQ. |

## EXAMPLE 4.4–1 SEQUENCE OF LINE SIGNALS – CARD READ

| G–20 | PC–10 | REMARKS |
|---|---|---|
| CALs | GRN | PC–10 enters CALLED. |
| QIL | GRN | Query Interlock. |
| CLQ | GRN | Insures Query flip-flops are cleared. |
| STT | GRN | Start card feed. G–20 returns to program. |
| .... | .... | ...... |
| [A] | INT | Reader is in position to read. |
| CALs | GRN | Interrupt routine calls first station of PC–10. |
| QIN | RED | Identifies the PC–10 which interrupted. |
| [1] QRD | GRN | Identifies unit which is ready to read. Go to [2]. |
|  | RED | Continue to look for unit which is ready. |
| CALs | GRN | Call next station of PC–10. Go to [1]. |
| [2] QLR | RED | Last row, go to [3]. |
|  | GRN | Not last row. Continue. |
| QIL | GRN | Go to [4]. |
| [3] CFD | GRN | Continue feed. |
| [4] TRA | GRN | Prepare to transmit. |
| SDT |  | Start data transfer. Enters MESSAGE state. |
| REQ | ddd | First data character. |
| .... | .... | .... |
| REQ | ddd | Last data character. |
| REQ | END |  |
| CALs | GRN | Call any station of the PC–10. |
| QER | GRN | Query for error. |
| CLQ | GRN | Clear Query flip-flops. G–20 returns to program.<br>At next interrupt program returns to [A]. |

### EXAMPLE 4.4–2 SEQUENCE OF LINE SIGNALS – INITIAL LOAD

| G–20 | PC–10 | REMARKS |
|---|---|---|
| | | Operator starts initial load by carrying out operations shown in Section 4.4–7. |
| | ddd | ddd $\rightarrow {}_{31}(64)_{24}$ in G–20. |
| REQ | ddd | ddd $\rightarrow {}_{23}(64)_{16}$ in G–20. |
| .... | .... | .... |
| REQ | ddd | 120th data character $\rightarrow {}_{7}(93)_{0}$ in G–20. |
| REQ | END | PC–10 enters STANDBY state . |

## EXAMPLE 4.4–3 SEQUENCE OF LINE SIGNALS – CARD PUNCH

| G–20 | PC–10 | REMARKS |
|---|---|---|
| CALs | GRN | PC–10 enters CALLED state. |
| QIL | GRN | Query interlock. |
| CLQ | GRN | Insures that Query flip-flops are cleared. |
| STT | GRN | Start card feed. G–20 returns to program. |
| .... | .... | .... |
| [A] | INT | Punch is in position to punch. |
| CALs | GRN | Interrupt routine calls first station of PC–10. |
| QIN | RED | Identifies the PC–10 which interrupted. |
| [1] QRD | GRN | Identifies unit which is ready to punch. Go to [2]. |
|  | RED | Continue to look for unit which is ready. |
| CALs | GRN | Call next station of PC–10. Go to [1]. |
| [2] QLR | RED | Last row. Go to [3]. |
|  | GRN | Not last row. Continue. |
| QIL | GRN | Go to [4]. |
| [3] CFD | GRN | Continue feed. |
| [4] RCV | GRN | Prepare to receive. |
| SDT | REQ | Start data transfer. |
| ddd | REQ | First character. |
| .... | .... | .... |
| ddd | REQ | Last character. |
| END |  |  |
| CALs | GRN | Call any station of the PC–10. |
| QER | GRN | Query for error. |
| CLQ | GRN | Clear query flip-flops. G–20 returns to program. At next interrupt program returns to [A]. |

## EXAMPLE 4.4-4 SEQUENCE OF LINE SIGNALS-HIGH-SPEED LINE PRINTING, LP-10

| G-20 | PC-10 | REMARKS |
|---|---|---|
| CALs | GRN | Unit enters CALLED state. |
| CLQ | GRN | Clears query flip-flops. |
| PAN | GRN | Set up to print Alphanumeric. |
| [A] SIN | GRN | Send index interrupt. G-20 returns to its program. |
| .... | .... | .... |
|  | INT | Index interrupt is sent. |
| CALs | GRN | G-20 services the interrupt. Unit enters CALLED state. |
| QIN | RED | Acknowledges interrupt. |
| QIP | GRN | Query index pulse. |
| [B] QRD | GRN | Query ready. |
| RCV | GRN | Prepare to print. |
| SDT | REQ | Start data transfer. |
| ddd | REQ | First character received. |
| .... | .... | .... |
| ddd | REQ | Last character received. |
| END |  | End of print row. Unit returns to STANDBY. |
| CALs | GRN | Unit enters CALLED State. Return to [B] 63 times, then continue. |
| QRD | GRN | Query ready to start paper feed. |
| QER | GRN | No error or incomplete transmission. |
| CLQ | GRN | Clears query flip-flops. |
| STT | GRN | Paper feed starts. |
| 10n | GRN | Format line n is selected. Return to [A]. |

## SECTION 4.5–LINE PRINTERS, LP–10 AND LP–11

The Bendix LP–10 and LP–11 Printers provide for line-at-a-time printing of numeric or alphanumeric data at rates up to 1,500 lines per minute on forms as large as 120 characters wide and 44 inches deep. A continuously revolving print roll carries a number of complete sets of printing characters, each set occupying one circumferential track on the print roll. A line of print is created by driving the paper and ribbon against the print roll by means of a set of individually timed hammers, one for each print column. Timing of each hammer determines the character to be printed in that column. The paper remains stationary during printing and is upspaced after a complete line has been printed. A standard print roll has an alphabet of 63 printing characters and a blank. This 63–character alphabet is a subset of the ALGOL [International Algebraic Language] alphabet.

Both printers, LP–10 and LP–11, are available on standard order with the print roll layout shown in Table 4.5–1. Other alphabets are available only on special order. In the operation of either printer, each line character supplies eight bits of one row of the print image. These line characters are loaded into a register which, in the case of the LP–10, holds 72 bits and, in the case of the LP–11, holds 120 bits, corresponding to the 72 or 120 characters per line available on these two printers. These printers are driven by the PC–10 Card–Printer Coupler or the CB–11 Control Buffer. In either case, the print head must supply index pulses and character pulses to the driving device to indicate when the signals are to be supplied for actuating the hammers. For example, at the K position of the print roll a string of bits will be delivered to the printer which contains a "1" bit everywhere that there is to be a K printed in that particular line. The index pulse is sent out by the Line Printer at print roll position 63. Character pulses are supplied at all positions except 63 and 64. Position 65 is a character position which is not used on the LP–10 and LP–11, but 63 and 64 are not character positions. Paper feed for both machines is accomplished by a sprocket-type drive which is under control of the start paper and stop paper signals and the vertical format loop. The format loop is a paper tape containing eight channels, numbered 0 through 7, and must be punched to contain a hole corresponding to each point where it is desired to stop the paper during each of the eight formats.

**4.5–1 LP–10 Line Printer** The LP–10 shown in Figure 4.5–1 is housed in a cabinet 64 inches high, 34 inches wide, and 28 inches deep. It weighs 1,150 pounds and requires 2.9 KVA of 115–volt, 60–cycle, single–phase power.

The maximum printing speed of the LP–10 is 900 lines per minute. To achieve this speed it is necessary to perform all computations associated with the printing either during upspace time or during the time between rows of the print image. The upspace paper time for this machine is 22 milliseconds for single line. For larger spaces the rate of paper feed is 18 inches per second. Forms to be used on the LP–10 must be 8–1/2 inches wide including sprocket holes and the length of the form must not exceed 11 inches. Spacing of characters is 10 per inch horizontally and 6 per inch vertically, equivalent to pica type. The LP–10 will make an original and three carbon copies.

**4.5–2 LP–11 Line Printer** The LP–11 print mechanism is contained in a cabinet 55 inches high, 42 inches wide, and 28 inches deep, weighing 750 pounds. This unit is driven by a Printer Control Unit contained in a cabinet 64 inches high, 34 inches wide, and 28 inches deep, weighing 1,350 pounds. See Figures 4.5–2, 4.5–3, and 4.5–4. Total input power to the LP–11, is 5.2 KVA of 115–volt, 60–cycle, single–phase power. Two print roll speeds are available, permitting printing at 1,000 or 1,500 lines per minute. To achieve these speeds, all computations must be performed either between row times or during paper upspace time which is 16 milliseconds. Multiple line paper feeding is at 25 inches per second. The forms for use on the LP–11 may be of any width from 4 to 20 inches, including sprocket holes, and the form length may be anything up to 44 inches. The LP–11 prints up to 120 columns of characters spaced 10 per inch horizontally and 6 per inch vertically, equivalent to pica type. The LP–11 also incorporates a number of controls which can be adjusted while the printer is operating. These include penetration control, tractor lateral adjustment, paper tension, and vertical form adjustment.

FIGURE 4.5-1 LINE PRINTER, LP-10

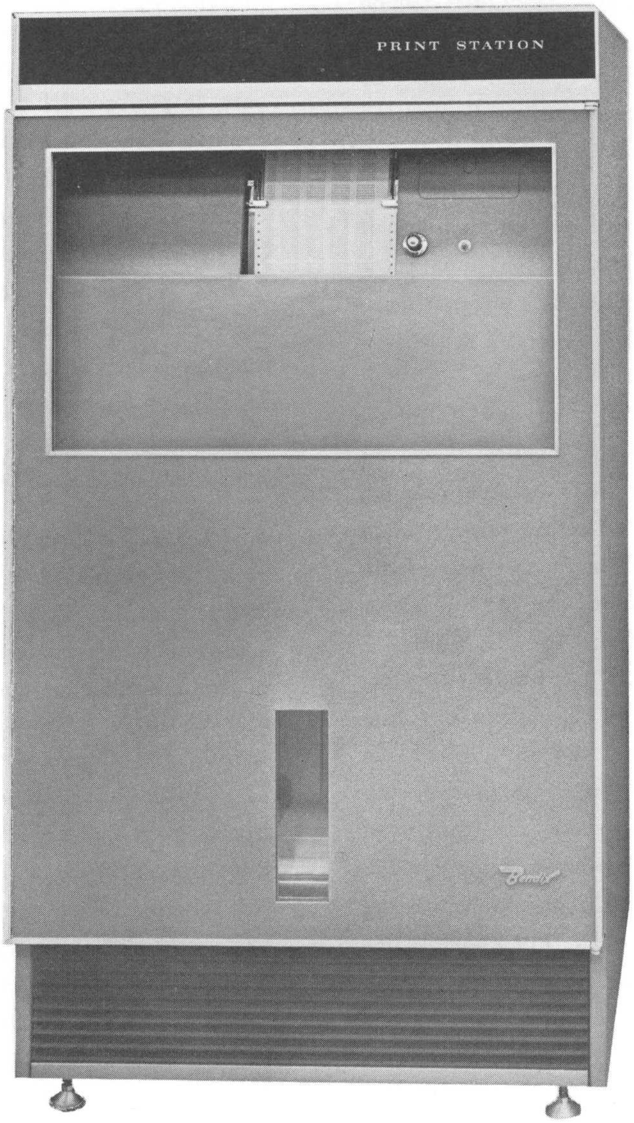
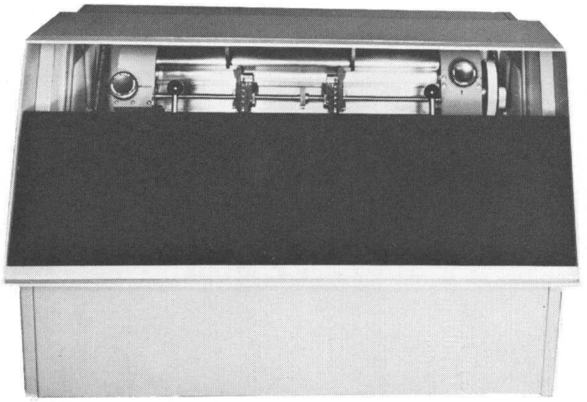
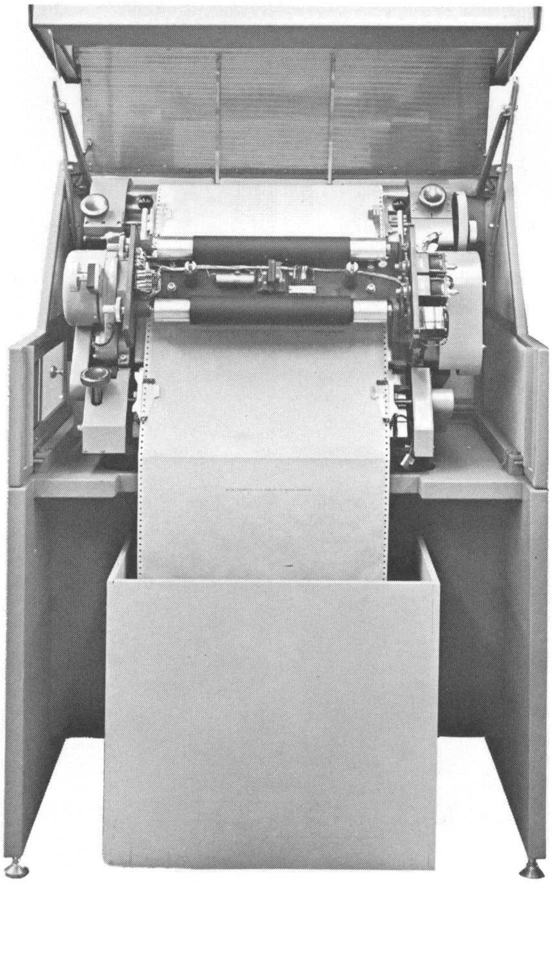


FIGURE 4.5–2 LINE PRINTER, LP–11 (a) EXTERNAL (b) WITH COVER OPEN



(a)



(b)

FIGURE 4.5–3 DETAIL OF LP–11 PRINT HEAD

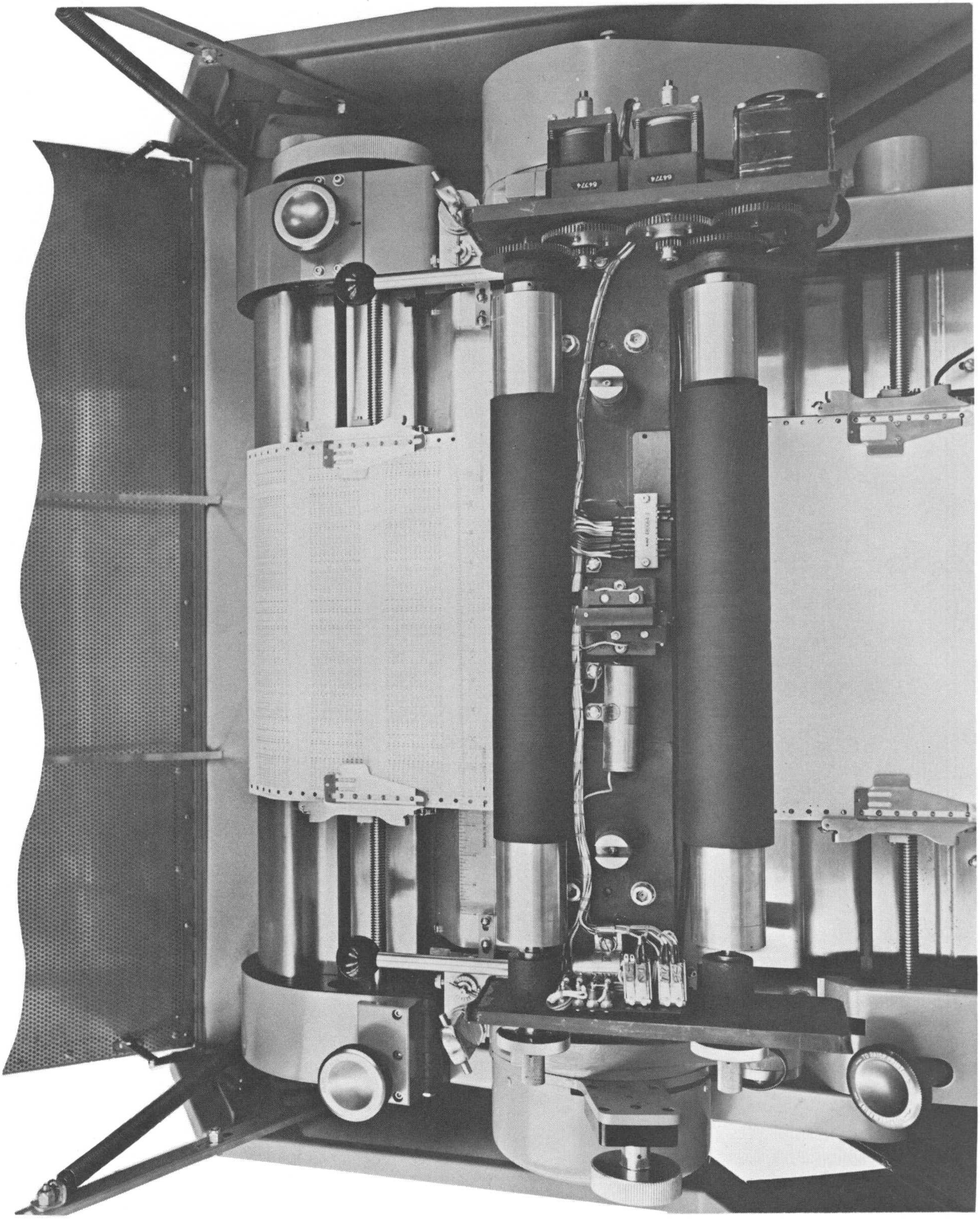
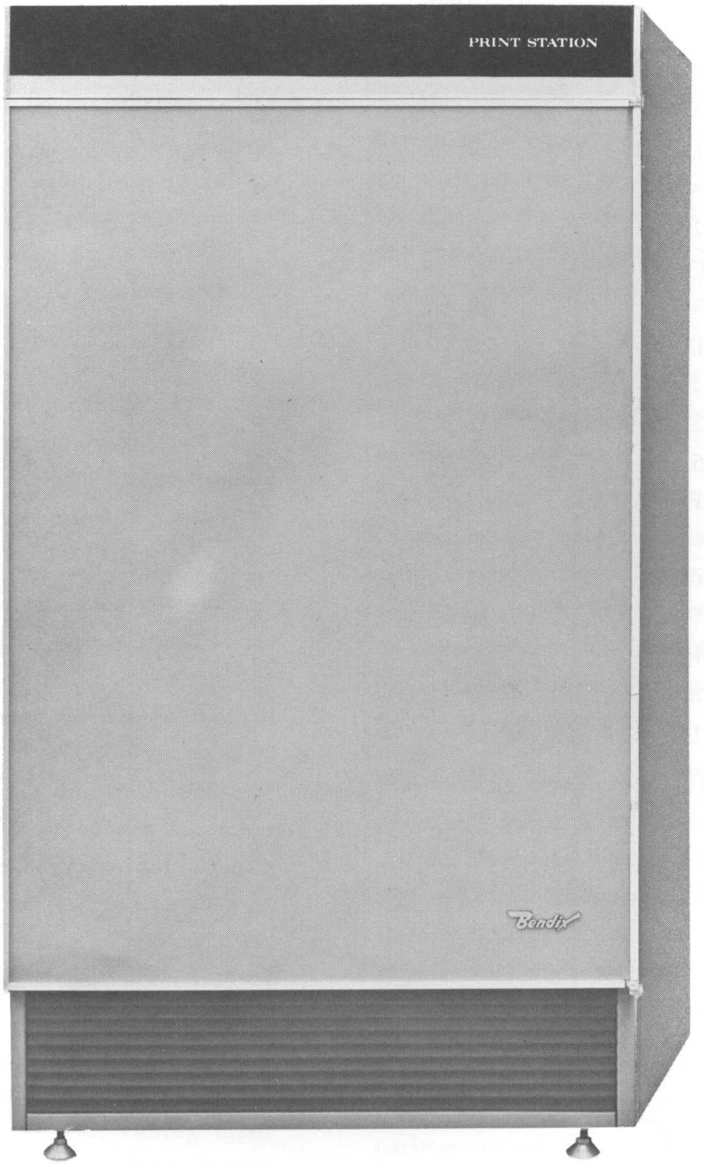


FIGURE 4.5–4 PRINT CONTROL, LP–11



## TABLE 4.5-1 CHARACTER SEQUENCE AROUND PRINT ROLL, LP-10, LP-11

| POSITION | CHARACTER | POSITION | CHARACTER | POSITION | CHARACTER |
|---|---|---|---|---|---|
| 0 | 0 | 22 | . | 44 | > |
| 1 | A | 23 | L | 45 | W |
| 2 | 1 | 24 | + | 46 | ; |
| 3 | B | 25 | M | 47 | X |
| 4 | 2 | 26 | – | 48 | ( |
| 5 | C | 27 | N | 49 | Y |
| 6 | 3 | 28 | * | 50 | [ |
| 7 | D | 29 | O | 51 | Z |
| 8 | 4 | 30 | / | 52 | ] |
| 9 | E | 31 | P | 53 | \| |
| 10 | 5 | 32 | = | 54 | ) |
| 11 | F | 33 | Q | 55 | ← |
| 12 | 6 | 34 | ∨ | 56 | ↓ |
| 13 | G | 35 | R | 57 | → |
| 14 | 7 | 36 | ≠ | 58 | ↑ |
| 15 | H | 37 | S | 59 | ¬ |
| 16 | 8 | 38 | ∧ | 60 | : |
| 17 | I | 39 | T | 61 | , |
| 18 | 9 | 40 | < | 62 | ' |
| 19 | J | 41 | U | 63 | None |
| 20 | $_{10}$ | 42 | $ | 64 | None |
| 21 | K | 43 | V | 65 | Not used |

Note: A character pulse is sent by the printer at positions 0–62 and 65. The character pulse at 65 is ignored by the PC–10 and CB–11. An index pulse is sent at position 63. There is no pulse at 64.

## SECTION 4.6–CONTROL BUFFER, CB–11

As implied by its name, the Control Buffer is a device used to control accessory equipment and to buffer input/output operations. It has a group of slave states in which it transmits or receives blocks of data under instructions from the Central Processor, and answers queries. In addition, it has a set of control states in which it can control accessories as well as handle off-line operations.

The CB–11 will read cards using the IBM 088, 514 or 528, punch cards using the IBM 544, 514, 523, or 528, and print using the Bendix Line Printers LP–10 and LP–11. [Row binary punching is permitted on IBM 544 only.] The Control Buffer will also print on an IBM 407. The CB–11 can control as satellites not more than three card read stations and one card punch or tabulator station [operating on not more than two independent feeds], and one Line Printer. The total processing time is shared among satellites. The CB–11 Control Buffer can also communicate with any of the devices which are connected directly to the communication line.

In operation the CB–11 holds in its own 4,096–character core memory a block of data and a group of instructions, all of which information can be dumped into the CB–11 at high speed from Central Processor or from a magnetic tape unit. The CB–11 can then process and output the data using its own stored program, thus freeing the Central Processor for other operations. The Control Buffer may also use its own internally stored program to read information in from a satellite device. Upon completion of the read-in, the Control Buffer then transmits the information at high speed to the Central Processor or to a magnetic tape unit. The CB–11 communicates with the Central Processor at 167,000 characters/second and with the MT–10 at 120,000 characters/second. During input or output operations the CB–11 can convert row binary to Extended Hollerith, Extended Hollerith to row binary, form a print image, and perform other special functions. The Control Buffer contains a program-controlled switch which permits it to connect itself to either of two communication lines.

The Control Buffer is housed in a cabinet 64 inches high, 34 inches wide, and 28 inches deep, and weighs 1,200 pounds. It requires 1.5 KVA at 115 volts, 60 cycles. See Figure 4.6–1.

**4.6–1 Organization Of The CB–11** As will be noted in Figure 4.6–2, the Control Buffer is a special purpose stored program computer, whose principal manipulations are performed by a shift register. It operates as a single address machine. An operation code always occupies one character while an address always occupies two characters. Thus, those commands not requiring an address take one character, while those requiring an address take three characters. An operand is usually either one character, two characters, or a block. The memory stores 4,096 characters, each containing eight information bits and one parity bit. The characters are numbered in octal 0000–7777, and the bits in a character are numbered from right to left starting with bit 0. Location 0000 follows location 7777.

There is associated with the memory, an Address Counter of two characters and a Memory Buffer register of one character.

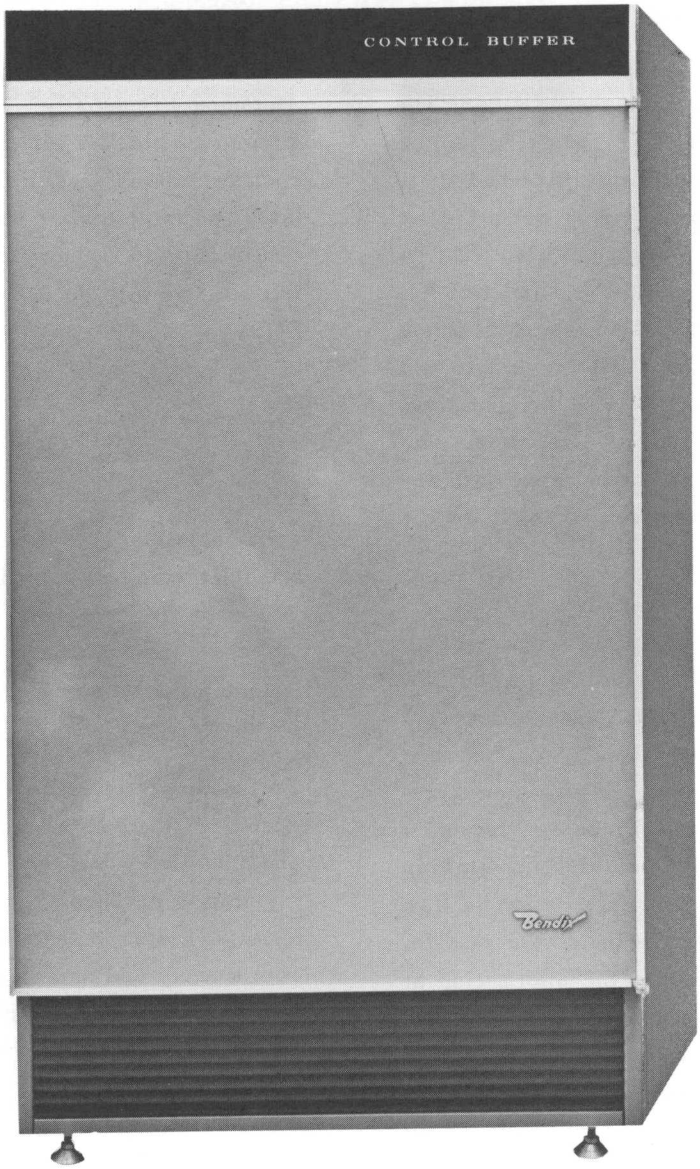
The Memory Buffer receives information from the communication line through the Line Register. Command Decoding is associated with the Line Register. The data flag which is present on a data character on the communication line is deleted at the Line Register. For outgoing characters the flag is supplied by the Memory Buffer.

For holding a reference value during comparison and decoding, there is a Reference Source register, RS, and for storing the limit in block operations a Limit register, LM. These registers communicate with the Memory Buffer. Communication between the CB–11 core memory and satellite devices is handled through a Shift register, SF, or Echo register, SH. SF and SH form a shift register for manipulations.

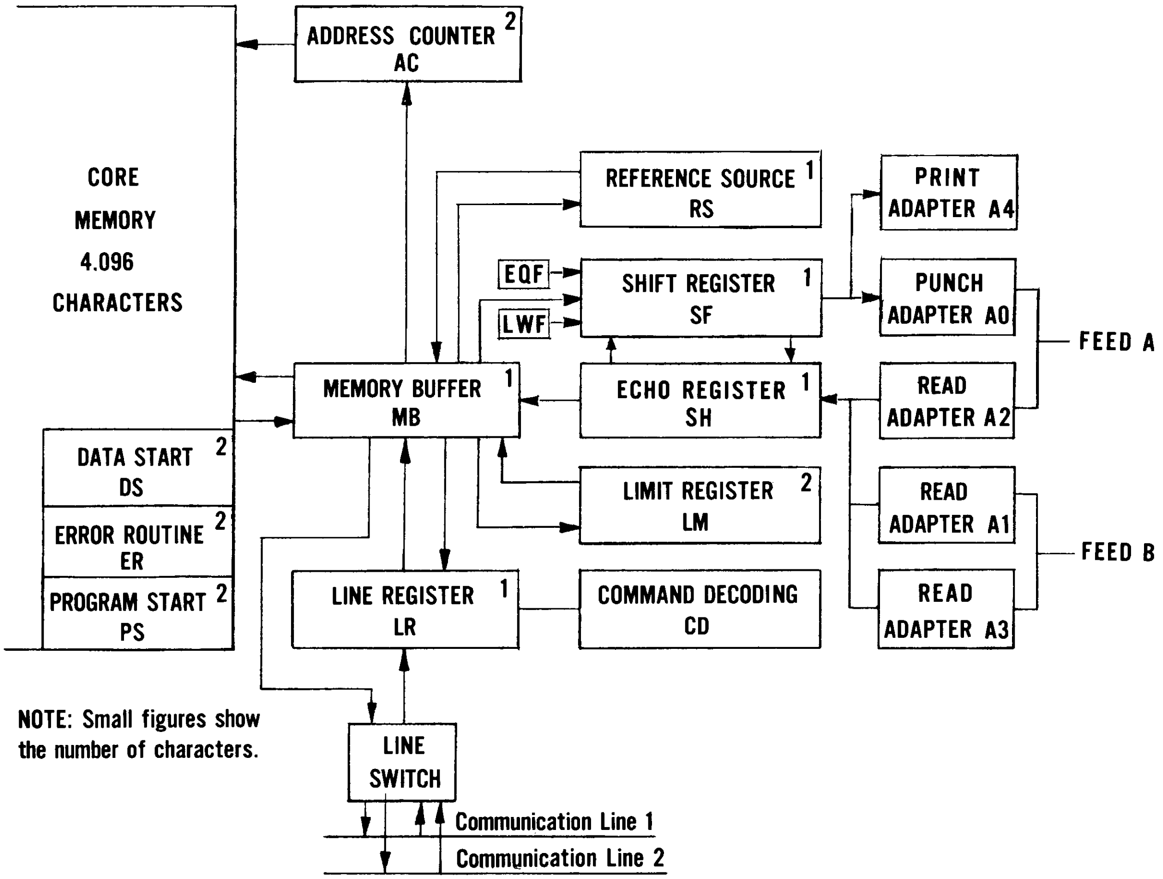
Flip-flops EQF and LWF are set to indicate comparison equal and comparison lower respectively.

The first six characters in core memory are used for special purposes and hence are not available as general storage. Characters 0 and 1 constitute the Program Starting register, PS; characters 2 and 3 carry the starting address, ER, of the error routine; characters 4 and 5 hold DS, the data starting address for block operations.

FIGURE 4.6-1 CONTROL BUFFER, CB-11



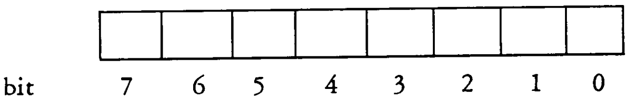
## FIGURE 4.6–2 SIMPLIFIED BLOCK DIAGRAM, CONTROL BUFFER



NOTE: Small figures show the number of characters.

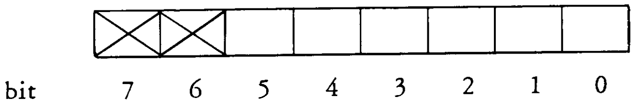
A command character or a data character in the CB–11 uses the full 8 bits:

CB–11 COMMAND OR DATA CHARACTER



while an address character uses only 6 bits leaving 2 bits unused:

CB–11 ADDRESS CHARACTER



The Line Register, LR, has 10 bits, bit 8 being the flag position, and bit 9 the parity position. The Memory Buffer has 9 bits, bit 8 being the parity bit. The other registers omit both flag and parity bits. The Address Counter, AC, and Limit register, LM, omit bits 6 and 7 and carry only bits 0–5 of each character. The entire 12 bits in LM are numbered from right to to left 0–11. When a character is transferred from LM to MB, zeros are supplied for bits 6 and 7 of each character. Inside the memory all characters carry an even parity bit in position 8.

A command requiring an address has the following configuration in CB–11 core memory:

| characters k | k + 1 | k + 2 |
|---|---|---|
| OP | a1 | a2 |

The situation is similar for a command executed from line signals except that the characters are received successively from the line rather than drawn successively from memory.

| characters k | k + 1 | k + 2 |
|---|---|---|
| OP | C1 | C2 |

Except when executing tests and transfers of control, characters are drawn from memory in increasing numerical order. In either slave or control states the characters a1 and a2 or C1 and C2 are the more significant and less significant characters of a 2-character operand designator, usually referred to by the symbol Y. The operand designator is equivalent to a 12–bit binary number.

The movement of information for a typical card read input situation [initial load CB–11] is shown in Table 4.6–1 and for a typical CB–11/G–20 data transfer [load the G–20 from the CB–11] is shown in Table 4.6–2. In these tables the letters refer to the boxes in Figure 4.6–2 and a parenthesis means "contents of". A right going arrow means "replaces". In the initial loading of the CB–11 the contents of a single card is read in row binary starting with columns 1–8 of the first row and continuing through columns 73–80 of the last row. In data transfer from the CB–11 to the G–20 a block of 4n characters in successively ascending locations in the CB–11 is transferred to a block of n successively ascending locations in the G–20. These two modes of data movement and their inverses are the principal ones involved in "buffering type operations" of the CB–11. Editing operations constitute another general class of data movement in the CB–11, to be discussed later under specific headings.

**4.6–2 Operating States** The Control Buffer has a group of slave states called, STANDBY, CALLED, INSTRUCTED, and MESSAGE and it has three control states called CONTROL INTERNAL, CONTROL TRANSMIT, and CONTROL MESSAGE. In addition, it possesses an OUT of Service state and an INITIAL LOAD state. See Figure 4.6–3.

OUT of Service. In the OUT of Service state a Control Buffer is electrically disconnected from the system and neither transmits nor receives any information and is deaf to any signals. When in this state it can be put into operation by operating the ON–LINE switch. This action will put it into the STANDBY state.

STANDBY. In the STANDBY state, the Control Buffer listens for its own call signal on the communication line. It decodes and rejects all signals except its own call with correct parity. Upon receiving this signal it goes into the CALLED state.

CALLED. In the CALLED state the Control Buffer receives and executes commands, receives and answers queries, and upon receiving a complete set of

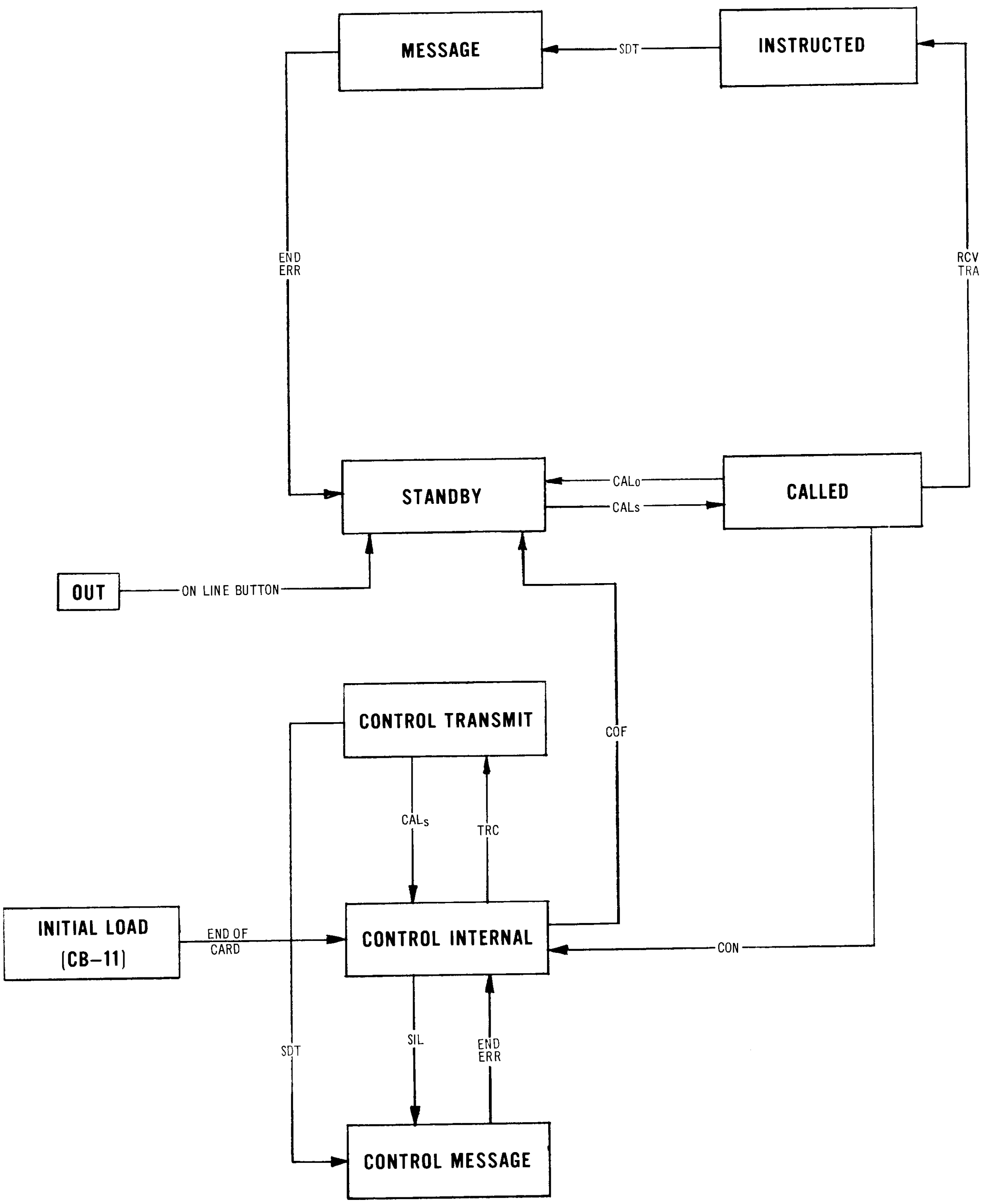
## TABLE 4.6-1 INITIAL LOAD CB-11

| | OPERATION | COMMENTS |
|---|---|---|
| | 0 → (AC) | Reset memory Address Counter. |
| | (A1) → (SH) | Take a character from station 1 to SH. |
| | (SH) → (MB) | Take the character from SH to MB. |
| 119 | (MB) → ((AC)) | Store character in memory. |
| | (AC) + 1 → (AC) | Increment memory Address Counter. [Repeat loop 119 more times.] |
| | (PS) → (AC) | Set memory Address Counter to Program Starting address. |
| | ((AC)) → (MB) | Access first character of program. |
| | (MB) → ((AC)) | Restore memory. |
| | (MB) → (CD) | Decode command. |
| | etc. | |

## TABLE 4.6-2 LOAD G-20 FROM CB-11, n WORDS

| | OPERATION | COMMENTS |
|---|---|---|
| | (DS) → (MB) | Access Data Start address. |
| | (MB) → (DS) | Restore memory. |
| | (MB) → (AC) | Load Data Start address into AC. |
| | ((AC)) → (MB) | Access character from memory. |
| | (MB) → ((AC)) | Restore memory. |
| 4n−1 | (MB) → ddd | Send character on the line. |
| | (AC) + 1 → (AC) | Increment memory Address Counter. Repeat loop [4n−1] more times. |

FIGURE 4.6-3 CONTROL BUFFER OPERATING STATES



instructions for doing some operation, enters the INSTRUCTED state.

INSTRUCTED. In the INSTRUCTED state the Control Buffer remembers the instructions it has received but ignores further signals until it has a command to start data transfer. Upon receiving an SDT it will enter the MESSAGE state and begin executing the operation which was set up by the previous group of instructions. A Control Buffer in the INSTRUCTED state will revert to the CALLED state if it hears its own calling signal.

MESSAGE. In the MESSAGE state the Control Buffer is being controlled by another device, typically the the Central Processor, with which it is communicating, and acting as a transmitter or receiver for a block of data. During the time that the Control Buffer is using one of the lines and is in the MESSAGE state, other devices in the system should avoid placing any signals on that line. Upon completion of a block transfer of information, the Control Buffer will revert to the STANDBY state.

CONTROL INTERNAL. Upon receipt of a CON [Control on] command, the CB–11 will enter the CONTROL INTERNAL state and execute commands from its own memory, beginning at the address specified by the contents of PS. Upon reading a TRC [Transmit Command], it will enter the CONTROL TRANSMIT state; upon reading an SDT [Start Data Transfer] command, it will enter the CONTROL MESSAGE state, and upon reading a COF [Control OFF] command, it will revert to the STANDBY state, and send an interrupt.

CONTROL TRANSMIT. In the CONTROL TRANSMIT state, the Control Buffer is reading commands from its own core memory and transmitting them over the communication line. Upon reading its own calling signal, the buffer will enter the CONTROL INTERNAL state and upon reading an SDT [Start Data Transfer] command, it will enter the CONTROL MESSAGE state. If there is no response to a line transmission in 1.5 seconds, control will be transferred to (ER).

CONTROL MESSAGE. In the CONTROL MESSAGE state the Control Buffer may act as a transmitter, or as a receiver, or as a control device not participating in the conversation. At the end of the block, which will be signalled by an END or ERR command, the Control Buffer will revert to the CONTROL INTERNAL state. If, while in the CONTROL MESSAGE state, the buffer is acting as controlling device rather than as a transmitter or receiver, it will place no signals upon the communication line. In the event of no response to a line transmission in 1.5 seconds, the Control Buffer will branch to the error routine as explained above.

INITIAL LOAD. INITIAL LOAD is a special state used in starting the CB–11 system by means of a card reader. In this state a card is read into the Control Buffer under manual control after which the Control Buffer enters the CONTROL INTERNAL state. The 120 characters read from the initial load card must constitute a program which the Control Buffer can execute to get the system started.

**4.6–3 Command List** A Control Buffer in any of the slave states, or in the CONTROL TRANSMIT or CONTROL MESSAGE state, understands the commands and gives the replies listed in Table 4.6–3.

In the CONTROL INTERNAL state, the Control Buffer reads from its own memory and executes the commands listed in Table 4.6–4.

In these tables the following conventions are observed:

| | | |
|---|---|---|
| ( ) | = | contents of |
| → | = | replaces |
| $_m[ABC]_n$ | = | Bits m through n of expression ABC |
| Y | = | operand designator formed from the next two characters following any opcode which has an address. The character first received or accessed is the more significant. Only the least significant 6 bits of a character are used in addresses. |
| B | = | block length in characters |

All register symbols refer to Figure 4.6–2.

**TABLE 4.6–3 CB–11 COMMAND CODES: ON–LINE, APPLICABLE TO ALL SLAVE STATES, CONTROL TRANSMIT, CONTROL MESSAGE [OPERATIONS INVOLVING THE COMMUNICATION LINE]**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 002 | GRN | A possible reply received by a CB–11 in the CONTROL TRANSMIT state. A GRN reply to a query sent by the CB–11 causes the CB–11 program to skip two characters.<br>Timing 5 microseconds.<br>A GRN reply to any other command causes the CB–11 program to continue in sequence.<br>Timing 3 microseconds.<br>[A possible response issued by a CB–11 in CALLED state.] |
| 003 | RED | A possible reply received by a CB–11 in the CONTROL TRANSMIT state. A RED reply to a query sent by the CB–11 causes CB–11 program to jump to Y. A RED reply to any other command causes the CB–11 program to jump to (ER):<br>(ER) → (AC)<br>Timing 15 microseconds.<br>[A possible response issued by a CB–11 in the CALLED state.] |
| 004 | END | End of Data block. A CB–11 in MESSAGE state upon hearing END enters STANDBY state. Does not reply.<br>A CB–11 in the CONTROL MESSAGE state upon hearing END enters CONTROL INTERNAL and begins executing program at (AC).<br>Timing 4 microseconds.<br>[A CB–11 in the CONTROL MESSAGE state will send END when the next Address Counter reaches the assigned limit: (LM) = (AC).] |
| 005 | ERR | Error. A CB–11 in the MESSAGE state upon hearing ERR enters STANDBY state and sets Error flip-flop [ERF]. Does not reply.<br>A CB–11 in the CONTROL MESSAGE state upon hearing ERR enters CONTROL INTERNAL and jumps to (ER).<br>Timing 16 microseconds. |

**TABLE 4.6-3 CB-11 COMMAND CODES: ON-LINE, APPLICABLE TO ALL SLAVE STATES, CONTROL TRANSMIT, CONTROL MESSAGE [OPERATIONS INVOLVING THE COMMUNICATION LINE] (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 006 | SW1 | Switch to Line 1. A CB–11 in the CALLED state upon hearing SW1 will answer GRN. If on primary line, no action will occur.<br>Timing 6 microseconds.<br>If on secondary line, main cable will switch to primary line and unit will not respond to commands for 6 milliseconds. |
| 007 | SW2 | Switch to Line 2. A CB–11 in the CALLED state upon hearing SW2 will answer GRN. If on secondary line, no action will occur.<br>Timing 6 microseconds.<br>If on primary line, main cable will switch to secondary line and unit will not respond to commands for 6 milliseconds. |
| 010 | SDT | Start Data Transfer. CB–11 in INSTRUCTED state will enter MESSAGE state and load (DS) into AC. If to transmit, will await REQ. If to receive, will send REQ.<br>Timing 16 microseconds.<br>A CB–11 in CONTROL TRANSMIT state upon reading SDT from its own memory will enter CONTROL MESSAGE state and load (DS) into AC. If to transmit, will await REQ. If to receive, will send REQ.<br>Timing 16 microseconds. |
| 014 | RCV | Prepare to Receive. A CB–11 in CALLED state replies GRN and enters INSTRUCTED state.<br>Timing 4 microseconds. |
| 016 | TRA | Prepare to Transmit. A CB–11 in CALLED state replies GRN and enters INSTRUCTED state.<br>Timing 4 microseconds. |
| 022 | SLM | Store Limit. A CB–11 in CALLED state answers GRN in 3 microseconds and stores contents of Limit register in accordance with Y which should consist of two command numerics:<br>$_{11}(LM)_6 \rightarrow (Y)$<br>$_5(LM)_0 \rightarrow (Y + 1)$ |

**TABLE 4.6-3 CB-11 COMMAND CODES: ON-LINE, APPLICABLE TO ALL SLAVE STATES, CONTROL TRANSMIT, CONTROL MESSAGE [OPERATIONS INVOLVING THE COMMUNICATION LINE] (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 026 | LLM | Load Limit register LM.A CB–11 in CALLED state answer GRN in 3 microseconds and loads Y into the Limit register:<br>$Y \rightarrow (LM)$ |
| 034 | CON | Enter CONTROL INTERNAL state. CB–11 in CALLED state upon hearing CON replies GRN in 6 microseconds, disconnects itself from the line switch, and jumps to (PS).<br>Total time 18 microseconds. |
| 042 | SIL | Start Initial Load. Used to start initial loading the Central Processor from the CB–11. CB–11 enters CONTROL MESSAGE state, sends first data character immediately and subsequent characters in response to REQ's. SIL is not transmitted on the line.<br>Timing [20 + 6B] microseconds. |
| 061 | QER | Query no Error. CB–11 in CALLED state upon hearing QER responds GRN if the Error flip-flop [ERF] is in the reset condition. If flip-flop is in set condition, CB–11 responds RED and resets the flip-flop.<br>Timing 4 microseconds. |
| 067 | QIN | Query no Interrupt. CB–11 in CALLED state upon hearing QIN responds GRN if the Interrupt flip-flop [NTF] is in the reset condition. If flip-flop is in set condition, CB–11 responds RED and resets the flip-flop.<br>Timing 4 microseconds. |
| 1nn | 1nn | A command numeric. Sent in pairs following LLM and SLM. Each numeric following LLM, and the first numeric following SLM, are acknowledged in 6 microseconds. The second numeric following SLM is acknowledged in 18 microseconds. |
| 200 + uuu | CALs | A call to the CB–11 itself. A CB–11 in the STANDBY state enters CALLED and answers GRN.<br>Timing 4 microseconds.<br>A CB–11 in the CONTROL TRANSMIT state, upon reading its own call from its memory to the line, enters CONTROL INTERNAL state and begins executing commands at (AC).<br>Timing 8 microseconds.<br>Each CB–11 has exactly one main line call signal. Not to be confused with the five adapter calls read from CB–11 memory and used to call the adapter stations during CONTROL INTERNAL operations. |

**TABLE 4.6–3 CB–11 COMMAND CODES: ON–LINE, APPLICABLE TO ALL SLAVE STATES, CONTROL TRANSMIT, CONTROL MESSAGE [OPERATIONS INVOLVING THE COMMUNICATION LINE] (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 200 +<br>uuu | CALo | A call to some other unit. CB–11 in CALLED state enters STANDBY state.<br>Timing 4 microseconds. |
| 400 +<br>ddd | ddd | A data character. [0 ≤ ddd ≤ 377.] CB–11 in MESSAGE state or CONTROL MESSAGE state places the data character in memory.<br>$ddd \rightarrow ((AC))$,<br>increments (AC),<br>$(AC) + 1 \rightarrow (AC)$<br>and sends REQ.<br>Timing 3 microseconds. |
| none | REQ | Request for next character.<br>Timing 3 microseconds. |

**TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.]**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 000 | NOP | No operation. Program continues. Timing 7 microseconds. |
| 006 | SW1 | Switch to Line 1. Takes 6 milliseconds if on Line 2, 9 microseconds if already on Line 1. |
| 007 | SW2 | Switch to Line 2. Takes 6 milliseconds if on Line 1, 9 microseconds if already on Line 2. |
| 014 | RCV | Receive. Prepares to receive data characters. This command must precede the TRC command which will be used in preparing to instruct the transmitter. |
| 015 | BTU | Block Test Unequal to operand. Compares corresponding characters of the data block beginning at (DS) and the operand block beginning at Y, thus:<br>$((DS) + k) : (Y + k)$ ; also $((DS) + k) \rightarrow (SF)$<br>Testing halts at the first unequal comparison, and the operand address is stored in LM. If no unequal comparison is found, the sequence terminates when the next data address equals (LM). The last operand address is placed in LM. The comparison flip-flops indicate the result of the test.<br>Timing [33 + 16B] microseconds. |
| 016 | TRA | Prepare to Transmit data characters. This command must precede the TRC command which will be used in preparing to instruct the receiver.<br>Timing 7 microseconds. |
| 017 | BTL | Block Test Less than operand. Compares corresponding characters of the data block beginning at (DS) with the operand block beginning at Y, thus:<br>$((DS) + k) : (Y + k)$ ; also $((DS) + k) \rightarrow (SF)$<br>Comparison halts the first time the data character is less than the operand. The operand address is placed in LM. If no data character is less than the corresponding operand character, the sequence terminates when the next data address equals (LM). The last operand address is placed in LM. The comparison flip-flops indicate the result of the test.<br>Timing [33 + 16B] microseconds. |
| 020 | RBH | Convert Row Binary to Extended Hollerith. When key punched cards are read on a parallel reader in row binary mode, the information is converted to the desired form using this opcode. Repeated application of this command will convert information from row binary format to serial format. The serial card code must be expressible as a union of row-functions corresponding to the holes in a card column. |

### TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.] (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| | | The row-function is held in RS and the 8–bit row binary "character" to be converted is held in SF. The Extended Hollerith codes<br>$(Y), (Y + 1), \ldots . (Y + n)$<br>are modified by union of the row-function (RS). Thus, if<br>$(SF)_{7-j} = 1$<br>$(RS) \vee (Y + j) \rightarrow (Y + j)$<br>and if<br>$(SF)_{7-k} = 0$<br>$(Y + k) \rightarrow (Y + k)$<br>The block terminates when the next operand address = (LM). The n may have any value from 0 to 7.<br>Timing [19 + 7B] microseconds. |
| 021 | HRB | Convert Extended Hollerith to Row Binary. Information which must appear on a card in serial or key punched form, but is to be punched with a parallel feed punch in row binary format, must first be converted using this opcode. Will convert to row binary format any serial card code expressible as a union of row-functions corresponding to the holes in a card column.<br>The row-function is held in RS and the row binary "character" for a group of 8 card columns is built up a bit at a time in SF. Whenever the 1–bits of the row-function appear in the Extended Hollerith character, a 1–bit is placed in the row binary image. Eight or fewer Extended Hollerith codes<br>$(Y), (Y + 1), \ldots . (Y + n)$<br>are used to determine bits in (SF) thus, if<br>$(RS) \wedge (Y + j) = (RS)$<br>$1 \rightarrow (SF)_{n-j}$<br>and if<br>$(RS) \wedge (Y + k) \neq (RS)$<br>$0 \rightarrow (SF)_{n-k}$<br>The limit is reached when the next operand address = (LM). The n may have any value from 0 to 7, but note that for $n < 7$ the bits of (SF) loaded are those at the right.<br>Timing [19 + 6B] microseconds. |

## TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.] (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 022 | SLM | Store LM. Stores (LM) in (Y) and (Y + 1):<br>$_{11}(LM)_6 \rightarrow (Y)$<br>$_5(LM)_0 \rightarrow (Y + 1).$<br>Timing 30 microseconds. |
| 023 | BCR | Block Clear. Clears the block of memory starting at Y and continuing until the next operand address = (LM).<br>Timing [19 + 6B] microseconds. |
| 024 | CPI | Construct Print Image. Converts 8 characters into 8 columns of print image. Character code at input must be print roll position. The image occupies the quadrant of memory<br>$XX0000000000_2$<br>to<br>$XX1111111111_2$,<br>with the zero row of print image in<br>$XX0000000000_2$<br>through<br>$XX0000001110_2$<br>where bits 0–3 signify the column group number [columns 1–8 =group 0], and bits 4–9 signify the character code of print roll position. LM should contain in bits 0–3 the proper column group number for the column group to be converted. The hardware performs the following operation for each value of k from 0 to 7:<br>$2^4 * (Y + k) \vee (LM) \rightarrow (LM)$<br>$2^{7-k} \rightarrow (SF)$<br>$(SF) \vee ((LM)) \rightarrow ((LM)).$<br>Thus a 1–bit is united with bit 7–k of column group character $_3(LM)_0$ in row $_9(LM)_5$ of print image. The print image must be constructed in multiples of 8 columns.<br>CPI is given once for each 8 columns.<br>Timing 115 microseconds. |

## TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.] (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 025 | RDC | Read Card. Reads a card row from left to right 8 bits at a time starting with column group $_3Y_0$ and stores the character in Y. Columns 1–8 are column group 0. Block terminates when the next operand address to be accessed equals (LM).<br>Timing [20 + 16B] microseconds. |
| 026 | LLM | Load LM. Loads LM with (Y), (Y + 1)<br>$(Y) \rightarrow {}_{11}(LM)_6$<br>$(Y + 1) \rightarrow {}_5(LM)_0.$<br>Timing 31 microseconds. |
| 027 | POP | Print Or Punch. Reads characters from memory beginning at (Y), to card or printer unit beginning at column group $_3(Y)_0$. Terminates when (AC) = (LM). Last character is left in SF.<br>Timing [20 + 16B] microseconds. |
| 030 | SSF | Store SF. Stores (SF) in Y:<br>$(SF) \rightarrow (Y).$<br>(SF) is not disturbed.<br>Timing 24 microseconds. |
| 031 | LRS | Load RS. Loads (Y) into RS:<br>$(Y) \rightarrow (RS).$<br>Timing 24 microseconds. |
| 032 | LSF | Load SF. Loads (Y) into SF:<br>$(Y) \rightarrow (SF).$<br>Timing 24 microseconds. |
| 033 | SRS | Store RS. Stores (RS) in Y:<br>$(RS) \rightarrow (Y).$<br>Timing 24 microseconds. |

**TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.] (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 035 | USF | Unite with SF. Unites (Y) with (SF):<br>$(SF) \vee (Y) \rightarrow (SF)$.<br>Timing 24 microseconds. |
| 036 | CSF | Compare SF with Operand. Compares (SF) with (Y). If $(SF) < (Y)$, sets Comparison Lower flip-flop. If $(SF) = (Y)$, sets Comparison Equal flip-flop. Otherwise, resets both flip-flops. (SF) is not disturbed.<br>Timing 28 microseconds. |
| 037 | INC | Increment. Increments (LM)<br>$(LM) + 1 \rightarrow (LM)$.<br>(SF) is not disturbed.<br>Timing 10 microseconds. |
| 040 | JMP | Jump. Program branches to Y.<br>$Y \rightarrow (AC)$.<br>Timing 19 microseconds. |
| 041 | BCS | Block Character Switch. Translates a block of characters from one character code to another code using the algorithm $((A)) \rightarrow (A)$. The block of characters to be translated starts at (Y). The switch table to be entered by using (Y) as an address is stored in $0011XXXXXXXX_2$. In operation the hardware performs the following for each value of k from 0 to the limit reached when next operand address = (LM).<br>$(11_2 * 2^8 + (Y + k)) \rightarrow (Y + k)$.<br>(SF) is not disturbed.<br>Timing [19 + 18B] microseconds. |
| 043 | BTF | Block Transfer to destination. Transfers the data block beginning at (DS) to the operand destination area beginning at Y.<br>$((DS)) \rightarrow (Y)$<br>$((DS) + 1) \rightarrow (Y + 1)$, etc |

## TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.] (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| | | The limit is reached when the next data address is equal to (LM). The final address of the destination area is placed in LM. (SF) is not disturbed.<br>Timing [33 + 12B] microseconds. |
| 044 | TRC | Transmit Commands. CB–11 enters CONTROL TRANSMIT state, connects itself to the communication line, and transmits subsequent characters as line commands.<br>Timing 7 microseconds. |
| 045 | JEQ | Jump if Equal. If Comparison Equal flip-flop [EQF] is set, program jumps to Y, otherwise skips two characters.<br>Timing 19 microseconds if jump taken. Otherwise, 9 microseconds. |
| 047 | JLO | Jump if Lower. If Comparison Lower flip-flop [LWF] is set, program jumps to Y, otherwise skips 2 characters.<br>Timing 19 microseconds if jump taken. Otherwise, 9 microseconds. |
| 050 | PAN | Print Alphanumeric. Resets (RS) at next index pulse. Shows ready at all character times 0–62. Remains in effect until PAB or PNU.<br>Timing 8 microseconds. |
| 051 | PNU | Print Numeric. Resets (RS) at next index pulse. Shows ready at even character times only. Remains in effect until PAN or PAB.<br>Timing 8 microseconds. |
| 051 | AS1 | Alternate Stacker 1. Causes alternate stacker 1 to be used for one card only in the card station addressed.<br>Timing 8 microseconds. |
| 052 | PAB | Print Alphabetic. Resets (RS) at next index pulse. Shows ready at odd character times only. Remains in effect until PAN or PNU.<br>Timing 8 microseconds. |

## TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.] (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 052 | AS2 | Alternate Stacker 2. Causes alternate stacker 2 to be used for one card only in the card station addressed.<br>Timing 8 microseconds. |
| 053 | COF | Control Off. Enters STANDBY state, connects itself to the and sends an interrupt.<br>Timing 8 microseconds. |
| 054 | CFD | Continue Feed. Continues feed of cards or paper. In card operations CFD must be given after the last row in order to feed another card. In printing, CFD is used with STT for double spacing. STT alone gives single spacing.<br>Timing 8 microseconds. |
| 055 | SFR | SF Right. Shifts (SF) right one bit<br>${}_7(SF)_1 \rightarrow {}_6(SF)_0$ ; $0 \rightarrow {}_7(SF)_7$.<br>Timing 8 microseconds. |
| 056 | STT | Start. Starts feed of cards or paper. In card operations feeds one card. In printing advances paper one line. See CFD and 10n.<br>Timing 8 microseconds. |
| 057 | SFL | SF Left. Shifts (SF) left one bit<br>${}_6(SF)_0 \rightarrow {}_7(SF)_1$ ; $0 \rightarrow {}_0(SF)_0$.<br>Timing 8 microseconds. |
| 060 | QRD | Query Ready. If the indicator is reset, i.e., if next card row or character print position is ready to be processed, 2 characters are skipped in CB–11 program. Otherwise, jumps to Y.<br>Timing 11 microseconds if ready. Otherwise, 20 microseconds. |
| 061 | QER | Query no Error. If there has been no parity error, incomplete transmission, nor open interlock, [ERF] is reset, CB–11 skips 2 characters. Otherwise jumps to Y.<br>Timing 20 microseconds if ERF set. Otherwise, 11 microseconds. |

**TABLE 4.6-4 CB-11 OPERATION CODES: APPLICABLE TO CONTROL INTERNAL STATE [OPERATIONS NOT INVOLVING THE COMMUNICATION LINE. TIMING SHOWN INCLUDES COMMAND ACCESS, OPERAND ACCESS, IF ANY, AND EXECUTION.] (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 062 | QIL | Query no Interlock. If there is no open interlock on a card machine, program skips 2 characters. Otherwise, jumps to Y.<br>Timing 20 microseconds if interlocked. Otherwise, 11 microseconds. |
| 063 | QCT | Query Complete Transmission. If there has been a complete transmission of a card or print row, program skips 2 characters. Otherwise, jumps to Y.<br>Timing 20 microseconds if incomplete transmission. Otherwise, 11 microseconds. |
| 065 | QIP | Query Index Pulse. If print roll is between position 63 [index] and 65, program skips 2 characters. Otherwise, jumps to Y.<br>Timing 20 microseconds if jump taken. Otherwise, 11 microseconds. |
| 065 | QLR | Query not Last Row. If not received during emitter time of last row, program skips 2 characters. If received during last row emitter time, program jumps to Y.<br>Timing 20 microseconds if last row. Otherwise, 11 microseconds. |
| 10n | 10n | Format channel n. This command numeric following STT [or CFD] causes the printer paper feed to halt at the next hole in format channel n.<br>Timing 8 microseconds. |
| lnn | lnn | A general command numeric. Used in pairs to indicate an address. Time for this command is included in timing for the opcodes. |
| XX0 | CL0 | Call to punch station. See Note. |
| XX1 | CL1 | Call to read station. |
| XX2 | CL2 | Call to read station. |
| XX3 | CL3 | Call to read station. |
| XX4 | CL4 | Call to print station. |
|  |  | Note: The digits XX are factory wired. The standard value is $20_8$. Other values $20 \leq XX \leq 37$ on special order. Calls to card or print stations take 6 microseconds. |
|  | CHP | Character Pulse from printer (RS) + 1 →(RS). |
|  | NDX | Index pulse from printer 0 →(RS). |
|  | DI | Digit pulse from card machine. Resets Ready flip-flop. |
|  | LR | Last Row pulse from card feed. Sets Last Row flip-flop. |

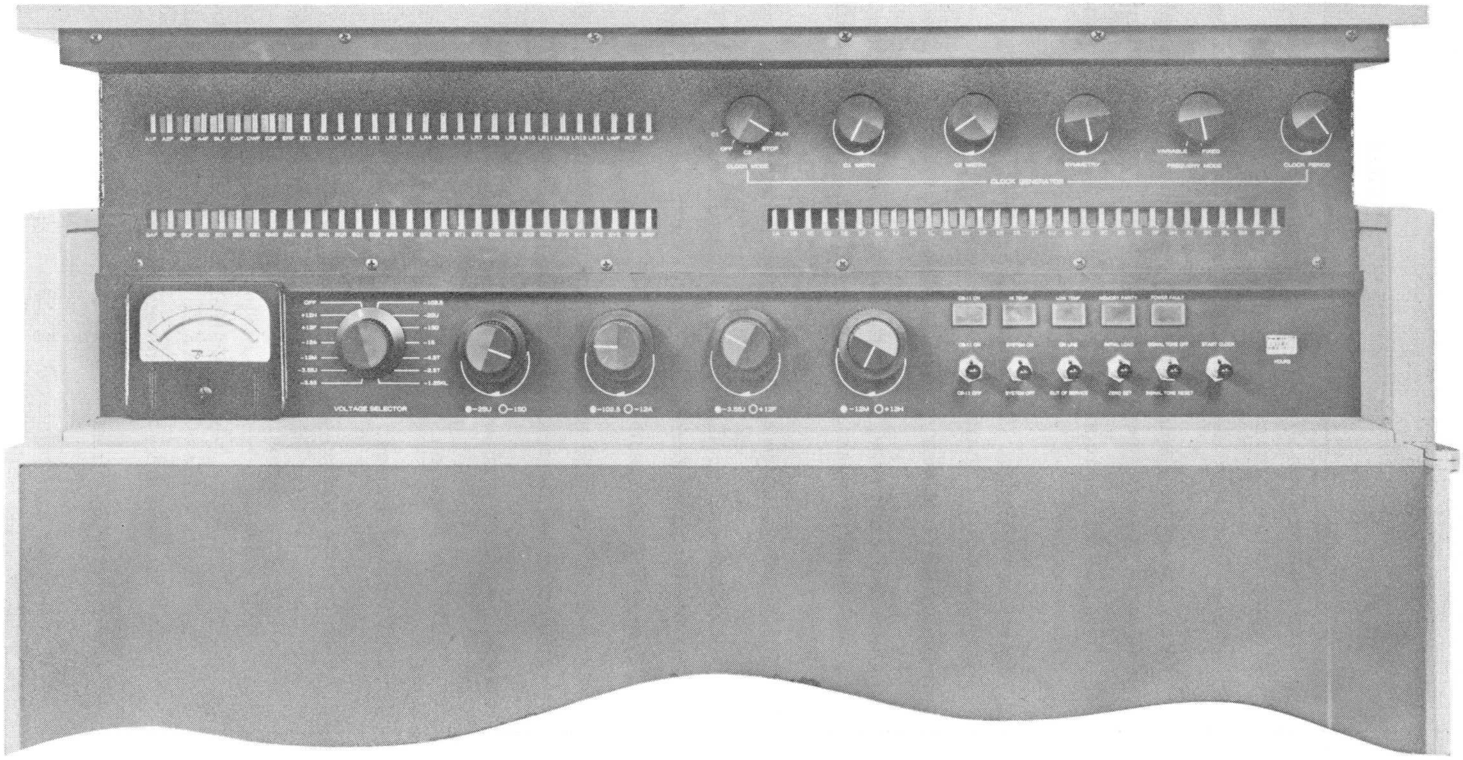


FIGURE 4.6–4 CONTROL BUFFER CONTROL PANEL

**4.6–4 Controls** The Controls of the CB–11 are shown in Figure 4.6–4. The Power switch has three positions: ON, SLAVE, and OFF. When the Power switch is in the SLAVE position, power is controlled by the system power signal. ON and OFF are unconditional. The SYSTEM ON/OFF switch places signals on the system power signal line. The INITIAL LOAD switch is used to start up the CB–11 by reading a card. The ON-LINE/OUT of Service switch is used to connect the CB–11 to the communication line or to disconnect it. In addition to the above items, there is a CB–11 ON light to indicate that power is on and a number of test indicator lights on the service panel. In the event of a memory parity error, power fault, or temperature out of limits, an indicator will be lighted and a tone generator will sound [unless disabled]. After a fault has been corrected, the tone generator must be reset.

Accessory devices are connected to the CB–11 through three types of adapters: the card read adapter used for reading with IBM readers, the card punch adapter used with IBM punches and tabulators, and the line print adapter used with the high–speed printers, LP–10 and LP–11.

**4.6–5 Operations On The Communication Line** Operations on the G–20 Communication System comprise slave operations and control operations. In the slave operations, the CB–11 receives commands from its control device [CALLED state], receives data from its control device [MESSAGE state], or transmits data to its control device [MESSAGE state]. In the control operations, the CB–11 sends commands [CONTROL TRANSMIT], transmits data [CONTROL MESSAGE] or receives data [CONTROL MESSAGE]. In these control operations the CB–11 may control Magnetic Tape, Paper Tape, buffered Line Printers, or other CB–11's in the slave state.

RECEIVE COMMANDS. The CB–11 in general receives commands and queries in the CALLED state. In the STANDBY state it can answer its own call, and in the INSTRUCTED state it can start data transfer. These commands are listed in Table 4.6–3.

RECEIVE DATA [MESSAGE State]. The CB–11 can receive data from the Central Processor at 167,000 characters per second. A typical sequence of line signals exchanged between the Central Processor and the CB–11 might be as shown in Example 4.6–1.

The CB–11 can be thought of as executing the single address program:

| OPCODE | ADDRESS | COMMENTS |
|---|---|---|
| LLM | START | [START] → (LM) |
| SLM | 0004 | (LM) → (DS) |
| RCV | | Prepare to receive |

TRANSMIT DATA [MESSAGE State]. The CB–11 transmits data to the Central Processor at 167,000 characters per second. A typical sequence of line signals for such a transmission is indicated in Example 4.6–2.

TRANSMIT COMMANDS. The CB–11 transmits commands from its own memory while in the CONTROL TRANSMIT state. A sequence of signals to slew a Magnetic Tape could be identical with Example 4.3–2 with the role of the Central Processor being taken by the CB–11.

TRANSMIT DATA [CONTROL MESSAGE State]. The CB–11 can transmit data to another unit on the communication line while acting under control of its own stored program. An example of writing on Magnetic Tape is given in Example 4.6–3. While instructing the tape, the CB–11 would be in the CONTROL TRANSMIT state, and while sending the data, would be in the CONTROL MESSAGE state.

The CB–11 program and line signals during the data transfer would be as shown in Example 4.6–3.

RECEIVE DATA [CONTROL MESSAGE State]. The CB–11 can receive data from another unit on the communication line while operating under control of its own stored program. An example of reading Magnetic Tape is given in Example 4.6–4. The CB–11 will be in the CONTROL TRANSMIT state while instructing the tape and in the CONTROL MESSAGE state during the data transfer.

**EXAMPLE 4.6-1 SEQUENCE OF LINE SIGNALS – CB-11 RECEIVES DATA FROM G-20**

| G–20 | CB–11 | COMMENTS | |
|---|---|---|---|
| CALs | GRN | Call CB–11* | CALLED |
| LLM | GRN | Load LM register | |
| lnn | GRN | Left half of starting address | |
| lnn | GRN | Right half of starting address | |
| SLM | GRN | Store LM register | |
| 100 | GRN | Stores starting address | |
| 104 | GRN | in DS | |
| RCV | GRN | Receive* | INSTRUCTED |
| SDT | REQ | Start data transfer | MESSAGE |
| ddd | REQ | ddd → ((DS)) | |
| ddd | REQ | ddd → ((DS) + 1) | |
| . . . . | . . . . | . . . . | |
| ddd | REQ | ddd → ((DS) + n) | |
| END | | * | STANDBY |

*The state changes on this command. The state shown is that existing after the command. See Table 4.6–3.

### EXAMPLE 4.6–2 SEQUENCE OF LINE SIGNALS — CB–11 TRANSMITS TO G–20

| G–20 | CB–11 | COMMENTS | |
|---|---|---|---|
| CALs | GRN | Call CB–11* | CALLED |
| LLM | GRN | Load LM register | |
| lnn | GRN | Left half of starting address | |
| lnn | GRN | Right half of starting address | |
| SLM | GRN | Store LM register | |
| 100 | GRN | Stores starting address | |
| 104 | GRN | in DS | |
| TRA | GRN | Transmit* | INSTRUCTED |
| SDT | | Start data transfer* | MESSAGE |
| REQ | ddd | ((DS)) → ddd | |
| REQ | ddd | ((DS) + 1) → ddd | |
| . . . . | . . . . | . . . . | |
| REQ | ddd | ((DS) + n) → ddd | |
| END | | * | STANDBY |

*The state changes on this command. The state shown is that existing after the command. See Table 4.6–3.

## EXAMPLE 4.6–3 CB–11 PROGRAM AND LINE SIGNALS — TRANSMIT TO MT–10

| PROGRAM | | COMMENTS | |
|---|---|---|---|
| OPCODE | ADDRESS | | |
| LLM | START | [START] → (LM) | CONTROL INTERNAL |
| SLM | 0004 | (LM) → (DS) | |
| TRA | | Prepare to transmit data | |
| TRC | | Transmit commands* | CONTROL TRANSMIT |
| LINE SIGNALS | | | |
| CB–11 | MT–10 | | |
| CAL | GRN | Call tape | |
| RCV | GRN | Receive | |
| SDT | REQ | Start data transfer* | CONTROL MESSAGE |
| ddd | REQ | | |
| . . . . | . . . . | . . . . | |
| ddd | REQ | | |
| END | | CB–11 takes next command at (AC)* | CONTROL INTERNAL |

* The state changes on this command. The state shown is that existing after the command. See Table 4.6–3.

## EXAMPLE 4.6–4 CB–11 PROGRAM AND LINE SIGNALS – RECEIVE FROM MT–10

| PROGRAM | | | |
|---|---|---|---|
| OPCODE | ADDRESS | COMMENTS | |
| LLM | START | [START] → (LM) | CONTROL INTERNAL |
| SLM | 0004 | (LM) → (DS) | |
| RCV | | Prepare to receive data | |
| TRC | | Transmit commands* | CONTROL TRANSMIT |
| LINE SIGNALS | | | |
| CB–11 | MT–10 | | |
| CAL | GRN | Calls tape | |
| TRA | GRN | Transmit | |
| SDT | | Start data transfer* | CONTROL MESSAGE |
| REQ | ddd | ddd → ((DS)) | |
| REQ | ddd | ddd → ((DS) + 1) | |
| . . . . | . . . . | . . . . | |
| REQ | ddd | ddd → ((DS) + n) | |
| END | | CB–11 goes to CONTROL INTERNAL and takes next command at (AC). | |

* The state changes on this command. The state shown is that existing after the command. See Table 4.6–3.

INITIAL LOAD G–20 FROM CB–11. The G–20 is initially loaded from the CB–11 using a sequence similar to that shown in Example 4.6–2, "CB–11 transmits to G–20". In this case, however, the CB–11 will be in the CONTROL MESSAGE state. The block of characters to be transmitted should be stored in successively ascending locations in CB–11 memory and the starting address of the block should be in DS. The G–20 should be prepared by operating its INITIAL LOAD lever switch and START CLOCK lever switch. Other equipment in the system should be in STANDBY.

The exchange shown in Example 4.6–5 will take place.

## EXAMPLE 4.6–5 CB–11 PROGRAM AND LINE SIGNALS — INITIAL LOAD G–20, n WORDS

| CB–11 | G–20 | COMMENTS | |
|---|---|---|---|
| [TRA] | | Prepare to transmit data | CONTROL INTERNAL |
| [TRC] | | Transmit commands * | CONTROL TRANSMIT |
| [SIL] | | Start initial load* | CONTROL MESSAGE |
| ddd | REQ | ddd → ${}_{31}(64)_{24}$ in G–20 | |
| ddd | REQ | ddd → ${}_{23}(64)_{16}$ | |
| . . . . | . . . . | . . . . | |
| ddd | REQ | ddd → ${}_{7}(64 + n-1)_{0}$ | |
| END | | * | CONTROL INTERNAL |
| COF | | CB–11 enters STANDBY state.<br>G–20 begins executing at 65 = 101$_8$. | |

Note: The signals shown in brackets are internal commands the CB–11 is executing preparatory to the transfer of data.

* The state changes on this command. The state shown is that existing after the command. See Table 4.6–3.

**4.6–6 Operations Off The Communication Line: Input/Output** While operating off the communication line, the CB–11 can read cards, punch cards, and line print, as well as perform data rearrangements. All of this is carried out under control of the CB–11's internally stored program. Since transfers of data between the CB–11 and parallel card equipment take place in row binary only, data rearrangement takes place between rows while reading or punching Hollerith. During line printing operations with the LP–10 and LP–11 the print image is constructed before each line is printed.

INITIAL LOAD. Initial loading of the CB–11 is accomplished by reading in one row binary card which is able to store 120 8–bit characters. With power on the card reader, and power on the CB–11, the row binary load card is placed in the hopper of the card reader and the INITIAL LOAD lever switch of the CB–11 is operated. The contents of the load card will be read into locations 0–119 of the CB–11 memory, whereupon the CB–11 will enter the CONTROL INTERNAL state, and begin executing the program at the location given by (0), (1). The movement of data in this operation is given in Table 4.6–1, "Initial Load CB–11".

READ CARDS. Card reading with the CB–11 takes place a single row at a time. The time between rows may be used to convert row binary to Extended Hollerith or perform other internal operations. A possible program segment for reading Hollerith cards is shown in Example 4.6–6. The CB–11 is in CONTROL INTERNAL.

PUNCH CARDS. Punching cards proceeds a row at a time the same as reading. The time between rows can be used for conversion from Hollerith to row binary. The program is similar to that for reading shown in Example 4.6–6.

## EXAMPLE 4.6-6 CB-11 PROGRAM - READ CARDS AND CONVERT TO EXTENDED HOLLERITH

Input from a row is stored in ROW to ROW + 9. Output in Extended Hollerith is stored in HOL to HOL + 79. Row functions are stored starting at ROFUN.

| LOCATION | OPCODE | ADDRESS | COMMENTS |
|---|---|---|---|
| | CL1 | | Calls card reader. |
| | STT | | Read feed starts. |
| | LLM | HOL + 80 | Clears output area. |
| | BCR | HOL | |
| | LLM | ROFUN | Initializes row function |
| | SLM | E1 + 1 | address. |
| E2 | QRD | E2 | Test ready loop. |
| | LLM | ROW + 10 | Sets limit for input block. |
| | RDC | ROW | Reads in row from card. |
| E1 | LRS | ROFUN | Loads row function into RS. |
| | LSF | ROW | Converts first 8 bits to |
| | LLM | HOL + 8 | Hollerith and unites |
| | RBH | HOL | with output block. |
| | LSF | ROW + 1 | |
| | LLM | HOL + 16 | Converts second 8 bits. |
| | RBH | HOL + 8 | |
| | . . . . | . . . . | . . . . |
| | LSF | ROW + 9 | |
| | LLM | HOL + 80 | Converts last 8 bits |
| | RBH | HOL + 72 | |
| | QLR | EXIT | Done with card. |
| | LLM | E1 + 1 | |
| | INC | | Loop back for next row. |
| | SLM | E1 + 1 | |
| | JMP | E2 | |

## EXAMPLE 4.6-7 CB-11 PROGRAM — SYNCHRONIZE FREE WHEEL PRINTING

| OPCODE | ADDRESS | COMMENTS |
|---|---|---|
| QRD | Loc [QRD] | Query ready loop |
| SRS | TEMP 1 | Store current row number |
| LSF | TEMP 1 | Load row number into SF |
| SFR | | Shift right |
| SFR | | Shift right |
| USF | NQUAD | (NQUAD) = 01XX0000 |
| | | Now (SF) = $01XXR_5R_4R_3R_2$ |
| SSF | Loc START 1 | START 1 = First character of START address |
| SSF | Loc LIMIT 1 | LIMIT 1 = First character of LIMIT address |
| LSF | TEMP 1 | |
| SFL | | Shift left |
| SFL | | Shift left |
| SFL | | Shift left |
| SFL | | Shift left |
| USF | N ZERO | (N ZERO) = 01000000 |
| SSF | Loc START 2 | START 2 = Second character of START address |
| USF | N LIM | (N LIM) = 01001111 |
| SSF | Loc LIMIT 2 | LIMIT 2 = Second character of LIMIT address |

LINE PRINTING ON LP–10, LP–11. The CB–11 prints on an LP–10 or LP–11 by reading successive rows of a ready-prepared print image. The print image for each row is constructed before the row can be printed. In the "fixed wheel" method of printing the program halts after construction of print image and upspacing, and waits for the index pulse at printer row 63. Then it delivers the print image 8 bits at a time from left to right starting with row 0 and ending with row 62. The program for one row can be as follows:

| OPCODE | ADDRESS | COMMENTS |
|---|---|---|
| QIP | Loc [QIP] | Query loop |
| LLM | LIMIT | [LIMIT] → (LM) |
| QRD | Loc [QRD] | Query loop |
| POP | START | [START] → (AC) |

In the "free wheeling" method of printing, the delivery of the print image begins immediately after image construction and paper upspacing have been completed. That row of the image corresponding to the current print roll position is then printed. Use is made here of the fact that the hardware register, RS, keeps in step with the print roll so that (RS) = row position number. Any print command PAN, PAB, or PNU synchronizes (RS) with the print roll at the next index pulse and (RS) remains synchronized automatically. A possible method for setting [START] and [LIMIT] in the above program is shown below. We define the following binary numbers:

XX = An arbitrary quadrant of memory

$R_5R_4R_3R_2R_1R_0$ = Print roll row time or character code

cccc = Column group number. Columns 1–8 are group 0, columns 113–120 are group 14.

Each 8–bit group into which the image is to be broken up, is stored in location XX $R_5R_4R_3R_2R_1R_0$ cccc. The quantities [START] and [LIMIT] in the above program must be established at the beginning of each print cycle so that the proper row of the print image is being delivered to the printer. To achieve this we must have:

[START] = XX $R_5R_4R_3R_2R_1R_0$ 0000

and

[LIMIT] = XX $R_5R_4R_3R_2R_1R_0$ cccc

where $R_5R_4R_3R_2R_1R_0$ = $_5(RS)_0$ and

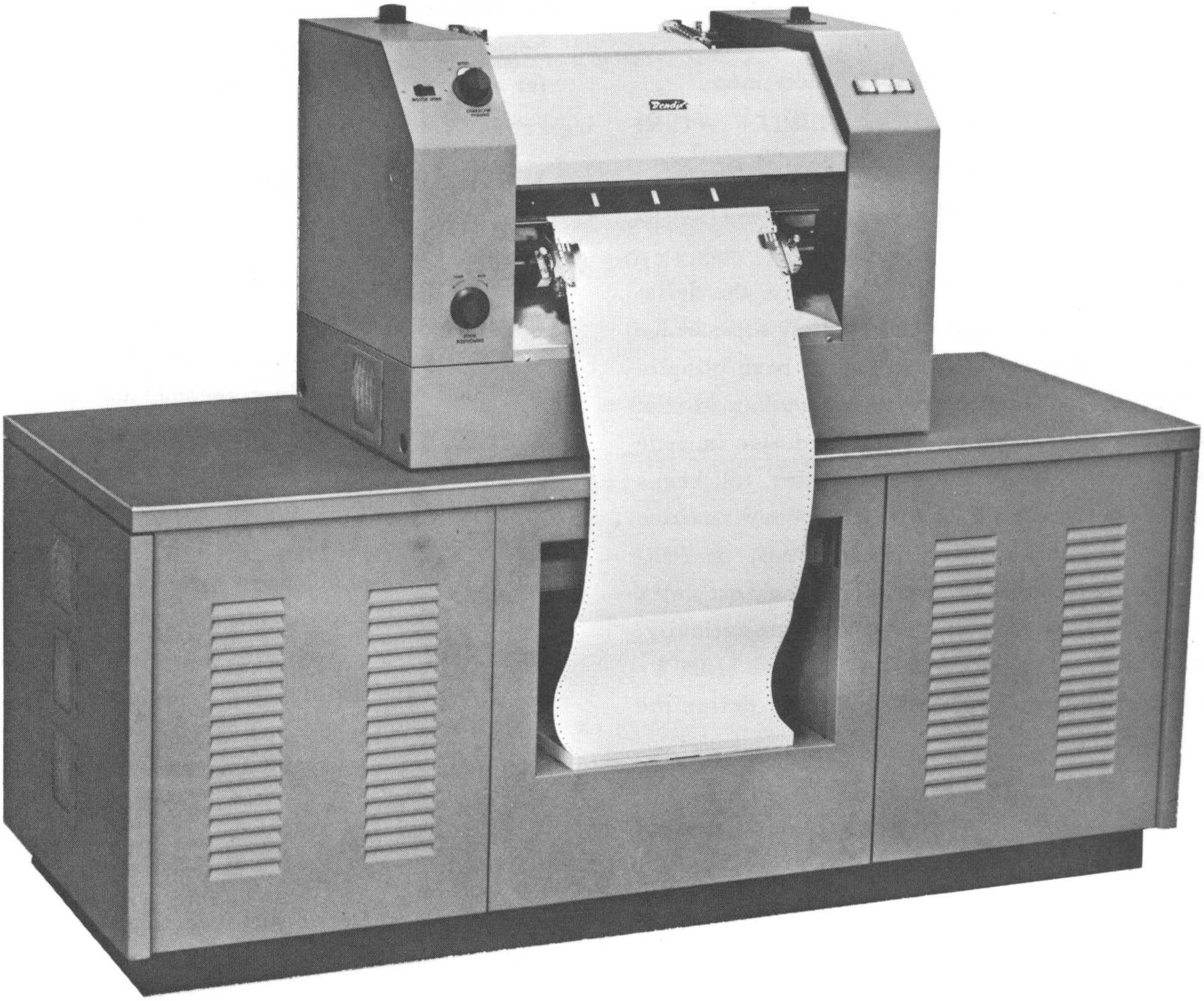
cccc = maximum column group number plus 1, that is, 15 for 120 column printing, 9 for 72 column printing, etc.

The commands SRS, LSF, SFL, SFR, USF, SSF, can be used to create and store these limit values at the beginning of the print cycle. One possible program for accomplishing this is shown in Example 4.6–7.

## SECTION 4.7 BUFFERED LINE PRINTER, LP–12

The LP–12 is a fully buffered printer operating within the Bendix Digital Communication System. It can accept a line of up to 120 characters at any rate up to 67,000 characters per second, and an instruction for paper feed. The printer then executes the print and paper advance sequentially. Maximum effective rate is 800 lines per minute for a 64–character alphabet, single spaced. Printing with a 48–character alphabet, single spaced, proceeds at rates up to 1,000 lines per minute. Printing of numerics can take place at up to 1,000 lines per minute with multiple spacing of paper. The 64–character alphabet consists of 4 sectors of 16 characters each, with a nominal 15 millisecond interval per sector. A 48–character alphabet occupies 3 of these segments, and the 16–character numeric "alphabet" is one of these 3 sectors. Single spacing of paper takes place during one sector time. The logic of the printer is arranged so as to process only those sectors of the print roll which contain characters to be printed. This permits upspacing of paper during the passage of unused sectors of the print roll, with resultant increase in effective speed. The LP–12 can remember a print command received during paper feed or a paper feed command received during printing. This printer handles forms 4–19 inches wide, and up to 22 inches deep, perforated for sprocket drive. It will also print on card stock. Spacing of characters is 10 per inch horizontally and 6 per inch vertically, equivalent to pica type.

FIGURE 4.7-1 BUFFERED LINE PRINTER, LP-12



A continuously revolving print roll carries 120 complete sets of printing characters, each set occupying one circumferential track on the print roll. A line of print is created by driving the paper and ribbon against the print roll by means of a set of individually timed hammers, one for each print column. Timing of each hammer determines the character to be printed in that column. The paper remains stationary during printing and is upspaced after a complete line has been printed.

Physical arrangement of the LP–12 is shown in Figure 4.7–1. The print head, 26 inches high, 35 inches wide, 22 1/2 inches deep, and weighing 350 pounds, is mounted on top of the printer control which is 30 1/2 inches high, 71 inches wide, 30 1/2 inches deep, and weighs 800 pounds. The equipment draws 2.0 KVA of 115-volt, 60-cycle, single-phase power.

**4.7–1 Organization of the LP–12** The block diagram in Figure 4.7–2 shows the principal functional elements of the LP–12. All incoming communication enters the unit via the Line register. Command characters are decoded and appropriate signals sent to the Print Control or Paper Feed Control. For purposes of command and control, paper feed and print are separate stations and are separately addressed. Request signals, answers to queries, and interrupt signals are sent out via reply logic from either control station to the line. Character pulses instructing Print Control when to print are derived from the print roll position.

Data characters arriving at the Line register as 8–bit characters plus flag and parity are stripped of all but bits 0–5, which are sent to the Print Buffer. Print characters are of the form P100XXXXXX where P is the parity bit, the 1–bit is the data flag, and the first two bits of the digit proper are zeros. The six bits shown as XXXXXX represent the print roll position and are used to set one of the 64 cores in the Print Buffer for the column in which the character is to be printed. The first two of these six bits are also the sector indication and are noted by Sector Control and used to set one of the four Sector flip-flops. When one line of print has been delivered to the LP–12, there will be a print image of 120 columns and 64 rows, with a single 1–bit in each column in which a character is to be printed. See Figure 4.7–3. Also the Sector flip-flop will be set for any sector in which there are any characters to be printed. The characters delivered to the printer are used to set up the print image from left to right.

The characters for one line of print may come from one or more blocks of data. If more than 120 characters are sent, the excess over 120 will be lost. Characters can be loaded during paper advance. Printing begins at the first character position of a sector and proceeds by delivering the contents of the corresponding row of the image to the hammer drivers. This row of the image is cleared to zero. At the completion of a sector, the Sector flip-flop is likewise cleared. When all Sector flip-flops show "reset", the printing is complete and paper feed can begin.

**4.7–2 Printing** The standard LP–12 alphabet is shown in Table 4.7–1. It will be noted that alphabetic printing requires sectors 0–1, while numeric printing requires sector 2 only; alphanumeric printing without special characters uses sectors 0, 1, 2. A character 400 received from the line is not interpreted as a print position and does not set a bit in the print image or set a Sector flip-flop, but merely causes one column to be skipped and left blank.

The states which are defined for the LP–12 Print Control are:

OUT of Service. The Print Control is OUT of Service whenever the Paper Feed Control is OUT of Service.

STANDBY. On-line and able to hear calling signals.

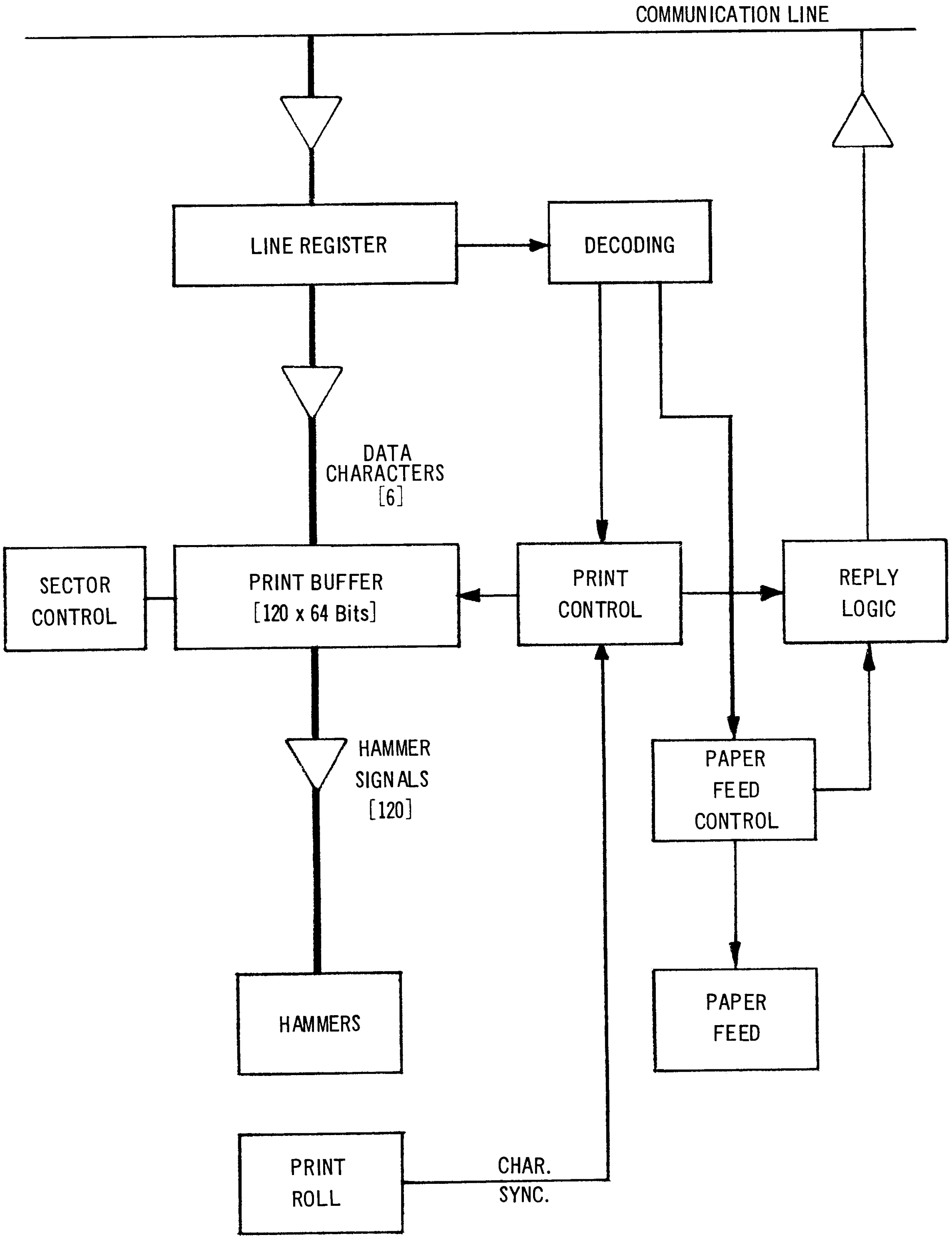
CALLED. Alerted and able to execute commands and answer queries.

BUSY-ALERT. Signifies unit is tied up, but is not transmitting or receiving on the line. Can answer queries, but cannot execute commands.

BUSY-QUIET. Signifies that unit in BUSY-ALERT has heard a call to another unit.

INSTRUCTED. Unit has been set up to perform a data transfer but data transfer has not started, Receipt of its own call will return the unit to CALLED and cancel the instructions previously received.

FIGURE 4.7-2 SIMPLIFIED BLOCK DIAGRAM, LP-12 BUFFERED PRINTER



MESSAGE. Unit is receiving data characters and sending REQ's.

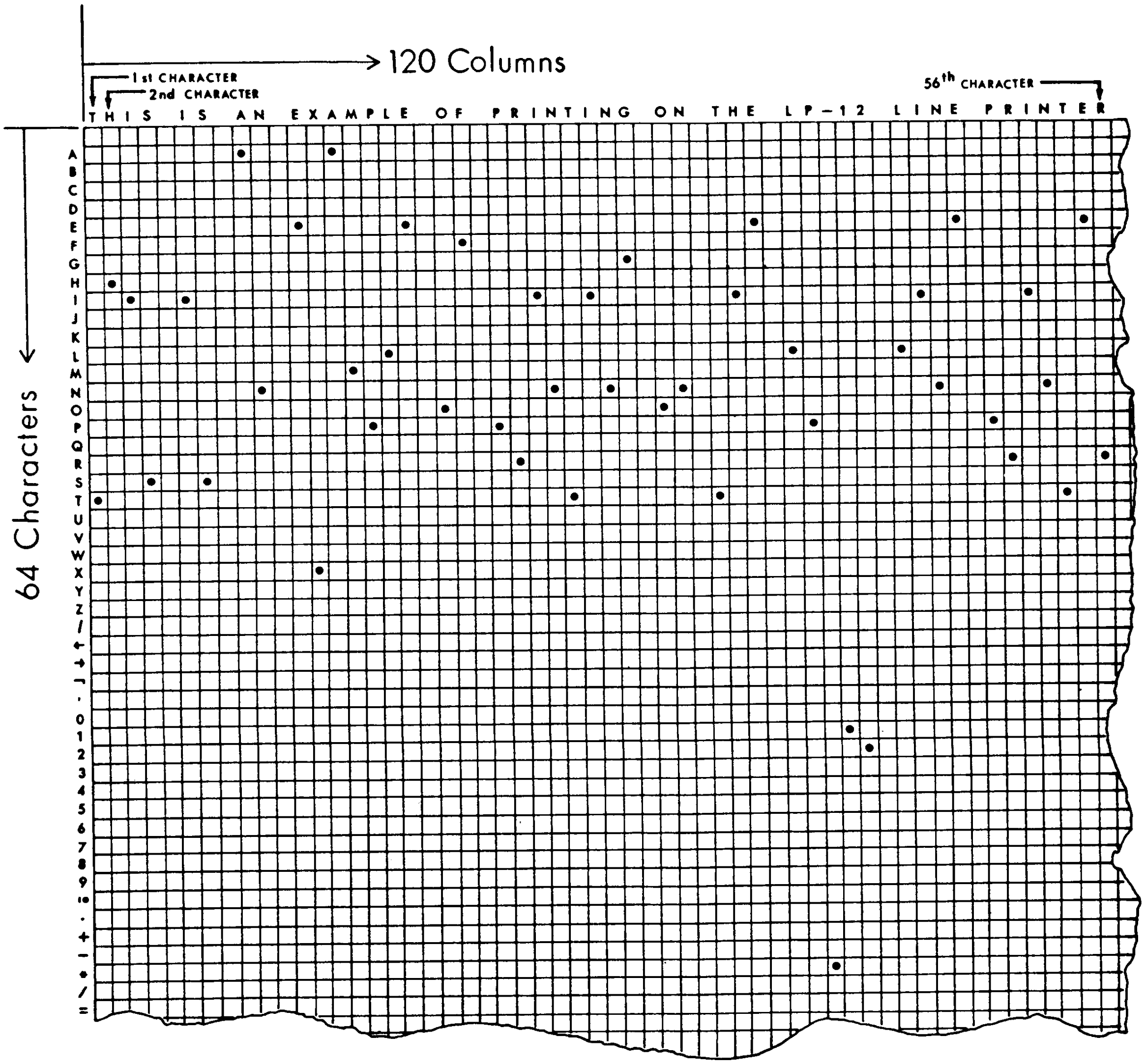
The line commands applicable to the Print Control of the LP–12 are shown in Table 4.7–2.

**TABLE 4.7–1 THE LP–12 ALPHABET [DATA CODES]**

| SECTOR 0 | | SECTOR 1 | | SECTOR 2 | | SECTOR 3 | |
|---|---|---|---|---|---|---|---|
| CODE | LETTER | CODE | LETTER | CODE | LETTER | CODE | LETTER |
| 400 | not used | 420 | P | 440 | 0 | 460 | = |
| 401 | A | 421 | Q | 441 | 1 | 461 | ∨ |
| 402 | B | 422 | R | 442 | 2 | 462 | ≠ |
| 403 | C | 423 | S | 443 | 3 | 463 | ∧ |
| 404 | D | 424 | T | 444 | 4 | 464 | < |
| 405 | E | 425 | U | 445 | 5 | 465 | $ |
| 406 | F | 426 | V | 446 | 6 | 466 | > |
| 407 | G | 427 | W | 447 | 7 | 467 | ; |
| 410 | H | 430 | X | 450 | 8 | 470 | ( |
| 411 | I | 431 | Y | 451 | 9 | 471 | [ |
| 412 | J | 432 | Z | 452 | 10 | 472 | ] |
| 413 | K | 433 | \| | 453 | · | 473 | ) |
| 414 | L | 434 | ← | 454 | + | 474 | ↓ |
| 415 | M | 435 | → | 455 | – | 475 | ↑ |
| 416 | N | 436 | ¬ | 456 | * | 476 | : |
| 417 | O | 437 | , | 457 | / | 477 | ' |

Note: The print roll position [in octal] is the same as the last 2 digits of the code. There are two unused spaces between row $77_8$ and row 0. The index pulse which resets the row address occurs after row $77_8$, but before row 0.

## FIGURE 4.7-3 PRINT IMAGE, LP-12


120 Columns
1st CHARACTER
2nd CHARACTER
56th CHARACTER
THIS IS AN EXAMPLE OF PRINTING ON THE LP-12 LINE PRINTER
64 Characters
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z / ← → ¬ . 0 1 2 3 4 5 6 7 8 9 " . + - * / =

## TABLE 4.7-2 LP-12 PRINT CONTROL COMMAND CODES

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 002 | GRN | Green. A response the unit gives to calls and queries. |
| 003 | RED | Red. A response the unit gives to queries. |
| 004 | END | End. Unit in MESSAGE state enters STANDBY state. Retains next column address if fewer than 120 characters have been loaded. No reply.<br>Timing 10 microseconds. |
| 005 | ERR | Error. Signifies the end of a data block containing an error. Otherwise, similar to END.<br>Timing 10 microseconds. |
| 010 | SDT | Start Data Transfer. Unit in INSTRUCTED state enters MESSAGE state, sends REQ.<br>Timing 15 microseconds. |
| 014 | RCV | Receive. Unit in CALLED state enters INSTRUCTED state and replies GRN.<br>Timing 10 microseconds. |
| 050 | CLB | Clear Buffer. Unit in CALLED state will answer GRN, enter BUSY-ALERT, clear Print Buffer, and initialize Print Control. Upon completion returns to CALLED if in BUSY-ALERT, or to STANDBY if in BUSY-QUIET.<br>Timing 10 microseconds for answer.<br>Clearing starts at next quadrant mark and takes one revolution. |
| 051 | PRT | Print. Unit in CALLED state will answer GRN, enter BUSY-ALERT and deliver the contents of the buffer to the print hammers, starting at the beginning of the next sector.<br>Timing of answer 10 microseconds.<br>Printing can take 1/4, 1/2, 3/4 or 1 revolution.<br>If this command is received during paper advance, execution will be delayed until paper has stopped. |
| 052 | PRI | Print and Interrupt. Same as PRT, except interrupts upon completion of task. |

## TABLE 4.7–2 LP–12 PRINT CONTROL COMMAND CODES (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 060 | QRD | Query Ready. Unit in CALLED state replies GRN if ready to accept data block or ready to accept print command. Answers RED in BUSY–ALERT, or if interlocked, engaged in power turn-on cycle, or has parity error.<br>Timing 10 microseconds. |
| 061 | QER | Query no Error. Unit in CALLED or BUSY–ALERT state answers GRN if there has been no error since last QER. Resets QER flip-flop. Answers RED otherwise.<br>Timing 10 microseconds. |
| 062 | QIL | Query no Interlock. Unit in CALLED or BUSY–ALERT state answers GRN if there is no open interlock. Answers RED if there is an open interlock. QIL resets the QIL flip-flop if the open interlock has been corrected.<br>Timing 10 microseconds. |
| 063 | QCO | Query no Column Overflow. Unit in CALLED or BUSY–ALERT answers GRN if there has been no column overflow, i.e., fewer than 121 characters have been delivered to the buffer. Otherwise answers RED. Resets QCO flip-flop.<br>Timing 10 microseconds. |
| 067 | QIN | Query no Interrupt. Unit in CALLED or BUSY–ALERT state answers GRN if unit has not interrupted since last QIN. Answers RED otherwise. Resets QIN flip-flop.<br>Timing 10 microseconds. |
| 100–<br>177 | . . . . | Not defined for LP–12 Print Control. |
| 200 +<br>uuu | CALs | A call to the unit itself. A unit in STANDBY goes to CALLED. Unit in BUSY–QUIET goes to BUSY–ALERT. Unit in INSTRUCTED goes to CALLED. Answers GRN. [Paper Feed and Print Control have distinct calls.]<br>Timing 10 microseconds. |
| 200 +<br>uuu | CALo | A call to some other unit. Print control in CALLED state goes to STAND–BY Print Control in BUSY–ALERT goes to BUSY–QUIET, No answer.<br>Timing 10 microseconds. |

**TABLE 4.7–2 LP–12 PRINT CONTROL COMMAND CODES (Continued)**

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 400 – 477 | ddd | Data characters. Unit stores image of character and sends REQ for next character.<br>Timing 10 microseconds.<br>If 121 or more characters are received, QCO flip-flop is set. Characters beyond 120 are lost. Data character also sets Sector flop-flip. |
| 500 – 777 | | Not defined for LP–12. LP–12 replies REQ, but executes no other action. |
| none | INT | An interrupt transmitted over the interrupt line. |
| none | REQ | A request transmitted over the request line. |
| . . . . | . . . | Any character with a parity error is stored as a data character. QER flip-flop is set. |

A typical sequence of line signals for printing a line of 120 characters directly from the memory of a control device is shown in Example 4.7–1.

The control device queries the Print Control until it gets a ready signal and then loads the data block. If the characters to be printed in a single line come from more than one block, the sequence CAL, RCV, SDT, ddd, . . ., ddd, END is repeated as many times as necessary. As data are loaded into the buffer, a record is kept of those print roll sectors which contain one or more characters to be printed. After the buffer is loaded, the Print Control waits until the paper has stopped moving before executing the command to print. The row address of the Print Buffer is set to correspond with the print roll sector appearing next under the print hammers, and the contents of that sector of the buffer are printed. An indicator is reset to show that that sector of the buffer is now empty, and Print Control now checks to see if there are any more non-empty sectors. Printing continues as long as there are non-empty sectors [at most 4 sector times or 60 milliseconds]. As soon as all sectors are empty, printing stops and paper feed can begin to move the paper again.

When the control device is not to supply the characters but is merely commanding that they be sent to the printer, the device to supply the characters is called and instructed, after RCV but before SDT. See Example 4.7–1.

At each index pulse time the row address carried in the Print Control is reset to zero. See notes in Table 4.7–1.

**4.7–3 Paper Feed** Paper feed in the LP–12 is controlled by command only. An "advance to top of form" is also provided which makes use of a paper tape loop for positioning after the printing form has been loaded in the prescribed fashion. All other control is under computer command "advance n lines" [n < 64]. The Paper Feed Control is considered to be a separate communication station from the Print Control and has a separate calling signal. The line commands which pertain to the Paper Feed Control are shown in Table 4.7–3.

A typical sequence of line signals for paper spacing is shown in Example 4.7–2.

The program in Example 4.7–2 makes use of the interrupt method for determining readiness. A program for CB–11 or DC–11 would need to use a test ready loop because these units have no interrupt facility.

The control device calls the Paper Feed Control and queries whether it is ready to accept a paper advance command. As soon as a GRN reply is received the upspacing command and number of lines can be given.

Because the portion of the form which is in print position at any given moment is obscured by the printer ribbon, it is necessary to use an indirect method for establishing the "top of form"· A paper tape loop traveling synchronously with the paper drive sprockets is used for this purpose. The mode of operation is depicted in Figure 4.7–4.

## EXAMPLE 4.7–1 SEQUENCE OF LINE SIGNALS — LP–12 PRINT CONTROL

| G–20 | LP–12 | REMARKS |
|---|---|---|
| CALs | GRN | Print Control enters CALLED state. |
| CLB | GRN | LP–12 clears buffer. |
| QRD | RED | If not ready, QRD is repeated. |
|  | GRN | If ready, program continues. |
| RCV | GRN | Receive. |
| . . . . | . . . . | [Instructions to transmitter other than G–20 can be supplied here.] |
| SDT | REQ | Start Data Transfer. LP–12 enters MESSAGE state. |
| ddd | REQ | [These characters can also be supplied by a transmitter other than the G–20 if one has been instructed as indicated above.] |
| . . . . | . . . . |  |
| . . . . | . . . . |  |
| ddd | REQ |  |
| END |  | LP–12 enters STANDBY state. |
| CALs | GRN | Print Control enters CALLED state. |
| QRD | RED | If not ready, branch to error routine. |
|  | GRN | If ready, program continues. |
| PRI | GRN | Print Control enters BUSY state. |
|  | INT | LP–12 interrupts to signal completion. Returns to STANDBY. |

## TABLE 4.7-3 LP-12 PAPER FEED CONTROL COMMAND CODES

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 002 | GRN | A response the Paper Feed gives to commands and queries. |
| 003 | RED | A response the Paper Feed gives to queries. |
| 011 | OUT | Go OUT of Service. Paper Feed in CALLED or BUSY–ALERT state answers GRN, and goes OUT of Service. This command shuts down both the Paper Feed and the Print Control.<br>Timing 10 microseconds. |
| 055 | API | Advance Paper and Interrupt. When this command is received in CALLED state prior to an advance paper command 1nn, an interrupt will be sent at completion of paper advance. Answers GRN.<br>Timing 10 microseconds. |
| 056 | TFM | Top of Form. Paper Feed in CALLED state enters BUSY–ALERT and advances to top of form [provided form has been properly loaded]. Answers GRN.<br>Timing for answer 10 microseconds.<br>Timing for upspacing 15 milliseconds for first space, 15 milliseconds for each additional 2 spaces.<br>Unit in BUSY–ALERT does not reply to TFM. |
| 057 | TFI | Top of Form and Interrupt. Same as TFM except LP–12 interrupts upon completing the task. |
| 060 | QRD | Query Ready. Unit in CALLED state replies GRN if there are no errors or interlocks.Unit in BUSY–ALERT state replies RED.<br>Timing 10 microseconds. |
| 062 | QIL | Query no Interlock. Unit in CALLED or BUSY–ALERT state replies GRN if there are no open interlocks. Replies RED if there is an open interlock due to paper jam, or out of paper, etc.<br>Timing 10 microseconds. |
| 067 | QIN | Query no Interrupt. Unit in CALLED or BUSY–ALERT state responds GRN if there has been no interrupt since the last QIN. Replies RED otherwise. This query resets the QIN flip-flop.<br>Timing 10 microseconds. |

## TABLE 4.7-3 LP-12 PAPER FEED CONTROL COMMAND CODES (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 1nn | 1nn | Paper Feed Control in CALLED state answers GRN, enters BUSY-ALERT, and advances paper nn lines.<br>Timing of reply 10 microseconds.<br>Timing of paper advance is 15 milliseconds for first line and 15 milliseconds for each additional 2 lines.<br>If 1nn is preceded by API, the paper advance will be followed by an interrupt. |
| 200 + uuu | CALs | Call to the Paper Feed itself. Unit in STANDBY state goes to CALLED state. Unit in CALLED state stays in CALLED state. Unit in BUSY-QUIET goes to BUSY-ALERT. Unit in BUSY-ALERT stays in BUSY-ALERT. Answer GRN.<br>Timing 10 microseconds. |
| 200 + uuu | CALo | A call to some other unit. Unit in CALLED state returns to STANDBY. Unit in BUSY-ALERT goes to BUSY-QUIET.<br>Timing 10 microseconds. |
| none | INT | An interrupt sent on the interrupt line signals completion of a paper advance or top of form command. |

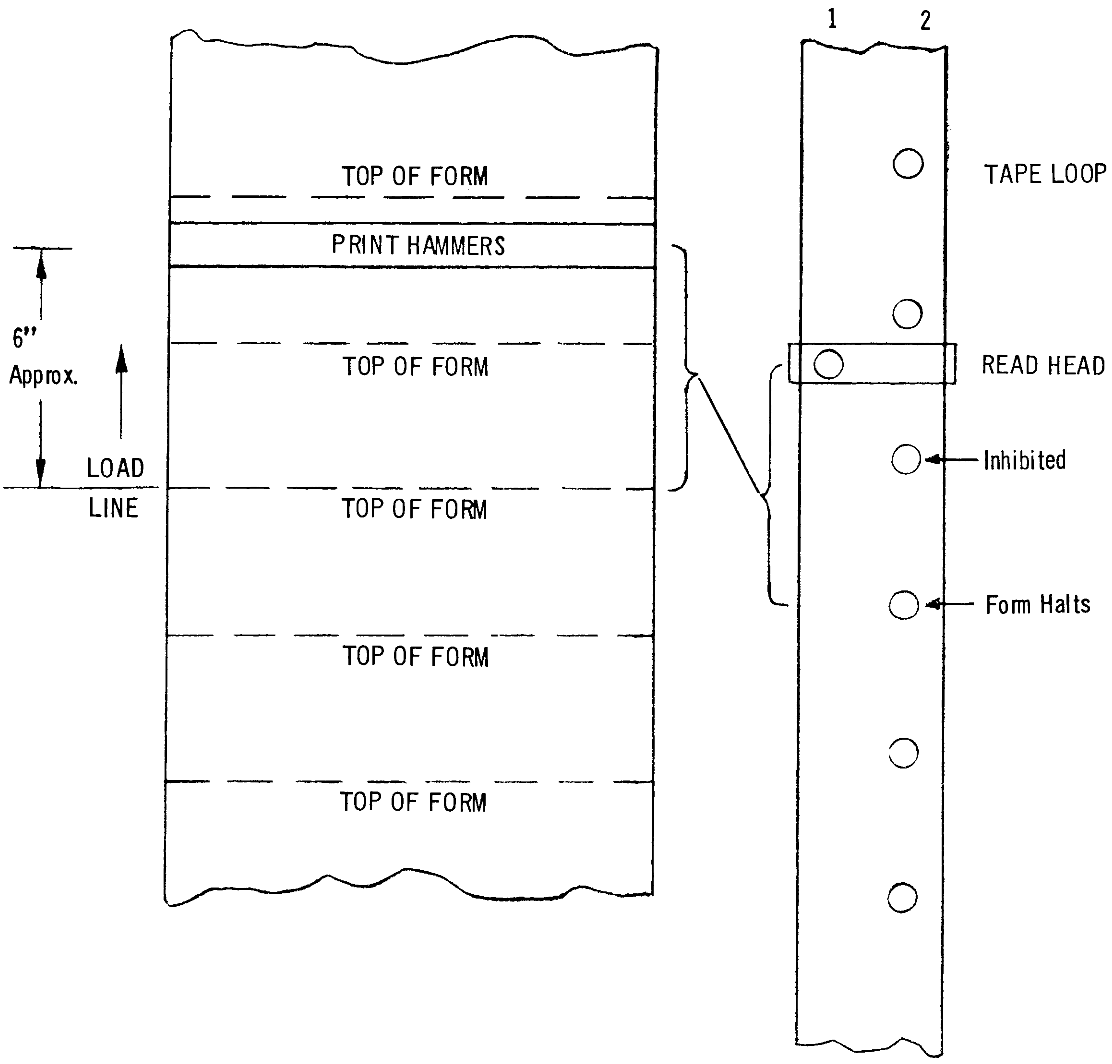
## EXAMPLE 4.7-2 SEQUENCE OF LINE SIGNALS.— LP-12 PAPER FEED CONTROL

| G-20 | LP-12 | REMARKS |
|---|---|---|
| CALs | GRN | Paper Feed enters CALLED state. |
| QRD | RED | If not ready, QRD returns to the internal program or can repeat the QRD in a loop. |
|  | GRN | If ready, program continues. |
| API | GRN | Prepares to advance paper. |
| 1nn | GRN | Spaces paper $nn_8$ lines. |
| . . . . | . . . . | . . . . |
|  | INT | Interrupts when task is completed. |
| CALs | GRN | G-20 calls Paper Feed. |
| QIN | RED | G-20 identifies source of interrupt. |

The LOAD button and OFF-LINE button are first depressed placing control in channel 1 which causes the mechanism to halt in the configuration shown. The form is then loaded to the load line. Depression of the TOP OF FORM button now causes paper movement to begin and initiates a time delay at the expiration of which control of paper feed is placed in channel 2. Channel 2 contains a hole corresponding to each top of form point. The purpose of the delay is to inhibit paper halt at any top of form points encountered in the approximately 6 inches between the load line and the print position. This could occur whenever the form length is less than 6 inches. Once the above loading procedure has been carried out, the form will always be halted at the next hole in channel 2 any time the command TFF or TFI is given.

When the LP-12 runs out of paper the fact is detected automatically and an interlock opened at the top of the next form.

## FIGURE 4.7-4 TOP OF FORM CONTROL



# CHAPTER 5

# DATA COMMUNICATOR, DC–11

The function of the Data Communicator, Figure 5.0–1, is to provide scatter-read, gather-write input/output capability independent of the Central Processor. The DC–11 communicates with external memory only [see Section 2.5], while the Central Processor communicates with both internal and external memory.

Since each external memory cabinet has its own core drivers and address and data buffers, accesses can take place in two or more cabinets simultaneously. Accesses to the same cabinet are handled on a priority basis. Processing of the four or five characters comprising one word occupies the memory drivers for one access time or 6 microseconds. The DC–11 therefore can handle several input/output operations in the same memory cabinet simultaneously. For example, the MT–10 processes information at the rate of 8.3 microseconds per character, whereas the DC–11 can handle the input or output of an entire word in 6 microseconds. Up to four such tape operations can proceed simultaneously in one cabinet.

In addition to input/output, the DC–11 can also provide intercommunication between components of a large computing complex. This may be by a DC–11/G–20 connection as shown in Figure 5.0–2 (a) or by a DC–11/DC–11 connection as shown in Figure 5.0–2 (b).

To effect certain manufacturing economies achieved by the sharing of hardware, DC–11's are furnished only in modules of two units. Such a module provides input/output over two communication lines.

A module of two DC–11's is housed in a cabinet 64 inches high, 34 inches wide, and 29 inches deep. This module weighs 1200 pounds and draws 1.3 KVA of 115-volt, 60-cycle, single-phase power.

The DC–11 module possesses a power switch with positions ON/SLAVE/OFF. In the slave position the power is controlled by system power control. ON and OFF are unconditional. An indicator shows when the power is on. An ON-LINE switch and indicator for each DC–11 places the unit ON-LINE or OUT of Service. The indicator lights show the on-line condition. A memory parity error will halt the operation and light a PARITY indicator on the console and a PARITY indicator on the affected DC–11. Neon test lamps indicate the contents of the principal registers for use in servicing or program debugging.

## SECTION 5.1 – CONTROL AND COMMAND STRUCTURE

The DC–11 has a group of *slave* states in which it operates under control of line commands received from the line. As a slave the DC–11 can receive and pack, or unpack and transmit 6-bit or 8-bit characters.

Scatter-read and gather-write operations of the DC–11 are executed according to a three-level command structure as shown in Figure 5.1–1. A Starting Command loaded into a Control Register directs the DC–11 to execute the Segment Commands contained in a segment command list. Each segment command specifies the location and length of a segment and states whether it consists of data characters or line command characters and what is to be done with them. The characters of each segment are processed and control is then returned to the next segment command. At the completion of the last character of the last segment [or at any other termination], a termination record is loaded into the control register. In the event of an interrupt or a faulty starting command, certain information is stored in an interrupt register. The cycle can be started again by the storage of a new start instruction in the control register.

FIGURE 5.0-1 DATA COMMUNICATOR, DC-11

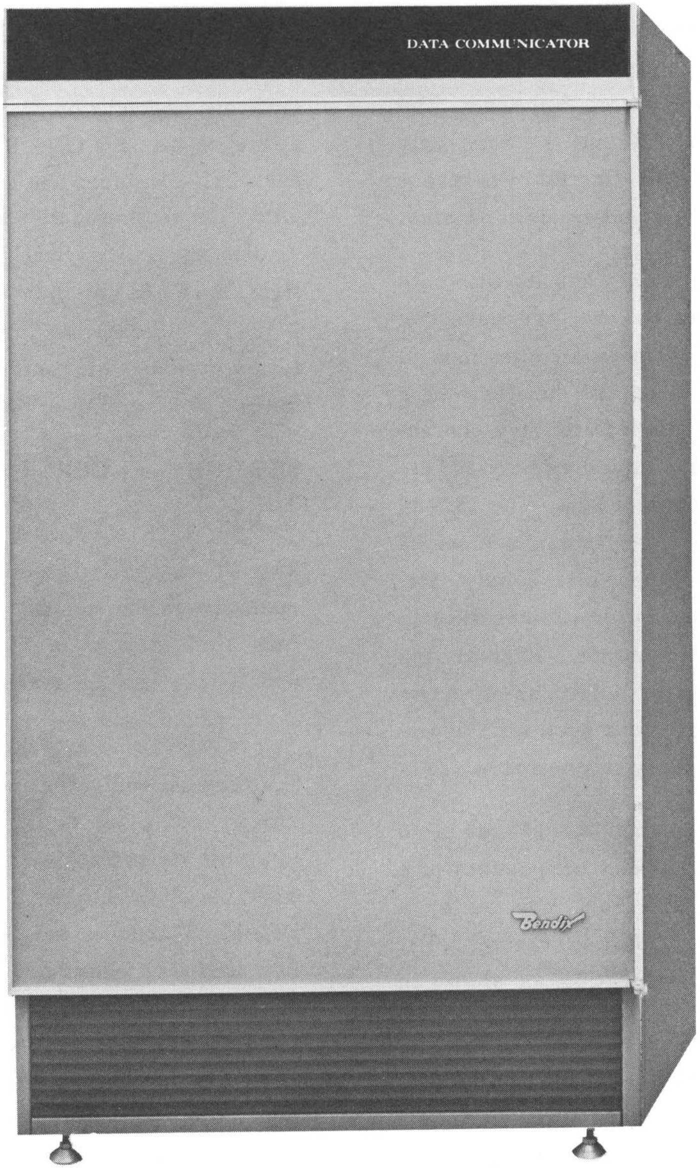
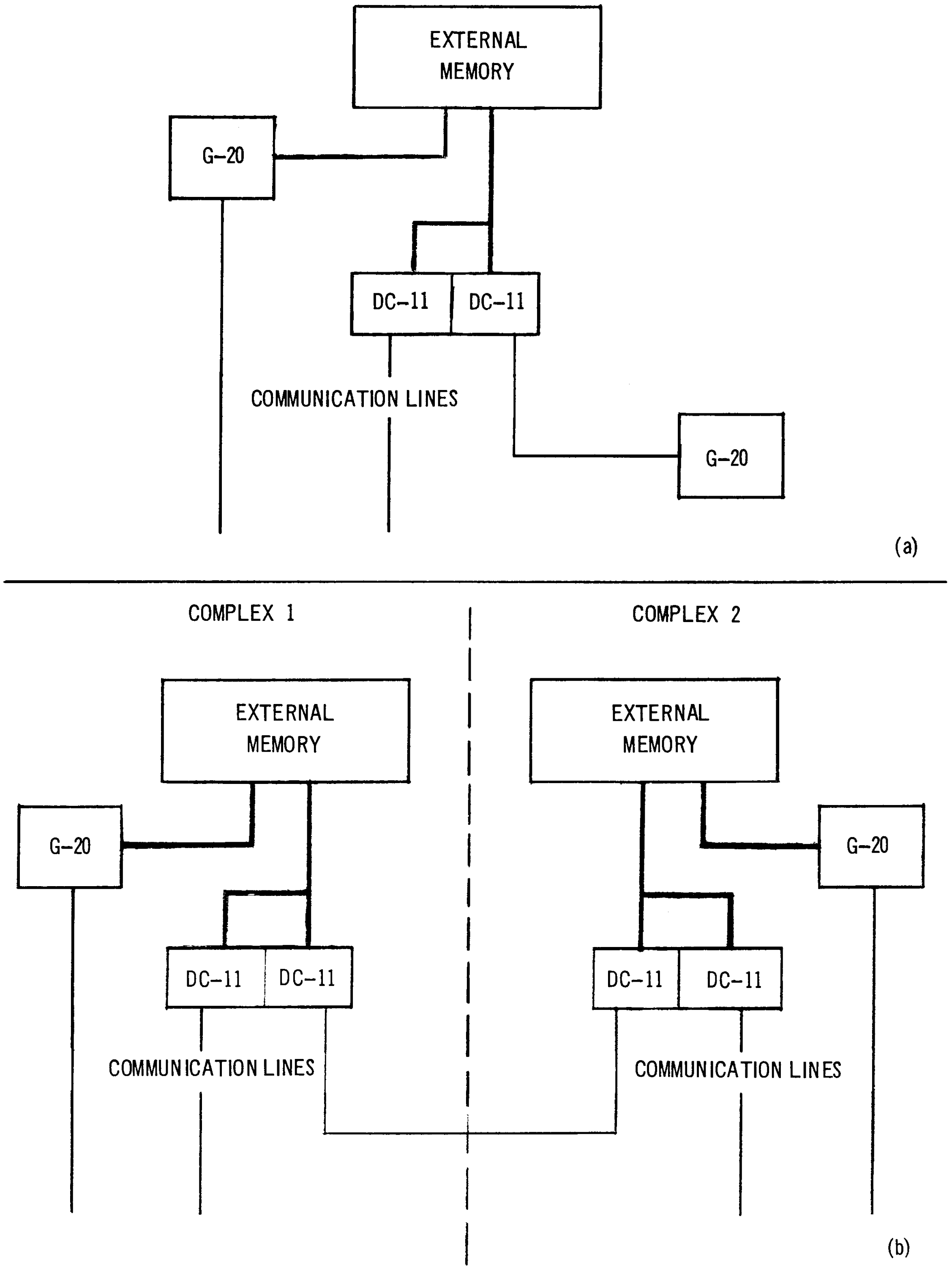


FIGURE 5.0–2 COMPUTING SYSTEM INTERCONNECTIONS USING A DC–11 (a) DC–11/G–20, (b) DC–11/DC–11



## FIGURE 5.1-1 DC-11 CONTROL STRUCTURE (SCHEMATIC)

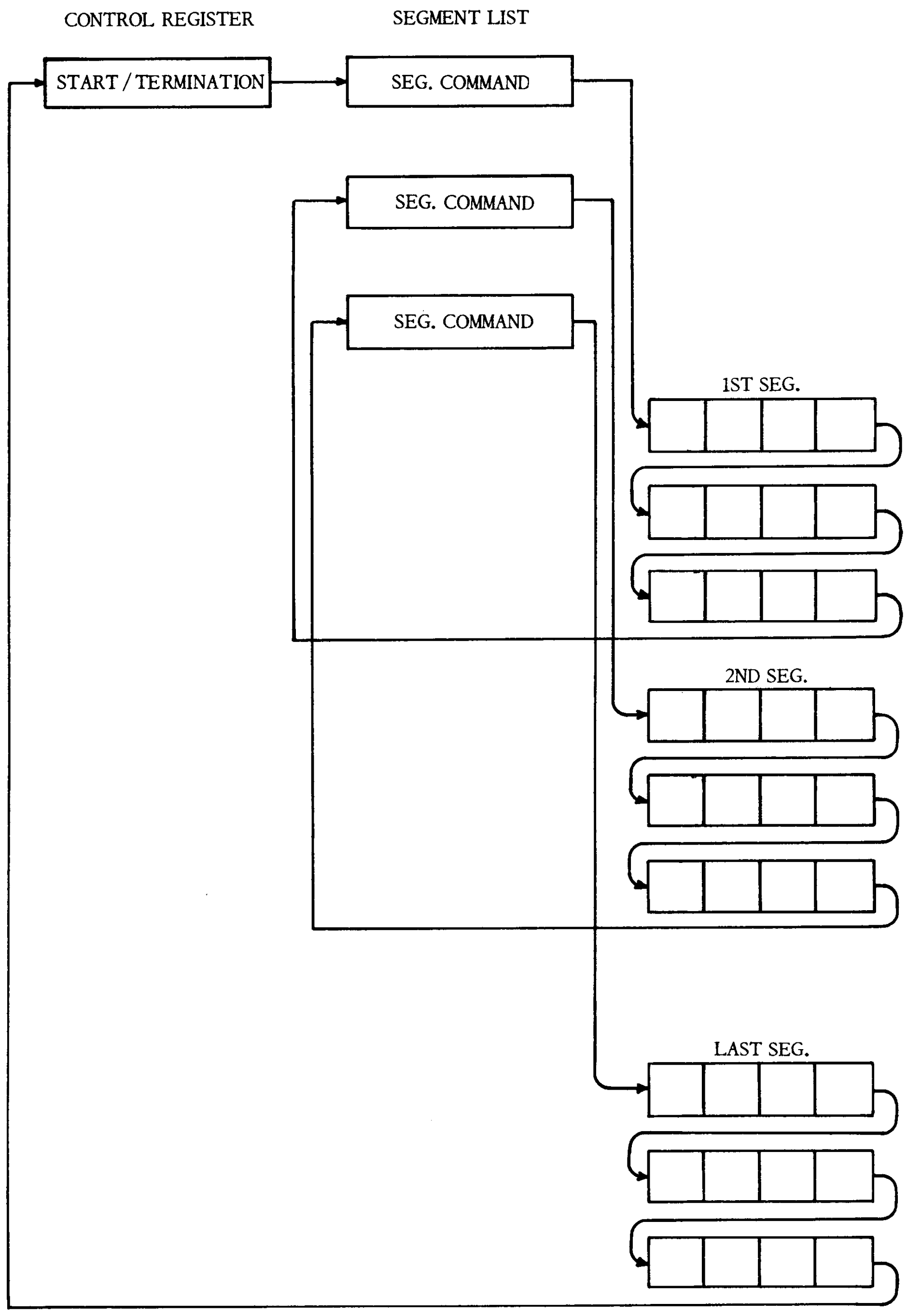
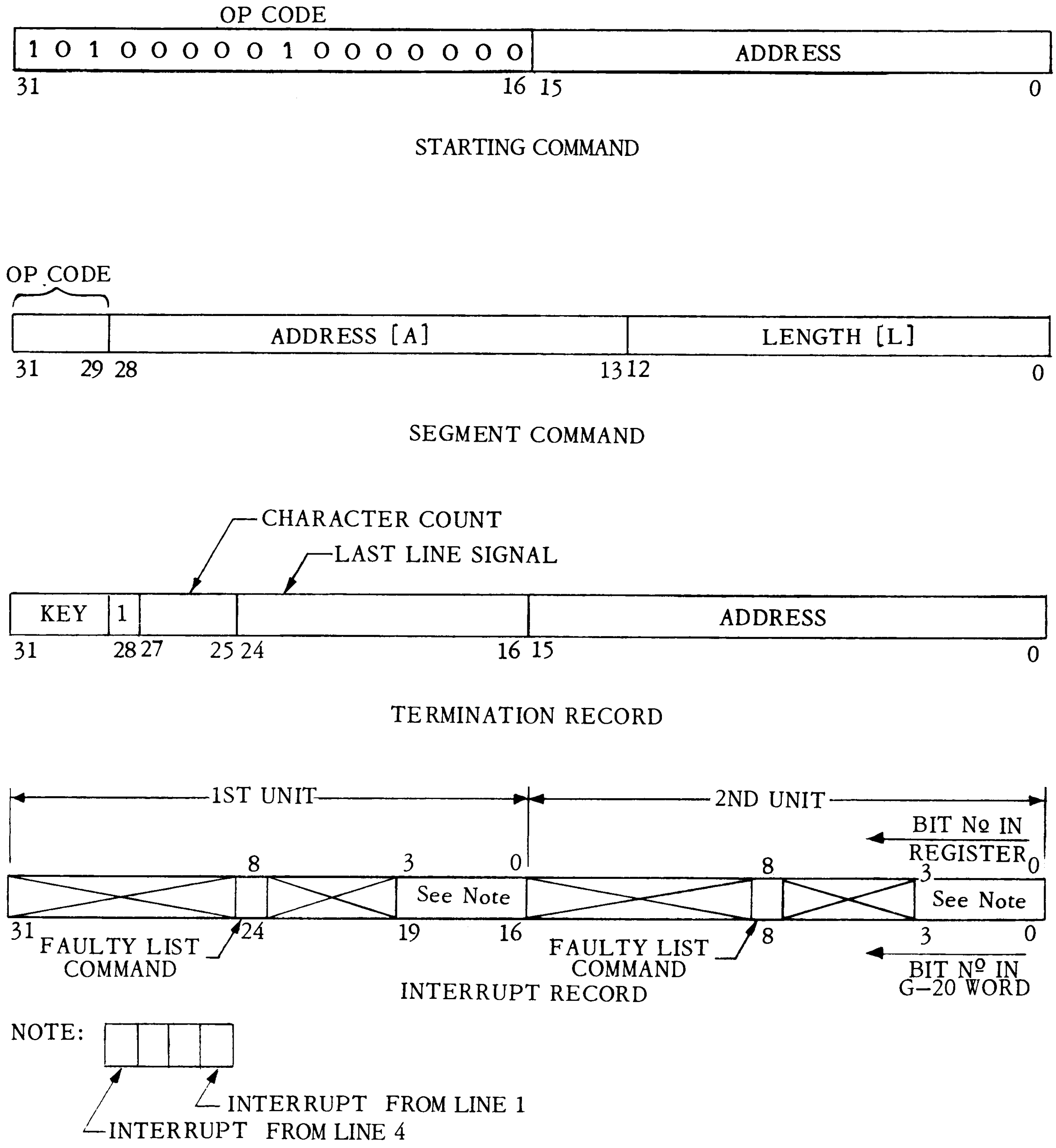


FIGURE 5.1–2 DC-11 WORD FORMATS



STARTING COMMAND

SEGMENT COMMAND

TERMINATION RECORD

INTERRUPT RECORD

## FIGURE 5.1–3 CORE ASSIGNMENT FOR DC–11 REGISTERS

| CORE LOCATION | REGISTER ASSIGNMENT | | NAME |
|---|---|---|---|
| n | CR1 | | Control Register 1 |
| n – 1 | CR2 | | Control Register 2 |
| n – 2 | IR1 | IR2 | Interrupt Registers 1, 2 |
| n – 3 | CR3 | | Control Register 3 |
| n – 4 | CR4 | | Control Register 4 |
| n – 5 | IR3 | IR4 | Interrupt Registers 3, 4 |
| etc. | etc. | | |

n is the highest numbered location in core.

The formats of the starting command, termination record, segment command, and interrupt record are shown in Figure 5.1–2. The start command and termination record occupy the control register, one of the highest numbered locations in core memory. A segment command can occupy any location in the external Core Memory. An interrupt register is actually one half of one of the upper locations in Core Memory. One such core location furnishes interrupt registers for both DC–11's of a module. The assignment of core locations for control registers and interrupt registers is shown in Figure 5.1–3.

It will be noted that the first 16 bits of a starting command must contain the unique starting code 1 010 000 010 000 000. If a data word were accidentally stored in CR due to a programming error, the error would be detected unless the first 16 bits happened to be those shown above.

The segment command opcodes have the meanings shown in Table 5.1–1. Note that the first bit signifies end of list, the second bit signifies that the segment is commands, and the third bit shows the query mode for command segments.

The meaning of the termination record stored in the control register at the termination of processing of a list will be explained in Section 5.2.

## TABLE 5.1–1 DC–11 SEGMENT COMMAND OPCODES

| OCTAL OPCODE | MEANING |
|---|---|
| 0 | Process as data. Next segment is data. |
| 1 | Process as data. Next segment is commands. |
| 2 | Process as commands. Query repeat mode. |
| 3 | Process as commands. Query halt mode. |
| 4 | Process as data. End of list. |
| 5 | Not used. |
| 6 | Process as commands, query repeat mode. End of list. |
| 7 | Process as commands, query halt mode. End of list. |

## SECTION 5.2 – ORGANIZATION OF THE DC–11

As noted in the block diagram, Figure 5.2–1, the DC–11 communicates with Core Memory through a one word buffer, WB, and a Memory Buffer, MB. The Character Distributor, CD, unpacks or packs characters going to or from the Line Register, LR, which couples to the communication line. Character count in a word is kept by counter CC, the most significant character being character 0. Command characters originating from Core Memory or from the line are decoded from the Line Register and used to drive the addressing circuitry at the top of the figure. The starting address stored in register CR1 is read into the Segment Counter, SC, through Memory Buffer MB. This address is then used to access the first segment address and length, which information goes into AT, the Address Tally, and AC, the Address Counter. As the program proceeds (AT) is decremented and (AC) is incremented. Meanwhile the Segment Counter is incremented in preparation for the following segment. The register IR1 stores interrupt requests the unit has received from its own communication line. Interrupts which the DC–11 sends to its own associated Central Processor are stored in the Central Processor's register IR.

Using the convention that parentheses ( ) mean "contents of", and right arrow → means replaces, we may indicate major types of information flow as shown in Examples 5.2–1 and 5.2–2. It is significant to note that the total buffering capacity of the DC–11 is two whole words, (WB) and (CD), plus one character, (LR). During writing at maximum rate all of this buffering capacity may be in use because Core Memory will be waiting for REQ's from the line. During reading, on the other hand, there will usually be at most one whole word plus one character in the DC–11 because a word will be stored as soon as it is completed in CD.

A diagrammatic representation of the manner in which core assignments are related to DC–11 operation is shown in Figure 5.2–2. This hypothetical layout shows the assignments for a segment list and the three segments to be handled in the first DC–11.

## SECTION 5.3 DC–11 STATES

The DC–11 can exist in the following states:

| | |
|---|---|
| OUT of Service | MESSAGE |
| STANDBY | CONTROL TRANSMIT |
| CALLED | CONTROL INTERNAL |
| INSTRUCTED | CONTROL MESSAGE |

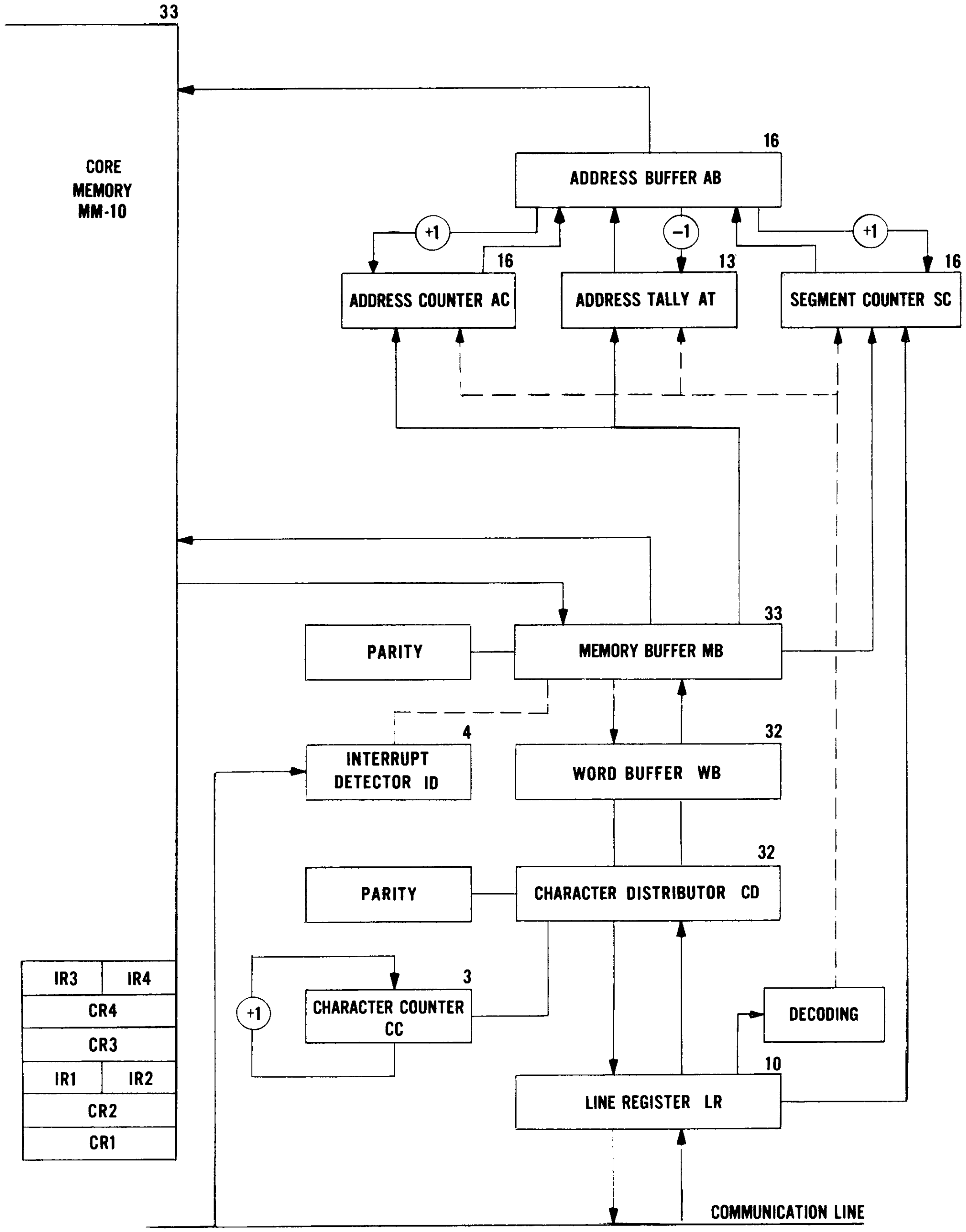
The OUT of Service state and the slave states STANDBY, CALLED, INSTRUCTED, and MESSAGE conform to the general rules applicable to G–20 peripheral equipment and are described in Chapter 3. The control states are described below. These are similar to the control states for the CB–11 with very minor differences. The relationship of these states is shown in Figure 5.3–1.

**5.3–1 Control Transmit** In this state the DC–11 accesses words from Core Memory, unpacks them into 8-bit characters and sends them one at a time on the line, flagged as commands. It awaits a GRN reply each time before sending the next character. It examines these characters as it transmits them, and if it detects its own call, will enter the CONTROL INTERNAL state. If it detects an SDT, it will enter the CONTROL MESSAGE state. CONTROL TRANSMIT can be considered the lowest or "base" state for control type activities.

**5.3–2 Control Internal** In the CONTROL INTERNAL state the DC–11 is accessing and unpacking words from Core Memory and transmitting them as command characters [LSC and DTZ are exceptions]. The DC–11 answers each command with a line response and executes it, just as if the command had originated from outside the DC–11. An SDT causes the DC–11 to enter the CONTROL MESSAGE state.

**5.3–3 Control Message** A DC–11 in this state has established a block data transfer. The DC–11 may be merely the control device or may also participate in the transfer acting as a transmitter or receiver. At the end of the block the DC–11 returns to the CONTROL TRANSMIT state, unless it is also at the end of the execution of the last item in a list. In this case the DC–11 goes to STANDBY.

## FIGURE 5.2-1 SIMPLIFIED BLOCK DIAGRAM, DC-11 DATA COMMUNICATOR



NOTE: ———— DATA PATH
- - - - CONTROL PATH
SMALL NUMBERS ON REGISTERS INDICATE
THE NUMBER OF BITS CARRIED.

## EXAMPLE 5.2–1 DC–11 INFORMATION FLOW: WRITE FROM CORE

| OPERATION | COMMENT |
|---|---|
| $((AB)) \rightarrow (MB)$ | Word is accessed. See Note below. |
| $(MB) \rightarrow (WB)$ | Word goes to buffer. |
| $(WB) \rightarrow (CD)$ | Word goes to character distributor. |
| ${}_{31}(CD)_{24} \rightarrow (LR)$ | First character to line register. |
| $(CC) + 1 \rightarrow (CC)$ | Character counter incremented. |
| ${}_{23}(CD)_{16} \rightarrow (LR)$, etc. | |

Note: AR, AT, and SC are modified by the operations:

| | |
|---|---|
| $(AC) \rightarrow (AB)$ | $(AB) + 1 \rightarrow (AC)$ |
| $(AT) \rightarrow (AB)$ | $(AB) - 1 \rightarrow (AT)$ |
| $(SC) \rightarrow (AB)$ | $(AB) + 1 \rightarrow (SC)$ |

## EXAMPLE 5.2–2 DC–11 INFORMATION FLOW: READ INTO CORE

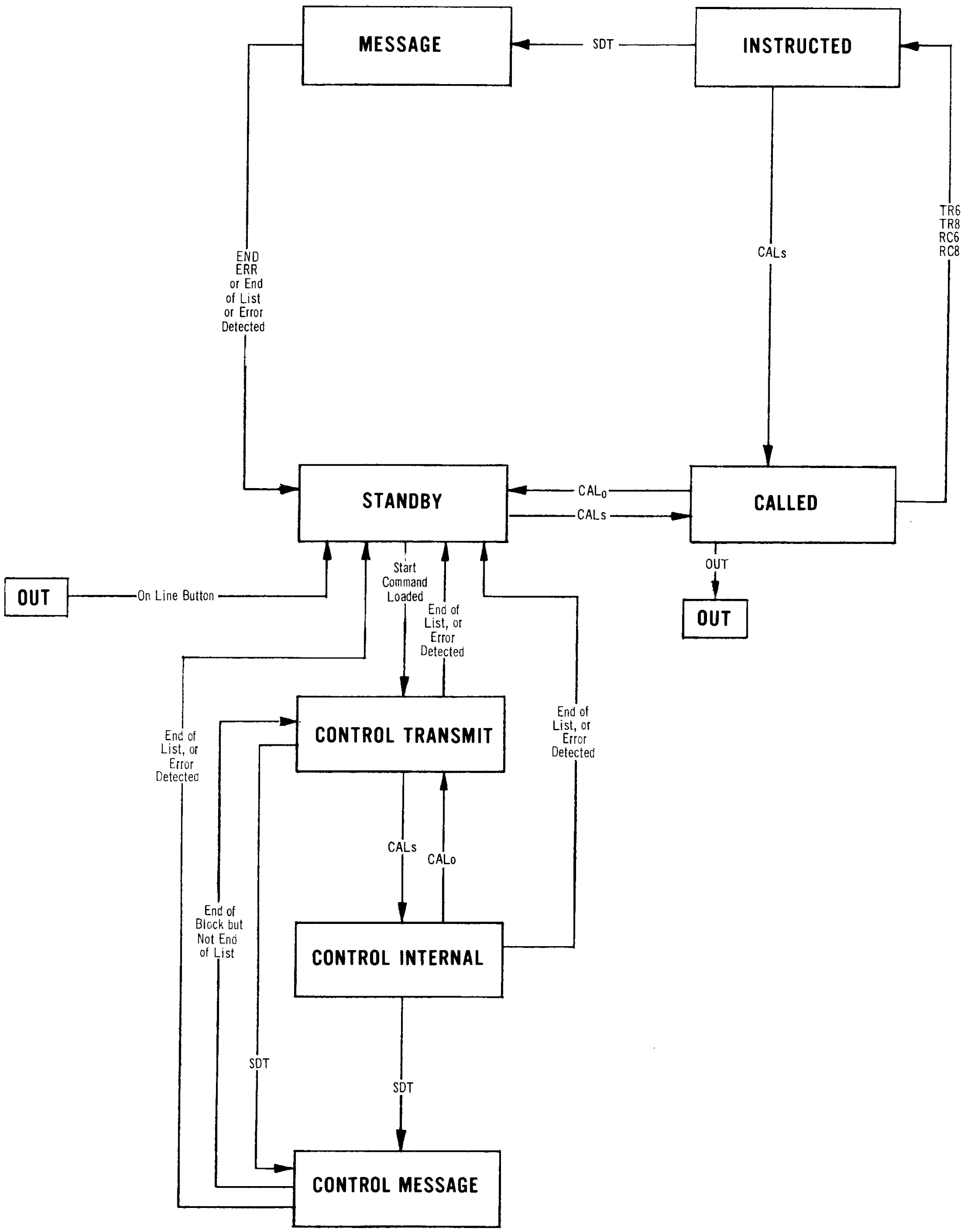
| OPERATION | COMMENT |
|---|---|
| $(LR) \rightarrow {}_{31}(CD)_{24}$ | First character loaded into CD. |
| $(CC) + 1 \rightarrow (CC)$ | Counter incremented. |
| $(LR) \rightarrow {}_{23}(CD)_{16}$, etc. | Next character loaded. |
| $\left.\begin{array}{c}\text{- - - - - - - - - - - - - -}\\ \text{- - - - - - - - - - - - - -}\end{array}\right\}$ | Other characters loaded. |
| ${}_{31}(CD)_{0} \rightarrow (WB)$ | Word transferred to word buffer. |
| $(WB) \rightarrow (MB)$ | Word transferred to memory buffer. |
| $(MB) \rightarrow ((AB))$ | Word stored in Core Memory. |

## FIGURE 5.2-2 CORE MEMORY ASSIGNMENTS FOR DC-11 OPERATIONS [DIAGRAMMATIC]

| LOC. | CONT. | LOC. | CONT. | LOC. | CONT. | LOC. | CONT. |
|---|---|---|---|---|---|---|---|
| 8192 | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| 8254 | — | 8333 | — | | | | |
| | — | | — | | | | |
| | — | | | | | | |
| | — | | | | | | |
| | | | | | | | |
| | | | | 8500 | OP84670007 | | |
| | | | | | OP82540004 | | |
| | | | | | OP83330002 | | |
| | | | | | | | |
| | | 8467 | — | | | | |
| | | | — | | | | |
| | | | — | | | | |
| | | | — | | | | |
| | | | — | | | | |
| | | | — | | | | |
| | | | — | | | | |
| | | | | | | 16378<br>1R3, 1R4 | |
| | | | | | | 16379<br>CR4 | |
| | | | | | | 16380<br>CR3 | |
| | | | | | | 16381<br>IR1, IR2 | |
| | | | | | | 16382<br>CR2 | |
| | | | | | | 16383<br>CR1 | SNL 8500 |

OP and SNL are opcodes

## FIGURE 5.3-1 DC-11 OPERATING STATES



## SECTION 5.4 COMMAND LIST OF THE DC–11

Table 5.4–1 shows the commands or answers applicable to the DC–11.

TABLE 5.4–1 DC–11 COMMAND CODES

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 000 | NOP | No operation. Effectively a space character. Program continues. No reply required. |
| 002 | GRN | Green. The reply of the DC–11 in the slave state to any line signals to which it replies. The signal sent following each character read and sent by the DC–11 in CONTROL INTERNAL.<br>A response the DC–11 can receive from line signals it sends in CONTROL TRANSMIT. DC–11 program continues. |
| 003 | RED | Red. A response the DC–11 can receive from line signals it sends in CONTROL TRANSMIT state. Red response to any query when in query repeat mode causes the query to be repeated. Under any other conditions, DC–11 loads termination record into the Control Register, interrupts, and goes to STANDBY. |
| 004 | END | End. DC–11 in MESSAGE state upon sending or receiving END goes to STANDBY. DC–11 in CONTROL MESSAGE state goes to CONTROL TRANSMIT. DC–11 in CONTROL MESSAGE which has sent SDT goes to CONTROL TRANSMIT upon hearing END. There is never a reply on the line to END. At end of list, loads termination record in Control Register, interrupts, and goes to STANDBY. |
| 005 | ERR | Error. Signifies the end of a block containing an error. DC–11 loads termination record into CR and interrupts Central Processor. Goes to STANDBY. No reply. |
| 010 | SDT | Start Data Transfer. DC–11 in INSTRUCTED state goes to MESSAGE state upon hearing SDT. Replies REQ if instructed to receive. DC–11 in CONTROL INTERNAL or CONTROL TRANSMIT will go to CONTROL MESSAGE. |
| 011 | OUT | Go OUT of Service. DC–11 in CALLED or CONTROL INTERNAL answers GRN and disconnects itself from the line. |
| 014 | RC8 | Receive 8–bit characters. DC–11 in CALLED answers GRN and goes to INSTRUCTED. DC–11 in CONTROL INTERNAL answers GRN. |
| 015 | RC6 | Receive 6–bit characters. See RC8. |
| 016 | TR8 | Transmit 8–bit characters. DC–11 in CALLED or CONTROL INTERNAL answers GRN and prepares to transmit 8–bit characters. Unit in CALLED state goes to INSTRUCTED state. |
| 017 | TR6 | Transmit 6–bit characters. See TR8. |

<table>
<tr><th colspan="3">TABLE 5.4–1 DC–11 COMMAND CODES (Continued)</th></tr>
<tr><th>OCTAL CODE</th><th>ALPHA CODE</th><th>DESCRIPTION</th></tr>
<tr><td>026</td><td>LSC</td><td>Load Segment Counter. DC–11 in CALLED state answers GRN and prepares to load the Segment Counter with the next two [data] characters. DC–11 in the CONTROL INTERNAL state answers GRN, loads next two [data] characters into SC responding GRN to each, and transfers control to the first character of the segment indicated by (SC). LSC and its operand must be in the same segment. Any fraction of a word remaining should be filled with NOP's.</td></tr>
<tr><td>044</td><td>UNP</td><td>Unpack. Prepare to operate in the unpacked mode, transmitting only the right 6 or 8 bits of the word or storing characters at the right end of successive words. DC–11 in CALLED or CONTROL INTERNAL will answer GRN. The command UNP must precede the transmit or receive command.</td></tr>
<tr><td>047</td><td>DTZ</td><td>Decrement and Test for Zero. This command, used for conditional transfer of control, is placed in the fifth word from the end of a command segment. Programming is as shown below.
<table>
<tr><th>WORD</th><th>CONTENTS</th></tr>
<tr><td>n</td><td><table><tr><td></td><td>DTZ</td><td>☒</td><td>☒</td></tr></table>DTZ can be any character of the word. The balance of the word is transmitted but not answered.</td></tr>
<tr><td>n + 1</td><td><table><tr><td>☒</td><td>☒</td><td>☒</td><td>☒</td></tr></table>Entire word transmitted but not answered.</td></tr>
<tr><td>n + 2</td><td><table><tr><td>☒</td><td>☒</td><td>☒</td><td>←TALLY 13 bits→</td></tr></table>First two characters transmitted but not answered. Tally represented by last 13 bits is decremented by 1 and placed in AT (third character truncated as shown).<br>Characters marked with X are ignored by the DC–11.</td></tr>
<tr><td>n + 3</td><td><table><tr><td>LSC</td><td colspan="2">← JUMP ADDRESS →</td><td>NOP</td></tr></table>If tally non-zero, goes to jump address. If tally is zero, goes to n + 4.</td></tr>
<tr><td>n + 4</td><td><table><tr><td></td><td></td><td></td><td></td></tr></table>This must be the last word in the list.</td></tr>
<tr><td>Note:</td><td>The tally is decremented using registers AT and AB. Tally must not exceed 8191.</td></tr>
</table></td></tr>
</table>

TABLE 5.4–1 DC–11 COMMAND CODES (Continued)

| OCTAL CODE | ALPHA CODE | DESCRIPTION |
|---|---|---|
| 200 +<br>uuu | CALs | Call to the unit itself. DC–11 in STANDBY answers GRN and goes to CALLED. DC–11 in CONTROL TRANSMIT answers GRN and goes to CONTROL INTERNAL. |
| 200 +<br>uuu | CALo | Call to some other unit. DC–11 in CALLED state goes to STANDBY. No answer. DC–11 in CONTROL INTERNAL goes to CONTROL TRANSMIT and awaits a reply. |
| 400 +<br>ddd | ddd | A data character. [$0 \leq$ ddd $\leq 377$.] |
| none | INT | An interrupt on the interrupt line. Sent by DC–11 at end of list or at detection of an error or incorrect code. Also sent by DC–11 upon receipt of an interrupt from peripheral equipment such as PC–10 or MT–10. |
| none | REQ | A request on the request line. Used in data transfers. |
| none | SNL | Start New List. The act of the Central Processor in storing start command in CR causes the DC–11 to begin executing the new list. If any operation is in progress, it will be terminated and the termination record stored back in CR. |

## SECTION 5.5 INTERRUPTS AND TERMINATIONS

The DC–11 receives interrupt requests from its communication line which it stores in IR. An interrupt request will be sent to the Central Processor if the DC–11 is in STANDBY. New interrupt bits are united with (IR). The four bits of the IR register for each DC–11 can be assigned arbitrarily at installation time to the units on the communication line of the DC–11.

In addition to the line interrupts there are interrupts generated by the DC–11 if:

(a) A list is completed.
(b) An error or malfunction has occurred.
(c) There was no reply to a command in 1 second.
(d) Red reply for 1 second when in query repeat mode.
(e) Any red reply when in query halt mode.
(f) A start command is stored in CR while a DC–11 operation is in progress.

In any of these cases an interrupt bit goes in position 28 of CR when the termination data are loaded.

Storage of a faulty start command [i. e. having incorrect opcode] in CR will cause an interrupt to be transmitted.

Whenever a list is terminated for any cause, the contents of CC, LR, and AC will be loaded into CR as indicated in Figure 5.1–2. There will also be a bit loaded into position 28. The significance of the 3-bit termination key loaded into bits 31–29 is shown in Table 5.5–1.

At an abnormal termination during writing, it will be necessary to rewrite the block of core. At an abnormal termination of a read-in operation, (AC) always corresponds to the next location to be used after the one being used when termination occurred. Character count (CC) always refers to the word involved in the termination. During transmission or reception of data, (CC) is incremented as soon as the character has been processed. In CONTROL TRANSMIT, (CC) is not incremented until the reply has been

**TABLE 5.5–1 TERMINATION KEY**

| KEY | MEANING |
|---|---|
| 000 | A normal termination. Caused by reading the end of a list or receiving an END command in the MESSAGE state or an OUT command in CALLED or CONTROL INTERNAL. |
| 001 | Line response termination. Occurs in MESSAGE or CONTROL MESSAGE upon receiving a *command* character[with correct parity and other than END]. Occurs in CONTROL TRANSMIT state, query halt mode, upon receiving any non-GRN response [with correct parity]. Occurs in CONTROL TRANSMIT state, query repeat mode upon receiving any [parity correct] response other than RED or GRN, or any response other than GRN after 1 second of RED responses. |
| 010 | Line parity termination. Caused by receiving any parity-incorrect character from the line. |
| 011 | Programming error. Caused when a non-existent address is encountered, when the first segment of a list consists of data, when a segment command calls out as data a segment previously specified as command (or vice versa), or when a word is loaded into the control register of a DC–11 which is on line but not in STANDBY. |
| 100 | No response. Caused by no response for 1 second. |
| 101 | Unassigned. |
| 110 | |
| 111 | |

received. Thus, a no response termination for the two cases shows a different character count. The characters of a word are numbered from left to right starting with 0 as the most significant character for both commands and data and 6-bit and 8-bit modes. In the unpacked mode, (CC) is not meaningful.

## SECTION 5.6 TIMING CONSIDERATIONS

The processing of a list requires one access for each address in the list plus one access for each word in a segment. Thus, each segment of n words requires a total of [n + 1] accesses. This fact may place some limitations on the number of input/output operations which can proceed simultaneously in one cabinet if the segments are very short. For segments of 10 words or more, four MT–10's can operate simultaneously into or out of one memory cabinet. In the worst case of one word per segment, only two MT–10's can operate simultaneously in one memory cabinet.

In the unpacked mode each character requires one access so that a segment of n characters takes [n + 1] accesses.

In the use of an external memory cabinet the DC–11 always has priority over the Central Processor. The Central Processor cannot do block input or output from an external memory cabinet during DC–11 operations in the cabinet.

## SECTION 5.7 DC–11 PROGRAMMING

The DC–11 will usually increase the effectiveness of a G–20 System that is input/output limited. For example, in sorting data on magnetic tapes it is possible to be reading tape blocks in, writing tape blocks out, and comparing keys, simultaneously. This may result in saving time by a factor of 2 or 3 or more. The scatter-read/gather-write capability will also obviate the need for moving data around inside the computer.

**TABLE 5.7–1 FORMAT OF LIST FOR DC–11 SCATTER-READ, GATHER-WRITE**

| OP | A | L |
|---|---|---|
| 2 | C | LC |
| 0 | S1 | L1 |
| 0 | S2 | L2 |
|  | . . . . |  |
|  | . . . . |  |
| 4 | Sn | Ln |

In this table the first word contains the address and length of a segment of commands and the remaining words contain the addresses and lengths of the segments of data to be gathered.

For scatter-read, the command segment may be as follows:

|  |  |  |  |  |
|---|---|---|---|---|
| C | CALs | RC8 | CALt | QRD |
| C +1 | TRA | SDT | 0 | 0 |

The DC–11 has called itself and instructed itself to become a receiver, then called and instructed the tape as a transmitter and finally sent SDT. The word is filled out with space characters [zeros].

For gather-write, the command segment may be as follows:

|  |  |  |  |  |
|---|---|---|---|---|
| C | CALs | TR8 | CALt | QRD |
| C +1 | RCV | SDT | 0 | 0 |

Here the DC–11 has called itself and instructed itself to transmit, then called a tape unit and instructed it to receive, and finally sent an SDT.

**5.7–1 Gather-Write** Table 5.7–1 shows the appearance of the list which must be stored in G–20 Core Memory to accomplish a gather-write from several areas in core onto one tape block.

**5.7–2 Scatter-Read** A scatter-read list is shown in Table 5.7–1.

**5.7–3 Print From Magnetic Tape** A program and task list have been selected to demonstrate the use of the DC–11 command list. It is assumed that some information to be printed has been edited and organized in tape blocks of 240 words, each block containing the information for 8 lines of 120 characters. The task list will appear as in Table 5.7–2. The actual contents of the program used repeatedly by the task list is shown in Table 5.7–3.

## TABLE 5.7-2 DC-11 TASK LIST – MAGNETIC TAPE TO LINE PRINTER

| LOCATION | OP | A | L | COMMENTS |
|---|---|---|---|---|
| LIST | 2 | PROG | 2 | Tape read commands. |
| | 1 | DATA | 360 | $360_8 = 240_{10}$. |
| | 2 | PROG +2 | 2 | Top of form and clear buffer. |
| | 2 | PROG +6 | 2 | Fill buffer. |
| | 1 | DATA | 36 | Data for buffer [line 1]. |
| | 2 | PROG +4 | 4 | Print, advance paper, fill buffer. |
| | 1 | DATA +36 | 36 | Data for buffer [line 2]. |
| | 2 | PROG +4 | 4 | |
| | 1 | DATA +74 | 36 | |
| | 2 | PROG +4 | 4 | |
| | 1 | DATA +132 | 36 | |
| | 2 | PROG +4 | 4 | |
| | 1 | DATA +170 | 36 | |
| | 2 | PROG +4 | 4 | |
| | 1 | DATA +226 | 36 | |
| | 2 | PROG +4 | 4 | |
| | 1 | DATA +264 | 36 | |
| | 2 | PROG +4 | 4 | |
| | 1 | DATA +322 | 36 | Data for buffer [last line]. |
| | 2 | PROG +4 | 2 | Print and upspace. |
| | 6 | PROG +8 | 5 | Test and branch. |

All numbers shown in octal. Locations DATA to DATA +357 contain the information for 8 lines of print, that for the first line being in DATA to DATA +35.

## TABLE 5.7-3 DC-11 PROGRAM – MAGNETIC TAPE TO LINE PRINTER

| LOCATION | 1st CHARACTER | 2nd CHARACTER | 3rd CHARACTER | 4th CHARACTER |
|---|---|---|---|---|
| PROG | CALs | RC8 | CALt | QRD |
| | TRA | SDT | 0 | 0 |
| PROG + 2 | CALf | QRD | TFM | CALp |
| | QRD | CLB | 0 | 0 |
| PROG + 4 | CALp | QRD | PRT | CALf |
| | QRD | 101 | 0 | 0 |
| PROG + 6 | CALs | TR8 | CALp | QRD |
| | RCV | SDT | 0 | 0 |
| PROG + 8 | CALs | DTZ | 0 | 0 |
| | 0 | 0 | 0 | 0 |
| | 0 | 0 | ← NUMBER OF BLOCKS → | |
| | LSC | ← LIST → | | 0 |
| | 0 | 0 | 0 | 0 |

NOTE: CALs = Call to self

CALt = Call to tape

CALf = Call to paper feed

CALp = Call to print control

# CHAPTER 6
# TIMING

Timing of the G–20 System operations can be discussed under the following categories:

a) Central Processor internal operations.
b) Central Processor input/output.
c) Multiplexed Core Memory [DC–11] input/output.
d) Buffered and off-line input/output.

Central Processor timing is mainly determined by the number of memory accesses required.

Central Processor input/output to a terminal device is in general determined only by the terminal device. Communication between the Central Processor and the Control Buffer, CB–11, or with the Print Buffer of the LP–12, is mainly influenced by the speed of the buffer circuitry.

Multiplexed Core Memory input/output timing using the DC–11 is subject to the same considerations noted above for Central Processor input/output.

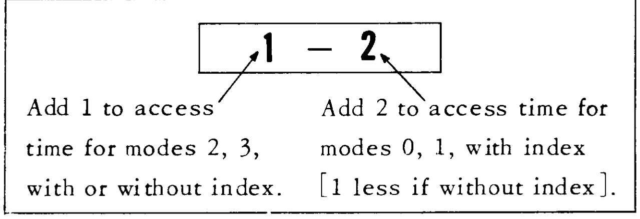
Buffered and off-line input/output using the CB–11 are determined by the terminal device connected to the CB–11.

## SECTION 6.1 – CENTRAL PROCESSOR TIMING

To a first approximation Central Processor timing in the G–20 System depends only on the number of memory accesses required for a particular command. This results from the fact that arithmetic can proceed simultaneously with memory operations. Deviations from this general rule are primarily encountered in multiply and divide operations and in modes 0 and 1 addressing. It is therefore possible to simplify the computation of timing by focussing attention on the clock cycles required over and above access time. In addition, charts are provided showing the number of clock cycles for accessing and executing each general class of commands for each of the four modes with and without index. The time for input/output operations is determined mainly by communication delays at the two ends of the line. A table summarizes the total time per character.

**6.1–1 Simple Commands** Simple commands include all of the opcodes not involving multiply, divide, input/output or block operations. Each simple command possesses a 2–digit timing number as indicated below. The first digit signifies the number of clock cycles which must be added to the basic memory access time to get the total time for accessing and executing commands in address modes 2 and 3.

**FIGURE 6.1–1**
**TIMING NUMBER – SIMPLE COMMANDS**



Add 1 to access time for modes 2, 3, with or without index.

Add 2 to access time for modes 0, 1, with index [1 less if without index].

The second digit of the timing number signifies the number of clock cycles to be added to access time for modes 0 and 1 with index. Without index the time will be 1 clock cycle less. The timing numbers are shown in the alphabetic opcode list, Appendix I.

**6.1–2 Allowance For Exponent, Complement, And Round-off** The timing numbers and timing chart show the basic or minimum times. Exponent adjustment, complementation, and round-off resulting from carry-out can add a few more clock cycles. This can occur at the following times:

1. When (OA) is combined with A or (A) in computing an address.

2. When the above result is combined with (I).
3. At the final access in modes 2 and 3, which require the exponent to be reduced to zero.
4. When the indicated operation is performed.

The following numbers of clock cycles should be added to the computed time:

| | |
|---|---|
| If (OA) $\neq 0$ at start of addressing. . . . | 1 |
| If exponent adjustment is required, n places . . . . . . . . . . . | n + 1 |
| If carry-out and round-off required . . . | 2 |

Allowance for complementation is determined as follows: Whenever an indicated add/subtract operation calls for a net difference of magnitudes, one of the numbers is complemented, the operation performed, and the answer recomplemented, if required. In some cases complementation and recomplementation require an extra clock cycle apiece.

When the numbers have equal exponents at the beginning of the operation, the one in OA or the Acc is complemented at a cost of one clock cycle. In the event that the one complemented was the larger magnitude, there will be a carry adjustment required which will take one clock cycle.

If the exponents were not equal, the exponents are adjusted before the complementation. The number with the larger exponent is first shifted left and if the exponents can thus be equalized, the number which has just undergone the shift is the one which is complemented, and at no cost in time. The complementation takes place during the last transfer required in the shifting. If the number thus shifted was the larger magnitude, the carry will be adjusted at a cost of one clock cycle.

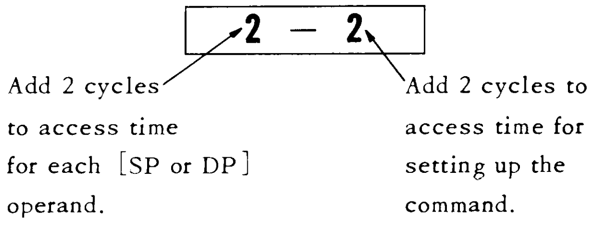
If left shift of the number with the larger exponent does not accomplish exponent equalization, no allowance need be made for complementation, as it will be performed during shift, and carry adjustment will never be required because the number complemented is necessarily the smaller magnitude.

**6.1–3 Allowance For Shift In Store Commands** The operation of shift and round-off, or shift and truncate, prior to storage using opcodes STS, STL, and STI requires an extra clock cycle for each octal shift.

**6.1–4 Repeat Commands** A repeat command possesses a 2–digit timing number indicating how much time must be allowed in addition to accesses.

The first digit indicates the time for processing each operand in addition to access, and the second digit shows the time for setting up the command in addition to accesses required.

**FIGURE 6.1–2**
**TIMING NUMBER – REPEAT COMMANDS**



Exactly the same considerations apply to exponent adjustment, complementation, and round-off as in simple commands. Anticipation of the sizes and magnitudes of the operands in a block command can reduce the execution time by a substantial percentage.

**6.1–5 Multiply** Multiplication time obeys timing chart 0 – 1 in Appendix II with addition of a figure to take care of multiplication proper. Multiplication proper for a random 7–octal multiplicand in the right end of Acc by the most favorable, average, and most unfavorable single precision multipliers is shown in the table below. Also shown are the figures for multiplication of a random 14–octal multiplicand by the most favorable, average, and most unfavorable double precision multipliers. These figures should be added to the accesses required.

**TABLE 6.1–1 MULTIPLICATION TIME**

| | Single Precision | Double Precision |
|---|---|---|
| Minimum | 15 | 8 |
| Average | 30 | 40 |
| Maximum | 38 | 52 |

A more detailed consideration of multiplication time requires some understanding of the algorithm. The multiplicand in Acc is octal-normalized by shifting left 3 bits at a time, each shift taking 1 clock cycle. It is then binary-normalized by taking 1 or 2 single bit shifts, if necessary. These also require 1 clock cycle each. Set-up to begin multiplication takes 3 cycles, except in modes 0 and 1 with an index, in which it takes 4 cycles. Multiplication proper takes from 1/3 of a cycle per bit for 3 like bits in succession, to 1 cycle per bit for alternating 1's and 0's. A statistically average multiplier takes .785 cycles per bit. Termination requires 3 cycles. The answer is octalized by shifting right 1 or 2 bits, if required, to make the first 3 bits correspond to an octal digit. Each shift takes 1 cycle. In the event that the exponent of the answer is less than –63 [exponent underflow] the mantissa will be shifted right 3 bits at a time until the exponent reaches –63 or the mantissa is shifted out of the register. Each shift takes 1 cycle.

Certain special cases of multiplication are of interest. Shifting, which is multiplying by 1, 2, or $4*8^{\pm NN}$ takes an average of 14 clock cycles in addition to timing chart 0 – 1 in Appendix II. Multiplication by unity [for normalizing] takes 6 clock cycles plus 1 clock cycle per octal shift in addition to timing chart 0 – 1. Multiplication by 10 using a mode 0 command takes 18 cycles total, and multiplication in which either operand is zero takes 3 cycles in addition to timing chart 0 – 1.

**6.1–6 Division** Division time obeys timing chart 0–1 in Appendix II with the addition of a figure to take care of division proper. Division time of a 7–octal numerator in the right end of Acc by the most favorable, average, and most unfavorable denominator is shown in the table below. Corresponding figures are shown for 14–octal numerators. These times should be added to timing chart 0–1.

**TABLE 6.1–2 DIVISION TIME**

| | Single Precision | Double Precision |
|---|---|---|
| Minimum | 33 | 19 |
| Average | 69 | 55 |
| Maximum | 105 | 91 |

A more detailed consideration of division time requires some understanding of the algorithm. The numerator and denominator are each octal-normalized and then binary-normalized, each shift of either variety requiring 1 cycle. Set up to begin division takes 3 cycles, except for modes 0 and 1 with an index, in which it takes 4 cycles. Division proper takes from 1/3 of a cycle per bit for the most favorable case to 2 cycles per bit for the most unfavorable case. The statistically average case takes 1.14 cycles per bit. Termination takes 2 cycles. The answer is octalized by shifting right 1 or 2 bits if required to make the first 3 bits correspond to an octal digit. Each shift takes 1 cycle. In the event that the exponent of the answer is less than –63 [exponent underflow] the mantissa will be shifted right 3 bits at a time until the exponent reaches –63 or the mantissa is shifted out of the register. Each shift takes 1 cycle.

Certain special cases of division may be of interest. Division into zero takes 3 cycles in addition to timing chart 0 – 1 in Appendix II. Division of $8^{14} -1$ by unity is the most unfavorable case shown in the "Division Time" table. Division of a number by itself is the most favorable case shown in the table.

**6.1–7 Allowance For External Memory** Preparation of an operand or address requiring use of the external memory requires an extra clock cycle. The only exception is mode 1 with an index, which requires no extra time.

For repeat commands each single or double precision operand in external memory adds one cycle to execution time.

Storage in external memory is 3 clock cycles faster than storage in internal memory, provided the next operation does not involve the same memory cabinet. [In modes 1, 2, 2I, 3, and 3I, the net saving will be reduced to 2 cycles if address preparation required the use of external memory.]

Commands may be in external memory without affecting execution time except for mode 0 without index, which is increased by one clock cycle.

**6.1–8 Central Processor Input/Output Timing** The following table shows the figures for Central Processor time involved in input/output operations.

TABLE 6.1–3 TIMING OF CENTRAL PROCESSOR INPUT/OUTPUT

| | SET-UP | PROCESSING TIME PER CHARACTER [In Microseconds] | | | | | |
|---|---|---|---|---|---|---|---|
| CODE | TIMING | CC–10 | PT–10 | MT–10 | PC–10 | LP–12 | CB–11 |
| TLC | 14 – 15 | 9.0 | 10. | 10. | 3. | 10. | 4. |
| TDC | 13 – 14 | 9.0 | 10. | – | – | – | – |
| BTC6 | 11 – 12 | 9.0 | 13.5 | 13.5 | 6.5 | 13.5 | 7.5 |
| BTC8 | 11 – 12 | 9.0 | 13.5 | 13.5 | 6.5 | 13.5 | 7.5 |
| BTD6 | 11 – 12 | – | 9.1 ms | 8.4 | 17. | 15. | 6. |
| BTD8 | 11 – 12 | – | 9.1 ms | 8.4 | 17. | 15. | 6. |
| BRD6 | 12 – 13 | – | 2./4. ms | 8.4 | 17. | – | 6. |
| BRD8 | 12 – 13 | – | 2./4. ms | 8.4 | 17. | – | 6. |

Notes: The total time for an input or output operation is the sum of the set-up time plus the processing time per character.

Set-up time is computed with the aid of the set-up timing number. The first figure in this number is the amount to be added to total access times for modes 2 and 3. The second figure is the amount to be added to total access time for modes 0 and 1 with an index; without index deduct one microsecond.

This table shows the maximum possible processing times for short communication lines involving negligible propagation delay. For long lines allow 1.75 nanoseconds per foot propagation delay in each direction.

For the CC–10 and the PT–10 the figures for a TDC opcode are the time for transmission of a character to or from a holding register in the accessory. The time for the mechanical operation in the accessory is not included. PT–10 read times show high/low speeds.

For the LP–12 the figures for block data transfer are the time for loading the Print Buffer and do not include printing time.

For all block operations the time includes delays introduced by the terminal device.

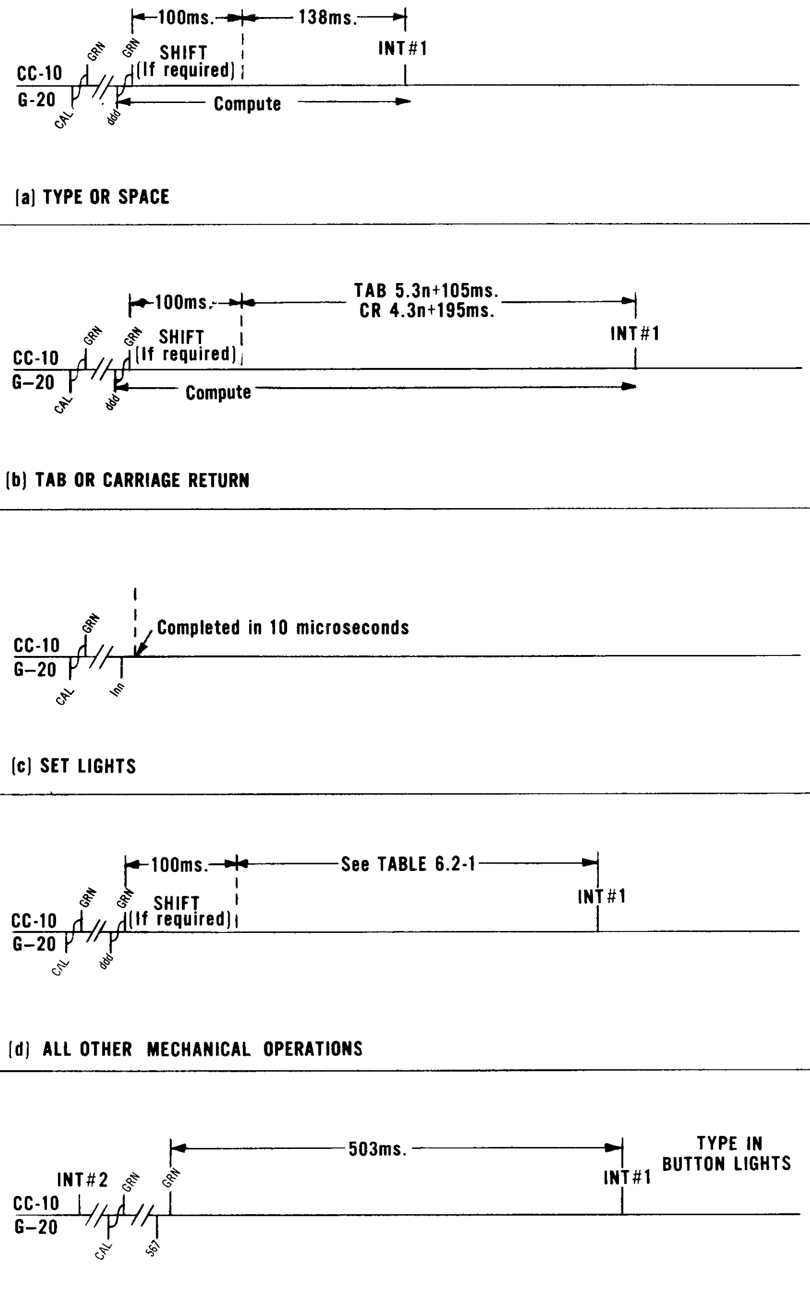
## SECTION 6.2 – CONTROL CONSOLE TIMING

The CC–10 Control Console responds to a line signal in 9.0 microseconds. A mechanical operation takes about 100 milliseconds, and its completion is signalled by an interrupt. Table 6.2–1 indicates the approximate timing for the various mechanical operations. Operation of the shift key when required is handled automatically by the typewriter logic, but operating or releasing the shift key takes 100 milliseconds as shown in the table.

During the typing of a series of characters the shift solenoid will be operated or released only when necessary, not at every character. SPACE is considered as an upper case character, and all other nonprinting characters [such as SET TAB, CARRIAGE RETURN, etc.] are considered as lower case.

The console timing diagrams, Figure 6.2–1, summarize the important features of console timing.

FIGURE 6.2-1 CONTROL CONSOLE TIMING DIAGRAMS (a) TYPE OR SPACE, (b) TAB OR CARRIAGE RETURN, (c) SET LIGHTS, (D) ALL OTHER MECHANICAL OPERATIONS, AND (e) TYPE-IN.



(a) TYPE OR SPACE

(b) TAB OR CARRIAGE RETURN

(c) SET LIGHTS

(d) ALL OTHER MECHANICAL OPERATIONS

(e) TYPE IN

NOTE: Commands used by the Interrupt Service Routine in establishing the source of interrupts are not shown.
A mark connecting two successive signals ⌐ indicates that the second is an immediate automatic response to the first.
The command sequences shown are for illustrative purposes only.

**TABLE 6.2–1 CONSOLE TIMING**

| | | |
|---|---|---|
| Line responses | 9.0 | microseconds |
| Type or space [without shift] | 138 | milliseconds |
| Up shift or down shift | 100 | milliseconds |
| Tab | 5.3 n + 105 | milliseconds |
| Return carriage | 4.3 n + 195 | milliseconds |
| Backspace | 205 | milliseconds |
| Tab set | 93 | milliseconds |
| Tab clear | 79 | milliseconds |
| Unlock keyboard | 503 | milliseconds |
| Lock keyboard | 540 | milliseconds |

Note: Up shift and down shift are taken only when necessary. SPACE is an upper case character, but all other nonprinting characters are lower case. The number n is the number of character positions through which the carriage moves.

## SECTION 6.3 – PAPER TAPE TIMING

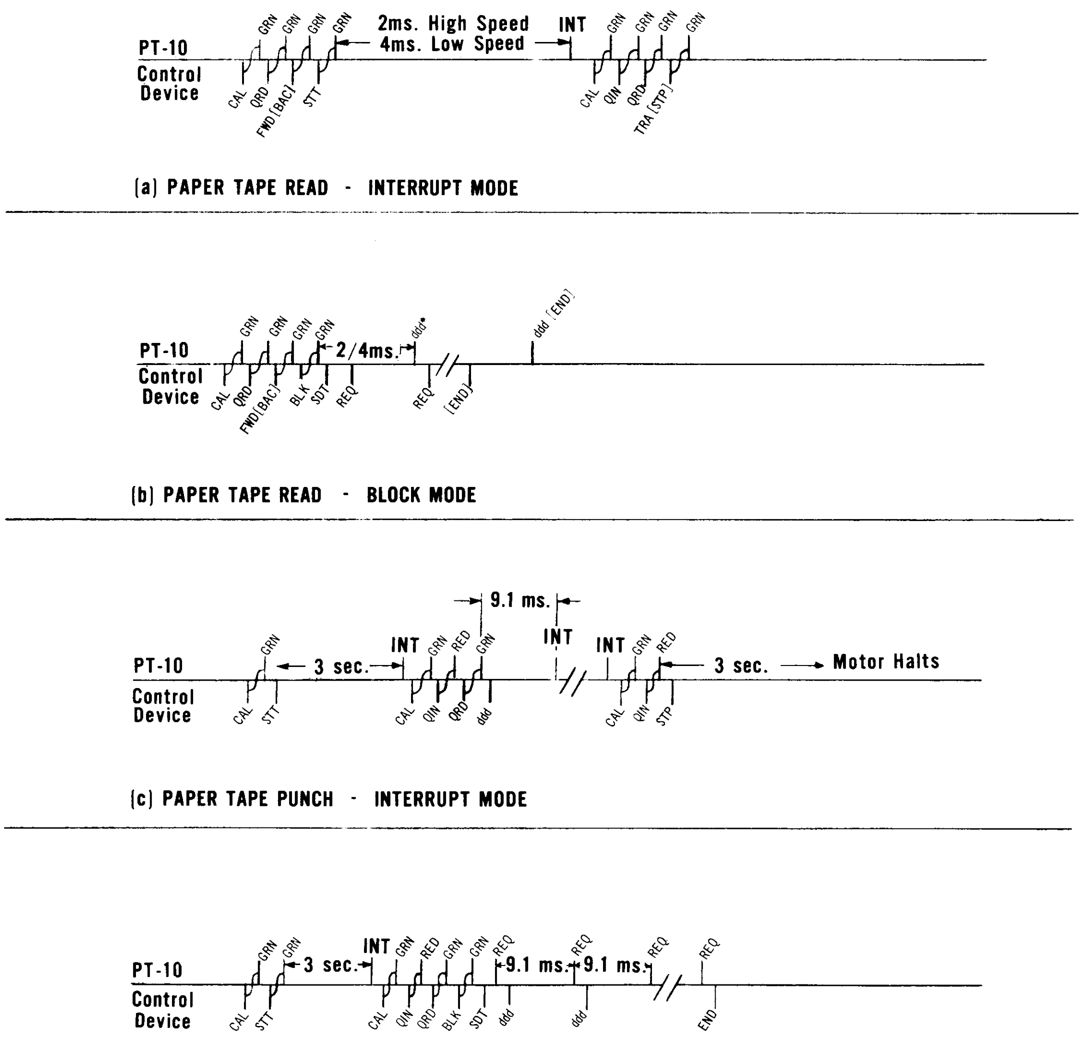
The paper tape reader and punch respond to calls and queries in less than 10 microseconds but require varying amounts of time to execute mechanical operations.

**TABLE 6.3–1 PAPER TAPE TIMING**

| | |
|---|---|
| Read single character and interrupt, high/low speed | 2/4 milliseconds |
| Read in block mode, per character, high/low speed | 2/4 milliseconds |
| Start punch motor and interrupt | 3 seconds |
| Punch single character and interrupt | 9.1 milliseconds |
| Punch in block mode, per character | 9.1 milliseconds |
| Halt punch motor | 3 seconds |

Timing charts for the principal operations are shown in Figure 6.3–1.

FIGURE 6.3-1 PAPER TAPE TIMING DIAGRAMS (a) PAPER TAPE READ-INTERRUPT MODE, (b) PAPER TAPE READ-BLOCK MODE, (c) PAPER TAPE PUNCH-INTERRUPT MODE, (d) PAPER TAPE PUNCH-BLOCK MODE



(a) PAPER TAPE READ - INTERRUPT MODE

(b) PAPER TAPE READ - BLOCK MODE

(c) PAPER TAPE PUNCH - INTERRUPT MODE

(d) PAPER TAPE PUNCH - BLOCK MODE

NOTE: Commands used by the Interrupt Service Routine in establishing the source of interrupts are not shown.
A mark connecting two successive signals ┌┘ indicates that the second is an immediate automatic response to the first. Responses require 6 microseconds.
The command sequences shown are for illustrative purposes only.

## SECTION 6.4 – MT–10 MAGNETIC TAPE TIMING

The MT–10 Magnetic Tape responds to line commands and queries in 10 microseconds. It transmits or receives data characters at the nominal rate of one character every 8.3 microseconds. Other significant times are as shown in Table 6.4–1.

**TABLE 6.4–1 MT–10 TIMING, MILLISECONDS**

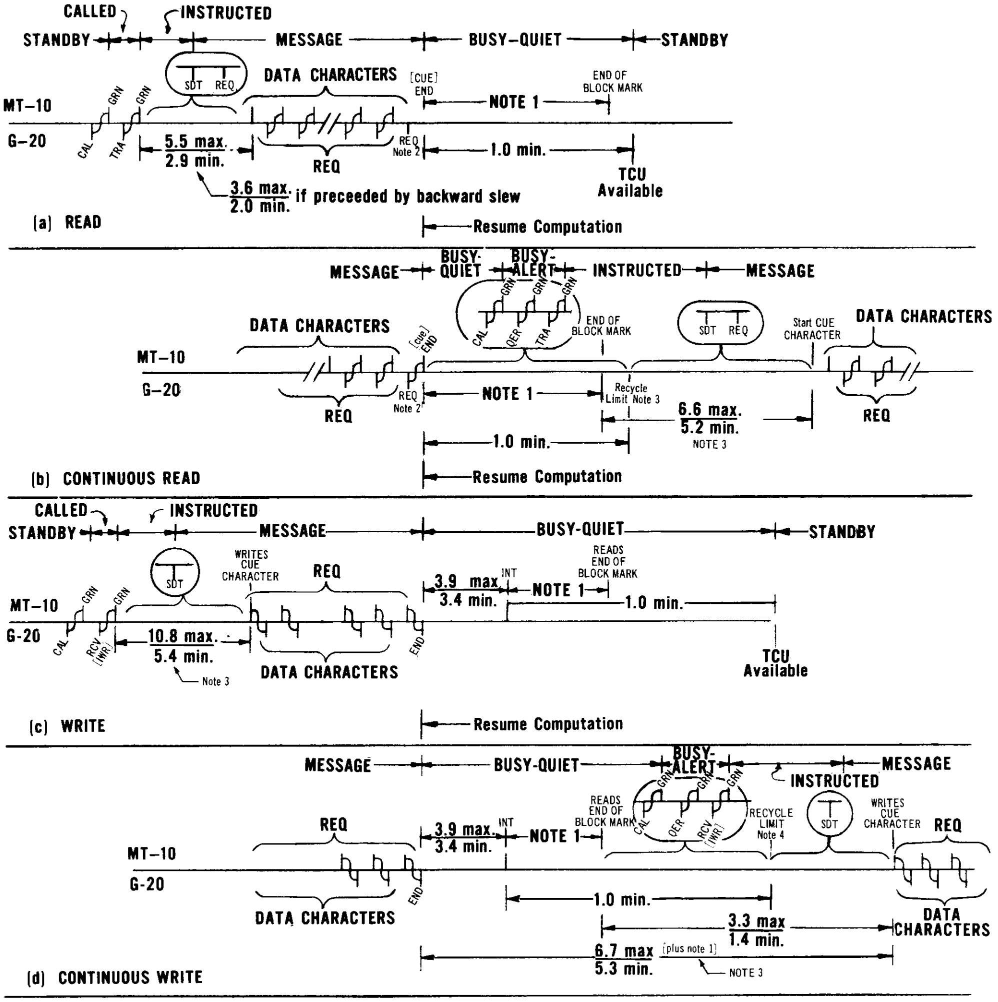
| | | MIN. | NOM. | MAX. |
|---|---|---|---|---|
| Interblock time read or write | | 5.3 | 6.0 | 6.7 |
| Start to read | | 2.9 | 4.1 | 5.5 |
| Stop reading | | 1.0 | 1.2 | 1.3 |
| Start to write | | 5.4 | 8.0 | 10.8 |
| Stop writing | | 4.4 | 4.7 | 4.9 |
| Halt slewing at 110 inches/second | | | | 8.0 |
| Halt slewing at 220 inches/second | | | | 1.5 sec. |
| Switch lines | | 5.0 | 5.6 | 6.2 |
| Slew | 0–511 blocks | 12 + 9L milliseconds | | |
| | 512–4095 blocks | 12 + 4.5 (L + K) milliseconds | | |

Note: L = length to be slewed, inches

K = length for 511 blocks, inches

Timing charts for the principal operations appear in Figure 6.4–1.

## FIGURE 6.4–1 MAGNETIC TAPE TIMING DIAGRAMS
## (a) READ, (b) CONTINUOUS READ, (c) WRITE, AND (d) CONTINUOUS WRITE



(a) READ

(b) CONTINUOUS READ

(c) WRITE

(d) CONTINUOUS WRITE

NOTE 1. THIS INTERVAL DEPENDS ON THE DIFFERENCE BETWEEN THE INITIALLY WRITTEN BLOCK LENGTH AND THE LENGTH OF DATA NOW OCCUPYING THE BLOCK. IT IS NEGLIGIBLE FOR IWR.

2. RECEIVER MAY TERMINATE BY SENDING END INSTEAD OF REQ.

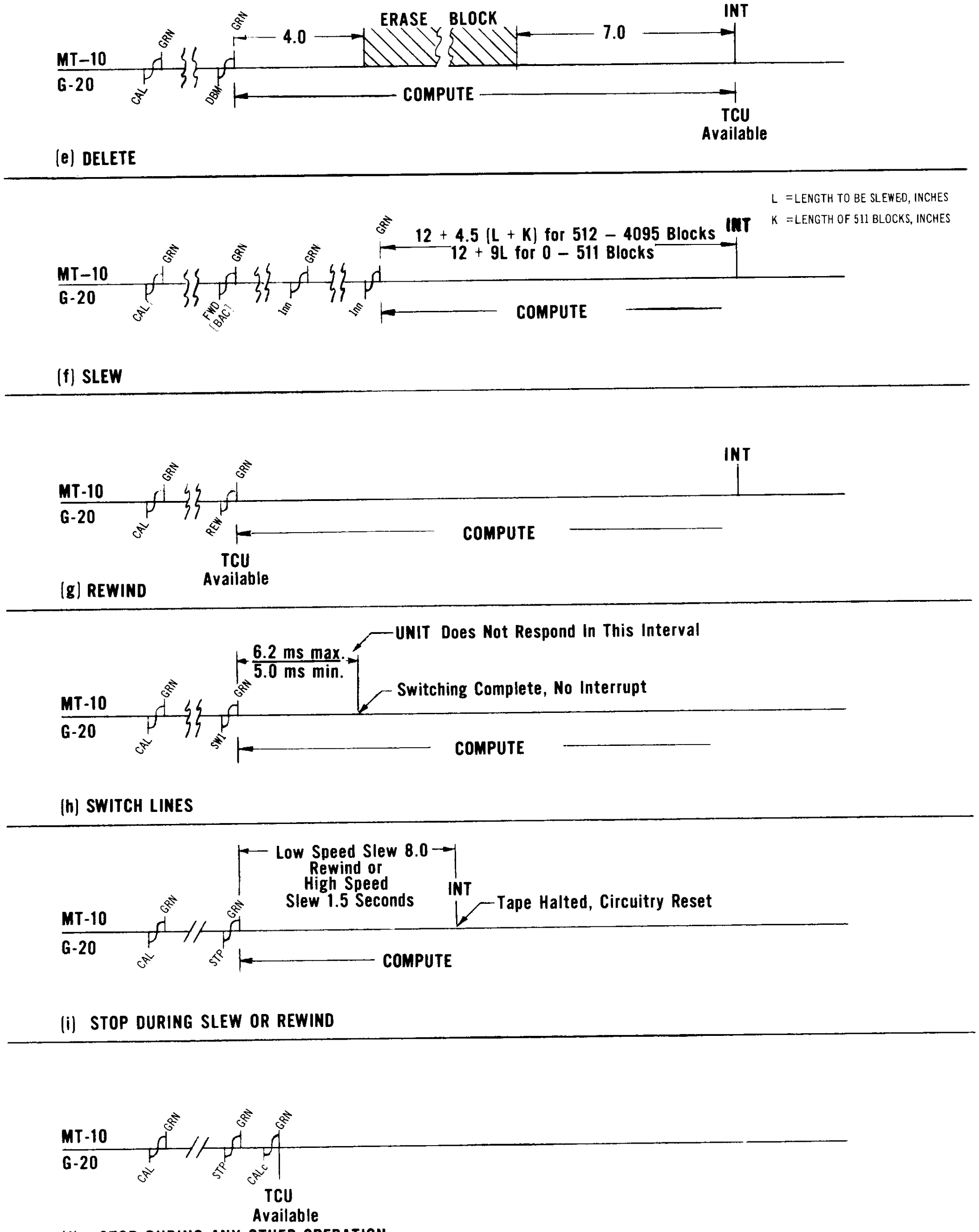
3. CONTROL DEVICE MAY EXECUTE A PROGRAM IN THIS INTERVAL DURING THE TIME NOT OCCUPIED BY LINE COMMANDS FOR STARTING THE NEXT BLOCK DATA TRANSMISSION.

4. AT THE RECYCLE LIMIT POINT THE TRANSPORT INITIATES THE HALT SEQUENCES UNLESS NEW CALL AND READ OR WRITE COMMAND HAVE BEEN RECEIVED.

5. A MARK CONNECTING TWO SIGNALS INDICATES THAT THE SECOND IS AN IMMEDIATE AUTOMATIC RESPONSE TO THE FIRST.

ALL TIMES ARE SHOWN IN MILLISECONDS UNLESS STATED OTHERWISE.

FIGURE 6.4–1 MAGNETIC TAPE TIMING DIAGRAMS [CONTINUED] (e) DELETE, (f) SLEW, (g) REWIND (h) SWITCH LINES (i) STOP DURING SLEW OR REWIND AND (j) STOP DURING ANY OTHER OPERATION



(e) DELETE

(f) SLEW

(g) REWIND

(h) SWITCH LINES

(i) STOP DURING SLEW OR REWIND

(j) STOP DURING ANY OTHER OPERATION

## SECTION 6.5 - CARD-PRINTER COUPLER TIMING

The Card-Printer Coupler, PC–10, is ordinarily operated in the interrupt mode, which means that the Central Processor [or other controlling device] is called into action only when required. Thus, timing of a card or printer operation has two aspects: first, the total time, which is usually determined by the input/output device; and second, the amount of Central Processor time required.

Timing of communication between the PC–10 and the possible controlling devices is shown in Table 6.5–1.

Charts for timing of the principal PC–10 operations are shown in Figures 6.5–1 and 6.5–2.

The time intervals on these charts applying to card readers, punches, and printers are shown in Tables 6.5–2 and 6.5–3. Figure 6.5–3 is a key to the symbols used in printer and card handler timing.

**TABLE 6.5–1 PC–10 COMMUNICATION TIMING, MICROSECONDS PER CHARACTER**

| | |
|---|---|
| Reply to single line command | 3. |
| Block commands from G–20 or DC–11 | 6.5 |
| Block data to G–20 or DC–11 | 17. |
| Block data from G–20 or DC–11 | 17. |

Note: The time for block transmission includes delays introduced by controlling devices.

Timing for card operations is ordinarily a function only of the speed of the card machine to be used. On the other hand, printing is influenced by several factors:

Print roll speed
Alphabet size used
Number of spaces between lines
Number of columns to be printed

With the LP–10 or LP–11 a prepared print image must be delivered to the printer beginning at the first character position and continuing to the last character row to be used [e.g., row 18 if only the digits 0–9 are to be used. See Table 4.5–1.] When the last row of the image has been delivered to the printer, upspacing of the paper and the construction of the next print image is begun. Upspacing and image construction proceed simultaneously. Printing can again begin at the next index time following the completion of the slower of these tasks. Therefore, to determine the print interval for one line:

1. Determine the upspace time and image construction time.
2. Add the larger of the above figures to the printing time [row time multiplied by number of rows in the image].
3. Take the integral number of print roll revolutions next larger than the above sum.

**TABLE 6.5–2 PRINTER TIMING LP–10, LP–11, MILLISECONDS**

| | PRINT ROLL CYCLE | ROW INTERVAL | UPSPACE TIME |
|---|---|---|---|
| LP–10, 900 RPM | 67 | 1.01 | 14 + 8n |
| LP–11, 1,000 RPM | 60 | .91 | 9 + 7n |
| LP–11, 1,500 RPM | 40 | .61 | 9 + 7n |

Image construction time in the G–20 = .100D + .215W
where n = number of lines to be upspaced
D = depth of print image
W = width of print image

Upspace time is shown in Table 6.5–2. The image construction time using an optimum G–20 program is

$$.100\ D + .215\ W \text{ milliseconds}$$

where D is effective alphabet size or depth of the image, and W is the effective number of columns or width of the image. The effective alphabet extends from character position 0 to the last character to be used. The effective width begins at column one and extends to the right-most column used, including intervening blank columns, if any. The printing time is calculated from the row time shown in Table 6.5–2.

It should be noted that in some circumstances a special print roll layout may substantially increase the effective speed for LP–10, LP–11 operations.

**EXAMPLE 6.5–1 CALCULATE THE MAXIMUM PRINTING RATE FOR DOUBLE SPACE PRINTING 72 COLUMNS WIDE ON LP–10 WITH FULL 63–CHARACTER ALPHABET.**

| | |
|---|---|
| Upspace time | = 30 milliseconds |
| Image construction | = .100D + .215W |
| | =(.1) (63) + (.215) (72) |
| | = 21.8 milliseconds |
| Printing time | = (1.01) (63) |
| | = 63.6 milliseconds |
| Total time | = 63.6 + 30 = 93.6 milliseconds |

Printing will require two revolutions per line or 134 milliseconds.

**EXAMPLE 6.5–2 CALCULATE THE PRINTING SPEED FOR SINGLE SPACE PRINTING 60 COLUMNS WIDE, NUMERIC ONLY, 27–CHARACTER ALPHABET ON LP–10.**

| | |
|---|---|
| Upspace time | = 22 milliseconds |
| Image construction | = .100D + .215W |
| | = (.1) (27) + (.215) (60) |
| | = 15.6 milliseconds |
| Printing time | = (1.01) (27) |
| | = 27.3 milliseconds |
| Total time | = 27.3 + 22 = 49.3 milliseconds |

Printing will require one revolution per line or 67 milliseconds.

## TABLE 6.5–3 CARD MACHINE TIMING, MILLISECONDS

| | C | E | D | R | S | T |
|---|---|---|---|---|---|---|
| | CARD | EMITTER | | ROW | START | CFD |
| | INTERVAL | TIME | DELAY | INTERVAL | TIME [Min./Max.] | LIMIT (b) |
| READ | | | | | | |
| 088* | 92 | 1.4 | .30 | 4.62 | 300/400 | 15 |
| 085 | 250 | 4.2 | 1.0 | 12.5 | 400/650 | 1.0 |
| 528 | 300 | 8.0 | 1.0 | 21. | 300/600 | 27. |
| 407 NR | .... | .... | .... | .... | .... | .... |
| 514 NR | .... | .... | .... | .... | .... | .... |
| 523 | 600 | 10. | 1.0 | 43. | 400/1000 | 47 |
| PUNCH | | | | | | |
| 544* | 240 | 3.0 (a) | .30 | 15. | 300/550 | 20 |
| 514 | 600 | 3.0 (a) | 1.0 | 43. | 350/950 | 45 |
| 528 | 600 | 3.0 (a) | 1.0 | 43. | 350/950 | 59 |
| 523 | 600 | 3.0 (a) | 1.0 | 43. | 350/950 | 47 |
| TABULATE | | | | | | |
| 407 NR | .... | .... | .... | .... | .... | .... |

* Preferred for use with Bendix PC–10.

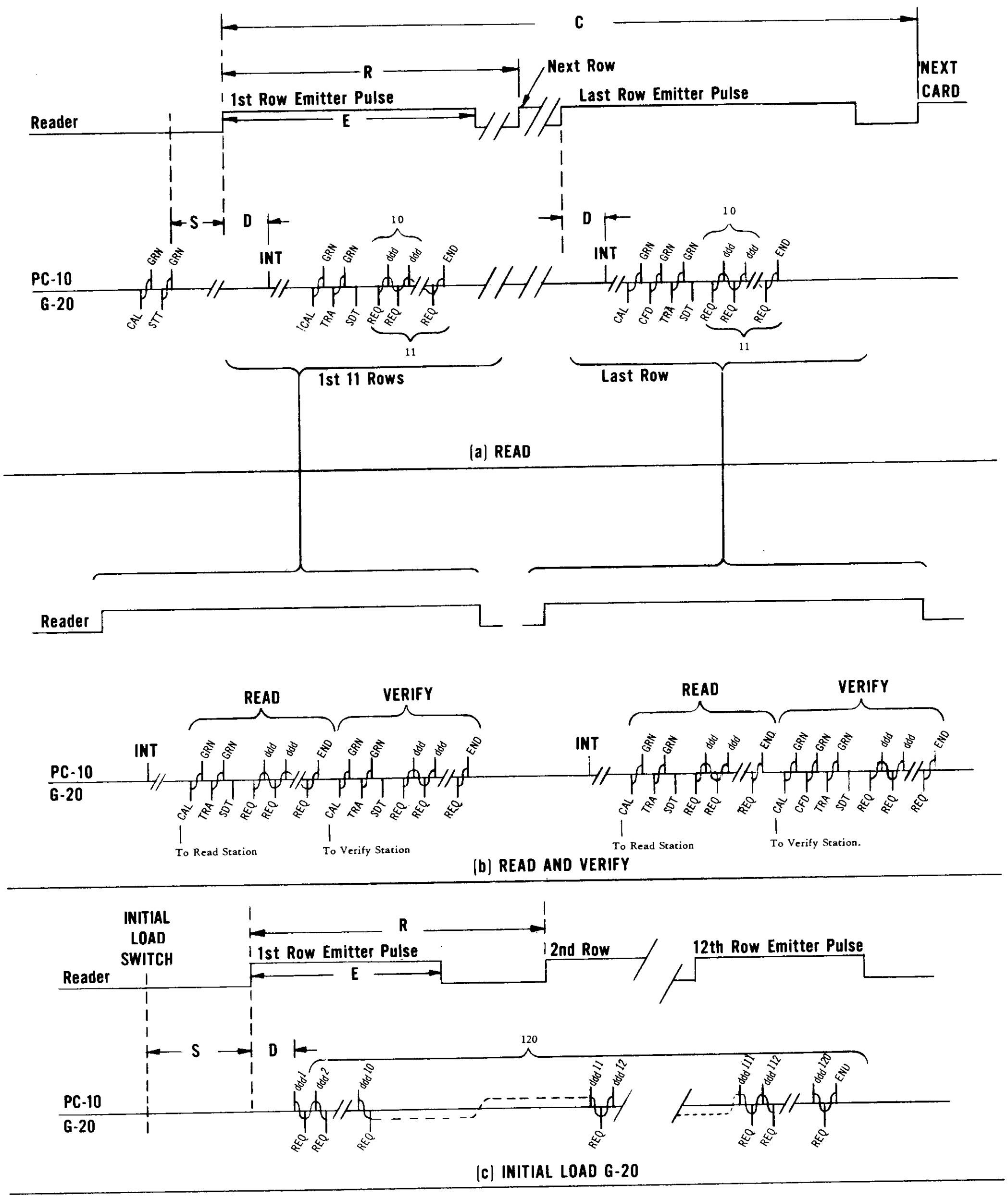
NR Normally not recommended for use with the Bendix G–20 System.

(a) Max. allowable time for reliable punching.

(b) CFD must be given within T milliseconds after last row to feed at maximum rate.

See Figure 6.5–3 for symbols.

## FIGURE 6.5-1 PC-10 CARD READ TIMING DIAGRAMS (a) READ (b) READ AND VERIFY (c) INITIAL LOAD G-20



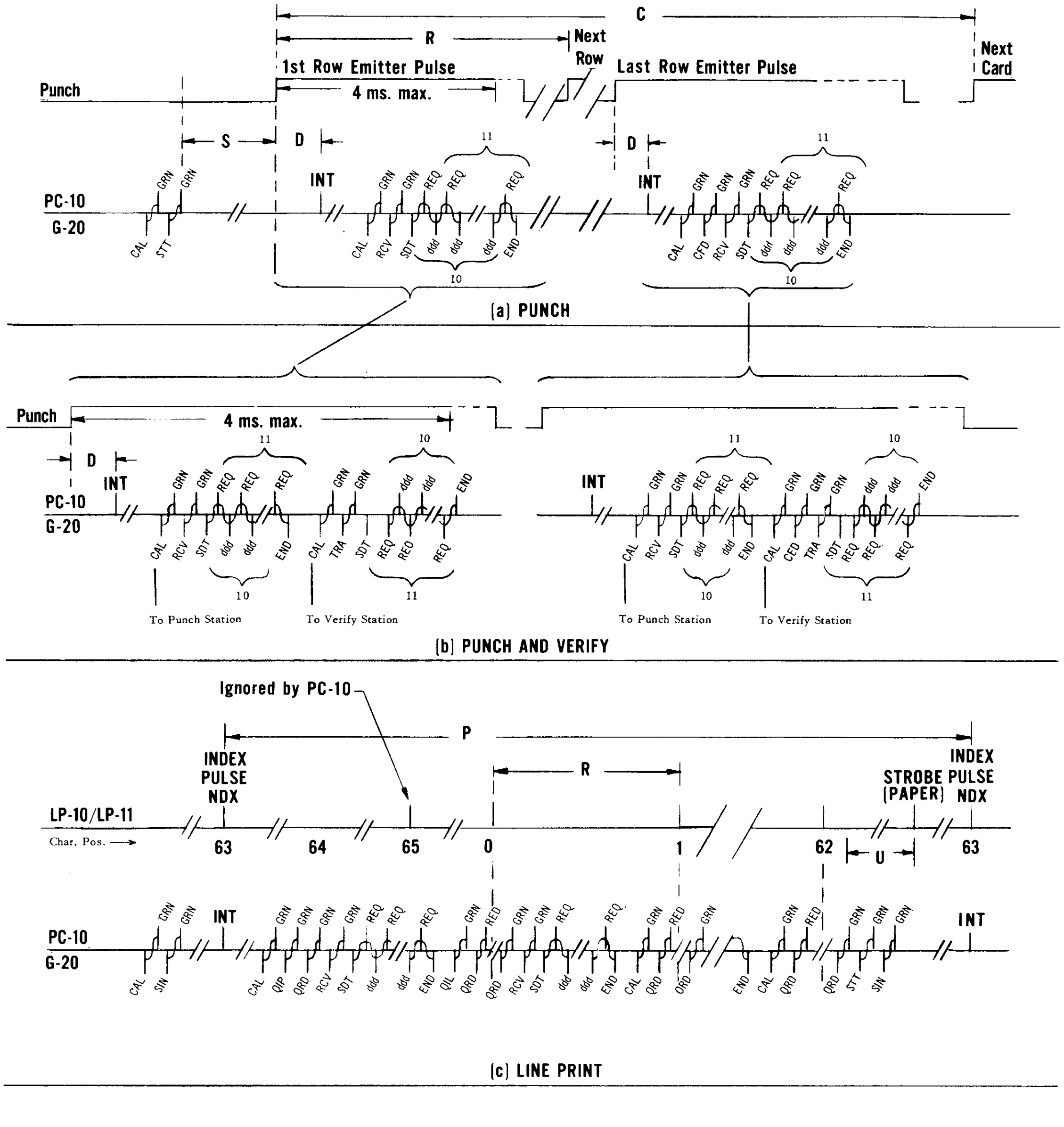
(a) READ

(b) READ AND VERIFY

(c) INITIAL LOAD G-20

NOTE: Commands used by the Interrupt Service Routine in establishing the source of interrupts are not shown.
A mark connecting two successive signals ┌┘ indicates that the second is an immediate automatic response to the first. Responses require 6 microseconds.
The command sequences shown are for illustrative purposes only.

## FIGURE 6.5-2 PC-10 CARD PUNCH AND LINE PRINT TIMING DIAGRAMS
## (a) PUNCH (b) PUNCH AND VERIFY (c) LINE PRINT



(a) PUNCH

(b) PUNCH AND VERIFY

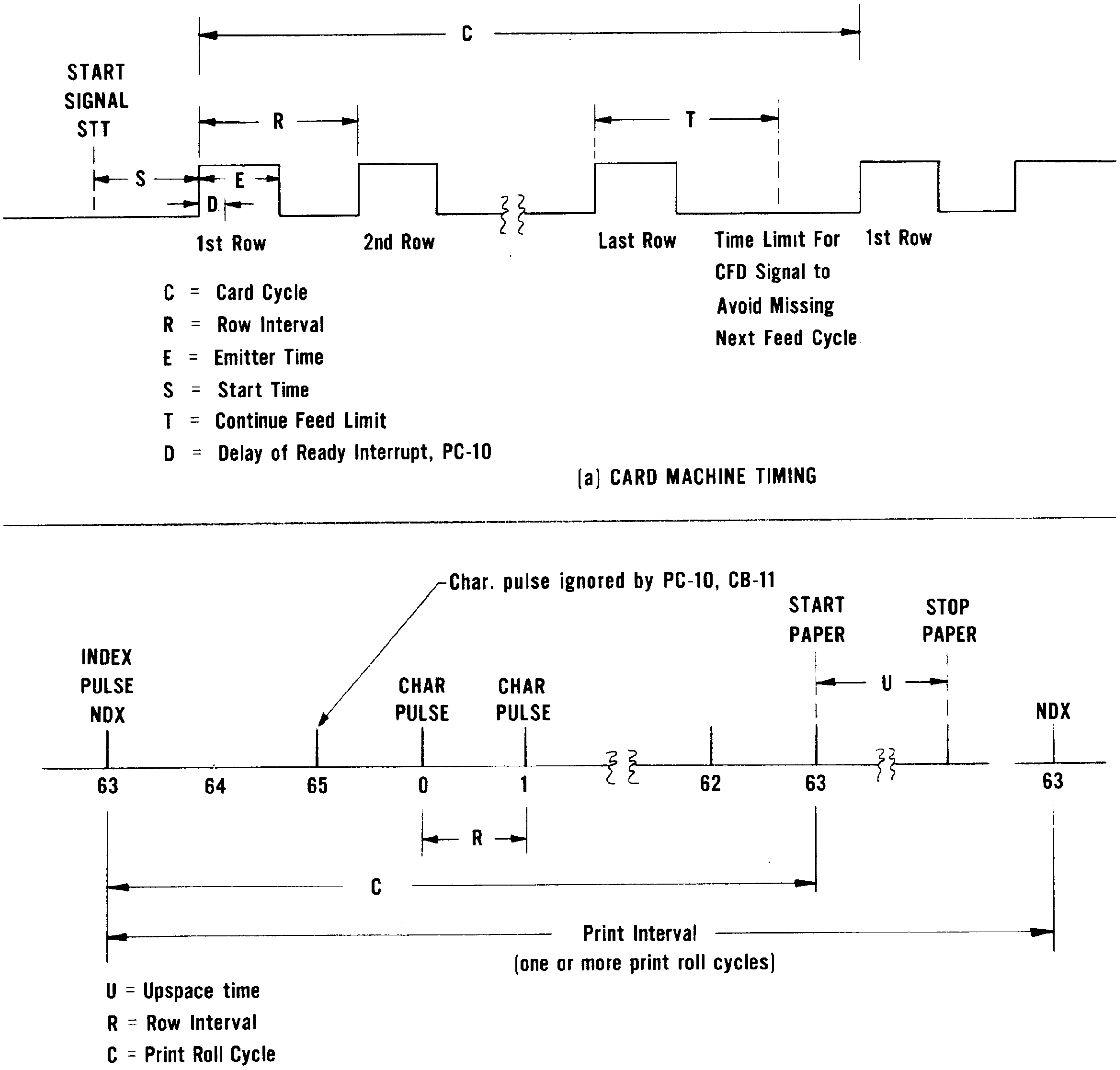
(c) LINE PRINT

NOTE: Commands used by the Interrupt Service Routine in establishing the source of interrupts are not snown.

A mark connecting two successive signals ⎍ indicates that the second is an immediate automatic response to the first. Responses require 6 microseconds.

The command sequences shown are for illustrative purposes only.

## FIGURE 6.5–3 KEY TO CARD AND PRINTER TIMING TABLES



(a) CARD MACHINE TIMING

(b) PRINTER TIMING

## SECTION 6.6 – CONTROL BUFFER TIMING

The timing of the CB–11 will be considered from three different aspects according to whether it is [a] sending or receiving line commands and replies, [b] sending or receiving a block of data, or [c] executing an internal program, including controlling a card machine or printer.

In case [a] the total time for an operation is composed of various intervals between signals and replies produced alternately by the CB–11 and the other party to the conversation.

In case [b] characters are transferred at a uniform rate ordinarily determined exclusively by the slower device in the exchange.

In case [c] it is simplest to consider the CB–11 as a single address computer and focus attention on the time for executing one command. Delays introduced while the CB–11 is waiting for a card machine or a printer will in general predominate over time for internal operations. See Figure 6.5–3 and Tables 6.5–2 and 6.5–3. For card operations the time will, in all ordinary cases, be just the time for the reader or punch. Conversion from serial to parallel format and vice versa can ordinarily be performed in the interval between successive emitter times. For a print operation using the "free wheeling mode" the total print cycle is the sum of upspace time and printing time. Image conversion is performed between rows and the printing can begin at any row. For fixed wheel printing see the discussion under printing in Section 6.5, noting that no consideration need be taken of image construction.

Figures 6.6–1 and 6.6–2 show typical situations involving line signals and replies as well as block data transmission. For internal operations use the times shown in Table 4.6–4 which includes command access, operand access, if any, and execution time.

## SECTION 6.7 – BUFFERED LINE PRINTER TIMING

Timing of communication between a Central Processor and the LP–12 can be determined from Tables 4.7–2, 4.7–3, and 6.1–3. Table 6.1–3 shows the overall time for block communication and for answering commands and queries. Tables 4.7–2 and 4.7–3 give the figures for each individual LP–12 code.

Timing of mechanical operations on the LP–12 occurs in multiples of the basic sector time of 15 milliseconds. The print roll revolves at 1,000 revolutions per minute, or 60 milliseconds per revolution. Printing takes place in units of one sector which is equivalent to 16 row times. To print with the full 64–character alphabet requires one whole revolution or 4 sector times per line. Printing with a 48–character alphabet requires 3 sector times, etc.

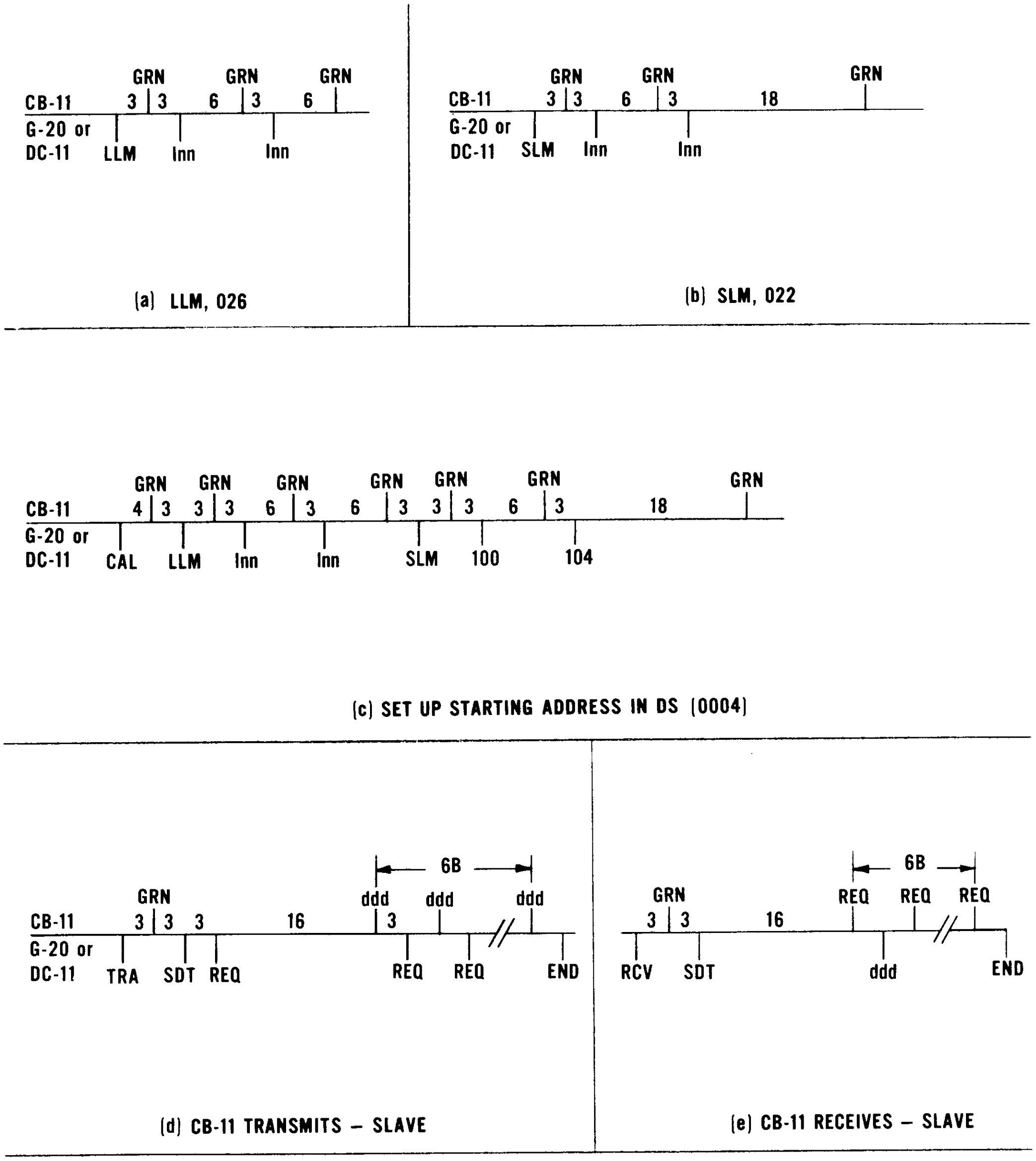
Paper spacing also takes place in increments of 15 milliseconds. The first space requires 15 milliseconds and each additional 2 spaces require 15 milliseconds. Because a print command can be issued during paper spacing and executed after the paper stops, and because, moreover, a paper spacing command can be issued during printing and executed after printing stops, there will be no delays if programs have been written properly. The characters to be printed can be loaded into the printer any time during the paper spacing cycle.

In printing alphanumeric with a 48–character alphabet, occupying sectors 0, 1, 2, it is possible to upspace the paper one line and reload the Print Buffer during sector 3, thus operating at the maximum speed of 1,000 lines per minute.

With smaller character sets such as letters only [sectors 0 and 1] or numbers only [sector 2], it is possible to operate at maximum speed with multiple spacing.

In case of the full alphabet [4 sectors] and single spacing, it is necessary to print for 4 sectors and upspace for one sector, for a total of 1 1/4 revolutions per line printed. This gives an effective speed of 800 lines per minute. Full alphabet printing with double or triple spacing requires 2 sector times for paper spacing and proceeds at an effective rate of 667 lines per minute.

FIGURE 6.6-1 CONTROL BUFFER TIMING DIAGRAMS - SLAVE OPERATIONS (a) LLM, 026, (b) SLM, 022, (c) SET UP STARTING ADDRESS IN DS [0004], (d) CB-11 TRANSMITS - SLAVE AND (e) CB-11 RECEIVES - SLAVE.



(a) LLM, 026

(b) SLM, 022

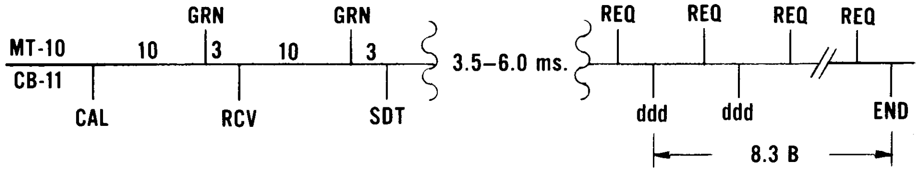
(c) SET UP STARTING ADDRESS IN DS (0004)

(d) CB-11 TRANSMITS - SLAVE

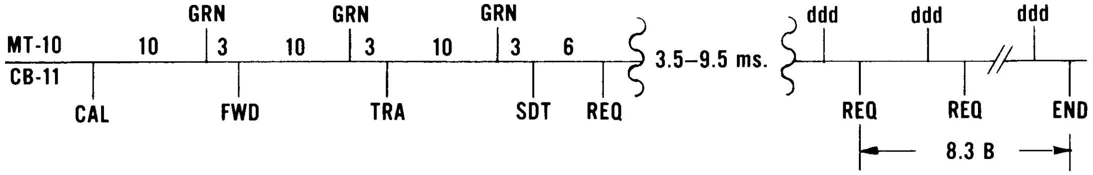
(e) CB-11 RECEIVES - SLAVE

NOTE: Times are shown in microseconds. B = Block length in characters.

FIGURE 6.6-2 CONTROL BUFFER TIMING DIAGRAMS - CONTROL TRANSMIT AND CONTROL MESSAGE (a) CB-11 WRITES ON MAGNETIC TAPE, AND (b) CB-11 READS MAGNETIC TAPE.



(a) CB-11 WRITES ON MAGNETIC TAPE



(b) CB-11 READS MAGNETIC TAPE

NOTE: Times are shown in microseconds unless noted otherwise. B = Block length in characters.

## SECTION 6.8 – DATA COMMUNICATOR TIMING

Because the function of the DC–11 is to handle multiplexed input to and output from external Core Memory, the most important aspect of its timing is the overall time for a block transfer. This information is shown in Table 6.8–1.

**TABLE 6.8–1 TIMING DC–11 BLOCK TRANSFER, MICROSECONDS PER CHARACTER**

| CODE | PT–10 | MT–10 | LP–12 | CB–11 |
|---|---|---|---|---|
| TR8 | 9.1 ms | 8.4 | 15. | 6. |
| TR6 | 9.1 ms | 8.4 | 15. | 6. |
| RC8 | 2./4. ms | 8.4 | — | 6. |
| RC6 | 2./4. ms | 8.4 | — | 6. |

# GLOSSARY OF TERMS

This glossary gives the meanings that have been attached to certain terms in the literature pertaining to the Bendix G–20 System. In general, these conform to the standards of the Association for Computing Machinery.

### ABSOLUTE ADDRESSING

Pertaining to fixed locations in memory as distinct from relative or relocatable addressing.

### ABSOLUTE VALUE

The magnitude of a number without regard to sign.

### ACCESS

The action of reading a word from memory. Also used as a verb.

### ACCESSORY

Any unit of the G–20 System not contained in the Central Processor.

### ACCUMULATOR

A hardware register containing the result of an arithmetic or logic operation.

### ADDRESS

A number or symbol signifying a location in which information is stored.

### ADDRESS, BASE

The address specified in a command or instruction before being modified by any index registers. The base address occupies the A field in a G–20 command.

### ADDRESS, INDEX

A number or a symbol signifying a location in which is stored a number to be used in modifying a base address. The index address can be in field A or field I.

### ADDRESS MODE

The number specified by bits 29 and 28 of a G–20 command, and whose function is to determine the manner in which the operand address is to be computed.

### ADDRESSING, GENERALIZED

Addressing that permits full use of the addressing facilities of the G–20.

### ADDRESSING, SECONDARY

An operation in which the first address specified is a location in which the actual address is to be found.

### ADDRESSING, SIMPLE

Addressing that makes use of no secondary or indexing features.

### ALGEBRAIC COMPILER

A program-generating routine in which the source language is algebraic.

### ALGORITHM

A procedure for obtaining a desired arithmetic result.

### ALGORITHM, DIVISION, NONRESTORING

A procedure in which the remainder is not restored after the divisor has been subtracted from it once too many times. Instead, at the next stage, addition is used instead of subtraction.

### ALPHANUMERIC CAPABILITY

A term used to signify that a piece of equipment is able to handle both letters and numbers in its memory and input/output operations.

### AUXILIARY CABLE

The part of the communication line containing the interrupt and system power lines.

### BIAS

In numerical analysis, a computation which produces an error systematically in one direction.

### BINARY

A number system based on Powers of 2.

### BIT

A contraction of binary digit.

### BLOCK

A group of successive locations in storage.

### BLOCK LIMIT

A number or symbol signifying the last element in a block.

### BOOLEAN

Of, or pertaining to, an operation involving symbolic logic. From George Boole who first set forth the principles of symbolic logic.

### BUFFERED OPERATION

Operation of an accessory device under control of the Control Buffer rather than the Central Processor.

### CASCADING [of addresses]

A serial operation in which the contents of each location accessed is used as the address of the next location to be accessed.

### CENTRAL PROCESSOR

The device in the functional center of a computing system which contains the Master Control and Arithmetic Unit. In the G–20 the Central Processor also contains 4,096 or 8,192 words of core memory.

### CHARACTER

A group of bits constituting a single transmission over the G–20 communication line. Also, a group of bits in a G–20 word which may represent a particular alphabetic, numeric, or other type symbol.

### CLOCK CYCLE

The elapsed time between two compute pulses in a synchronously timed computer. In the G–20 this is one microsecond.

### CODE

A synthetic language used in data processing.

### CODE, CODING

To prepare a set of commands for a digital computer. Also the resulting set of commands.

### COMMAND

The coded signal causing the computer to perform a particular operation. Also, a code composed of letters or other symbols but having a one-to-one correspondence with a hardware code.

### COMMAND, LINE

A command transmitted from one unit to another over the communication system, distinguished from a G–20 Central Processor command.

### COMMUNICATION

Transmission of characters or signals from one unit to another over the communication line.

### COMMUNICATION CODE

Any one of 512 possible configurations of bits which can be sent over the communication lines. These include data characters and command characters.

### COMMUNICATION LINE

A group of wires connecting one portion of a G–20 System to another portion of the system.

### COMPUTER

A device capable of performing arithmetic and logical operations. Usually a computer having an internally stored program is also implied.

### CONSOLE

An operating desk or area from which control can be exercised by an operator.

### CONTROL BUFFER

A G–20 accessory having the capacity to store data and to control its own operation and that of input/output devices.

### CORE MEMORY

A memory composed of small ring shaped permanent magnets [cores].

### DATA COMMUNICATOR

A G–20 accessory permitting multiple simultaneous input/output from Core Memory.

### DEBUG

To correct errors in a program or a portion of hardware.

### DESTRUCTIVE READ

A method of determining the state of magnetization of a core memory element in which an attempt is made to magnetize the element in the zero state. If a change in the state of magnetization occurs, it will mean that the element which was originally stored as a "1" has now been destroyed.

### DICTIONARY

A tabulation in computer memory identifying one set of symbols with another, or a set of symbols with memory locations, etc.

### DIRECT INPUT/OUTPUT

Communication between the Core Memory and the terminal equipment, without the interposition of a buffer, distinguished from buffered input/output.

### DOUBLE PRECISION

See Precision.

### EXPONENT

A power of 8 [or other base] by which the mantissa of a floating point number is to be multiplied.

### EXTRACT

A logic operation in which those bits of an operand are read which correspond to the "1" bits of a number called the extractor. Boolean AND.

### FAULT

An incorrect or illegal command, an arithmetic overflow, or a hardware failure.

### FIELD

A group of bits in a command or a group of columns on a card which are treated together for purposes of the computation.

### FLAG

A bit in a command word or character, or a bit in a hardware register or memory location used to signify a particular condition.

### FLOATING POINT

A mode of computation in which a quantity is expressed as a certain number of limited magnitude range multiplied by an integral power of a base such as 2, 8, 10, or 16.

### FLOW CHART

A diagram used to depict the flow of information in a program, without actually indicating the individual coding steps.

### GATHER-WRITE

A method of writing on magnetic tape or other storage medium in which a single block of the recording originates from several areas of the memory of the transmitting device.

### HARDWARE COMMAND

The actual binary code designating a particular operation in the computer.

### HOLLERITH CODE

A type of punched card code in which one card column represents one character. Hollerith is the code ordinarily produced by a key punch; distinguished from binary codes.

### HOLLERITH CODE, EXTENDED

A form of "one character per column" card code in which there are 256 possible characters. Also, the octal representation corresponding to such a card code.

### INDEX REGISTER

A hardware register or memory location containing tally information for repeating a series of commands. Ordinarily the index is used to modify an address or to store a count of operations. In the G–20 locations 1–63 of Core Memory are Index registers.

### INSTRUCTION

A set of characters denoting an operation to be performed. An instruction need not have any simple relation to a hardware command.

### INTEGER FLOATING POINT OCTAL

The type of arithmetic performed in the G–20 wherein a number is represented as a 7– or 14–octal integer multiplied by a positive or negative power of 8.

### INTERRUPT, INTERRUPT REQUEST

A halt in computation or a request for a halt in computation.

### INTERRUPT, FAULT

An interrupt necessitated by a fault. See also Fault.

### INTERRUPT, FLAG

An interrupt caused by an enabled flag on a word.

### INTERRUPT, INPUT/OUTPUT

Interrupt originating because of a signal transmitted from one unit to another over the communication lines.

### INTERRUPT, INTERNAL

An interrupt the cause for which originated in the Central Processor rather than in the input/output system.

### INTERRUPT MODE

A mode of operating one or more accessories or programs such that an activity which requires temporary use of the Central Processor may interrupt the activity occupying the Central Processor.

### INTERSECTION

A group of "1" bits common to two numbers. See also Extract. Result of a Boolean AND.

### JUMP

An unconditional transfer of control.

### KEY

A group of characters used to identify or characterize an item. In sorting a group of items the Keys are arranged in ascending or descending order.

### LANGUAGE

Any set of codes or symbols used in communication between elements of the G–20 System or between the system and the outside world.

### LINE PRINTER

An output device which produces a horizontal line of print in a single operation.

### LINE SIGNAL

Any communication between units of the G–20 System.

### LOAD

To place a number in a memory location or hardware register.

### LOGIC FORMAT

In the G–20 that convention by which logic quantities are represented in the accumulator and in memory. See the text.

### LOGIC OPERATION

A logical union, intersection, negation, etc. performed by the computer hardware in accordance with set theoretic rules.

### LOGIC WORD

A word whose bits are interpreted as individual yes/no decisions, not necessarily related to each other.

### MACHINE LANGUAGE

The actual binary codes used inside the G–20 System for designating particular operations. More loosely, any set of codes having a very close connection with machine language.

### MANTISSA

Originally the fractional part of a logarithm, hence in floating point computation the portion of the number containing the significant digits as distinguished from the portion specifying the exponent of the multiplier.

### MARK PLACE

To store in a known location, from which it can be subsequently recovered, the address of the command next in sequence.

### MEMORY

The medium in which information (i.e., commands, data, etc.) is stored for later use by the computer, Also called "store".

### MODULE

A hardware building block of standard size established for simplicity in manufacture and maintenance.

### NON-BIASING

See under Bias.

### NORMALIZE

To place an arithmetic quantity in a standard form. Normal form for the G–20 has a non-zero leading digit.

### NORMALIZE, BINARY

To shift a number to the left until its leading bit is a one.

### NORMALIZE, OCTAL

To shift a number to the left three bits at a time until its leading group of three bits is non-zero.

### NUMBER FORMAT

In the G–20 that convention by which numerical quantities are represented in the Accumulator and in memory. See the text.

### OBJECT PROGRAM

The machine language program produced by translation from a source language in which the program is coded.

### OCTAL

Of or pertaining to the number system based on powers of 8.

### OCTALIZE

To shift a binary mantissa to the right until its least significant three bits represent one octal digit. One or two bits at the right will be lost.

### OFF-LINE

Pertaining to a group of elements disconnected from the Central Processor.

### OPCODE

See under Operation Code.

### OPCODE, CLASS 1

An opcode whose operand designator refers to an operand.

### OPCODE, CLASS 2

An opcode whose operand designator refers to an address.

### OPERATION CODE

A group of symbols signifying an operation to be performed by the computer. An operation code can be a command but may also, in some contexts, refer to a code not closely related to the hardware.

### OPERAND

The number on which an arithmetic or logic operation is to be performed.

### OPERAND ASSEMBLY

The final operand or operand address for a G–20 operation.

### OPERAND ASSEMBLY REGISTER

A hardware register in which the operand or address is assembled.

### OPERAND DESIGNATOR

In the G–20 the number prepared from the base address and index address fields of a command and used to specify the operand or the address of the operand.

### OVERFLOW

The result of an attempt to generate a number too large to be contained in a particular register.

### PARALLEL ARITHMETIC

Arithmetic which is performed on all the digits of the number simultaneously, as distinguished from arithmetic performed one digit at a time.

### PARITY

The residue module 2 of the sums of the digits of a number, i.e. the excess of the sum of the digits over the nearer, smaller [or equal] multiple of 2. A quantity which has an odd number of "1" bits has "odd" parity. In the G–20 memory an extra bit is annexed to each word so as to make the parity odd. On the G–20 communication line the parity is even. A hardware error which involves a single bit will be detected because the parity will then be incorrect.

### PICKAPOINT

A special mode of storage for the G–20, a floating point computer, in which the number is automatically shifted to a form having a previously determined exponent.

### PRECISION

The number of digits carried in a computation in the G–20. Double precision makes use of all the 14 octal digits generated by the Arithmetic Unit, while single precision makes use of 7 [or in pickapoint, 9] of the digits generated.

### PREPARATION OPCODE

A special class of commands in the G–20 used principally for computing addresses.

### PROCEDURE

A logical set of coding used by one or more programs or sub-programs. Same as a subroutine.

### PROCESSOR

See Central Processor.

### PROGRAM

A set of instructions for performing a computation.

### PROGRAMMING, BASIC

Programming using a simple assembly routine. Basic programming is sometimes loosely referred to as machine language programming, although the codes used are not actually the hardware codes but merely symbols for the hardware codes.

### PROGRAMMING, GENERALIZED

Preparation oruse of a program in which the principal parameters may be changed to suit a particular problem. Generalized programming might for example use the same program for updating a master payroll list, updating inventory, or updating backlog orders.

### PROGRAMMING, MACHINE LANGUAGE

Programming in absolute binary or octal using the codes which appear in the hardware. Machine language programming is not recommended for the G–20.

### PROGRAMMING, SYMBOLIC

Programming in which quantities and memory locations can be referred to by arbitrary symbols, the meanings of which are stored in a dictionary.

### READ

To recover information from storage [e.g., core memory, magnetic tape, or paper tape].

### REGISTER

A group of flip-flops, or a storage location, in which numbers or control information can be stored.

### REGISTER, ACCUMULATOR

See Accumulator.

### REGISTER, BUS

One of five hardware registers used in control of the G–20. Usually abbreviated to register.

### REGISTER, OPERAND ASSEMBLY

See Operand Assembly Register.

### RELOCATABLE

Pertaining to a body of data or code which is so constructed that it can be moved around in memory without changing its meaning or operation.

### REPEAT COMMAND

A special class of 32 commands in the G–20. A repeat command continues to operate on successive memory locations until terminated by a block limit or an interrupt.

### RESTORE

To rewrite in memory a number destroyed by reading. See Destructive Read.

### REVERSE DIVIDE

A division in which the number obtained is the operand divided by the contents of the Accumulator.

### REVERSE SUBTRACT

A subtraction in which the answer obtained is the operand minus the contents of the Accumulator.

### ROUND

To discard according to a prescribed rule some of the least significant digits of a quantity. Some rounding rules produce bias. The G–20 uses a non-biasing rounding rule.

### ROUTINE

A generalized program or program preparation sequence.

### ROUTINE, SERVICE

Same as Routine.

### ROW BINARY

Mode of punched card operation in which the bits of a quantity are represented as a horizontal row of holes.

### SCATTER-READ

A method of reading from one storage medium to another, such that a single block of data from the first medium is distributed over several areas of the second medium.

### SECURE

To secure a program is to halt computation and make a record of the contents of pertinent hardware registers and of the current command location.

### SEGMENTATION

The decomposition of a program into smaller portions.

### SHIFT

To move a number left or right in relation to the radix point.

### SHIFT, BINARY

To move a number by single bit increments.

### SHIFT, OCTAL

To move a number by increments of three bits.

### SIGNAL, LINE

See under Line Signal.

### SINGLE ADDRESS

Said of a computer each of whose command words specifies a single operand or location. Distinguished from TWO ADDRESS, etc.

### SINGLE PRECISION

See Precision.

### SLEW

A term used with G–20 Magnetic Tape to signify a rapid non-stop repositioning to a desired block.

### SOURCE INSTRUCTION

An instruction written in a source language.

### SOURCE LANGUAGE

The language in which a program is originally stated, usually oriented to the programmer or to the problem.

### STORAGE

The act of placing a group of digits in core memory, on magnetic tape, on paper tape, etc.

### STRING

A group of successive characters.

### SUBROUTINE

A logical segment of coding so prepared that it can be used by a number of programs.

### TRANSFER CONTROL

Programmer terminology meaning to execute next some other command than the one next in sequence.

### TRUNCATE

To discard digits from either end of a number without the use of a round-off rule.

### UNDERFLOW

An attempt to generate a number too small in magnitude to be expressed in the registers available. In the G-20 this results in the number being replaced by zero.

### UNION

A number containing a bit in any of the locations where either of the operands contains a bit. There can also be a union formed from a number of quantities. Boolean OR.

### UNITE

To perform the logic involved in a union.

### WORD

The contents of one of the core storage locations of the G–20 interpreted as a command, a floating point number, etc.

### WRITE

To record information in some form of storage.

### ZONE

In Hollerith card operations, a punch in one of the first three positions in a column is known as a zone punch. In Bendix Extended Hollerith, the entire column is divided into four zones.

# APPENDIX I

# CENTRAL PROCESSOR OPERATION CODES AND TIMING NUMBERS

The timing number indicates the time for executing a command in addition to access time.

For a non-repeated command the first digit is the number of clock cycles to be added to the access time for modes 2 and 3 with or without an index. The second digit is the amount to be added for modes 0 and 1 with an index; without an index subtract one clock cycle from the time with an index.

For repeat commands the first digit is the amount to be added to the access time for each operand. The second digit is the amount to be added to the access time to get set-up time.

The total time figures are also shown in the tables in Appendix II.

The figures [30/40 and 69/55] on multiplication and division indicate the average time for the arithmetic operation proper, for single and double precision, respectively. See also Section 6.1.

**CENTRAL PROCESSOR OPERATION CODES AND TIMING NUMBERS**

| USERS CODE | OCTAL CODE | TIMING NO. | REF. PAGE | NAME |
|---|---|---|---|---|
| ADA | 145 | 0–3 | 13 | Add and take absolute value. |
| ADD | 045 | 0–3 | 13 | Add. |
| ADL | 055 | 0–3 | 14 | Add logic word. |
| ADN | 105 | 0–3 | 13 | Add and negate. |
| ADX | 002 | 1–2 | 21 | Add to index. |
| AXT | 006 | 1–2 | 21 | Add to index and test. |
| BRD6 | 033<br>020 | Note 1 | 23 | Block receive data in 6–bit characters.<br>See Note 3 |
| BRD8 | 033<br>120 | Note 1 | 23 | Block receive data in 8–bit characters.<br>See Note 3 |
| BTC6 | 033<br>040 | Note 1 | 23 | Block transmit commands in 6–bit characters.<br>See Note 3 |
| BTC8 | 033<br>140 | Note 1 | 23 | Block transmit commands in 8–bit characters.<br>See Note 3. |
| BTD6 | 033<br>000 | Note 1 | 23 | Block transmit data in 6–bit characters.<br>See Note 3 |
| BTD8 | 033<br>100 | Note 1 | 23 | Block transmit data in 8–bit characters.<br>See Note 3. |
| CAL | 015 | 0–0 | 14 | Clear and add logic word. |
| CCL | 035 | 0–0 | 14 | Clear and add complement of logic word. |
| CLA | 005 | 0–0 | 13 | Clear and add. |
| CLS | 025 | 0–0 | 13 | Clear and subtract. |
| DIV | 053 | 0–1<br>+69/55 | 18 | Divide. |
| ECL | 135 | 0–0 | 14 | Extract with complement of logic word. |
| ERA | 072 | 1–2 | 25 | Extract register into Acc. |
| ERO | 052 | 0–1 | 25 | Extract register into OA. |
| EXL | 115 | 0–0 | 14 | Extract with logic word. |
| EXR | 076 | 0–0 | 25 | Extract to register. |
| FGO | 061 | 1–5 | 13 | If greater than operand. |

## CENTRAL PROCESSOR OPERATION CODES AND TIMING NUMBERS (Continued)

| USERS CODE | OCTAL CODE | TIMING NO. | REF. PAGE | NAME |
|---|---|---|---|---|
| FLO | 121 | 1–5 | 13 | If less than operand. |
| FOM | 021 | 0–1 | 13 | If operand minus. |
| FOP | 001 | 0–1 | 13 | If operand plus. |
| FSM | 101 | 1–5 | 13 | If sum minus. |
| FSN | 141 | 1–5 | 13 | If sum non–zero. |
| FSP | 041 | 1–5 | 13 | If sum plus. |
| FUO | 161 | 1–5 | 13 | If unequal to operand. |
| ICZ | 031 | 2–3 | 15 | If complement zero. |
| IEC | 131 | 2–3 | 15 | If extraction with complement zero. |
| IEZ | 111 | 2–3 | 15 | If extraction zero. |
| IOZ | 011 | 2–3 | 15 | If operand zero. |
| ISN | 051 | 2–5 | 15 | If sum non-zero. |
| IUC | 171 | 3–4 | 15 | If union with complement zero. |
| IUO | 071 | 2–5 | 15 | If logic word unequal to operand. |
| IUZ | 151 | 3–4 | 15 | If union zero. |
| LDR | 056 | 0–0 | 25 | Load register. |
| LXM | 032 | 1–2 | 21 | Load index minus. |
| LXP | 012 | 1–2 | 21 | Load index plus. |
| MPY | 077 | 0–1<br>+ 30/40 | 18 | Multiply |
| OAA | 140 | 0–4 | 12 | Add and take absolute value; [answer to OA]. |
| OAD | 040 | 0–4 | 12 | Add; [answer to OA]. |
| OAN | 100 | 0–4 | 12 | Add and negate; [answer to OA]. |
| OCA | 000 | 0–0 | 12 | Clear and add; [answer to OA]. |
| OCS | 020 | 0–0 | 12 | Clear and subtract; [answer to OA]. |
| OSA | 160 | 0–4 | 12 | Subtract and take absolute value; [answer to OA]. |
| OSN | 120 | 0–4 | 12 | Subtract and negate; [answer to OA]. |
| OSU | 060 | 0–4 | 12 | Subtract; [answer to OA]. |
| RADA | 013<br>145 | 2–0 | 16 | Repeat add and take absolute value. |
| RADD | 013<br>045 | 2–0 | 16 | Repeat add. |
| RADL | 013<br>055 | 2–0 | 17 | Repeat add logic word. |
| RADN | 013<br>105 | 2–0 | 16 | Repeat add and negate. |
| RCAL | 013<br>015 | 2–0 | 17 | Repeat clear and add logic word. |
| RCCL | 013<br>035 | 2–0 | 17 | Repeat clear and add complement of logic word. |
| RCLA | 013<br>005 | 1–0 | 16 | Repeat clear and add. |
| RCLS | 013<br>025 | 1–0 | 16 | Repeat clear and subtract. |

## CENTRAL PROCESSOR OPERATION CODES AND TIMING NUMBERS (Continued)

| USERS CODE | OCTAL CODE | TIMING NO. | REF. PAGE | NAME |
|---|---|---|---|---|
| RDV | 057 | 0–1<br>+69/55 | 18 | Reverse divide. |
| RECL | 013<br>135 | 2–0 | 17 | Repeat extract with complement of logic word. |
| REXL | 013<br>115 | 2–0 | 17 | Repeat extract with logic word. |
| RFGO | 013<br>061 | 2–2 | 17 | Repeat if greater than operand. |
| RFLO | 013<br>121 | 2–2 | 17 | Repeat if less than operand. |
| RFOM | 013<br>021 | 1–2 | 17 | Repeat if operand minus. |
| RFOP | 013<br>001 | 1–2 | 17 | Repeat if operand plus. |
| RFSM | 013<br>101 | 2–2 | 17 | Repeat if sum minus. |
| RFSN | 013<br>141 | 2–2 | 17 | Repeat if sum non-zero. |
| RFSP | 013<br>041 | 2–2 | 17 | Repeat if sum plus. |
| RFUO | 013<br>161 | 2–2 | 17 | Repeat if unequal to operand. |
| RICZ | 013<br>031 | 3–2 | 18 | Repeat if complement zero. |
| RIEC | 013<br>131 | 3–2 | 18 | Repeat if extraction with complement zero. |
| RIEZ | 013<br>111 | 3–2 | 18 | Repeat if extraction zero. |
| RIOZ | 013<br>011 | 3–2 | 18 | Repeat if operand zero. |
| RISN | 013<br>051 | 3–2 | 18 | Repeat if sum non-zero. |
| RIUC | 013<br>171 | 4–2 | 18 | Repeat if union with complement zero. |
| RIUO | 013<br>071 | 3–2 | 18 | Repeat if logic word unequal to operand. |
| RIUZ | 013<br>151 | 4–2 | 18 | Repeat if union zero. |
| RSUA | 013<br>165 | 2–0 | 16 | Repeat subtract and take absolute value. |
| RSUB | 013<br>065 | 2–0 | 16 | Repeat subtract. |

## CENTRAL PROCESSOR OPERATION CODES AND TIMING NUMBERS (Continued)

| USERS CODE | OCTAL CODE | TIMING NO. | REF. PAGE | NAME |
|---|---|---|---|---|
| RSUL | 013<br>075 | 2–0 | 17 | Repeat subtract logic word. |
| RSUN | 013<br>125 | 2–0 | 16 | Repeat subtract and negate. |
| RUCL | 013<br>175 | 3–0 | 17 | Repeat unite with complement of logic word. |
| RUNL | 013<br>155 | 3–0 | 17 | Repeat unite with logic word. |
| SKP | 137 | 2–3 | 22 | Skip. |
| STD | 153 | 3–4 | 19 | Store double precision. |
| STI | 133 | 2–3<br>+ shifts | 19 | Store integer. |
| STL | 173 | 2–3 | 19 | Store logic word. |
| STS | 113 | Note 2 | 19 | Store single precision. |
| STZ | 073 | 2–3 | 19 | Store zero. |
| SUA | 165 | 0–3 | 13 | Subtract and take absolute value. |
| SUB | 065 | 0–3 | 13 | Subtract. |
| SUL | 075 | 0–3 | 14 | Subtract logic word. |
| SUN | 125 | 0–3 | 13 | Subtract and negate. |
| SUX | 022 | 1–2 | 21 | Subtract from index. |
| SXT | 026 | 1–2 | 21 | Subtract from index and test. |
| TDC | 117 | Note 1 | 22 | Transmit data character. |
| TLC | 157 | Note 1 | 22 | Transmit line command. |
| TRA | 017 | 0–0 | 22 | Transfer. |
| TRE | 037 | 0–0 | 22 | Transfer and enable interrupts. |
| TRM | 177 | 3–4 | 22 | Transfer and mark. |
| UCL | 175 | 0–0 | 14 | Unite with complement of logic word. |
| UNL | 155 | 0–0 | 14 | Unite with logic word. |
| XMT | 036 | 1–2 | 21 | Load index minus and test. |
| XPT | 016 | 1–2 | 21 | Load index plus and test. |

NOTE 1: See Table 6.1–3

NOTE 2: Timing for STS floating point mode: 2–3 + shifts.
Timing for STS pickapoint mode: 3–4 + shifts.

NOTE 3: The opcode information for the second word is shown in standard octal opcode notation. For the actual bits see Table 2.3–14 and Figure 2.3–4.

# APPENDIX II
# TIMING CHARTS

**NON-REPEATED CENTRAL PROCESSOR COMMANDS**

| MODE AND INDEX | X | TIMING NUMBER | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 – 0 | | 0 – 1 | | 0 – 3 | | 0 – 4 | | 1 – 2 | |
| | | S | D | S | D | S | D | S | D | S | D |
| 0 | (OA) + A | 6 | 6 | 6 | 6 | 8 | 8 | 9 | 9 | 7 | 7 |
| 0 1 | (OA) + A + (I) | 12 | 12 | 13 | 13 | 15 | 15 | 16 | 16 | 14 | 14 |
| 1 | (OA) + (A) | 12 | 18 | 12 | 18 | 14 | 20 | 15 | 21 | 13 | 19 |
| 1 I | (OA) + (A) + (I) | 18 | 24 | 19 | 25 | 21 | 27 | 22 | 28 | 20 | 26 |
| 2 | ((OA) + A) | 12 | 18 | 12 | 18 | 12 | 18 | 12 | 18 | 13 | 19 |
| 2 I | ((OA) + A + (I)) | 18 | 24 | 18 | 24 | 18 | 24 | 18 | 24 | 19 | 25 |
| 3 | ((OA) + (A)) | 18 | 24 | 18 | 24 | 18 | 24 | 18 | 24 | 19 | 25 |
| 3 I | ((OA) + (A) + (I)) | 24 | 30 | 24 | 30 | 24 | 30 | 24 | 30 | 25 | 31 |
| | | Note 1 | | | | | | | | Note 2 | |

| MODE AND INDEX | X | TIMING NUMBER | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 1 – 5 | | 2 – 3 | | 2 – 5 | | 3 – 4 | |
| | | S | D | S | D | S | D | S | D |
| 0 | (OA) + A | 10 | 10 | 8 | 8 | 10 | 10 | 9 | 9 |
| 0 I | (OA) + A + (I) | 17 | 17 | 15 | 15 | 17 | 17 | 16 | 16 |
| 1 | (OA) + (A) | 16 | 22 | 14 | 20 | 16 | 22 | 15 | 21 |
| 1 I | (OA) + (A) + (I) | 23 | 29 | 21 | 27 | 23 | 29 | 22 | 28 |
| 2 | ((OA) + A) | 13 | 19 | 14 | 20 | 14 | 20 | 15 | 21 |
| 2 I | ((OA) + A + (I)) | 19 | 25 | 20 | 26 | 20 | 26 | 21 | 27 |
| 3 | ((OA) + (A)) | 19 | 25 | 20 | 26 | 20 | 26 | 21 | 27 |
| 3 I | ((OA) + (A) + (I)) | 25 | 31 | 26 | 32 | 26 | 32 | 27 | 33 |
| | | | | Note 3 | | | | Note 4 | |

All times are in clock cycles. S = Single precision. D = Double precision. (OA) is taken to be double precision. (I) is taken to be single precision. Final access is single or double precision as indicated. Intermediate accesses are taken to be single precision. If one or more of the accesses is in external memory, add 1 cycle to the execution time for modes 1, 2, 2I, 3, 3I.

Notes: 1. In OCA, OCS there is 1 cycle available for arithmetic in modes 0I, 1I, and 2 cycles in all other modes. In CLA, CLS, TRA, TRE, LDR, EXR, there is 1 cycle available for arithmetic except in modes 0I, 1I.

2. For index codes, there is an extra memory access to store the modified index; therefore, 6 cycles must be added to these times.

3. Store commands have an extra access; therefore, add 6 cycles.

4. For TRM, STS pickapoint, add 6 cycles. For STD use first column and add 12.

5. For storage in external memory, subtract 3 cycles in modes 0, 0I, 1I, and 2 cycles for the other modes.

## REPEATED CENTRAL PROCESSOR COMMANDS SET-UP TIME

| MODE AND INDEX | X | TIMING NUMBER<br>1 – 0<br>2 – 0<br>3 – 0 | TIMING NUMBER<br>1 – 2<br>3 – 2<br>4 – 2 |
|---|---|---|---|
| 0 | (OA) + A | 12 | 14 |
| 0 I | (OA) + A + (I) | 18 | 20 |
| 1 | (OA) + (A) | 18 | 20 |
| 1 I | (OA) + (A) + (I) | 24 | 26 |
| 2 | ((OA) + A) | 18 | 20 |
| 2 I | ((OA) + A + (I)) | 24 | 26 |
| 3 | ((OA) + (A)) | 24 | 26 |
| 3 I | ((OA) + (A) + (I)) | 30 | 32 |

## REPEATED CENTRAL PROCESSOR COMMANDS EXECUTION TIME PER OPERAND

| TIMING NUMBER | TIME PER OPERAND | |
|---|---|---|
| | S | D |
| 1 – 0 | 7 | 13 |
| 1 – 2 | 7 | 13 |
| 2 – 0 | 8 | 14 |
| 3 – 0 | 9 | — |
| 3 – 2 | 9 | — |
| 4 – 2 | 10 | — |

All times are in clock cycles. S=Single precision. D = Double precision.

If one or more of the set-up accesses is in external memory, add 1 cycle to set-up time in modes 1, 2, 2I, 3, 31.

If the operands are in external memory, add 1 cycle to execution time for each single or double precision operand.

# APPENDIX III
# ALPHABETIC LIST OF COMMUNICATION LINE SIGNALS AND BUFFER CODES

| ALPHA | OCTAL | APPLICABILITY | NAME |
|---|---|---|---|
| API | 055 | LP–12 | Advance Paper and Interrupt. |
| AS1 | 051 | CB–11<br>PC–10 | Alternate Stacker 1 [card equipment]. |
| AS2 | 052 | CB–11<br>PC–10 | Alternate Stacker 2 [card equipment]. |
| BAC | 045 | MT–10<br>PT–10 | Backward. |
| BCR | 023 | CB–11 | Block Clear. |
| BCS | 041 | CB–11 | Block Character Switch. |
| BLK | 013 | PT–10 | Block mode. |
| BTF | 043 | CB–11 | Block Transfer. |
| BTL | 017 | CB–11 | Block Test Less than operand. |
| BTU | 015 | CB–11 | Block Test Unequal to operand. |
| CAL | 200 + uuu | All Units | Abbreviation used to indicate call signal to unit uuu. |
| CFD | 054 | CB–11<br>PC–10 | Continue Feed [of cards or paper]. |
| CLB | 050 | LP–12 | Clear Buffer. |
| CLQ | 001 | PC–10 | Clear Query flip-flops. |
| COF | 053 | CB–11 | Control Off. |
| CON | 034 | CB–11 | Control On. |
| CPI | 024 | CB–11 | Construct Print Image. |
| CSF | 036 | CB–11 | Compare SF with operand. |
| DBM | 056 | MT–10 | Delete. |
| DTZ | 047 | DC–11 | Decrement and Test for Zero. |
| END | 004 | All Units[1] | End of data block. |
| ERR | 005 | All Units[1] | Error. |
| FWD | 044 | MT–10<br>PT–10 | Forward. |
| GRN | 002 | All Units | Green. |
| HRB | 021 | CB–11 | Convert Extended Hollerith to Row Binary. |
| INC | 037 | CB–11 | Increment (LM). |
| INT | | All Units | Interrupt. Signal sent on an interrupt line. |

| ALPHA | OCTAL | APPLICABILITY | NAME |
|---|---|---|---|
| IWR | 046 | MT–10 | Initial Write. |
| JEQ | 045 | CB–11 | Jump if Equal. |
| JMP | 040 | CB–11 | Jump. |
| JLO | 047 | CB–11 | Jump if Lower. |
| LLM | 026 | CB–11 | Load LM register. |
| LRS | 031 | CB–11 | Load RS register. |
| LSC | 026 | DC–11 | Load Segment Counter. |
| LSF | 032 | CB–11 | Load SF register. |
| NOP | 000 | DC–11, CB–11 | No Operation. |
| OUT | 011 | CC–10 | Go OUT of Service. |
| | | MT–10 | |
| | | PT–10 | |
| | | LP–12 | |
| | | DC–11 | |
| PAB | 052 | CB–11 | Print Alphabetic. |
| | | PC–10 | |
| PAN | 050 | CB–11 | Print Alphanumeric. |
| | | PC–10 | |
| PNU | 051 | CB–11 | Print Numeric. |
| | | PC–10 | |
| POP | 027 | CB–11 | Punch Or Print. |
| PRI | 052 | LP–12 | Print and Interrupt. |
| PRT | 051 | LP–12 | Print. |
| QBT | 064 | MT–10 | Query not at Beginning of Tape. |
| QCO | 063 | LP–12 | Query no Column Overflow. |
| QCT | 063 | PC–10 | Query Complete Transmission. |
| | | CB–11 | |
| QER | 061 | CB–11 | Query no Error. |
| | | LP–12 | |
| | | MT–10 | |
| | | PC–10 | |
| QET | 065 | MT–10 | Query not at End of Tape. |
| QIL | 062 | CB–11 | Query no Interlock. |
| | | CC–10 | |
| | | PC–10 | |
| | | PT–10 | |
| | | LP–12 | |
| QIN | 067 | CB–11 | Query no Interrupt. |
| | | PC–10 | |
| | | PT–10 | |
| | | LP–12 | |
| | | MT–10 | |
| QIP | 065 | CB–11 | Query Index Pulse. |
| | | PC–10 | |
| QLR | 065 | CB–11 | Query not Last Row [cards]. |
| | | PC–10 | |

| ALPHA | OCTAL | APPLICABILITY | NAME |
|---|---|---|---|
| QN1 | 067 | CC–10 | Query no NT1 Interrupt. |
| QN2 | 066 | CC–10 | Query no NT2 Interrupt. |
| QRD | 060 | All Units[2] | Query Ready. |
| QSL | 066 | MT–10 | Query Slewing. |
| QUT | 063 | MT–10 | Query not on Unrecorded Tape. |
| RBH | 020 | CB–11 | Convert Row Binary to Extended Hollerith. |
| RCV | 014 | CB–11<br>MT–10<br>PC–10<br>LP–12 | Receive. |
| RC6 | 015 | DC–11 | Receive 6–bit characters. |
| RC8 | 014 | DC–11 | Receive 8–bit characters. |
| RDC | 025 | CB–11 | Read Card. |
| RED | 003 | All Units | Red. |
| REQ | | All Units[1] | Request for characters. Signal sent on request line. |
| REW | 047 | MT–10 | Rewind. |
| SDT | 010 | All Units[1] | Start Data Transfer. |
| SFL | 057 | CB–11 | Shift SF Left. |
| SFR | 055 | CB–11 | Shift SF Right. |
| SIL | 042 | CB–11 | Start Initial Load. |
| SIN | 055 | PC–10 | Send Index Interrupt. |
| SLM | 022 | CB–11 | Store LM register. |
| SPA | 000 | DC–11 | Space. |
| SRS | 033 | CB–11 | Store RS register. |
| SSF | 030 | CB–11 | Store SF register. |
| STP | 050 | MT–10<br>PT–10 | Stop. |
| STT | 056 | CB–11<br>PT–10<br>PC–10 | Start [paper or card feed]. |
| SW1 | 006 | CB–11<br>MT–10 | Switch to Line 1. |
| SW2 | 007 | CB–11<br>MT–10 | Switch to Line 2. |
| TFM | 056 | LP–12 | Top of Form. |
| TFI | 057 | LP–12 | Top of Form and Interrupt. |
| TRA | 016 | CC–10<br>MT–10<br>PC–10<br>CB–11<br>PT–10 | Transmit. |
| TR6 | 017 | DC–11 | Transmit 6–bit characters. |
| TR8 | 016 | DC–11 | Transmit 8–bit characters. |
| TRC | 044 | CB–11 | Transmit Commands. |
| UNP | 044 | DC–11 | Unpack [use Unpacked mode]. |
| USF | 035 | CB–11 | Unite with SF register. |

Notes: 1. Except the CC–10 2. Except the DC–11.

# APPENDIX IV

## NUMERIC LIST OF COMMUNICATION LINE SIGNALS AND BUFFER CODES

| OCTAL CODE | CC-10 | PT-10 PUNCH | PT-10 READ | MT-10 | LP-12 PRINT | LP-12 FEED | PC-10 PUNCH | PC-10 READ | PC-10 PRINT | CB-11 ON-LINE | CB-11 IN-TERNAL | DC-11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 000 | | | | | | | | | | | NOP | NOP |
| 001 | | | | | | | CLQ | CLQ | CLQ | | | |
| 002 | GRN | GRN | GRN | GRN | GRN | GRN | GRN | GRN | GRN | GRN | | GRN |
| 003 | RED | RED | RED | RED | RED | RED | RED | RED | RED | RED | | RED |
| 004 | | END | END | END | END | | END | END | END | END | | END |
| 005 | | ERR | ERR | ERR | ERR | | ERR | ERR | ERR | ERR | | ERR |
| 006 | | | | SW1 | | | | | | SW1 | SW1 | |
| 007 | | | | SW2 | | | | | | SW2 | SW2 | |
| 010 | | SDT | SDT | SDT | SDT | | SDT | SDT | SDT | SDT | | SDT |
| 011 | OUT | OUT | OUT | OUT | | OUT | | | | | | OUT |
| 012 | | | | | | | | | | | | |
| 013 | | BLK | BLK | | | | | | | | | |
| 014 | | | | RCV | RCV | | RCV | | RCV | RCV | RCV | RC8 |
| 015 | | | | | | | | | | | BTU | RC6 |
| 016 | TRA | | TRA | TRA | | | | TRA | | TRA | TRA | TR8 |
| 017 | | | | | | | | | | | BTL | TR6 |
| 020 | | | | | | | | | | | RBH | |
| 021 | | | | | | | | | | | HRB | |
| 022 | | | | | | | | | | SLM | SLM | |
| 023 | | | | | | | | | | | BCR | |
| 024 | | | | | | | | | | | CPI | |
| 025 | | | | | | | | | | | RDC | |
| 026 | | | | | | | | | | LLM | LLM | LSC |
| 027 | | | | | | | | | | | POP | |
| 030 | | | | | | | | | | | SSF | |
| 031 | | | | | | | | | | | LRS | |
| 032 | | | | | | | | | | | LSF | |
| 033 | | | | | | | | | | | SRS | |
| 034 | | | | | | | | | | CON | | |
| 035 | | | | | | | | | | | USF | |
| 036 | | | | | | | | | | | CSF | |
| 037 | | | | | | | | | | | INC | |
| 040 | | | | | | | | | | | JMP | |
| 041 | | | | | | | | | | | BCS | |
| 042 | | | | | | | | | | SIL | | |
| 043 | | | | | | | | | | | BTF | |

## APPENDIX IV NUMERIC LIST OF COMMUNICATION LINE SIGNALS AND BUFFER CODES (Continued)

| OCTAL CODE | CC–10 | PT–10 PUNCH | PT–10 READ | MT–10 | LP–12 PRINT | LP–12 FEED | PC–10 PUNCH | PC–10 READ | PC–10 PRINT | CB–11 ON–LINE | CB–11 IN-TERNAL | DC–11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 044 | | | FWD | FWD | | | | | | | TRC | UNP |
| 045 | | | BAC | BAC | | | | | | | JEQ | |
| 046 | | | | IWR | | | | | | | | |
| 047 | | | | REW | | | | | | | JLO | DTZ |
| 050 | | STP | STP | STP | CLB | | | | PAN | | PAN | |
| 051 | | | | | PRT | | AS1 | AS1 | PNU | | PNU/AS1 | |
| 052 | | | | | PRI | | AS2 | AS2 | PAB | | PAB/AS2 | |
| 053 | | | | | | | | | | | COF | |
| 054 | | | | | | | CFD | CFD | CFD | | CFD | |
| 055 | | | | | | API | | | SIN | | SFR | |
| 056 | | STT | STT | DBM | | TFM | STT | STT | STT | | STT | |
| 057 | | | | | | TFI | | | | | SFL | |
| 060 | QRD | QRD | QRD | QRD | QRD | QRD | QRD | QRD | QRD | | QRD | |
| 061 | | | | QER | QER | | QER | QER | QER | QER | QER | |
| 062 | QIL | QIL | QIL | | QIL | QIL | QIL | QIL | | | QIL | |
| 063 | | | | QUT | QCO | | | | QCT | | QCT | |
| 064 | | | | QBT | | | | | | | | |
| 065 | | | | QET | | | QLR | QLR | QIP | | QIP/QLR | |
| 066 | QN2 | | | QSL | | | | | | | | |
| 067 | QN1 | QIN | QIN | QIN | QIN | QIN | QIN | QIN | QIN | QIN | | |
| 070 | | | | | | | | | | | | |
| 071 | | | | | | | | | | | | |
| 072 | | | | | | | | | | | | |
| 073 | | | | | | | | | | | | |
| 074 | | | | | | | | | | | | |
| 075 | | | | | | | | | | | | |
| 076 | | | | | | | | | | | | |
| 077 | | | | | | | | | | | | |
| 100 | | | | | | | | | FORMAT | | FORMAT | |
| 1 | | | | | | | | | FORMAT | | FORMAT | |
| 2 | | | | | | | | | FORMAT | | FORMAT | |
| 3 | | | | | | | | | FORMAT | | FORMAT | |
| 4 | | | | | | | | | FORMAT | | FORMAT | |
| 5 | | | | | | | | | FORMAT | | FORMAT | |
| 6 | | | | | | | | | FORMAT | | FORMAT | |
| 7 | | | | | | | | | FORMAT | | FORMAT | |
| 1nn | PROG. LIGHTS | | | SLEW | | FEED | | | | OPERAND | OPERAND | |

# APPENDIX V

# COMMUNICATION LINE PIN ASSIGNMENTS

| PIN | MAIN CABLE | AUXILIARY CABLE |
|---|---|---|
| A | Line Character Bit 0 (+) | Spare |
| B | Line Character Bit 0 (–) | Spare |
| C | Line Character Bit 1 (+) | Spare |
| D | Line Character Bit 1 (–) | Spare |
| E | Line Character Bit 2 (+) | Send Interrupt 4 ( + ) |
| F | Line Character Bit 2 (–) | Send Interrupt 4 ( – ) |
| G | Line Character Bit 3 (+) | Send Interrupt 3 ( + ) |
| H | Line Character Bit 3 (–) | Send Interrupt 3 ( – ) |
| J | Line Character Bit 4 (+) | Power On ( + ) |
| K | Line Character Bit 5 (+) | Receive Interrupt 2 ( + ) |
| L | Line Character Bit 5 (–) | Receive Interrupt 2 ( – ) |
| M | Line Character Bit 6 (+) | Receive Interrupt 1 ( + ) |
| N | Line Character Bit 6 (–) | Receive Interrupt 1 (–) |
| P | Line Character Bit 7 (+) | Spare |
| Q | Line Character Bit 4 (–) | Power Off ( + ) |
| R | Data Flag, Bit 8 (+) | Receive Interrupt 4 ( + ) |
| S | Data Flag, Bit 8 (–) | Receive Interrupt 4 ( – ) |
| T | Character Parity, Bit 9 (+) | Receive Interrupt 3 ( + ) |
| U | Character Parity, Bit 9 (–) | Receive Interrupt 3 (–) |
| V | Line Character Bit 7 (–) | Spare |
| W | REQ Line (+) | Spare |
| X | REQ Line (–) | Spare |
| Y | Spare | Spare |
| Z | Spare | Spare |

# APPENDIX VI

# 47–AND 63–CHARACTER G–20 ALPHABETS

| 63–CHARACTER ALPHABET | CARD CODE | EXTENDED[1] HOLLERITH | G–20[1] CODE | 47–CHARACTER[2] ALPHABET | 47–CHARACTER[3] ALCOM |
|---|---|---|---|---|---|
| 0 | 0 | 700 | 440 | 0 | — |
| 1 | 1 | 403 | 441 | 1 | — |
| 2 | 2 | 420 | 442 | 2 | — |
| 3 | 3 | 440 | 443 | 3 | — |
| 4 | 4 | 460 | 444 | 4 | — |
| 5 | 5 | 404 | 445 | 5 | — |
| 6 | 6 | 410 | 446 | 6 | — |
| 7 | 7 | 414 | 447 | 7 | — |
| 8 | 8 | 401 | 450 | 8 | — |
| 9 | 9 | 402 | 451 | 9 | — |
| A | 12–1 | 503 | 401 | A | — |
| B | 12–2 | 520 | 402 | B | — |
| C | 12–3 | 540 | 403 | C | — |
| D | 12–4 | 560 | 404 | D | — |
| E | 12–5 | 504 | 405 | E | — |
| F | 12–6 | 510 | 406 | F | — |
| G | 12–7 | 514 | 407 | G | — |
| H | 12–8 | 501 | 410 | H | — |
| I | 12–9 | 502 | 411 | I | — |
| J | 11–1 | 603 | 412 | J | — |
| K | 11–2 | 620 | 413 | K | — |
| L | 11–3 | 640 | 414 | L | — |
| M | 11–4 | 660 | 415 | M | — |
| N | 11–5 | 604 | 416 | N | — |
| O | 11–6 | 610 | 417 | O | — |
| P | 11–7 | 614 | 420 | P | — |
| Q | 11–8 | 601 | 421 | Q | — |
| R | 11–9 | 602 | 422 | R | — |
| S | 0–2 | 720 | 423 | S | — |
| T | 0–3 | 740 | 424 | T | — |
| U | 0–4 | 760 | 425 | U | — |
| V | 0–5 | 704 | 426 | V | — |

## APPENDIX VI 47 – AND 63–CHARACTER G–20 ALPHABETS (Continued)

| 63–CHARACTER ALPHABET | CARD CODE | EXTENDED HOLLERITH[1] | G–20 CODE[1] | 47–CHARACTER ALPHABET[2] | 47–CHARACTER ALCOM[3] |
|---|---|---|---|---|---|
| W | 0–6 | 710 | 427 | W | —— |
| X | 0–7 | 714 | 430 | X | —— |
| Y | 0–8 | 701 | 431 | Y | —— |
| Z | 0–9 | 702 | 432 | Z | —— |
| / | 0–1 | 703 | 457 | / | —— |
| \| | 8–2 | 421 | 433 |  | ABS |
| = | 8–3 | 441 | 460 | = | —— |
| ; | 8–4 | 461 | 467 | ; | —— |
| ' | 8–5 | 405 | 477 |  | / |
| ← | 8–6 | 411 | 434 |  | not used |
| ↓ | 8–7 | 415 | 474 |  | not used |
| ↑ | 12–8–2 | 521 | 475 |  | ** |
| . | 12–8–3 | 541 | 453 | . | —— |
| ) | 12–8–4 | 561 | 473 | ) | —— |
| ¬ | 12–8–5 | 505 | 436 |  | NOT |
| ∧ | 12–8–6 | 511 | 463 |  | AND |
| ∨ | 12–8–7 | 515 | 461 |  | OR |
| ≠ | 11–8–2 | 621 | 462 |  | NEQL |
| $ | 11–8–3 | 641 | 465 | $ | —— |
| * | 11–8–4 | 661 | 456 | * | —— |
| < | 11–8–5 | 605 | 464 |  | LESS |
| > | 11–8–6 | 611 | 466 |  | GRTR |
| → | 11–8–7 | 615 | 435 |  | REPLACES |
| : | 0–8–2 | 721 | 476 |  | . . |
| , | 0–8–3 | 741 | 437 | , | —— |
| ( | 0–8–4 | 761 | 470 | ( | —— |
| [ | 0–8–5 | 705 | 471 |  | ( |
| ] | 0–8–6 | 711 | 472 |  | ) |
| $_{10}$ | 0–8–7 | 715 | 452 |  | E |
| + | 12 | 500 | 454 | + | —— |
| – | 11 | 600 | 455 | – | —— |

Notes: 1. The code as shown in this table includes the data flag, bit 8, as the character would appear on the communication line. Inside the computer the data flag is deleted. See Figure 3.4–1.

2. The 47–character alphabet is the same as FORTRAN except for semicolon which replaces one of the FORTRAN minus signs and is punched 8–4.

3. In order to keypunch ALCOM on a FORTRAN keypunch without using multiple punches the missing characters are keypunched as shown in the last column. A dash indicates that the character is available in FORTRAN.

# APPENDIX VII

## EXTENDED HOLLERITH CODE

| EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER | EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER |
|---|---|---|---|---|---|
| 400 | Blank | Space | 441 | 3–8 | = |
| 401 | 8 | 8 | 442 | 3–9 | —— |
| 402 | 9 | 9 | 443 | 3–1 | —— |
| 403 | 1 | 1 | 444 | 3–5 | —— |
| 404 | 5 | 5 | 445 | 3–5–8 | —— |
| 405 | 5–8 | ' | 446 | 3–5–9 | —— |
| 406 | 5–9 | —— | 447 | 3–5–1 | —— |
| 407 | 5–1 | —— | 450 | 3–6 | —— |
| 410 | 6 | 6 | 451 | 3–6–8 | —— |
| 411 | 6–8 | ← | 452 | 3–6–9 | —— |
| 412 | 6–9 | —— | 453 | 3–6–1 | —— |
| 413 | 6–1 | —— | 454 | 3–7 | —— |
| 414 | 7 | 7 | 455 | 3–7–8 | —— |
| 415 | 7–8 | ↓ | 456 | 3–7–9 | —— |
| 416 | 7–9 | —— | 457 | 3–7–1 | —— |
| 417 | 7–1 | —— | 460 | 4 | 4 |
| 420 | 2 | 2 | 461 | 4–8 | ; |
| 421 | 2–8 | \| | 462 | 4–9 | —— |
| 422 | 2–9 | —— | 463 | 4–1 | —— |
| 423 | 2–1 | —— | 464 | 4–5 | —— |
| 424 | 2–5 | —— | 465 | 4–5–8 | —— |
| 425 | 2–5–8 | —— | 466 | 4–5–9 | —— |
| 426 | 2–5–9 | —— | 467 | 4–5–1 | —— |
| 427 | 2–5–1 | —— | 470 | 4–6 | —— |
| 430 | 2–6 | —— | 471 | 4–6–8 | —— |
| 431 | 2–6–8 | —— | 472 | 4–6–9 | —— |
| 432 | 2–6–9 | —— | 473 | 4–6–1 | —— |
| 433 | 2–6–1 | —— | 474 | 4–7 | —— |
| 434 | 2–7 | —— | 475 | 4–7–8 | —— |
| 435 | 2–7–8 | —— | 476 | 4–7–9 | —— |
| 436 | 2–7–9 | —— | 477 | 4–7–1 | —— |
| 437 | 2–7–1 | —— | 500 | 12 | + |
| 440 | 3 | 3 | 501 | 12–8 | H |

## APPENDIX VII EXTENDED HOLLERITH CODE (Continued)

| EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER | EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER |
|---|---|---|---|---|---|
| 502 | 12–9 | I | 552 | 12–3–6–9 | —— |
| 503 | 12–1 | A | 553 | 12–3–6–1 | —— |
| 504 | 12–5 | E | 554 | 12–3–7 | C.R. (CG) |
| 505 | 12–5–8 | ¬ | 555 | 12–3–7–8 | —— |
| 506 | 12–5–9 | e (EI) | 556 | 12–3–7–9 | —— |
| 507 | 12–5–1 | a (AE) | 557 | 12–3–7–1 | —— |
| 510 | 12–6 | F | 560 | 12–4 | D |
| 511 | 12–6–8 | ∧ | 561 | 12–4–8 | ) |
| 512 | 12–6–9 | —— | 562 | 12–4–9 | i (ID) |
| 513 | 12–6–1 | f (FA) | 563 | 12–4–1 | —— |
| 514 | 12–7 | G | 564 | 12–4–5 | d (DE) |
| 515 | 12–7–8 | ∨ | 565 | 12–4–5–8 | —— |
| 516 | 12–7–9 | h (GI) | 566 | 12–4–5–9 | —— |
| 517 | 12–7–1 | g (GA) | 567 | 12–4–5–1 | —— |
| 520 | 12–2 | B | 570 | 12–4–6 | —— |
| 521 | 12–2–8 | ↑ | 571 | 12–4–6–8 | —— |
| 522 | 12–2–9 | —— | 572 | 12–4–6–9 | —— |
| 523 | 12–2–1 | —— | 573 | 12–4–6–1 | —— |
| 524 | 12–2–5 | b (BE) | 574 | 12–4–7 | —— |
| 525 | 12–2–5–8 | —— | 575 | 12–4–7–8 | —— |
| 526 | 12–2–5–9 | —— | 576 | 12–4–7–9 | —— |
| 527 | 12–2–5–1 | —— | 577 | 12–4–7–1 | —— |
| 530 | 12–2–6 | —— | 600 | 11 | – |
| 531 | 12–2–6–8 | —— | 601 | 11–8 | Q |
| 532 | 12–2–6–9 | —— | 602 | 11–9 | R |
| 533 | 12–2–6–1 | —— | 603 | 11–1 | J |
| 534 | 12–2–7 | —— | 604 | 11–5 | N |
| 535 | 12–2–7–8 | —— | 605 | 11–5–8 | < |
| 536 | 12–2–7–9 | —— | 606 | 11–5–9 | n (NR) |
| 537 | 12–2–7–1 | —— | 607 | 11–5–1 | —— |
| 540 | 12–3 | C | 610 | 11–6 | O |
| 541 | 12–3–8 | . | 611 | 11–6–8 | > |
| 542 | 12–3–9 | —— | 612 | 11–6–9 | o (OR) |
| 543 | 12–3–1 | —— | 613 | 11–6–1 | j (JO) |
| 544 | 12–3–5 | c (CE) | 614 | 11–7 | P |
| 545 | 12–3–5–8 | —— | 615 | 11–7–8 | → |
| 546 | 12–3–5–9 | —— | 616 | 11–7–9 | q (PR) |
| 547 | 12–3–5–1 | —— | 617 | 11–7–1 | —— |
| 550 | 12–3–6 | —— | 620 | 11–2 | K |
| 551 | 12–3–6–8 | —— | 621 | 11–2–8 | ≠ |

## APPENDIX VII EXTENDED HOLLERITH CODE (Continued)

| EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER | EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER |
|---|---|---|---|---|---|
| 622 | 11–2–9 | —— | 672 | 11–4–6–9 | —— |
| 623 | 11–2–1 | —— | 673 | 11–4–6–1 | —— |
| 624 | 11–2–5 | —— | 674 | 11–4–7 | p (PM) |
| 625 | 11–2–5–8 | —— | 675 | 11–4–7–8 | —— |
| 626 | 11–2–5–9 | —— | 676 | 11–4–7–9 | —— |
| 627 | 11–2–5–1 | —— | 677 | 11–4–7–1 | —— |
| 630 | 11–2–6 | k (KO) | 700 | 0 | 0 |
| 631 | 11–2–6–8 | —— | 701 | 0–8 | Y |
| 632 | 11–2–6–9 | —— | 702 | 0–9 | Z |
| 633 | 11–2–6–1 | —— | 703 | 0–1 | / |
| 634 | 11–2–7 | —— | 704 | 0–5 | V |
| 635 | 11–2–7–8 | —— | 705 | 0–5–8 | [ |
| 636 | 11–2–7–9 | —— | 706 | 0–5–9 | —— |
| 637 | 11–2–7–1 | —— | 707 | 0–5–1 | —— |
| 640 | 11–3 | L | 710 | 0–6 | W |
| 641 | 11–3–8 | $ | 711 | 0–6–8 | ] |
| 642 | 11–3–9 | —— | 712 | 0–6–9 | —— |
| 643 | 11–3–1 | —— | 713 | 0–6–1 | —— |
| 644 | 11–3–5 | —— | 714 | 0–7 | X |
| 645 | 11–3–5–8 | —— | 715 | 0–7–8 | $_{10}$ |
| 646 | 11–3–5–9 | —— | 716 | 0–7–9 | y (XZ) |
| 647 | 11–3–5–1 | —— | 717 | 0–7–1 | —— |
| 650 | 11–3–6 | l (LO) | 720 | 0–2 | S |
| 651 | 11–3–6–8 | —— | 721 | 0–2–8 | : |
| 652 | 11–3–6–9 | —— | 722 | 0–2–9 | —— |
| 653 | 11–3–6–1 | —— | 723 | 0–2–1 | —— |
| 654 | 11–3–7 | —— | 724 | 0–2–5 | v (VS) |
| 655 | 11–3–7–8 | —— | 725 | 0–2–5–8 | —— |
| 656 | 11–3–7–9 | —— | 726 | 0–2–5–9 | —— |
| 657 | 11–3–7–1 | —— | 727 | 0–2–5–1 | —— |
| 660 | 11–4 | M | 730 | 0–2–6 | —— |
| 661 | 11–4–8 | * | 731 | 0–2–6–8 | —— |
| 662 | 11–4–9 | r (RM) | 732 | 0–2–6–9 | —— |
| 663 | 11–4–1 | —— | 733 | 0–2–6–1 | —— |
| 664 | 11–4–5 | —— | 734 | 0–2–7 | s (SX) |
| 665 | 11–4–5–8 | —— | 735 | 0–2–7–8 | —— |
| 666 | 11–4–5–9 | —— | 736 | 0–2–7–9 | —— |
| 667 | 11–4–5–1 | —— | 737 | 0–2–7–1 | —— |
| 670 | 11–4–6 | m (MO) | 740 | 0–3 | T |
| 671 | 11–4–6–8 | —— | 741 | 0–3–8 | , |

## APPENDIX VII EXTENDED HOLLERITH CODE (Continued)

| EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER | EXTENDED[1] HOLLERITH | CARD CODE | BENDIX[2] CHARACTER |
|---|---|---|---|---|---|
| 742 | 0–3–9 | tab (TZ) | 761 | 0–4–8 | ( |
| 743 | 0–3–1 | —— | 762 | 0–4–9 | z (ZU) |
| 744 | 0–3–5 | t (TV) | 763 | 0–4–1 | —— |
| 745 | 0–3–5–8 | —— | 764 | 0–4–5 | u (UV) |
| 746 | 0–3–5–9 | —— | 765 | 0–4–5–8 | —— |
| 747 | 0–3–5–1 | —— | 766 | 0–4–5–9 | —— |
| 750 | 0–3–6 | —— | 767 | 0–4–5–1 | —— |
| 751 | 0–3–6–8 | —— | 770 | 0–4–6 | w (WU) |
| 752 | 0 –3–6–9 | —— | 771 | 0–4–6–8 | —— |
| 753 | 0–3–6–1 | —— | 772 | 0–4–6–9 | —— |
| 754 | 0–3–7 | —— | 773 | 0–4–6–1 | —— |
| 755 | 0–3–7–8 | —— | 774 | 0–4–7 | x (XU) |
| 756 | 0–3–7–9 | —— | 775 | 0–4–7–8 | —— |
| 757 | 0–3–7–1 | —— | 776 | 0–4–7–9 | —— |
| 760 | 0–4 | U | 777 | 0–4–7–1 | —— |

Notes: 1. Extended Hollerith divides a card column into 4 zones:

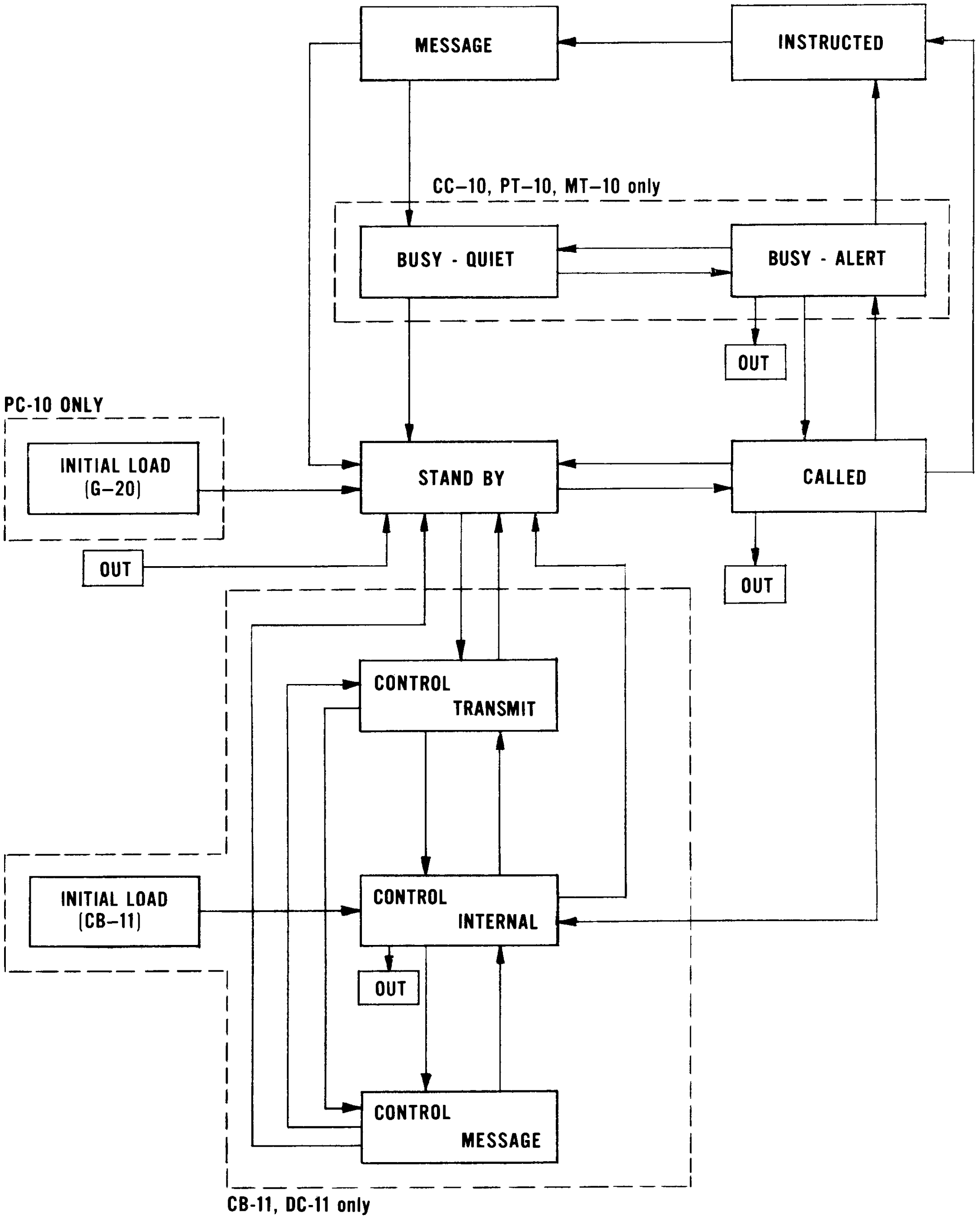
| | |
|---|---|
| 1st zone | 12, 11, 0 |
| 2nd zone | 2, 3, 4 |
| 3rd zone | 5, 6, 7 |
| 4th zone | 8, 9, 1 |

Each zone produces 2 bits of the 8–bit code representing the character. No more than one punch is permitted in a zone. The first zone determines the first two bits of the octal code, etc.

| ZONE PUNCH | CODE |
|---|---|
| None | 00 |
| First hole | 01 |
| Second hole | 10 |
| Third hole | 11 |

2. The suggested set of lower case characters can be double punched as shown in parentheses. The double punchings are mnemonic except for h, which is punched with the preceding and following letters GI, q, which is punched as PR, and y, which is punched as XZ.

# APPENDIX VIII G–20 ACCESSORY OPERATING STATES (COMPLETE)



# INDEX

**Bendix Computer Division**
LOS ANGELES 45, CALIFORNIA

THE Bendix
CORPORATION

May 1962

BER- 05624