

## CONTENTS

	Page
GENERAL ASSEMBLY PROGRAM	
Program Coding Sheet . . . . .	A-1
Symbol (Columns 1-6) . . . . .	A-2
Operation (Columns 8-10) . . . . .	A-3
Operand (Columns 12-19) . . . . .	A-3
X (Column 20) . . . . .	A-4
Remarks (Columns 31-75) . . . . .	A-4
Sequence (Columns 76-80) . . . . .	A-4
Relative Addressing . . . . .	A-4
Pseudo-Instructions . . . . .	A-5
Special Symbols . . . . .	A-11
Detected Coding Errors . . . . .	A-11
Assembly Stops . . . . .	A-13
Assembly Operations . . . . .	A-13
A-Register Input Switches . . . . .	A-14
Switch Combinations and Requirements . . . . .	A-16
Card-to-Card Operations . . . . .	A-16
Pass 0 . . . . .	A-16
Output From Pass 0 . . . . .	A-18
Pass 1 . . . . .	A-18
Pass 2 . . . . .	A-19
Card Operations with 4k Memory . . . . .	A-21
Systems Tape . . . . .	A-21
Paper Tape Conversion, Program 3 . . . . .	A-22
Magnetic Tape Updating, Program 4 . . . . .	A-24

## ILLUSTRATIONS

Figure		Page
A-1	Sample General Assembly Program Coding Sheet . . . . .	A-2
A-2	Diagram of General Assembly Program . . . . .	A-14
A-3	Arrangement of Input for Pass 2 . . . . .	A-20
A-4	Flow Chart for Assembly Program 3 . . . . .	A-23



## APPENDIX A

### DATANET-30

### GENERAL ASSEMBLY PROGRAM

The DATANET-30 General Assembly Program (CD225F2.001/2) is a modified version of the GE-225 General Assembly Program II. With this program, DATANET-30 source programs are assembled on a GE-200 Series computer system. The program permits writing DATANET-30 programs in symbolic codes (mnemonics) rather than in absolute computer coding (machine language). The mnemonic of each instruction is as self-explanatory as possible. The assembler examines each mnemonic and translates it into its corresponding machine language code. The output of the assembler is an object program (the original source program converted to absolute code) in machine-readable form on punch cards, magnetic tape, or perforated tape.

The minimum GE-200 system configurations required for the various assembly operations are given on pages A-14, A-22, and A-25.

In addition to the mnemonic code for the communication processor's instructions, the assembly program uses other mnemonic codes called pseudo-instructions. A pseudo-instruction is not executed by the communication processor but is used by the assembler to generate constants and control the assembly program in the preparation of object programs. For example, ORG is a pseudo-instruction used to instruct the assembler that the starting address for a program is to begin at the location indicated in the operand. ORG 256 tells the assembler that the first instruction of the program is to start at memory location 256 (decimal). The assembler automatically assigns the first instruction to location 400 (octal) and succeeding memory locations to the instructions which follow. The programmer need only specify the starting address where the first instruction of the program is to be stored.

The assembly program provides the following advantages in addition to translating the mnemonics into machine language:

1. Various clerical errors are detected during program assembly. This effects a substantial savings in program debugging effort, because the error can be rectified prior to debugging.
2. The assembler generates punched cards and/or a listing on the high-speed printer that include all error indications, the assembled object program, and a complete list of symbols used, with their assigned memory locations. This provides an accurate record of the program plus helpful auxiliary information.

#### PROGRAM CODING SHEET

The coding sheet CD 34 (shown in Figure A-1) is divided into six fields: Symbol, Operation, Operand, X, Remarks, and Sequence. The numbers 1-80 in the heading correspond to the column numbers of a standard 80-column punch card.

Figure A-1. Sample General Assembly Program Coding Sheet

The purpose of each field is described in the following:

**Symbol (Columns 1-6)**

The Symbol field allows symbols to be assigned to memory locations. The following rules apply to the use of symbols:

1. Symbols may consist of one to six characters in length and be any combination of alphabetic or numeric.
  2. One character of a symbol must be alphabetic. HOPE and CONST3 are legitimate symbols, 345 is not a legitimate symbol.

3. Special characters other than plus (+), minus (-), or asterisk (\*) can be used in symbols. Plus or minus cannot be used because they are used for relative addressing. An asterisk cannot be used because it serves a special purpose in the assembler.
4. A blank (space) is ignored in the Symbol field. Symbols may start at any point within the field because leading and inserted blanks are ignored by the assembler.
5. A symbol may appear in this field only once in the program.

An entry in the Symbol field, along with its associated information, is assigned a specific memory location by the assembler. Thus, the programmer need not know the actual memory address but can refer to the symbolic address when needed.

### Operation (Columns 8-10)

Any of the three-character mnemonic codes for the communication processor instructions (LDC, BRU, SR6, etc.) or for the pseudo-instructions (ORG, INC, etc.) are placed in the Operation field. An invalid mnemonic will be flagged on the object program listing as a Conditional Halt (HLT).

### Operand (Columns 12-19)

Operands cannot exceed six characters in length. The following rules apply to operands:

1. They must be alphabetic or alphanumeric symbols, or numeric constants.
2. A symbol appearing in this field must also appear in the Symbol field somewhere in the program.
3. Operands can be positioned to start anywhere in the Operand field, as long as they do not exceed six characters in length or extend beyond column 19.
4. A single asterisk placed in the Operand field denotes reference to self. This is equivalent to writing the same symbolic name in both the Symbol and Operand fields.
5. Operands can be arithmetic combinations not exceeding eight characters of sums and differences of numbers, symbols, and asterisks.
6. Plus (+) and minus (-) signs are used to express a relative address or an assigned constant. See "Relative Addressing," page A-4.
7. All numbers appearing in the Operand field are considered to be decimal except when following the OCT, ALF, NAL, LOC, and EQO pseudo-instructions. These are interpreted as follows:
  - a. Numbers following OCT, LOC, and EQO are assumed to be octal and are converted to their binary equivalent by the assembler.
  - b. Digits following ALF and NAL are converted to their binary-coded-decimal (BCD) equivalents by the assembler.
8. Blanks (spaces) in the Operand field are ignored, unless they follow pseudo-instructions ALF or NAL.

### X (Column 20)

The X-column is used for address modification. If an X appears in column 20, the assembler inserts the indirect address bit (bit 16 and/or 17) into the assembled instruction word. If a character other than an X or blank appears, an error will be flagged on the listing of the object program. A blank in column 20 indicates that address modification is not to be performed.

### Remarks (Columns 31-75)

Remarks are written in this field. Remarks are punched in the source program cards; however, the assembler does not assign them memory locations within the assembled object program, nor do they affect the assembly process. The information in the Remarks field is obtained only on the printed listing of the object program. This field should be used extensively for adequate documentation of the program.

### Sequence (Columns 76-80)

Each card of the source program deck should be numbered so a deck can be sorted into proper order should the cards get out of sequence.

## RELATIVE ADDRESSING

The assembler provides the facility for assigning addresses relative to a starting point or symbolic memory location. Plus (+), minus (-), and asterisk (\*) characters are used to indicate the address to which the base (symbol) address is to be added to find the machine (memory) address. An example follows:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00400	000400	B	EQU	256
{	{		LDA	B
00443	400400		{	
{	400407		LDA	B+7
00523	400520		{	{
			LDA	*-3

The EQU pseudo-instruction equates the symbol B to memory location 00400; the LDA (Load-A-register) loads the A-register with the contents of memory location 00400. The next LDA, some program steps later, loads the A-register with the contents of location 000400 + 7, or location 407.

The LDA instruction at memory location 00523, loads the A-register with the contents of location 400520. The asterisk indicates reference to self.

## PSEUDO-INSTRUCTIONS

Pseudo-instructions are not executed by the processor but are used by the assembly program to generate constants, control the assembler, and provide information on the program listing. The pseudo-instructions used by the DATANET-30 assembler follow in alphabetical order:

### ALPHANUMERIC

#### ALF

This instruction generates a constant of the first three characters in the Operand field to binary-coded-decimal characters and assigns a memory location. Blanks are considered characters, and octal 60 is inserted in the word. Only columns 12, 13, and 14 of the Operand field can contain the data for the instruction.

Example: To assign PLANT CODE, the program would read:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
01202	474321	TYPE	ALF	PLA
01203	456360		ALF	NT
01204	234724		ALF	COD
01205	236060		ALF	E

Example: SET SW 18 was to be assigned, and the program reads as follows:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
01202	622563		ALF	SET
01203	626660		ALF	SW
01204	011060		ALF	18

The assigned information would read SETSW 18. The words at location 01203 and 01204 should read 606266 and 600110 for proper spacing.

### BLOCK STARTED BY SYMBOL

#### BSS

The memory allocation register (MAR) in the assembler is increased by the number specified in the Operand field. This instruction is used to reserve a block of memory locations. The operand may be decimal or symbolic. If decimal, the number is converted to binary. If symbolic, the symbol must be predefined. A BSS instruction may be used as often as desired.

Example:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
	000500		ORG	320
	000034	CRD	BSS	28
	000012	SAVE	EQU	10
	000535	STORE	BSS	SAVE
	000003	INDEX	BSS	3
00551	400510		LDA	CRD+8

DECIMAL

DEC

Decimal constants are converted to binary and entered into the memory location assigned by the assembler. The operand may be symbolic providing at least one character is other than 0-9, plus (+), minus (-), decimal point (.), B, or E. Leading zeros are ignored, and the number is right-justified by the assembler. If no sign is present, the number is assumed to be positive. A minus (-) sign specifies a negative number, resulting in the 2's complement of the number being placed in memory. Examples follow.

Example: Positive and negative numbers.

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00102	000005		DEC	5
00103	000200		DEC	128
00104	777773		DEC	-5
00105	777600		DEC	-128

Example: Scaling operands with . and E.

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00120	000005		DEC	5.5
00121	001046		DEC	5.5E2
00122	010460		DEC	5.5B14E2

Example: Scaling operands with character B.

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00120	000050		DEC	5B14
00121	777730		DEC	-5B14

The character B is used to specify a binary scale for either positive or negative numbers. The number following the B is used to position the binary point for the decimal constant preceding the B in the Operand field. If no scale is specified, the assembler assumes a binary scale of 17.

## DOUBLE DECIMAL

### DDC

Double-length decimal constants are converted to binary and entered into two consecutive memory locations. The first even-numbered location is assigned to the most significant half of the constant, and the least significant half of the constant is stored in the sequential odd-numbered location. DDC is normally used to store decimal constants larger than the value 131,071. If no scaling is specified, the assembler assumes a binary scale of 34.

#### Example:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00140	000000		DDC	12
00141	000014			
00142	000001		DDC	132000
00143	001640			
00144	050000		DDC	24B5
00145	000000			

## END OF PROGRAM

### END

An END instruction indicates the end-of-program and terminates assembly. It must be used only once and must be the last instruction of the source program. When the assembler encounters an END instruction, a transfer card is generated to transfer control to the location specified in the Operand field. When the object program is loaded into memory for execution, the transfer card transfers control to the operand location to start execution.

## EQUALS

### EQU

This instruction equates a new symbol to some symbolic memory location already known to the assembler or equates a new symbol to a specific decimal memory location. This instruction does not affect the memory allocation register of the assembler; thus it may be used as often as necessary and at any point within the source program without disturbing the memory assignment sequence.

#### Example:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
000400	CRD	EQU		256
000400	AREA	EQU		CRD
000450	AREA2	EQU		CRD+40

### EQUALS OCTAL

#### EQO

This instruction is the same as the EQU except that the Operand field must be in octal. Leading zeros are ignored.

### INDEX BY A-REGISTER

#### INA

The INA pseudo-instruction generates a word which has bit 16 set and the machine address in bits 1-14. When the INA is addressed indirectly, the effective address will be the address in bits 1-14 of the INA plus the contents of the A-register.

#### Example:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>	<u>X</u>
02000	000400	CRD	EQU	256	
	704500		STB	STORE	X
{ 02500	{ 100400	STORE	INA	{ CRD	

The A-register contains 000020, the contents of the B-register will be stored in memory location 00420, common data bank.

### INDEX BY B-REGISTER

#### INB

Index By B-register is the same as INA except that the B-register is used instead of the A-register, and the assembler sets bit 17 in the instruction word.

### INDEX BY C-REGISTER

#### INC

Index By C-register is the same as INA except that the C-register is used instead of the A-register, and the assembler sets both bits 16 and 17 in the instruction word.

## INDIRECT ADDRESS

IND

The Indirect Address instruction generates the absolute memory address of the operand. The operand may be symbolic, or numeric. If symbolic, the address assigned to the symbol is used. If numeric, it is assumed to be a decimal number and is converted to binary, denoting a memory address.

## LOCATION IN OCTAL

LOC

The Location In Octal instruction resets the memory allocation register (MAR) of the assembler to the value of the operand and assigns the instructions which follow to the sequential addresses starting with the location shown in the operand. An LOC can be used as often as desired. The address in the operand is assumed by the assembler to be in octal.

Example: In a program sequence, a subroutine is to begin at location 2000 (octal).

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00321	000001	RPT	DEC	1
	002000		LOC	2000
02000	400120	TEST	LDA	RESULT

## NEGATIVE ALPHANUMERIC

NAL

The Negative Alphanumeric instruction is used to enter a 2's complement of an alphanumeric constant in the assigned memory location. It is useful for a comparison operation by adding. Only columns 12, 13, and 14 may contain data.

Example:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00742	567674	CODE	NAL	A14
00743	565576		NAL	AB2
00744	565552		NAL	ABF

---

**OCTAL**

---

**OCT**

---

An Octal constant in the Operand field is converted to binary, right-justified, and assigned one memory location determined by the memory allocation register (MAR) of the assembler. Leading zeros are ignored. Leading plus (+) or minus (-) signs will set the leading bit of the constant to 0 or 1.

**Example:**

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
03402	000377		OCT	377
03403	400377		OCT	-377
03404	000077		OCT	0077

---

**ORIGIN**

---

---

**ORG**

---

The Origin instruction performs the same function as an LOC with the exception that the Operand field must be a decimal number or a symbol. If the operand is a symbol, the symbol must be predefined. If no ORG or LOC is included in the assembly, the assembler automatically begins assembly at location 00000.

Any number of ORG instructions may be used in one assembly; however, each time it is encountered, MAR is reset to the operand value.

---

**REMARKS**

---

---

**REM**

---

The REM instruction is used to provide additional information on the object program listing. A memory location is not assigned to an REM instruction. The information in the card, columns 1-80, is printed on the program listing. This REM instruction can be used instead of an asterisk (\*) in the Symbol field. See "Special Symbols," page A-11.

---

**TRANSFER CARD**

---

---

**TCD**

---

The Transfer Card instruction generates a transfer of control to the location specified in the Operand field when the object program is being loaded. The operand may be decimal or symbolic. If decimal, the address is converted to binary. If symbolic, the symbol used must be predefined. TCD may be used as often as desired in the source program. This instruction has no affect on the memory allocation register (MAR), so that the memory assignment by the assembler will continue in sequence.

## Z (Octal Operation Code)

### Z (XX)

The Z pseudo-instruction is used to set the operation bits of the assembled instruction to any desired configuration. Z is followed by two octal digits. These digits become the operation code portion of the generated instruction word. The operand can be decimal, symbolic, or a combination of both.

Example: LDA operation with location at TEMP.

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
00120	400074		Z40	TEMP

## **SPECIAL SYMBOLS**

Special symbols are provided by the assembler as follows:

- \* An asterisk (\*) in column 1 of the Symbol field tells the assembler that the information appearing on the source card (columns 2-80) are remarks. The information on this card will be printed on the listing of the object program and not affect memory allocation.
- \*#\* # equals Hollerith multiple code 12, 7, and 8. This symbol in columns 1, 2, and 3 of the Symbol field causes the printer to slew to the top of the next page and the information appearing on the card, columns 4-80, will be printed as remarks on the listing of the object program.
- \$ A dollar sign (\$) in the leading position of the Symbol field tells the assembler that this symbol is to have a channel table address referenced by the instruction. The assembler automatically inserts address mode 3 (bits 10 and 11).

Example:

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
		LOC	1160	
001160	\$ERROR	BSS	16	
		LDA		\$ERROR
02000	403160			

## **DETECTED CODING ERRORS**

Errors or suspected errors found during assembly are listed with the following codes. The objective of the error codes is to convey as much error information as possible to the programmer.

Following is a list of the codes which will appear on the printed listing of the object program:

A Error or Suspected Error in the Operand Field:

1. An instruction normally requiring an address has a blank Operand field.
2. An operand entry appears in a line which normally should be blank.
3. The numeric value of the operand does not meet the requirement of the line in which it was used. The value of the operand address will be logically ORed into the instruction.

M Multidefined Symbol

When the same symbol is used to define more than one instruction, it is flagged as an error and the assembler performs the following, dependent upon the sequential appearance of the symbol:

1. If a symbol is defined by more than one instruction, the assembler flags the first and each succeeding line in which the symbol appears, and assigns a new address in its sequential order of appearance.

Example: Symbol HOPE is defined at two locations in the source program.

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
M 01526	400320	HOPE	DLD	SUBR
M 01616	400024	HOPE	LDA	20

2. If a symbol appears in the Operand field of an instruction and it has been flagged as a multidefined symbol, the assembler assigns the instruction to the address as it appeared the last time it was defined.

Example: When HOPE is later used and it is the intent to branch to the location first defined, the assembler will assign the address of the second defined symbol.

<u>Memory Address</u>	<u>Instruction</u>	<u>Symbol</u>	<u>Opr</u>	<u>Operand</u>
M 01640	101616		BRU	HOPE

The program will branch to location 1616 and not to 1526 as desired.

O Illegal Mnemonic in the Opr Field

When an illegal mnemonic is detected in the Operation field (columns 8-10), a Conditional Halt (HLT) will be generated. On the DATANET-30, an HLT instruction is executed, dependent upon the setting of the INHIBIT HALT switch.

S Scale Factors in DEC

This denotes that the specified binary and decimal scales are incompatible. Two decimal or binary scales have been specified in the pseudo-instruction generating constants.

T Error or Suspected Error in the X Field

The X field contains an entry for an instruction which does not access memory, or the X field contains any character other than X or a blank.

U Undefined Symbol

The symbolic name appearing in the Operand field has not appeared in the Symbol field of any instruction. A constant 0000 is inserted as an operand address by the assembler.

\$ Channel Table

The dollar sign (\$) character in the first position of a symbol indicates to the assembler that this instruction is to be treated as a channel table address. When this symbol is used, the programmer must indicate a memory address that is a multiple of  $16_{10}$ . The dollar sign (\$) error symbol indicates that either the specified address was not modulo 16 or not less than 8192 or both.

## ASSEMBLY STOPS

During assembly, the GE-225 central processor will stop under the following two conditions:

1. The number of special symbolic operands exceeds the size of the symbol table (symbol table overflow) allowed by the assembler.
2. During the final phase (pass 3) of assembly, a name appearing in the Symbol field cannot be found in the symbol table (lost symbol).

When these errors occur, a typeout can result, and the central processor will go into a programmed loop. The assembly can be forced to continue by manually setting switch 19, and the following will result:

1. The special symbolic operands encountered after the error halt are not entered in symbol table 1. This may result in the improper assignment of a memory address to the symbols in the following passes.
2. The symbols following the error halt are not entered into symbol table 2. This will result in the detection of undefined symbols during the final pass.
3. Assembly will continue. If the symbolic name was a special operand, the assignment of memory locations to the instructions following the error halt may be out of phase with the numeric assignment performed by the previous pass of the assembly.

## ASSEMBLY OPERATIONS

The DATANET-30 General Assembly Program, CD225F1.001/2, has operating procedures identical to those for the GE-225 program, CD225F1.006/7. Operating instructions depend upon whether a magnetic tape subsystem is available for assembly. The assembler operates from either cards or a systems tape in three passes. Each pass is a separate program. A flow diagram of the three passes is shown in Figure A-2.

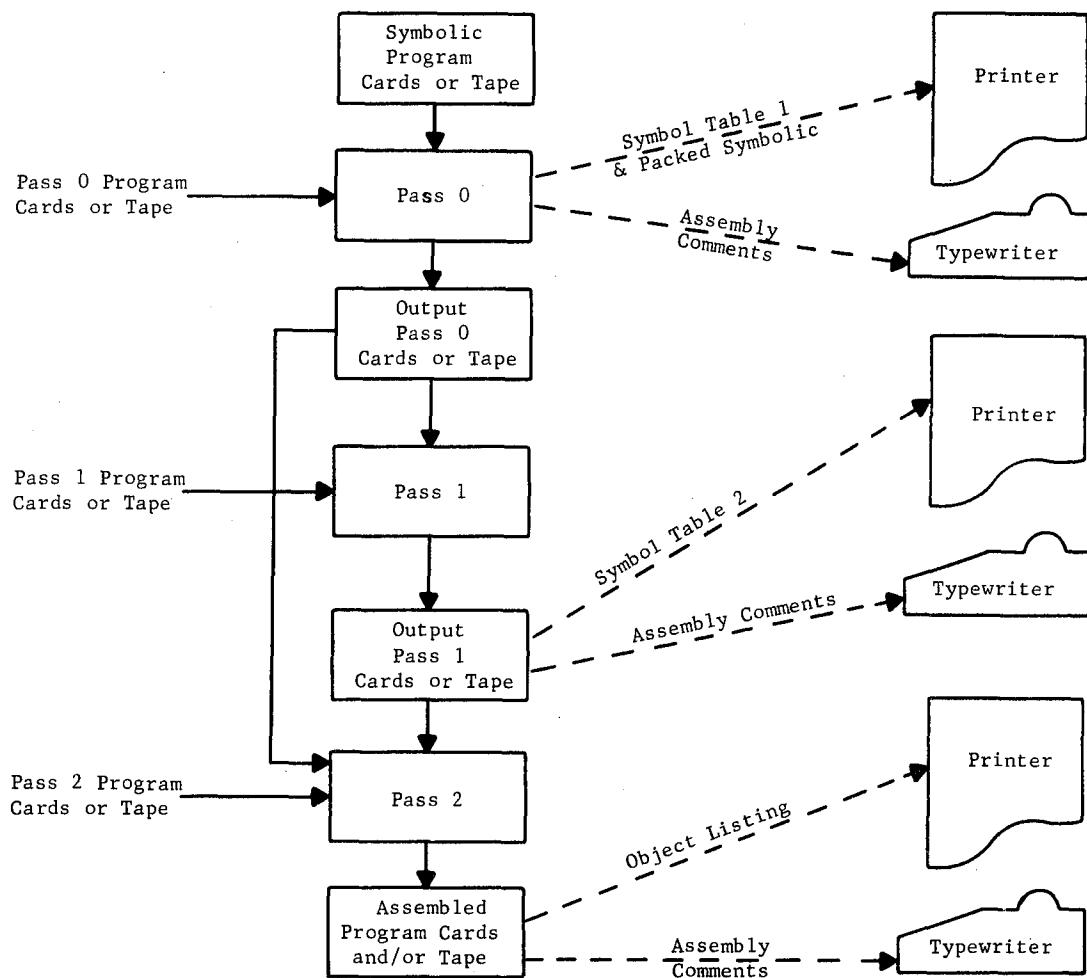


Figure A-2. Diagram of General Assembly Program

The minimum hardware requirements for the operation of General Assembly Program, CD225F1.001/2 are:

1. GE-200 Series central processor with 8k memory
2. Card reader subsystem
3. Card punch subsystem or magnetic tape controller with 4 tape units
4. Typewriter

#### A-Register Input Switches

The A-register input switches of the GE-225 are used to indicate the peripheral configuration available while using the assembly program. All switches, with the exception of switch 19, should be set initially and remain the same through all passes of the program.

<u>Switch 1</u>	Normal:	Output is absolute.
<u>Switch 2</u>	Normal:	Printer is on line.
	Down:	No printer is on line. An octal program deck is punched instead of a binary program deck.
<u>Switch 3</u>	Normal:	Tape 3 is used to <u>print comments</u> by pass 2.
	Down:	Comments are omitted from pass 2. (If switch 4 is down, switch 3 is ignored.)
<u>Switch 4</u>	Normal:	Tapes 4 and 5 are used as output/input to assembly program passes 0, 1, and 2, respectively.
	Down:	Cards instead of tapes are used as output/input to passes 0, 1, and 2, respectively. (Switch 4 down overrides switches 3 and 6.)
<u>Switch 5</u>		Not used.
<u>Switch 6</u>	Normal:	The binary program output is not written on tape 6.
	Down:	The binary program output from pass 2 is written on tape 6.
<u>Switch 7</u>	Normal:	This setting is ignored.
	Down:	Go to assembly 3 upon completion of general assembly.
<u>Switch 8</u>		Not used.
<u>Switch 9</u>	Normal:	Card punch is on line.
	Down:	No card punch is on line.
<u>Switches 10-13</u>		Not used.
<u>Switch 14</u>	Normal:	There is no packed symbolic listing.
	Down:	There is a packed symbolic listing.
<u>Switch 15</u>	Normal:	This setting is ignored by the assembly program.
	Down:	Symbolic program deck is written on tape 3 before any processing is done by the assembly program.
<u>Switch 16</u>	Normal:	Input to pass 0 is the symbolic program card deck.
	Down:	Input to pass 0 is on tape 3. Tape 3, the comments tape, may be changed instead of the symbolic card deck through an updating routine before making a second assembly.
<u>Switch 17</u>		Not used.
<u>Switch 18</u>	Normal:	Types or prints "no reference symbols" after pass 0.
	Down:	Suppresses the typing or printing of "no reference symbols" after pass 0.
<u>Switch 19</u>		Toggling of switch 19 bypasses "symbol table overflow" stop during passes 0 and 1, and "symbol lost" stop during pass 2.

## Switch Combinations and Requirements

The table below shows different switch combinations and their requirements.

Programs	Switch Down	Comment Tape #3	Working Tape #4 & #5	Binary Program Tape #6	Punch On-Line	Printer On-Line
<u>CARD</u> General Assembly Program	2 & 4 4	No No	No No	No No	Yes Yes	No Yes
<u>TAPE</u> General Assembly Program	None 2 3 6 6 & 9* 3 & 6 3,6 & 9*	Yes Yes No Yes Yes No Yes	Yes Yes Yes Yes Yes Yes	No No No Yes Yes Yes	Yes Yes Yes Yes No Yes	Yes No Yes Yes Yes Yes

\* When switch 9 is used, switch 6 also must be set, because the punch is off line and the output must be written on tape.

When switch 16 is used, and switch 3 is in normal position, the input to pass 9, the original symbolic deck, is read from tape 3 instead of cards.

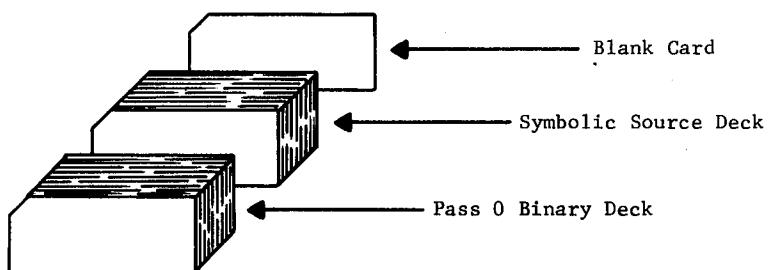
The other switches pertain to format and may be used at the discretion of the programmer, providing the hardware is available.

## CARD-TO-CARD OPERATIONS

When the assembly program is loaded and is to operate from source program cards, it operates under the following procedures:

### Pass 0

1. Place the source program deck followed by one blank card behind the binary pass 0 program deck.



2. Set A-register input switches as desired.
3. Load cards in card reader; depress LOAD CARD, RESET ALARM, and RESET P; place central processor in AUTO mode; depress START.
4. If switch 4 is down, the output from pass 0 will be punched cards. These cards must be arranged in the order described on page A-17. This output will also be listed if the printer is on line. In addition, a packed list of special symbols\* (which may be suppressed by a switch setting), a list of undefined symbols, a list of multiple symbols, and the symbolic names which are not referenced in the program are printed. If no high-speed printer is available on line, the above lists will be typed on the typewriter.

**MESSAGES:**

NO END CARD	Indicates the symbolic deck does not terminate with an END card. Assembly will continue to the normal end of job.
END OF PASS 0	Signifies the end of assembly program run 0.
SYMBOL TABLE OVERFLOW 1	Indicates the number of special operands exceeds 250. Program goes into loop which may be overridden by setting switch 19. This causes pass 0 to continue but special symbols following this loop are not placed in the table.
SYMBOL TABLE OVERFLOW 2	Indicates the total number of symbols exceeds 1000. The program goes into a loop which may be overridden by setting switch 19. Pass 0 then continues but symbols following the loop are not analyzed as undefined, multiple, or no reference symbols.
CARD READ ERROR	<p>Action Required:</p> <ol style="list-style-type: none"> <li>1. Place computer in manual mode.</li> <li>2. Backspace card reader by removing the cards from the hopper and the card from the read platform. Place these cards in front of the deck, replace the deck in the card hopper, and place the last card read in the read platform.</li> <li>3. Press A→I button.</li> <li>4. Place processor in AUTO mode and depress START.</li> </ol>
UNDEFINED SYMBOLS	A symbol or symbols were referenced but were not defined.
MULTIPLE SYMBOLS	A multidefined symbol in the Symbol or Operand fields.
NO REFERENCE	No reference was made to the symbols following this message.
XXX ERRORS TAPE 3	If tape 3 is used for comments, this typeout signifies the number of bad spots on tape 3. (Switch 3 in normal position.)
XXX ERRORS TAPE 4	Signifies the number of bad spots on tape 4. (Switch 4 in normal position.)

---

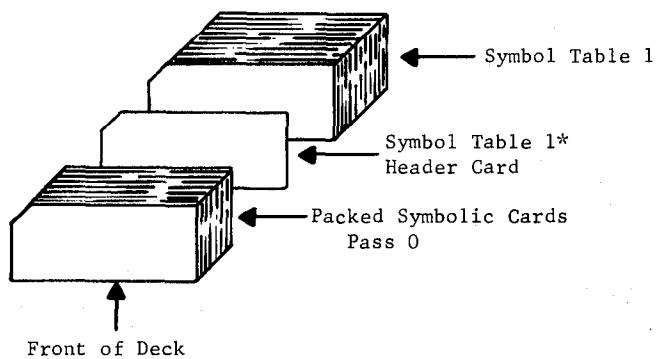
\* Special symbolic operands which are referred to by double-length instruction.

**ACTION REQUIRED AFTER ALL OTHER STOPS:**

1. Place processor in MANUAL mode.
2. Check the CARD PUNCH READY indicator. If punch is not in Ready status, place in Ready status and depress START.
3. Check the N-REGISTER READY light. If not in Ready status, manually type spaces until N-register becomes ready and depress START. If the CARD READER alarm indicator is on, check the card deck for damaged cards. Replace if necessary and reload the program from the beginning.

**Output From Pass 0**

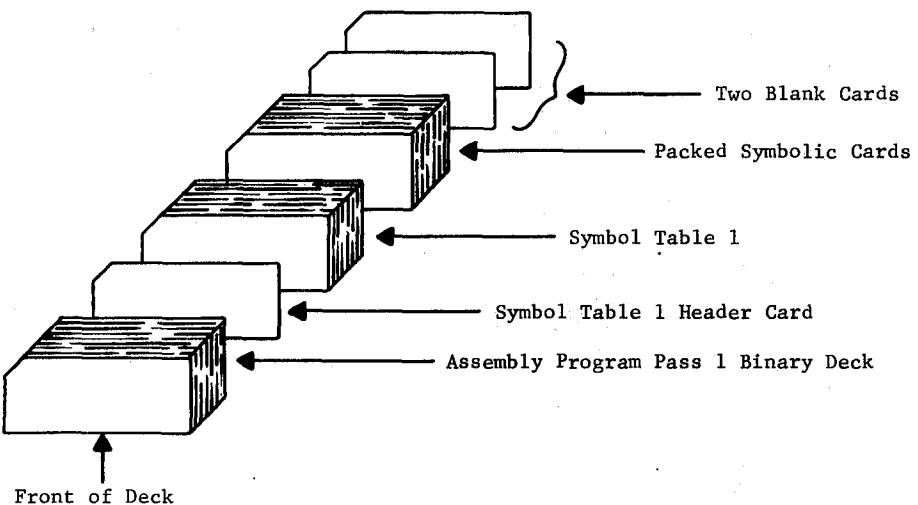
The output from pass 0 is shown below:



\* The symbol table 1 header card may be recognized by the Hollerith character ST1 punched in columns 1, 2, and 3. "S" in Hollerith code is a 0-2 punch. "T" is a 0-3 punch. The code for "1" is a 1 punch.

**Pass 1**

Rearrange the output from pass 0 as shown below:



1. If console switch 4 is down, the output from pass 0 is a packed program with sequence numbers starting with 20000 (columns 74-78) followed by a table of special symbolic operands with sequence numbers starting at 10000 (columns 74-78). The cards that have sequence numbers beginning with 1xxxx should be placed in front of those cards starting with 2xxxx prior to combining them with the pass 1 binary program deck, followed by the rearranged output from pass 0, followed by two blank cards. (See Figure A-3.)
2. Load cards.
3. If switch 4 is down, the output from pass 1 is a sorted table of symbols and equivalent locations, which is punched out. If the printer is on line, these are listed on the high-speed printer. In addition, a list of all multidefined symbols, together with all of the equivalent values associated with each symbol, is printed (or typed if no printer is available).

If switch 4 is in normal position, the output from pass 1 is written on tapes 4 and 5, in which case only the pass 2 program and two blanks are loaded into the card reader.

Errors or possible errors detected in the Operand field of a BSS, EQU, or ORG instruction are printed or typed with the present setting of the memory allocation register, the card type, and the error code. The error codes are:

U - an undefined symbol  
A - a possible error in an address

#### MESSAGES:

NO END CARD	Indicates the symbolic deck does not terminate with an END card. Assembly continues to the normal end of job.
MULTIPLE SYMBOLS	Indicates a multidefined symbol in the Symbol or Operand fields.
END OF PASS 1	Signifies the end of the assembly program pass 1 run.
SYMBOL TABLE OVERFLOW	Messages (and action to be taken) are the same as for the General Assembly Program 0 run.

For all other errors, repeat previous load procedure described for pass 1.

#### **Pass 2**

The procedure for pass 2 is as follows:

1. The input for pass 2 is the output for pass 0 and pass 1. If console switch 4 is down, set up the input deck as follows:

Assembly program pass 2 binary deck followed by the output of pass 1 followed by the rearranged output from pass 0. (See Figure A-3.)

2. Load cards.

MESSAGES:

ERRORS                  Indicates presence of a real or suspected source program error.

NO ERRORS                Indicates no errors were found.

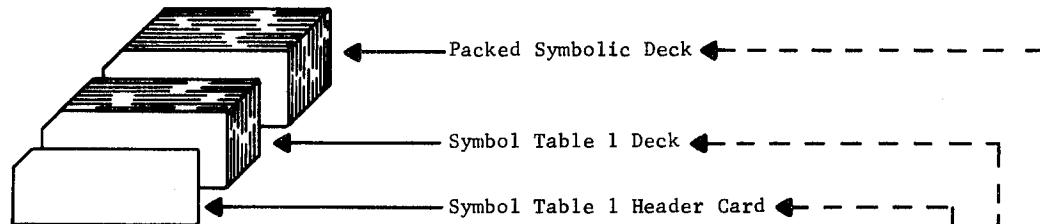
END OF PASS 2           Signifies the end of the pass 2 run.

SYMBOL LOST              Is typed with the setting of the memory allocation register and the symbol in question when a symbol appearing in the Symbol field cannot be found in the symbol table. This is caused by a machine error and may necessitate a reassembly. Action required for SYMBOL LOST:

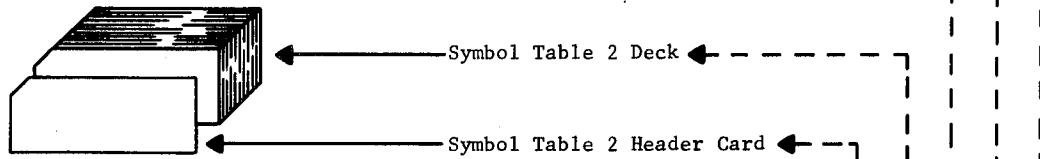
1. List all output from pass 0.
2. Correct cards as necessary.
3. Restart assembly at assembly program pass 0, 1, or 2, as required.

The output from pass 2 is an octal punched card deck, if no printer is on-line, or a printer listing, if a high-speed printer is available, and binary cards. The listing (or octal cards) contains the octal memory location assigned to the instruction in octal, the symbolic instruction, and the codes for real or suspected errors in the instruction.

Rearranged Output from Assembly Program Pass 0, 1, 2



Output from Assembly Program Pass 0, 1, 2



Input to Assembly Program Pass 0, 1, 2

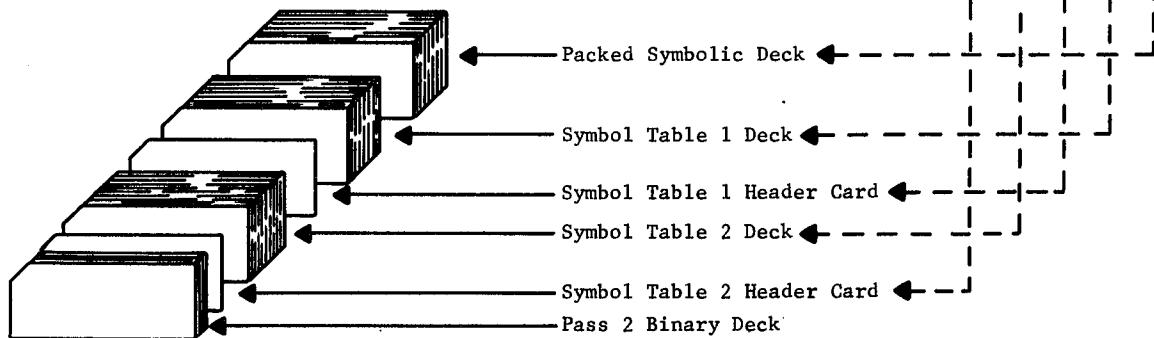


Figure A-3. Arrangement of Input for Pass 2

## CARD OPERATIONS WITH 4k MEMORY

The assembly program can be modified for 4k memory and to accept up to 500 symbols. Binary corrections as listed below should be inserted into the program decks.

PASS 0. Insert the two following cards at:

<u>Location</u> 00053 <sub>8</sub>	<u>Contents</u> 76 <sub>8</sub>
<u>Location</u> 00054 <sub>8</sub>	<u>Contents</u> 7776 <sub>8</sub>

PASS 1. Insert the following card at:

<u>Location</u> 01110 <sub>8</sub>	<u>Contents</u> 764 <sub>8</sub>
------------------------------------	----------------------------------

## SYSTEMS TAPE

The systems tape, CD225F2.002, contains the following programs, linked together in this order:

General Assembly Program 4 (Magnetic Tape Updating)  
General Assembly Program, Pass 0, 1, and 2.  
General Assembly Program 3 (Paper Tape Conversion).

To run program 4, it is necessary to call the program from the systems tape with a GENERAL ASSEMBLY PROGRAM 4 call card. Pass 0, 1, and 2 can be run after program 4 (with switches 1 and 16 down), or by itself (using DATANET-30 GENERAL ASSEMBLY PROGRAM call card). Program 3 can be run by itself after Pass 0, 1, and 2 using GENERAL ASSEMBLY PROGRAM 3 call card.

Program 4 is used to update symbolic source programs. It updates magnetic tape by comparing the sequence number of a card to the sequence number of a record on tape and inserts the card in the correct position.

Program 3 is used to convert punched cards or magnetic tape to perforated tape. It records on perforated tape in either of two formats, depending on the GE-225 console switch setting.

To run the systems tape, follow this procedure:

1. Mount the systems tape on tape unit 1. Mount work tapes on tape units 3, 4, and 5, with the write-permit rings in place. If console switch 6 is set, mount a work tape on tape unit 6.
2. Load the input deck into the card reader. It should be set up as follows:
  - a. DATANET-30 GENERAL ASSEMBLY PROGRAM call card.
  - b. Symbolic program to be assembled. (If symbolic program is on magnetic tape, mount the tape on tape unit 3 and set console switch 16.)
  - c. Two blank cards.
3. Depress RESET ALARM and RESET A, LOAD CARD, and RESET P. Depress AUTO and START.

The assembly program will be called in and will run from start to completion. Error messages are the same as described in preceding pages.

### Paper Tape Conversion, Program 3

Program 3 is a magnetic tape-to-perforated tape or cards-to-perforated tape conversion program. It may be run from the systems tape or by loading assembly program 3 from punched cards. The minimum hardware requirements are as follows:

1. GE-200 Series central processor with 4k memory
2. Card reader or magnetic tape control with tape units
3. Perforated tape punch subsystem
4. Typewriter

Figure A-4 is a flow chart of the operation of assembly program 3.

The input to assembly program 3 may be magnetic tape, binary cards, or octal cards. Type of input (cards or tape) is determined by console switch setting. The format of perforated tape output is determined by switch setting.

Assembly program 3 examines the console switch settings and types a message to the operator instructing him to set the perforated tape punch to the mode specified by the console switches. After acknowledgment by the operator of correct tape mode, the program punches a leader of 18 inches.

Assembly program 3 then reads in the input to be punched and punches it in the desired mode. Punching is continuous until a transfer card or tape end-of-file is detected, at which time assembly program 3 terminates the punching of data.

The program then tests to see if there is more data to be punched in either the card reader or tape unit; if so, it is punched in the specified format. If no more data is to be punched, the program punches 18 inches of trailer, types "end" messages, and terminates.

For perforated tape punches in hardware-load format or program-load format, it is necessary for the system to have a free-standing perforated tape unit with the 8-level, straight transfer mode feature (Model 4WGA652).

**ASSEMBLY PROGRAM 3 SWITCH SETTINGS.** The switch settings for assembly program 3 are as follows:

- |                 |         |   |
|-----------------|---------|---|
| <u>Switch 6</u> | Normal: | Input to assembly program 3 is on cards.  |
|                 | Down:   | Input to assembly program 3 is on tape 6, channel 1.  |
| <u>Switch 7</u> | Normal: | Ignore.   |
|                 | Down:   | Read in assembly program 3 from systems tape program after completion of DATANET-30 General Assembly Program. |

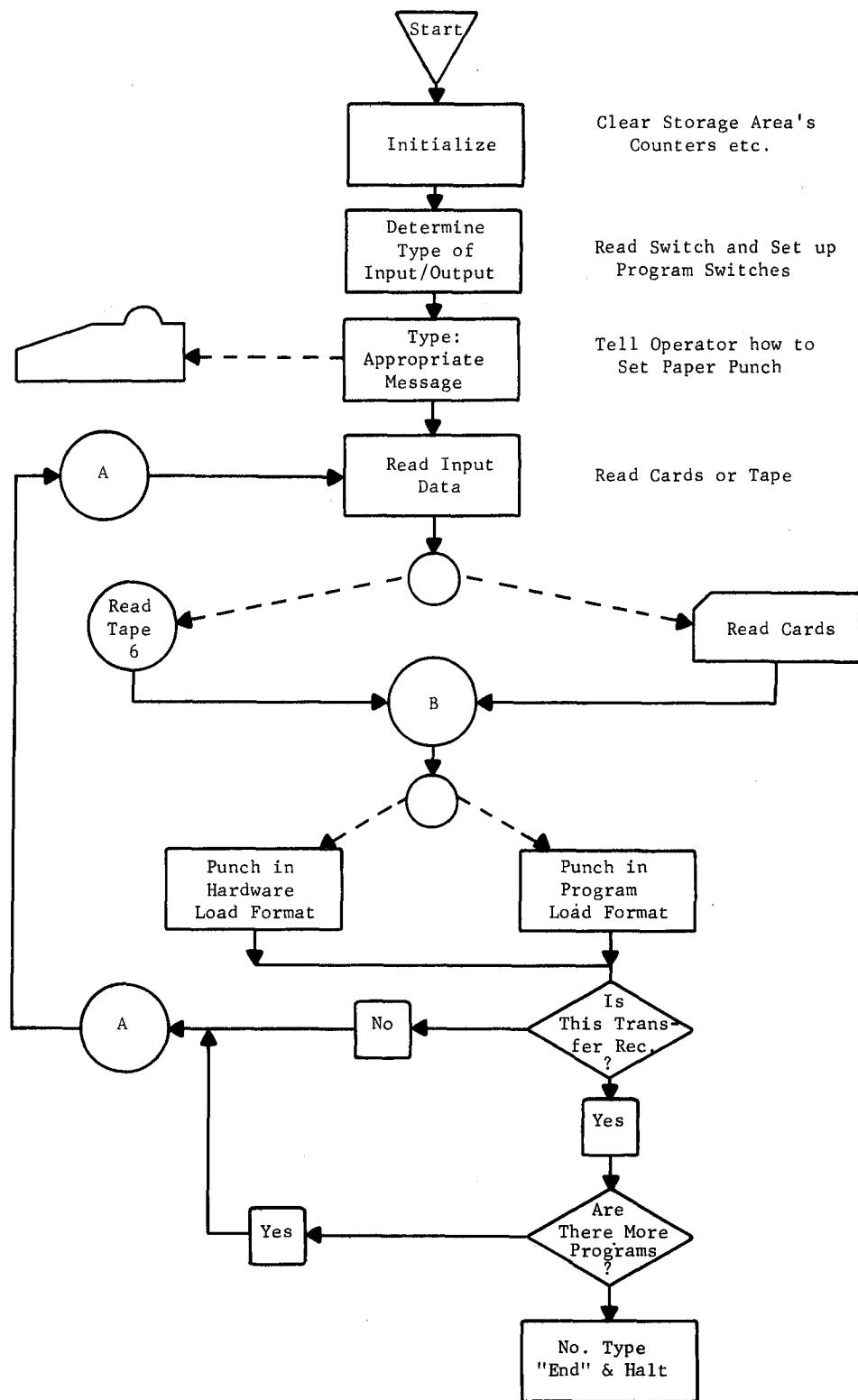


Figure A-4. Flow Chart for Assembly Program 3

Console switches 10, 11, and 12 define the mode in which the output is to be punched. No other modes exist at this time.

FORMAT	SWITCHES		
	10	11	12
Hardware Load Format	Norm	Norm	Norm
Program Load Format	Norm	Norm	Down

**OPERATING INSTRUCTIONS.** If a DATANET-30 systems tape is available, mount the systems tape on tape unit 1, controller 1, and assemble deck as follows:

1. Place assembly program 3 call card in the card reader followed by deck to be punched. If input is on tape, mount the tape on tape unit 6 and set console switch 6.
2. Two blank cards.
3. Set console switch as desired for mode.
4. Depress LOAD CARD, RESET P, AUTO, and START.

If a DATANET-30 systems tape is not available, assemble deck as follows:

1. Place the assembly program 3 program on cards on the card reader followed by the deck to be punched. If input is on tape, mount the tape on tape unit 6 and set console switch 6.
2. Two blanks.
3. Set console switches as desired.
4. Depress LOAD CARD, RESET P, AUTO, and START.

If assembly program 3 is to be run following a DATANET-30 General Assembly Program, the systems tape must be on tape 1:

1. Set console switch 7. This causes assembly program 3 to be read in after completion of the DATANET-30 General Assembly Program.
2. Set console switches as desired for proper mode.
3. Set console switch 6. This writes the output of DATANET-30 General Assembly Program on tape 6.
4. Run the assembly program as previously described.

#### **Magnetic Tape Updating, Program 4**

Assembly program 4 is a magnetic tape generating and updating routine. It may be used to make the symbolic source tapes input to the DATANET-30 General Assembly Program.

Assembly program 4 is included on the DATANET-30 systems tape and may be called in and executed with the assembly program 4 call card, or it may be loaded from punched cards.

The minimum system configuration required is:

1. GE-200 Series central processor with 8k memory
2. Typewriter
3. High-speed printer
4. Magnetic tape controller with two tape units
5. Card reader

CONTROL CARDS. The following control cards are used:

NEW	Characters NEW punched in columns 8, 9, and 10.
FIN	Characters FIN punched in columns 8, 9, and 10.
DEL	Characters DEL punched in columns 8, 9, and 10.

There are two types of DEL control cards:

1. Range delete

<u>Columns 8 - 10</u>	<u>Columns 12 - 16</u>	<u>Columns 76 - 80</u>
DEL	(TO)	(FROM)

Records on the old master starting with sequence number (FROM) to record starting with sequence number (TO) are deleted.

2. Single delete

<u>Columns 8 - 10</u>	<u>Columns 76 - 80</u>
DEL	(THIS)

Record on old master with sequence number (THIS) is deleted.

ACTION ON DETECTING CONTROL CARDS. The action below takes place upon detection of control cards:

1. NEW - The remaining cards in the card reader are written on tape 3 (new master) until an FIN card is detected.
2. FIN - Signifies to assembly program 4 that there are no more cards to read. If any records exist on the old master they are copied to the new master with new sequence numbers inserted. Old master and new master tapes are closed and rewound (new master with end-of-file record). Program is terminated.
3. DEL - Range - Old master is copied to new master with new sequence numbers until columns 76 - 80 of DEL card are equal to columns 76 - 80 of old master. Old master is then searched until columns 76 - 80 of old master are greater than columns 12 - 16 of DEL cards. Another card is read in and processing continues.

**DEL - Single** - Old master is copied to new master with new sequence numbers until columns 76 - 80 of DEL card are equal to columns 76 - 80 of old master.  
Another card is read in and the old master is advanced to the next record.  
Processing continues.

All other cards are assumed to be updating cards. They are inserted according to their sequence number. If the sequence number of an input card is equal to the sequence number of a record on the old master, the input card will replace the old master record.

**Note:** All input cards including control cards must be in sequence columns 76 - 80. Any card out of sequence will be ignored and error flagged. All input decks must end with an FIN card (no sequence number needed) and two blanks.

**OPERATING PROCEDURE.** The operating procedure for assembly program 4 is as follows:

1. Mount the systems tape on tape unit 1, channel 1. If assembly program 4 is to be run from cards, place assembly program 4 in the card reader.
2. Place old master tape to be updated on tape unit 2, channel 1. If a new tape is to be generated, place a work tape on tape unit 2, channel 1.
3. Place a good tape with a write-permit ring on tape unit 3, channel 1. This is the new master.
4. If assembly program 4 is to be run from a systems tape, place a GENERAL ASSEMBLY PROGRAM 4 call card followed by the updating deck in the card reader. If assembly program 4 is to be run from cards, place the updating deck behind the assembly program 4 deck.
5. Depress LOAD CARD, RESET ALARM, RESET P, and START.
6. If DATANET-30 General Assembly Program is to be run following assembly program 4, place the following console switches down:

**Switch 1.** This calls in DATANET-30 General Assembly Program after completion of assembly program 4.

**Switch 16.** This switch is pertinent to DATANET-30 General Assembly Program only. It indicates that the source program is on tape 3.

For other switch settings see operating instructions, page A-14.

## APPENDIX B

### CHARACTERISTICS SUMMARY

#### COMMUNICATIONS PROCESSOR

Single address  
Stored program  
Read/compute/write cycle  
Binary  
18 bit word length  
Parallel  
128 buffer selector channels  
Automatic program reload  
Memory interrupt feature  
Automatic bit buffer scan command  
Elapsed time program interrupt counter  
78 basic instructions  
Indirect addressing  
Indexing  
6.94 microsecond word time

#### MEMORY

6.94 microsecond memory cycle  
Memory size (words):

4,096  
8,192  
16,384

#### HARDWARE SCAN

Bit buffer units only  
5-, 6-, 7-, or 8-level codes  
Scan time: 21 microseconds per simplex, half-duplex, or full-duplex channel.

## INSTRUCTION SUMMARY

		<u>Time in Microseconds</u>
Load	Single and double word	14 and 21 *
Store	Single and double word	14 and 21
Arithmetic	18 bit parallel addition	14
Logical	AND, OR and EXCLUSIVE OR	14
Branch	Conditional and unconditional To subroutine	7 21
Register Transfer		7

## BUFFER SELECTOR BUFFER UNITS

### Bit Buffer Unit

10 simplex channels input and 10 simplex channels output/module  
10 half-duplex channels/module  
10 full-duplex (or echoplex) channels/module

Module data rates (bits/sec)

45  
50  
56.25  
75  
110  
150

Code level: 5, 6, 7, or 8 bits/character

Character format: start/stop bit asynchronous;  
one stop bit (minimum).

Compatible digital subsets: 103A; 103F.  
20 ma d-c loop, bipolar voltage interface, or VCA

## CHARACTER/WORD UNIT (CWU930)

### Character Buffer Channel (CBC930)

2 simplex channels/module  
2 half-duplex channels/module  
1 full-duplex channel/module

\* For ease of computation the 6.94  $\mu$ sec memory cycle is rounded to 7.0  $\mu$ sec.

Channel data rates

300 bits/sec to 2400 bits/sec

Code level: 5, 6, 7, or 8 bits/character

Character format: start/stop bit asynchronous;  
one stop bit (minimum).

Compatible digital subsets: 202A; 202C/D; 103A/F  
Bipolar voltage interface. VCA

#### WORD BUFFER CHANNEL (WBC930)

2 simplex channels/module  
2 half-duplex channels/module  
1 full-duplex channel/module

Channel data rates (bits/sec)

1200  
1800  
2000  
2400

Code level: 20 bits

Character format: start/stop bit asynchronous;  
one stop bit (minimum).  
Compatible digital subsets: 202A; 202C/D; 103A/F  
Bipolar voltage interface, VCA

#### COMPUTER INTERFACE UNIT (CIU930)

GE-200 Series computer interface  
1 CIU/2 modules  
20-bit parallel transfer  
Transfer rate determined by DATANET-30 program

#### COMPUTER INTERFACE UNIT (CIU931)

GE-400/600 Series computer interface  
1 CIU/2 modules  
3 characters per transfer  
Transfer rate synchronized by the DATANET-30

## CONTROLLER SELECTOR UNIT (CSU930)

Maximum transfer rate	28,800 words/sec
Data transfer cycle time	17.34 microseconds
DATANET-30 memory interrupt time	7 microseconds/word
Execute status request	28-70 microseconds

Peripheral Combination Chart:

<u>Peripheral</u>	<u>** Possible Channel Address (plug no.)</u>	<u>* Load Factor Per Peripheral</u>
Single access DSU	0, 1 }	.1
Dual access DSU	0, 1 }	.1
15 kc tape controller	2, 3, 4, 5 }	.3
41.5 kc tape controller	2, 3, 4, 5 }	.45
Printer	6, 7 only	.05

## CONTROLLER SELECTOR UNIT (CSU931)

Maximum transfer rate	57,600 words/sec
Data transfer cycle time	17.34 microseconds
DATANET-30 memory interrupt time	7 microseconds/word

Peripheral Combination Chart:

<u>Peripheral</u>	<u>** Possible Channel Address (plug no.)</u>	<u>* Load Factor Per Peripheral</u>
Single access DSU	0-7	.55
Dual access DSU	0-7	.55
15 kc tape controller	0-7	.1
41.5 kc tape controller	0-7	.28
Printer	6, 7 only	.05

\* The load factor represents the index for peripherals that may be run concurrently if sum of load factors does not exceed 1.00.

\*\* The assignment of a priority channel to a peripheral, and therefore which plug number, depends upon the data transfer rate of the peripheral equipment. The peripheral equipment with the higher transfer rate must have priority over a peripheral with a slower transfer rate.

### **DIALING ADAPTOR UNIT (DAU930)**

10 Dialing Adaptor Units/Module  
Associated buffer units per DAU  
    1 bit buffer  
    1 character buffer  
    1 word buffer  
Telephone company equipment  
    1 automatic calling unit per DAU  
    1 data set per buffer  
Numbers dialed under program control  
Code level: Not applicable  
Character format: Binary to automatic calling unit  
Compatible digital subsets: 103A, 202C

### **PROCESSOR INTERRUPT UNIT (PIU930)**

DATANET-30 to DATANET-30 interface  
1 PIU/module  
18-bit parallel transfer  
Transfer rate 28,800 words/sec

### **COMMON PERIPHERAL CHANNEL (CPC930)**

1 peripheral per CPC  
1 CPC/2 modules  
6-bit character transfer  
Transfer rate up to 57,600 words/sec

### **CARD READER UNIT (CRU930)**

1 card reader per CRU  
1 CRU/module  
Reads 400 cards per minute

### **PUNCH/READER UNIT (PRU930)**

1 card reader and 1 card punch per PRU  
1 PRU/2 modules  
Reads 400 cards per minute  
Punches 100 cards per minute

### **CHARACTER BUFFER UNIT (CBU931)**

2 character buffer channels/module  
Interface with the UNIVAC 1004 system

### **PARALLEL CHANNEL ADAPTOR (PCA930)**

Will parallel one communication line via one VCA (or DSS) into two Bit Buffer Channels (BBC931G4), two Character Buffer Channels (CBC930), or two Word Buffer Channels (WBC930) of two separate DATANET-30 processors.

### **PARALLEL CHANNEL ADAPTOR (PCA931)**

Will parallel ten communication lines via one VCA931 (or VCA940) into ten Bit Buffer Channels (BBC931G4) of two separate DATANET-30 processors. May be combined with a CBC.

### **DUAL CHANNEL ADAPTOR**

Transfers control of a peripheral controller between a DATANET-30 with a CPC930 and a GE-400 or -600 Series computer.

## APPENDIX C

### INSTRUCTION SUMMARY CONVERSION TABLE, 5-LEVEL BAUDOT TO OCTAL

#### MACRO COMMANDS

The DATANET-30 assembly program recognizes various macro commands, and will assemble them as follows:

CL2	F, T . . . . .	CL1	F, T CL1	T, T	CR3	F, T . . . . .	CR1	F, T CR1	T, T CR1	T, T	
CL3	F, T . . . . .	CL1	F, T CL1	T, T CL1	T, T	CR4	F, T . . . . .	CR6	F, T CL1	T, T CL1	T, T
CL4	F, T . . . . .	CL6	F, T CR1	T, T CR1	T, T	CR5	F, T . . . . .	CR6	F, T CL1	T, T	
CL5	F, T . . . . .	CL6	F, T CR1	T, T	CR7	F, T . . . . .	CR6	F, T CR1	T, T		
CL7	F, T . . . . .	CL6	F, T CL1	T, T	CR8	F, T . . . . .	CR6	F, T CR1	T, T CR1	T, T	
CL8	F, T . . . . .	CL6	F, T CL1	T, T CL1	T, T	CR9	F, T . . . . .	CR6	F, T CR1	T, T CR1	T, T
CL9	F, T . . . . .	CL6	F, T CL1	T, T CL1	T, T CL1	T, T					
CR2	F, T . . . . .	CR1	F, T CR1	T, T	SL2	F, T . . . . .	SL1	F, T SL1	T, T SL1	T, T	
					SL3	F, T . . . . .	SL1	F, T SL1	T, T SL1	T, T	

SL4	F, T . . . . .	SL1	F, T SL1 T, T SL1 T, T SL1 T, T	SR7	F, T . . . . .	SR6	F, T SR1 T, T
SL5	F, T . . . . .	SL1	F, T SL1 T, T SL1 T, T SL1 T, T SL1 T, T	SR8	F, T . . . . .	SR6	F, T SR1 T, T SR1 T, T
SL7	F, T . . . . .	SL6	F, T SL1 T, T	SR9	F, T . . . . .	SR6	F, T SR1 T, T SR1 T, T SR1 T, T
SL8	F, T . . . . .	SL6	F, T SL1 T, T SL1 T, T	SAM	$\alpha$ . . . . .	CMM	$\alpha$ AAM $\alpha$ CMM $\alpha$
SL9	F, T . . . . .	SL6	F, T SL1 T, T SL1 T, T SL1 T, T	SBM	$\alpha$ . . . . .	CMM	$\alpha$ ABM $\alpha$ CMM $\alpha$
SR2	F, T . . . . .	SR1	F, T SR1 T, T	SMA	$\alpha$ . . . . .	TRC	A, A AMA TRC $\alpha$ A, A
SR3	F, T . . . . .	SR1	F, T SR1 T, T SR1 T, T	SMB	$\alpha$ . . . . .	TRC	B, B AMB $\alpha$ TRC B, B
SR4	F, T . . . . .	SR1	F, T SR1 T, T SR1 T, T SR1 T, T	SLD	I . . . . .	(SLS (SL1	A, A B, B
SR5	F, T . . . . .	SR1	F, T SR1 T, T SR1 T, T SR1 T, T SR1 T, T	SRD	I . . . . .	(SRS (SRI	B, B A, A
						I times . . . . .	A, A B, B

F is the Register FROM  
T is the Register TO

The macro commands that are register transfer commands (with the exception of the double shifts) have the same error checks as a non-macro register transfer command, plus some additional checks. An error will be flagged when the user attempts to:

Register Transfer MACRO 0, anything  
Register Transfer MACRO anything, Z  
Register Transfer MACRO anything, T

The FROM-TO bits in the instruction will not be deleted on any of the above errors. The error tag only signifies that the instruction should be examined to see if it is correct.

The macro commands SMA, SMB, SAM and SBM, will have the same error checks and same addressing capabilities as non-macro commands requiring a memory address.

No error checks are performed on the macro double shift commands SLD and SRD. The operand must be decimal and must be left-justified in the operand field.

**BAUDOT TO OCTAL  
CONVERSION TABLE  
(Alphabetic Sequence)**

LETTERS	FIGURES	LEFT JUSTIFIED	RIGHT JUSTIFIED
A	-	06	03
B	?	62	31
C	:	34	16
D	\$	22	11
E	3	02	01
F	! *)	32	15
G	+	64	32
H	#	50	24
I	8	14	06
J	' *)	26	13
K	(	36	17
L	)	44	22
M	.	70	34
N	,	30	14
Ø	9	60	30
P	0	54	26
Q	1	56	27
R	4	24	12
S	BELL *)	12	05
T	5	40	20
U	7	16	07
V	;) *)	74	36
W	2	46	23
X	/	72	35
Y	6	52	25
Z	" *)	42	21
0		54	26
1		56	27
2		46	23
3		02	01
4		24	12
5		40	20
6		52	25
7		16	07
8		14	06
9		60	30
BLANK	BLANK	00	00
LTRS.	LTRS.	76	37
FIGS.	FIGS.	66	33
L.FEED	L.FEED	04	02
SPACE	SPACE	10	04
CR. RET.	CR. RET.	20	10

LETTERS	HIGH-SPEED PRINTER OCTAL	FIGURES	OCTAL
Alphabetic Sequence			
A	21	-	40
B	22		55
C	23		57
D	24	\$	43
E	25	3	03
F	26	,	73
G	27	+	20
H	30	#	13
I	31	8	10
J	41	J	41
K	42	(	75
L	43	)	76
M	44	.	33
N	45	,	73
Ø	46	9	11
P	47	0	00
Q	50	1	01
R	51	4	04
S	62	S	62
T	63	5	05
U	64	7	07
V	65	,	73
W	66	2	02
X	67	/	61
Y	70	6	06
Z	71	#	13
Numerical Sequence			
	0	00	
	1	01	
	2	02	
	3	03	
	4	04	
	5	05	
	6	06	
	7	07	
	8	10	
	9	11	
	Δ	60	
	@	14	
	*	54	
	=	16	
	%	15	
		74	

\*) Note: These symbols are not on printer; for convenience, however, they are printed on this form.

**BAUDOT TO OCTAL  
CONVERSION TABLE  
(Numerical Sequence)**

LETTERS	FIGURES	LEFT JUSTIFIED	RIGHT JUSTIFIED
Blank	Blank	00	00
E	3	02	01
Line Feed	Line Feed	04	02
A	-	06	03
Space	Space	10	04
S	BELL *)	12	05
I	8	14	06
U	7	16	07
Carr. Ret.	Carr. Ret.	20	10
D	\$	22	11
R	4	24	12
J	' *)	26	13
N	,	30	14
F	! *)	32	15
C	:	34	16
K	(	36	17
T	5	40	20
Z	" *)	42	21
L	)	44	22
W	2	46	23
H	#	50	24
Y	6	52	25
P	0	54	26
Q	1	56	27
Ø	9	60	30
B	?	62	31
G	+	64	32
Figs.	Figs.	66	33
M	.	70	34
X	/	72	35
V	;) *)	74	36
Ltrs.	Ltrs.	76	37

LETTERS	HIGH-SPEED PRINTER OCTAL	FIGURES	OCTAL
Δ	60	Δ	60
E	25	3	03
=	16	=	16
A	21	-	40
-	15	-	15
S	62	S	62
I	31	8	10
U	64	7	07
%	74	%	74
D	24	\$	53
R	51	4	04
J	41	J	41
N	45	,	73
F	26	,	73
C	23	,	57
K	42	(	75
T	63	5	05
Z	71	#	13
L	43	)	76
W	66	2	02
H	30	#	13
Y	70	6	06
P	47	0	00
Q	50	1	01
Ø	46	9	11
B	22		55
G	27	+	20
*	54	*	54
M	44	.	33
X	67	/	61
V	65	,	73
@	14	@	14

\*) Note: These symbols are not on printer; for convenience, however, they are printed on this form.

CHARACTER	300 LPM PRINTER SYMBOLS	CONSOLE TYPEWRITER CHARACTER OR ACTION	PERFORATED PAPER TAPE (8 LEVEL)	HOLLERITH CODE (PUNCH IN ROWS)	BCD MEMORY (OCTAL)**	BCD MAGNETIC TAPE (OCTAL)
0	0	0	Space	0	00	12
1	1	1	1	1	01	01
2	2	2	2	2	02	02
3	3	3	3	3	03	03
4	4	4	4	4	04	04
5	5	5	5	5	05	05
6	6	6	6	6	06	06
7	7	7	7	7	07	07
8	8	8	8	8	10	10
9	9	9	9	9	11	11
A	A	A	/	12-1	21	61
B	B	B	S	12-2	22	62
C	C	C	T	12-3	23	63
D	D	D	U	12-4	24	64
E	E	E	V	12-5	25	65
F	F	F	W	12-6	26	66
G	G	G	X	12-7	27	67
H	H	H	Y	12-8	30	70
I	I	I	Z	12-9	31	71
J	J	J	J	11-1	41	41
K	K	K	K	11-2	42	42
L	L	L	L	11-3	43	43
M	M	M	M	11-4	44	44
N	N	N	N	11-5	45	45
O	O	O	O	11-6	46	46
P	P	P	P	11-7	47	47
Q	Q	Q	Q	11-8	50	50
R	R	R	R	11-9	51	51
S	S	S	B	0-2	62	22
T	T	T	C	0-3	63	23
U	U	U	D	0-4	64	24
V	V	V	E	0-5	65	25
W	W	W	F	0-6	66	26
X	X	X	G	0-7	67	27
Y	Y	Y	H	0-8	70	30
Z	Z	Z	I	0-9	71	31
+	+	-	0	12	20	60
-	-	-	-	11	40	40
(Space)	(Blank)	(Blank)	&	Blank	60	20
/	/	/	A	0-1	61	21
#	#	/		2-8	12	12
@	@		Stop	3-8	13	13
(Underline)	=	=		4-8	14	14
=	=	=		5-8	15	15
				6-8	16	16
				7-8	17	
				12-2-8	32*	72
+0				12-0	32*	
.	.	.		12-3-8	33	73
				12-4-8	34	
				12-5-8	35	75
			Tab	12-6-8	36	76
-0			Carriage Return	12-7-8	37	77
\$	\$	\$		11-0	52*	52
*	*	*		11-2-8	52*	52
				11-3-8	53	53
				11-4-8	54	54
				11-5-8	55	55
				11-6-8	56	56
				11-7-8	57	57
%	%		Print Red	0-2-8	72	32
(	[		Print Black	0-3-8	73	33
)	]		Tab	0-4-8	74	34
			Delete	0-5-8	75	35
				0-6-8	76	36
				0-7-8	77	37

GE-225 Representation of Characters

Standard Character Set	GE-Internal Machine Code	Octal Code	Hollerith Card Code	Standard Character Set	GE-Internal Machine Code	Octal Code	Hollerith Card Code
0	00 0000	00	0	↑	10 0000	40	11-0
1	00 0001	01	1	J	10 0001	41	11-1
2	00 0010	02	2	K	10 0010	42	11-2
3	00 0011	03	3	L	10 0011	43	11-3
4	00 0100	04	4	M	10 0100	44	11-4
5	00 0101	05	5	N	10 0101	45	11-5
6	00 0110	06	6	O	10 0110	46	11-6
7	00 0111	07	7	P	10 0111	47	11-7
8	00 1000	10	8	Q	10 1000	50	11-8
9	00 1001	11	9	R	10 1001	51	11-9
[	00 1010	12	2-8	-	10 1010	52	11
#	00 1011	13	3-8	\$	10 1011	53	11-3-8
@	00 1100	14	4-8	*	10 1100	54	11-4-8
:	00 1101	15	5-8	)	10 1101	55	11-5-8
>	00 1110	16	6-8	;	10 1110	56	11-6-8
?	00 1111	17	7-8	!	10 1111	57	11-7-8
\$	01 0000	20	(blank)	+	11 0000	60	12-0
A	01 0001	21	12-1	/	11 0001	61	0-1
B	01 0010	22	12-2	S	11 0010	62	0-2
C	01 0011	23	12-3	T	11 0011	63	0-3
D	01 0100	24	12-4	U	11 0100	64	0-4
E	01 0101	25	12-5	V	11 0101	65	0-5
F	01 0110	26	12-6	W	11 0110	66	0-6
G	01 0111	27	12-7	X	11 0111	67	0-7
H	01 1000	30	12-8	Y	11 1000	70	0-8
I	01 1001	31	12-9	Z	11 1001	71	0-9
&	01 1010	32	12	◀	11 1010	72	0-2-8
.	01 1011	33	12-3-8	,	11 1011	73	0-3-8
]	01 1100	34	12-4-8	%	11 1100	74	0-4-8
(	01 1101	35	12-5-8	=	11 1101	75	0-5-8
<	01 1110	36	12-6-8	"	11 1110	76	0-6-8
\	01 1111	37	12-7-8	:	11 1111	77	0-7-8

Characters inside ||x|| have special Edit functions.  
 Characters inside (x) normally do not print.

#### GE-Compatibles/400 Standard Character Set

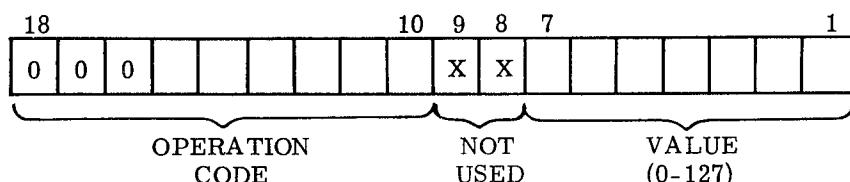
Standard Character Set	GE-Internal Machine Code	Octal Code	Hollerith Card Code	Standard Character Set	GE-Internal Machine Code	Octal Code	Hollerith Card Code
0	00 0000	00	0	↑	10 0000	40	11-0
1	00 0001	01	1	J	10 0001	41	11-1
2	00 0010	02	2	K	10 0010	42	11-2
3	00 0011	03	3	L	10 0011	43	11-3
4	00 0100	04	4	M	10 0100	44	11-4
5	00 0101	05	5	N	10 0101	45	11-5
6	00 0110	06	6	O	10 0110	46	11-6
7	00 0111	07	7	P	10 0111	47	11-7
8	00 1000	10	8	Q	10 1000	50	11-8
9	00 1001	11	9	R	10 1001	51	11-9
[	00 1010	12	2-8	-	10 1010	52	11
#	00 1011	13	3-8	\$	10 1011	53	11-3-8
@	00 1100	14	4-8	*	10 1100	54	11-4-8
:	00 1101	15	5-8	)	10 1101	55	11-5-8
>	00 1110	16	6-8	;	10 1110	56	11-6-8
?	00 1111	17	7-8	'	10 1111	57	11-7-8
£	01 0000	20	(blank)	+	11 0000	60	12-0
A	01 0001	21	12-1	/	11 0001	61	0-1
B	01 0010	22	12-2	S	11 0010	62	0-2
C	01 0011	23	12-3	T	11 0011	63	0-3
D	01 0100	24	12-4	U	11 0100	64	0-4
E	01 0101	25	12-5	V	11 0101	65	0-5
F	01 0110	26	12-6	W	11 0110	66	0-6
G	01 0111	27	12-7	X	11 0111	67	0-7
H	01 1000	30	12-8	Y	11 1000	70	0-8
I	01 1001	31	12-9	Z	11 1001	71	0-9
&	01 1010	32	12	←	11 1010	72	0-2-8
.	01 1011	33	12-3-8	,	11 1011	73	0-3-8
J	01 1100	34	12-4-8	%	11 1100	74	0-4-8
(	01 1101	35	12-5-8	=	11 1101	75	0-5-8
<	01 1110	36	12-6-8	"	11 1110	76	0-6-8
\	01 1111	37	12-7-8	!	11 1111	77	0-7-8

GE-625/635 Standard Character Set

## INSTRUCTION FORMATS SUMMARY

### A. Instructions Without Memory Addressing.

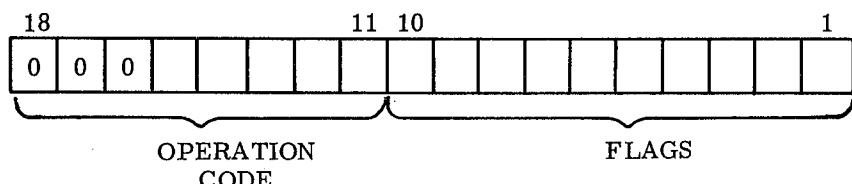
#### 1. C-Register instructions.



Code - Bits 18-10

010-014. -AIC, PIC, NCZ, XCZ.

#### 2. Other nonmemory addressing instructions.

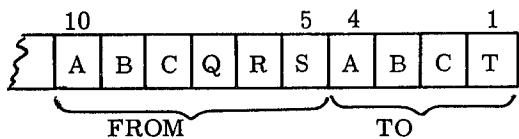


Code - Bits 18-11, even-numbered codes only.

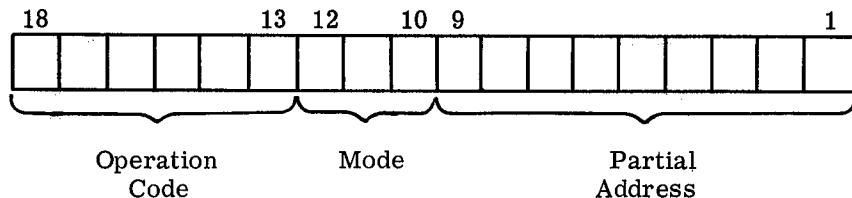
- 00X. Conditional Halt - HLT.
  - 020-026. Status lines and function drivers - NIS, NES, DIF, DEF.
  - 030-032. Hardware scan and controller selector - SCN, CSR.
  - 040-046. Shift - SL1, SR1, SL6, SR6.
  - 050-056. Circulate - CL1, CR1, CL6, CR6.
  - 060-066. Transfer - TRA, TRC, BC0, BC1.
  - 070-076. Special shift and circulate - SLS, SRS.
- Flags - Bits 10-1

Status Lines and Function Drivers - Bit position number (10-1) corresponds to line or driver number.

Shift, Circulate, and Transfer - Bits 10-5 indicate "From"; Bits 4-1 indicate "To". Each bit position has register significance (see below). No bits in "From" indicates zero. No bits in "To" indicates Z-drivers.



## B. Instructions With Memory Addressing.



### Codes - Bits 18-13

- 10-17. Branch - BRU, BRS, BZE, BNZ, BPL, BMI, BEV, BOD.
- 20-26. From memory to registers, non A or B.  
LDC, LDD, LDZ, LDQ, LDT, LDF.
- 30-37. To memory from registers, non A or B.  
STC, STD, STZ, CMM, ADO, SBO, STF, AMD.
- 40-46. From memory to A - LDA, CMA, AMA, NMA, RMA, XMA, AAZ.
- 50-57. To memory from A - STA, CAM, AAM, NAM, RAM, XAM, NAZ, XAZ.
- 60-66. From memory to B - LDB, CMB, AMB, NMB, RMB, XMB, ABZ.
- 70-77. To memory from B - STB, CBM, ABM, NBM, RBM, XBM, NBZ, XBZ.

### Modes - Bits 12-10

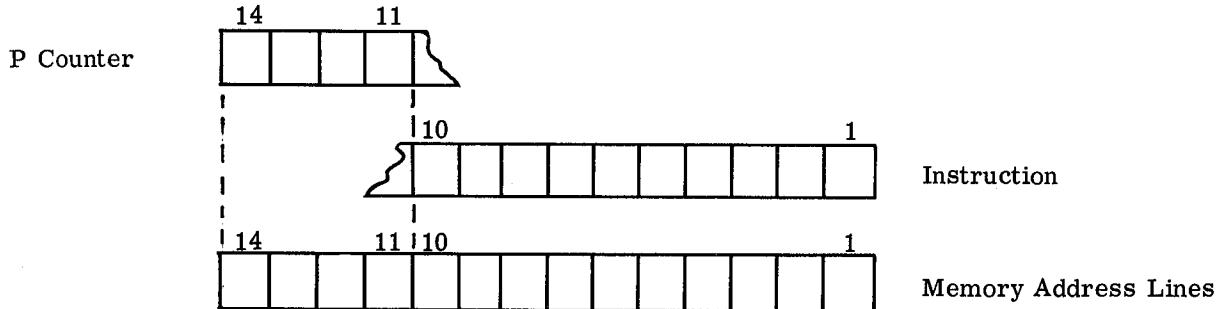
- |       |                             |
|-------|-----------------------------|
| 0-3.  | Direct addressing           |
| 0, 1. | Program bank addressing     |
| 2.    | Common data bank addressing |
| 3.    | Channel table addressing    |
| 4-7.  | Indirect addressing         |
| 4, 5. | Program bank addressing     |
| 6.    | Common data bank addressing |
| 7.    | Channel table addressing    |

		Bits		
		12	11	10
Direct	0	0	0	P.B.
	1	0	1	P.B.
	2	0	1	C.D.B.
	3	0	1	C.T.
Indirect	4	1	0	P.B.
	5	1	0	P.B.
	6	1	1	C.D.B.
	7	1	1	C.T.

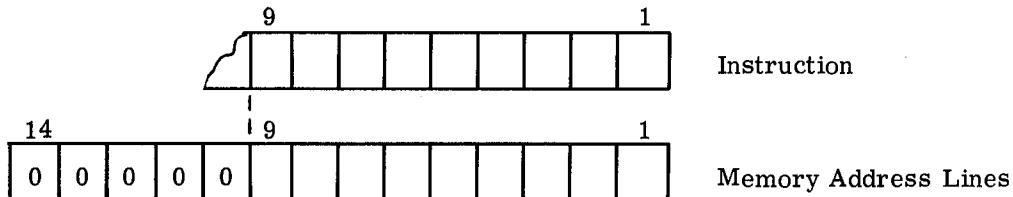
## HARDWARE ADDRESSING RESPONSE TO INSTRUCTION FORMAT

### A. Direct Addressing - Bit 12 of Instruction, Off.

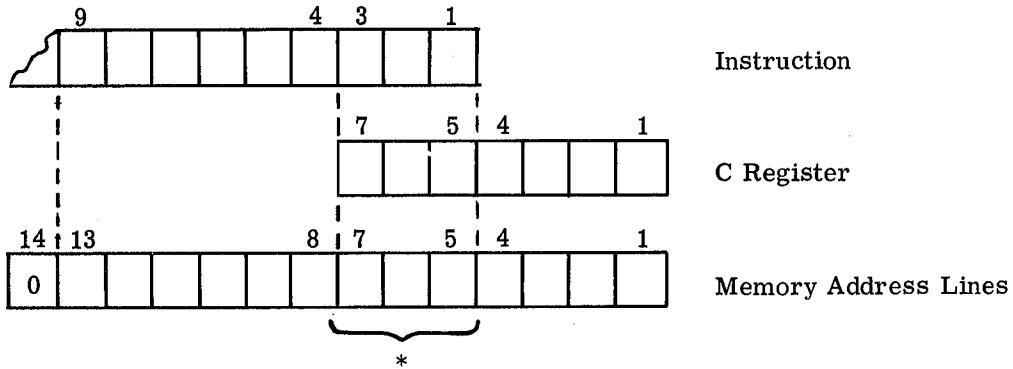
1. Program bank - 4, 8, or 16 groups of 1024 words, depending on memory size.



2. Common data bank - first 512 words only.



3. Channel table - First 4096 or 8192 words only, depending on memory size.

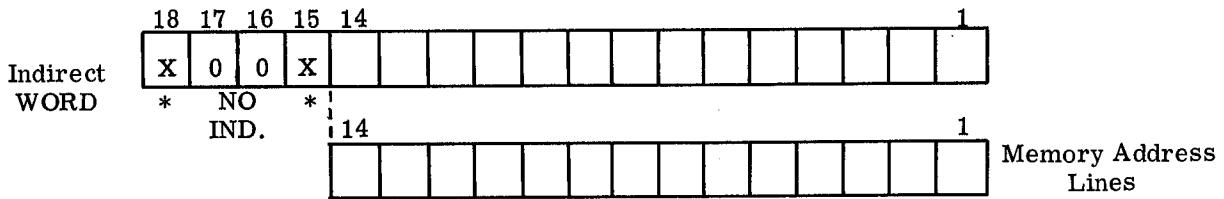


**\*Note:** Bits 9-1 of the instruction and the entire C register are logically or'ed -not added- into the memory address lines offset as shown. Since three bits overlap, care must be exercised with respect to designating channel table length and location to insure valid addressing.

B. Indirect Addressing - Bit 12 of Instruction, On.

An address is first generated as a direct address. This address is the memory location of a word containing a 14-bit address that is used as a base address for instruction execution.

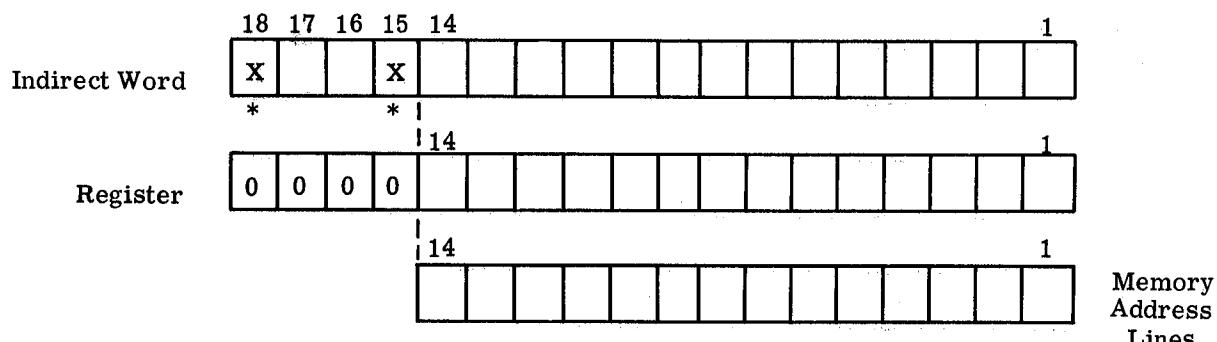
1. Indirect addressing without register indexing. - IND



\* Not Used

The 14-bit address in the indirect word is the address used for execution.

2. Indirect addressing with register indexing. -INA, INB, INC

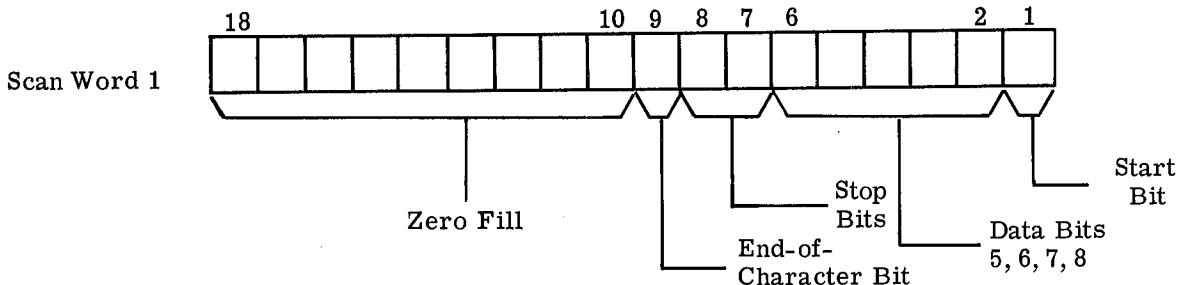


Bits 14-1 of the indirect word and the contents of the specified register are added together into the memory address lines for the execution address. The register is specified by bits 17 and 16 of the indirect word as shown below.

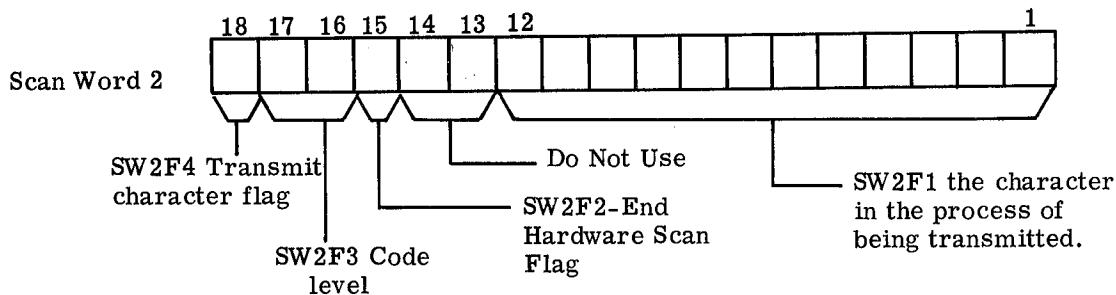
Bits		
17	16	No Register Indexing
0	0	Register A
0	1	Register B
1	0	Register C
1	1	

## SCAN WORD SUMMARY

The next character to be transmitted

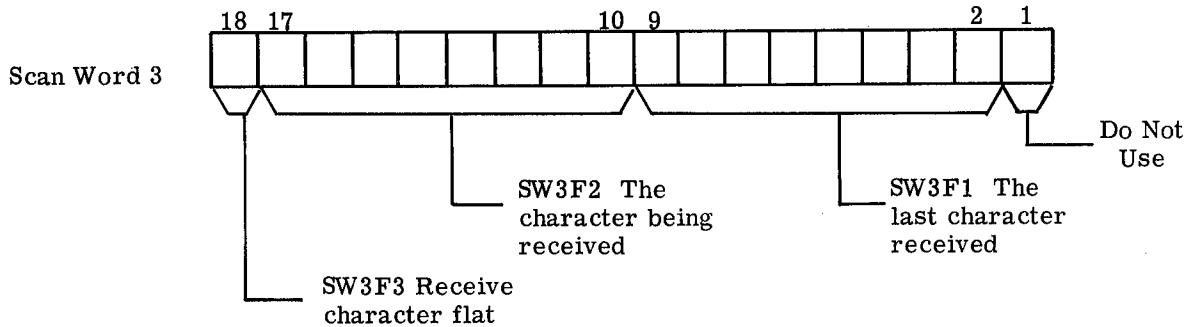


**Initialize:** Start, data bits, stop and end-of-character bits to all 1's. Zero fill.



Code Level	Bits	
	17	16
5	0	0
6	0	1
7	1	0
8	1	1

Initialize: Bits 1-12 as in Scan Word 1. Bits 16, 17 to indicate code level and bit 15 for end scan at last channel. Bit 18 as zero.

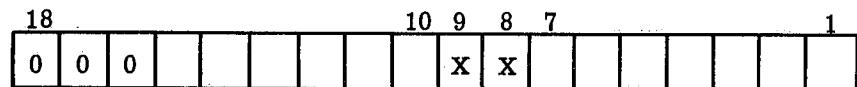


Initialize: As all zeros.

## INSTRUCTION WORD FORMATS SUMMARY

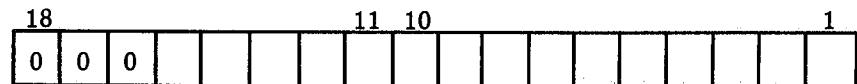
### Non-Memory Addressing

#### C REGISTER



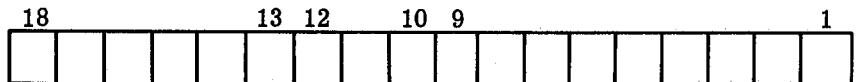
OPERATION CODE                   BITS 10-18  
NOT USED                         BITS 8, 9  
VALUE                             BITS 1-7

#### OTHER



OPERATION CODE                   BITS 11-18  
VARIATION                       BITS 1-10

### Memory Addressing



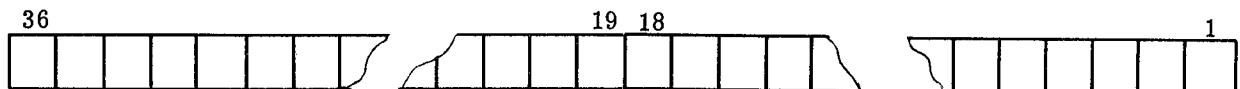
OPERATION CODE                   BITS 13-18  
ADDRESSING MODE                 BITS 10-12  
PARTIAL ADDRESS                 BITS 1-9

## DATA WORD FORMATS SUMMARY



BINARY SINGLE LENGTH

BIT 18 IS SIGN



BINARY DOUBLE LENGTH

BIT 36 IS SIGN



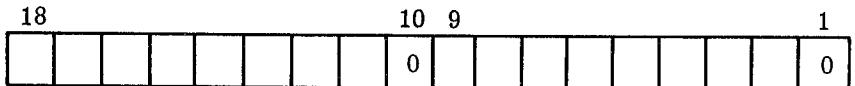
BCD CHARACTERS

3 TO A WORD



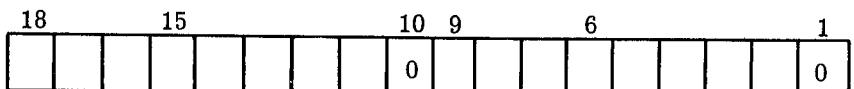
5 LEVEL TELETYPE CHARACTERS  
BITS 1, 7, AND 13 ARE START BITS

3 TO A WORD



8 LEVEL TELETYPE CHARACTERS  
BITS 1 AND 10 ARE START BITS

2 TO A WORD



8 LEVEL FRIDEN CHARACTERS  
BITS 1 AND 10 ARE START BITS  
BITS 6 AND 15 ARE PARITY BITS  
BITS 9 AND 8 ARE CONTROL BITS

2 TO A WORD

## DATANET-30 DATA COMMUNICATIONS PROCESSOR

### INSTRUCTION REPERTOIRE

WORD TIMES	CODE OCTAL	OPERAND	FUNCTIONAL DESCRIPTION
<b>***** LOAD INSTRUCTIONS</b>			
2	40	LDA M	LOAD A FROM M
2	60	LDB M	LOAD B FROM M
2	20	LDC M	LOAD C FROM M
3	21	LDD M	LOAD DOUBLE -- A FROM M, B FROM M+1
2	26	LDF M	LOAD SPECIAL FLIP-FLOPS FROM M
2	23	LDQ M	LOAD Q FROM M
2	25	LDT M	LOAD T -- SEND M TO TRANSMIT DATA DRIVERS
2	22	LDZ M	LOAD Z -- SEND M TO Z DRIVERS (NO FURTHER)
2	41	CMA M	LOAD A WITH M-NOT (COMPLEMENT M TO A)
2	61	CMB M	LOAD B WITH M-NOT (COMPLEMENT M TO B)
1	011	PIC I	PLACE I IN C
<b>***** STORE INSTRUCTIONS</b>			
2	50	STA M	STORE A IN M
2	70	STB M	STORE B IN M
2	30	STC M	STORE C IN M
3	31	STD M	STORE DOUBLE -- A IN M, B IN M+1
2	36	STF M	STORE SPECIAL FLIP-FLOPS IN M
2	32	STZ M	STORE ZERO IN M
2	51	CAM M	STORE A-NOT IN M (COMPLEMENT A TO M)
2	71	CBM M	STORE B-NOT IN M (COMPLEMENT B TO M)
2	33	CMM M	STORE M-NOT IN M (COMPLEMENT M TO M)
<b>***** ARITHMETIC INSTRUCTIONS</b>			
2	42	AMA M	ADD M TO A
2	52	AAM M	ADD A TO M
2	46	AAZ M	ADD A, M - RESULT TO Z DRIVERS
2	62	AMB M	ADD M TO B
2	72	ABM M	ADD B TO M
2	66	ABZ M	ADD B, M - RESULT TO Z DRIVERS
2	37	AMD M	ADD DOUBLE LENGTH WORD M-(M+1) TO A-B
2	34	ADO M	ADD ONE TO M
2	35	SBO M	SUBTRACT ONE FROM M
1	010	AIC I	ADD I TO C

WORD TIMES	CODE OCTAL	OPERAND	FUNCTIONAL DESCRIPTION
---------------	---------------	---------	------------------------

\*\*\*\*\* BRANCH INSTRUCTIONS

1	10	BRU	M	BRANCH UNCONDITIONALLY
3	11	BRS	M	BRANCH TO SUBROUTINE
1	12	BZE	M	BRANCH IF ZERO FF IS ZERO
1	13	BNZ	M	BRANCH IF ZERO FF IS NON-ZERO
1	14	BPL	M	BRANCH IF PLUS FF IS PLUS
1	15	BMI	M	BRANCH IF PLUS FF IS MINUS
1	16	BEV	M	BRANCH IF EVEN FF IS EVEN
1	17	BOD	M	BRANCH IF EVEN FF IS ODD

\*\*\*\*\* LOGICAL OPERATION INSTRUCTIONS

2	43	NMA	M	M AND A TO A
2	53	NAM	M	M AND A TO M
2	63	NMB	M	M AND B TO B
2	73	NBM	M	M AND B TO M
2	56	NAZ	M	M AND A TO Z ONLY
2	76	NBZ	M	M AND B TO Z ONLY
1	012	NCZ	I	I AND C TO Z ONLY
1	020	NIS	I	I AND INTERNAL STATUS LINES TO Z ONLY
1	022	NES	I	I AND EXTERNAL STATUS LINES TO Z ONLY
2	44	BMA	M	M OR A TO A
2	54	RAM	M	M OR A TO M
2	64	RMB	M	M OR B TO B
2	74	RBM	M	M OR B TO M
2	45	XMA	M	M XOR A TO A
2	55	XAM	M	M XOR A TO M
2	65	XMB	M	M XOR B TO B
2	75	XBM	M	M XOR B TO M
2	57	XAZ	M	M XOR A TO Z ONLY
2	77	XBZ	M	M XOR B TO Z ONLY
1	014	XCZ	I	I XOR C TO Z ONLY

\*\*\*\*\* REGISTER TRANSFER INSTRUCTIONS FROM ABCQRS - TO ABCTZ

1	060	TRA	FROM, TO	TRANSFER
1	062	TRC	FROM, TO	TRANSFER COMPLEMENT
1	040	SL1	FROM, TO	SHIFT LEFT ONE
1	042	SR1	FROM, TO	SHIFT RIGHT ONE
1	044	SL6	FROM, TO	SHIFT LEFT SIX
1	046	SR6	FROM, TO	SHIFT RIGHT SIX
1	070	SLS	FROM, TO	SHIFT LEFT SPECIAL
1	072	SRS	FROM, TO	SHIFT RIGHT SPECIAL
1	050	CL1	FROM, TO	CIRCULATE LEFT ONE
1	052	CRI	FROM, TO	CIRCULATE RIGHT ONE
1	054	CL6	FROM, TO	CIRCULATE LEFT SIX
1	056	CR6	FROM, TO	CIRCULATE RIGHT SIX
1	064	BC0	FROM, TO	BIT CHANGE ZERO (8-LEVEL LINE TO 6-BIT)
1	066	BC1	FROM, TO	BIT CHANGE ONE (6-BIT TO 8-LEVEL LINE)

WORD TIMES	CODE OCTAL	OPERAND	FUNCTIONAL DESCRIPTION
---------------	---------------	---------	------------------------

\*\*\*\*\* SPECIAL INSTRUCTIONS

1	00	HLT		CONDITIONAL HALT
1	024	DIF	I	DRIVE INTERNAL FUNCTION LINES
1	026	DEF	I	DRIVE EXTERNAL FUNCTION LINES
	030	SCN	I	SCAN BIT BUFFERS
	032	CSR	I	CONTROLLER STATUS REQUEST

ABBREVIATED  
INSTRUCTION REPERTOIRE

FIRST OCTAL DIGIT	SECOND OCTAL DIGIT								THIRD OCTAL DIGIT
	0	1	2	3	4	5	6	7	
0	HLT	AIC PIC	NIS	SCN	SL1	CL1	TRA	SLS	0 1
0	HLT	NCZ	NES	CSR	SR1	CR1	TRC	SRS	2
0	HLT	XCZ	DIF		SL6	CL6	BG0		4
0	HLT		DEF		SR6	CR6	BC1		6
1	BRU	BRS	BZE	BNZ	BPL	BMI	BEV	BOD	P-counter
2	LDC	LDD	LDZ	LDQ		LDT	LDF		NON- GENERAL
3	STC	STD	STZ	CMM	ADO	SBO	STF	AMD	
4	LDA	CMA	AMA	NMA	RMA	XMA	AAZ		
5	STA	CAM	AAM	NAM	RAM	XAM	NAZ	XAZ	
6	LDB	CMB	AMB	NMB	RMB	XMB	ABZ		
7	STB	CBM	ABM	NBM	RBM	XBM	NBZ	XBZ	

## ALPHANUMERIC LISTING

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
GROUP	MNEMONIC			INTERNAL INSTRUCTIONS	
GROUP	MNEMONIC		BBC	BIT BUFFER CHANNEL INSTRUCTIONS	
GROUP	MNEMONIC		BSU	BUFFER SELECTOR UNIT INSTRUCTIONS	
GROUP	MNEMONIC		CBC	CHARACTER BUFFER CHANNEL INSTRUCTIONS	
GROUP	MNEMONIC		CIU	COMPUTER INTERFACE UNIT INSTRUCTIONS	
GROUP	MNEMONIC		CSU	CONTROLLER SELECTOR UNIT INSTRUCTIONS	
GROUP	MNEMONIC		DSU	DISC STORAGE UNIT INSTRUCTIONS	
GROUP	MNEMONIC		HSP	HIGH SPEED PRINTER INSTRUCTIONS	
GROUP	MNEMONIC		MACRO	GENERAL ASSEMBLY PROGRAM MACRO INSTRUCTIONS	
GROUP	MNEMONIC		MTS	MAGNETIC TAPE SYSTEM INSTRUCTIONS	
GROUP	MNEMONIC		PTR	PAPER TAPE READER INSTRUCTIONS	
GROUP	MNEMONIC		WBC	WORD BUFFER CHANNEL INSTRUCTIONS	
AAM	M	520000		ADD A TO M	3
AAZ	M	460000		ADD A,M - RESULT TO Z DRIVERS	2
ABM	M	720000		ADD B TO M	3
ABZ	M	660000		ADD B,M - RESULT TO Z DRIVERS	2
ADO	M	340000		ADD ONE TO M	3
AIC	I	010000		ADD I TO C	1
AMA	M	420000		ADD M TO A	2
AMB	M	620000		ADD M TO B	2
AMD	M	240000		ADD DOUBLE -- ADD M,M+1 TO A,B	3
BC0	FROM,TO	064000		BIT CHANGE ZERO (8-LEVEL LINE TO 6-BIT)	1
BC1	FROM,TO	066000		BIT CHANGE ONE (6-BIT TO 8-LEVEL LINE)	1
BEV	M	160000		BRANCH IF EVEN FF IS EVEN	1
BKW			MTS	BACKSPACE AND POSITION WRITE HEAD	1+3
BMI	M	150000		BRANCH IF PLUS FF IS MINUS	1
BNZ	M	130000		BRANCH IF ZERO FF IS NON-ZERO	1
BOD	M	170000		BRANCH IF EVEN FF IS ODD	1
BPL	M	140000		BRANCH IF PLUS FF IS PLUS	1
BRS	M	110000		BRANCH TO SUBROUTINE	3
BRU	M	100000		BRANCH UNCONDITIONALLY	1
BZE	M	120000		BRANCH IF ZERO FF IS ZERO	1
CAM	M	510000		STORE A-NOT IN M (COMPLEMENT A TO M)	2
CBM	M	710000		STORE B-NOT IN M (COMPLEMENT B TO M)	2
CL1	FROM,TO	050000		CIRCULATE LEFT 1	1
CL2	FROM,TO		MACRO	CIRCULATE LEFT 2	2
CL3	FROM,TO		MACRO	CIRCULATE LEFT 3	3
CL4	FROM,TO		MACRO	CIRCULATE LEFT 4	3
CL5	FROM,TO		MACRO	CIRCULATE LEFT 5	2
CL6	FROM,TO	054000		CIRCULATE LEFT 6	1
CL7	FROM,TO		MACRO	CIRCULATE LEFT 7	2
CL8	FROM,TO		MACRO	CIRCULATE LEFT 8	3

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
CL9	FROM, TO		MACRO	CIRCULATE LEFT 9	4
CMA	M	410000		LOAD A WITH M-NOT (COMPLEMENT M TO A)	2
CMB	M	610000		LOAD B WITH M-NOT (COMPLEMENT M TO B)	2
CMM	M	330000		STORE M-NOT IN M (COMPLEMENT M TO M)	2
CR1	FROM, TO	052000	MACRO	CIRCULATE RIGHT 1	1
CR2	FROM, TO		MACRO	CIRCULATE RIGHT 2	2
CR3	FROM, TO		MACRO	CIRCULATE RIGHT 3	3
CR4	FROM, TO		MACRO	CIRCULATE RIGHT 4	3
CR5	FROM, TO		MACRO	CIRCULATE RIGHT 5	2
CR6	FROM, TO	056000	MACRO	CIRCULATE RIGHT 6	1
CR7	FROM, TO		MACRO	CIRCULATE RIGHT 7	2
CR8	FROM, TO		MACRO	CIRCULATE RIGHT 8	3
CR9	FROM, TO		MACRO	CIRCULATE RIGHT 9	4
CSR	I	032000	CSU	CONTROLLER STATUS REQUEST	3-10
DEF	I	026000	BSU	DRIVE EXTERNAL FUNCTION	1
DEF	1	026001	BBC	RESET RECEIVE FLAG AND DATA BUFFER	1
DEF	2	026002	BBC	RESET TRANSMIT FLAG AND DATA BUFFER	1
DEF	3	026004	BBC	TURN CARRIER OFF	1
DEF	4	026010	BBC	TURN CARRIER ON	1
DEF	5	026020	BBC	RESET RECEIVE CLOCK	1
DEF	6	026040	BBC	SET ECHO MODE	1
DEF	7	026100	BBC	RESET ECHO MODE	1
DEF	1	026001	CBC	RESET RECEIVE FLAG AND DATA BUFFER	1
DEF	2	026002	CBC	RESET TRANSMIT FLAG AND DATA BUFFER	1
DEF	3	026004	CBC	TURN CARRIER OFF	1
DEF	4	026010	CBC	TURN CARRIER ON	1
DEF	9	026400	CBC	ANSWER INCOMING CALL	1
DEF	0	027000	CBC	DISCONNECT CALL	1
DEF	1	026001	CIU	RESET FLAG AND BUFFER, SET RECEIVE MODE	1
DEF	2	026002	CIU	RESET FLAG AND BUFFER, SET TRANSMIT MODE	1
DEF	9	026400	CIU	AUTOMATIC PRIORITY INTERRUPT THE 225	1
DEF	0	027000	CIU	RESET THE ADDRESS REGISTER	
DEF	1	026001	PTR	RESET FLAG AND READ NEXT CHARACTER	1

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
DEF 1	026001	WBC		RESET RECEIVE FLAG AND DATA BUFFER	1
DEF 2	026002	WBC		RESET TRANSMIT FLAG AND DATA BUFFER	1
DEF 3	026004	WBC		TURN CARRIER OFF	1
DEF 4	026010	WBC		TURN CARRIER ON	1
DIF I	024000			DRIVE INTERNAL FUNCTION	1
DIF 1	024001			RESET CB 1 AND 2, AND RESET PARITY BIT FF	1
DIF 2	024002			RESET BUZZER FLIP-FLOP	1
DIF 3	024004			SET BUZZER FLIP-FLOP	1
DIF 4	024010			INITIATE HARDWARE LOAD PROCESS	1
DIF 7	024100		CSU	SELECT PERIPHERAL CONTROLLER	1+3
DIF 8	024200			SET CONTROL BIT FLIP-FLOP 1	1
DIF 9	024400			SET CONTROL BIT FLIP-FLOP 2	1
DIF 0	025000			SET THE PARITY BIT FLIP-FLOP	1
HLT I	000000			CONDITIONAL HALT	1
LDA M	400000			LOAD A FROM M	2
LDB M	600000			LOAD B FROM M	2
LDC M	200000			LOAD C FROM M	2
LDD M	210000			LOAD DOUBLE -- A,B FROM M,M+1	3
LDF M	260000			LOAD SPECIAL FLIP-FLOPS FROM M	2
LDQ M	230000			LOAD Q FROM M	2
LDT M	250000	BSU		LOAD T (TRANSMIT DATA DRIVERS) FROM M	2
LDZ M	220000			LOAD Z (BRANCH FLIP-FLOPS) FROM M	2
NAM M	530000			M AND A TO M	2
NAZ M	560000			M AND A TO Z ONLY	2
NBM M	730000			M AND B TO M	2
NBZ M	760000			M AND B TO Z ONLY	2
NCZ I	012000			I AND C TO Z ONLY	1
NES I	022000	BSU		I AND EXTERNAL STATUS LINES TO Z ONLY	1
NES 1	022001	BBC		RC FLAG SET (BUFFER CONTAINS A NEW BIT)	1
NES 2	022002	BBC		TX FLAG SET (BUFFER READY FOR A NEW BIT)	1
NES 5	022020	BBC		INTERLOCK ON	1
NES 6	022040	BBC		CARRIER ON	1
NES 1	022001	CBC		RC FLAG SET (BUFFER CONTAINS A NEW CHAR.)	1
NES 2	022002	CBC		TX FLAG SET (BUFFER READY FOR A NEW CHAR.)	1
NES 3	022004	CBC		CALL IN PROGRESS	1
NES 4	022010	CBC		REQUEST ANSWER	1
NES 5	022020	CBC		DATA MODE	1
NES 6	022040	CBC		CARRIER ON	1
NES 7	022100	CBC		CLEAR TO SEND	1
NES 1	022001	CIU		FLAG SET (BUFFER READY)	1
NES 1	022001	PTR		READ FLAG SET (BUFFER CONTAINS A NEW CHAR.)	1

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
NES	1	022001	WBC	RC FLAG SET (BUFFER CONTAINS A NEW WORD)	1
NES	2	022002	WBC	TX FLAG SET (BUFFER READY FOR A NEW WORD)	1
NIS	I	020000		I AND INTERNAL STATUS LINES TO Z ONLY	1
NIS	1	020001		CHARACTER PARITY OUTPUT	1
NIS	2	020002		WORD PARITY OUTPUT	1
NIS	3	020004		CB FF 2 AND WORD PARITY OUTPUT ARE EQUAL	1
NIS	4	020010	CSU	SWITCH IS IN THE MAINTENANCE MODE	1
NIS	7	020100	CSU	SELECT COMMAND IS COMPLETED	1
NIS	8	020200		CB FF 1	1
NIS	9	020400		CB FF 2	1
NIS	0	021000		PARITY FF	
NMA	M	430000		M AND A TO A	2
NMB	M	630000		M AND B TO B	2
PIC	I	011000		PLACE I IN C	1
PRF			DSU	POSITION DISC STORAGE UNIT	
RAM	M	540000		M OR A TO M	2
RBD			MTS	READ BACKWARD DECIMAL	1+3
RBM	M	740000		M OR B TO M	2
RBS			MTS	READ BACKWARD BINARY	1+3
RMA	M	440000		M OR A TO A	2
RMB	M	640000		M OR B TO B	2
RRF			DSU	READ DSU	
RTB			MTS	READ TAPE BINARY	1+3
RTD			MTS	READ TAPE DECIMAL	1+3
RWD			MTS	REWIND	1+3
SAM	M		MACRO	SUBTRACT A FROM M	7
SBM	M		MACRO	SUBTRACT B FROM M	7
SBO	M	350000		SUBTRACT ONE FROM M	3
SCN	I	030000	BBC	SCAN BIT BUFFER UNITS	1+3N
SEL		024100	CSU	SELECT PERIPHERAL CONTROLLER	1+3
SL1	FROM, TO	040000		SHIFT LEFT 1	1
SL2	FROM, TO		MACRO	SHIFT LEFT 2	2
SL3	FROM, TO		MACRO	SHIFT LEFT 3	3
SL4	FROM, TO		MACRO	SHIFT LEFT 4	4
SL5	FROM, TO		MACRO	SHIFT LEFT 5	5
SL6	FROM, TO	044000		SHIFT LEFT 6	1
SL7	FROM, TO		MACRO	SHIFT LEFT 7	2
SL8	FROM, TO		MACRO	SHIFT LEFT 8	3
SL9	FROM, TO		MACRO	SHIFT LEFT 9	4
SLD	I		MACRO	SHIFT A,B LEFT I BITS	2 I

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
SLS	FROM, TO	070000		SHIFT LEFT SPECIAL	1
SLT			HSP	SLEW PAPER TO TAPE PUNCH	1+3
SLW			HSP	SLEWING OF PAPER	1+3
SMA	M		MACRO	SUBTRACT M FROM A	4
SMB	M		MACRO	SUBTRACT M FROM B	4
SMD	M		MACRO	SUBTRACT M, M+1 FROM A,B	7
SR1	FROM, TO	042000		SHIFT RIGHT 1	1
SR2	FROM, TO		MACRO	SHIFT RIGHT 2	2
SR3	FROM, TO		MACRO	SHIFT RIGHT 3	3
SR4	FROM, TO		MACRO	SHIFT RIGHT 4	4
SR5	FROM, TO		MACRO	SHIFT RIGHT 5	5
SR6	FROM, TO	046000		SHIFT RIGHT 6	1
SR7	FROM, TO		MACRO	SHIFT RIGHT 7	2
SR8	FROM, TO		MACRO	SHIFT RIGHT 8	3
SR9	FROM, TO		MACRO	SHIFT RIGHT 9	4
SRD	I		MACRO	SHIFT A,B RIGHT I BITS	2 I
SRS	FROM, TO	072000		CIRCULATE RIGHT SPECIAL	1
STA	M	500000		STORE A IN M	2
STB	M	700000		STORE B IN M	2
STC	M	300000		STORE C IN M	2
STD	M	310000		STORE DOUBLE -- A,B IN M,M+1	3
STF	M	360000		STORE SPECIAL FLIP-FLOPS	2
STZ	M	320000		STORE ZERO IN M	2
TRA	FROM, TO	060000		TRANSFER	1
TRC	FROM, TO	062000		TRANSFER COMPLEMENT	1
WEF			MTS	WRITE END OF FILE	1+3
WFL			HSP	WRITE FORMAT LINE	1+3
WPL			HSP	WRITE PRINT LINE	1+3
WRF			DSU	WRITE DSU	
WTB			MTS	WRITE TAPE BINARY	1+3
WTD			MTS	WRITE TAPE DECIMAL	1+3
XAM	M	550000		M XOR A TO M	2
XAZ	M	570000		M XOR A TO Z ONLY	2
XBM	M	750000		M XOR B TO M	2
XBZ	M	770000		M XOR B TO Z ONLY	2
XCZ	I	014000		I XOR C TO Z ONLY	1
XMA	M	450000		M XOR A TO A	2
XMB	M	650000		M XOR B TO B	2

## OCTAL LISTING

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
GROUP	MNEMONIC			INTERNAL INSTRUCTIONS	
GROUP	MNEMONIC		BBU	BIT BUFFER UNIT INSTRUCTIONS	
GROUP	MNEMONIC		BSU	BUFFER SELECTOR UNIT INSTRUCTIONS	
GROUP	MNEMONIC		CBU	CHARACTER BUFFER UNIT INSTRUCTIONS	
GROUP	MNEMONIC		CIU	COMPUTER INTERFACE UNIT INSTRUCTIONS	
GROUP	MNEMONIC		CSU	CONTROLLER SELECTOR UNIT INSTRUCTIONS	
GROUP	MNEMONIC		HSP	HIGH SPEED PRINTER INSTRUCTIONS	
GROUP	MNEMONIC		MACRO	GENERAL ASSEMBLY PROGRAM MACRO INSTRUCTIONS	
GROUP	MNEMONIC		MTS	MAGNETIC TAPE SYSTEM INSTRUCTIONS	
GROUP	MNEMONIC		PTR	PAPER TAPE READER INSTRUCTIONS	
GROUP	MNEMONIC		WBU	WORD BUFFER UNIT INSTRUCTIONS	
HLT	I	000000		CONDITIONAL HALT	1
AIC	I	010000		ADD I TO C	1
PIC	I	011000		PLACE I IN C	1
NCZ	I	012000		I AND C TO Z ONLY	1
XCZ	I	014000		I XOR C TO Z ONLY	1
NIS	I	020000		I AND INTERNAL STATUS LINES TO Z ONLY	1
NIS	1	020001		CHARACTER PARITY OUTPUT	1
NIS	2	020002		WORD PARITY OUTPUT	1
NIS	3	020004		CB FF 2 AND WORD PARITY OUTPUT ARE EQUAL	1
NIS	4	020010		SWITCH IS IN THE MANUAL MODE	1
NIS	7	020100	CSU	SELECT COMMAND IS COMPLETED	1
NIS	8	020200		CB FF 1	1
NIS	9	020400		CB FF 2	1
NIS	0	021000		PARITY FF	1
NES	I	022000	BSU	I AND EXTERNAL STATUS LINES TO Z ONLY	1
NES	1	022001	BBC	RC FLAG SET (BUFFER CONTAINS A NEW BIT)	1
NES	1	022001	CBC	RC FLAG SET (BUFFER CONTAINS A NEW CHAR.)	1
NES	1	022001	CIU	FLAG SET (BUFFER READY)	1
NES	1	022001	PTR	READ FLAG SET (BUFFER CONTAINS A NEW CHAR.)	1
NES	1	022001	WBC	RC FLAG SET (BUFFER CONTAINS A NEW WORD)	1
NES	2	022002	BBC	TX FLAG SET (BUFFER READY FOR A NEW BIT)	1
NES	2	022002	CBC	TX FLAG SET (BUFFER READY FOR A NEW CHAR.)	1
NES	2	022002	WBC	WBC FLAG SET (BUFFER READY FOR A NEW WORD)	1
NES	3	022004	CBC	CALL IN PROGRESS	1
NES	4	022010	CBC	REQUEST ANSWER	1
NES	5	022020	CBC	DATA MODE	1

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
NES	6	022040	CBC	CARRIER ON	1
NES	7	022100	CBC	CLEAR TO SEND	1
DIF	I	024000		DRIVE INTERNAL FUNCTION	1
DIF	1	024001		RESET CB 1 AND 2, AND RESET PARITY BIT FF	1
DIF	2	024002		RESET BUZZER FLIP-FLOP	1
DIF	3	024004		SET BUZZER FLIP-FLOP )	1
DIF	4	024010		INITIATE HARDWARE LOAD PROCESS	1
DIF	7	024100	CSU	SELECT PERIPHERAL CONTROLLER	1+3
SEL		024100	CSU	SELECT PERIPHERAL CONTROLLER	1+3
DIF	8	024200		SET CONTROL BIT FLIP-FLOP 1	1
DIF	9	024400		SET CONTROL BIT FLIP-FLOP 2	1
DIF	0	025000		SET THE PARITY BIT FLIP-FLOP	1
DEF	I	026000	BSU	DRIVE EXTERNAL FUNCTION	1
DEF	1	026001	BBC	RESET RECEIVE FLAG AND DATA BUFFER	1
DEF	1	026001	CBC	RESET RECEIVE FLAG AND DATA BUFFER	1
DEF	1	026001	CIU	RESET FLAG AND BUFFER, SET RECEIVE MODE	1
DEF	1	026001	PTR	RESET FLAG AND READ NEXT CHARACTER	1
DEF	1	026001	WBC	RESET RECEIVE FLAG AND DATA BUFFER	1
DEF	2	026002	BBC	RESET TRANSMIT FLAG AND DATA BUFFER	1
DEF	2	026002	CBC	RESET TRANSMIT FLAG AND DATA BUFFER	1
DEF	2	026002	CIU	RESET FLAG AND BUFFER, SET TRANSMIT MODE	1
DEF	2	026002	WBC	RESET TRANSMIT FLAG AND DATA BUFFER	1
DEF	3	026004	BBC	TURN CARRIER OFF	1
DEF	3	026004	CBC	TURN CARRIER OFF	1
DEF	3	026004	WBC	TURN CARRIER OFF	1
DEF	4	026010	BBC	TURN CARRIER ON	1
DEF	4	026010	CBC	TURN CARRIER ON	1
DEF	4	026010	WBC	TURN CARRIER ON	1
DEF	5	026020	BBC	RESET RECEIVE CLOCK	1
DEF	6	026040	BBC	SET ECHO MODE	1

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
DEF	7	026100	BBC	RESET ECHO MODE	1
DEF	9	026400	CBC	ANSWER INCOMING CALL	1
DEF	9	026400	CIU	AUTOMATIC PRIORITY INTERRUPT THE 225	1
DEF	0	027000	CBC	DISCONNECT CALL	1
DEF	0	027000	CIU	RESET THE ADDRESS REGISTER	1
SCN	I	030000	BBC	SCAN BIT BUFFER UNITS	1+3N
CSR	I	032000	CSU	CONTROLLER STATUS REQUEST	3-10
SL1	FROM, TO	040000		SHIFT LEFT 1	1
SR1	FROM, TO	042000		SHIFT RIGHT 1	1
SL6	FROM, TO	044000		SHIFT LEFT 6	1
SR6	FROM, TO	046000		SHIFT RIGHT 6	1
CL1	FROM, TO	050000		CIRCULATE LEFT 1	1
CR1	FROM, TO	052000		CIRCULATE RIGHT 1	1
CL6	FROM, TO	054000		CIRCULATE LEFT 6	1
CR6	FROM, TO	056000		CIRCULATE RIGHT 6	1
TRA	FROM, TO	060000		TRANSFER	1
TRC	FROM, TO	062000		TRANSFER COMPLEMENT	1
BC0	FROM, TO	064000		BIT CHANGE ZERO (8-LEVEL LINE TO 6-BIT)	1
BC1	FROM, TO	066000		BIT CHANGE ONE (6-BIT TO 8-LEVEL LINE)	1
SLS	FROM, TO	070000		SHIFT LEFT SPECIAL	1
SRS	FROM, TO	072000		CIRCULATE RIGHT SPECIAL	1
BRU	M	100000		BRANCH UNCONDITIONALLY	1
BRS	M	110000		BRANCH TO SUBROUTINE	3
BZE	M	120000		BRANCH IF ZERO FF IS ZERO	1
BNZ	M	130000		BRANCH IF ZERO FF IS NON-ZERO	1
BPL	M	140000		BRANCH IF PLUS FF IS PLUS	1
BMI	M	150000		BRANCH IF PLUS FF IS MINUS	1
BEV	M	160000		BRANCH IF EVEN FF IS EVEN	1
BOD	M	170000		BRANCH IF EVEN FF IS ODD	1
LDC	M	200000		LOAD C FROM M	2
LDD	M	210000		LOAD DOUBLE -- A,B FROM M,M+1	3
LDZ	M	220000		LOAD Z (BRANCH FLIP-FLOPS) FROM M	2
LDQ	M	230000		LOAD Q FROM M	2
AMD	M	240000		ADD DOUBLE -- ADD M,M+1 TO A,B	3
LDT	M	250000	BSU	LOAD T (TRANSMIT DATA DRIVERS) FROM M	2
LDF	M	260000		LOAD SPECIAL FLIP-FLOPS FROM M	2

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W,T
STC	M	300000		STORE C IN M	2
STD	M	310000		STORE DOUBLE -- A,B IN M,M+1	3
STZ	M	320000		STORE ZERO IN M	2
CMM	M	330000		STORE M-NOT IN M (COMPLEMENT M TO M)	2
ADO	M	340000		ADD ONE TO M	3
SBO	M	350000		SUBTRACT ONE FROM M	3
STF	M	360000		STORE SPECIAL FLIP-FLOPS	2
LDA	M	400000		LOAD A FROM M	2
CMA	M	410000		LOAD A WITH M-NOT (COMPLEMENT M TO A)	2
AMA	M	420000		ADD M TO A	2
NMA	M	430000		M AND A TO A	2
RMA	M	440000		M OR A TO A	2
XMA	M	450000		M XOR A TO A	2
AAZ	M	460000		ADD A,M - RESULT TO Z DRIVERS	2
STA	M	500000		STORE A IN M	2
CAM	M	510000		STORE A-NOT IN M (COMPLEMENT A TO M)	2
AAM	M	520000		ADD A TO M	3
NAM	M	530000		M AND A TO M	2
RAM	M	540000		M OR A TO M	2
XAM	M	550000		M XOR A TO M	2
NAZ	M	560000		M AND A TO Z ONLY	2
XAZ	M	570000		M XOR A TO Z ONLY	2
LDB	M	600000		LOAD B FROM M	2
CMB	M	610000		LOAD B WITH M-NOT (COMPLEMENT M TO B)	2
AMB	M	620000		ADD M TO B	2
NMB	M	630000		M AND B TO B	2
RMB	M	640000		M OR B TO B	2
XMB	M	650000		M XOR B TO B	2
ABZ	M	660000		ADD B,M - RESULT TO Z DRIVERS	2
STB	M	700000		STORE B IN M	2
CBM	M	710000		STORE B-NOT IN M (COMPLEMENT B TO M)	2
ABM	M	720000		ADD B TO M	3
NBM	M	730000		M AND B TO M	2
RBM	M	740000		M OR B TO M	2
XBM	M	750000		M XOR B TO M	2
NBZ	M	760000		M AND B TO Z ONLY	2
XBZ	M	770000		M XOR B TO Z ONLY	2
GROUP	MNEMONIC		HSP	HIGH SPEED PRINTER INSTRUCTIONS	
SLT			HSP	SLEW PAPER TO TAPE PUNCH	1+3
SLW			HSP	SLEWING OF PAPER	1+3
WFL			HSP	WRITE FORMAT LINE	1+3
WPL			HSP	WRITE PRINT LINE	1+3
GROUP	MNEMONIC		DSU	DISC STORAGE UNIT INSTRUCTIONS	
PRF			DSU	POSITION DISC STORAGE UNIT	
RRF			DSU	READ DSU	
WRF			DSU	WRITE DSU	

OPR	OPERAND	OCTAL	GROUP	DESCRIPTION	W.T.
GROUP	MNEMONIC		MTS	MAGNETIC TAPE SYSTEM INSTRUCTIONS	
BKW			MTS	BACKSPACE AND POSITION WRITE HEAD	1+3
RBD			MTS	READ BACKWARD DECIMAL	1+3
RBS			MTS	READ BACKWARD BINARY	1+3
RTB			MTS	READ TAPE BINARY	1+3
RTD			MTS	READ TAPE DECIMAL	1+3
RWD			MTS	REWIND	1+3
WEF			MTS	WRITE END OF FILE	1+3
WTB			MTS	WRITE TAPE BINARY	1+3
WTD			MTS	WRITE TAPE DECIMAL	1+3
GROUP	MNEMONIC		MACRO	GENERAL ASSEMBLY PROGRAM MACRO INSTRUCTIONS	
CL2	FROM, TO		MACRO	CIRCULATE LEFT 2	2
CL3	FROM, TO		MACRO	CIRCULATE LEFT 3	3
CL4	FROM, TO		MACRO	CIRCULATE LEFT 4	3
CL5	FROM, TO		MACRO	CIRCULATE LEFT 5	2
CL7	FROM, TO		MACRO	CIRCULATE LEFT 7	2
CL8	FROM, TO		MACRO	CIRCULATE LEFT 8	3
CL9	FROM, TO		MACRO	CIRCULATE LEFT 9	4
CR2	FROM, TO		MACRO	CIRCULATE RIGHT 2	2
CR3	FROM, TO		MACRO	CIRCULATE RIGHT 3	3
CR4	FROM, TO		MACRO	CIRCULATE RIGHT 4	3
CR5	FROM, TO		MACRO	CIRCULATE RIGHT 5	2
CR7	FROM, TO		MACRO	CIRCULATE RIGHT 7	2
CR8	FROM, TO		MACRO	CIRCULATE RIGHT 8	3
CR9	FROM, TO		MACRO	CIRCULATE RIGHT 9	4
SAM	M		MACRO	SUBTRACT A FROM M	7
SBM	M		MACRO	SUBTRACT B FROM M	7
SL2	FROM, TO		MACRO	SHIFT LEFT 2	2
SL3	FROM, TO		MACRO	SHIFT LEFT 3	3
SL4	FROM, TO		MACRO	SHIFT LEFT 4	4
SL5	FROM, TO		MACRO	SHIFT LEFT 5	5
SL7	FROM, TO		MACRO	SHIFT LEFT 7	2
SL8	FROM, TO		MACRO	SHIFT LEFT 8	3
SL9	FROM, TO		MACRO	SHIFT LEFT 9	4
SLD	I		MACRO	SHIFT A,B LEFT I BITS	2
SMA	M		MACRO	SUBTRACT M FROM A	4
SMB	M		MACRO	SUBTRACT M FROM B	4
SMD	M		MACRO	SUBTRACT M,M+1 FROM A,B	7
SR2	FROM, TO		MACRO	SHIFT RIGHT 2	2
SR3	FROM, TO		MACRO	SHIFT RIGHT 3	3
SR4	FROM, TO		MACRO	SHIFT RIGHT 4	4
SR5	FROM, TO		MACRO	SHIFT RIGHT 5	5
SR7	FROM, TO		MACRO	SHIFT RIGHT 7	2
SR8	FROM, TO		MACRO	SHIFT RIGHT 8	3
SR9	FROM, TO		MACRO	SHIFT RIGHT 9	4
SRD	I		MACRO	SHIFT A,B RIGHT I BITS	2

## INDEX TO INSTRUCTIONS

LOAD INSTRUCTIONS			PAGE	LOGICAL OPERATION INSTRUCTIONS			PAGE
40	LDA	M	II-3	43	NMA	M	II-8
60	LDB	M	3	53	NAM	M	8
20	LDC	M	3	63	NMB	M	8
21	LDD	M	3	73	NBM	M	8
26	LDF	M	24	56	NAZ	M	8
23	LDQ	M	3	76	NBZ	M	9
25	LDT	M	25	012	NCZ	I	9
22	LDZ	M	3	020	NIS	I	22
41	CMA	M	3	022	NES	I	25
61	CMB	M	4	44	RMA	M	9
011	PIC	I	4	54	RAM	M	9
STORE INSTRUCTIONS				64	RMB	M	9
50	STA	M	5	74	RBM	M	10
70	STB	M	5	45	XMA	M	10
30	STC	M	5	55	XAM	M	10
31	STD	M	5	65	XMB	M	10
36	STF	M	24	75	XBM	M	10
32	STZ	M	5	57	XAZ	M	11
51	CAM	M	5	77	XBZ	M	11
71	CBM	M	5	014	XCZ	I	11
33	CMM	M	5	REGISTER TRANSFER INSTRUCTIONS			
ARITHMETIC INSTRUCTIONS				060	TRA	FROM, TO	13
42	AMA	M	6	062	TRC	FROM, TO	13
52	AAM	M	6	040	SL1	FROM, TO	13
46	AAZ	M	7	042	SR1	FROM, TO	13
62	AMB	M	6	044	SL6	FROM, TO	13
72	ABM	M	6	046	SR6	FROM, TO	13
66	ABZ	M	7	070	SLS	FROM, TO	14
37	AMD	M	6	072	SRS	FROM, TO	14
34	ADO	M	6	050	CL1	FROM, TO	14
35	SBO	M	7	052	CR1	FROM, TO	14
010	AIC	I	6	054	CL6	FROM, TO	14
BRANCH INSTRUCTIONS				056	CR6	FROM, TO	14
10	BRU	M	16	064	BC0	FROM, TO	15
11	BRS	M	16	066	BC1	FROM, TO	15
12	BZE	M	16	SPECIAL INSTRUCTIONS			
13	BNZ	M	16	00	HLT		24
14	BPL	M	16	024	DIF	I	22
15	BMI	M	16	026	DEF	I	25
16	BEV	M	16	030	SCN	I	25
17	BOD	M	17				



## CONTENTS

	Page
<b>GENERAL DESCRIPTION . . . . .</b>	<b>D-1</b>
Data Transfer . . . . .	D-2
Performance Summary . . . . .	D-3
<b>STATUS CONDITIONS . . . . .</b>	<b>D-3</b>
<b>ERROR CONDITIONS . . . . .</b>	<b>D-4</b>
Parity Error . . . . .	D-4
Data Transfer Timing Error . . . . .	D-5
Terminate . . . . .	D-6
Special Interrupt . . . . .	D-6
Functional Message . . . . .	D-6
<b>INSTRUCTION REPERTOIRE . . . . .</b>	<b>D-7</b>
Drive External Function Instructions . . . . .	D-7
External Status Lines . . . . .	D-8
Register Transfer Instruction . . . . .	D-9
<b>PROGRAMMING CONSIDERATIONS . . . . .</b>	<b>D-10</b>
Output Signals to the DATANET-30 . . . . .	D-10
Instruction Sequence of Operations . . . . .	D-11
<b>PROGRAMMING CONVENTIONS . . . . .</b>	<b>D-18</b>
DATANET-30 Conventions . . . . .	D-18
External Computer Conventions . . . . .	D-18
<b>PROGRAMMING EXAMPLES . . . . .</b>	<b>D-19</b>
Receive Example . . . . .	D-19
Transmit Example . . . . .	D-19

## ILLUSTRATIONS

Figure	Page
D- 1. CIU-931 Connecting a DATANET-30 with an External Computer . . . . .	D-1
D- 2. DATANET-30 with Two CIU-931's . . . . .	D-2
D- 3. Data Transfer Through a CIU-931 . . . . .	D-2
D- 4. Permissible Status Returns to the DATANET-30 . . . . .	D-4
D- 5. Transfer Sequence from External Computer . . . . .	D-12
D- 6. Data-Not-Accepted Sequence (External Computer to DATANET-30) . . . . .	D-13
D- 7. Transfer Sequence from DATANET-30 . . . . .	D-15
D- 8. Data-Not-Accepted Sequence (DATANET-30 to External Computer) . . . . .	D-16
D- 9. Error Condition Sequence . . . . .	D-17
D-10. Sample Receive Program . . . . .	D-20
D-11. Sample Transmit Program . . . . .	D-21

## APPENDIX D

### COMPUTER INTERFACE UNIT (CIU931)

#### GENERAL DESCRIPTION

The CIU-931 Computer Interface Unit provides the means for connecting the DATANET-30 and a GE-400 or -600 Series computer (Figure D-1).

The computer interface unit (CIU) connects into the buffer selector of the DATANET-30 and into one standard input/output channel of the other computer. The buffer selector address of the CIU is specified by the buffer selector address channel for the CIU. Since the CIU connects the DATANET-30 with either a GE-400 or -600 Series computer, for the purposes here, both of these computers will be referred to as the "external computer."

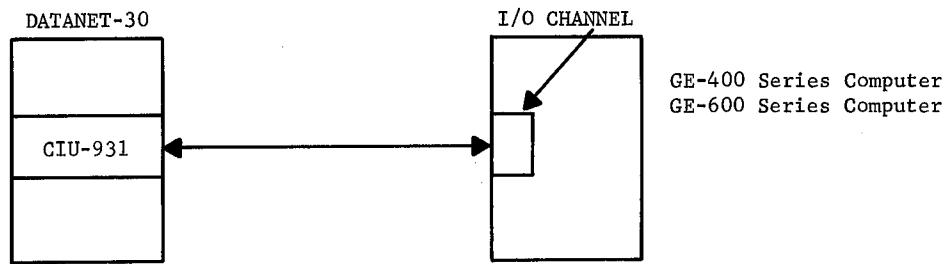


Figure D-1. CIU-931 Connecting a DATANET-30 with an External Computer

Data is transferred in both directions but only in one direction at a time. Control of the direction of data flow is established and maintained by either the external computer or the DATANET-30. Should both the DATANET-30 and the external computer desire control at exactly the same time, the external computer is given priority.

As shown in Figure D-2 more than one CIU can be installed in a DATANET-30. Each CIU occupies two modules. There is no restriction on the number used, other than the physical space occupied.

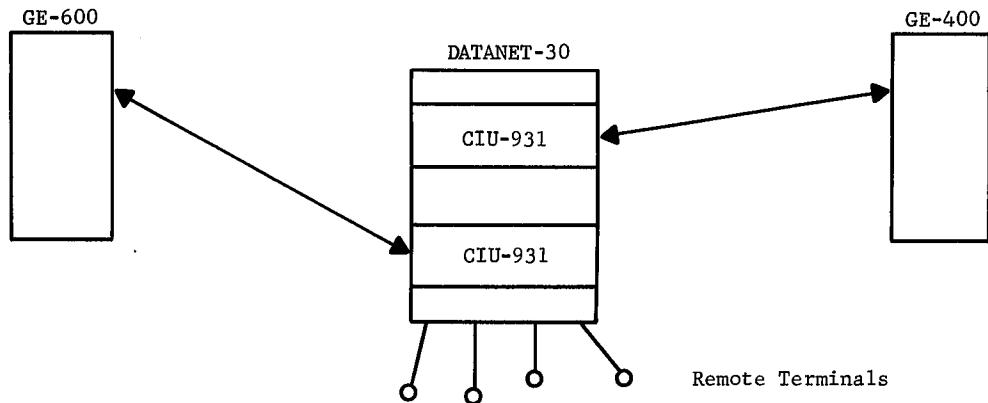


Figure D-2. DATANET-30 with Two CIU-931's

### Data Transfer

The CIU has an 18-bit buffer. Data is transferred one character at a time to or from the external computer and one DATANET-30 word at a time to or from the DATANET-30. The CIU contains the necessary shift circuitry to accumulate characters into words and separate sequential characters, depending on the direction of data transfer. Data transfer is illustrated in Figure D-3.

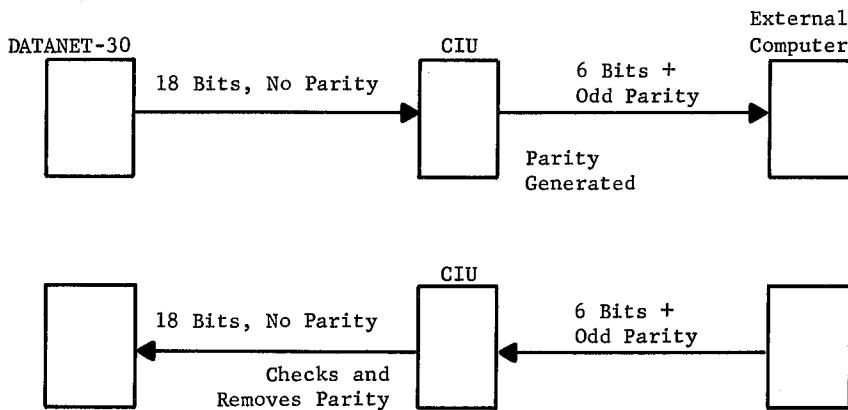


Figure D-3. Data Transfer Through a CIU-931

**DATANET-30 TO THE COMPUTER INTERFACE UNIT.** When the CIU has received a word from the DATANET-30, a signal for the external computer to read is generated. The external computer initiates the transfer of one character at a time from the CIU. When the three characters have been transferred, the CIU notifies the DATANET-30 that another word can be transferred. The DATANET-30 program then transfers another word to the CIU. This process continues until the DATANET-30 or external computer sends an End Data Transfer signal to the CIU.

**EXTERNAL COMPUTER TO THE COMPUTER INTERFACE UNIT.** When the CIU has received three characters from the external computer, a signal for the DATANET-30 to transfer the three characters is sent to the DATANET-30. The DATANET-30 program must detect this signal and

transfer the data from the CIU. When the data has been transferred from the CIU, the external computer can transfer three more characters to the CIU. This process continues until the external computer or DATANET-30 sends an End Data Transfer signal to the CIU.

**DATA FLOW THROUGH THE COMPUTER INTERFACE UNIT.** When data is transferred from the CIU to the external computer, the most significant character of the DATANET-30 word is transferred first and the least significant last.

Characters received by the CIU from the external computer are assembled into a three-character word. The first character is transferred to the most significant part of the DATANET-30 word and the third character is transferred as the least significant.

If an End Data Transfer signal is received from the external computer before a full three-character word is shifted into the CIU, the characters in the data register are shifted to the most significant positions. Zeros are inserted by the CIU into the unfilled character positions before the word is transferred to the DATANET-30.

### **Performance Summary**

The following lists the performance characteristics of the CIU-931:

Data medium--Contents of memory.

Speed--Up to 39,000 character per second, depending on the DATANET-30 program and the type of a standard channel in the external computer.

Data format--Oriented in accordance with the program.

Operational mode--On line under program control.

Checks--Parity check performed by the CIU on each character during data transfer from the external computer to the CIU and parity generated on each character during data transfer from the CIU to the external computer.

Controls and indicators--None in the usual sense, except for manual reset button on the control panel of the DATANET-30. All other controls and indicators are by program only.

### **STATUS CONDITIONS**

A total of six status indications may be presented to the DATANET-30. Status is transmitted on the four external status lines, numbers 3, 4, 5 and 6. Status codes are defined in Figure D-4.

<u>CODE</u>	<u>STATUS</u>	<u>MEANING</u>
NES 6543		
0000	Ready	The CIU is ready to receive a Write Initiate instruction.

<u>CODE</u>	<u>STATUS</u>	<u>MEANING</u>
1000	Busy	The previous command has been accepted. No more commands may be transmitted until termination of this command.
0010	Command Rejected	This status is caused only by an improper sequence of instructions from the DATANET-30. The DATANET-30 is not notified if a command by the external computer is rejected by the CIU.
0100	Intermediate	The CIU is waiting for a Read Response command from the computer to which Intermediate status is presented.
0001	Parity Error	This indicates that a parity error occurred.
0011	Time-Out	This indicates a time-out occurred.

Figure D-4. Permissible Status Returns to the DATANET-30.

## ERROR CONDITIONS

### Parity Error

A parity error occurs only during data transfer from the external computer to the CIU. A Time-Out or Parity Error status and a Terminate are presented to both the external computer and the DATANET-30. The status must be detected by each.

Detection of a parity error on data from the external computer causes the data transfer to cease. The data flag (NES1) is not set. Data included in that transfer to the CIU is lost. The DATANET-30 program needs to determine Time-Out or Parity Error status because the external computer will try to send the same data again, starting with the functional message. The Time-Out and Parity Error status and Terminate must be reset by the DATANET-30 program.

In the event of a parity error, the following conditions prevail:

1. The external computer must start entire transmission over again.
2. The external computer must start with a Write Initiate instruction.
3. The status remains in the CIU until the external computer issues a Write Initiate instruction. Prior to the Write Initiate, the following status is presented to the DATANET-30:
  - a. Time-Out or Parity Error
  - b. Terminate
  - c. External status line 1 (NES1) not set

After the Write Initiate, the following status is presented to the DATANET-30:

- a. Time-Out or Parity Error
- b. Terminate
- c. Special Interrupt
- d. NES1 not set

The DATANET-30 must interrogate for over-all status and operating conditions. Time-Out or Parity Error and Terminate must be reset. The Special Interrupt indicates a Write Initiate instruction from the external computer. The above combination indicates that the preceding transmission will probably be sent again.

The indication to the DATANET-30 that the entire data transfer has taken place is that a Terminate has been received with a Ready status and/or that the total number of words received agrees with the functional message.

#### **Data Transfer Timing Error (Time-Out)**

The data transfer timer is used as an error condition check on the operation of both programs. The timer is set for 100 milliseconds and is started from zero after each of the following:

1. A Write Initiate (DEF 8) instruction from the DATANET-30.
2. A Write Initiate (WDN) instruction from the external computer.
3. A Read Response (DEF 9) instruction from the DATANET-30.
4. A Read Response (RDN) instruction from the external computer.
5. Data transfer to or from the CIU.

The timer is running upon the completion of the functional message. Thus the DATANET-30 and the external computer must analyze the functional message and reply within 100 milliseconds after the last part of the functional message is transferred from the CIU.

In the event that the timer reaches 100 milliseconds (time-out) without either a Read Response or data transfer taking place, an error condition is assumed; and a Time-Out status with Terminate is presented to both the DATANET-30 and the external computer. Special Interrupt to the DATANET-30 is reset if the DATANET-30 does not answer a Write Initiate from the external computer.

Should a time-out occur, the data transfer from the external computer ceases and the data flag (NES1) is reset. The entire data transfer must be sent again regardless of the direction of data transfer. The same status conditions prevail as for a parity error.

The DATANET-30 must detect the over-all status. Time-Out status and Terminate must be reset. If the direction of transfer was from the DATANET-30, the transfer sequence must start with a Write Initiate instruction, functional message, etc. The status to the external computer must be reset by the external computer. The timer starts again when the DATANET-30 issues the Write Initiate.

If the direction of transfer was to the DATANET-30, the status in the CIU must be reset; and the DATANET-30 must wait for the external computer to start the transfer sequence.

### **Terminate**

The Terminate signal normally indicates the termination of data transfer. The Terminate signal is transmitted to both the external computer and the DATANET-30 along with the current status. The Terminate signal results normally from an End Data Transfer instruction to the CIU. The Terminate signal must be reset by the DATANET-30 program. A Terminate signal is also generated because of a time-out or parity error.

### **Special Interrupt**

The Special Interrupt signal indicates to the receiving program that a valid Write Initiate has been accepted by the CIU. The CIU is now waiting for a Read Response from the program receiving the Special Interrupt.

### **Functional Message**

The functional message consists of four DATANET-30 words. The CIU circuitry detects the transfer of the first four words (12 characters to the CIU) and temporarily halts data transfer.

The contents of the 12 characters of the functional message can be in any format or sequence and contain whatever information is desired by the two operating programs. The functional message can contain the number of words to be transferred, the type of data, what is to be done with the data (store on DSU, punch on cards, request a reply), etc. Depending on the system operation, the first 12 characters can also be the first part of a message--that is, a header plus part of the text. In short, the first 12 characters transferred can be anything agreed upon by the two operating programs.

## INSTRUCTION REPERTOIRE

The instructions are classified under buffer selector instructions for the DATANET-30. They are the Drive External Function (DEF), External Status Lines (NES), and Register Transfer (TRA) instructions. The rules for using these instructions are the same as those for other DATANET-30 buffer selector instructions.

### Drive External Function Instructions

DEF1	Reset Data Flag--Resets the data flag of the shift register and enables three more characters to be transferred into the CIU from the external computer.
DEF2	Reset Data Flag--Resets the data flag of the shift register before another word is transferred into the CIU from the DATANET-30.
DEF3,4	Not assigned.
DEF5	Reset Terminate--This instruction must be given each time Terminate is presented to the DATANET-30.
DEF6	Reset Status--Resets all status conditions to the DATANET-30 except Intermediate. The CIU goes to Ready status.
DEF7	Not assigned.
DEF8	Write Initiate--Establishes control in the CIU for direction of transfer of data. The CIU sends a Special Interrupt signal and Intermediate status to the external computer. In order to start data transfer, the external computer receiving the Special Interrupt must then issue a Read Response command.

A Write Initiate command is terminated by any of the following actions:

Either the DATANET-30 or the external computer generates an End Data Transfer signal.

When the Write Initiate is from the external computer and after data transfer from the external computer has started, the DATANET-30 issues a Write Initiate instruction (after the functional message).

The CIU is reflecting Busy status to either the DATANET-30 or external computer or both and no data has transferred for 100 milliseconds.

If the CIU receives simultaneous Write Initiate instructions, the external computer is given priority. The instruction issued by the DATANET-30 is not accepted and Intermediate status is indicated to the DATANET-30.

The external computer can also take priority if the external computer issues a Write Initiate after the DATANET-30 instead of a Read Response. The Write Initiate from the DATANET-30 is canceled, and Intermediate status with Special Interrupt is indicated to the

DATANET-30. A Read Response instruction must now be given by the DATANET-30. The Write Initiate from the external computer replaces and cancels the Write Initiate from the DATANET-30.

DEF9

Read Response--Enables transfer of data from the external computer to the CIU. This is issued by the DATANET-30 in reply to a Special Interrupt from the CIU and after the functional message.

A Read Response instruction is accepted by the CIU only when it is reflecting Intermediate status. Acceptance of the instruction changes status to Busy.

DEF0

End Data Transfer--Indicates to the CIU that data transfer has ceased. This instruction is issued by the program when the entire data transfer has been completed by the DATANET-30.

The End Data Transfer signal causes the CIU to:

1. Store the End Data Transfer signal until the external computer has taken all three characters before issuing the Terminate signal.
2. Send a Terminate signal to both the external computer and the DATANET-30.
3. Leave the CIU data flag (NES1) in a reset state.
4. Indicate Ready status to both.

### External Status Lines

NES1

The data flag--Set when the CIU is ready to transfer another word to or from the DATANET-30. Reset when a word is transferred to or from the DATANET-30. This condition is indicated for data flow in both directions.

NES2

Not used.

NES3

NES4

NES5

NES6

} CIU status lines--These four lines represent the code for the status conditions. Status is present at all times on these lines.

NES7

Not used.

NES8

Not used.

NES9

The Special Interrupt signal--Set when the CIU sends a Special Interrupt as the result of a Write Initiate command from the external computer to the CIU. The DATANET-30 program must test this line every basic program cycle or in time to prevent the 100-millisecond timer from running out.

NES0                    The Terminate signal--Set when the CIU sends a Terminate signal indicating end of data transfer.

### **Register Transfer Instruction**

TRA                    FROM, TO            Data is transferred from or to the CIU.

#### Examples:

TRA R,B                Data is transferred from the CIU to the B-register, and the CIU data flag (NES1) is reset.

TRA B,T                Data is transferred from the B-register to the CIU, and the CIU data flag (NES1) is reset.

## **PROGRAMMING CONSIDERATIONS**

### **Output Signals to the DATANET-30**

Output signals from the CIU to the DATANET-30 are generated in the CIU as shown in the following table.

<b>From External Computer to the CIU</b>	<b>From the CIU to DATANET-30</b>
End Data Transfer	Terminate with Ready status
Data transfer causing a parity error or a time-out error	Terminate with Parity Error status
Write Initiate	Special Interrupt with Intermediate status; Terminate with Intermediate status
Read Response	Data flag (NES1) set with Busy status
3 characters	Data flag (NES1) set with Busy status
Commands being Rejected because of Parity error Invalid code Improper sequence	No signal
Request status	No signal
Reset status	No signal

## **Instruction Sequence of Operations**

**EXAMPLE A.** In Figure D-5, assume that nothing is happening and the CIU is in Ready status. When the external computer issues a Write Initiate (WDN) instruction, the CIU sends a Special Interrupt signal to the DATANET-30. The CIU is now busy to the external computer and in Intermediate status to the DATANET-30. When the DATANET-30 issues a Read Response, the CIU goes busy to the DATANET-30; and data transfer occurs. The functional (control) message is transferred first. After the functional message has been transferred, the CIU halts data transfer and sends a Terminate and Intermediate status to the DATANET-30. There are two possibilities at this point:

1. The DATANET-30 can accept data (Figure D-5). A Read Response is issued by the DATANET-30, and data transfer continues until an End Data Transfer is sent from the DATANET-30 or the external computer. The CIU sends Terminate signals to both computers.
2. The DATANET-30 cannot accept data (Figure D-6). The DATANET-30 issues a Write Initiate to the CIU, and a Terminate is sent to the external computer. The external computer issues a Read Response to find out why the DATANET-30 cannot accept data. A functional message is transferred, the DATANET-30 issues End Data Transfer, and the CIU sends a Terminate to both computers ending the sequence. This ends the sequence started by the external computer.

If the DATANET-30 issues a WRI for the purpose of rejecting a functional message, the DATANET-30 must issue an End Data Transfer after the functional message for reject. If the DATANET-30 is then to transfer data, it must originate a new WRI sequence.

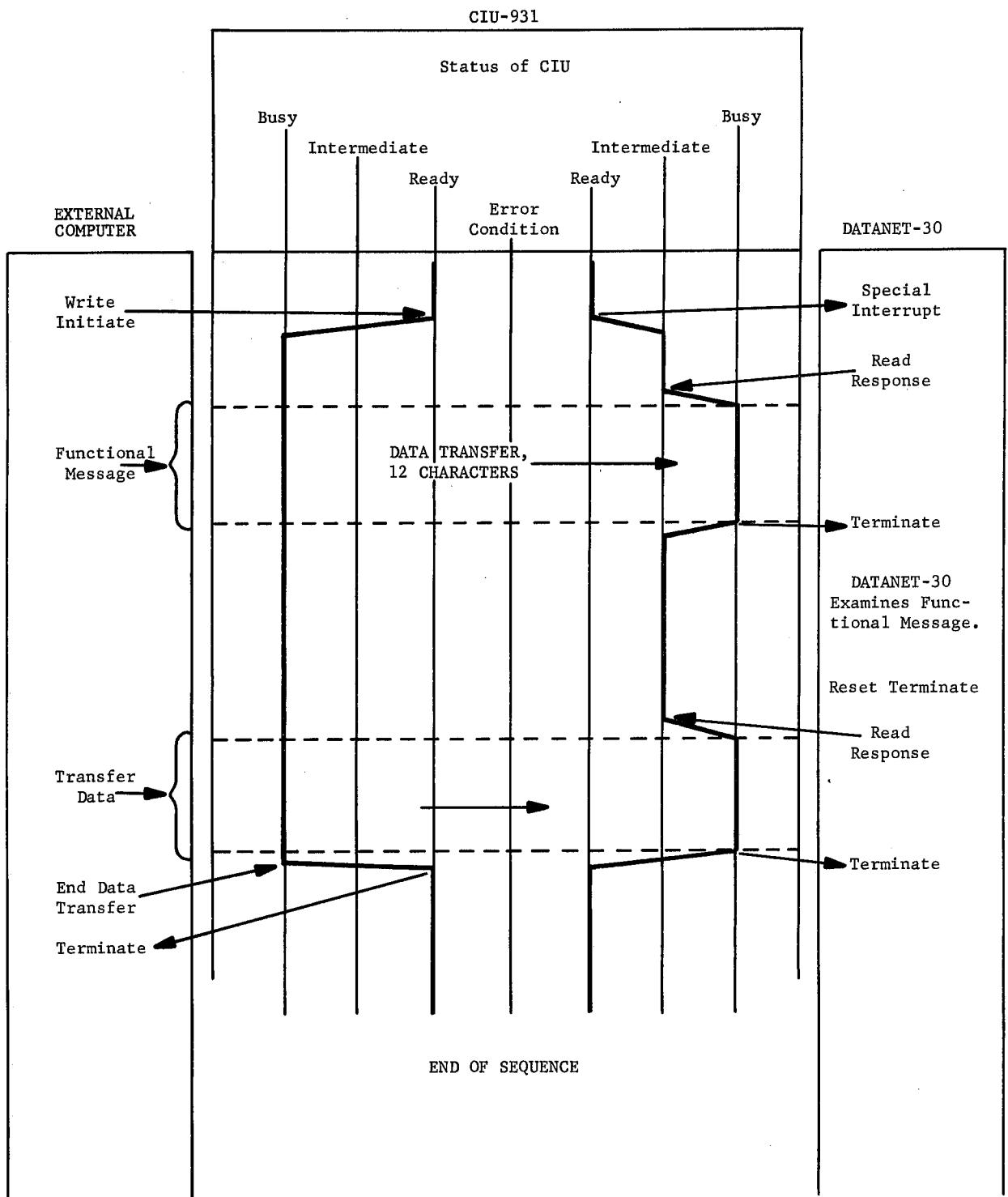


Figure D-5. Transfer Sequence from External Computer

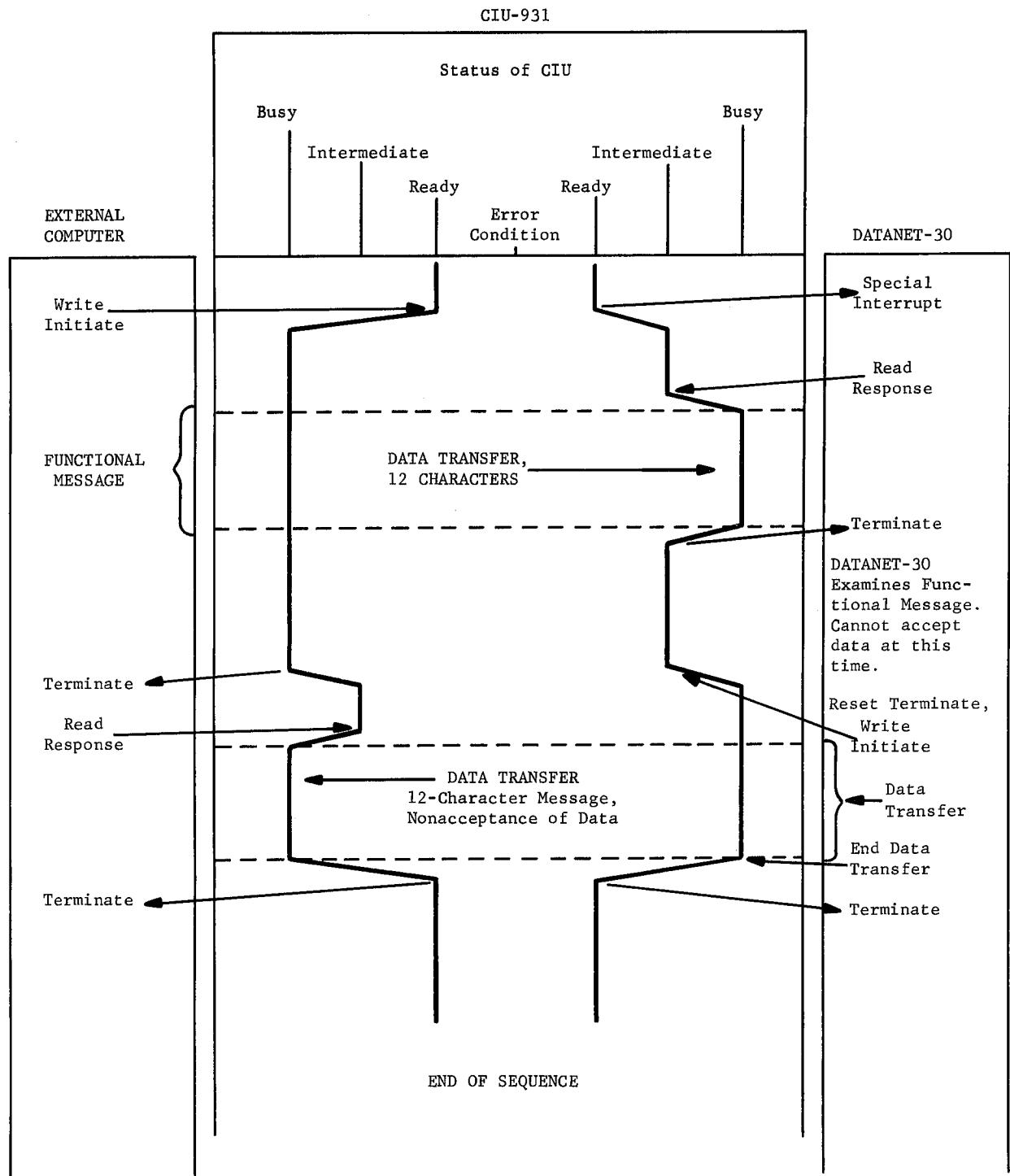


Figure D-6. Data-Not-Accepted Sequence (External Computer to DATANET-30)

EXAMPLE B. In Figure D-7, assume that nothing is happening and the CIU is in Ready status. When the DATANET-30 issues a Write Initiate instruction, the CIU sends a Special Interrupt signal to the external computer. The CIU is now busy to the DATANET-30 and in an Intermediate status to the external computer. When the external computer issues a Read Response, the CIU goes busy to the external computer, the data flag is set (NES1), and data transfer occurs.

The functional (control) message is first transferred. After the functional message has been transferred, the CIU halts data transfer and sends a Terminate and Intermediate status to the external computer while the external computer processes the functional message. The CIU is again in an Intermediate status. This status remains until the external computer is ready to accept more data. When the external computer issues a Read Response, the data flag is set (NES1); and data transfer can again occur. Data transfer continues until an End Data Transfer signal from the DATANET-30. The CIU sends a Terminate to both computers and indicates Ready status to both.

The external computer may also issue an End Data Transfer command if it has received characters in a multiple of 3. A Terminate signal with Ready status is presented to both computers.

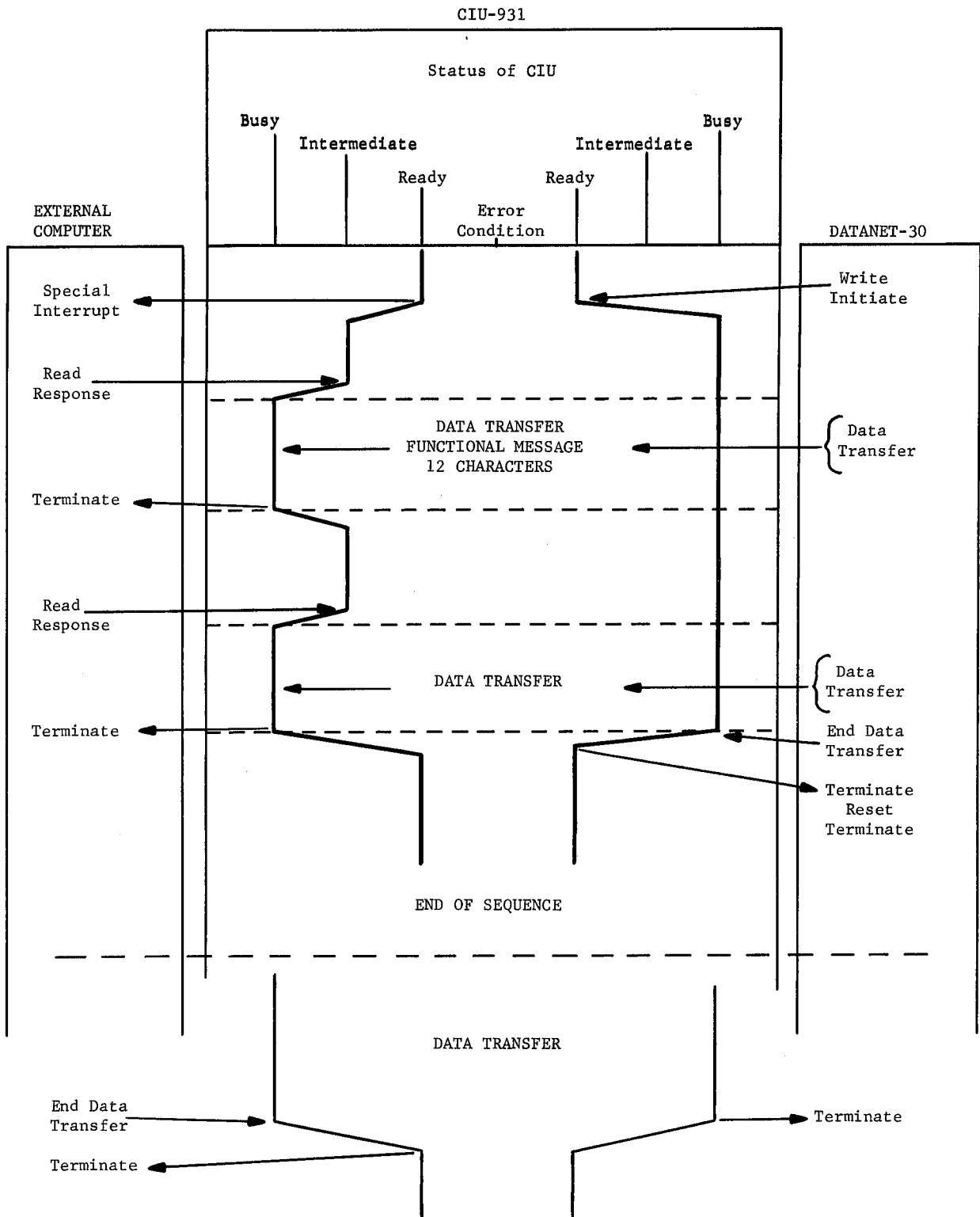


Figure D-7. Transfer Sequence from DATANET-30

**EXAMPLE C.** Figure D-8 shows a sequence when the external computer cannot accept data. The CIU is in a Ready status, and the DATANET-30 issues a Write Initiate instruction. The sequence for transfer of the functional (control) message is the same as that described in the previous section. Having transferred the functional message, the CIU halts the transfer of data.

At this point, the external computer cannot accept the data and issues a Write Initiate instruction. This causes a Terminate signal to be sent to the DATANET-30 and causes Intermediate status. DATANET-30 must now answer with a Read Response. The external computer sends a functional message indicating nonacceptance of data. The DATANET-30 must now issue an End Data Transfer and terminate the sequence. The DATANET-30 receives a Terminate signal. The DATANET-30 cannot attempt to do another Write Initiate (DEF8) until the CIU is again in Ready status and Terminate is reset.

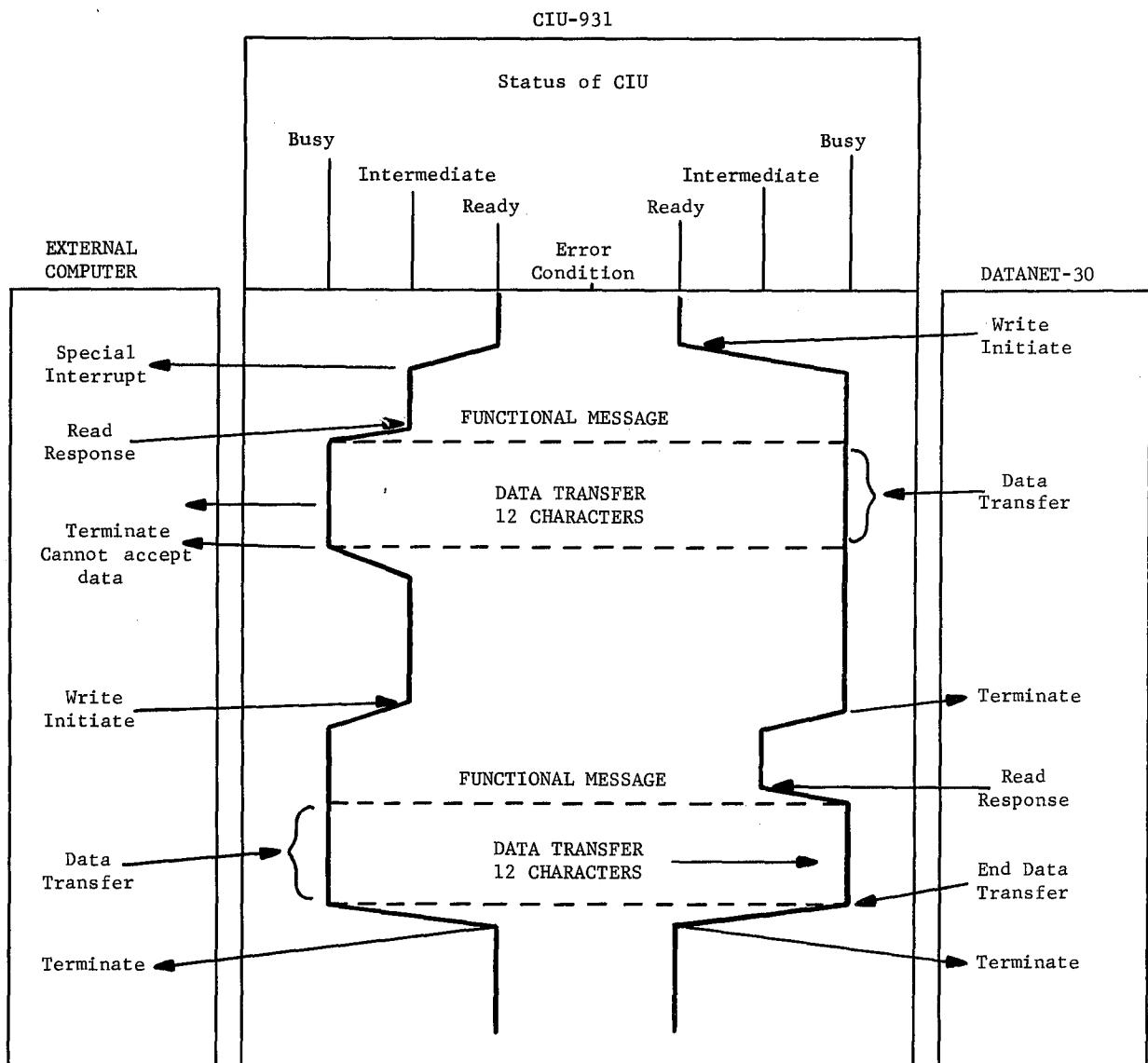


Figure D-8. Data-Not-Accepted Sequence (DATANET-30 to External Computer)

EXAMPLE D. If a parity error occurs (Figure D-9), the CIU halts data transfer, sends Terminate to both the external computer and the DATANET-30, and indicates Parity Error status to both. A time-out condition has the same results.

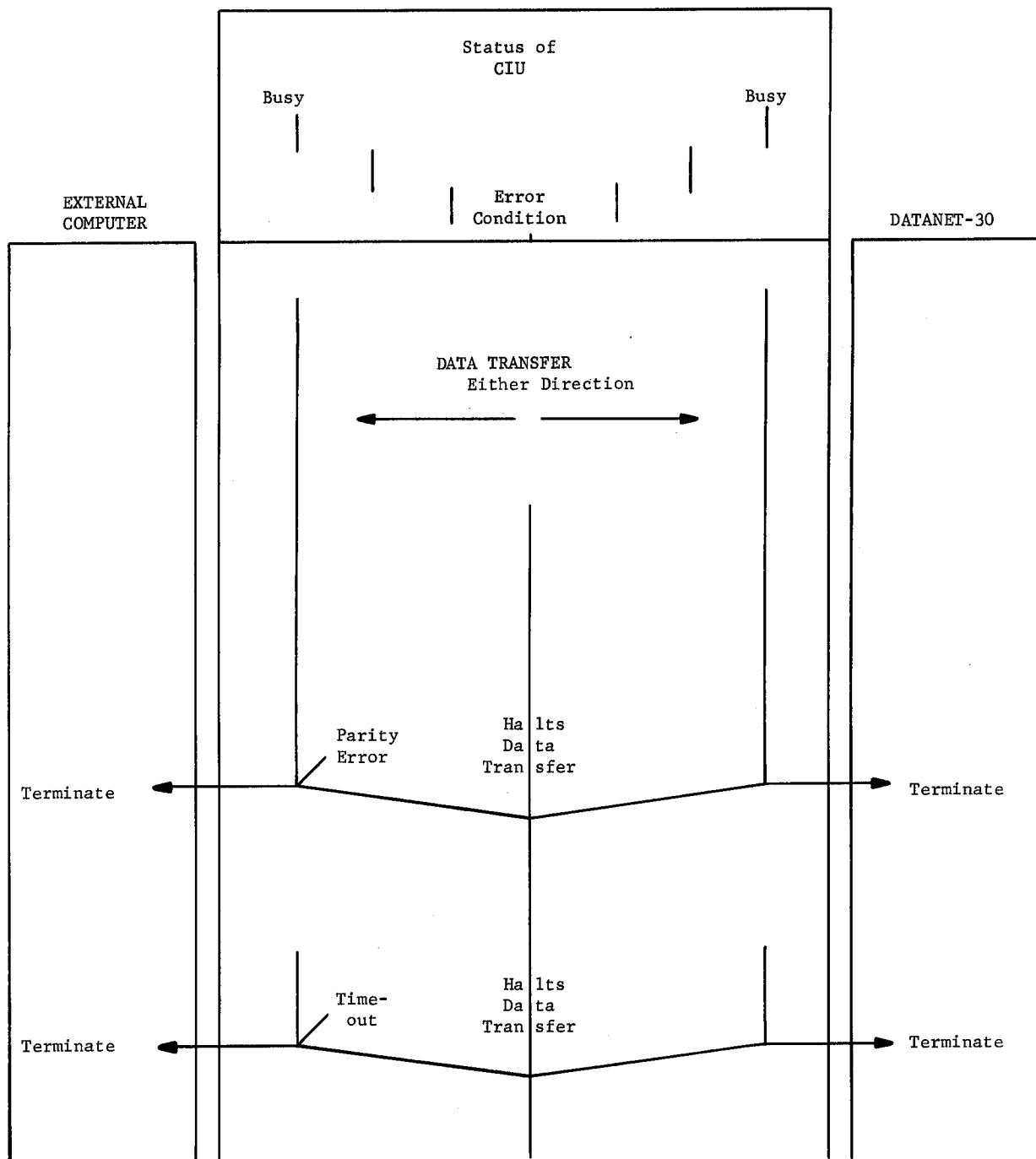


Figure D-9. Error Condition Sequence

## **PROGRAMMING CONVENTIONS**

The following items cover programming conventions, restrictions, and suggestions that must be considered for maximum programming efficiency.

### **DATANET-30 Conventions**

The DATANET-30 conventions are as follows:

1. The status of the CIU is always indicated in the status lines (NES3, 4, 5, and 6).
2. External status line 9 (NES9) must be checked within every 100 milliseconds to prevent time-out in the CIU when a Special Interrupt condition exists.
3. External status line 1 (NES1) must be checked within every 100 milliseconds to prevent time-out in the CIU during data transfer.
4. A Reset Status (RSS) (DEF6) can reset any status except Intermediate. The instruction must be issued to reset Parity Error, Time-Out or Command Reject.
5. The CIU halts data transfer after the first four DATANET-30 words. This is the halt after the functional message.
6. If necessary, by issuing a Write Initiate instruction, the DATANET-30 can interrupt and terminate data transfer from the external computer during data transfer after the functional message. The external computer status goes from Busy to Intermediate. The DATANET-30 status remains Busy.
7. The program must always reset the Terminate signal (DEF5) if Terminate is present, before issuing a Read Response or Write Initiate.
8. If a time-out occurs before a Read Response by the DATANET-30 is given, Intermediate status and Special Interrupt are replaced with Time-Out status and Terminate.
9. An End Data Transfer signal at any time from the external computer terminates data transfer from the DATANET-30, providing a multiple of 3 characters have been received by the external computer.
10. There are two ready conditions. The first is Ready status (NES3, 4, 5 and 6), when the CIU is ready to accept a Write Initiate instruction. The second occurs when the CIU is ready to transfer data (NES1) either to or from the DATANET-30.
11. The DATANET-30 must send at least one more word to the CIU/external computer following the functional message.

### **External Computer Conventions**

The CIU is in all respects a peripheral to the external computer, except that:

1. The CIU halts data transfer after the first 12 characters.
2. If the external computer issues a Write Initiate and does not get a Read Response from the DATANET-30 before the 100-millisecond timer times out, the CIU drops the WDN, sends Time-Out status and Terminate to both computers. The external computer must again try the Write Initiate.

3. Data transfer from the DATANET-30 can be interrupted once the Read Response has been issued, if the external computer issues an End Data Transfer signal.
4. The DATANET-30 can interrupt and terminate data transfer from the external computer after the functional message has been transferred.
5. A Reset Status (RSS) instruction to either computer must be issued only to reset Parity Error or Time-Out status.
6. The timer starts after completion of each command sequence and after each transfer of data.
7. After transfer of the functional message from the DATANET-30, three more characters must be transferred from the CIU. If this transfer does not take place, a time out will occur. If only one or two characters are transferred and then an End Data Transfer is issued by the external computer, the CIU will time out.
8. The external computer must always receive data from the CIU in a multiple of three characters before issuing an End Data Transfer. If all characters are not transferred from the CIU, a time out will occur.

## **PROGRAMMING EXAMPLES**

### **Receive Example**

The example in Figure D-10 shows one way information may be received. This is not necessarily the way it would be done in an operating program. The example assumes that (1) the DATANET-30 can wait for the external computer to send data, (2) a block of 50 words will be received, (3) an area in memory has been reserved for the functional message, (4) a subroutine will analyze the functional message, and (5) the data will be stored in a properly assigned area. Also, some error checking has been omitted.

Three symbols have been left undefined intentionally. CHECK and EXIT serve as a means for checking status and returning to the main program. ANALCH1 will depend upon the contents of the functional message.

### **Transmit Example**

The example in Figure D-11 shows one way information may be transmitted. This is not necessarily the way it would be done in an operating program. The example assumes that (1) the DATANET-30 can wait at various times in the program, (2) the functional message has been prepared, (3) the data is in memory ready to be transferred, and (4) a method for determining when to send the End Data Transfer signal has been established. Also, some error checking has been omitted.

Four symbols have been left undefined intentionally. These serve as means for checking status of the CIU and returning to the main program. The development of CHECK1, CHECK2, OUT, and EXIT will depend upon program design.

PAGE 001						
00010*	00020*	SAMPLE PROGRAM FOR THE CIU 931				
004000 00030	ORG 2048	ORIGIN LOCATION				
04000 104001 00040	FUNTAB INA *+1	TABLE FOR FUNCTIONAL MSG				
000004 00050	BSS 4					
* 04005AQ 000000 00060	CHECK	WHY STATUS IS NOT INTERMEDIATE				
04006 104007 00070	DATASTR INA *+1	DATA RECEIVED STORAGE AREA				
000062 00080	BSS 50					
* 04072U 000000 00090	ANALYCHR LNK ANALYCH1	ANALYSE FUNCTIONAL MSG ROUTINE				
* 04073U 000000						
* 04074A0 000000 00100	EXIT	RETURN TO MAIN PROGRAM				
04075 000001 00110	D1 DEC 1	CONTROL CONSTANT				
000732 00120	CIUADDR EQU '26	CIU ADDRESS				
04076 000062 00130	FIFTY DEC 50	WORDS RECEIVED CONSTANT				
00140*						
00150*	RECEIVE FUNCTIONAL MESSAGE					
00160*						
04077 011032 00170	READ PIC CIUADDR	PUT CIU ADDRESS IN C REG				
04100 022400 00180	READY NES 9	IS SPEC INTERRUPT ON				
04101 120100 00190	BZE *-1	NO CHECK AGAIN				
04102 026400 00200	DEF 9	READ RESPONSE				
04103 022001 00210	DF NES 1	IS DATA FLAG SET				
04104 120103 00220	BZE *-1	NO CHECK AGAIN				
04105 060044 00230	TRA R,B	YES RECEIVE WORD				
04106 704000 00240	STB FUNTAB,	STORE IN MSG TABLE				
04107 023000 00250	NES 0	IS TERMINATE SET				
04110 120103 00260	BZE DF	NO CHECK DATA FLAG				
04111 100112 00270	BRU RECDAT	YES TWELVE CHAR RECEIVED				
00280*						
00290*	RECEIVE DATA					
00300*						
04112 022020 00310	RECDAT NES 5	IS STATUS INTERMEDIATE				
04113 120005 00320	BZE CHECK	NO WHAT IS IT				
04114 026020 00330	DEF 5	YES RESET TERMINATE				
04115 110072 00340	BRS ANALYCHR	ANALYSE FUNCTIONAL MESSAGE				
04116 026400 00350	DEF 9	READ RESPONSE				
04117 060010 00360	TRA 0,A	SET COUNT IN A REG				
04120 022001 00370	DFD NES 1	IS DATA FLAG SET				
04121 120120 00380	BZE *-1	NO CHECK AGAIN				
04122 060044 00390	TRA R,B	YES RECEIVE WORD				
04123 704006 00400	STB DATASTR,	STORE IN DATA WORD AREA				
04124 420075 00410	AMA D1	ADD ON TO COUNTER				
04125 570076 00420	XAZ FIFTY	IS COUNT FIFTY				
04126 120074 00430	BZE EXIT	RETURN TO MAIN PROGRAM				
04127 023000 00440	NES 0	IS TERMINATE SET				
04130 120120 00450	BZE DFD	NO CHECK DATA FLAG				
04131 026020 00460	DEF 5	YES RESET TERMINATE				
04132 100074 00470	BRU EXIT	RETURN TO MAIN PROGRAM				
004077 00480	END READ					

Figure D-10. Sample Receive Program

PAGE 001					
			ORG 4000	ORIGIN LOCATION	
* 07640AO	000000	00020	OUT	A ROUTINE TO CHECK	
		00030*		WHY CIU IS NOT READY	
* 07641AO	000000	00040	CHECK1	A ROUTINE TO CHECK	
		00050*		WHY CIU DID NOT GO BUSY	
07642	107643	00060	FUNWORD	INA *+1	FUNCTIONAL MSG AREA
	007643	00070		BSS 4	
* 07647AO	000000	00080	CHECK2	A ROUTINE TO CHECK	
		00090*		WHY STATUS IS NOT BUSY	
07650	107651	00100	DATASND	INA *+1	DATA TO BE SENT
	007651	00110		BSS 50	
	0000032	00120	CIUADDR	EQU 26	CIU ADDRESS
07733	000001	00130	DACOUNT	OCT 1	CONSTANT TO ADD TO COUNT
* 07734AO	000000	00140	EXIT		RETURN TO MAIN PROGRAM
07735	000000	00150	LAST	DEC 0	TOTAL COUNT TO BE SENT
07736	000004	00160	LASTFUN	DEC 4	FOURTH FUNCTIONAL MSG WORD
		00170*			
		00180*	SEND FUNCTIONAL MESSAGE		
		00190*			
07737	011032	00200	WRITE	PIC CIUADDR	PUT CIU ADDRESS IN C REG
07740	026020	00210		DEF 5	RESET TERMINATE
07741	022074	00220	WRITE1	NES 3456	IS STATUS READY
07742	131640	00230		BNZ OUT	NO GET OUT
07743	026200	00240		DEF 8	YES ISSUE WRITE INITIATE
07744	022040	00250		NES 6	IS STATUS BUSY
07745	131747	00260		BNZ *+2	YES
07746	101641	00270		BRU CHECK1	NO WHAT IS IT
07747	060010	00280		TRA 0,A	LOAD ZEROS IN A REGISTER
07750	022001	00290	W1,	NES 1	IS DATA FLAG SET
07751	121750	00300		BZE *-1	NO CHECK AGAIN
07752	605642	00310	LDB FUNWORD,		YES TRANSFER FUNCTIONAL MSG
07753	060401	00320		TRA B,T	TRANSFER WORD
07754	421733	00330	AMA DACOUNT		INCREMENT WORD COUNT
07755	571736	00340	XAZ LASTFUN		IS THIS THE FOURTH WORD
07756	131750	00350		BNZ W1	NO SEND NEXT WORD
07757	101760	00360	BRU SEND		YES SEND DATA
		00370*			
		00380*	TRANSMIT DATA		
		00390*			
07760	060010	00400	SEND	TRA 0,A	LOAD ZEROS IN A REGISTFR
07761	022040	00410	S1	NES 6	IS STATUS BUSY
07762	121647	00420		BZE CHECK2	NO WHAT IS IT
07763	022001	00430		NES 1	IS DATA FLAG SET
07764	121761	00440		BZE *-3	NO CHECK STATUS
07765	605650	00450	LDB DATASND,		LOAD DATA WORD
07766	060401	00460		TRA B,T	TRANSFER WORD
07767	421733	00470	AMA DACOUNT		INCREMENT WORD COUNT
07770	571735	00480	XAZ LAST		IS THIS THE LAST WORD
07771	131761	00490		BNZ S1	NO TRANSFER NEXT WORD
07772	027000	00500		DEF 0	YES END DATA TRANSFER
07773	101734	00510	BRU EXIT		GO TO MAIN PROGRAM
*	U	000000	00520	END READ	

Figure D-11. Sample Transmit Program



## APPENDIX E

# **CIU-930 COMPUTER INTERFACE UNIT**

## **GENERAL**

The CIU-930 provides the interface for the DATANET-30 and a Compatibles/200 information processing system and is used to transfer 21-bit words between them. The words are transferred in parallel. The CIU-930 connects into any channel of the DATANET-30 buffer selector in the same manner as any other DATANET-30 buffer. On the processor side, the CIU-930 connects into any priority control channel. The buffer selector address of the CIU is specified by the wiring of the buffer selector address plug for the CIU module. There is no DATANET-30 hardware restriction on the number of CIU's which may be used, other than the physical space occupied. Each CIU-930 occupies one module. The CIU is asynchronous. It has no service rate and is program controlled.

## **CIU-930 INSTRUCTIONS**

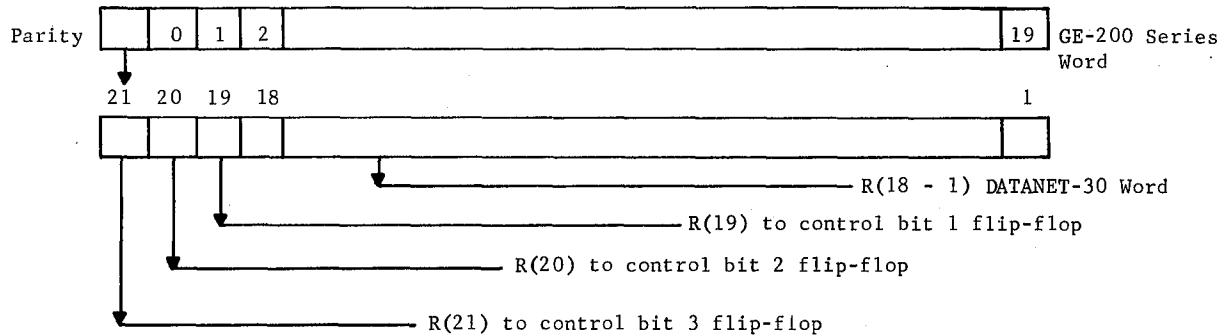
Following are the CIU-930 instructions:

Register Transfer TRA R,B

Five things are accomplished (see illustration below):

1. The data word contained in the CIU data register is transferred to B: CIU (18 - 1) to B (18 - 1), CIU (19) to control bit 1 flip-flop, CIU (20) to control bit 2 flip-flop and CIU (21) to control bit 3 flip-flop.
  2. The CIU data register is reset.
  3. The address register is increased by 1 and receive mode is set (DEF1).
  4. The transfer of another word is initiated.
  5. The CIU is put in the busy condition.

GE-200 System to CIU-930 to DATANET-30



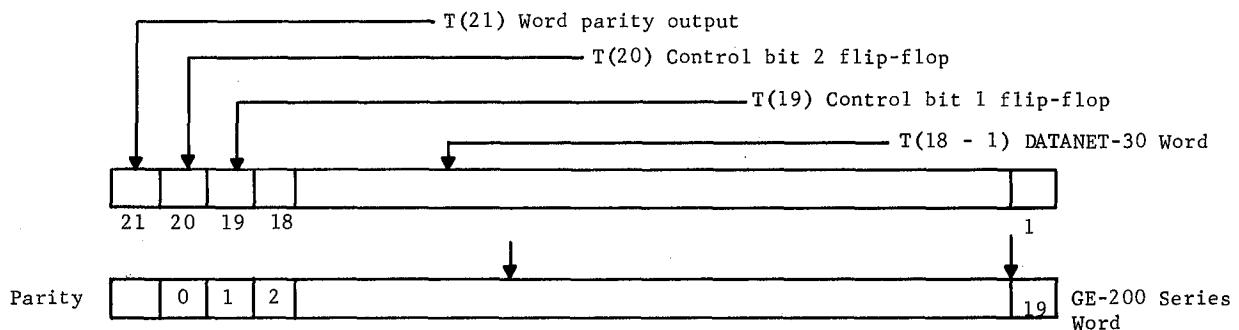
Register Transfer

TRA B, T

Four things are accomplished (see illustration below):

1. The 18-bit word contained in the B-register is sent to the transmit buffer positions 18 - 1. Control bit 1 flip-flop goes to position 19, control bit 2 flip-flop to position 20, and word parity output to position 21.
2. The transfer of the word from the CIU to the Compatibles/200 system is initiated.
3. The address register is increased by 1 and transmit mode is set (DEF2).
4. The CIU is put in the busy condition.

DATANET-30 to CIU-930 to GE-200 System



Mnemonic

Operand

Word Times

LDT

M

1

The contents of the specified memory address M are sent to the address register of the CIU.

DATANET-30

<u>Mnemonic</u>	<u>Operand</u>	<u>Word Times</u>
DEF	I	1
DEF 1	Resets data register, increases address register by 1, and puts CIU in receive mode. Puts the CIU in busy condition, which initiates the transfer of another word from the computer.	
DEF 2	Resets data register (before data comes from a register during a register transfer instruction). Puts the CIU in the transmit mode, increases address register by 1.	
DEF 3 - 7	Not used.	
DEF 8	Reset Busy.	
DEF 9	Sends an automatic program interrupt signal to the computer.	
DEF 0	Resets the address register (before the address comes from the program during a Load T (LDT) instruction).	

## EXTERNAL STATUS LINES

External status line indications are as follows:

- NES 1      The CIU is not busy.
- NES 2 - 8    Not used.
- NES 9      API is set in the GE-200 Series computer.
- NES 10     Not used.

## RECEIVE OPERATION

Assume that nothing is happening as far as the CIU is concerned. At some point, the program in the DATANET-30 initiates taking a block of words from the central processor memory. The program puts a number equal to one less than the initial memory address of the block in the address register of the CIU by means of an LDT instruction. Then the program sends a control signal to the CIU, via the external function drivers, which increases the address register by 1, puts the CIU in the receive mode, resets the data register, and initiates the transfer of the word from the specified central processor memory location to the data register in the CIU. After the word is in the data register, the CIU is no longer busy.

This condition can be tested via external status line 1 (NES 1). The program now executes a register transfer instruction to take the word out of the data register of the CIU and into the DATANET-30. This register transfer instruction also increases the address in the CIU address register by 1, puts the CIU in the receive mode, resets the data register, and initiates the transfer of another word from the central processor memory. This process repeats until the DATANET-30 program has received a sufficient number of words.

An example of receive operation is shown below:

<u>Location</u>	<u>Instruction</u>	<u>Symbol</u>	<u>OPR</u>	<u>Operand</u>	<u>X</u>	<u>Remarks</u>
	05670		ORG	3000		ORIGIN LOCATION
05670	011031		PIC	25		PLACE CIU ADDRESS IN C COUNTER
05671	024001		DIF	1		RESET CB1,2 AND 3
05672	251763		LDT	ADRESS		LOAD CIU WITH 225 MEMORY (225 ADD.-1)
05673	026001		DEF	1		SETS REC. MODE
05674	022001	GETWD	NES	1		CIU BUSY
05675	121673		BZE	*-1		YES, GO BACK
05676	060044		TRA	R,B		NO, TRANSFER WORD TO B
05677	020002		NIS	2		CHECK OUTPUT OF WORD PARITY NETWORK
05700	135710		BZE	ERROR	X	IF OUTPUT IS ZERO EXIT TO ERROR
05701	705706		STB	DATAIN	X	PARITY OK STORE IN MEMORY
05702	771705		XBZ	END		IS THIS THE LAST WORD?
05703	125711		BZE	EXIT	X	YES, EXIT
05704	341706		ADO	DATAIN		NO, ADD ONE TO MEMORY ADDRESS
05705	101673		BRU	GETWD		GO BACK GET NEXT WORD
05706	777777	END	OCT	777777		END CONSTANT
05707	015530	DATAIN	IND	7000		INPUT ADDRESS
05710	000763	ADRESS	IND	499		225 ADDRESS-1
05711	005752	ERROR	IND	3050		ERROR ADDRESS
05712	005757	EXIT	IND	3055		NORMAL EXIT ADDRESS

Initially the CIU address is put into the C-register. The address register of the CIU is loaded with the desired central processor memory address. The address must be less than the desired starting address, because the DEF 1 instruction which puts the CIU into the receive mode, also increments the address counter by 1. The CIU is tested for a busy condition by the NES1 command and the program stays in a loop until the CIU becomes ready. When the CIU becomes ready, the word is transferred to the B-register and the address counter is automatically counted up 1. The word is stored in memory, then tested for end-of-block condition. If the end-of-block condition is not found, control is transferred back to get another word.

### Transmit Operation

Assume that nothing is happening as far as the CIU is concerned. At some point, the program in the DATANET-30 decides to put a block of words into the Compatibles/200 system. The program puts a number equal to one less than the initial memory address into the address register of the CIU with an LDT instruction.

Then the program transfers a word into the CIU data register with a register transfer instruction. This register transfer instruction also puts the CIU in the busy condition mode, increases the address in the address register by 1, and initiates the transfer of the word from the data register into the central processor memory. After the word has been written into memory, the CIU is

no longer busy. This condition can be tested via external status line 1 (NES 1). The DATANET-30 program can now put another word in the data register and send it to the central processor. This process repeats until the DATANET-30 program decides that sufficient words have been transferred to the Compatibles/200 system.

The transmit example works just the reverse of receive with the exception of the DEF 2 instruction to set the CIU to the transmit mode and the DIF 1 to reset the CB 1, CB 2, and parity flip-flops.

<u>Location</u>	<u>Instruction</u>	<u>Symbol</u>	<u>OPR</u>	<u>Operand X</u>	<u>Remarks</u>
	07640		REM		TRANSMIT TO 225 VIA CIU
07640	011031		ORG	4000	ORIGIN LOCATION 4000
07641	026001		PIC	25	PLACE CIU ADDRESS IN C
07642	251747		DIF	1	RESET CB1, 2 AND 3
07643	022001	SENDWD	LDT	ADRESS	LOAD 225 ADDRESS INTO CIU & SET TRANS
07644	121643		NES	1	CIU BUSY MODE
07645	601653		BZE	*-1	YES TRY AGAIN
07646	060401		LDB	DATOUT	X NO LOAD WORD TO BE TRANSFERRED
07647	771654		TRA	B,T	TRANSFER TO CIU DATA BUFFER
07650	121655		XBX	ENDWD	IS THIS THE LAST WORD?
07651	341653		BZE	TEXIT	YES EXIT
07652	101643		ADO	DATOUT	NO ADD ONE TO MEMORY ADDRESS
07653	013560	DATOUT	BRU	SENDWD	GO BACK TRANSMIT NEXT WORD
07654	777777	ENDWD	IND	6000	DATANET-30 OUTPUT ADDRESS
07655	007722	TEXIT	OCT	777777	END WORD CONSTANT
07656	001747	ADRESS	IND	4050	EXIT ADDRESS
				999	225 ADDRESS-1

The DEF 8 instruction--Reset Busy--is used when an error condition, such as parity error, causes the GE-200 system to halt when the CIU is busy. When the error condition is reset at the GE-200 operator's console, the CIU indicates busy when it actually is not. The busy indication remains until reset either by manual reset at the DATANET-30 or a DEF 8 instruction.

## **Programming Parity**

When data is received by the DATANET-30 from a GE-200 Series computer, the memory parity bit comes into the PFF, bits 0 and 1 come into CB2 and CB1 respectively, and bits 2-19 come into bits 18-1 of the B-register.

It is important to note the PFF contains the contents of the parity bit that was in memory of the GE-200 system. It may be either zero or one, depending on the contents of the 20 data bits of the word. Interrogating the parity flip-flop does not tell the programmer whether or not the data transferred correctly from the GE-200 Series computer to the DATANET-30. To check that, the programmer must interrogate the word parity signal. Because the GE-200 Series computers use odd parity, the word parity signal should be odd (that is, nonzero) for correct parity.

When transmitting from the DATANET-30 to a GE-200 Series, the CIU picks up parity, not from the PFF, but from the output of the word parity network. The word parity signal is computed from the contents of the entire B-register and all three flip-flops. One might say that this is an unimportant differentiation because the PFF is included in the calculation of the word parity network. However, it makes a difference to the programmer. He does not need to be concerned with outgoing parity except to always reset the PFF before transmitting data and let the word parity network tell the CIU whether the parity bit should be odd or even.

It is important to understand that the word parity signal and character parity signal are not error signals. They do not indicate whether or not parity is correct. They merely indicate whether the data contains an even or an odd number of bits. (If the data contains an odd number of bits, the result of the NIS will be nonzero.) If the peripheral device being addressed uses odd parity, a nonzero result from an NIS indicates correct parity. If the device uses even parity, the reverse is true.

For additional information, see "Parity," Chapter I.

## CONTENTS

	Page
DIALING ADAPTOR UNIT (DAU-930)	
General Description . . . . .	F-1
Buffering . . . . .	F-1
Configurations . . . . .	F-1
Automatic Calling Unit (ACU) . . . . .	F-3
The Abandon Call and Retry (ACR) Timer . . . . .	F-4
Instructions . . . . .	F-4
Drive External Function . . . . .	F-4
External Status Lines . . . . .	F-5
Call Completed . . . . .	F-5
Call Termination . . . . .	F-5
Sequence of Placing a Call . . . . .	F-6
Controls and Indicators . . . . .	F-6
Dialing Program Example . . . . .	F-7
The Buffer Operations . . . . .	F-7

## ILLUSTRATIONS

Figure	Page
1. Dialing Adapter Units with Telephone Equipment and Bit Buffers. . . . .	F-2
2. Dialing Adapter Units with Telephone Equipment and Character or Word Buffers. . . . .	F-2
3. Dialing Adapter Unit with Telephone Equipment and a Receive Parallel Buffer Unit. . . . .	F-3



## APPENDIX F

### DIALING ADAPTER UNIT (DAU-930)

#### **GENERAL DESCRIPTION**

The General Electric Dialing Adapter Unit (DAU), an optional feature of the DATANET-30, provides the means for dialing telephone numbers of remote locations to transmit or receive data. The DAU, with its associated telephone equipment, enables the DATANET-30 to dial virtually any telephone number connected with the Bell System.

The associated telephone equipment consists of an Automatic Calling Unit (ACU) and a Data Set. The General Electric DAU is used only to place the telephone call. The ACU actually dials the number and makes the connection.

The dialing adaptor module may contain from one to ten Dialing Adaptor Units. The module is mounted in one of the spaces provided for buffer modules that interface with the buffer selector. Each DAU is an independent unit operating through the buffer selector. Each DAU is associated with one transmit/receive buffer, such as a bit buffer, or character buffer, etc. Two buffer selector addresses are required, one for the DAU and one for the associated buffer.

The transmission or reception of data takes place through the associated Data Set and buffer on the buffer selector. Only one transmission line is associated with each DAU, ACU, and Data Set. If it is desired to dial out on more than one line, a separate DAU, ACU, and Data Set are used for each line (or call). Incoming calls are handled by the Data Set.

#### **BUFFERING**

The DAU provides one four-bit digit buffer. One four-bit binary-coded decimal digit ( $0_{10} - 9_{10}$ ) at a time is presented to the DAU by the DATANET-30 program. This digit occupies the four least-significant positions of a DATANET-30 word. The DAU then presents this digit to the telephone company ACU for dialing.

#### **CONFIGURATIONS**

A DAU may be set up in the configurations shown in Figures 1, 2, and 3.

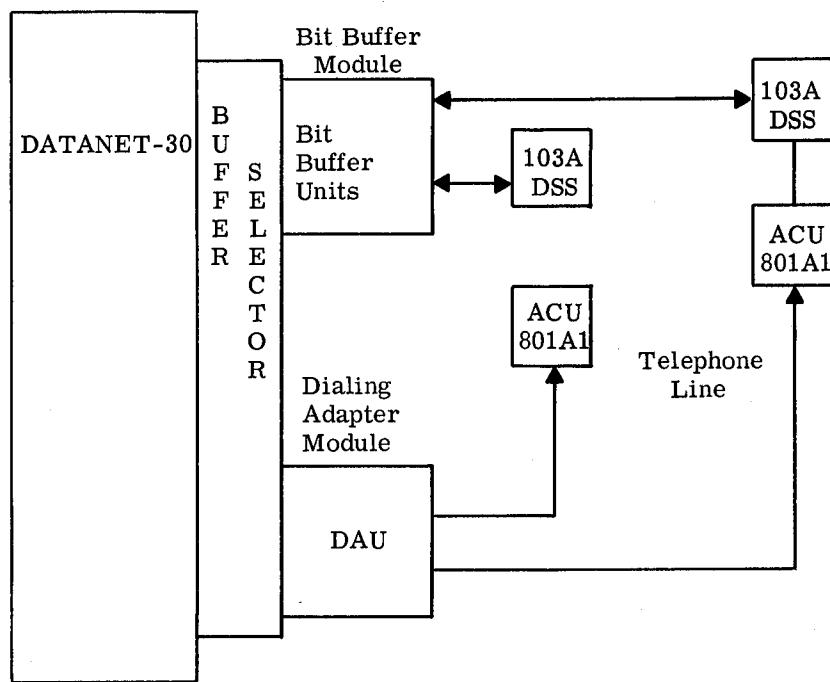


Figure 1. Dialing Adapter Units with Telephone Equipment and Bit Buffers.

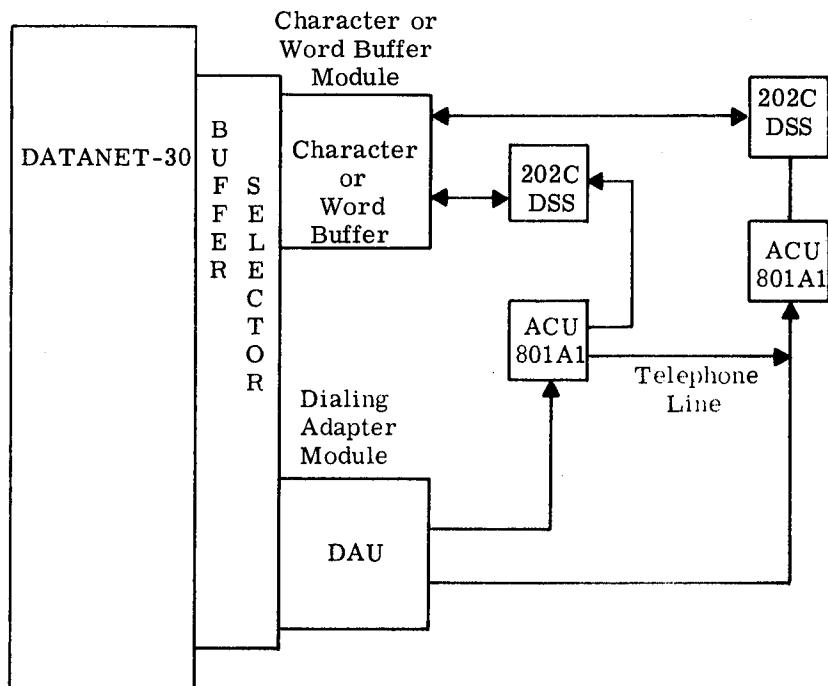


Figure 2. Dialing Adapter Units with Telephone Equipment and Character or Word Buffers.

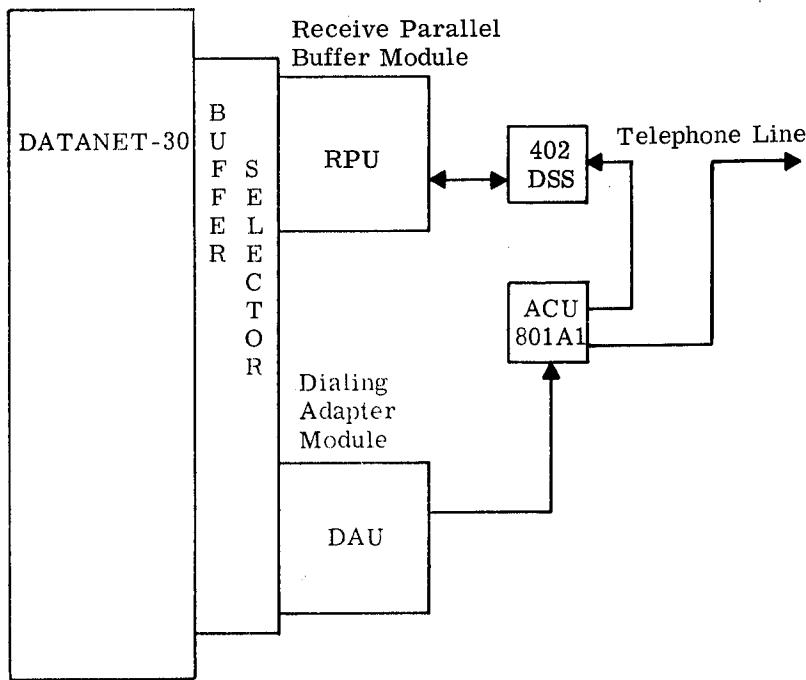


Figure 3. Dialing Adapter Unit with Telephone Equipment and a Receive Parallel Buffer Unit.

#### AUTOMATIC CALLING UNIT (ACU)

The Telephone Company Data Auxiliary Set 801A for automatic calling is not covered in great detail. Anyone interested in more information than contained here may obtain a copy of the Data Auxiliary Set 801A-Interface Specification (March 1964) from:

Data and Teletypewriter Planning Engineer  
American Telephone and Telegraph Company  
195 Broadway  
New York, New York 10007

The operating features of the ACU pertinent to the operation of the DAU are:

1. The ACU obtains the dial tone in response to a DEF 3, Set Call Request.
2. The ACU dials the digit received from the DAU.
3. When the last digit has been dialed, the ACU waits for the called location to answer.
4. The ACU detects the answer signal.
5. The ACU then transfers control of the call (telephone line) to the Data Set connected to the ACU and the transmit/receive buffer.
6. If the call cannot be successfully completed, the Abandon Call and Retry timer notifies the program.

## **THE ABANDON CALL AND RETRY (ACR) TIMER**

The Abandon Call and Retry (ACR) timer indicates that the call was not completed because of a busy condition, no answer, wrong number or a delay between digits dialed. The exact cause of an ACR condition is not indicated.

The timer setting is adjustable to 7, 10, 15, 25, and 40 seconds. A time period relative to the time required to set up calls by the telephone company can be selected. Local calls take less time than calls going through several telephone switching centers.

Any number of digits may be dialed through the ACU, but the ACU does not know when the last digit was dialed. Therefore, the ACR timer is reset to zero after each digit is dialed. Thus, the delay between dialing each digit cannot exceed the time setting for the ACR timer. Should this occur, the timer presents an ACR signal to the DAU and terminates the call before all numbers have been presented to the ACU for dialing.

The ACU does not detect a busy signal. When the number dialed is busy, the ACU waits for an answer signal. When no answer signal is detected because of a busy signal, the timer presents an ACR signal to the DAU and terminates the call.

## **INSTRUCTIONS**

Instructions for the DAU follow the same rules as other units on the buffer selector. The following DEF and NES instructions apply to the DAU with an ACU model 801A1.

### **Drive External Function**

DEF 1 - Not used.

DEF 2 - Reset Digit Buffer--Used to reset the digit buffer in the DAU before transferring a new digit to the DAU.

DEF 3 - Set Call Request--Request dial mode at the ACU. The DATANET-30 is initiating a call.

DEF 4 - Reset Digit Present--The Digit Present signal is removed from the ACU. This instruction must be used each time the ACU accepts a digit for dialing and turns off the Present Next Digit (PND) signal.

DEF 5 - Reset Call Request--Turns off the dial mode at the ACU. The ACU goes from busy to not busy. It is used to terminate a call after a call has been successfully completed or an ACR signal has occurred.

DEF 6 - Set Digit Present--Signals the ACU that a digit is in the DAU to be dialed.

DEF 7 - 0 - Not Used.

## **External Status Lines**

NES 1 - Not Used.

NES 2 - Present Next Digit (PND)--A one indicates that the ACU is ready to receive a digit.

NES 3 - Power On Indicator (PWI)--A one indicates power on at the ACU and the ACU is operative.

NES 4 - Abandon Call and Retry (ACR)--A one indicates that the present call cannot be completed and should be terminated.

NES 5 - Busy--A one indicates a Busy condition of the DAU, the ACU and/or the telephone line associated with the ACU. The busy conditions are: a call in progress (a DEF 3 instruction has been executed), the telephone line is busy, or the ACU is in the test mode. The ACU will ignore a call request when the telephone line is occupied (busy).

NES 6 - Data Set Status (DSS)--A one indicates that dialing has been completed, the call was answered and line control has been transferred from the ACU to the Data Set connected to the associated transmit/receive buffer. The Data Set is in the data mode to transmit or receive data. This signal is also present if the Data Set answers an incoming call.

## **CALL COMPLETED**

After the last digit is dialed, the program must determine whether or not the call was successfully completed. There are two indicators for this. First, the ACR signal did not come up. Second, and more positively, the DSS line did come up.

The Abandon Call and Retry signal is mentioned because if 7-40 seconds elapse after receiving a digit and the ACU has not received another digit from the DATANET-30 or an answer signal from the telephone line, the ACU will give an ACR signal. Therefore, a check must be made for both the ACR and the DSS signal after giving the last digit to be dialed to the ACU.

The Abandon Call and Retry may occur quite frequently, since it may represent a busy signal. The program must terminate and dial the number again. If there is a heavy load, the program may go on to another number and try the busy number again later.

## **CALL TERMINATION**

The program must terminate (hang up) one call before placing the next call. The telephone company equipment requires about 1.5 seconds to hang up after the signal has been given to do so. The program must wait for the hang-up to finish before attempting to place another call. The Not Busy condition indicates the call has been terminated.

There are various methods of terminating a call. However, the only one used at the present time with the DATANET-30 is as follows:

1. The Set Call Request (DEF 3) signal is held ON until the call is to be terminated.

2. A Reset Call Request (DEF 5) instruction starts the terminate sequence.
3. The Set Call Request signal (DEF 3) cannot be used to initiate a new call until the Line Busy (NES 5) signal goes off. The timing of Busy going off is a function of the telephone company equipment. The Not Busy condition will then be indicated.
4. The call has been terminated when the Busy/Not Busy indication (NES 5) is 0 (Not Busy).

### **Sequence of Placing a Call**

When ready to place a call, the program must go through the following sequence:

1. Check for Busy (NES 5).
2. Set dialing mode (DEF 3) at the ACU.
3. Wait for time to Present Next Digit (PND) (NES 2).
4. When the ACU is ready to accept a digit, a PND signal (NES 2) is given by the ACU to the DAU. The program then transfers the digit to be dialed to the DAU.
5. Set Digit Present (DEF 6) in the DAU to tell the ACU that the digit is ready.
6. At this point, a maximum delay of 1.6 seconds is required for dialing the digit.
7. When the ACU takes the number, the PND (NES 2) signal goes to zero. The program must now Reset Digit Present (DEF 4). The PND signal cannot come up until after a Reset Digit Present.
8. The program must wait for the Present Next Digit (PND) (NES 2) signal to be presented before another digit can be transferred to the DAU.
9. When PND is again present, the program repeats from step 4.
10. When all digits have been dialed, the program looks for a completed call. The program may then transmit or receive data through the associated buffer.
11. If NES 6 (DSS) does not indicate a completed call, the program should look for an ACR (NES 4) signal. A one indicates that the call was not completed within a specified time and that the call must be abandoned and retried later. The "specified time" is controlled by the timer in the ACU.
12. To terminate a call, the program will Reset Call Request (DEF 5). A minimum of 1.5 seconds must be allowed between the termination of one call and the start of another on the same line.

### **CONTROLS AND INDICATORS**

There are no controls or indicators in the usual sense except for the MANUAL CLEAR button on the DATANET-30 control panel. All other controls and indicators are by program only.

## DIALING PROGRAM EXAMPLE

The following example shows one way to use the Dialing Adapter Unit to dial a number. This is not necessarily the way it would be done in an operating program. The example assumes that the DATANET-30 can wait for the telephone functions to be accomplished, a subroutine will provide numbers to be dialed, and that the main program will control the telephone line after a call has been completed.

In this example program, three symbols have been left undefined intentionally. These are TRANREC, MAINACR, and CHCKBUF. The TRANREC routine is used in this example to return to the main program and integrate the competed call into the Scan instruction.

The MAINACR routine would be some method of notifying the dialing program that a call was not completed. Action taken in this case would depend upon system considerations.

The CHCKBUF routine may or may not actually be used. If a line is busy, the program might search for a line that is not busy. However, if a line is busy when it should not be, an error condition could exist requiring action by the operator. This again depends upon system design and the number of lines available for dialing.

Definition of the above programming aspects will vary with system design and development.

The TYPPWR symbol represents a routine to notify the operator that an ACU is without power. This condition should never exist because power should always be on the Telephone Company equipment. A power failure indication would require operator action.

## THE BUFFER OPERATIONS

The Dialing Adapter Unit and the Automatic Calling Unit may be used with several different digital subsets and transmit/receive buffers. The buffers in turn can connect to a variety of terminal equipment. The DEF and NES instructions for the transmit/receive buffers vary in accordance with the buffer used, the digital subset used, and the terminal equipment.

A discussion of all the different combinations of input/output buffers, digital subsets, terminal equipment and instructions (DEF and NES) for each combination as well as programming considerations for each is beyond the scope and intent of this section. Refer to the appropriate sections in this manual for information on a particular buffer.

PAGE 001

\* \* \* SAMPLE PROGRAM FOR DIALING A TELEPHONE NUMBER

000060	000067	010	D7	ORG 4A	NUMBER OF DIGITS TO BE DIALED
000061	100061	n20	DIAL NUM	INA *+0	THE NUMBER TO BE DIALED
*	000062	n30	RSS 7	NUMBER TABLE	
00071	777777	n40	DFCN1	OCT 777777	STATUS ONE CONSTANT
00072A0	000000	n50	TRANREC		TRANSIT OR RECEIVE 20. TIC
*	000073	n60	TYPPWR	RSS 16	NSC CHECK ADDRESS
00113A0	000000	n70	MAILACK		A ROUTINE IN MAIN PROGRAM FOR
*	00114A0	n80	*	CHKBUF	DIALING THE SAME NUMBER AGAIN
*	000020	100	DATAADR	EQH 1K	CHECK BUFFER FOR STATUS OF LINE
002000	011020	n130			ADDRESS OF ADU
02001	022004	n140	START	PIC DATAADR	
02002	122073	n141		NES 3	TS POWER ON
02003	022020	n142		RZF TYPPWR	NSC NO POWER ACU X
02004	132114	n150		NES 5	TS THE LINE BUSY
02005	026004	n160		RNZ CHKBUF	YES CHECK LINE STATUS
02006	402060	n170		DEF 3	NO SET DIAL MODE AT ACU
02007	022002	n180	DIAL	LDA n7	NUMBER OF DIGITS DIALED
02010	120007	n191		NES 2	TS PRESENT NEXT DIGIT ON
02011	606061	n200		RZE *-1	NO CHECK AGAIN
02012	060401	n210		LDR DIALNTR,	BUT DIGIT TO BE DIALED IN
02013	026040	n220		TRA R,T	LAST DIGIT
02014	022002	n230		DEF K	SFT DIGIT PRESENT
02015	130014	n240		NES 2	IS PRESENT NEXT DIGIT NOT
02016	026010	n250		RNZ *-1	NO CHECK AGAIN
02017	422071	n260		DEF 4	YES RESET DIGIT PRESENT
02020	130007	n270		AMA DEM1	INCREMENT DIGITS DIALED
02021	022010	n280		RNZ DIAL	TF NOT LAST DIGIT DIAL NEXT
02022	120024	n290		NES 4	TS ABANDON CALL UP
02023	102113	n300		RZE MSSWK	AC CHECK DS
02024	022040	n310		ARU MAINT	YES ACTIVATE MAIN PROGRAM
02025	120021	n320		YES K	TS CALL COMPLETED
02026	102072	n330		RZE CHRCALL	AC CHECK ABANDON CALL
				ARU TRUNCK	TRANSIT OR RECEIVE ISC
				EJT	

## CONTENTS

	Page
<b>CPC-930 COMMON PERIPHERAL CHANNEL</b>	
General Description . . . . .	G1-1
Data Transfer . . . . .	G1-1
Transfer Capacity . . . . .	G1-2
Block Diagram . . . . .	G1-2
Character Counter . . . . .	G1-3
Address Counter . . . . .	G1-3
Parity . . . . .	G1-3
Controls and Indicators . . . . .	G1-3
Interrupt Cycles . . . . .	G1-5
Cycle Assignment . . . . .	G1-5
Status and Substatus . . . . .	G1-6
Special Interrupt . . . . .	G1-6
Instruction Repertoire . . . . .	G1-7
DATANET-30 Instructions . . . . .	G1-7
DEF Instructions . . . . .	G1-7
NES Instructions . . . . .	G1-7
Register Transfer Instructions . . . . .	G1-8
Command Words . . . . .	G1-9
Data Transfer Commands . . . . .	G1-9
Non-Data-Transfer Commands . . . . .	G1-10
Programming Considerations . . . . .	G1-11
General Electric Standard Character Set . . . . .	G1-11
Disc Storage Unit (DSU) Subsystem . . . . .	G2-1
DS-20 Subsystem Instructions . . . . .	G2-1
Physical Organization of DSU Subsystem . . . . .	G2-2
DSU Controller . . . . .	G2-2
DSU . . . . .	G2-2
Logical Organization of DSU Controller . . . . .	G2-2
Device Code . . . . .	G2-4
Control Characters . . . . .	G2-4
Combined On-Line/Off-Line Commands . . . . .	G2-5
Control Characters 1, 2, and 3 . . . . .	G2-6
Control Characters 4, 5, and 6 . . . . .	G2-8
Input/Output Control . . . . .	G2-8
Instructions Requiring Transfer of Control Characters . . . . .	G2-10
Instructions Not Requiring Transfer of Control Characters . . . . .	G2-13
Status and Substatus . . . . .	G2-16
Magnetic Tape Units . . . . .	G3-1
Magnetic Tape Instructions . . . . .	G3-1
Command Words . . . . .	G3-2
Status and Substatus . . . . .	G3-2

	Page
<b>Printer . . . . .</b>	<b>G4-1</b>
Printer Instructions . . . . .	G4-1
Command Words . . . . .	G4-1
End Print Line . . . . .	G4-1
Print in Edit Mode . . . . .	G4-2
Special Characters . . . . .	G4-2
Escape Character . . . . .	G4-2
Ignore Character . . . . .	G4-2
Space Character . . . . .	G4-2
Skip Character . . . . .	G4-2
Slew Character . . . . .	G4-3
Summary of Special Characters . . . . .	G4-3
Print In Nonedit Mode . . . . .	G4-3
Automatic Slewling . . . . .	G4-4
Controls and Indicators . . . . .	G4-5
Status and Substatus . . . . .	G4-5
 <b>Card Reader . . . . .</b>	<b>G5-1</b>
CR-20 Card Reader Instructions . . . . .	G5-1
Timing Considerations . . . . .	G5-1
Data Transfer . . . . .	G5-2
Program Load . . . . .	G5-3
Status and Substatus . . . . .	G5-3
 <b>CP-10 Card Punch . . . . .</b>	<b>G6-1</b>
CP-10 Card Punch Instructions . . . . .	G6-1
Command Words . . . . .	G6-1
Timing Considerations . . . . .	G6-1
Data Transfer . . . . .	G6-3
Status and Substatus . . . . .	G6-5
 <b>CP-20 Card Punch . . . . .</b>	<b>G7-1</b>
CP-20 Card Punch Instructions . . . . .	G7-1
Command Words . . . . .	G7-1
Functional Description . . . . .	G7-1
Timing Considerations . . . . .	G7-2
Status and Substatus . . . . .	G7-3

## ILLUSTRATIONS

Figure		Page
G-1	Data Transfer Through a CPC-930 . . . . .	G1-2
G-2	Block Diagram. . . . .	G1-4
G-3	Cable Connector and Switch Bracket . . . . .	G1-5
G-4	Block Diagram of Data Flow of DSU Subsystem . . . . .	G2-2
G-5	Buffer Addresses and Functions of the 1024- Character Buffer . . . . .	G2-3
G-6	Sector Address Control Character Format . . . . .	G2-7
G-7	Examples of Instruction Sequence for a DS-20 Read Instruction. . . . .	G2-9
G-8	Status and Codes for the DS-20 Subsystem . . . . .	G2-16
G-9	Status and Codes for Magnetic Tape Subsystems . . . . .	G3-4
G-10	Summary of Special Characters Used in Edit Mode. . . . .	G4-4
G-11	Status and Codes for PR-20 Printer. . . . .	G4-6
G-12	Status and Codes for PR-21 Printer. . . . .	G4-7
G-13	Character Sequence for Read Card Binary Instruction . . . . .	G5-2
G-14	Character Sequence for Read Card Decimal Instruction . . . . .	G5-2
G-15	Status and Codes for CR-20 Card Reader . . . . .	G5-3
G-16	Card Punch Timing Cycle (in Milliseconds). . . . .	G6-2
G-17	Character Sequence for Punch Card Binary Instruction . . . . .	G6-3
G-18	Character Sequence for Punch Card Decimal Instruction . . . . .	G6-4
G-19	Status and Codes for CP-10 Card Punch . . . . .	G6-5
G-20	CP-20 Punch Cycle. . . . .	G7-2
G-21	Status and Codes for CP-20 Card Punch . . . . .	G7-3



## APPENDIX G

### CPC-930 COMMON PERIPHERAL CHANNEL

#### GENERAL DESCRIPTION

One DATANET-30\* Common Peripheral Channel (CPC-930) allows data transfer to or from memory between the DATANET-30 and one of the following peripheral devices:

- DS-20 disc storage unit (DSU) with all options
- MT-17, Mt-19, Mt-21, or MT-23 magnetic tape unit
- PR-20 or PR-21 printer
- CR-20 card reader
- CP-10 (100 cpm) or CP-20 (300 cpm) card punch

Each CPC occupies two option module spaces and can be installed only in rack 2. Each CPC is assigned an address on the buffer selector. The address is specified by the address plug for the CPC.

From one to four CPC's can be installed in a DATANET-30. The maximum is determined by the fact that only four CPC's can be installed in rack 2 or by the number of memory interrupt cycles used. Five memory interrupt cycles are available for assignment to the following equipment.

<u>NAME</u>	<u>CYCLES</u>	<u>CYCLE NO.</u>
Controller selector unit (CSU-931)	2	1 and 3
Processor interrupt unit (PIU-930)	1	2
CPC with DSU	2	x,x
CPC with magnetic tape units	1	x
CPC with printer	1	x
CPC with card reader	1	x
CPC with card punch	1	x

#### Data Transfer

Data is transferred via the CPC one character at a time to or from the peripheral and one DATA-NET-30 word at a time to or from the DATANET-30. The CPC contains the necessary shift circuitry to accumulate characters into words or to separate sequential characters, depending on the direction of transfer. Data transfer is illustrated in Figure G-1.

---

\*DATANET is a registered trademark of General Electric Company

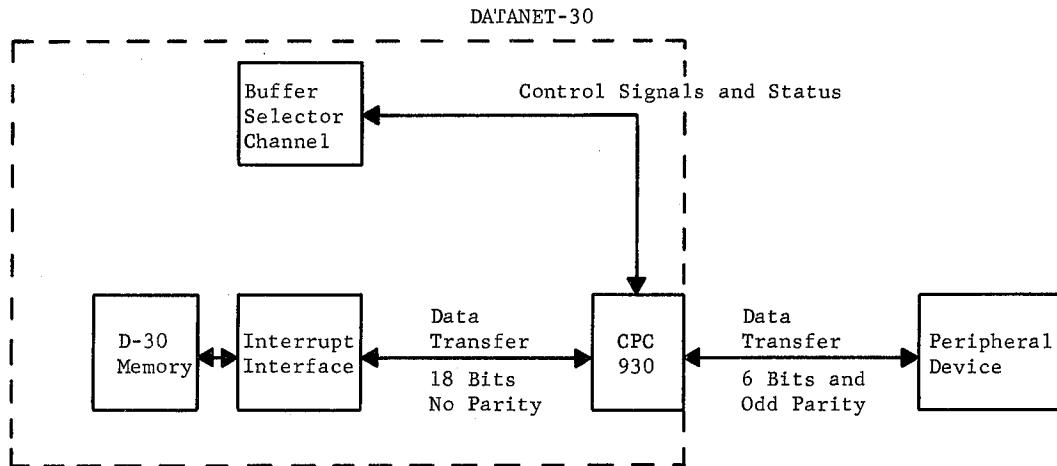


Figure G-1. Data Transfer Through a CPC-930

When data is transferred from the CPC to the peripheral device, the most significant character of the DATANET-30 word is transferred first and the least significant, last.

Characters received by the CPC from the peripheral equipment are assembled into three-character words. The first character is shifted to the most significant part of the DATANET-30 word, and the third character is transferred as the least significant part.

If a Terminate signal is received from the peripheral before a full three-character word is shifted into the CPC, the character or characters in the shift register are shifted to the most-significant position. The CPC inserts zeros in the unfilled character positions before the word is transferred to memory.

### Transfer Capacity

The CPC can transfer any number of characters from 1 to 49,152 (16,384-word memory at three characters/word) between a peripheral and the DATANET-30 memory. Any starting address within a 16k memory may be specified. It is not necessary to specify the number of characters in multiples of 3. The CPC accesses memory sequentially under memory interrupt control.

### Block Diagram

Figure G-2 shows the basic configuration of the CPC. It contains a shift register, two 18-bit word buffers, an address counter, a character counter, peripheral status registers, and command-sequence and data-transfer control logic.

The CPC has three interfaces: (1) common peripheral interface (CPI), (2) DATANET-30 buffer selector, and (3) DATANET-30 interrupt interface.

The common peripheral interface provides the necessary data, status, and control logic to allow operation of the standard line of peripherals. The peripherals detect no difference between a common peripheral channel on a GE-400 or GE-600 Series computer and the CPC on the DATANET-30.

The DATANET-30 buffer selector interface presents peripheral and CPC status to the program and accepts control signals and command words from the program. One of the 127 possible buffer channel addresses must be assigned.

The DATANET-30 interrupt interface is used for data transfer between the CPC and the DATANET-30 memory locations specified by the address counter. The interrupt cycles to be assigned to a peripheral device (the CPC) are switch selectable and should not coincide with interrupts assigned to another device.

### Character Counter

The character counter is used for read and write operations. It is counted down 1 each time a character is transferred to or from the CPC. An End Data Transfer signal is generated and sent to the peripheral device when the counter equals a count of zero.

The character counter is a straight binary counter. When loading the counter, the binary 1's complement must be transferred to the counter. Therefore, the number (CW3) must be complemented before it is loaded into the character counter. The counter is loaded with the third LDT instruction of the initializing instructions sequence.

### Address Counter

The CPC addresses memory sequentially, according to the address in the address counter. The address counter is counted up 1 each time a DATANET-30 word is transferred to or from memory. The address counter must be loaded with the starting memory address of data to be transferred or received. The counter is loaded by the second LDT instruction in the sequence of instructions to the CPC.

### Parity

Parity (odd) is generated on each character before being transferred to the peripheral device. Parity is checked by the CPC on each character received from the peripheral device. In case of an error, NES 5 is set. A parity error does not affect CPC operation. NES 5 is reset by the next instruction to the CPC.

### Controls and Indicators

The interrupt cycle select switches and the manual reset switch on the control panel of the DATANET-30 are the only manual controls for the CPC. Pressing the manual reset switch of the DATANET-30 also resets the peripheral device connected to the CPC. All other controls and indicators are by program or are on the control panels of the peripheral equipment.

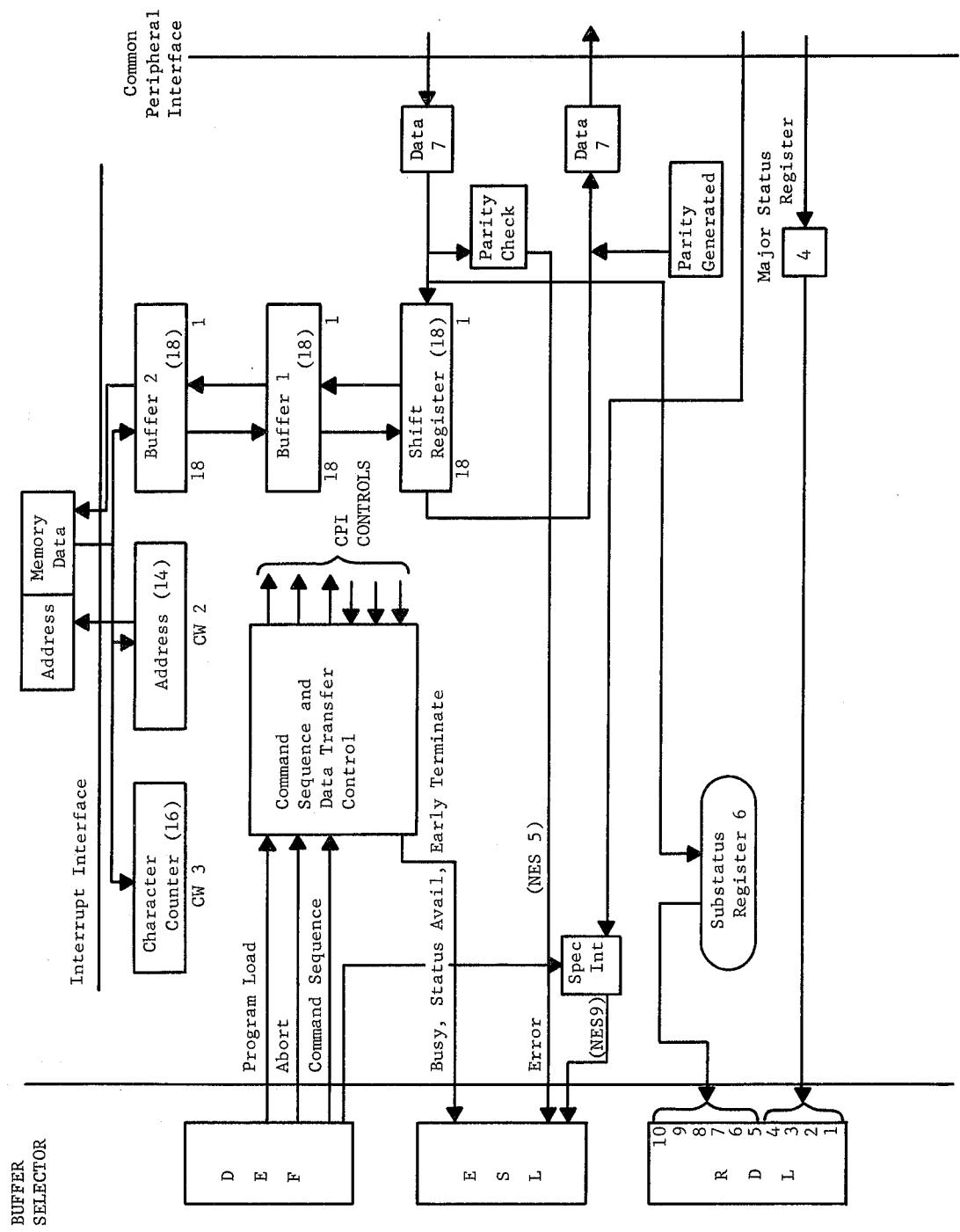


Figure G-2. Block Diagram

## Interrupt Cycles

The CPC operates only with a DATANET-30 which has the interrupt interface (Mod III). The interrupt interface uses the DATANET-30 timing to create groups of five memory cycles which, in turn, can be individually assigned to devices such as the CPC. Therefore, when the CPC needs a memory cycle to transfer data, it can obtain the assigned cycle and be guaranteed no interference.

Five toggle switches on the connector bracket (Figure G-3) assign cycles to the CPC. Any one switch allows a transfer rate up to 86.4kc. Any two switches allow a transfer rate up to 172.8kc. The selected switch designates which interrupt cycle will be available to the CPC for data transfer.

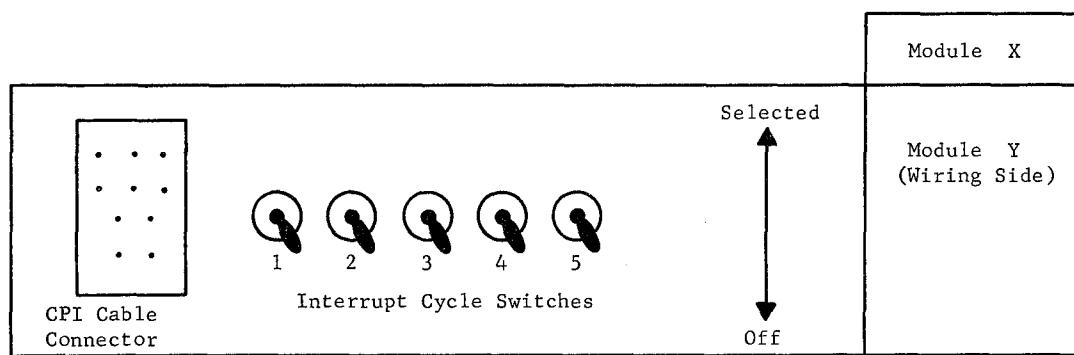


Figure G-3. Cable Connector and Switch Bracket

## Cycle Assignment

To assign cycles to a CPC, the following procedure is used: (1) determine the maximum transfer rate of the peripheral, (2) determine the number of cycles to be assigned (1 cycle up to 86.4kc, 2 cycles up to 172.8kc), and (3) assign the cycles by lifting the proper switches on the bracket.

If two cycles out of the five are needed for a peripheral, do not assign two consecutive cycles. This is necessary to avoid losing data when an interrupt is not granted in time. Use combinations 1-3, 1-4, 2-4, 2-5, or 3-5. If a Controller Selector Unit (CSU-931) is also connected to the DATANET-30, cycles 1 and 3 are assigned to the CSU. These cycles cannot be used by the CPC.

Normally, the same memory interrupt cycle number is not assigned to two different CPC modules or other modules connected to the interrupt interface, such as the Controller Selector Unit (CSU-931) or the Processor Interrupt Unit (PIU). With careful programming, two peripherals can be assigned the same cycle if a program interlock prevents using both peripherals at the same time.

## Status and Substatus

Status conditions are presented to the DATANET-30 on received data lines 1 through 10.

Receive Data Line	10	9	8	7	6	5	4	3	2	1		
	$2^6$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$	$2^3$	$2^2$	$2^1$	$2^0$		
	Substatus						Major Status					

Major status lines 1, 2, and 3 are set by the peripheral equipment and remain set until the first LDT instruction of a new command sequence.

Major status line 4 (Busy) changes as the busy status of the peripheral changes.

Substatus indicated at the CPC does not change immediately with substatus changes at the peripheral. Substatus is set in the substatus register after each command sequence and command termination and remains set until changed as a result of another command sequence or a Request Status instruction. The substatus of a peripheral may change many times without this change being indicated at the CPC.

When the program desires the status of the peripheral and NES 4 indicates status available, a register transfer instruction from R to any register (such as TRA R,B) transfers the last presented major status and the last presented substatus information into the register for examination.

The NES instructions provide status information regarding overall operation of the CPC and the peripheral device.

The various major status and substatus conditions for the different peripheral equipment are summarized later in this appendix.

Refer to the applicable equipment manual for additional information on status.

## Special Interrupt

A Special Interrupt signal (NES 9) is generated by the peripheral equipment for various reasons. This does not cause a program interrupt but sets NES 9 to nonzero. Programming considerations for the peripheral dictate action required when a special interrupt occurs. Refer to the applicable peripheral manual for the conditions relating to the Special Interrupt signal. A DEF 9 is used to reset NES 9.

## INSTRUCTION REPERTOIRE

The instructions are classified under two groups: the buffer selector instructions for the DATANET-30 and the peripheral command words. The buffer selector instructions are the Drive External Function (DEF), External Status Lines (NES), and Register Transfer (TRA and LDT) instructions. The rules for using these instructions are the same as those for using other DATANET-30 buffer selector instructions. The peripheral command words are used with the LDT instruction of the DATANET-30.

### DATANET-30 Instructions

DEF INSTRUCTIONS. The Drive External Function instructions are as follows:

DEF 3 - Program Load  
DEF 8 - Abort  
DEF 9 - Reset Special Interrupt  
DEF 0 - Command Sequence  
DEF 1-2 - Not Assigned  
DEF 4-7 - Not Assigned

DEF 3 - Program Load--Activates the program load line to the peripheral. (Refer to the applicable peripheral manual for results.) The DEF 3 instruction must always be preceded by manual reset or a Request Status operation in order to clear the address and character counters in the CPC. The information from the peripheral is then entered into memory, starting with address 0.

DEF 8 - Abort--Causes the immediate termination of any command in progress by forcing an EDT, if given immediately after the command sequence. If the instruction is given during data transfer, the EDT is not issued until the end of any third character (end of word) is transferred in or out of the shift register. This instruction also results in a pseudo "manual clear" to the CPC, and all CPC registers are cleared to zero. The early terminate line is activated if all the data has not been transferred.

DEF 9 - Reset Special Interrupt--Must be used by the program to reset a special interrupt (NES 9) signal.

DEF 0 - Command Sequence--Activated automatically by the LDT instruction used for initiating commands. The program cannot use DEF 0.

NES INSTRUCTIONS. Nonzero indicates the following:

NES 1 - Busy  
NES 4 - Status Available

NES 5 - Receive Data Error  
NES 8 - Early Termination  
NES 9 - Special Interrupt  
NES 0 - Not Assigned  
NES 2-3 - Not Assigned  
NES 6-7 - Not Assigned

Nes 1 - Busy--A 1 indicates the CPC is not available to receive a new command. The CPC reflects the busy state after CW 1 is given and until the 2<sup>3</sup> major status bit (peripheral busy) from the peripheral is reset or until buffer 1 and buffer 2 are empty on a read operation.

Ready--A zero indicates the CPC can receive a new command.

NES 4 - Status Available--Indicates the program can transfer the current major status and the last substatus information into a DATANET-30 register for examination. The status is not available when the CPC is transferring commands to the peripheral.

NES 5 - Receive Data Error--Indicates a parity error was detected on the data received from the peripheral during the last command.

NES 8 - Early Termination--Indicates the last data transfer was terminated by the peripheral before all characters were transferred and before the character counter reached zero.

NES 9 - Special Interrupt--Indicates the peripheral activated the special interrupt line. This signal must be reset by DEF 9.

The NES signals for Status Available, Receive Data Error, and Early Termination are reset by the next command word.

#### REGISTER TRANSFER INSTRUCTIONS. The Register Transfer instructions are as follows:

TRA From, TØ Major status and substatus are transferred from the CPC.

TRA R,

LDT This instruction is used to transfer command words 1, 2, and 3 to the CPC.

TRA B,T Not used.

#### Example:

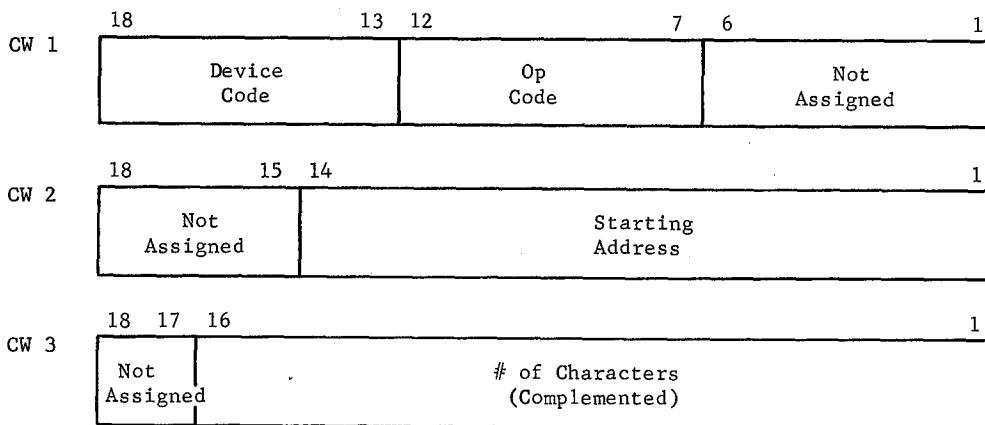
    NES 4                  Is status available?  
    BZE \*-1                No wait.  
    TRA R,B                Transfer major status and substatus to B-register.

## Command Words

Peripherals respond to two types of commands: data transfer and non-data-transfer. Refer to the applicable programming manual for the specific instructions to use.

**DATA TRANSFER COMMANDS.** The CPC requires three command words to initiate and execute a data transfer command. The command words are provided to the CPC by three LDT instructions. For data transfer, the command words must be given in the sequential order of CW 1, CW 2, and CW 3.

The formats of the command words are as follows:



Command Word 1--Bits 7-12 contain the operation code. Bits 13-18 contain the device code. Bits 1-6 are not assigned and may be used by the program.

Command Word 2--Bits 1-14 contain the starting address from or to which data transfer is to take place. Bits 15-18 are not assigned and may be used by the program.

Command Word 3--Bits 1-16 contain the 1's complement of the number of characters to be transferred. Bits 17-18 are not assigned and may be used by the program.

On receipt of CW 3, the CPC initiates the I/O sequence to send the device code and the operation code to the peripheral. The Busy external status line indicates busy until termination of the command. Termination may occur for one of five reasons:

1. The character counter indicates all the characters have been transferred.
2. The peripheral sends a Terminate signal indicating it has reached the end of a record, card, or other unit of data as prescribed for the particular peripheral device.
3. The peripheral sends a Terminate signal indicating some error or alert condition has occurred which requires stopping the data transfer.
4. The program issues an Abort command.
5. The DATANET-30 enters Hardware Load.

On receipt of the Terminate signal for any of the first three reasons during a read command, the CPC left-justifies the characters in the shift register prior to transferring the word to memory. On a write command, data transfer stops immediately. An Abort or Hardware Load causes termination, with no further data transfers to memory. Coding to initiate a data transfer command is as follows:

PIC CPCADD	Select CPC channel
NES 1	Test busy line
BNZ Busy	
LDT CW1	Load CW 1
LDT CW2	Load CW 2
LDT CW3	Load CW 3
NES 4	Test status available
BZE NOT	
TRA R,B	Pick up status for examination

Since the CPC interfaces with different types of peripheral equipment, any additional command information needed by a peripheral device must be contained in the first words transferred to the peripheral.

NON-DATA-TRANSFER COMMANDS. A non-data-transfer command requires only CW 1 from the program. The CPC decodes the non-data-transfer condition and initiates the I/O sequence without requiring CW 2 and 3. The command, by definition, terminates at the end of the I/O sequence. The busy line indicates busy during the I/O sequence. The coding is as follows:

PIC CPCADD	Select CPC channel
NES 1	Test busy
BNZ Busy	
LDT CW1	Load CW 1
NES 4	Test status availale
BZE NOT	
TRA R,B	Pick up status for examination

The non-data-transfer command operation codes may be recognized by the fact that either the high-order bit = 1 or the four low-order bits = zero (1x0000).

## Programming Considerations

In programming the CPC, the following factors must be considered:

1. The frequency for checking the special interrupt line should be determined for each peripheral.
2. The program must reset the Special Interrupt signal.
3. Status recovery procedures depend upon the peripheral equipment.
4. When Request Status and Reset Status are used, the rules of the peripheral in use must be followed. Request Status cannot be given during data transfer.
5. The peripheral instructions are not recognized by the assembly program in mnemonic form. The instruction mnemonics are included for reference only. Each peripheral instruction must be established in memory in octal form (machine language). Command words may then be referred to symbolically for the instruction desired.
6. The extra information required by the peripheral equipment not contained in CW 1 must be located in the first memory location(s) transferred to the peripheral.
7. The instructions for the various peripheral devices are included here for reference only. Refer to the applicable manual for detailed explanation of the instructions and additional information.

## General Electric Standard Character Set

Standard Character Set	GE-Internal Machine Code	Octal Code	Hollerith Card Code	Standard Character Set	GE-Internal Machine Code	Octal Code	Hollerith Card Code
0	00 0000	00	0	↑	10 0000	40	11-0
1	00 0001	01	1	J	10 0001	41	11-1
2	00 0010	02	2	K	10 0010	42	11-2
3	00 0011	03	3	L	10 0011	43	11-3
4	00 0100	04	4	M	10 0100	44	11-4
5	00 0101	05	5	N	10 0101	45	11-5
6	00 0110	06	6	O	10 0110	46	11-6
7	00 0111	07	7	P	10 0111	47	11-7
8	00 1000	10	8	Q	10 1000	50	11-8
9	00 1001	11	9	R	10 1001	51	11-9
[	00 1010	12	2-8	-	10 1010	52	11
#	00 1011	13	3-8	\$	10 1011	53	11-3-8
@	00 1100	14	4-8	*	10 1100	54	11-4-8
:	00 1101	15	5-8	)	10 1101	55	11-5-8
>	00 1110	16	6-8	;	10 1110	56	11-6-8
?	00 1111	17	7-8	:	10 1111	57	11-7-8
£	01 0000	20	(blank)	+	11 0000	60	12-0
A	01 0001	21	12-1	/	11 0001	61	0-1
B	01 0010	22	12-2	S	11 0010	62	0-2
C	01 0011	23	12-3	T	11 0011	63	0-3
D	01 0100	24	12-4	U	11 0100	64	0-4
E	01 0101	25	12-5	V	11 0101	65	0-5
F	01 0110	26	12-6	W	11 0110	66	0-6
G	01 0111	27	12-7	X	11 0111	67	0-7
H	01 1000	30	12-8	Y	11 1000	70	0-8
I	01 1001	31	12-9	Z	11 1001	71	0-9
&	01 1010	32	12	→	11 1010	72	0-2-8
:	01 1011	33	12-3-8	,	11 1011	73	0-3-8
]	01 1100	34	12-4-8	%	11 1100	74	0-4-8
(	01 1101	35	12-5-8	=	11 1101	75	0-5-8
<	01 1110	36	12-6-8	"	11 1110	76	0-6-8
\	01 1111	37	12-7-8	!	11 1111	77	0-7-8



## DISC STORAGE UNIT (DSU) SUBSYSTEM

Information on the DS-20 Disc Storage Unit (DSU) subsystem not provided in this discussion may be found in the DS-20 subsystem reference manuals.

### DS-20 Subsystem Instructions

The DS-20 subsystem instructions are summarized in the following chart.

INSTRUCTION	MNEMONIC	CODE (OCTAL)	INSTRUCTION	MNEMONIC	CODE (OCTAL)
Seek File	SF	34	Write File and Verify	WFV	53
Seek/Read File	SRF	14	Read File Continuous	RFCR	25
Read File	RF	54	Write File Continuous	WFCR	31
Seek/Read File and Release Seek	SRFR	15	Write File Continuous and Verify	WFCV	33
Read File and Release Seek	RFR	55	Seek/Compare	SCPR	17
Seek/Read File and Increment Address	SRFI	16	Compare	CPR	57
Read File and Increment Address	RFI	56	Seek/Link	SLNK	35
Seek/Write File	SWF	10	Link	LNK	75
Write File	WF	50	Accept Buffer Address	ABA	32
Seek/Write File and Release Seek	SWFR	11	Read Buffer	RB	24
Write File and Release Seek	WFR	51	Write Buffer	WB	30
Seek/Write File and Increment Address	SWFI	12	Load Buffer for Compare	LBFC	36
Write File and Increment Address	WFI	52	Move Data	MVDT	37
Seek/Write File and Verify	SWFV	13	Request Status	RQS	00
			Reset Status	RSS	40

## Physical Organization of DSU Subsystem

Data flow in a DSU subsystem is shown in Figure G-4.

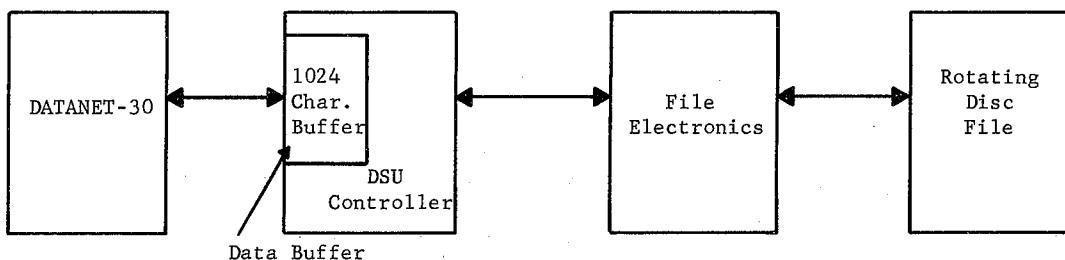


Figure G-4. Block Diagram of Data Flow of DSU Subsystem

DSU CONTROLLER. The controller is a single free-standing enclosure. This unit links the DSU to the DATANET-30. Up to four DSU's can be connected to one DSU controller (one to four devices).

DSU. A DSU consists of a rotating assembly subunit (file unit) and a file electronics subunit. The rotating assembly subunit houses the rotating discs. The file electronics houses the electronics necessary for direct control of the file unit.

## Logical Organization of DSU Controller

The DSU controller is organized around a 1024-character core buffer made up of four data blocks.

The logical subdivision of the buffer must be clearly understood by anyone concerned with programming the DSU subsystem.

The following table relates the buffer addresses (in octal) and their functions:

<u>Addresses</u>	<u>Function</u>
0000-0357	Buffer section W data block storage; contains one data block.
0360	Check character for data block in buffer section W.
0361-0377	Buffer section W control block storage. Control characters for the DSU on file channel 1, (device 1) are stored here. The first nine addresses (0361-0371) contain storage for file electronics subunit 1. The remaining six addresses (0372-0377) contain the control characters for device 1. (See Figure G-5.)
0400-0757	Buffer section X data block storage.
0760	Check character for data block in buffer section X.

<u>Addresses</u>	<u>Function</u>
0761-0777	Buffer section X control block storage. Used by device 2 (control characters stored in 0772-0777).
1000-1357	Buffer section Y data block storage.
1360	Check character for data block in buffer section Y.
1361-1377	Buffer section Y control block storage. Used by device 3 (control characters stored in 1372-1377).
1400-1757	Buffer section Z data block storage.
1760	Check character for data block in buffer section Z.
1761-1777	Buffer section Z control block storage, used by device 4 (control characters stored in 1772-1777).

The check character is generated when a record is written on a disc. It is a modulo-64 count of the number of 1-bits in the data block being written. During read operations, the check character is transferred to an applicable check character section of the buffer.

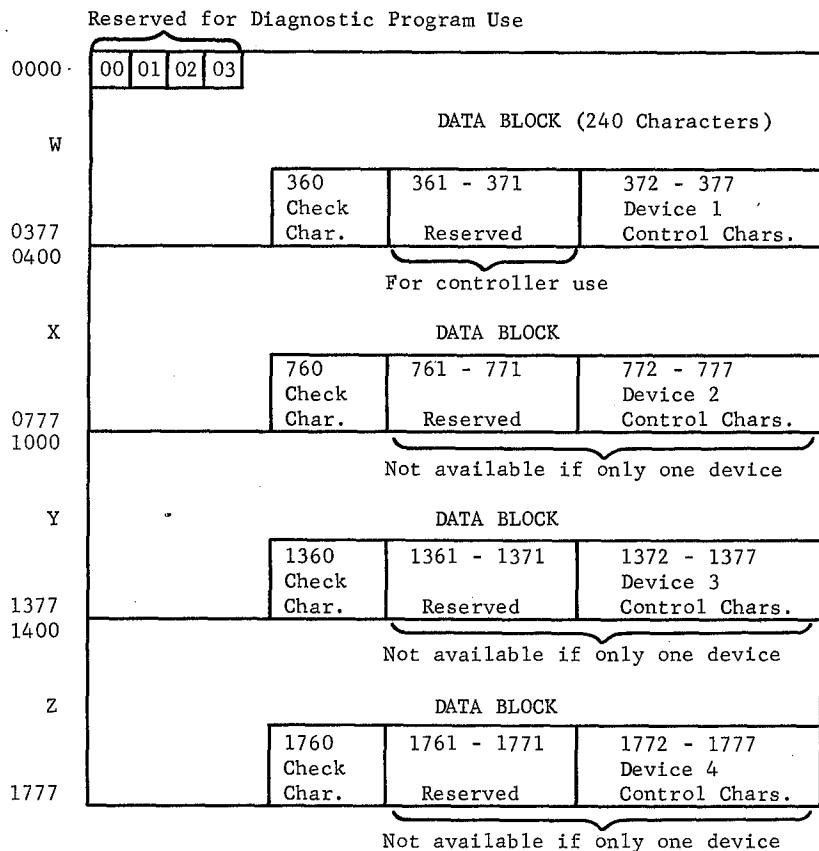


Figure G-5. Buffer Addresses and Functions of the 1024-Character Buffer

When a command requiring control characters is addressed to a particular device, the applicable control block section of the buffer is used, as shown below:

Device 1	Buffer section W control block
Device 2	Buffer section X control block
Device 3	Buffer section Y control block
Device 4	Buffer section Z control block

Note that the buffer section data blocks are not restricted to any one device. Thus, device 1 can read or write to any or all of the buffer section data blocks. Once a write or read operation is initiated, the controller logic commits the applicable buffer section data blocks to the particular device being used. When the operation is completed, the buffer sections are released for the commitment. Thus, buffer sections Y and Z can be used for device 1, while buffer sections W and X are being used for device 2.

Note: A special core checkout switch (the SPCCIR switch) has been added for testing the DSU buffer. When this switch is ON, the buffer is logically just a 1024-character buffer. The various addresses and functions noted above are overridden.

The first four consecutive data blocks starting at address 00 should never be considered as data block addresses for operating programs. These addresses are used by diagnostic programs.

### Device Code

The DS-20 subsystem requires a device code character indicating the specific file unit to which the instruction is directed. The device code characters are defined as follows:

Device Code	Meaning
00	Instruction involves controller only *
01	Instruction is addressed to file unit 1
02	Instruction is addressed to file unit 2
03	Instruction is addressed to file unit 3
04	Instruction is addressed to file unit 4

\*Instructions not involving a specific file unit ignore the device code character.

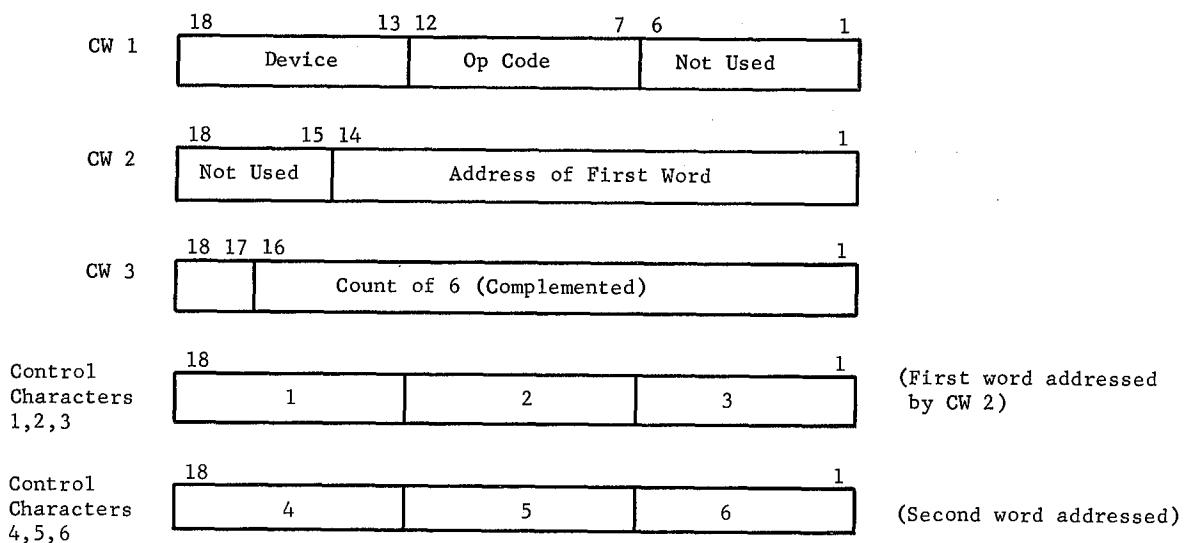
### Control Characters

An important consideration for programming purposes is that the instructions can be divided into two groups. One group requires the transfer of six control characters as part of the command sequence. The other group does not require the transfer of six control characters but uses certain control characters already in the controller buffer.

The details for coding the control characters cannot be generalized. The coding depends upon the instructions. A brief discussion is included here, but for more information regarding each instruction refer to the DS-20 subsystem reference manual.

The six control characters are contained in two DATANET-30 words. These words are transferred to the DSU following the command sequence (CW 1, 2, 3). The control characters must be considered as part of the command sequence for the DS-20 subsystem.

For example:



Since the character counter contains the count of 6, the command sequence ends at the DATANET-30. The controller then executes the command.

COMBINED ON-LINE/OFF-LINE COMMANDS. Combined on-line/off-line instruction execution involves a time on line while address-definition (control) characters are being transferred from computer memory to the controller and into a buffer control block. The remainder of execution time is off line, using the address-definition characters stored in the buffer control block assigned to the file unit being addressed by the instruction.

The seek instructions listed below result in a combined on-line/off-line type of operation:

- Seek File
- Seek/Read File
- Seek/Read File and Release Seek
- Seek/Read File and Increment Address
- Seek/Write File
- Seek/Write File and Release Seek
- Seek/Write File and Increment Address
- Seek/Write File and Verify
- Seek/Compare
- Seek/Link

Upon receipt of an on-line/off-line instruction, the controller reverts to a subsystem Busy status and initiates a request for six control characters from the central processor. The control characters are stored in the controller in the last six character positions of the control block of the buffer section assigned to the file unit being addressed by the instruction.

All six control characters must be received by the DS-20 subsystem. If it receives the End Data Transfer signal before all control characters are received, instruction execution ends, and the Data Alert, Invalid Control Character status condition is set in the controller.

If all control characters are received and stored in the buffer and no errors are detected, the controller reverts to a Ready status and transmits a not busy signal to the central processor to indicate that the on-line portion of instruction execution is complete.

The control characters are encoded as follows:

- |                        |  |
|------------------------|--|
| Characters 1, 2, and 3 | - These characters indicate the address of the file sector to be accessed during execution of the instruction. |
| Characters 4, 5, and 6 | - The encoding of these characters is determined by the instruction being used.                                |

#### CONTROL CHARACTERS 1, 2, and 3.

Sector Addressing. There is no format restriction of the 240 data characters within a sector of data. These characters are generated by the central processor. Sectors are continuously addressable within each individual file unit. Up to 32 contiguous sectors are continuously addressable before an actuator arm movement occurs.

The check character is generated by the controller during a write operation and is automatically written on the file unit as the last (241st) character of each sector being written. The controller generates the check character by making a modulo-64 count of the number of 1-bits in the sector being written. The modulo-64 check character represents the remainder from dividing the number of 1-bits stored in the sector by 64.

The check character is never transmitted to the central processor during a normal read operation. However, the check character may be transferred to the central processor by first issuing an Accept Buffer Address and then a Read Buffer instruction.

The bit configuration for a data sector address within any one file unit is illustrated in Figure G-6.

This configuration of address bits provides for continuous sector addresses through an entire file unit. For any one specified disc and actuator position, a zone code of 00 addresses the 32 sectors under the four heads covering both the upper and lower surfaces of the disc in the inner zone. A zone code of 01 addresses the 32 sectors under the two heads covering the upper surface of the disc in the outer zone. A zone code of 10 addresses the 32 sectors under the two heads covering the lower surface of the disc in the outer zone.

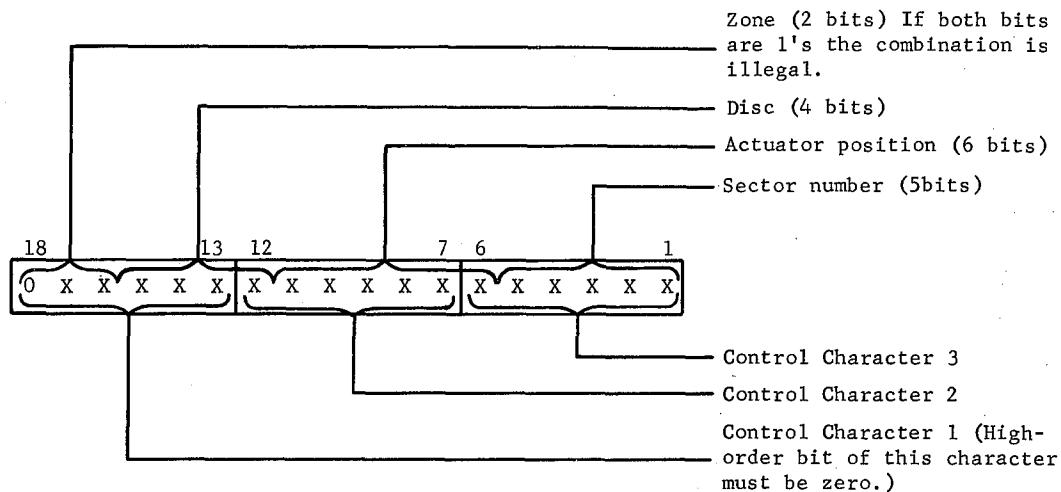


Figure G-6. Sector Address Control Character Format

Any data sector whose address contains a 1 in all five low-order bit positions is the last consecutively addressable sector available from a file unit without requiring repositioning of the actuator. Instruction execution always ends after the last consecutive sector is accessed.

The four consecutive data sectors on each file unit, starting at address 00 0000 000000 000000, are reserved for diagnostic program purposes and should not be used by the programmer. Although there is no hardware to prevent a reserved diagnostic sector from being accessed, in using these sectors there is considerable risk of having a diagnostic program overwrite the contents of the reserved sectors.

Fast-Access Disc Addressing. Each disc used in a fast-access option is addressed by expressing its number, in binary form, in the four disc bits of the file address. Discs 12 through 15 are used for fast-access option 1, and discs 8 through 15 are used for fast-access option 2. The six actuator position bits of the file address must be zero when addressing a fast-access disc. Each fast-access disc contains three groups of data, one group for each of the three legal combinations of zone codes in the file address. Each of the three groups contains 32 sectors of data.

The central processor must transmit the sector address to the DS-20 subsystem to indicate which data sectors are to be accessed.

The following charts show the coding used for control characters 1, 2, and 3 as defined in Figure G-6.

DISC BITS

<u>Code</u>	<u>Disc</u>
0000	0
0001	1
0010	2
0011	3
1000	8
1111	15

ZONE BITS

<u>Code</u>	<u>Zone</u>
00	Inner Zone Both
01	Outer Zone Top
10	Outer Zone Bottom
11	Illegal

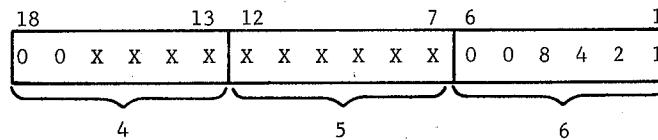
ACTUATOR POSITION BITS

<u>Code</u>	<u>Position</u>
000000	0
000001	1
000010	2
001000	8
010000	16
100000	32
111111	63

SECTOR (BLOCK) NUMBER BITS

<u>Code</u>	<u>Sector</u>
00000	0
00001	1
01000	8
10000	16
11111	31

CONTROL CHARACTERS 4, 5, AND 6. The details for encoding characters 4, 5, and 6 (see sketch, below) cannot be generalized. The encoding for these characters is determined by the command to be executed.



### Input/Output Control

Before a read or write operation can be executed, a seek operation must be done to position the actuator properly for the segment being addressed.

Upon completion of the seek, the read or write operation can be executed.

Data read from the disc is stored in the data buffer.

Upon completion of the read, the data is transferred from the data buffer to memory of the DATANET-30.

The write operation is essentially the reverse of the read. Examples of the instruction sequence for a read are shown in Figure G-7.

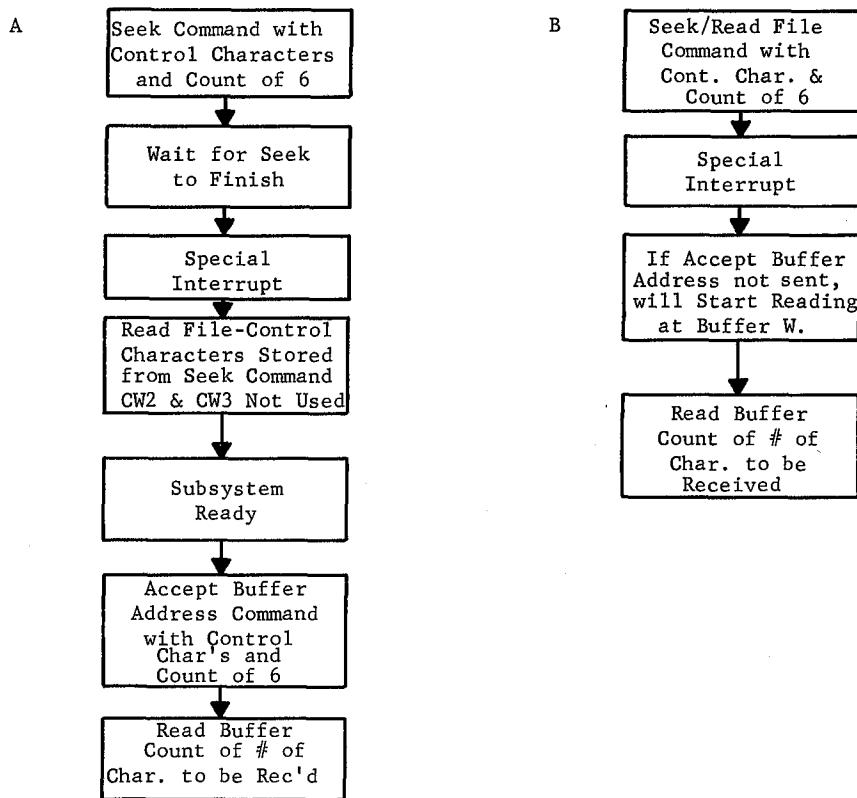


Figure G-7. Examples of Instruction Sequence for a DS-20 Read Instruction

The off-line portion of instruction execution begins as soon as the Subsystem Not Busy signal is sent to the CPC. The off-line operation to be executed is defined by the control characters stored in the assigned buffer control block.

If an error is detected in the control characters during the on-line portion of instruction execution, the off-line portion is not initiated. Instead, the controller reverts to the appropriate error status condition and sends the not busy signal to the central processor. The outcome of instruction execution can be determined by issuing a Request Status instruction.

As soon as the off-line portion of an instruction is executed, the DSU sends a Special Program Interrupt signal to the central processor. When the Special Program Interrupt signal is received by the central processor, the outcome of execution of the off-line portion of the instruction can be determined by addressing a Request Status instruction to the file unit involved.

While an instruction is being executed off-line, the controller is free to accept and execute other instructions, provided that execution of these other instructions does not require the use of a file unit previously committed to an off-line operation.

Combined on-line/off-line instructions, with the exception of Seek File, are the only instructions that cause the buffer sections involved to be committed. The involved buffer sections remain committed until the off-line portion of instruction execution ends. The Seek File instruction does not commit the involved buffer sections.

### Instructions Requiring Transfer of Control Characters

The commands requiring transfer of control characters as part of the command sequence are again divided as follows:

#### Controller Only.

<u>MNEMONIC</u>	<u>DESCRIPTION</u>
ABA	Accept Buffer Address Sends buffer address to controller. Control characters: Characters 1 and 2 contain the buffer address. Characters 3, 4, 5, and 6 are ignored.
MVDT	Move Data Moves up to 240 characters from one address to another. Control characters: Characters 1 and 2 contain "to" address. Characters 3 and 4 contain number of characters. Characters 5 and 6 contain "from" address.

#### File Only.

SF	Seek File Seeks the block specified by control characters 1, 2, and 3. Control Characters: Characters 1, 2, and 3 contain the block address. Characters 4, 5, and 6 are not pertinent to this command, but may be significant to subsequent commands.
SRF	Seek/Read File Seeks the block specified by control characters 1, 2, and 3. When the block is found, blocks are read into the buffer sections specified by control characters 5 and 6. From one to four blocks can be read. Control Characters: Characters 1, 2, and 3 contain the block address. Character 4 is not pertinent. Character 5 indicates the buffer sections into which file blocks are to be read: 1 in $2^0$ position indicates buffer section Z 1 in $2^1$ position indicates buffer section Y 1 in $2^2$ position indicates buffer section X 1 in $2^3$ position indicates buffer section W

<u>MNEMONIC</u>	<u>DESCRIPTION</u>
	<p>Character 6 indicates the buffer section into which the first data block read is to be placed:</p> <ul style="list-style-type: none"> <li>1 in <math>2^0</math> position indicates buffer section Z</li> <li>1 in <math>2^1</math> position indicates buffer section Y</li> <li>1 in <math>2^2</math> position indicates buffer section X</li> <li>1 in <math>2^3</math> position indicates buffer section W</li> </ul> <p>The two high-order bits of character 6 must be zero.</p>
SRFR	<p>Seek/Read File and Release Seek</p> <p>Identical with the SRF command until termination of the command is reached. At this time, the power to the actuator involved is dropped.</p> <p>Control Characters:</p> <ul style="list-style-type: none"> <li>Same as those for SRF.</li> </ul>
SRFI	<p>Seek/Read File and Increment Address</p> <p>Identical with the SRF command until termination of the command is reached. At this time, the block address in the control characters is counted up to the next consecutive address. Thus, if two blocks were read, the original block address is counted up by two.</p> <p>Control Characters:</p> <ul style="list-style-type: none"> <li>Same as those for SRF.</li> </ul>
SWF	<p>Seek/Write File</p> <p>Seeks the block specified by control characters 1, 2, and 3. When the block is found, blocks are written from the buffer sections specified by control characters 5 and 6. From one to four blocks can be written.</p> <p>Control Characters:</p> <ul style="list-style-type: none"> <li>Characters 1, 2, and 3 contain the block address.</li> <li>Character 4 is not applicable.</li> <li>Character 5 indicates the buffer sections from which data is to be written on the file. (Character 5 coding is identical with that for character 5 for the SRF command.)</li> <li>Character 6 indicates the buffer section from which the first data block to be written is taken. (Character 6 coding is identical with that for character 6 for the SRF command.)</li> </ul>
SWFR	<p>Seek/Write File and Release Seek</p> <p>Identical with the SWF command until termination of the command is reached. At this time, the power to the actuator involved is dropped.</p> <p>Control Characters:</p> <ul style="list-style-type: none"> <li>Same as those for SWF.</li> </ul>
SWFI	<p>Seek/Write File and Increment Address</p> <p>Identical with the SWF command until termination of the command is reached. At this time, the block address in the control characters is counted up to the next consecutive address. Thus, if three blocks were read, the original block address is counted up by 3.</p> <p>Control Characters:</p> <ul style="list-style-type: none"> <li>Same as those for SWF.</li> </ul>

<u>MNEMONIC</u>	<u>DESCRIPTION</u>
SWFV	<p>Seek/Write File and Verify</p> <p>Seeks the block specified by control characters 1, 2, and 3. When the block is found, blocks are written from the buffer sections specified by control characters 5 and 6. From one to four blocks can be written. At the conclusion of the write operation, a latency is incurred and the blocks written on the file are compared bit by bit with the original block information stored in the buffer.</p> <p>Control Characters:</p> <p>Same as those for SWF.</p>
SCPR	<p>Seek Compare</p> <p>Seeks the block specified by control characters 1, 2, and 3. When the block is found, it is compared to the buffer section specified by control character 6. Comparison is made on the basis of "fields." A field is defined as a group of sequential characters bounded by Ignore characters (octal 17) or bounded by an Ignore character and the beginning of the data block or bounded by an Ignore character and the end of the block. A field may contain as many fields as desired within the limits defined above. The compare operation is considered a success if <u>any one field</u> of the buffer data block compares exactly with the corresponding field of the file data block. If the compare is not successful, the next sequential file block is read and compared. This continues until the last consecutive block has been read or a successful compare is made. If the compare is successful, blocks are read into the buffer sections specified by control character 5. The block address of the file block for which the successful compare was made will replace control characters 1, 2, and 3 in the file control block.</p> <p>Control Characters:</p> <p>Characters 1, 2, and 3 contain the block address. Character 4 is not applicable.</p> <p>Character 5 indicates the buffer sections into which blocks are to be written if a compare is made. (Character 5 coding is identical with that for character 5 for SRF command.)</p> <p>Character 6 indicates the buffer section to which the file blocks are to be compared and, if a comparison is being made, indicates the first buffer section into which the first file data block read is to be placed. (Character 6 coding is identical with that for character 6 for the SRF command.)</p>
SLNK	<p>Seek Link</p> <p>Seeks the block specified by control characters 1, 2, and 3. When the block is found, it is read from the file and written in the buffer section indicated by control character 6. At this time, three characters are pulled from the buffer, starting at the buffer location specified in control characters 4 and 5. If the high-order bit of the first character read is a 1, the operation is terminated. If, however, the high-order bit is a 0, the three characters are stored in the applicable control block section of the buffer, replacing the three characters of file address (that is, control characters 1, 2, and 3).</p>

**Control Characters:**

Characters 1, 2, and 3 contain the block address. Characters 4 and 5 contain the buffer address of the first of the three characters to be checked for link. Character 6 indicates the buffer section into which the file block is to be read. (Character 6 coding is identical with that for character 6 for the SRF command.)

**Instructions Not Requiring Transfer of Control Characters**

The commands that require certain control characters but do not request control characters as part of the command sequence from the DATANET-30 are divided as shown below.

The control characters needed by each of the following commands must be part of the command preceding it:

**Controller Only.**

**MNEMONIC**

**DESCRIPTION**

RB

Read Buffer

Read the buffer, using a stored address as the starting address. Data read is sent to the DATANET-30 memory.

WB

Write Buffer

Writes the buffer, using the stored address as the starting address.

LBFC

Load Buffer For Compare

Writes up to 240 characters in one data block. If there are fewer than 240 characters in one data block, the rest of the block is filled with characters (octal 17).

**File Only.**

RF

Read File

The applicable control block section of the buffer provides the control characters for this command. The control characters were placed there by a previous command. Once the control characters are obtained, the command is identical with the SRF command.

RFR

Read File and Release Seek

Identical with the RF commands until termination of the command is reached. At this time, the power to the actuator involved is dropped.

RFI	Read File and Increment Address Identical with the RF command until termination of the command is reached. At this time, the block address in the control characters is counted up to the next consecutive block.
WF	Write File The applicable control block section of the buffer provides the control characters for this command. The control characters were placed there by a previous command. Once the control characters are obtained, the command is identical with the SWF command.
WFR	Write File and Release Seek Identical with the WF command until the termination of the command is reached. At this time, the power to the actuator involved is dropped.
WFI	Write File and Increment Address Identical with the WF command until termination of the command is reached. At this time, the block address in the control characters is counted up to the next consecutive address.
WVF	Write File and Verify The applicable control block section of the buffer provides the control characters for this command. The control characters were placed there by a previous command. Once the control characters are obtained, the command is identical with the SWF command.
RFCR	Read File Continuous The applicable control block section of the buffer provides the control characters for this command. The command seeks the block specified by control characters 1, 2, and 3. At the completion of the seek command, the block is read into buffer section W. As soon as W is filled, the data is transferred to the DATANET-30 memory. The next sequential file block is then read into buffer section X. As soon as X is filled, the data is transferred to the DATANET-30 memory. This continues in serial fashion until one of the following occurs: <ul style="list-style-type: none"> <li>1. An error condition is detected.</li> <li>2. The last consecutive block is read.</li> <li>3. The CPC sends an End Data Transfer. (Note that 32 blocks can be read in this fashion.)</li> </ul>

WFCR

Write File Continuous

The applicable control block section of the buffer provides the control characters for this command. The control characters were placed there by a previous command. Once the control characters are obtained, the command seeks the block specified by control characters 1, 2, and 3. At the completion of the seek command, buffer section W is filled with data from the external user. As soon as buffer section W is filled, the data is transferred to the file block. Buffer section X is then filled and the next sequential file block is written from X. This continues in a serial fashion until one of the combinations listed under the RFCR command occurs. (Note that 32 blocks can be written in this fashion.)

WFCV

Write File Continuous and Verify

This command is identical with the WFCR command, with the following exception: after all blocks are written a file latency occurs and all blocks which were written are read, and the check character for each block is verified.

CPR

Compare

The applicable control block section of the buffer provides the control characters for this command. The control characters were placed there by a previous command. Once the control characters are obtained, the command is identical with the Seek/Compare command.

LNK

Link

The applicable control block section of the buffer provides the control characters for this command. The control characters for this command were placed there by a previous command. Once the control characters are obtained, the command is identical with the Seek/Link command.

## Status and Substatus

The major status and substatus conditions associated with the DS-20 Disc Storage unit subsystem are summarized in Figure G-8.

Status and Substatus	Major Status Code	Substatus Code
CHANNEL READY	0000	
DEVICE BUSY  Overridable Not Overridable	0001	000000 000001
ATTENTION	0010	
DATA ALERT  Transfer Timing Alert Transmission Parity Alert Invalid Control Character Internal Alert Check Character Alert Block Compare Alert Buffer Committed	0011	000001 0xxx10 0xx1x0 xx1xx0 x1xxx0 1xxxxx 11xxxx
END OF FILE	0100	
COMMAND REJECTED  Invalid Operation Code Invalid Device Code Illegal Buffer Address, or Buffer Committed Internal Error	0101	0000x1 00001x  00x1xx 001xxx
INTERMEDIATE	0110	
CHANNEL BUSY	1000	
CHANNEL ABSENT	1001	
CHANNEL ALERT	1010	

x = 0 or 1

Figure G-8. Status and Codes for the DS-20 Subsystem

## MAGNETIC TAPE UNITS

Provision of detailed information on the programming and operation of magnetic tape units is beyond the scope of this manual.

Additional information for the magnetic tape units may be found in the MT-20 magnetic tape subsystem reference manuals.

### Magnetic Tape Instructions

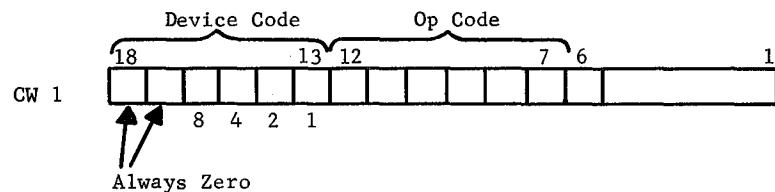
Brief functional descriptions of magnetic tape instructions are given below.

<u>Instruction</u>	<u>Mnemonic</u>	<u>Op Code Octal</u>
Read Tape	RT	05
Read Tape Decimal	RTD	04
Reread Tape	RR	07
Special Reread	SRR	06
Write Tape	WT	15
Write Tape Decimal	WTD	14
Write End-of-File Record	WEF	55
Forward Space One Record	FSR	44
Forward Space One File	FSF	45
Backspace One Record	BSR	46
Backspace One File	BSF	47
Erase	ERS	54
Set Density High	SDH	60
Set Density Low	SDL	61
Rewind	RWD	70
Rewind/Standby	RWS	72
Request Status	RQS	00
Reset Status	RSS	40

<u>Instructions</u>	<u>Mnemonic</u>
READ TAPE: Reads one tape record in the binary (standard) format.	RT
READ TAPE DECIMAL: Reads one tape record in the special decimal (BCD) format.	RTD
REREAD TAPE: Rereads binary records at normal, high, and low gain.	RR
SPECIAL REREAD TAPE: Rereads decimal records at normal, high, and low gain.	SRR
WRITE TAPE: Writes one tape record in the binary (standard) format.	WT
WRITE TAPE DECIMAL: Writes one tape record in the special decimal (BCD) format.	WTD
WRITE END-OF-FILE RECORD: Writes an end-of-file record on tape.	WEF
BACKSPACE ONE RECORD: Moves the tape backward without data transfer over one record and positions the tape to read or write the record passed over.	BSR
BACKSPACE ONE FILE: Moves tape backward without data transfer until end-of-file record is detected and positions the tape to again read or write the file just passed over.	BSF
FORWARD SPACE ONE RECORD: Moves tape forward over one record without data transfer and positions the tape to read or write over the next sequential record.	FSR
FORWARD SPACE ONE FILE: Moves tape forward without data transfer until an end-of-file record is detected and positions the tape to read or write the record following the detected end-of-file record.	FSF
ERASE: Erases approximately 8-1/2 inches of tape, beginning at the tape's position when the instruction is received.	ERS
SET DENSITY HIGH: Sets the addressed tape unit to the high-density mode.	SDH
SET DENSITY LOW: Sets the addressed tape unit to the low-density mode.	SDL
REWIND: Moves tape backward without data transfer until the beginning-of-tape marker is sensed.	RWD
REWIND/STANDBY: Moves tape backward without data transfer until the beginning-of-tape marker is sensed and puts the tape unit in a standby status.	RWS
REQUEST STATUS: Obtains and stores the status existing within the addressed tape unit or subsystem.	RQS
RESET STATUS: Resets all resettable status in the related subsystem controller then stores the status of the addressed tape unit at the specified second address after the reset operation.	RSS

## Command Words

Command words 1, 2 and 3 to the CPC will contain the information defined for the command words. However, the device code field of CW 1 needs further definition.



The device code is right justified and contains the binary number for the tape unit being addressed.

## Status and Substatus

The major status and substatus associated with the magnetic tape subsystem are summarized in Figure G-9.

Condition	Major Status Code	Substatus Code
CHANNEL READY  Tape Unit Write Inhibited Tape Reel on Load Point	0000	0000x1 00001x
DEVICE BUSY	0001	
ATTENTION  Tape Write Inhibited No Such Tape Unit Tape Unit Standby Tape Unit Check Blank Tape on Write *	0010	0xxxx1 0xxx1x 0x01xx 0x10xx 01xxxx
DATA ALERT  Transfer Timing Alert Blank Tape on Read Transmission Parity Alert Lateral Parity Alert Longitudinal Parity Alert End of Tape Bit Detected During Erase	0011	000001 xxxx10 xxx1x0 xx1xx0 x1xxx0 1xxxx0 xxxx11
END OF FILE  Single Data Character Data Alert	0100	xxxxxx 111111
COMMAND REJECTED  Invalid Operation Code Invalid Device Code Parity Alert on Device Operation Code Tape on Load Point Read After Write on Same Unit	0101	0xxxx1 0xxx1x  0xx1xx 0x1xxx  01xxx
LOAD OPERATION COMPLETE	0111	
CHANNEL BUSY	1000	
CHANNEL ABSENT	1001	
CHANNEL ALERT	1010	

x = 0 or 1

\* Blank Tape on Write also causes Tape Unit Standby; combined code will be 01x1xx, unless an instruction has reset the Blank Tape on Write.

Figure G-9. Status and Codes for Magnetic Tape Subsystem

## PRINTER

Additional information for the PR-20 and PR-21 printers may be found in their reference manuals.

### Printer Instructions

Printer instructions are shown in the chart below.

Instruction	Op Code (Octal)
Print in the Edit Mode--No Slew Information	30
Print in the Edit Mode--Slew Single Line	31
Print in the Edit Mode--Slew Double Line	32
Print in the Edit Mode--Slew to Top of Page	33
Print in the Nonedit Mode--Slew No Lines	10
Print in the Nonedit Mode--Slew Single Line	11
Print in the Nonedit Mode--Slew Double Line	12
Print in the Nonedit Mode--Slew to Top of Page	13
Slew Printer Single Line	61
Slew Printer Double Line	62
Slew Printer to Top of Page	63
Request Status	00
Reset Status	40

### Command Words

CW 1 to the CPC contains the operation code for the printer operation. CW 2 contains the starting address of the characters to be printed. CW 3 contains the count of the number of characters in the print line.

For each print line, the CPC must be given a new sequence of command words. When print and slew commands are combined, the printing is always executed first. If any slew other than single line, double line, or top of page is desired, a special slew character must be used, preceded by a print in edit mode command.

### End Print Line

The computer system must indicate that print-line data is complete by transmitting the End Data Transfer signal. The End Data Transfer signal is generated by the character counter in the CPC. When the character counter counts to zero, an End Data Transfer signal is sent to the printer. Depending upon the nature of the print line, the counter may have to be changed for each line.

## Print in Edit Mode

Additional information required by the printer must be contained in the print line data. The additional information is in the form of five special characters when printing in the edited mode.

### Special Characters

The special characters are Escape, Ignore, Space, Skip and Slew.

Character	Symbol	Binary	Octal
Escape	!	111111	77
Ignore	?	001111	17
Space	þ	010000	20
Skip	None	10xxxx	Varies
Slew	None	0xxxxx	Varies

ESCAPE CHARACTER. When the Escape character (111111) is received by the printer, special consideration is given to the character that follows. If only one Escape character is used, the Escape character and the character that follows are both deleted from the print line. When two consecutive Escape characters are used, both characters are deleted; and the character that follows the two Escape characters is printed. Thus, the Escape character can be used to delete or force printing of the special characters discussed below.

IGNORE CHARACTER. An Ignore character (001111) not preceded by an Escape character will be deleted from the print line; and the succeeding characters will be printed left-justified. An Ignore character preceded by one Escape character is treated as a Slew character (slew 15 lines by countdown). An Ignore character preceded by two consecutive Escape characters will be printed as a ?; both Escape characters will be deleted.

SPACE CHARACTER. The Space character (010000) used without Escape characters will cause a blank the width of a print character to be inserted in the print line. As many Space characters as desired can be used to achieve a true spacing effect, and they can be used in combination with the Skip character, the Space character becomes a Slew character (slew to top of page by VFU tape). When the Space character follows two consecutive Escape characters, the Space character is printed as a þ; both Escape characters are deleted.

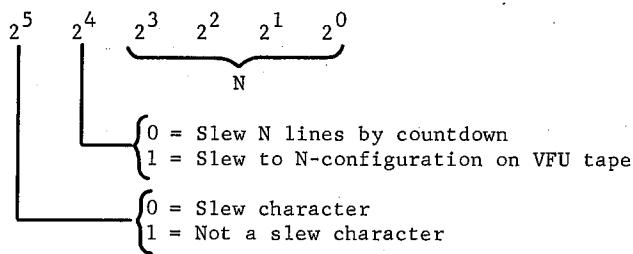
SKIP CHARACTER. When one Escape character is followed by a Skip character (10xxxx), the printer inserts in the print line the number of spaces indicated by the pattern of 1-bits in the four low-order positions of the Skip Character. The bits are read in multiples of 8 as follows:

Bit	Meaning
$2^5 - 1$	Indicate skip character
$2^4 - 0$	
$2^3 - -$	Insert 64 spaces
$2^2 - -$	Insert 32 spaces
$2^1 - -$	Insert 16 spaces
$2^0 - -$	Insert 8 spaces

A minimum of 8 spaces and a maximum of 120 spaces can be inserted with one Skip character. It can be combined with Space characters to achieve numbers which are not multiples of 8. These character spaces must be counted as print-line characters when figuring the maximum of 136 characters per line.

**SLEW CHARACTER.** Any character which has a zero in the high-order bit and is immediately preceded by an Escape character is treated as a Slew character. The four low-order bits contain the slew information, referred to as N. The next-to-highest-order bit tells the printer whether to slew by countdown or by VFU tape.

The meaning of each bit is illustrated below:



The 00xxxx configuration calls for a slew of N lines by countdown, using a 4-bit counter in the logic to keep track of the number of lines slewed. In this case, the four low-order bits are read as a binary value to represent the number of lines to be slewed. Paper can be slewed from 0 to 15 lines by countdown.

The 01xxxx configuration causes the printer to slew until the VFU detects a punched configuration on the VFU tape that matches the configuration in the N-portion of the slew character. Fifteen unique configurations can be used on the VFU tape per form page. The character 010000 is interpreted as a slew to top of page.

The printer interprets the receipt of a Slew character as an indication that the transfer of print-line data has been completed. Also, a Slew character transmitted with print-line data will always override any slew information in a slew command.

**SUMMARY OF SPECIAL CHARACTERS.** Eleven possible configurations of the special editing characters are summarized in Figure G-10. These are presented in binary for simplicity, since some characters cannot be converted to octal until X-values are filled in.

#### Print in Nonedit Mode

The four print commands for the nonedit mode are similar to those for printing in the edit mode, except that special editing characters are not recognized by the printer logic. Slewing information must be included as part of each command.

Character and Sequence			Deleted from Print Line	Printed	Other Action
No. 1	No. 2	No. 3			
111111	11xxxx		Both		None, unless No. 2 is an Escape character
111111	111111	xxxxxx	No. 1 & 2	No. 3	Prints any No. 3 character
111111	00xxxx		Both		Slew xxxx (N) lines by countdown
111111	01xxxx		Both		Slew to xxxx (N) on VFU tape
111111	10xxxx		Both		Insert xxxx character spaces in print line in multiples of 8
001111			No. 1		Prints remaining characters left-justified
111111	001111		Both		Ignore, treated as slew 15 lines by countdown
111111	111111	001111	No. 1 & 2	No. 3	Prints Ignore as a question mark
010000				No. 1	Prints as a blank character space
111111	010000		Both		Space character treated as slew to top of page by VFU tape
111111	111111	010000	No. 1 & 2	No. 3	Space character prints as a <b>b</b>

Figure G-10. Summary of Special Characters Used in Edit Mode

Special characters that are deleted from the print line in the edit mode are printed in the nonedit mode. Escape is printed as an exclamation point, Ignore as a question mark, and Space as a blank character space. Slew and Skip will print as one of the 64 characters of the standard character set, depending on the octal value assigned by the programmer. Because the Slew character is not recognized in this mode, it is possible to slew only one space, two spaces, or to top of page, by command. The Slew character cannot be substituted for an End Data Transfer signal to indicate the end of the print line.

#### Automatic Slewing

Automatic starting and stopping of slewing from punches in channels 5 and 6 of the VFU tape loops require special consideration by the programmer. The action of the printer depends on the method of giving slew information and the circumstances under which the punches are encountered.

When slewing by a program, if a punch is detected in channel 5, paper moves until the channel-6 punch stops the automatic slew. The printer continues counting lines while the automatic slew is in effect. If the countdown is satisfied before the channel-6 hole is reached, the slew continues to the automatic-stop punch in channel 6. If the countdown slew is not completed, then slewing continues until all lines called for by countdown are slewed.

If the command calls for a slew to N on the VFU tape, a channel-5 punch causes automatic slew to take over until the channel-6 punch is reached. If the N-configuration on the tape has not been reached after the automatic slew, the slew resumes until N is reached. If the N-configuration is reached before the automatic stop hole is reached, the automatic slew continues until the channel-6 punch is detected.

In both types of command slewing, the rule to remember is that paper slews to the farthest spot before the combination of programmed and automatic slewing is considered complete.

As mentioned in the discussion of Slew characters, automatic slewing is ignored when slewing is accomplished by a Slew character. However, if the Slew character causes the paper to stop on a channel-5 punch, then the automatic slew to the channel-6 punch occurs.

### **Controls and Indicators**

The MANUAL CLEAR button on the DATANET-30 control panel also resets the printer. The OPERATE/RESET button on the printer must be pressed after manual clear to return the printer to the operate mode.

### **Status and Substatus**

Major status and substatus associated with the PR-20 and PR-21 printers are shown in Figures G-11 and G-12.

Status and Substatus	Major Status Code	Substatus Code
CHANNEL READY	0000	
Attention	0010	
Out of Paper		00xxxx
Manual Halt		00xx1x
VFU Alert		00xlxx
Check		001xxx
DATA ALERT	0011	
Transfer Timing Alert		000011
Alert Before Printing Started		xxxx1x
Alert After Printing Started		xxx1xx
Paper Low		xxlxxx
Slew Alert		x1xxxx
Top of Page Echo		1xxxxx
COMMAND REJECTED	0101	
Invalid Operation Code		xx0001
Slew Alert		x1000x
Top of Page Echo		1x000x
CHANNEL BUSY	1000	
CHANNEL ABSENT	1001	

x = 0 or 1

Figure G-11. Status Codes for PR-20 Printer

Status and Substatus	Major Status Code	Substatus Code
CHANNEL READY	0000	000000 000001 000010 000011 000100 000101 000110 000111
ATTENTION	0010	00xxx1 00xx1x 00x1xx 001xxx
DATA ALERT	0011	xxx011 xxx01x xxx100 0x1xxx x1xxxx 1x0xxx
COMMAND REJECTED	0101	xx0001 x1000x 1x00xx
CHANNEL BUSY	1000	000000
CHANNEL ABSENT*	1001	000000

x = 0 or 1

\* Returned by I/O channel of GE-400 Series only

Figure G-12. Status and Codes for PR-21 Printer



## CARD READER

For further information refer to the CR-20 card reader reference manual.

### CR-20 Card Reader Instructions

Following are descriptions of CR-20 card reader instructions.

<u>INSTRUCTION</u>	<u>MNEMONIC</u>	<u>OP CODE (OCTAL)</u>
Read Card Binary	RCB	01
Read Card Decimal	RCD	02
Read Card Mixed	RCM	03
Request Status	RQS	00
Reset Status	RSS	40

<u>INSTRUCTION</u>	<u>DESCRIPTION</u>
Read Card Binary	Data from the top half of each column (rows 12-3) is transmitted first, supplemented by an odd parity bit. Data from the lower half of the column (rows 4-9) is transmitted second. One and one-half card columns fill each memory locations. Columns 1-79 fill the first 53 memory words. The 54th word contains the first character of data from column 80.
Read Card Decimal	The contents of each card column punched in the Hollerith format are converted to the equivalent 6-bit character and transmitted with an odd parity bit. In addition, a check is made to see that the punches in each column represent a valid character of the General Electric standard character set. Three card columns of data fill one memory word. Thus, columns 1-78 occupy 26 locations of memory. Word 27 contains data from columns 79 and 80.
Read Card Mixed	As each card is read, column 1 is examined to determine the type of card and, thus, how the data is to be handled. If the "7" and "9" of column 1 are punched, regardless of what other punches occur in the column, the card is read in the binary mode. If either or both of these two punches are missing, the card is read in the alphanumeric mode. In either case, the full 80 columns of card information are transmitted, including column 1. No Hollerith characters are represented by both the 7 and 9 punches, either with or without other punches. Therefore, the card reader interprets a punch configuration in column 1 with the special identifying 7 and 9 punches as a binary card.

### Timing Considerations

Reading is done on a single-card-per-instruction basis. A new instruction must be given for each card. To maintain reading at the rate of 900 cards per minute (15 cards per second), a new read instruction must be given within 1 millisecond after the card reader generates a Terminate signal for the previous card. If the card reader does not receive a new read instruction within this time, speed drops in proportion to the delay.

## Data Transfer

The sequence for reading characters from a card in the binary mode is shown in Figure G-13.

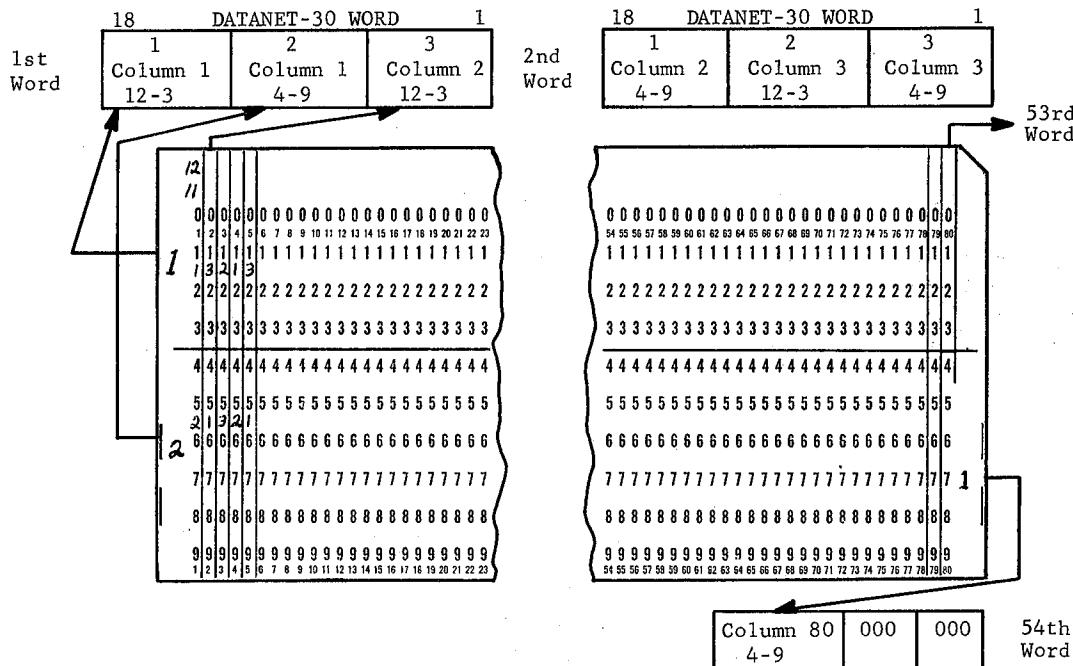


Figure G-13. Character Sequence for Read Card Binary Instruction

The sequence for reading characters from a card in the decimal mode is shown in Figure G-14.

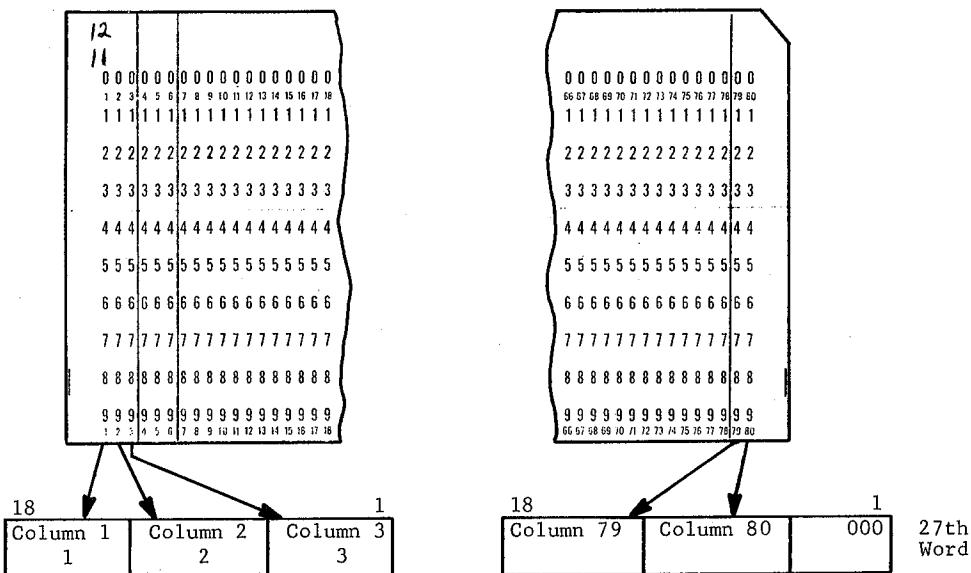


Figure G-14. Character Sequence for Read Card Decimal Instruction

## Program Load

The DEF 3 instruction to the CPC is sent to the card reader as a program load command. The card reader will read one card in binary and halt. A Not Busy major status is returned to the CPC.

## Status and Substatus

The major status and substatus associated with the CR-20 card reader are shown in Figure G-15.

Condition	Major Status Code	Substatus Code
CHANNEL READY	0 0 0 0	
ATTENTION	0 0 1 0	
Hopper/Stacker Alert		x x x x x 1
Manual Halt		x x x x 1 x
Last Batch		x x x 1 x x
Feed Alert		x x 1 x x x
Card Jam		x 1 x x x x
Read Alert		1 x x x x x
DATA ALERT	0 0 1 1	
Transfer Timing Alert		0 0 0 0 0 1
Validity Alert		0 0 0 0 1 0
COMMAND REJECTED	0 1 0 1	
LOAD OPERATION COMPLETE	0 1 1 1	
CHANNEL BUSY	1 0 0 0	
CHANNEL ABSENT	1 0 0 1	
CHANNEL ALERT	1 0 1 0	

x = 0 or 1

Figure G-15. Status and Codes for CR-20 Card Reader



## CP-10 CARD PUNCH

Additional information for the CP-10 (100 cards per minute) card punch may be found in the CP-10 card punch reference manual.

### CP-10 Card Punch Instructions

<u>INSTRUCTION</u>	<u>MNEMONIC</u>	<u>OP CODE (OCTAL)</u>
Punch Card Binary	PCB	11
Punch Card Decimal	PCD	12
Punch Card Edited Decimal	PCE	13
Request Status	RQS	00
Reset Status	RSS	40

<u>INSTRUCTION</u>	<u>DESCRIPTION</u>
Punch Card Binary	Two 6-bit binary characters are punched into each card column. The first character goes into card rows 12-3 (row 12 contains the most-significant bit). The second character is punched into card rows 4-9 (row 4 is the most-significant bit). The third and fourth characters are punched in column 2, etc.
Punch Card Decimal	Each 6-bit data character received is converted into the standard Hollerith punch code and punched into a single card column.
Punch Card Edited Decimal	Identical with the alphanumeric mode, but with the additional feature of allowing the punch to delete any Ignore characters (octal 17) which it may receive. Whenever an Ignore character is deleted, the next valid character received is punched, with no blank column intervening.

### Command Words

Command words 1, 2, and 3 to the CPC must contain the operation code, starting address, and character count, as applicable. To punch a card, the same command sequence and data must be transferred to the punch 12 times.

### Timing Considerations

As described above, a punch command must be received for punching each of the 12 rows of the card. The same set of punch data is transmitted to the punch for each row. The punch picks out the applicable bit from each data character, based on the row to be punched. The DATANET-30 must issue each punch command and data in time to punch each row. Approximately 600 milliseconds are required to punch each card.

There are 28.4 milliseconds between rows in which to give the punch command and complete data transfer. The punch circuitry requires 3.2 milliseconds to complete the data transfer. Therefore, the DATANET-30 must issue the command sequence to the punch for the next row within 25.6

milliseconds after each Terminate signal. The punch command sequence can be transferred to the CPC only after each Terminate signal from the punch. The punching time for each row is 14.4 milliseconds. Following this, there are 4.4 milliseconds for reading the corresponding row of the preceding card. Parity is generated and started as each row is punched. When the row is read, parity is checked. If parity does not check, the error is indicated after the card following the one being checked has been punched.

After all the cards have been punched, the last card is still in the punch. Two more cards must be fed through the punch either manually or by program in order to clear the punch mechanism of all punched cards. The card punch timing cycle is shown in Figure G-16.

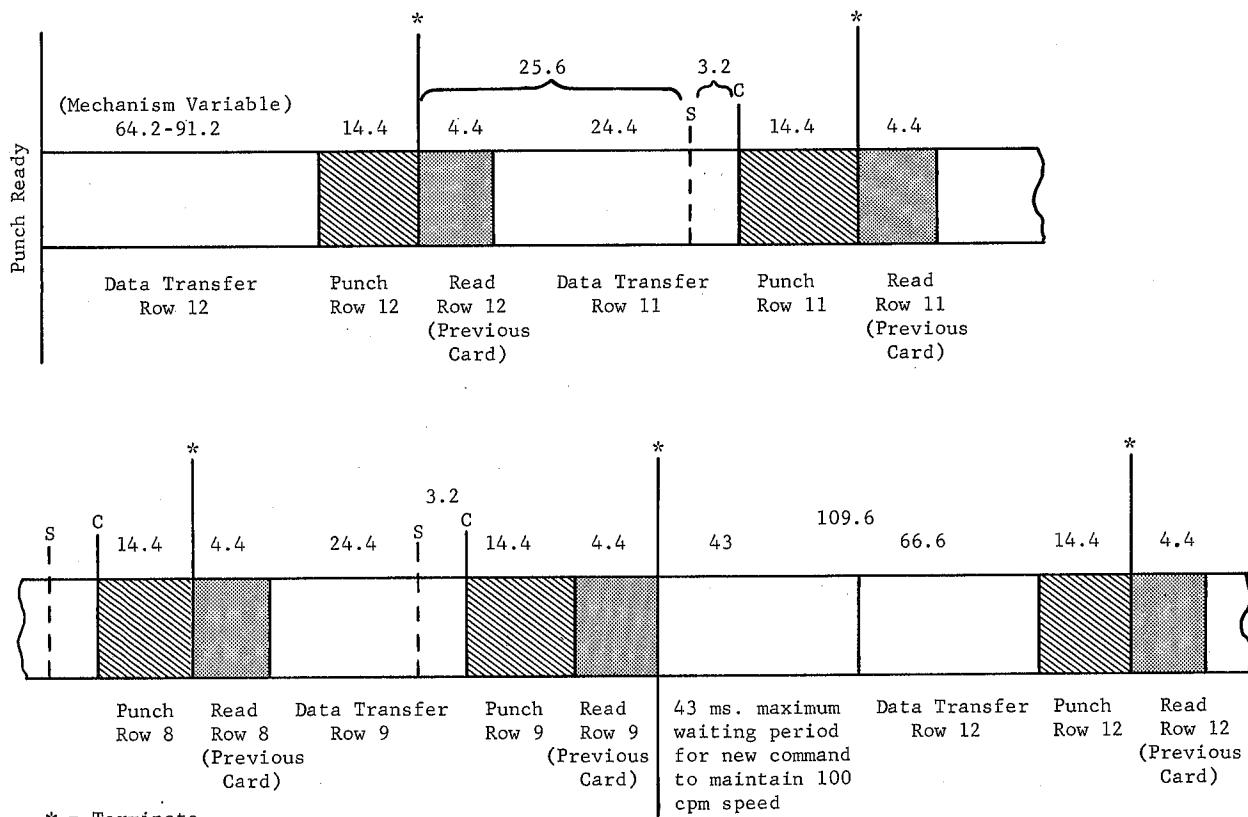


Figure G-16. Card Punch Timing Cycle (in Milliseconds)

After the last row of the card is punched and row 9 of the preceding card is read, a punch command may be received for row 12 of the next card within 43 milliseconds. This permits the 100 card-per-minute punching rate to be maintained. If the new command is received during this 43-millisecond interval, approximately 66 additional milliseconds are available for the transfer of punch data for the first row. If a punch command is not received during this period, there is a 7-card-per-minute reduction in the over-all card punching rate. There will be an additional 7-card-per-minute reduction in speed for every additional 43 milliseconds of delay in receiving this first command.

## Data Transfer

The sequence for punching characters into a card in the binary mode is shown in Figure G-17.

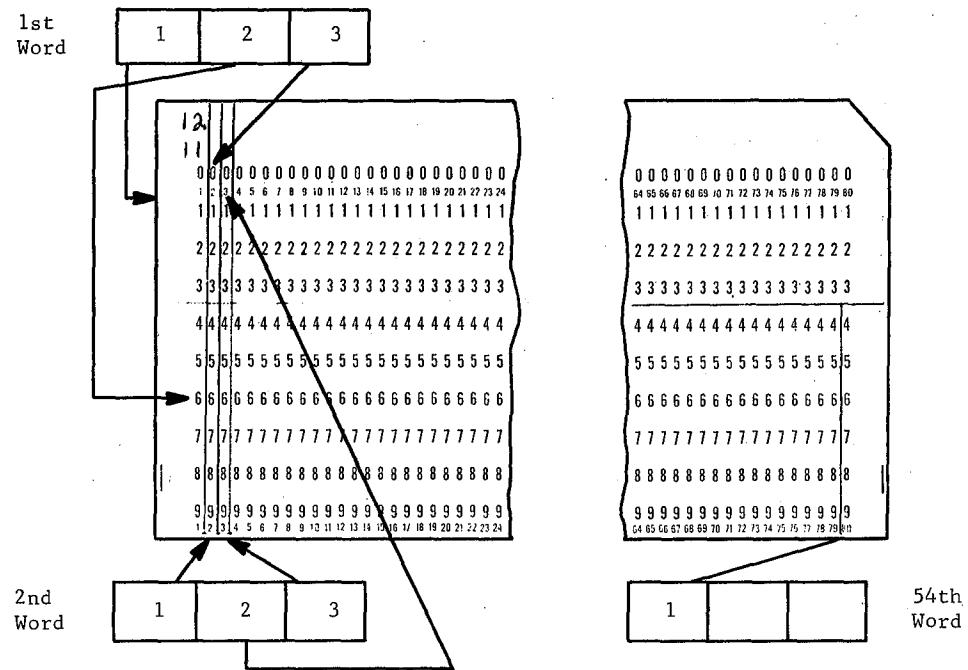


Figure G-17. Character Sequence for Punch Card Binary Instruction

The sequence for punching characters into a card in the decimal mode is shown in Figure G-18.

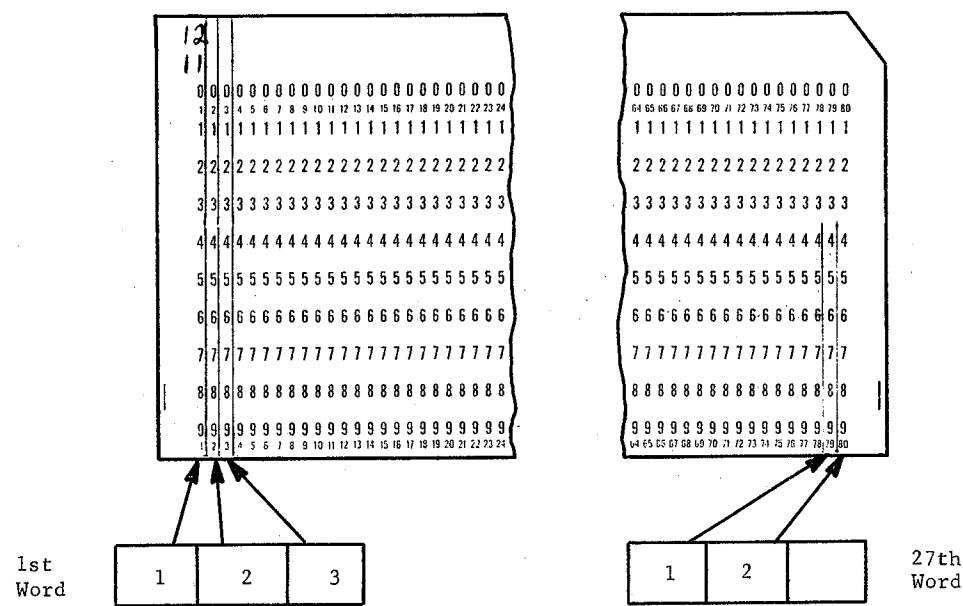


Figure G-18 Character Sequence for Punch Card Decimal Instruction

## Status and Substatus

Major status and substatus for the CP-10 card punch are listed in Figure G-19.

Status and Substatus	Major Status Code	Substatus Code
CARD PUNCH READY	0000	
ATTENTION	0010	
Hopper/Stacker Alert		0xxxx1
Manual Halt		0xxx1x
Chad Box Full		0xx1xx
Feed Alert		0x1xxx
Card Jam		01xxxx
DATA ALERT	0011	
Transfer Timing Alert		000xx1
Transmission Parity Alert		000x1x
Punch Alert		0001xx
COMMAND REJECTED	0101	
INTERMEDIATE	0110	
CHANNEL BUSY	1000	
CHANNEL ABSENT	1001	

x = 0 or 1

Figure G-19. Status and Codes for CP-10 Card Punch



## CP-20 CARD PUNCH

Additional information for the CP-20 (300 cards per minute) card punch may be found in the CP-20 card punch reference manual.

### CP-20 Card Punch Instructions

Instructions for the CP-20 card punch are described below.

<u>INSTRUCTION</u>	<u>MNEMONIC</u>	<u>OP CODE (OCTAL)</u>
Punch Card Binary	PCB	11
Punch Card Decimal	PCD	12
Punch Card in Edited Mode	PCE	13
Request Status	RQS	00
Reset Status	RSS	40

<u>INSTRUCTION</u>	<u>DESCRIPTION</u>
Punch Card Binary	Two 6-bit binary characters are punched into each card column. The first character goes into card rows 12-3 (row 12 contains the most-significant bit). The second character is punched into card rows 4-9 (row 4 contains the most-significant bit). The third and fourth characters are punched in column 2, etc.
Punch Card Decimal	Each 6-bit data character is converted into the standard Hollerith punch code and punched into a single card column.
Punch Card in Edited Mode	Identical with the alphanumeric mode, but with an additional feature--the punch deletes any Ignore characters (octal 17) from the data. Whenever an Ignore character is deleted, the next valid character is punched, with no blank column intervening.

### Command Words

Command Words 1, 2 and 3 to the CPC must contain the operation code, starting address, and character count as applicable. To punch a card, the command sequence and data transfer are transferred to the punch once.

### Functional Description

The subsystem becomes busy when a punch command is received and accepted. When the punch becomes busy, it requests data for the card. Data is transmitted, character by character, from the processing system to the card image buffer of the punch controller. The parity of each character is checked as it is received. If the command is Punch Card Decimal the character is converted by the controller to Hollerith code before the data enters the buffer.

As row 12 of the first card passes over the punch dies, the control electronics extract the row-12 information from the card image buffer and activate the punching mechanism. The punch remains

in the busy state, punching data row by row until the last row of the card is completed. It then transmits a Terminate signal, indicating to the processing system that the punch has completed the operation.

The first card is checked while the second card is being punched. As row 12 of the second card enters the punch area, row 12 of the first card starts through the read head. There, row parity is compared against the parity established for the row when the data was first received by the card image buffer. If the two parity checks do not agree, a punch alert signals the discrepancy. After the second card is punched, the results of the read check of the first card are transmitted to the processing system. When the last row has been read and checked, the card continues to the output stacker.

### Timing Considerations

The punch controller requires that a punch command and all data for the card image buffer be received before a card is fed and punched. No command or data will be accepted for the next card until a card has been completely punched.

A complete cycle of transferring data to the punch and punching a card requires 200 milliseconds. This is illustrated in Figure G-20. The time is divided into two phases. Phase 1 is transfer of the command and data to the punch. Phase 2 is the punch operation. After the punch controller has received the punch command and all the data, a signal to feed a card is given. The card is punched and a Not Busy status is indicated in the CPC. When the CPC goes not busy, another punch command sequence must be given for the next card.

The transfer of the command and all data to the punch requires approximately 20.5 milliseconds. Since 29 milliseconds are allowed to complete data transfer, a new command sequence must be issued to the CPC (punch) within approximately 8.5 milliseconds after the CPC goes not busy.

If the CPC command sequence is not issued in time to complete the data transfer within the 29 milliseconds, the punch mechanism continues to cycle (at 200 ms/cycle) and will punch a card the next feed cycle time.

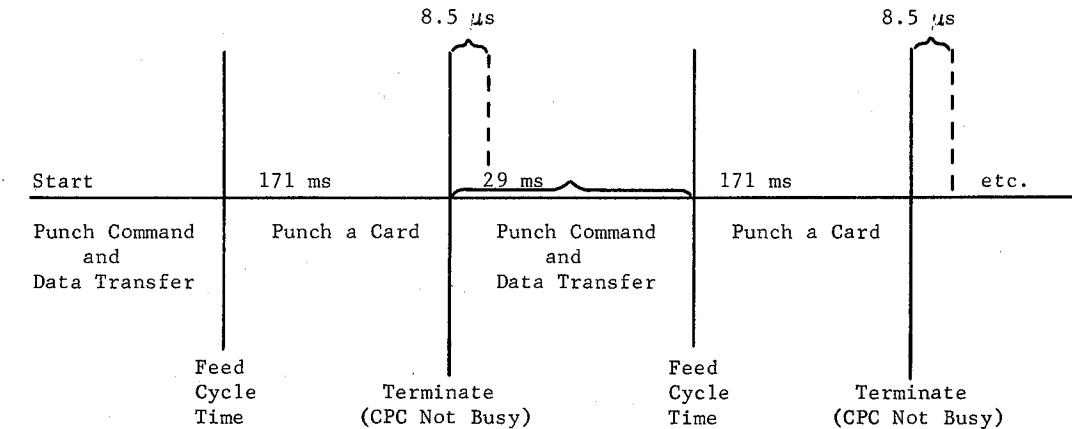


Figure G-20. CP-20 Punch Cycle

## Status and Substatus

Major status and substatus for the CP-20 Card Punch are listed in Figure G-21.

Status and Substatus	Major Status Code	Substatus Code
CARD PUNCH READY	0000	
ATTENTION	0010	
Hopper/Stacker Alert		0xxxx1
Manual Halt		0xxx1x
Chad Box Full		0xx1xx
Feed Alert		0x1xxx
Card Jam		01xxxx
DATA ALERT	0011	
Transfer Timing Alert		000xxx
Transmission Parity Alert		000x1x
Punch Alert		0001xx
COMMAND REJECTED	0101	
INTERMEDIATE	0110	
CHANNEL BUSY	1000	
CHANNEL ABSENT	1001	

x = 0 or 1

Figure G-21. Status and Codes for CP-20 Card Punch



## APPENDIX H

### DATANET-30 PROGRAMMING REFERENCE MANUAL

#### GENERAL DESCRIPTION

The DATANET-30 Processor Interrupt Unit (PIU-930) allows transfer of data from the memory of one DATANET-30 to the memory of another. One processor interrupt unit (PIU) for each DATANET-30 is required. Each PIU occupies one option module space and is assigned an address on the buffer selector. The address is specified by the address plug for the PIU.

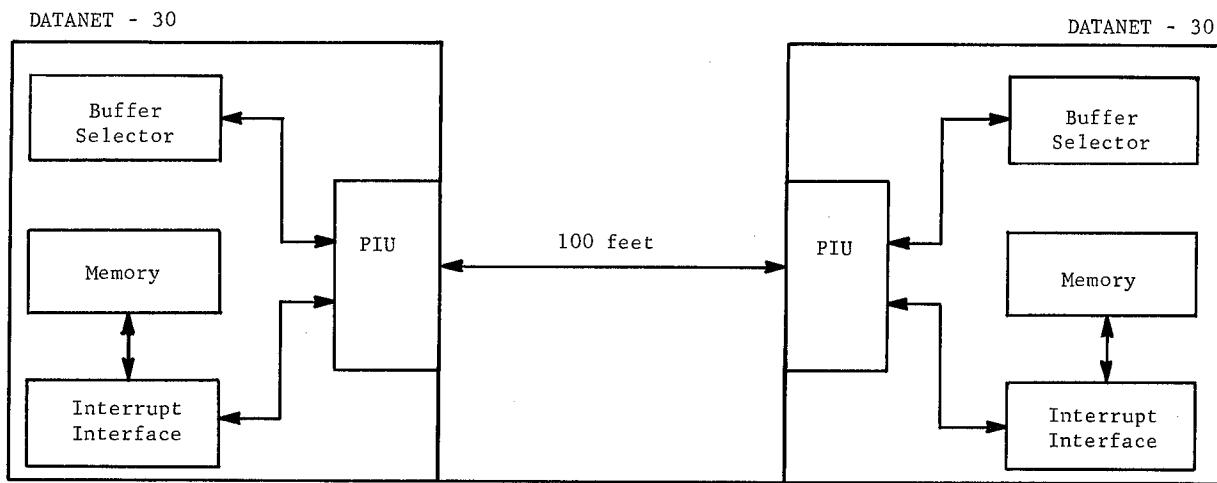


Figure 1. PIU Connections to Two DATANET-30's

#### Data Transfer

The PIU connects to the memory interrupt interface and is normally assigned interrupt cycle two. Only one interrupt cycle may be assigned to the PIU. This allows a transfer rate of 28,800 DATANET-30 words per second.

Data is transferred in both directions, but only in one direction at a time. Control of the direction of data transfer is established by either DATANET-30 on a receive only basis.

The PIU does not allow a program to initiate transmission. The DATANET-30 which is to receive data must initiate the data transfer. This method of operation protects the memory in one DATANET-30 from being destroyed by mistake by the other DATANET-30.

With only the ability to receive, each DATANET-30 must know how much, where and when to initiate data transfer. Various techniques may be used to accomplish this coordination. Two techniques are the master-slave relationship and the mailbox.

Master-Slave Relation. The master-slave relationship is a programming consideration only and can be changed by programming.

The master-slave arrangement here only decides which DATANET-30 may initiate data transfer. The master may initiate transfer if the PIU is not busy and if the master has not granted the slave permission to use the PIU. The slave may initiate transfer only after the master has granted permission to do so, and if the PIU is not busy.

Two lines (signals) allow communication between the programs for master-slave purposes. These are line A (NES 1) and line B (NES 4). The lines are set by DEF 6 and DEF 4 instructions, respectively.

The programming convention for using these lines is as follows:

One DATANET-30 must be assigned as master and the other as slave. The slave requests control to initiate operations by setting line B. The master then grants permission by setting line A. This and only this resets line B. After the slave has received a message, the End of Message signal resets line A.

When the master wants control, line A indicates which way control is currently established. Line A set means the slave has control or is receiving data. Line A reset means that the slave does not have control and the master may assume control to initiate data transfer. The NES 2 and NES 3 lines also indicate busy conditions. Therefore, the master must periodically check line B to grant permission to the slave.

Since setting and interrogating line B is a programming consideration, the master can load the PIU even when line B is set, without setting line A. Lines A and B are used only for coordinating the programs and do not inhibit any hardware or program action. The slave can also load the PIU's without permission; however, this violates the rules for using control lines A and B.

To summarize, the conditions are illustrated in the chart below.

Line A	Line B	
0	0	Master may load PIU. Slave is not requesting.
0	1	Master may load PIU. Slave is requesting.
1	0	Slave may load PIU. Master may not.
1	1	Not Possible.

The programs can exchange master control of the PIU's as desired. Exchange can be based on time of day, volume of traffic, busy hour, or other suitable factors. The master must agree to relinquish control as master before the slave assumes the master status.

The use of lines A and B in this manner is not mandatory. Any one of a number of methods could be used to prevent interference between programs. For example, each program could maintain a memory location as a timer. Then, by program control, each DATANET-30 could have unrestricted use of the PIU at certain times. One DATANET-30 could be considered as a standard time and the other could reset the timer to match. This method would prevent one DATANET-30 from waiting on the other for undetermined periods of time. Lines A and B could then be used for other communication purposes.

Mailbox. The mailbox is one or more words in memory used to store information for the other DATANET-30.

The contents of the mailbox word may be used to indicate that data is ready to be transferred, data has been transferred, the starting memory address, the number of words, which DATANET-30 is master, or other pertinent information. Each DATANET-30 would periodically transfer the mailbox word for examination.

Since the transmitting DATANET-30 has no control over the timing of the data transfer, it has no direct way of knowing that data transfer has taken place. It can tell that data has been transferred from memory only by interrogating the mailbox of the other unit. Therefore, the program design must include a method of notifying the transmitting program of the status of data transferred. Updating the contents of the mailbox permits the orderly transfer of data.

### Block Diagram

Figure 2 shows the basic configuration of the PIU. It contains one data register (buffer), an address counter, a word counter, control lines, and data transfer control logic.

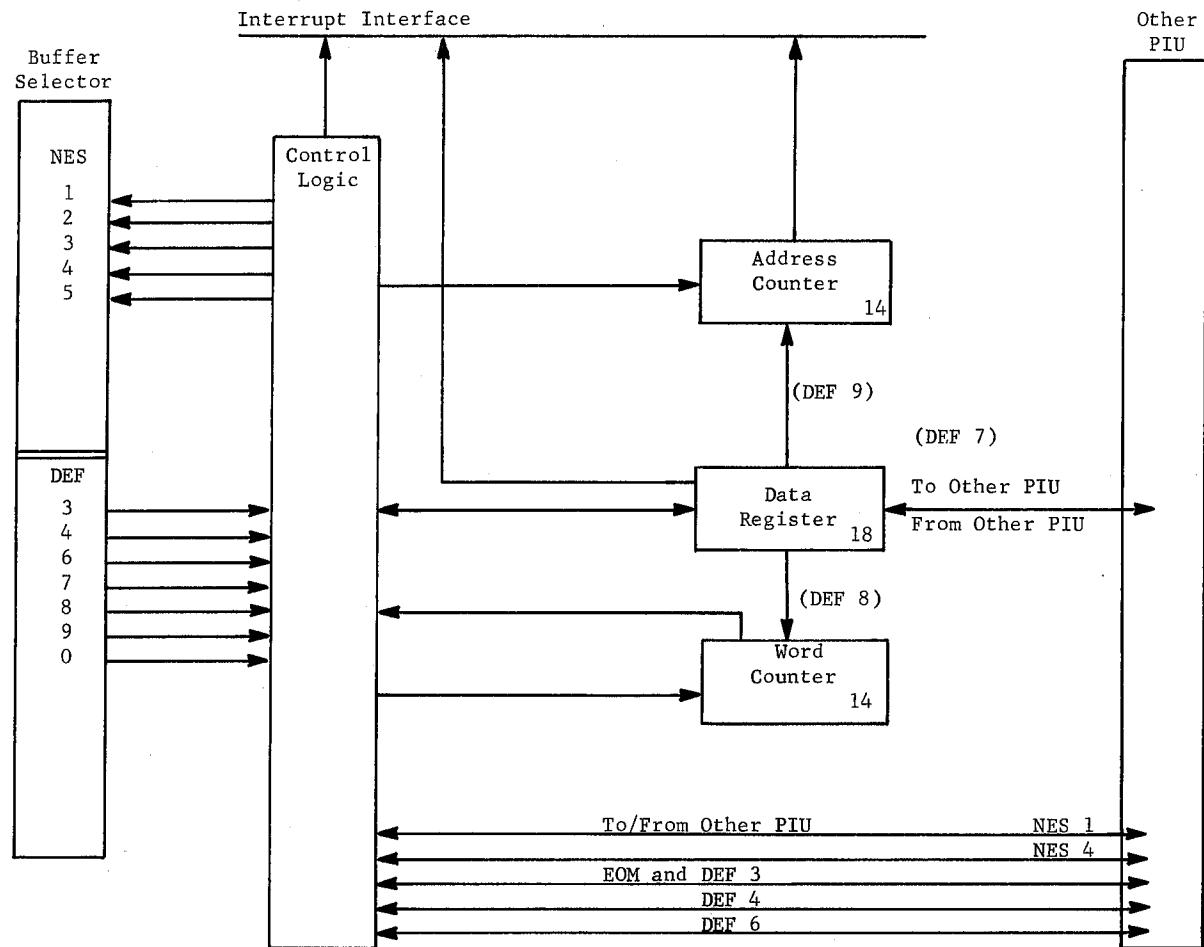


Figure 2. PIU Block Diagram

The PIU has three interfaces: (1) the DATANET-30 buffer selector, (2) the memory interrupt interface, and (3) the other PIU.

The buffer selector interface presents status (NES lines) to the program and accepts control signals (DEF lines) and load instructions (LDT) from the program. The PIU is also addressed via the buffer selector.

The memory interrupt interface is used for data transfer under memory interrupt control between the PIU and the memory locations specified by the address counter. Interrupt cycle 2 is arbitrarily assigned but can be changed by a wiring change.

The interface to the other PIU provides the necessary data and status lines to coordinate data transfer from one PIU to the other.

Word Counter. The word counter is used only by the receiving PIU to terminate data transfer when a specified number of words has been received. The counter is counted down one each time a word is transferred into memory. When the counter reaches zero, an "End of Message" signal is generated. The word counter is a 14-position straight binary counter and must be loaded with the binary equivalent of the number of words to be transferred.

End of Message. The End of Message (EOM) signal causes the following actions:

1. The word counter is reset to all 1's and the address counters to all 0's.
2. Both PIU's are reset to the starting configuration.
3. Line A is reset.
4. Data transfer is halted.

The End of Message signal is generated in either PIU when the word counter counts to zero, when a DEF 3 instruction is given, or when the manual reset button is pressed.

Address Counter. The address counter in the receiving PIU addresses memory to store the data received into the proper memory location. The address counter in the sending PIU is used to access the proper memory location for data to transfer. Both address counters must be loaded by the receiving DATANET-30. Each address counter is counted up one each time a word is transferred. The starting memory address is loaded into each address counter.

Data Register. The data register holds one DATANET-30 word transferred to or from memory, or to or from the other PIU, depending upon the direction of transfer.

## INSTRUCTION REPERTOIRE

The instructions are classified under buffer selector instructions for the DATANET-30. They are the Drive External Function (DEF), the External Status Lines (NES) and the Load Transmit Drivers (LDT) instructions. The rules for using the DEF and NES instructions are the same as those for using other DATANET-30 buffer selector instructions. The LDT instruction is special.

The Drive External Function instructions are:

- DEF 1-2 - Not used.
- DEF 3 - End of Message (EOM). Terminates operation by providing an EOM signal.
- DEF 4 - Set control line B. Sets control line B in each DATANET-30. Line B can be reset only when line A is set. Line B cannot be set when line A is set.
- DEF 5 - Not used.
- DEF 6 - Set control line A. Sets control line A in each DATANET-30. End of Message resets line A.
- DEF 7 - Load address counter. Transfers the contents of the data register to the other PIU. The starting memory address at which the transmitting DATANET-30 will begin data transfer is sent to the address counter of the other PIU.
- DEF 8 - Load word counter. A count previously loaded into the PIU is transferred to the word counter of the receiving PIU.
- DEF 9 - Load address counter. Transfers the contents of the data register to the address counter. The starting memory address of the receiving DATANET-30 must be in the data register for transfer to the address counter of the receiving PIU.
- DEF 0 - Load data register in the PIU. This signal is automatically generated by an LDT instruction. This instruction cannot be used for any other purpose.

The Load Transmit Drivers instruction.

- LDT - Transfer a word to the PIU. Transfers a control word from a specified memory location to the PIU data register. A DEF 0 instruction is automatically executed as part of an LDT instruction. An LDT instruction must precede a DEF 7, DEF 8, or DEF 9 instruction. LDT may not be used for any other purpose in relation to the PIU.

The External Status Lines instructions.

- NES 1 - Line A set. A 1 indicates that the line has been set. Either DATANET-30 can set line A. By convention, for the master-slave relationship, only the master should set line A. Line A set indicates permission granted for the slave to receive. If NES 1 is set, NES 4 cannot be set. NES 1 is reset by End of Message.
- NES 2 - Busy-Transmit mode. A 1 indicates that the PIU is sending data and the transfer is not complete. The PIU is in Transmit mode after the DEF 7 instruction in the receiving DATANET-30.
- NES 3 - Busy-Receive mode. A 1 indicates that the PIU is receiving data and the transfer is not complete. The PIU is busy after the DEF 7 of the loading sequence.

- NES 4 - Line B is set. Either DATANET-30 can set line B. A 1 indicates that the line has been set and that line A is not set. By convention, if the master-slave relationship is used, only the slave should set line B. Line B set indicates a request for permission to initiate data transfer. Setting line A will reset line B.
- NES 5 - No transfer. A 1 indicates that data transfer has stopped without an End of Message. This line has no significance in the receiving PIU until 14 DATANET-30 word times after the DEF 7 of the loading sequence.

### No Transfer

Each PIU will detect no transfer and set NES 5 independent of the other. Each DATANET-30 program must interrogate for the NES 5. In the receiving DATANET-30, NES 5 means that the other DATANET-30 is not transmitting. This signifies that the transmitting DATANET-30 either has a hardware load or power failure or has attempted to load the PIU. In the transmitting DATANET-30 NES 5 means that the other DATANET-30 is not receiving. This signifies there is a hardware load or power failure in the receiving DATANET-30.

The halt in data transfer caused by hardware load is temporary. Data transfer continues after completion of hardware load.

A halt in data transfer caused by an attempt to load a PIU when in the transmit mode is an error. Recovery is either by the manual reset button on the control panel or a DEF 3 instruction. A DEF 3 by either DATANET-30 will reset both PIU's. In this case, the data transfer should be reinitiated.

### CONTROLS AND INDICATORS

All controls and indicators are by program only, except for manual reset on the control panel.

### LOADING THE PIU

The PIU is loaded by the DATANET-30 that will receive data. In order to initiate operations, the starting address must be placed in each PIU and the word count in the receiving PIU. This is accomplished by using the LDT and DEF 9, DEF 8, and DEF 7 instructions. The DATANET-30 executing the following series of instructions will receive and the other will send.

LDT	RECAD	Load receiving starting address
DEF	9	Transfer RECAD to receiving address counter
LDT	WCNT	Load word count
DEF	8	Transfer count to receiving PIU word counter
LDT	SENDAD	Load starting address of the sending PIU
DEF	7	Transfer SENDAD to other PIU and set busy
RECAD	OCT 0	Receive starting address
WCNT	OCT 0	Word count
SENDAD	OCT 0	Sender starting address

## PROGRAMMING CONSIDERATIONS

1. The PIU must not be loaded when busy.
2. The maximum word count that can be put in the PIU word counter is 16,383 (all 1's).
3. Depending upon the manner of use, the unit designated as slave must not try to use the PIU if NES 1 is not set.
4. Each transfer of a word takes one memory cycle from each DATANET-30.
5. When one PIU is busy, the other is also busy.
6. The memory addresses and word count numbers must be right justified (low order position of the register).
7. The word counter must never be loaded with zero.
8. Either DATANET-30 can set line A without line B being set.
9. It is a hardware restriction that line B cannot be set when line A is set.



## CONTENTS

	Page
DUAL ACCESS DSU CONTROLLER	
General Description .....	1
DATANET-30/GE-225 System Configurations .....	1
DATANET-30/DATANET-30 System Configurations .....	2
Display Panel .....	4
Master Designation Switch .....	5
The Special (SPC) Switch .....	5
DSU Command Words .....	6
Command Word Format .....	6
Computer Lockout .....	6
Program Control .....	7
Automatic Priority Interrupt (API) .....	8
Test and Branch Conditions .....	8
Additional Test and Branch Conditions for the GE-225 .....	8
Additional DATANET-30 Test and Branch Conditions .....	9
Alternate Access .....	9
Conditions for Access .....	10
Sample Flow Charts .....	11

## ILLUSTRATIONS

### Figure

1. Dual Access Controller Block Diagram	4
2. Controller Selector Command Word Relation to Memory Locations	6
3. Sample DSU Subroutine for GE-225	12
4. Part of Executive Program for DATANET-30	13



## APPENDIX I DUAL ACCESS DSU CONTROLLER

### GENERAL DESCRIPTION

The dual access controller (DAC) operates with two central processors so they can share the same disc storage unit (DSU) but not simultaneously. Both central processors are connected to the dual access controller at all times. One is connected in the usual manner and the second is connected to a second controller selector plug mounted in the dual access controller. The central processors with which the dual access DSU is used are:

GE-200 Series  
GE-200 Series  
DATANET-30

GE-200 Series  
DATANET\*-30  
DATANET-30

The dual access controller has all the features of a single access controller plus additional features because of the dual access capability.

It is assumed that one is familiar with the single DSU controller and the controller selector unit (CSU-931) as covered in Appendix L of the DATANET-30 Programming Reference Manual, CPB-1019A. Since the dual access DSU controller is an extension of the single access controller, the additional DATANET-30 information needed is covered in this appendix. Detailed information relative to a GE-200 Series computer is not included.

### DATANET-30/GE-225 SYSTEM CONSIDERATIONS

The DATANET-30 and the GE-225 share a DSU through a dual access controller. The DATANET-30 is operating in real-time and DSU access must be given to the DATANET-30 when access is requested or data may be lost. This is very important and all system design characteristics must consider this fact. The GE-225 will be performing various batch processing jobs during the DATANET-30 operating time and some of these programs will require access to the DSU. These programs requiring access to the DSU should not be run during the busy hours of the DATANET-30. If they are, the GE-225 programs will run slowly because the DATANET-30 is using most of the DSU real-time during the busy hours.

The DATANET-30 system should be designed to operate on a cyclic basis. The cycle time selected should be large enough to allow DSU real-time to do the message switching and still have some DSU real-time available for the GE-225.

---

\*Reg. Trademark of the General Electric Company.

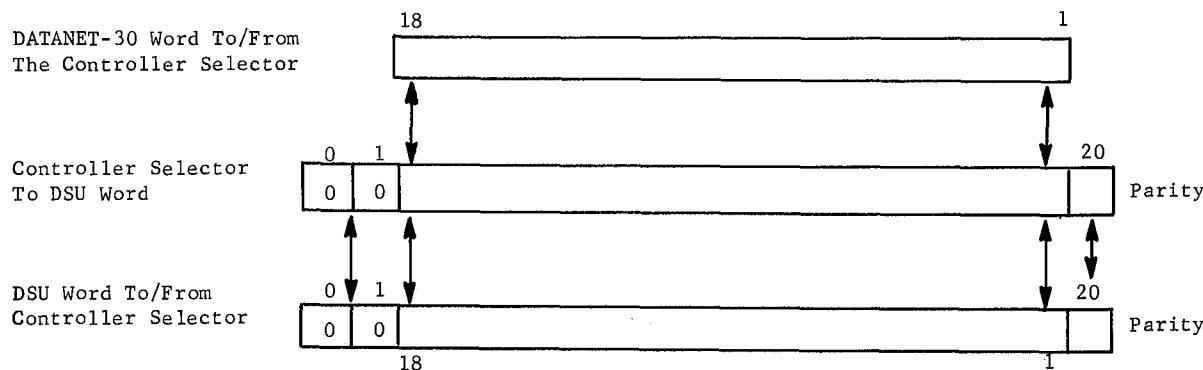
The DSU programs in the GE-225 must not lock out the DATANET-30 for more than 1-2 seconds at one time. This means that a DSU seek, read or write and all error checks done by the GE-225 have to be completed in 1-2 seconds, and control returned to the DATANET-30. The GE-225 can and must lock out the DATANET-30 to do error checking, but the lockout time must only be 1-2 seconds. Access could be immediately returned to the GE-225 if the DATANET-30 does not require access to the file.

As an example, assume that the DATANET-30 is on a cycle time of 18 seconds; that is, every 18 seconds the DATANET-30 requires access to the file. Also assume that the DATANET-30 system is designed to use from 3 to 12 seconds of this cycle time, depending on traffic volume, to complete real-time DSU processing. This means that the GE-225 could have access to the DSU for 6 to 15 seconds out of each cycle time depending on the DATANET-30 activity. If the GE-225 has 15 seconds of cycle time, it could perform many DSU operations before the DATANET-30 takes control. However, after each DSU operation, the GE-225 must release control in case the DATANET-30 requires access. The DATANET-30 locks out the GE-225 until the DATANET-30 real-time DSU processing is complete.

The DSU must be laid out with preference to the message switching to keep the DSU access times to a minimum. Timing consideration must be given to the fact that the GE-225 may disrupt any pattern of DSU access. The GE-225 may move the actuator arms and cause an increase in the DSU access time.

If the DATANET-30 is to accumulate data for the GE-225 on the DSU, control information must be stored on the DSU for both the DATANET-30 and the GE-225. Consideration must be given to the fact that the DATANET-30 has an 18-bit word and the GE-225 has a 20-bit word.

Each word recorded on a disc by the DATANET-30 consists of 18 information bits plus 2 zero bits, plus an odd parity bit which is generated by the DSU controller as shown below.



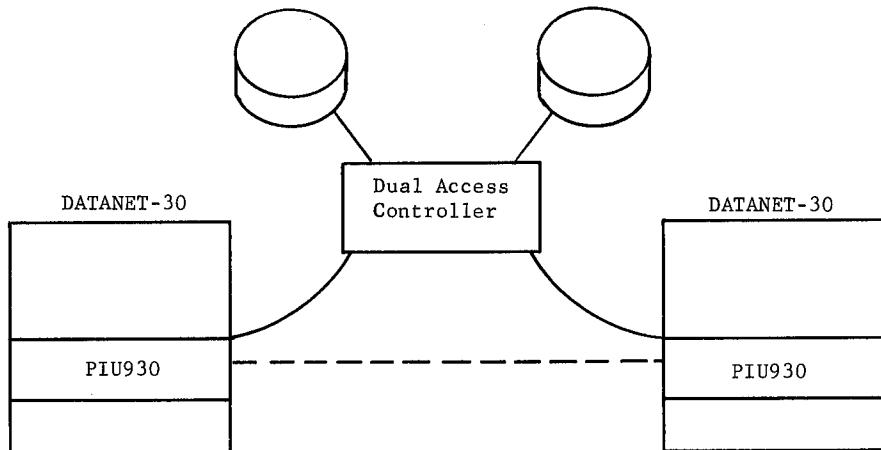
If a computer interface unit (CIU) is installed between the DATANET-30 and the GE-225, control information can be exchanged more efficiently.

## DATANET-30/DATANET-30 SYSTEM CONSIDERATIONS

When two DATANET-30 computers are sharing a DSU, the system considerations become quite different from the GE-225/DATANET-30 system.

Each DATANET-30 can be operating in real-time. Therefore the following capabilities and program controls of the DAC must be carefully considered.

1. Master-slave relation.
2. Use of lockout bit.
3. Timing for each DATANET-30 to use the DSU on an alternate access or timed access basis.
4. Program control of master.
5. Method of exchanging control information:
  - a. Is a PIU930 in the system?
  - b. Control information on the DSU.



Normally one DATANET-30 will be master and the other one slave, depending on how the programs require access to the DSU. Also, it might be more important for one to write than the other. Each DATANET-30 program should be designed to operate on a cyclic basis such that the cycle time is large enough to allow sufficient real-time for each program to read or write as necessary.

If the lockout bit concept is used, the system must permit locking out one DATANET-30. If the lockout concept is not used, alternate access on a timed basis is the next choice.

If a processor interrupt unit (PIU930) is included in the system, and one DATANET-30 is considered master, the other DATANET-30 would access the DSU only upon direction of the master.

Program control of master depends strictly upon the purpose and design of the system. The many variables involved are beyond the scope of this appendix.

The method for exchanging control information varies according to equipment used. Without a PIU930, control information must be stored on the DSU. With a PIU930, control information can be exchanged directly and more efficiently.

## DATANET-30

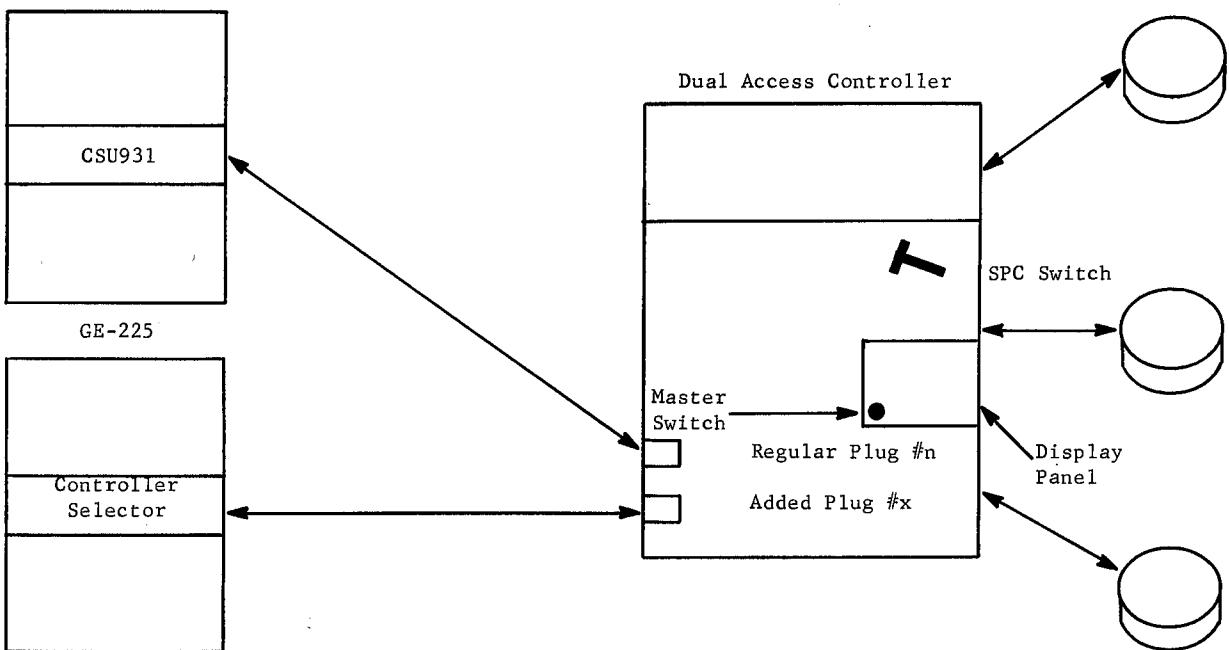


Figure 1. Dual Access Controller Block Diagram

## DISPLAY PANEL

Associated with the DAC electronics is a display panel inside the controller. This display panel has eight (8) lights to designate the status of the controller. These lights are labeled: 12, 13, 14, SW, RBS, ABS, ADD, and REG. If the lights are on:

1. "12" indicates that the processor in control has bit 12 set, that is, the master bit.
2. "13" indicates that the processor in control has bit 13 set, that is, the lockout bit.
3. "14" indicates that the processor in control has bit 14 set, that is, the API bit.
4. "SW" indicates that the last processor to have control was the processor on the Regular plug.
5. "RBS" indicates that the processor on the Regular plug has issued a branch or select command.
6. "ABS" indicates that the processor on the Added plug has issued a branch or select command.
7. "ADD" indicates that the processor on the Added plug has control.
8. "REG" indicates that the processor on the Regular plug has control.

There is also a RESET button on the panel to reset these conditions. Several of the above lights will be on at any one time when the DAC is being accessed. If there is no activity, then only the SW light will be on.

## MASTER DESIGNATION SWITCH

When both computers request access to the DSU, one is assumed to be the master and receive first access. To accomplish this priority, the three-way switch functions in the following manner:

<u>Position</u>	<u>Action</u>
Regular	The computer attached to regular Plug #n will be the master.
Added	The computer attached to Added plug #x will be the master.
Program Control	Master is determined by program control.

A master can and must be designated between the DATANET-30 and the GE-225. The reason for this is to establish a priority for granting accesses. The DATANET-30 must be the master.

The three-way switch in the DAC used to designate the master, must be in the regular position. The DATANET-30 must be plugged to the Regular plug #n. This will designate the DATANET-30 as the master.

The reasons for this are:

1. When only one processor has power on, it must be plugged to the Regular plug to access the DSU. The GE-225 could have power off during DATANET-30 operating time.
2. If the three-way switch was in the program control position and the RESET button on the DAC was pushed to initialize the controller, the processor on the Regular plug becomes the master.
3. If the SPC switch is in the special position, the processor on the Regular plug can take control at any time.

## THE SPECIAL (SPC) SWITCH

The DAC has a toggle switch that will place the DAC in either the "normal mode" or "special mode". The special mode allows the processor on the Regular plug to take control at any time, even when the dual access controller is busy.

The normal mode causes an echo alarm if the dual access controller is selected when busy.

The special mode of operation is necessary because it is possible to cause the DSU to become inaccessible. If a processor gives a command to the DAC and then the processor hangs up before the DAC completes the DSU command sequence, the DAC remains in a busy condition. The other processor cannot gain access to the DSU until the busy condition is manually cleared. The DATANET-30 programs are designed to operate continually and therefore do not normally create such a condition. However, the GE-225 programs may not be designed this way since the GE-225 is doing batch processing or even debug runs. The DATANET-30 must be able to take control of the DAC if the GE-225 hangs up and creates a busy condition on the DAC.

## DSU COMMAND WORDS

The command words for the dual access controller are the same as for the single access controller plus added functions related to the dual access capability. The added functions are:

1. Lockout
2. Master Designation
3. Automatic Priority Interrupt
4. Test and Branch

## COMMAND WORD FORMAT

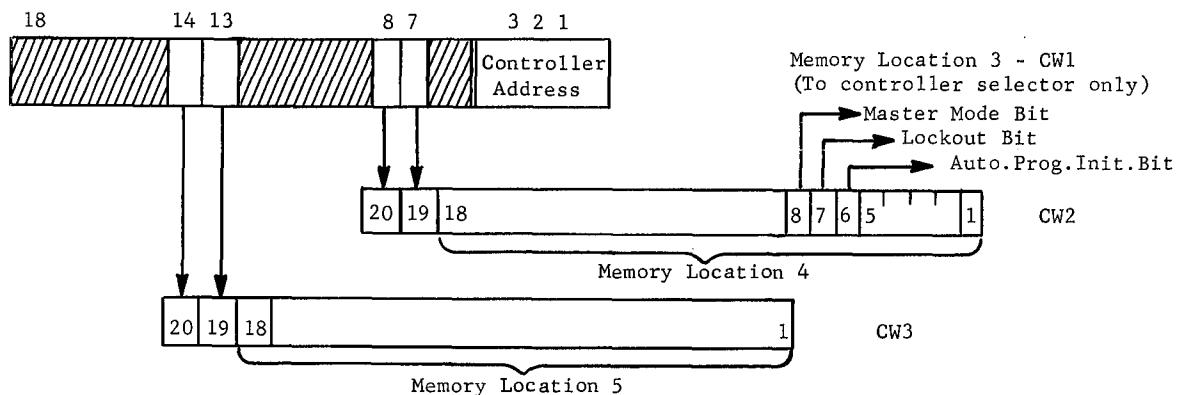


Figure 2. Controller Selector Command Word Relation to Memory Locations

## COMPUTER LOCKOUT

One computer may perform a lockout that will deny the other computer access to the DSU. That is accomplished by use of bit 7 of CW2 of the DATANET-30 command words for the controller selector unit. This bit functions in the following manner:

### Bit 7 - CW2

Off

### Function

At the completion of this command, either computer can receive a controller ready indication by use of a ready/not-ready, test and branch instruction.

The computer that issued the command will receive a ready condition if the other computer did not perform a test and branch instruction.

On

At the completion of this command, only the computer that issued the command can receive a controller ready indication by use of a test and branch instruction.

The operating programs of the DATANET-30 must have bit 7 set in command word 2 (CW2) following the select command (either read or write) to lockout the GE-225 from gaining access to the DSU. The lockout is released when the DATANET-30 gives a read or write command with bit 7 off. When the DATANET-30 takes control again, bit 7 must be set again.

The GE-225 must also set the lockout bit in order to do error checking. This lockout must be released immediately after the error check. If it is not and the DATANET-30 needs access to the DSU, the DATANET-30 can take control and possibly cause an error condition in the GE-225.

The lockout may be released by the DATANET-30 by issuing a seek followed by a read after write (RAW) of one record with bit 7 off.

The following example illustrates the use of the lockout bit.

DATANET-30 Operation

The GE-225 test will see the following as a result of issuing test and branch instructions.

(a)	SEEK	Not ready
(b)	READ W/BIT 7 ON ERROR TEST	Not ready
(c)	SEEK	Not ready
(d)	WRITE W/BIT 7 ON ERROR TEST	Not ready
(e)	SEEK	Not ready
(f)	RAW 1 RECORD W/BIT 7 OFF	Ready

Notice that the write instruction must have the lockout bit ON to be sure to retain control so that a test may be made for any errors.

## PROGRAM CONTROL

Bit 8 of CW2 (memory location 4) is used to denote the master in the following manner:

Bit 8 - CW2

Function

On

All subsequent simultaneous requests for access will consider the computer issuing this command as the master.

Off

The master computer is the one that issued the last command that had bit 8 of CW2 on.

When the controller is initialized by use of the RESET button and the switch is in program control position, the computer on Regular plug #n will be the master until a change is indicated by use of bit 8 of CW2.

Bit 8 of command word 2 (CW2) for the DATANET-30 and bit 12 of command word 1 of the GE-225 following the select command, should be OFF in both the DATANET-30 and GE225 DSU instructions. Program control of which computer is to be master is not necessary, since the DATANET-30 is always the master because it is on the Regular plug and the three-way switch is in the regular position.

## AUTOMATIC PRIORITY INTERRUPT (API)

The API signal generated may be controlled in the following manner:

1. API will be inhibited to both computers when the API inhibit switch is set to "inhibit" or "off" mode. This switch is inside the controller on the test panel.
2. With this switch in the "allow" or "on" mode, API signals will be generated for the four standard reasons as defined in the DSU manual and will be available under the following conditions:

<u>Bit 6 - CW2</u>	<u>Function</u>
Off	Will allow the API signal only to the computer on the regular plug. Source of the API may be (1) the regular computer or, (2) the added computer provided the regular computer had issued any DSU test and branch.
On	Will allow the API signal only to the computer on the added plug. Source of the API may be either the added computer or the regular computer provided the added computer had issued any DSU test and branch.

If the DAC is in the API allow mode and an API signal is generated, the signal will be valid only for the GE-225. The DATANET-30 does not have API. Therefore, bit 14 of CW2 may be ON only in the GE-225 DSU instruction. Bit 6 in the CW2 of the DATANET-30 DSU instruction should be OFF.

## TEST AND BRANCH CONDITIONS

All test and branch conditions, as defined in the GE-225 DSU Reference Manual, and the following four may be interrogated at any time by either computer. The ability of the "locked out" computer to test conditions may lend itself to developing control and monitor routines.

### Additional Test and Branch Conditions for the GE-225

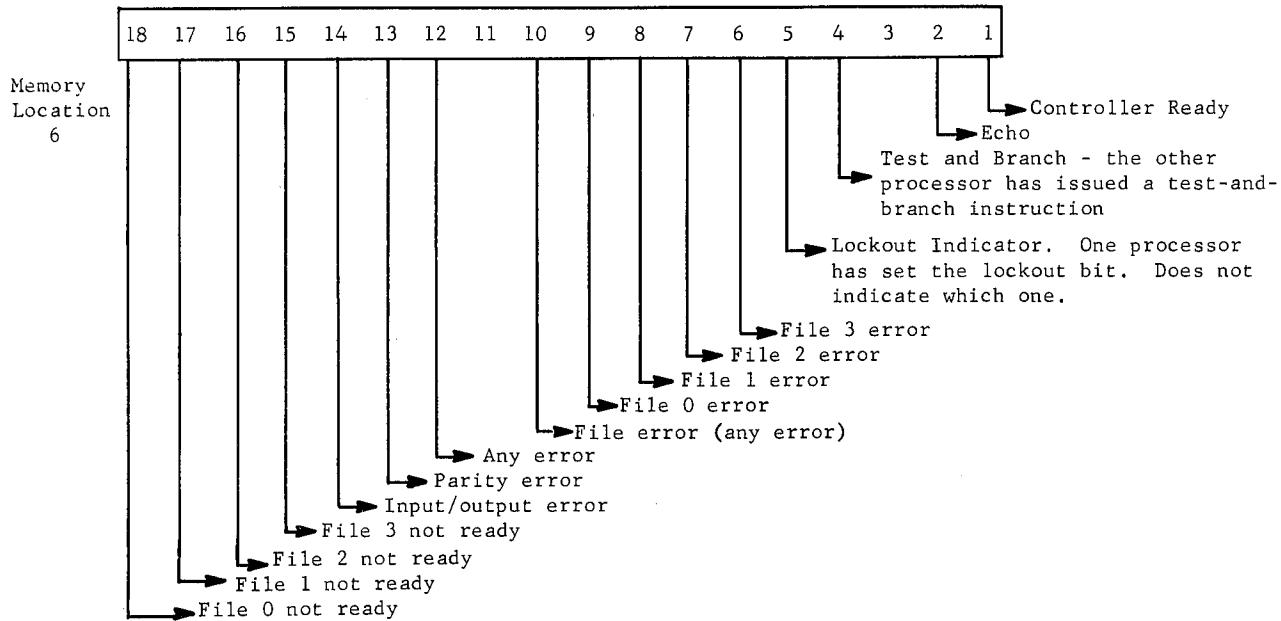
<u>Condition</u>	<u>Octal</u>
Lockout bit has been set by either Computer	2514P36
Lockout bit has not been set by either Computer	2516P36

<u>Condition</u>	<u>Octal</u>
Flip-flop is set indicating other Computer did perform a test and branch instruction	2514P37
Flip-flop is reset indicating other Computer did not perform a test and branch instruction	2516P37

### Additional DATANET-30 Test and Branch Conditions

When a DAC is part of the system, two more bits in memory location 6 take on value in a CRS command. These are bits 4 and 5.

1. Bit 4 when ON, indicates that the other processor has issued a branch or select command.
2. Bit 5 when ON, indicates that a processor has the lockout set. Note that this does not distinguish which processor.



### ALTERNATE ACCESS

The DAC is designed to grant alternate access to the DSU if the lockout bit is not set by a processor. The access is granted to a processor immediately after the execution of a DSU read or write operation of the other processor.

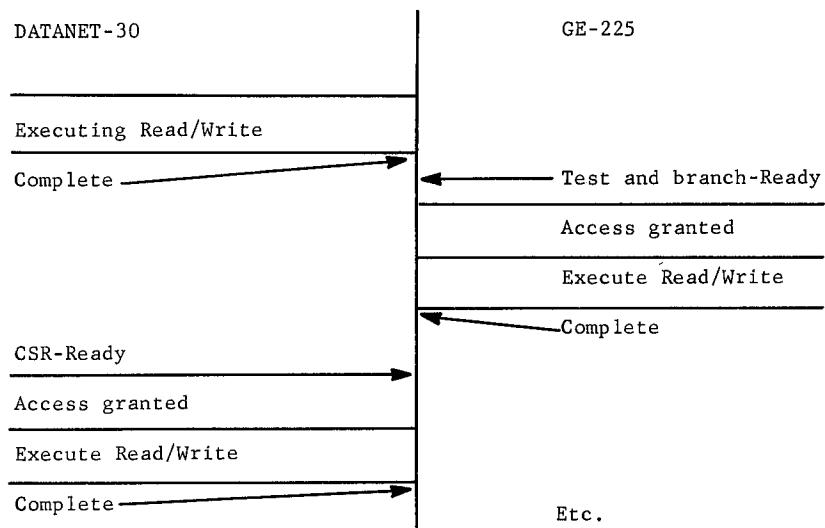
An error condition can be created if the processor that granted access to the DSU did not check for DSU ready before attempting to gain access. The error occurs because the other processor could be transferring data when it lost control of the DSU.

Access is requested by a branch or test instruction, that is, a CSR from the DATANET-30 will cause the DATANET-30 to be given access to the DSU as soon as the DSU is available.

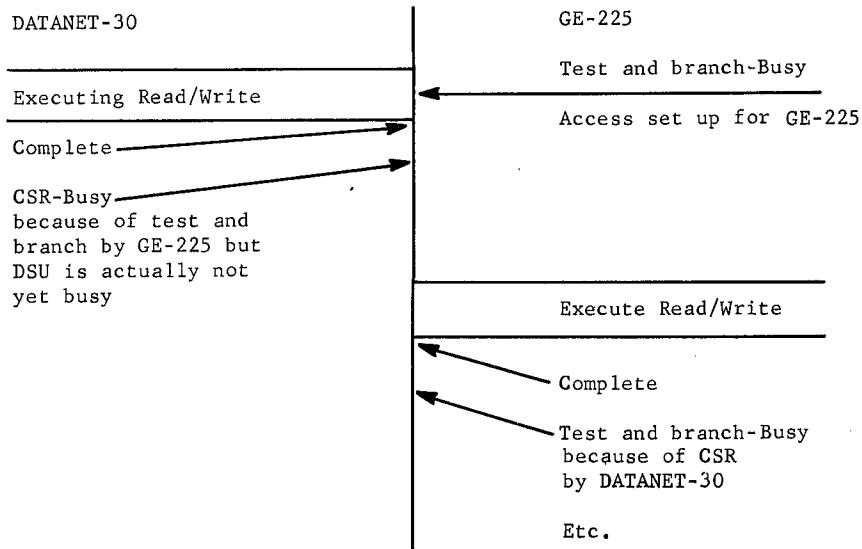
The GE-225 requests access by a DSU ready test or a DSU error test. Therefore, the programs should only test the DAC when a DSU operation is ready to be performed.

## CONDITIONS FOR ACCESS

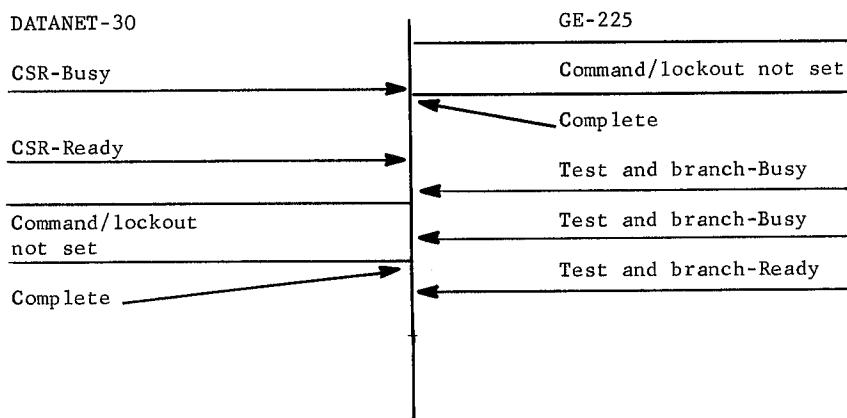
### 1. LOCKOUT BIT OFF



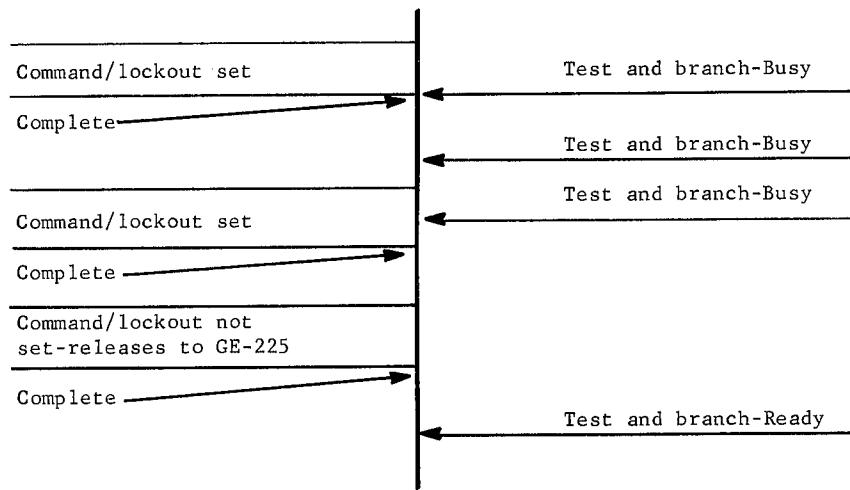
### 2. LOCKOUT BIT OFF



### 3. LOCKOUT BIT OFF



### 4. LOCKOUT BIT ON



## SAMPLE FLOW CHARTS

Figure 3 is an example of a DSU subroutine for the GE-225. On entering the subroutine, the GE-225 locks out the DATANET-30 by setting bit 13 in CW1. Then, the seek is performed and error tested. Three seek commands are attempted before an error exit occurs. If the seek is complete, then the read or write command is performed and error tested. Three read or write commands are attempted before an error exit occurs. Before the exit from the subroutine occurs, the GE-225 releases the lockout.

In order to reread or rewrite the DSU following an error, the DSU must be reinitialized; that is, reseek, then reread/rewrite.

Error messages (to the typewriter for example) must be written after release of the DSU to the DATANET-30.

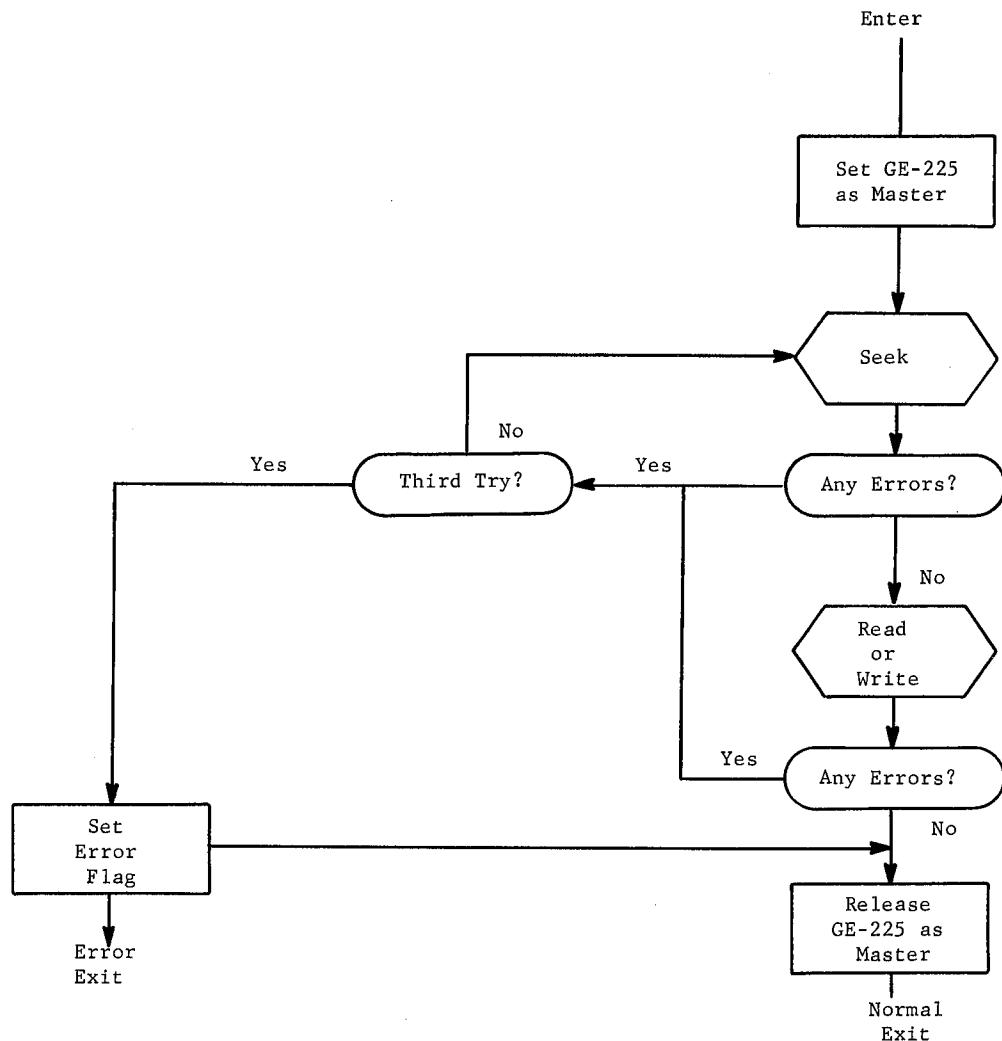


Figure 3. Sample DSU Subroutine for GE-225

Figure 4 is an example of a portion of a DATANET-30 executive program to determine if the DATANET-30 requires access to the DSU and how to take control if access is not granted.

When the DATANET-30 has control of the DSU, the program flag is set. The flag is necessary because the CSR command only tells whether a processor has the lockout bit set not which processor. If this flag is set, the DATANET-30 can continue with the DSU processing because the GE-225 is locked out. If the flag is not set, then the CSR is interrogated to see if the GE-225 did set the lockout bit.

If the GE-225 did set the lockout bit, the DATANET-30 then waits for at least 2 seconds for the GE-225 to reset the lockout. After 2 seconds have elapsed and the DSU is not released, the DATANET-30 takes control.

The DATANET-30 takes control by issuing a position and read with bit 7 ON in CW2 of the read command. There should be provisions for two retries of the position command. One position command may not take control from the GE-225.

When the DATANET-30 completes the DSU real-time processing, it releases the lockout of the DSU. The fastest way to do this is to do a position command with the read next sector bit on and then do a read after write (RAW). The GE-225 should release the lockout the same way.

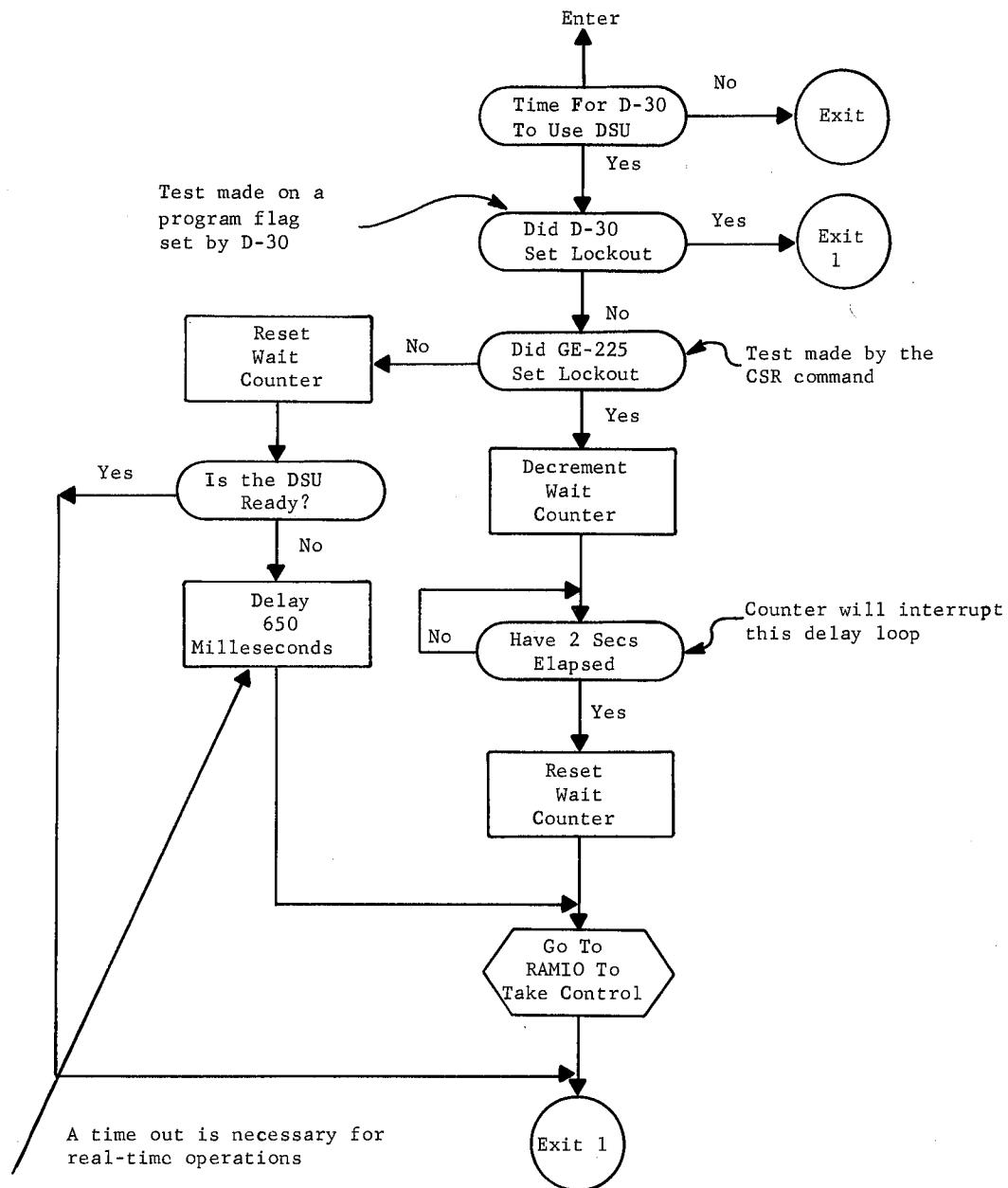


Figure 4. Part of Executive Program for DATANET-30.



## CONTENTS

	Page
APPENDIX J - PUNCH READER UNIT (PRU-930)	
General Description . . . . .	J-1
Block Diagram . . . . .	J-1
Eighty-Bit Row Register . . . . .	J-1
Data Flag . . . . .	J-1
Punch Ready Flag . . . . .	J-2
Punch Magnet Drivers . . . . .	J-2
Instruction Repertoire . . . . .	J-3
Drive External Function Instructions . . . . .	J-3
Load Transmit Drivers Instructions . . . . .	J-3
Register Transfer Instructions . . . . .	J-3
External Status Lines Instructions . . . . .	J-4
Controls and Indicators at the DATANET-30 . . . . .	J-4
Programming Considerations . . . . .	J-4
Corner Turning . . . . .	J-4
Data Word for the Row Register . . . . .	J-5
Buffer Selector Addresses . . . . .	J-7
Filling the Row Register . . . . .	J-7
Punching Cycle . . . . .	J-8
Punch Timing . . . . .	J-8
Card Format . . . . .	J-9
BCD to Hollerith . . . . .	J-10
Programming Example . . . . .	J-11

## ILLUSTRATIONS

Figure	Page
J-1	Card Punch Block Diagram . . . . .
J-2	Data Word Transfer to the Row Register . . . . .
J-3	DATANET-30 Instruction Card . . . . .
J-4	Forming the Data Word . . . . .



## APPENDIX J

### PUNCH READER UNIT (PRU-930)

#### **GENERAL DESCRIPTION**

The punch reader unit (PRU-930) is composed of the card reader unit and the card punch unit. The card reader unit is covered in Appendix K. All references to the PRU-930 in this appendix will refer to the card punch unit only.

The card punch used with the PRU-930 is the GE-200 Series 100-card-per-minute punch. It operates via the buffer selector and requires 8 buffer selector addresses specified in the buffer selector channel address plug for the PRU. The addresses may be anywhere between 1 and 127. The PRU-930 consists of 2 modules and can only be mounted in rack 2.

Functional control of the PRU-930 is by program via the buffer selector interface. Addressing the PRU is done by putting the buffer selector channel address in the C-register.

#### **BLOCK DIAGRAM**

A block diagram of the card punch is shown in Figure J-1. Refer to the diagram for items discussed in the following paragraphs.

#### **Eighty-Bit Row Register**

The 80-bit row register is loaded via the buffer selector in blocks of 10 bits. Each block of 10 bits has a separate address on the buffer selector. Data is loaded into the row register with either an LDT or TRA instruction. The row register is reset to zero automatically after each row is transferred to the punch.

#### **Data Flag**

When the data flag is set (NES 2 = 1), it indicates that the row register can be loaded with the next row of data. The data flag is not set until a row is punched. There are 24 milliseconds after it is set in which to load the row register. Even if the register is not filled at the time for punching, the configuration in the register is punched.

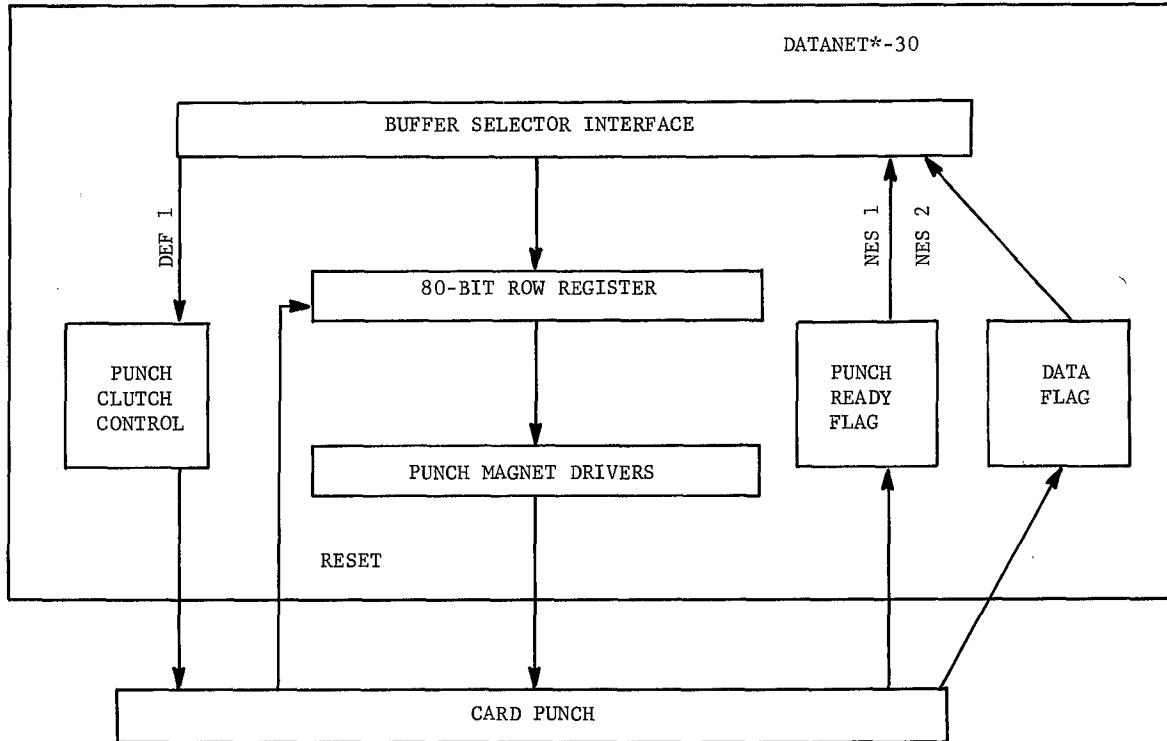


Figure J-1. Card Punch Block Diagram

### Punch Ready Flag

If the punch ready flag is set (**NES 1 = 1**), it indicates that the punch is ready in all respects to punch cards. The conditions which cause a not-ready signal are listed under the **NES 1** instruction.

### Punch Magnet Drivers

The signal from the punch magnet drivers is used by the punch to determine which positions in a row to punch. A 1-bit in the row register activates a magnet driver, which in turn activates the proper punch magnet.

---

\* Reg. Trademark of the General Electric Company.

## INSTRUCTION REPERTOIRE

Card punch instructions are classified under buffer selector instructions for the DATANET-30. They are the Drive External Function (DEF), External Status Lines (NES), Register Transfer, and Load Transmit Drivers (LDT) instructions. The rules for using the DEF, NES, and Register Transfer instructions are the same as those for other DATANET-30 buffer selector instructions. The LDT instruction is special.

### Drive External Function Instructions

The DEF instructions are as follows. (Note: The DEF instructions are effective only when the buffer selector channel address of the first 10 bit positions of the row register is in the C-register.)

- DEF 1      The clutch in the punch mechanism is activated to feed a card to be punched. The data flag is also set to 1.
- DEF 2      Resets the data flag. (The DATANET-30 gives a DEF 2 automatically following the execution of a TRA  $\rightarrow$  T instruction).
- DEF 3-9     Not used.
- DEF 0      Resets the data flag. (The DATANET-30 gives a DEF 0 automatically following the execution of an LDT instruction. No other use of this instruction can be made by the program.)

### Load Transmit Drivers Instruction

The LDT instruction is as follows.

- LDT  $\rightarrow$    A word is transferred from a specified memory location to the PRU row register positions addressed by the buffer selector channel address in the C-register. (A DEF 0 instruction is automatically executed as part of the LDT instruction.)

### Register Transfer Instructions

The TRA instructions are as follows.

- TRA  $\rightarrow$ , T   Data is transferred from a working register to the PRU row register positions addressed by the buffer selector channel address in the C-register.
- TRA R, $\rightarrow$    Not used.

## **External Status Lines Instructions**

The NES instructions are as follows. (Note: The NES instructions are effective only when the buffer selector channel address of the first 10 bit positions of the row register is in the C-register.)

**NES 1** A 1-bit indicates the punch is ready to punch a card. A 0-bit indicates any one of the following:

- a. Busy flip-flop "on."
- b. Punch end line from the punch mechanism is energized.
- c. Punch ready line from the punch mechanism is not energized.
- d. DPBC error.

**NES 2** This is the data flag. A 1-bit indicates that the row register is ready to receive the next row of data.

**NES 3-0** Not used.

## **CONTROLS AND INDICATORS AT THE DATANET-30**

There are no manual controls or indicators on the punch portion of the PRU-930. All controls and indicators for the punch portion are by program only. The manual controls and indicators for the punch mechanism are all on the punch operating panel.

## **PROGRAMMING CONSIDERATIONS**

### **Corner Turning**

One principal operating characteristic of the card punch mechanism is that a card is punched one row at a time, starting with row 12. Rows 12 and 11 are not punched on a binary card; but they still exist and must be accounted for in the DATANET-30 program when programming to punch a card.

Another related principal characteristic of the card reader mechanism is that a card is read one column at a time, starting with column 1. An 80-column card, fully punched, therefore presents 80 possible read, or data, conditions to a computer.

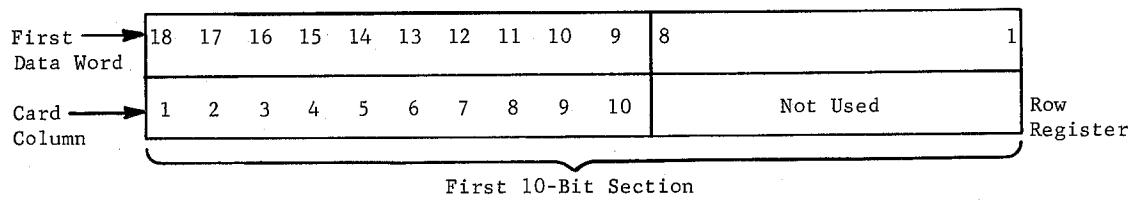
It is therefore necessary to present the data to the card punch row-by-row in such a way that, upon completion of the punching operation, the data can be read column-by-column. This rearrangement of data for punching purposes is known as "corner turning." Most computers accomplish corner turning by hardware. In the DATANET-30, the corner turning must be done entirely by program.

The 80 columns of a card represent 960 bit positions (80 columns x 12 rows = 960). Each row contains 80 bit positions; the row register in the PRU-930 thus represents the 80 column (bit) positions of the particular row to be punched. Loading the row register with the desired bit configuration for the row to be punched is the problem which has been solved as follows.

The row register has been designed to accept 10 data bits at a time. The 10 bits are transferred to the row register by an LDT or TRA instruction, according to the buffer selector channel address in the C-register. The C-register will contain one out of eight buffer selector addresses. To fully load the row register, 8 transfers of 10 bits each are required, each transfer to a different 10-bit section of the row register. Each transfer is from a definite memory location or working register.

### Data Word for the Row Register

The data word for transfer to the row register has the following format. This word may be formed in memory or a working register.



The data for the row register is left-justified. When the first data word is transferred to the row register, bit 18 goes into bit 1 of the row register, bit 17 into bit 2, etc. When the second data word is transferred to the row register, bit 18 goes to bit 11 of the row register, bit 17 to bit 12, etc.

The from/to positions are illustrated in the following chart and in Figure J-2.

<u>DATANET-30 Word Bit Pos.</u>	<u>Card Column No.</u>
18	1, 11, 21, 31, 41, 51, 61, 71
17	2, 12, 22, etc.
16	3, 13, 23, etc.
15	4, 14, 24, etc.
14	5, 15, 25, etc.
13	6, 16, 26, etc.
12	7, 17, 27, etc.
11	8, 18, 28, etc.
10	9, 19, 29, etc.
9	10, 20, 30, etc.

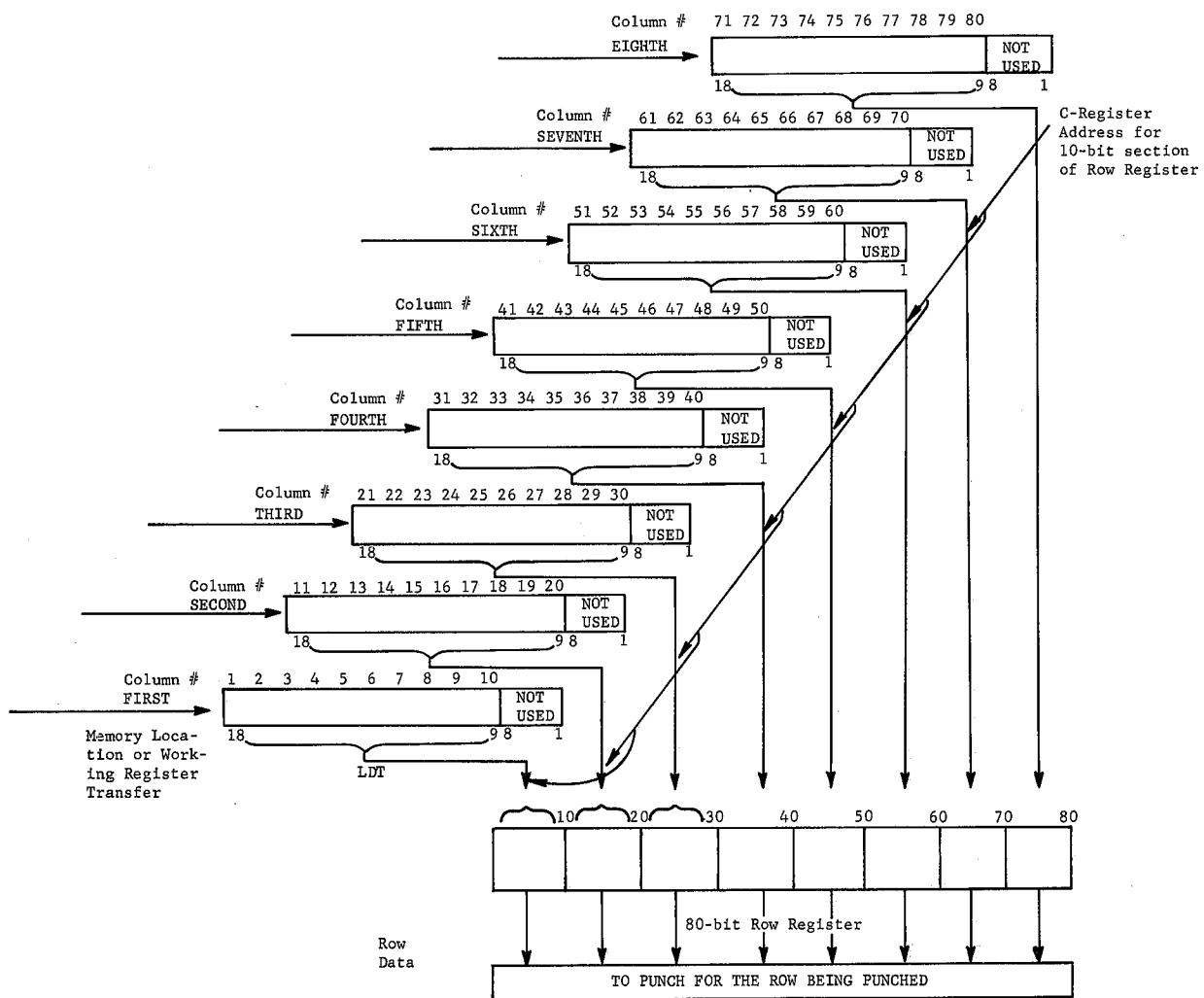
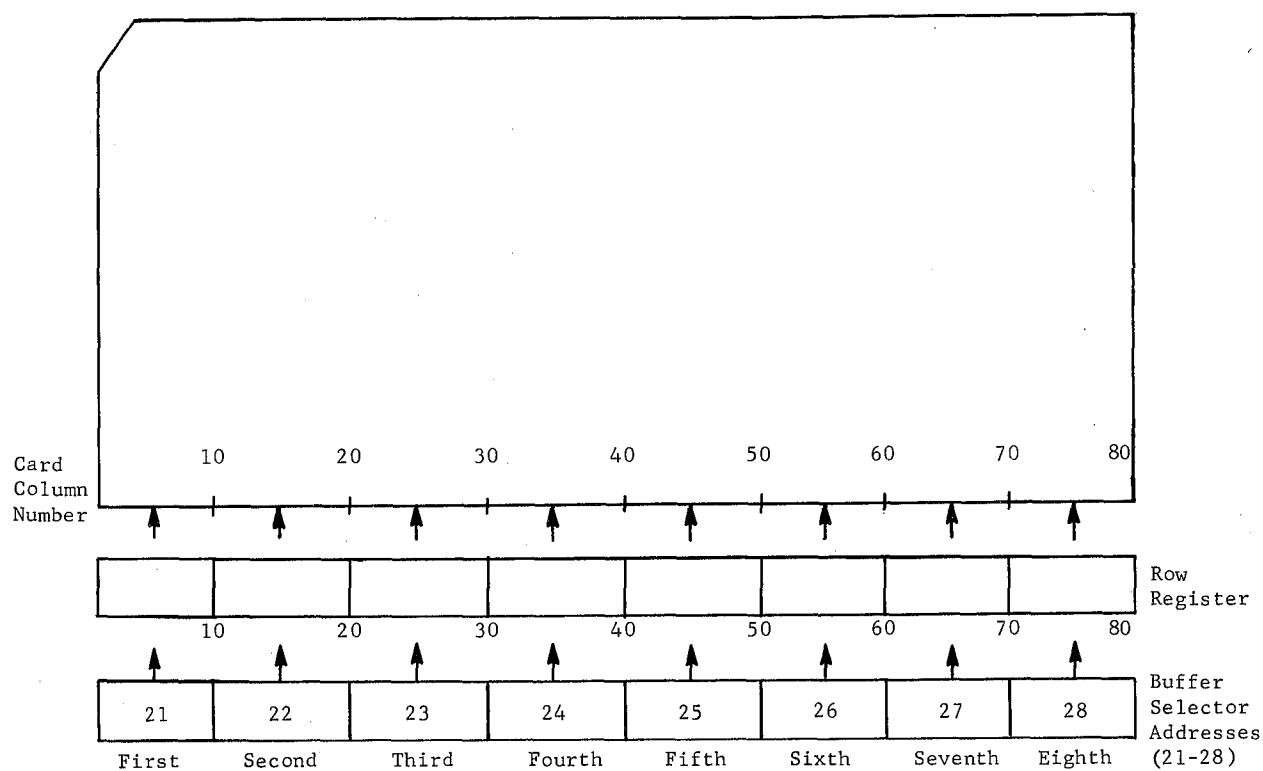


Figure J-2. Data Word Transfer to the Row Register

## Buffer Selector Addresses

Eight buffer selector addresses are used to load the 8 blocks for the row register into the desired location in the row, as shown below. (For purposes of illustration, buffer selector channel addresses 21-28 have been used.)



## Filling the Row Register

The following coding shows the steps necessary to fill the row register for a punch card program. It is assumed that the row register data words have been formed (the corner has been turned), a punch instruction has been issued, the punch is operational, and the program can wait for the row register to become ready to receive the next row. This is not necessarily the coding that would be used in an operating program.

TRA 0,A			
PNCHCARD	PIC	PUNCHADD	Address of first 10 row register bit positions
	NES	2	Is Data Flag Set?
	BNZ	*-1	No, check again
PNCHCARD 1	LDT	FIRST	Load positions 1-10 of row register
	AIC	1	Address of second 10 bits
	LDT	SECOND	Load positions 11-20 of row register

AIC	1	Address of third 10 bits
LDT	THIRD	Load positions 21-30 of row register
AIC	1	Address of fourth 10 bits
LDT	FOURTH	Load positions 31-40 of row register
AIC	1	Address of fifth 10 bits
LDT	FIFTH	Load positions 41-50 of row register
AIC	1	Address of sixth 10 bits
LDT	SIXTH	Load positions 51-60 of row register
AIC	1	Address of seventh 10 bits
LDT	SEVENTH	Load positions 61-70 of row register
AIC	1	Address of eighth 10 bits
LDT	EIGHTH	Load last 10 positions of row register
AMA	ROWCOUNT	Rowcount = 1
XAZ	COUNT 12	Last row?
BNZ	PNCHCARD	

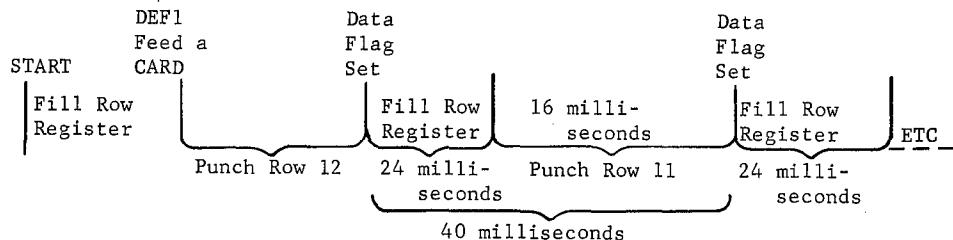
(Repeat 12 times, once for each row. Program must count rows punched.)

### Punching Cycle

The program must start a punch cycle (DEF 1). Once the cycle has started, the program must fill the 80-bit row register in time to punch the next row. There are approximately 40 milliseconds between one punch and the next. The timing of actually punching the row is a function of the punch mechanism. A flag is set (NES 2) after each row has been punched. The program must maintain a count of the rows punched.

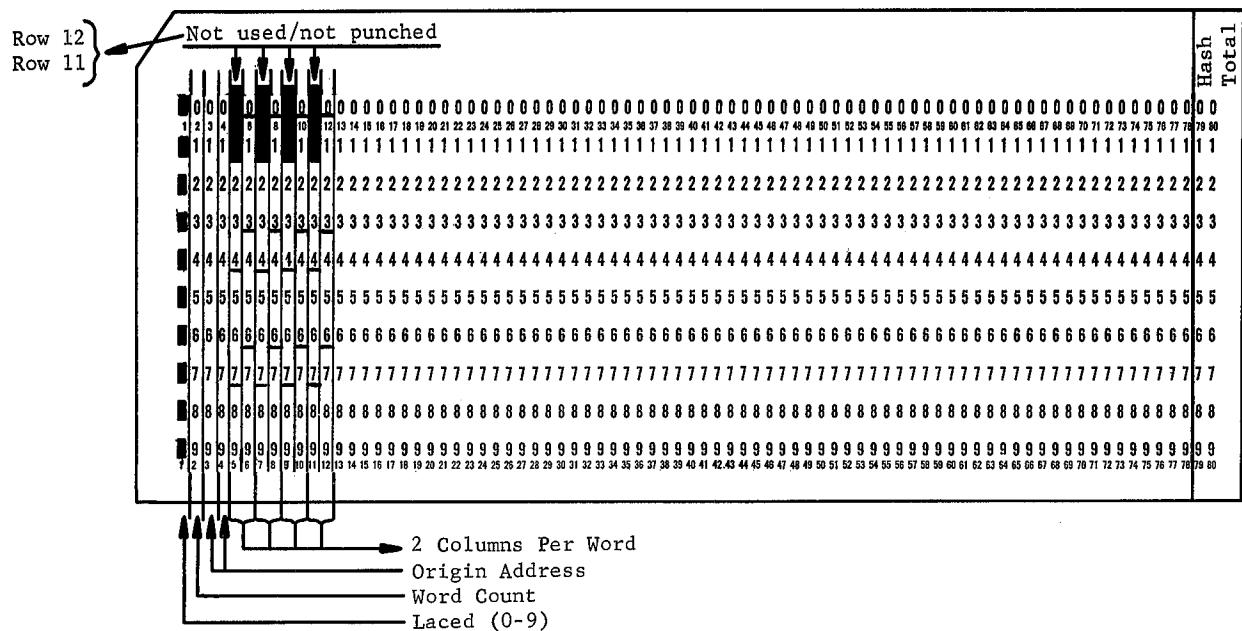
### Punch Timing

The row register cannot be filled with the new punch information until after the data flag is set. The actual punch operation requires 16 milliseconds out of the 40 milliseconds between rows. This leaves 24 milliseconds during which time the program must fill the row register with the information to be punched in the next row. (See diagram below.)



## Card Format

The format for punching a card is determined by the format in which the card will be read (BCD, Hollerith, etc.). One DATANET-30 Binary Instruction card format will be discussed here. (See Figure J-3.)



Control information in first four columns

Figure J-3. DATANET-30 Instruction Card

Since there are 12 rows per card, punch information must be supplied 12 times per card. Even though the 12 and 11 rows will not be punched, the information of no punches in these rows must be given to the punch. Therefore, the data words for the row register must contain all zeros.

When it is time to put row 0 information in the row register, the data words must be formed for all columns to be punched in that row according to Figure J-4. The process continues for rows 1, 2, etc.

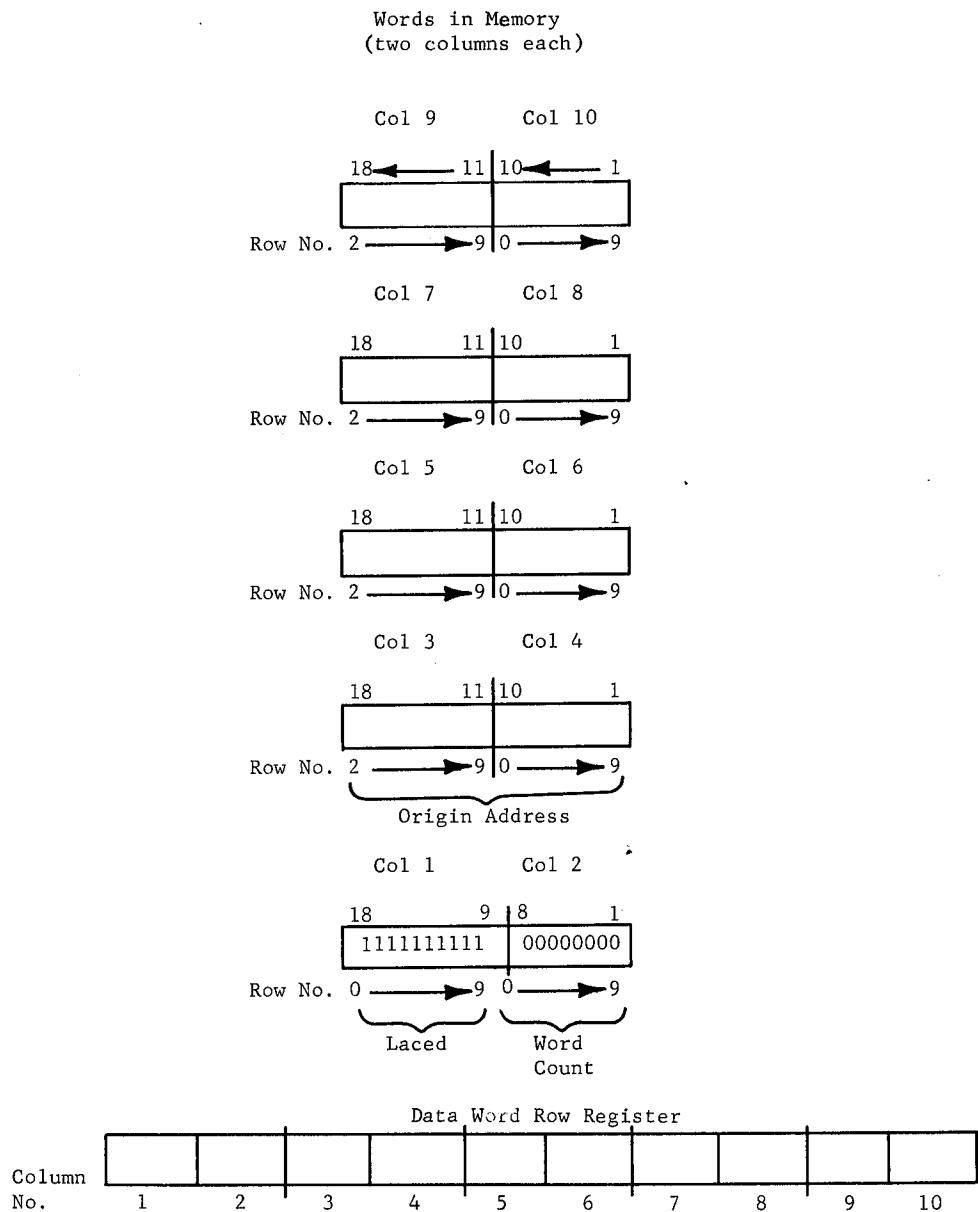


Figure J-4. Forming the Data Word

#### **BCD to Hollerith**

When punching in Hollerith code, a BCD-to-Hollerith conversion must be made before the row register data word is formed. The word is formed in an identical manner, so that the punches on the card are in Hollerith code.

The various programming methods for punching a card will not be discussed here. Examination of the instruction repertoire will suggest at least two: (1) transfer of data word from memory or (2) transfer of a data word from a working register.

## PROGRAMMING EXAMPLE

The following example shows one method to turn the corner and punch a card. However, if a DATANET-30 binary card format is to be punched, the example does not include the laced punching of column 1, a word count for the card, origin address for columns 3 and 4, or a hash total for the card. The example is intended to show an approach and some programming considerations for punching a card. This is not necessarily the way it would be done in an operating program.

### Calling Sequence to Punch Subroutine

	LDA ADDATA	Pick up data address to be punched.
	BRS PUNSBR	
PUNSBR	IND 0	
	IND *+1	
	PIC PUNADD	Select punch.
	NES 1	
	BZE *-1	{ Wait punch ready.
	DEF 1	
	STA DATA~A	
	STA DATA~B	{ Save address of output data.
PUN~A	LDC ROWCNT	
	NCZ 112	
	BZE PUN~B	
	ADO ROWCNT	
	BRU PUN~D	{ Test if row 12 or 11. Row is 0 through 9. Go punch dummy rows 11 and 12.
PUN~B	LDA DATA~A X	
	NAZ MASK~1 X	
	BZE *+2	
	RMB D1	
	SL1 B,B	{ Test for bit depending on row count. Set B = 1.
	NAZ MASK~2 X	
	BZE *+2	
	RMB D1	
	SL1 B,B	
	SBO PUN~Z	Test if pattern word is filled.
	BZE PUN~C yes	
	ADO DATA~A	
	BRU PUN~B	
PUN~C	SL6 B,B	
	SL1 B,B	{ Position pattern word.

PIC	5		X	{ Reset pattern word count. Pick pattern table count. Save P register in pattern table.  Test if complete pattern for a row is done.
STC	PUN~Z			
LDC	PAT~CNT			
STB	PAT~TAB			
AIC	1			
STC	PAT~CNT			

ADO	ROWCNT		{ Increment row count Reset data address.
LDA	DATA~B		
STA	DATA~A		
STZ	PAT~CNT		

PUN~D

PIC	PUNADD		{ Send pattern to punch.
NES	2		
BZE	*-1		
LDT	PAT~TABA	X	
AIC	1		
XCZ	PUNADD+9		

PIC	0		{ Clear pattern area.
STZ	PAT~TAB	X	
AIC	1		
XCZ	8		
BNZ	*-3		

LDC	ROWCNT		{ Test if card has been punched. Set row count = row 12.
XCZ	10		
BNZ	PUN~A		
PIC	126		
STC	ROWCNT		

BRU	PUNSBR	X	Exit subroutine.
-----	--------	---	------------------

DATA~A	DEC	0	
DATA~B	DEC	-2	
ROWCNT		5	
PUN~Z	DEC	0	
PAT~CNT	INC	*+2-PUNADD	
PAT~TABA	INC	*+1	
PAT~TAB	DEC	0	Column 1 - 10
			11 - 20
			21 - 30
			31 - 40
			41 - 50
			51 - 60
			61 - 70
			71 - 80

MASK~1

INC	*+1	
OCT	0	Row 0
		1
OCT	-0	2
OCT	200000	3
	100000	4
	40000	5
	20000	6
	10000	7
	4000	8
OCT	2000	9

MASK~2

INC	*+1	
OCT	1000	Row 0
	400	1
	200	2
	100	3
	40	4
	20	5
	10	6
	4	7
	2	8
OCT	1	9

PUNADD

DEC

Address of 1st punch.



## APPENDIX K

### CRU-930 CARD READER UNIT

#### GENERAL DESCRIPTION

One Card Reader Unit (CRU-930) connects the CRF-930 card reader to the DATANET-30. The Card Reader Unit occupies one option module space and is assigned an address on the buffer selector. The buffer selector address is specified by the address plug for the CRU.

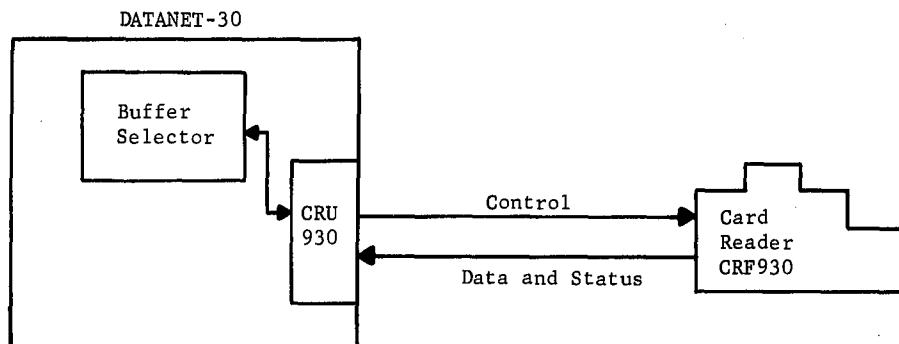


Figure 1. DATANET-30 With A Card Reader

#### INSTRUCTIONS

##### DEF Instructions

- DEF 1--Reset data flag
- DEF 3--Read card
- DEF 5--Reset alarm
- DEF 2, 4, 6-10--Not used

DEF 1. Reset the data flag. The data flag and column register are reset. This instruction occurs automatically with a register transfer instruction such as TRA R,B.

DEF 3. Read a card. The clutch in the card reader is activated to feed one card. The card reader goes not ready. The Data Flag, the Column Register and Reader Alarm signals are reset.

DEF 5. Reset alarm. The Reader Alarm signal is reset.

## NES Instructions

- NES 1--Data flag
- NES 3--Hopper full
- NES 4--Card reader ready
- NES 5--Reader alarm
- NES 2, 6-10--Not used

NES 1. Data flag. A one indicates that the column register contains data.

NES 3. Hopper full. A one indicates that cards are in the hopper. This line goes to zero when the last card leaves the hopper.

NES 4. Card reader ready. A one indicates that the card reader is ready for a Read Card instruction.

NES 5. Reader alarm. A one indicates an alarm condition of either no card on the sensing platform when a Read Card instruction was given or a card jam occurred.

## BLOCK DIAGRAM

The functional block diagram, Figure 2, shows the 12-bit column register, the data flag, the hopper full, ready and reader alarm flags, and the read card control.

The 12-bit column register receives data from the card reader column by column as the card is read. The data flag is set after data is in the column register. The data flag and column register are reset when the data is transferred out of the column register.

The hopper full flag is used to indicate that cards are in the hopper. When the last card leaves the hopper, this is indicated by NES 3 = 0. When the hopper is empty, a Read Card instruction is ignored and the last card will remain on the feed platform. It is therefore necessary to place at least one blank card following the last card to be read.

The ready indicator will indicate a ready condition when the card reader is not busy reading a card, when there is at least one card in the hopper, and if there is no reader alarm.

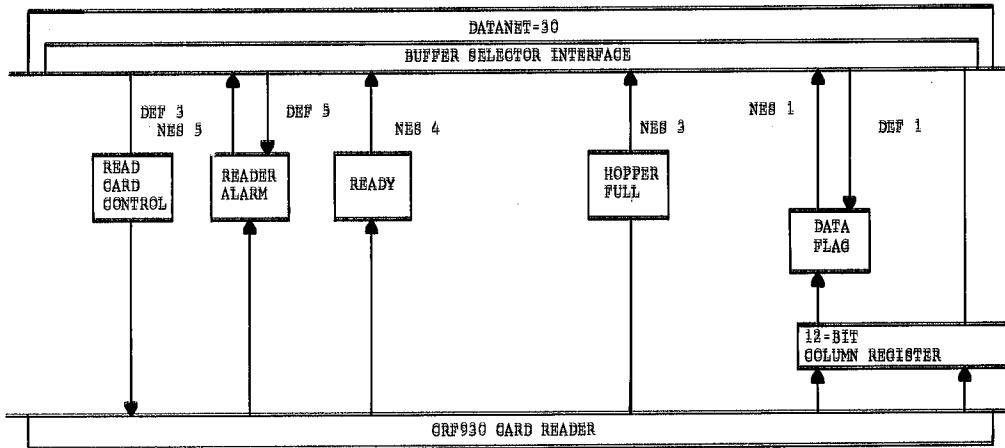


Figure 2. CRU-930 Block Diagram

DATANET-30

Reader alarm is set (NES 5 = 1) to indicate that a card was not fed after a Read Card instruction. This would indicate either a misfeed or card jam condition. A Read Card instruction will not be executed as long as the Reader Alarm signal is set. DEF 5 will reset the alarm indication, but should not be used without some indication to the program that action has been taken to clear the card reader.

Read card control functions are as described in the DEF 3 instruction.

## CONTROLS AND INDICATORS

The card reader unit (CRU) module has one light and one switch. The light turns on to indicate a card reader alarm. The switch, primarily used for maintenance purposes, has three positions.

Center position--normal  
Up position--cards are fed continuously  
Down position--stops feeding cards continuously

The switch must be returned to center position for normal operation.

All other controls or indicators are by program or on the card reader.

## DATA TRANSFER

The cards are read column by column. A punch equals a one; no punch equals a zero.

Data in any of the 12 rows per column will be transferred from the card reader to the 12-bit column register. There is no decoding in either the card reader or the card reader unit module (CRU-930).

From the 12-bit column register, the data is received via the buffer selector over the Receive Data lines 1-12. The card row number to the Receive Data line transfer is as shown in Figure 3. (See page 4.)

Thus, data transferred with a TRA R,B will be right-justified. Data transferred with a circulate or shift instruction will be justified in accordance with direction of circulate or shift.

## CARD FORMAT

There is no code identification or conversion in the CRU-930.

The card reader unit transfers data straight through so that the 12-bit positions in a register correspond to the row position in each column of a card. The DATANET-30 program must do the code conversion and formatting of a word in memory.

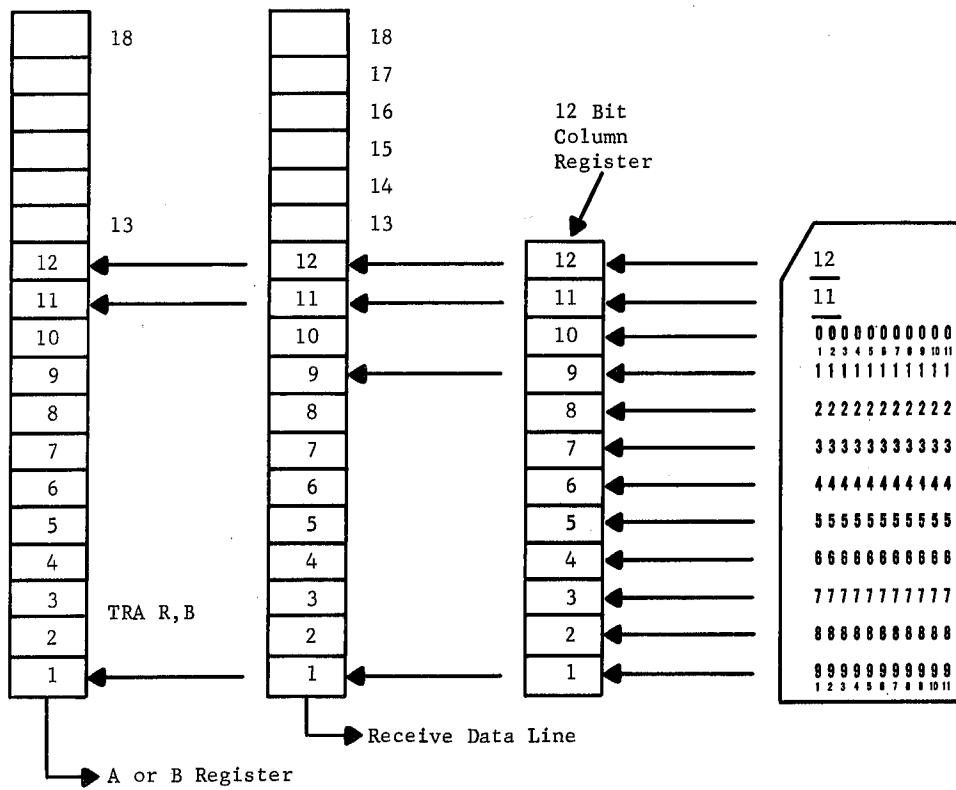


Figure 3. Card Row Number to Receive Data Line

The card format for binary program cards is established by the Card Loader routine (CDD30B1.001) to recognize binary card data as described below. Refer to Figure 4.

COLUMN 1. Rows zero through nine of column 1 are punched for a binary card.

COLUMN 2. Number of DATANET-30 words contained on the card is punched in column 2.

COLUMNS 3 AND 4. These two columns are punched to indicate the starting address in memory.

ODD-NUMBERED COLUMNS. The odd-numbered columns contain 8 bits of each word.

EVEN-NUMBERED COLUMNS. The even-numbered columns contain 10 bits of each word. A pair of odd- and even-numbered columns comprise an 18-bit instruction word.

LAST TWO COLUMNS PUNCHED. The last two columns punched on the card contain the hash total (total punches on that card).

Since each instruction is comprised of an 18-bit word, 2 columns of the card are needed for a single word. The odd-numbered columns contain 8 of these bits and the remaining 10 bits are contained on the even-numbered columns. The most-significant digit of the DATANET-30 word is row 2 of the odd-numbered column and least-significant digit is row 9 of the even-numbered columns. The two columns per word are arranged in the proper order by program.

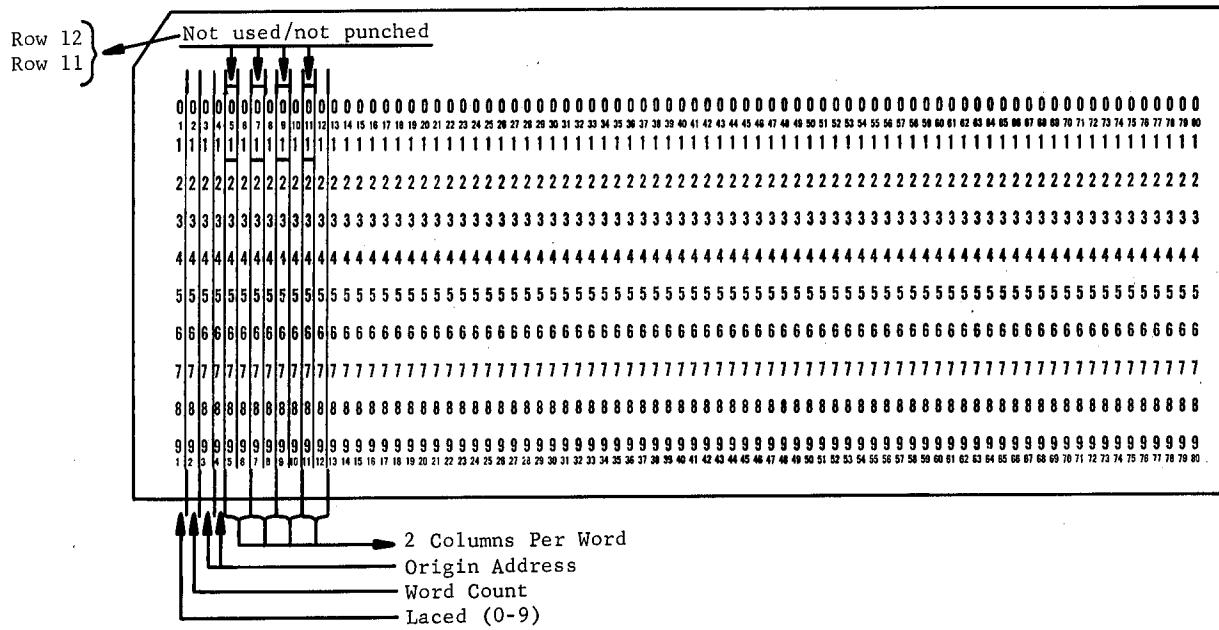


Figure 4. DATANET-30 Binary Format

## HASH TOTAL

The hash total may appear in any two columns including and following columns 7/8. The hash total columns are read in the same sequence as the rest of the card. Then the hash total compare is made with the accumulated total for the card. If the compare is not zero, an error occurred. The hash total is the arithmetic sum with no carry of all words, including word-count and origin, which precede the hash total.

## TRANSFER CARD

The first column is punched all ones, the same as a data card, but the second column is not punched in any row. The "no punch" in column 2 identifies the transfer card. Columns 3 and 4 contain the transfer address. The card contains the necessary information to return the program to the normal run after the card reader cycle is complete.

## LAST CARD

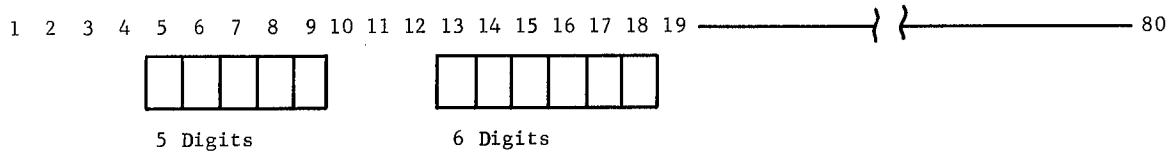
The last card in the deck is blank. This card holds the hopper switch open, preventing a premature halt. The card reader will automatically stop when this card is on the feed table.

## HOLLERITH CARD

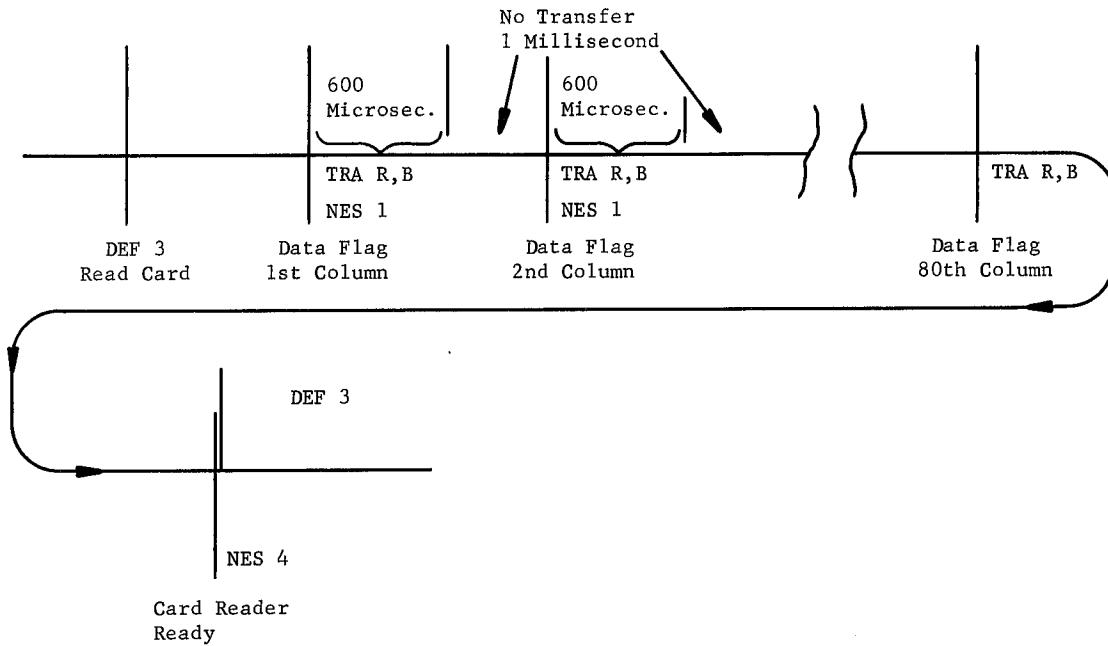
The Hollerith card is not identified as such by punches in the first column. Therefore, the absence of all punches in the first column indicates to the program that conversion from Hollerith to BCD is necessary.

## OCTAL PATCH CARD

The format for an octal patch card is 5 digits (memory location) with leading zeros starting at column 5 and 6 digits (the instruction) starting at column 13.



## TIMING CONSIDERATIONS



After a Read Card instruction, the program has approximately 600 microseconds to transfer the data out of the column register after the data flag is set. It is recommended that the program respond to the Data Flag Set signal within a few word times. If the hash total appears before the end of the card (word count equals zero), the rest of the card may be ignored.

A new Read Card command may be issued immediately upon detection of Card Reader Ready (NES 4).

## PROGRAMMING

There are two programs available from the program library. These are:

1. CDD30B1.001--A card loader for program cards and octal patches.
2. CDD30E1.001--A card reader subroutine to read Hollerith cards and convert to BCD.



## CONTENTS

	Page
THE CONTROLLER SELECTOR UNIT	
General Description .....	1
Controller Selector Instruction .....	1
Peripheral Command Words .....	2
Branch Conditions .....	3
Disc Storage Unit (DSU) .....	3
Disc Storage Unit Command Word Format .....	4
Position Command Words .....	5
Read/Write Command Words .....	7
Branch Conditions .....	8
Sample Coding to Address a DSU .....	9
Diagnostic Program Areas .....	10
Magnetic Tape Units .....	10
Decimal Mode .....	11
Binary Mode .....	11
Record Length .....	11
Magnetic Tape Instructions .....	12
Command Words .....	13
Programming Example .....	14
Tape Unit Conditions .....	15
Branch Conditions .....	15
High Speed Printer .....	16
Programming the High-Speed Printer .....	17
Command Words .....	18
Printer Instructions .....	18
Slewing Paper .....	19
Branch Conditions .....	20
Print Line Termination .....	20



## APPENDIX L THE CONTROLLER SELECTOR UNIT

### GENERAL DESCRIPTION

The controller selector is a common control and transfer point for such peripheral units as the GE-225 magnetic tape system, disc storage unit and printer. Through the use of plug-in connectors, peripheral units can be connected and interchanged according to the requirements of the system.

Eight peripheral controllers may be connected to the controller selector. Transfer of data to and from the DATANET\*-30 is on a memory interrupt basis. The eight peripheral controllers, numbered 0 through 7, operate on a priority basis, with the controller on the CSU channel 0 having the highest priority and the controller on channel 7 the lowest.

The CSU-931 is a synchronous controller selector allowing a transfer rate between the DATANET-30 and peripheral equipment of up to a maximum rate of 57,600 DATANET-30 words per second. The CSU interfaces with the interrupt interface and is allocated two cycles out of five; namely, cycles 1 and 3.

The following is the recommended controller selector channel priority assignment:

Channels 0, 1	--	Single-access disc storage unit controller
	--	Dual-access disc storage unit controller
Channels 2, 3, 4, 5	--	Magnetic tape controller
		Dual access magnetic tape controller
Channels 6, 7	--	High-speed printer controller

Each disc storage unit controller may have four disc storage units.

Each magnetic tape controller may have eight tape units.

Each high-speed printer controller may have one printer.

### CONTROLLER SELECTOR INSTRUCTIONS

There are four instructions which apply to the controller selector for all peripherals.

<u>Mnemonic</u>	<u>Operand</u>	<u>Word Times</u>
CSR	I	1
CONTROLLER STATUS REQUEST		Loads the image of the status lines of the peripheral controller specified by I into memory location 6. I is the plug number of the peripheral controller on the controller selector.

\*Reg. Trademark of the General Electric Company.

NIS

6

1

AND INTERNAL STATUS LINE

Interrogates the controller selector to determine if the CSR instruction has been completed and that the status is in memory location 6.

1 = The CSR is complete

0 = The CSR is not complete

From 3 to 10 word times are required after a CSR for status to be placed in memory.

NIS

7

AND INTERNAL STATUS LINE 7

Interrogates the controller selector to determine if the last select (SEL) command issued has been completed. Sets the branch zero flip-flop:

1 = the select is finished

0 = the select has not been completed

SEL

1

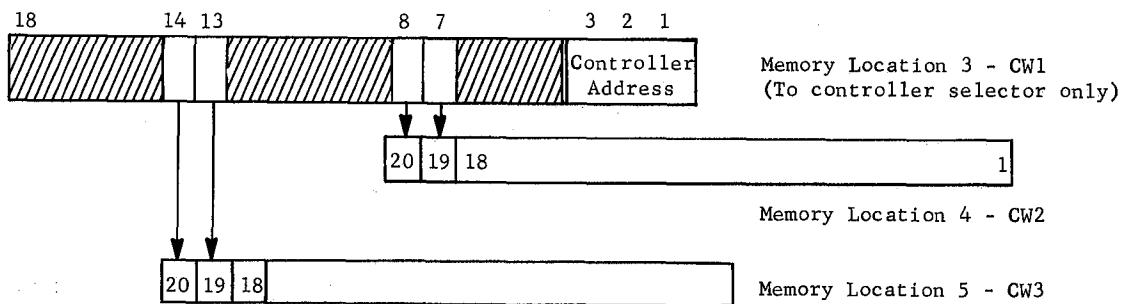
SELECT

Initiates operations as specified by locations 3, 4 and 5 of memory. This instruction is equivalent to a DIF 7.

## PERIPHERAL COMMAND WORDS

The peripheral command words are stored in memory in octal form, generally in a constants area. In order to transfer the command words to a peripheral, they are moved from the constants area to memory locations 3, 4, and 5; and a select (SEL) instruction is executed.

Memory location 3 contains the address of the selected peripheral in bit positions 1, 2, and 3. Since the commands for the peripherals are 20 bits in length and the DATANET-30 word only contains 18 bits, two bits must be added to the two peripheral command words contained in memory locations 4 and 5. The two extra bits for command word 3 come from positions 13 and 14 of location 3. Two extra bits for command word 2 come from positions 8 and 7 of location 3:



Upon executing a select instruction (SEL) the CSU will access memory cells 3, 4, and 5. The command words will be transmitted to the controller whose plug number corresponds to bits 1, 2 and 3 of memory location 3.

Bits 19 and 20 are taken from memory location 3 as shown.

NIS 7 will result in nonzero only after the command words have been transmitted and a controller has been selected for operation.

The controller selector stores the bits from positions 7, 8, 13, and 14 of command word 1 and automatically adds the bits 19 and 20 onto command words 2 and 3 when they are sent to the peripheral unit in order to have the proper length and bit configuration. This pattern is followed for all peripherals on the controller selector.

## BRANCH CONDITIONS

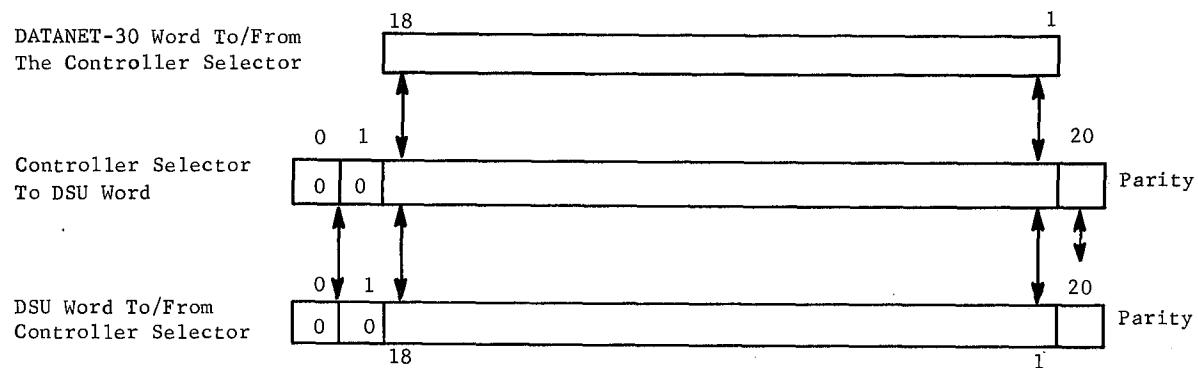
The status lines of a peripheral controller are classified as branch conditions for the particular peripheral device. Refer to the proper section of this appendix for the conditions relating to the particular peripheral device.

## DISC STORAGE UNIT (DSU)

This section contains only information special to programming the disc storage unit (DSU) from the DATANET-30. Additional information may be found in Compatibles/200 Disc Storage Unit and other related publications.

The maximum bit transfer rate between main memory and a DSU is 500kc (five hundred thousand bits per second). However, information is transferred between memory and the controller or between the controller and the DSU in groups of 18-bit words, or at a rate of 25kc (twenty-five thousand words per second.) At this rate, a DSU demands memory access every fifth word time. For this reason, a DSU should be given the highest priority of any of the peripherals connected into the controller selector. The recommended plug address for a DSU is 0, but it could have any plug address from 0 to 7 provided it had the highest priority (lowest address number) of the particular configuration connected into the controller selector.

Each word recorded on a disc consists of 18 information bits plus 2 zero bits, plus an odd parity bit which is generated by the DSU controller. The minimum amount of information which can be transferred in either direction by one instruction is 64 words, or one frame. The maximum amount of information which can be transferred in either direction is sixteen 64-word records.



Each disc is served by a positioning arm. Each positioning arm contains eight read/write heads; four heads serve the upper surface of the disc and four serve the lower surface. An actuator for each positioning arm can move the arm parallel to the disc, so that all 256 tracks on each surface can be served. The heads are numbered 0-7. Heads 0-3 serve the 128 inner tracks (2 for each side of the disc). Heads 4-7 serve the 128 outer tracks (2 for each side of the disc). Because there are 4 heads for each side of the disc, the actuator must move the positioning arm a maximum of 63 track positions to serve the 256 tracks on a disc surface.

### Disc Storage Unit Command Word Format

The DSU command word format is shown below.

<u>Operation</u>	<u>Octal Code</u>		<u>Disc Unit No. (F)</u>	<u>Bits 15 14 13</u>
PRF	00020P 5F0000 MMMMMM	1st command word 2nd command word 3rd command word	0 1 2 3	001 010 111 100
POSITION		One of the DSU disc units is positioned to receive or transmit a specific record. P is the plug number of the DSU controller on the controller selector. F is the disc unit number on the controller selected by P. M is the actual address (octal) for positioning the disc unit F.		
RRF	00010P 2F00NN 0MMMMM	1st command word 2nd command word 3rd command word		
READ		N is the number (1-16) of 64-word records to be transmitted from disc storage to core storage. F is the number (0-8) of the selected disc unit. M is the core memory address into which the first word of the record is stored. P is the plug number of the DSU controller on the controller selector.		
WRF	00030P 7F00NN 0MMMMM	1st command word 2nd command word 3rd command word		

## Operation

### WRITE

N is the number (1-16) of 64-word records to be transmitted from core storage to disc storage. F is the number (0-3) of the selected disc unit. M is the memory location of the first word to be transmitted from core storage to disc storage. P is the plug number of the DSU controller on the controller selector.

NOTE: The mnemonics are never used in actual coding. The command words must be written in octal form.

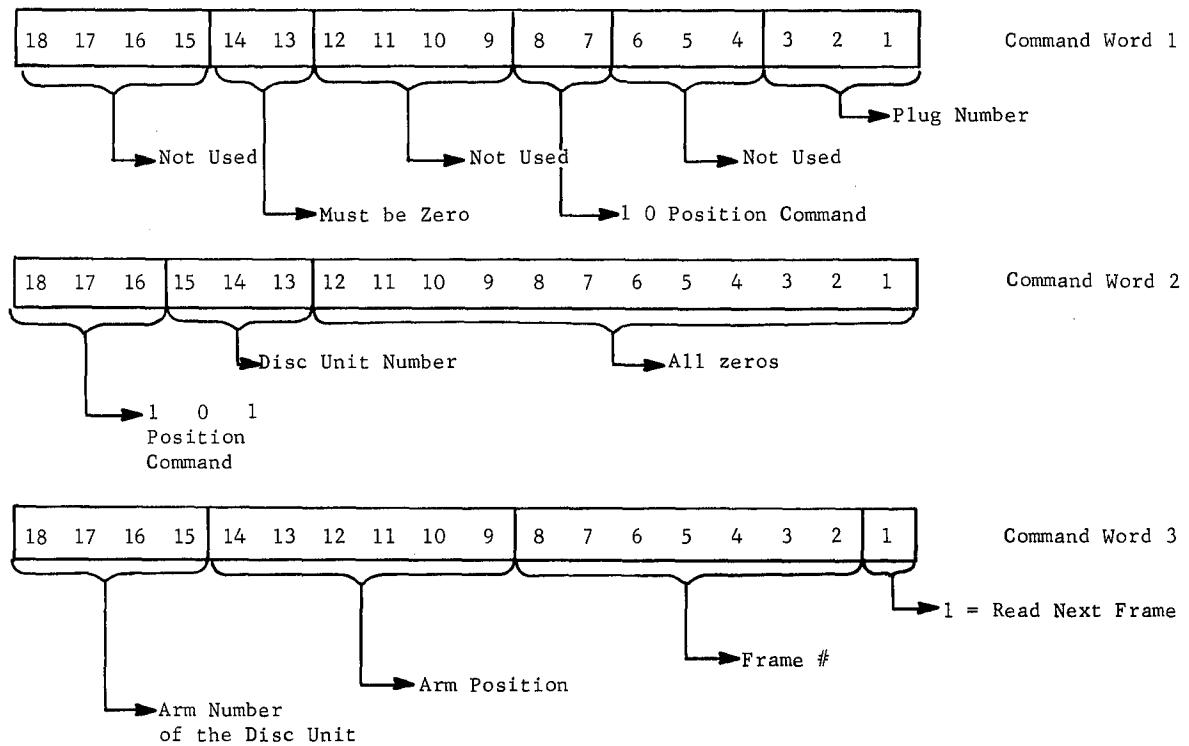
The sequence for addressing the DSU is to select the DSU to be addressed and position one of the arms. The access time varies depending upon the distance the arm must travel to be in position and upon latency time. After the desired arm is in position, the read/write instructions may be executed.

## Position Command Words

When the command words are to be executed, it is first necessary to store the three command words in memory locations 3, 4, and 5. A SEL instruction executes the positioning instructions (command words).

The third word sent from memory to the controller selects the arm, the arm position, and the address of the frame or frames to be transferred. Bit positions 14 and 13 of command word 1 and bit positions 18-15 of command word 2 select the arm (0-15) which contains the head or heads to do the writing or reading. Bits 14-9 of command word 3 select the arm position (0-63) involved in the transfer of information. Bits 8-2 select the first frame (0-95) to be read or written.

Below is shown the command word format for positioning an arm to a desired track and frame.



Selection of the frame to be transferred also automatically selects the head which is to perform the read or write operation. Each of the eight heads on the positioning arm can read a specified number of frames as follows:

<u>Frame Number Per Arm Position</u>	<u>Head Number</u>
0 - 7	0
8 - 15	1
16 - 23	2
24 - 31	3
32 - 47	4
48 - 63	5
64 - 79	6
80 - 95	7

The seven bits of the frame number (command word 3) designate the head which is to perform the read or write operation as well as the number of the frame. All the 96 frames capable of being read when the positioning arm is in a given position can be addressed by the seven frame number bits whose binary value varies from 0000000 for frame 0 to 1011111 for frame 95, as follows:

Command Word 3  
Bits 8 7 6 5 4 3 2

0 0 0 0 0 0 0	Inner Tracks 0-63	Top Side
to		
0 0 0 0 1 1 1	Frames 0 through 7	
0 0 0 1 0 0 0	Inner Tracks 0-63	Top Side
to		
0 0 0 1 1 1 1	Frames 8 through 15	
0 0 1 0 0 0 0	Inner Tracks 0-63	Bottom Side
to		
0 0 1 0 1 1 1	Frames 16 through 23	
0 0 1 1 0 0 0	Inner Tracks 0-63	Bottom Side
to		
0 0 1 1 1 1 1	Frames 24 through 31	
0 1 0 0 0 0 0	Outer Tracks 0-63	Top Side
to		
0 1 0 1 1 1 1	Frames 32 through 47	
0 1 1 0 0 0 0	Outer Tracks 0-63	Top Side
to		
0 1 1 1 1 1 1	Frames 48 through 63	
1 0 0 0 0 0 0	Outer Tracks 0-63	Bottom Side
to		
1 0 0 1 1 1 1	Frames 64 through 79	
1 0 1 0 0 0 0	Outer Tracks 0-63	Bottom Side
to		
1 0 1 1 1 1 1	Frames 80 through 95	

Command Word 3  
Bits 8 7 6 5 4 3 2

1100000 to 96 through 127  
1111111 Invalid Address

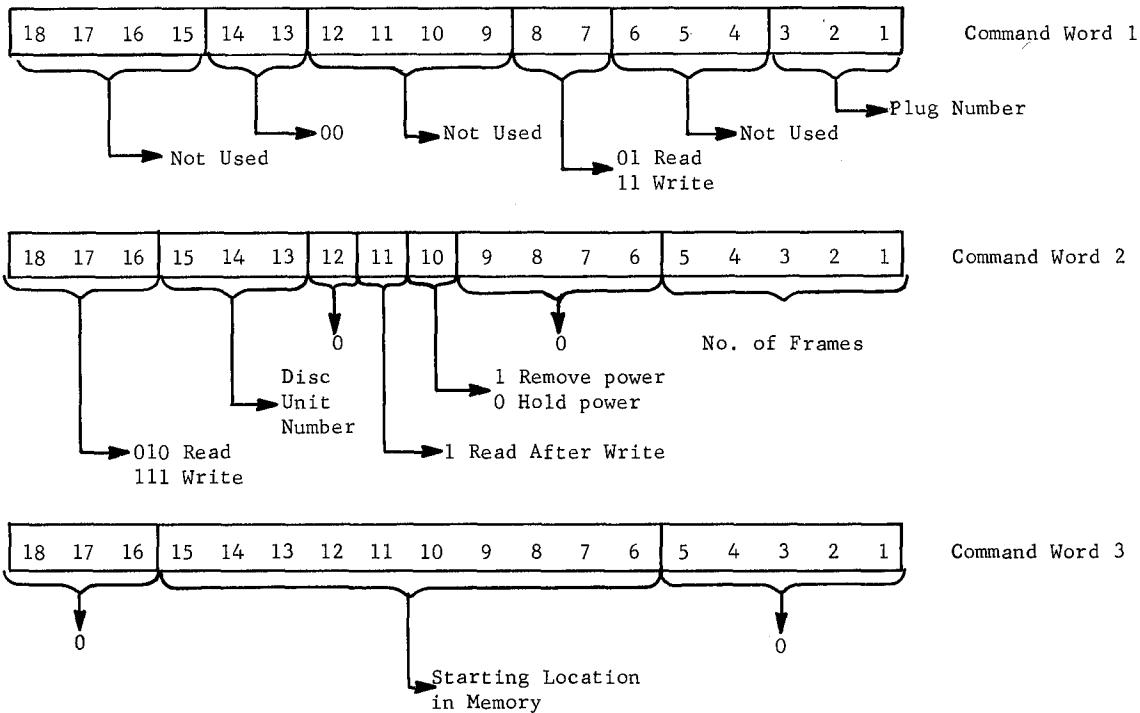
The maximum number of frames which can be transferred by one instruction is 16. It is not necessary that these 16 frames (or any part of 16 frames) all be in the outer tracks or all be in the inner tracks. The transfer of information during the execution of an instruction can start in the inner tracks and continue in the outer tracks. As frames are being transferred, a count is maintained in the DSU controller, so that the read or write operation continues for the specified number of frames. As already explained, the sequential incrementing of the frame address in the controller automatically results in the proper head switching. Because frame 95 is the highest valid address, incrementing the address in the controller beyond 95 causes frame 0 to be the next frame transferred.

Bit 1 of word 3 is identified by read next frame. When this bit is on (contains a 1) the seven bits of the frame address are ignored and the subsequent reading or writing operation takes place in the next frame. Bit 1 of command word 3 is used when it is desired to sample a frame from any given position of the positioning arm. Rather than search for a specific frame out of the 96 possible, the next frame can be read. This form of addressing can also be used when it is known that every frame in a track is to be transferred and it does not make any difference which is read first.

## Read/Write Command Words

After a DSU arm has been positioned the DSU can then be addressed for a read or write operation. It is first necessary to store the three command words for the read/write operation in memory locations 3, 4, and 5. A SEL instruction executes the read or write.

The command word format for read or write operations is as follows:



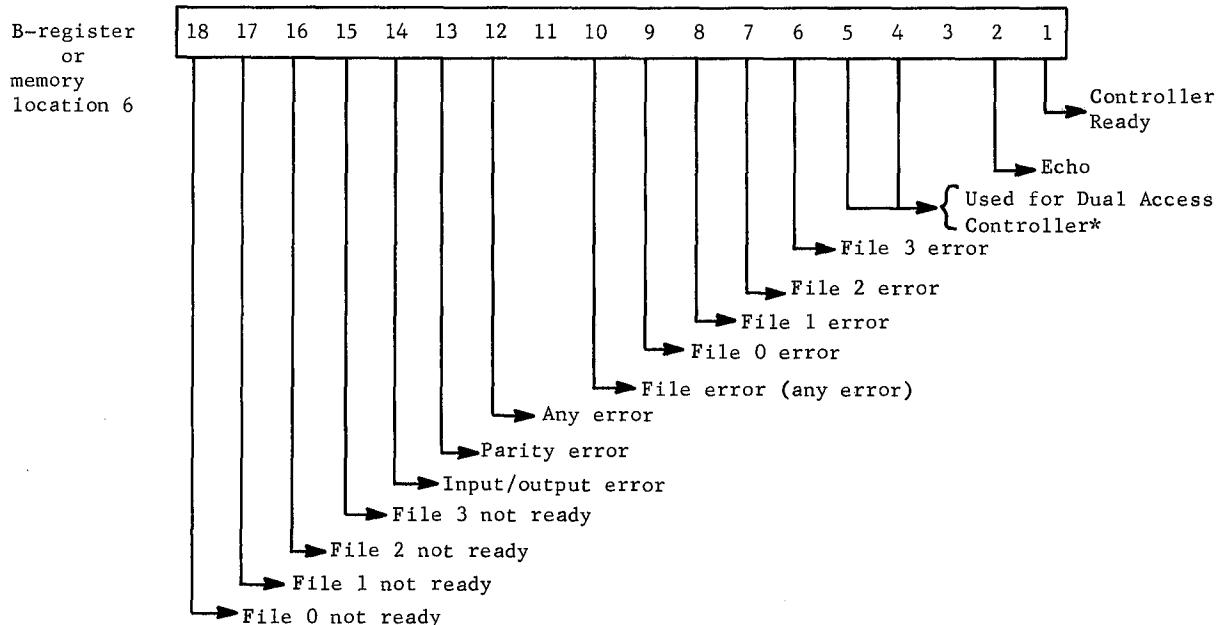
Command word 1 selects the controller selector plug (P) into which the DSU is connected. Once the DSU unit has been selected by the controller selector, the DSU controller goes into the busy state and waits for the next two words from memory. The next two words are sent to the DSU controller and indicate the operation to be performed (read or write), the file which is to perform the operation, the number of frames to be transferred, and the starting address in memory where information is to be sent or retrieved.

Bit positions 8 and 7 of word 1, and 18-16 of word 2 cause the file to read (octal 12) or write (octal 37). Bits 15-13 of word 2 indicate the file which is to perform the read or write operation. Bits 5-1 of word 2 indicate the number of records (0-16) which can be transferred. A "1" in bit position 10 of command word 2 of a read or write sequence will cause power to be removed from the positioning motor upon completion of that sequence. A "0" in bit position 10 of command word 2 holds power to the positioning motor. A "1" in bit position 11 initiates the read-after-write parity check function of the DSU.

Bits 15-7 of word 3 transferred to the controller indicate the starting location in memory of the read or write operation. The nine bits of the starting location address allow this address in the controller to be stepped 1024 times or, in other words, to count the 1024 words of 16 records, the maximum which can be transferred by one instruction. Because bits 6-1 of word 3 are not used, the starting location address must be a multiple of 64. Bits 15-7 can address memory capacities up to 32,767 words.

### Branch Conditions

Single-access DSU branch conditions may be tested by examining the bits, as indicated below, after a CSR command. When the particular bit is on, the condition is true.



\*Dual Access Controller information is covered in Appendix I of CPB-1019, DATANET-30 Programming Reference Manual.

When executing a CSR command, the CSU-931 will quiz the controller whose address corresponds to bits 1, 2 and 3 of the CRS instruction, and place the bits received from the controller into memory location 6 in the format shown on the previous page.

After the CSR, and until memory location 6 has been loaded, NIS 6 will result in a zero state. When NIS 6 results in nonzero, the program may examine the contents of memory location 6 to determine controller status.

### Sample Coding to Address a DSU

Below is given a sample coding for positioning the arm:

```
NIS    7
BZE   *-1
CSR    0
NIS    6
BZE   *-1
LDB    6
BEV   *-4
LDB    1st Word (Command)
STB    3
LDD    2nd and 3rd Word (Command)
STD    4
SEL    (Arm starts seek for Position. DSU goes ready when in position. Now issue Read or
      Write.)
```

Below is given a sample coding for a read or write operation:

```
NIS    7
BZE   *-1
CSR    0
NIS    6
BZE   *-1
LDB    6
BEV   *-4
LDB    1st Word (Command) Read/Write
STB    3
LDD    2nd and 3rd Word (Command) Read/Write
STD    4
SEL    (To Execute Read/Write.)
```

## Diagnostic Program Areas

The following disc sectors are reserved for diagnostic programming functions and should not be used for other purposes.

Disc Unit Configuration		File Addresses Reserved		
		Disc	Position	Record
Drum Mode Module		15	00	95
Data Model 4	{	0	00	0
		3	63	80-95
Data Model 8	{	0	00	0
		7	63	80-95
Data Model 12	{	0	00	0
		11	63	80-95
Data Model 16	{	0	00	0
		15	63	80-95

## MAGNETIC TAPE UNITS

This section contains only information special to programming the magnetic tapes from the DATANET-30. More detailed information on magnetic tape and additional programming information may be found in Compatibles/200 Magnetic Tape manual and other related publications.

Magnetic tape units can be operated in two different modes: decimal and special binary. During forward movement of the tape, information can be written on or read from tape in both modes. During backward movement, information can be read from tape in both modes.

During the decimal mode of operation the zone bits--the two most significant bits of each 6 bit binary-coded decimal (BCD) character--are altered during transfer of information between magnetic tape and memory. This alteration of the zone bits takes place automatically in the tape controller as follows:

<u>BCD Character in Memory</u>	<u>BCD Character on Tape</u>
00xxxx	00xxxx
01xxxx	11xxxx
10xxxx	10xxxx
11xxxx	01xxxx

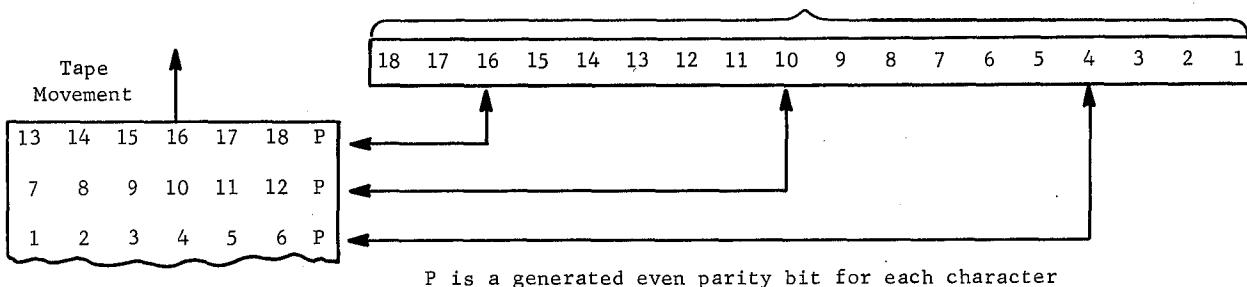
The four least significant bits (xxxx) of each BCD character are the same in memory or on tape with one exception: a BCD 0 in memory is 000000 but on magnetic tape it is 001010. The alteration of information during the decimal mode takes place for any configuration of bits (there are no illegal bit configurations). The alteration of information during the decimal mode of operation makes the DATANET-30 magnetic tapes compatible with magnetic tape formats now in use.

During binary operations of magnetic tapes, information is transferred between magnetic tapes and memory without alteration of bits.

## Decimal Mode

In the decimal mode of magnetic tape operations, 18 bits (18-1) of a memory word correspond to 3 BCD characters. Each word from memory is checked for parity in the tape controller. When information is read from tape, bits 0 and 1 are made 0 when three BCD characters enter a memory cell. The following illustration shows the relationship between a word in memory and the three BCD characters on magnetic tape.

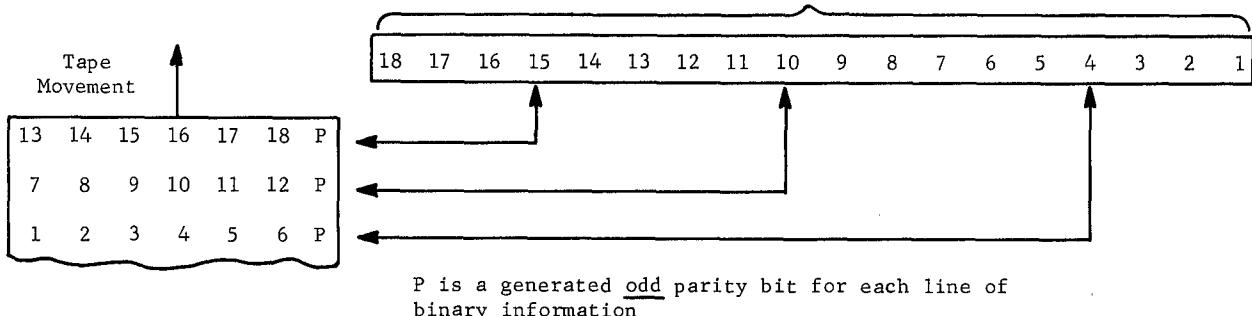
3 - Six Bit BCD Characters in Memory



## Binary Mode

During binary mode operations, 18 bits (1-18) are written on tape as three lines of information.

18 - Bit Binary Word in Memory



The format of information on tape in the binary mode is the same as in the decimal mode. In the binary mode, however, the zone bits and 0 are not altered during the transfer of information. Also, in the binary mode, the parity bit P generated for each line on tape is an odd parity bit.

## Record Length

After reading (in binary or decimal mode) N words from magnetic tape into memory starting at location M, memory location M + N will contain zeros if exactly N words were read from a record on tape containing N words. If the number of words contained in the record currently read is less than N, then only the contents of the record will be stored in memory and the 2's complement of the difference (N - record length) will be stored in memory cell M + N with a 1-bit in position 18. If the number of words in the record is greater than N, then only N words will be stored in memory and the increment (record length - N) will be stored in memory cell M + N with a 0 in the sign position. M is not automatically modified. In order to forward space (skip) one record, the RTS,

RTD, or RTB command is used with N set equal to 0. This statement also applies to the read tape backward instructions except that M - N will contain zeros if exactly N words were read from a record on tape containing N words. M - N will contain the 2's complement of the difference (N - record length) with a 1 in position 18 if the number of words contained in the record currently read is less than N. M - N will contain the increment (record length - N) if the number of words in the record is greater than N.

### Magnetic Tape Instructions

#### Operation

WTD  
WRITE TAPE DECIMAL.

#### Octal Code

0T000P - CW1  
2MMMMM - CW2  
TNNNNN - CW3

N decimal words from memory starting at location M are written on unit T. P is the plug number of the tape controller.

RTD  
READ TAPE DECIMAL.

0T000P  
4MMMMM  
TNNNNN

A maximum of N decimal words is read by tape unit T and placed in memory starting at location M.

WTB  
WRITE TAPE BINARY.

0T020P  
3MMMMM  
TNNNNN

N words of information from memory starting at location M are written by tape unit T. Bits 18-1 are written on tape exactly as in memory.

RTB  
READ TAPE BINARY.

0T020P  
5MMMMM  
TNNNNN

A maximum of N words is read by tape unit T and stored in memory starting at location M.

RBD  
READ BACKWARD DECIMAL.

0T010P  
4MMMMM  
TNNNNN

Decimal information is read from tape moving backwards. A maximum of N words is read into memory, the first word being placed in location M. The second word is placed in M - 1 and so on until N words are read. The tape controller alters the zone bits of characters read so that they conform to G-E Compatibles/200 internal BCD characters.

Operation

RBB  
READ BACKWARD BINARY.

Octal Code

0T030P  
5MMMMM  
TNNNNN

Information is read from tape moving backwards. Contents of bit positions 2-19 of each word read are placed in memory exactly as on tape (zone bits are not altered). A maximum of N words is read into memory, the first word being placed in M. The second word read is placed in M - 1 and so forth until N words are read.

RWD  
REWIND.

0T020P  
000000  
T00000

Rewind tape unit T to leader.

WEF  
WRITE END-OF-FILE.

0T000P  
200000  
T00000

The end-of-file character (0001111) and end-of-file gap are written on tape by tape unit T.

BKW  
BACKSPACE AND POSITION WRITE HEAD.

0T010P  
600000  
T00000

The tape on tape unit T is backspaced one record and the write head is positioned to write.

**Command Words**

The table below shows the digits used for specifying each tape unit:

Tape Unit								
Command Word	0	1	2	3	4	5	6	7
1st Word	0 <u>00</u> Y <sub>1</sub> OP	0 <u>00</u> Y <sub>1</sub> OP	0 <u>00</u> Y <sub>1</sub> OP	0 <u>1</u> Y <sub>1</sub> OP	0 <u>2</u> Y <sub>1</sub> OP	0 <u>2</u> Y <sub>1</sub> OP	0 <u>2</u> Y <sub>1</sub> OP	0 <u>3</u> Y <sub>1</sub> OP
2nd Word	Y <sub>2</sub> MMMMM	Y <sub>2</sub> MMMMM	Y <sub>2</sub> MMMMM	Y <sub>2</sub> MMMMM	Y <sub>2</sub> MMMMM	Y <sub>2</sub> MMMMM	Y <sub>2</sub> MMMMM	Y <sub>2</sub> MMMMM
3rd Word	<u>1</u> NNNNN	<u>2</u> NNNNN	<u>4</u> NNNNN	<u>0</u> NNNNN	<u>1</u> NNNNN	<u>2</u> NNNNN	<u>4</u> NNNNN	<u>0</u> NNNNN

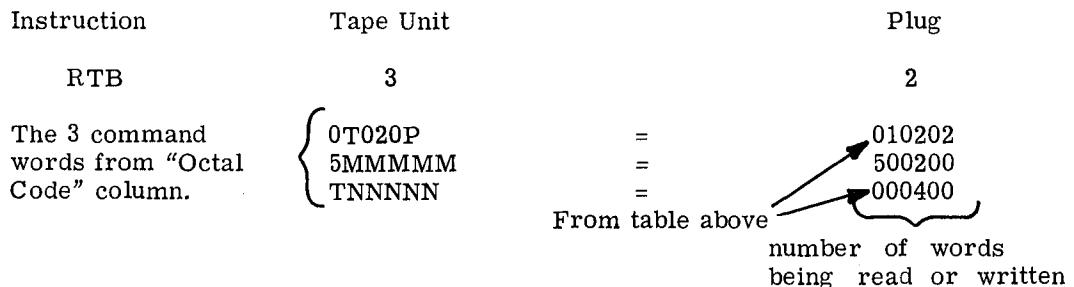
Y<sub>1</sub> Y<sub>2</sub> are the octal digits for the different tape instructions. The numbers underlined are used for specifying the unit number.

P is the plug number of the tape controller on the controller selector.

M is the address being written out of or read into memory.

N is the number of words being read or written - that is, record length.

For Example:



Inst.	Y <sub>1</sub>	Y <sub>2</sub>
WTD	0	2
WTB	2	3
RTD	0	4
RTB	2	5
RBD	1	4
RBB	3	5
RWD	2	0
WEF	0	2
BKW	1	6

Unit Number	T for Command Word 1	T for Command Word 3
0	0	1
1	0	2
2	0	4
3	1	0
4	2	1
5	2	2
6	2	4
7	3	0

The above tables can be used when setting up the three command words. Before issuing the command words to the tape controller, the command words are first transferred to memory locations 3, 4, and 5. A SEL instruction executes the transfer of the command words to the magnetic tape controller. When the command word instructions are coded, the above octal coding is used as the operand.

### Programming Example

The following example shows how to write 64-word records on magnetic tape unit 2 out of location 500:

<u>Symbol</u>	<u>OPR</u>	<u>Operand</u>	<u>Remarks</u>
WTB	NIS	7	SELECT DONE?
	BZE	*-1	NO WAIT
WTB1	CSR	2	GET STATUS
	NIS	6	IS STATUS IN MEMORY
	BZE	*-1	NO, WAIT
	LDB	6	PUT STATUS IN B
	BEV	*-4	NOT READY, GO BACK
	LDB	1STWD	GET COMMAND WORD 1
	STB	3	STORE IN LOCATION 3
	LDD	WD2,3	GET WORDS 2 AND 3
	STD	4	STORE IN LOCATIONS 4 AND 5
	SEL		SELECT PERIPHERAL
	NIS	7	SELECT DONE?

<u>Symbol</u>	<u>OPR</u>	<u>Operand</u>	<u>Remarks</u>
	BZE	*-1	NO GO, WAIT
	BRU	WTB1	
READY	OCT	000001	BIT 1 TO TEST READY
1STWD	OCT	000201	COMMAND WORD 1
WD2,3	OCT	300500	COMMAND WORD 2
	OCT	200100	COMMAND WORD 3

In the preceding example, initially the CSR command is executed to test the ready status of the tape controller. When the controller becomes ready, the 3 command words are loaded from their temporary storage locations and put into locations 3, 4, and 5. The SEL command initiates operation of the controller selector unit and the commands are automatically sent to the tape controller. Next, the NIS 7 interrogates the controller selector to see if the last controller select is finished. When the select has been finished the program returns to write a new record.

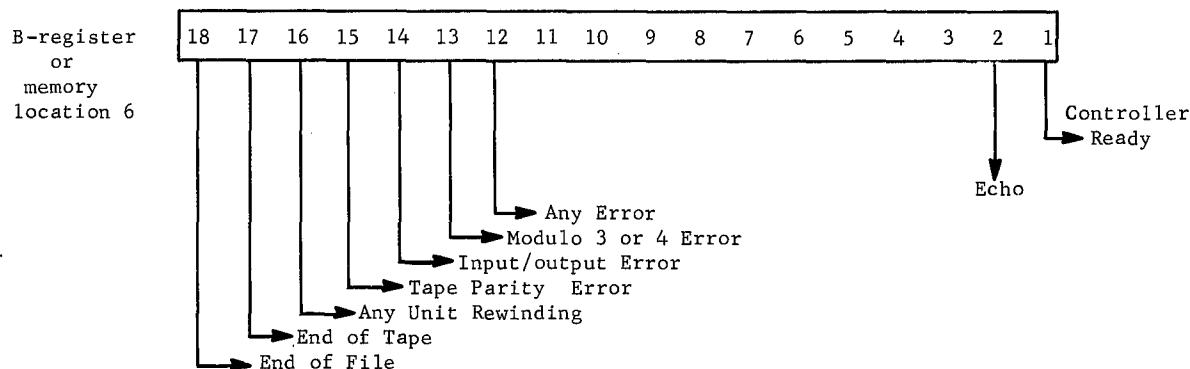
### Tape Unit Conditions

Tapes contain a silver spot to signal the physical end of the tape. When detected by a photoelectric cell within the tape unit, an indicator on the tape controller is set. The condition of the indicator should be tested by programmed instructions after reading or writing each record. If the indicator is not set, normal processing will continue. If it is set, an end-of-tape branch will jump into specified subroutines--normally rewinding the current reel and switching to a new reel. The end of file sentinel is the magnetic representation of the binary code 001111 preceded by an erased section of the tape 3-3/4 inches long.

During magnetic tape operations several other exceptional conditions may occur which are secondary to the main processing job. Handling of these exceptional conditions may be conveniently assigned to "executive routines." These conditions are handled as branch conditions.

### Branch Conditions

The branch conditions concerned with the tape controller may be tested by examining the bits after a CSR instruction. When the particular bit is on, the condition is true, as shown below:



Examples:

To Get Status (Controller on plug 4)

CSR	4	Request Status
NIS	6	Is Status in Memory
BZE	*-1	No, Wait
LDB	6	Put Status in B

When status is in the B register, the following checks can be made. The LDB 6 instruction is repeated for illustration only.

Controller Ready

LDB      6  
BEV

(not ready)

Echo

LDB      6  
SR1  
BEV

B,Z

No error

Any Error

LDB      6  
NBZ      004L  
BZE

No error

Tape Parity Error

LDB      6  
NBZ      04L  
BZE

No error

004L OCT      004000

04L      OCT      040000

Mod 3 or 4 Error

LDB      6  
NBZ      01L  
BZE

No error

Any Unit Rewinding

LDB      6  
NBZ      1L  
BZE

No error

01L OCT      010000

1L      OCT      100000

## HIGH SPEED PRINTER

The printer equipment consists of the printer mechanism and a printer controller. Information to be printed on one line is transferred from the DATANET-30 memory to the printer controller which stores this information in a "buffer." After this transfer is complete, printing commences completely "off-line" from central processor operations, and no further interruption of processor operation is necessary. After the line has been printed, the paper is slewed (spaced or advanced) for one or more lines, to provide vertical format arrangement and to permit additional editing functions to be performed by the DATANET-30 while the line is being printed. In one minute, 900 lines of alphanumeric information, single space only, or 600 lines single or double space can be printed. Speeds may be selected by an operator switch.

## Programming the High-Speed Printer

This section contains information concerning programming the printer from the DATANET-30. More detailed information on printer operation may be found in GE-200 Series High-Speed On-Line Printer Reference Manual, CPB-321 and other publications.

The printer will print the equivalent of 10 numeric, 26 alphabetic and 14 special characters when information is correctly represented in binary coded decimal form. A single print command will cause printing of up to 120 characters on one line from a block of 40 words of BCD data in memory. When the print command is given, the printer receives information directly from the main memory through the controller selector.

Printer operations may be programmed to overlap or share time with other processing. The printer can be interrogated for "ready" status (completion of previous print commands) by the CSR command. The program may transfer control directly back to the CSR command in a loop for repeated interrogation until the printer is ready, or the program may branch to another part of the program to execute other operations and return later.

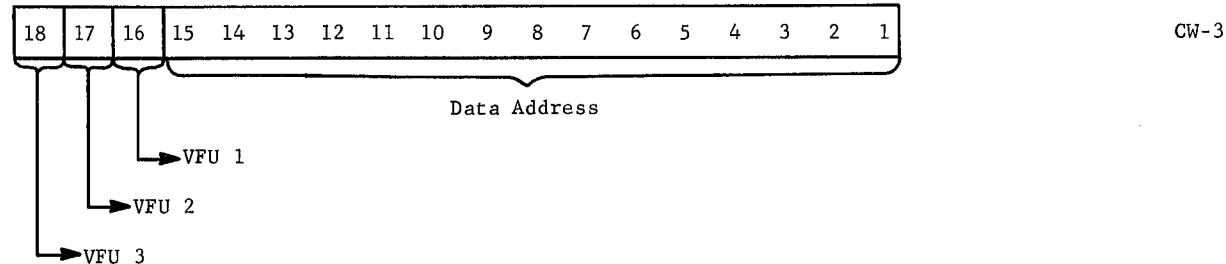
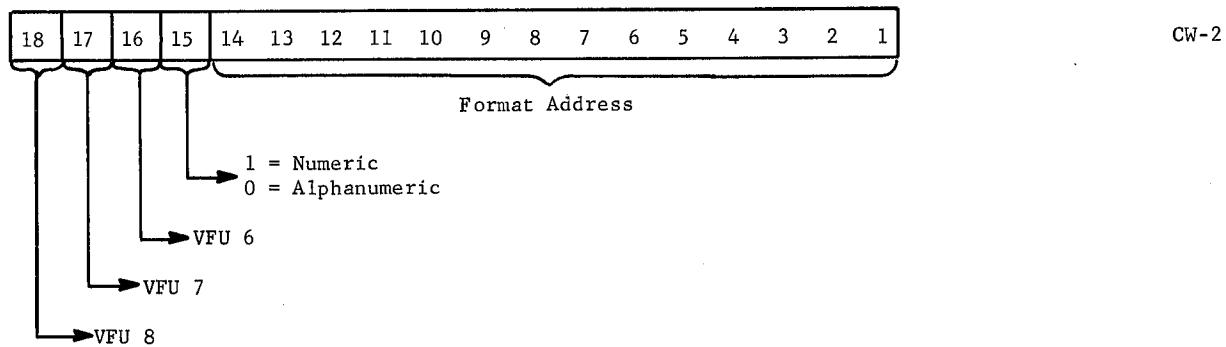
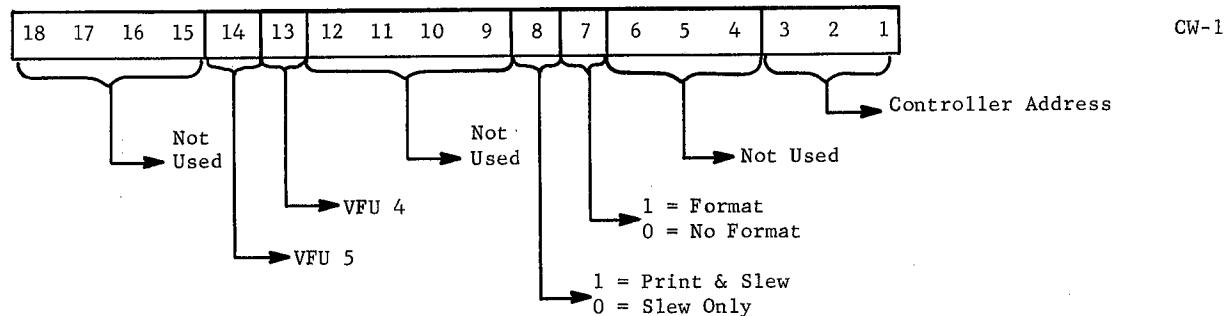
If a number previously calculated within the computer is to be printed, the binary number must first be converted to the binary coded decimal equivalent using an appropriate conversion subroutine to convert the high-speed printer character set. If the number is a negative number, the sign of the number will be reflected by the insertion of a binary coded decimal hyphen in the high-order position (or low-order position at the discretion of the programmer) of the number field, so that the number will print as -xxxxx or xxxx-. An alternate symbol might be chosen as the symbol to represent a negative number in lieu of the hyphen. Alphabetic and alphanumeric data could be carried in BCD form throughout processing to eliminate the need for conversions.

The following three methods for editing data to be printed may be used in any combination:

1. The initial establishment of a record to conform exactly to the print line format desired.
2. The rearrangement of the record in memory by programmed insertion or deletion of characters.
3. The automatic editing performed by logic circuits in the printer controller, based on a combination of related data and format words transferred to the printer controller.

## Command Words

An instruction for the high-speed printer consists of three command words. The bit configurations of the high-speed printer instructions are as follows:



## Printer Instructions

### Operation

WPL

### Octal Code

00020P -CW1  
S00000 -CW2  
1YYYYYY-CW3

Print 1 line then slew 1 line.

### Write Print Line

One line of BCD information, 1 to 120 characters long, is printed. Y is the starting location in memory

of the information to be printed. P is the plug number of the printer controller. By convention, P is either 6 or 7. S designates to slew by countdown (6) or slew to top of page (4). The 1 in CW3 indicates slew 1 line.

Operation	Octal Code
-----------	------------

WFL	00030P SYYYYY 1XXXXX
-----	----------------------------

Print 1 line and slew 1 line.

Write Format Line.	One line of BCD information is printed under format control. Y is the starting location in memory of the format control words. X is the starting location of information to be printed. S designates slew by countdown (6) or slew to top of page (4). The 1 in CW3 indicates slew 1 line.
--------------------	--

### Slewing Paper

The paper can be slewed by two methods: (a) the countdown, and (b) the slew to a channel.

The countdown method allows slewing a fixed number of lines. Bit positions for VFU channels 1-6 in the printer instruction words contain the binary count for the number of lines of paper to be slewed. Bit positions for VFU channels 7 and 8 must contain 1 bits for countdown slew.

Slewing to a channel punch is accomplished by means of an eight-channel paper tape loop. The paper tape loop controls vertical spacing on printed forms by slewing to a punch in a specified channel in the paper tape loop. If only one of the 8 VFU bits is set, paper will be slewed until a hole is detected in the VFU tape channel designated by that bit. For example, if VFU 3 is set to a 1 bit, paper will slew until a hole is detected in the VFU tape channel 3. VFU channel 8 is normally used for slewing the paper to the top of page.

Operation	Octal Code
SLW	ONOX0P 600000 N00000

The printer paper is slewed N(0-63) lines. The N specified in the first word is the most significant 2 bits of 5-bit number of lines to slew. If more than 31 lines are to be slewed, X in the first word is used for the added bits.

SLT	0X000P X00000 X00000
-----	----------------------------

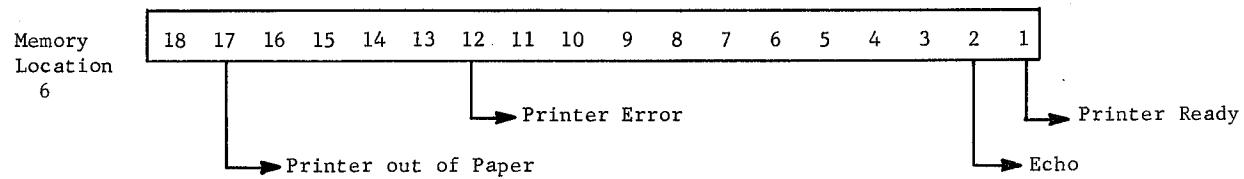
Slew Paper to Tape Punch.

The printer paper is spaced until a hole is detected in the vertical formattape. X varies with the channel specified.

NOTE: The mnemonics are never used in actual coding. The command words must be written in octal form.

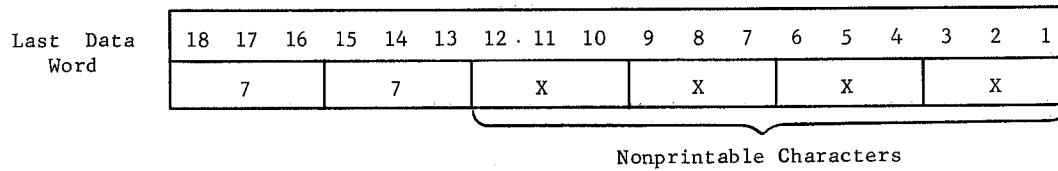
### Branch Conditions

The branch conditions associated with the printer may be tested by examining the bits as shown below after a CSR instruction.



### Print Line Termination

The line printed on the high-speed printer may be from 1 to 120 characters in length. If a full line is not printed (120th character) the line is terminated by making the most significant 6 bits of the word following the last data word all 1's, the least significant 12 bits must be two nonprintable characters.



## CONTENTS

	Page
THE CHARACTER/WORD BUFFER UNIT (CWU-931)	
General Description .....	M-1
Character Buffer Channel (CBC-931) .....	M-1
Character Buffer Instructions .....	M-2
Transfer Rate .....	M-3
Programming Considerations .....	M-3
Receive Operation .....	M-3
Transmit Operation .....	M-5

## ILLUSTRATIONS

Figure	Page
M-1 Character Buffer Channel Block Diagram .....	M-2
M-2 CBC Data Transfer .....	M-4

## EXAMPLES

A Transmit .....	M-6
B Receive .....	M-6



## APPENDIX M

### THE CHARACTER/WORD BUFFER UNIT (CWU-931)

#### GENERAL DESCRIPTION

The function of a character buffer is to transmit data to and receive data from a UNIVAC 1004 remote terminal, or similar equipment, on a character basis. Transmission to and from the terminal is on a bit-serial synchronous basis.

The character/word buffer unit module can contain two character buffer channels (CBC). Each CWU occupies one module space.

#### CHARACTER BUFFER CHANNEL (CBC-931)

One buffer channel provides the interface between the DATANET-30 and a half-duplex transmission line. The channel is synchronized with the digital subset connected to the transmission line.

The buffer control unit contains hardware to control the character length. Buffers in a module may be operating at different character lengths. The buffer channel operates with a character oriented device at speeds determined by the subset and remote terminal. A synchronous digital subset must be on each end of the transmission line. The standard bit rates are 2000 and 2400 bits per second.

The code level may be from 5- to 16-bit codes with character synchronization (no start/stop bits). The code level (character length) is selected or changed by a code level connector for each CBC. By changing connectors, the code level may be varied to meet different remote terminal operations or programming techniques.

The connector defines the bit configuration of the synchronizing character, the number of bits per character, and where the receive lines will enter the data bits into the "working" register (A or B register); that is, the high- or low-order position of the A or B register. The code level connector can be arranged to accept two characters (8 level) before setting the receive flag. Also, the connector can be arranged to mask off a bit.

The line interface is 201A or 201B Data Set.

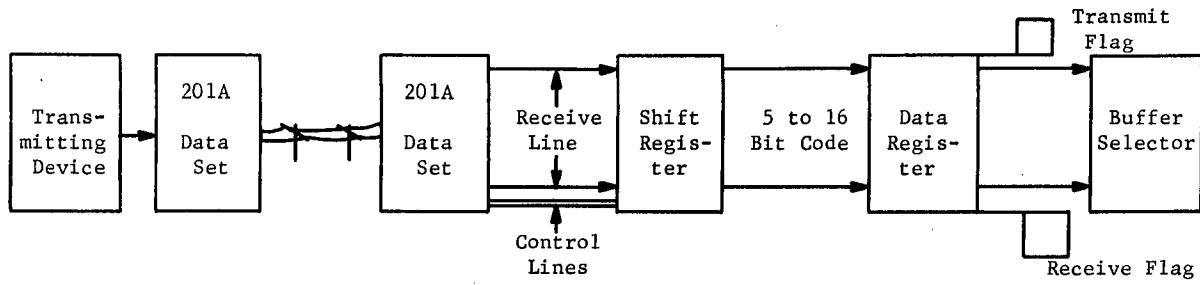


Figure M-1. Character Buffer Channel Block Diagram

### Character Buffer Instructions

<u>Mnemonic</u>	<u>Operand</u>
-----------------	----------------

Register Transfer      TRA      ,T      (TRA from \_\_\_\_\_ to T)

The least significant bits of the register, as specified by the size of the character buffer, are sent to the transmit data buffer and the transmit flag is reset.

Register Transfer      TRA      R,      (TRA from R to \_\_\_\_\_ )

The bits, as specified by the size of the character buffer, are transferred from R. The receive data buffer and flag are reset (DEF 1).

### DEF Instructions

DEF 1	Reset receive flag
DEF 2	Reset transmit flag
DEF 3	Set receive mode
DEF 4	Set transmit mode
DEF 5	Reset buffer
DEF 6-8	Not used
DEF 9	Answer incoming call
DEF 0	Disconnect call

### NES Instructions

NES 1	Receive flag set
NES 2	Transmit flag set
NES 3	Call in progress
NES 4	Request answer call
NES 5	Data mode (interlock on)
NES 6	Carrier on
NES 7	Clear to send
NES 8-10	Not used

LDT - Do not use

SCN - Do not use

### Transfer Rate

The transfer rate varies with the data set used. The 201A Data Set allows 2000 bits per second and the 201B Data Set allows a transfer rate of 2400 bits per second over a private line. If an eight-level code is used with 201A Data Set for example, the rate is 250 characters per second. Since the decode plug can be wired to accept 5-15 bits, two 8-bit characters can be received in the shift register before being transferred to the data register. Thus, the receive flag is set once per every two characters. Although the transmission rate remains the same, the rate of transfer to or from the character buffer becomes 125 transfers per second. Therefore, the period for program scan must be set to occur as required for the character length and data set used.

## PROGRAMMING CONSIDERATIONS

The SCN instruction cannot be used with the character buffer. The program must check the receive (transmit) flag often enough to maintain a continuous transfer of characters.

Character recognition by hardware is used to identify characters being received for the synchronizing character. The synchronizing character to be recognized is set up at the decode plug and can be changed by the decode plug wiring. The synchronizing character is recognized before the hardware sets the receive flag. The Start of Message (SOM) character must be recognized by the program. The End-of-Message (EOM) character is not recognized by hardware. The program must recognize the EOM character or otherwise determine a full message as having been received and stops receiving characters. Once started, the character buffer will continue transfers to the data register and set the receive flag until it is stopped by the program with a DEF 5 or until the mode is changed to transmit.

A subroutine for an acknowledgement from the remote terminal should be used after an EOM character has been transmitted to the terminal. Conversely, an acknowledgement must be transmitted to the terminal after an EOM has been received.

### Receive Operation

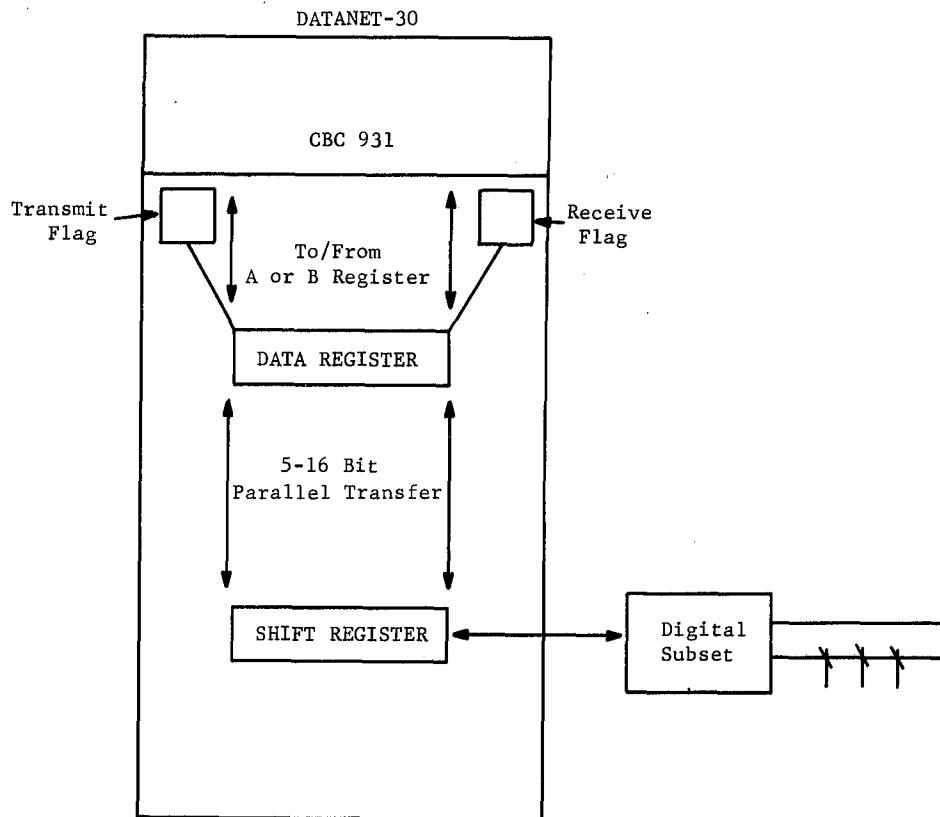
The CBC will accept a serial transmission of up to 16 bits. The code level can be reduced to fit the code in use when fewer than 16 bits are used. Characters may be transferred to the least significant bit positions of the DATANET-30 word.

The CBC is a double buffer device. The character received will exist in the data buffer for one character time while the next character is being received. As the remote terminal transmits each succeeding character, the new character appears in the shift register. A character must be shifted out of the data register before the next one is completely received or the character in there will be lost.

Assume that nothing is being received and that the CBC is in a receive mode. When the sync character is recognized, the hardware will consider the next bit to be the first bit of the first character of the message. When a full character is received, it is sent to the data buffer and a flag is set. The program has been periodically interrogating the state of the flag. When the flag is set, and when the program detects this condition, the program transfers the character out of the CBC data buffer and the flag is reset. The CBC will continue receiving characters until reset by the program.

The CBC can utilize automatic answering and answer-back features of the digital subset. Both hardware automatic answering and program answering can be done. Hardware answering is done by the digital subset. When the subset answers, the external status line (NES-3) call-in-progress signal will be set until the call is terminated. The program can terminate a call with a Drive External Function (DEF-0)--Disconnect Call instruction.

When the program does the answering, upon the receipt of an incoming call, an External Status Line (NES 4) signal--request to answer call--is set, but the call is not answered until the program does so. The program answers with a Drive External Function (DEF 9) instruction--Answer Incoming Call. The call is terminated with the DEF instruction--Disconnect Call.



M-2. CBC Data Transfer

## Transmit Operation

Assuming that the buffer is in the receive mode, it must be set to the transmit mode with a DEF 4 instruction. DEF 4 is a Request to Send signal to the data set. The data set will return a Clear to Send (NES 7) signal which will set the transmit flag. The character to be transmitted may then be transferred to the data register, which resets the transmit flag. The character is then transferred to the shift register and the transmit flag (NES 2) is set. The program must put another character into the data register before the last bit of the preceding character is transmitted, or the shift register will transmit the same character until a new character is transferred to the data register. After the last character has been transferred to the shift register a delay of at least one character time must be allowed to permit transmission of the last character before a DEF 5 (reset buffer) instruction.

The CBC should be returned to the receive mode upon completion of a transmission. When changing from the transmit to the receive mode, the DEF 3 and DEF 5 instructions must be given as two separate instructions in the order of DEF 3 followed by DEF 5. Completion of transmission includes waiting 2 character times plus 1 millisecond after the last character is transferred to the data register to allow the characters to be transmitted from the buffer and the data set.

TRANSMIT EXAMPLE. The transmit example (Example A on Page M-6) shows one way characters may be transmitted. This is not necessarily the way it would be done in an operating program. This example assumes that:

1. The DATANET-30 can wait for the digital subset to perform various functions;
2. The synchronizing character and other special characters have been defined, such as, first and second End of Message characters and Start of Message characters;
3. Subroutines will provide for functions not programmed here;
4. Connection has been made to the remote terminal and the CBC is in the receive mode.

RECEIVE EXAMPLE. The receive example (Example B on Page M-6) shows how characters may be received. This is not necessarily the way it would be done in an operating program. This example assumes that:

1. The DATANET-30 can wait for the digital subset to perform various functions;
2. An area has been reserved for incoming data;
3. Subroutines will provide for functions not programmed here;
4. Program answering is established;
5. Various special characters have been defined;
6. The CBC is in the receive mode.

The subroutine for EOM could include sending an acknowledgement to the remote terminal.

#### Example A. Transmit

OPERATION SYMBOL	OPCODE	OPNAME	REMARK OR CONTINUATION OF OPNAME
PIC	CBCADD		
TRANS	DEF	2,4,5	SET TRANSMIT MODE
	LDB	SOM1	FIRST SYNC CHARACTER
	LDA	CONST	NUMBER OF SYNC CHARACTERS SENT
TRANS1	NES	2	IS TRANSMIT FLAG SET
	BZE	*,-1	NO WAIT
	TRA	B,T	SEND SYNCHRONIZING CHARACTER
	SR1	A,A	SHIFT COUNTER
	BNZ	TRANS1	SEND NEXT SYNC CHARACTER
	LDB	SOM2	GET START OF MESSAGE CODE
	NES	2	IS TRANSMIT FLAG SET
	BZE	*,-1	NO WAIT
	TRA	B,T	SEND START OF MESSAGE CODE
TRANS2	LDB	\$TABOUT	LOAD B WITH MESSAGE CHARACTER
	ADO	\$TABOUT	ADD ONE TO SEND POINTER
	NES	2	IS TRANSMIT FLAG SET
	BZE	*,-1	NO WAIT
	TRA	B,T	SEND CHARACTER
	ADD	CHRCNT	ADD ONE TO COUNTER
	BZE	TRA.EOM	SEND END OF MESSAGE CODE
	BRU	TRANS2	SEND NEXT CHARACTER
TRA.EOM			A subroutine to read End of Message characters as well as at least 1 dummy character

#### Example B. Receive

HSC930  
HIGH-SPEED CHANNEL  
FOR DATANET-30

REFERENCE MANUAL

Appendix N to DATANET-30  
Programming Reference Manual  
(CPB-1019)

March 1967

GENERAL  ELECTRIC

INFORMATION SYSTEMS DIVISION

## PREFACE

Appendix N to the DATANET-30 Programming Reference Manual (CPB-1019) describes the operation and programming of the HSC930 High-Speed Channel for the DATANET-30 communications processor.

Suggestions and criticisms relative to form, content, purpose, or use of this manual are invited. Comments may be sent on the Document Review Sheet in the back of this manual or may be addressed directly to Document Standards and Publications, B-84, Computer Equipment Department, General Electric Company, 13430 North Black Canyon Highway, Phoenix, Arizona 85029.

(c) 1967 by General Electric Company  
(1.5M 4-67)

DATANET-30

## CONTENTS

1. HIGH-SPEED SYNCHRONOUS CHANNEL UNIT (HSC930)	N-1
General Description .....	N-1
Data Transfer .....	N-2
Checking Features .....	N-3
Character Parity Checking .....	N-4
Block Parity .....	N-4
Character Parity Generation .....	N-4
Block Check Character Generation .....	N-5
Address Overflow .....	N-5
Block Diagram .....	N-5
Address Counter .....	N-5
Buffer Register .....	N-6
Universal Character Recognition .....	N-7
Shift Register .....	N-7
Instruction Repertoire .....	N-7
Drive External Function Instructions .....	N-7
External Status Lines Instructions .....	N-9
Receive Data Lines Instructions .....	N-10
Controls and Indicators .....	N-10
Address Selection Switches .....	N-10
Mode Selection Switches .....	N-10
Sample Program .....	N-11

## ILLUSTRATIONS

N-1 Half-Duplex High-Speed Channel Unit with Telephone Company Data Sets and an Interrupt Scanner Unit .....	N-1
N-2 Full-Duplex High-Speed Channel Unit with Telephone Company Data Sets and an Interrupt Scanner Unit .....	N-2
N-3 Internal Code Format (Receive Mode) .....	N-3
N-4 Internal Code Format (Transmit Mode) .....	N-4
N-5 Block Diagram .....	N-6
N-6 Sample Program .....	N-11



## HIGH-SPEED SYNCHRONOUS CHANNEL UNIT (HSC930)

### GENERAL DESCRIPTION

The High-Speed Channel, known as the HSC930, is a two-row module that can be mounted into option racks 2 or 4 of the DATANET-30\* processor.

The HSC930 provides the DATANET-30 with the capability of operating with high-speed data transmission equipment over broadband lines at speeds up to 50 kilobits per second, or over voice grade lines at 2,000 or 2,400 bits per second. The HSC930 is capable of operating in either the half- or full-duplex mode (full-duplex requires two channels). (See Figures N-1 and N-2.)

The HSC930 has three interfaces with the exterior environment -- the DATANET-30 buffer selector, Interrupt Scanner Unit (ISU930), and the Data Set.

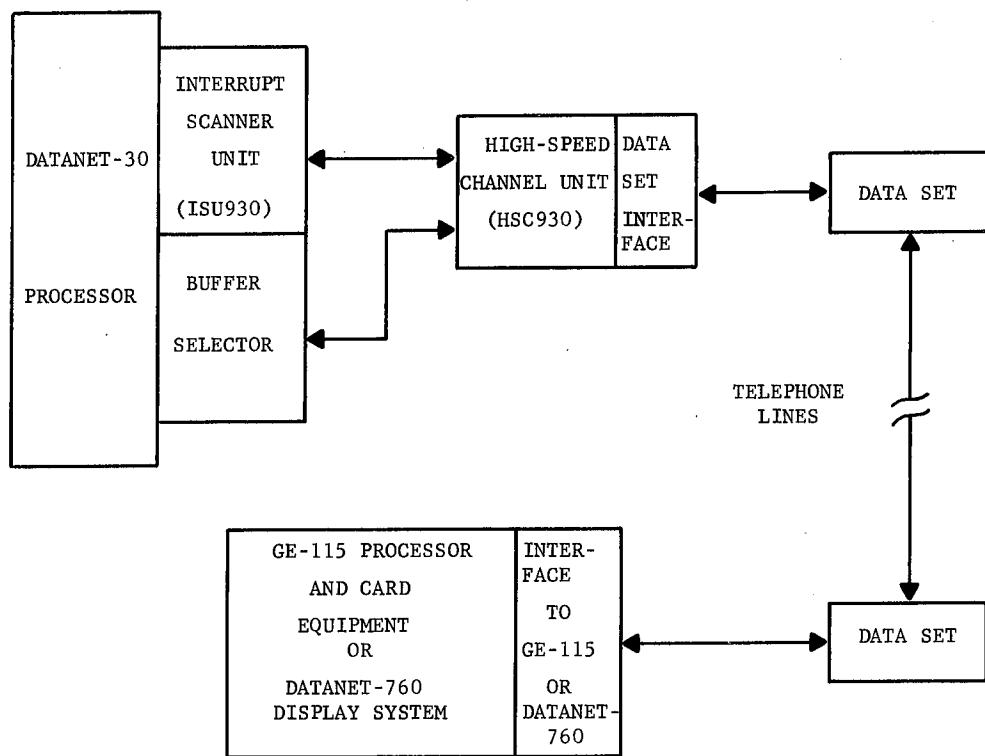


Figure N-1. Half-Duplex High-Speed Channel Unit with Telephone Company Data Sets and an Interrupt Scanner Unit.

\*DATANET, Reg. Trademark of the General Electric Company.

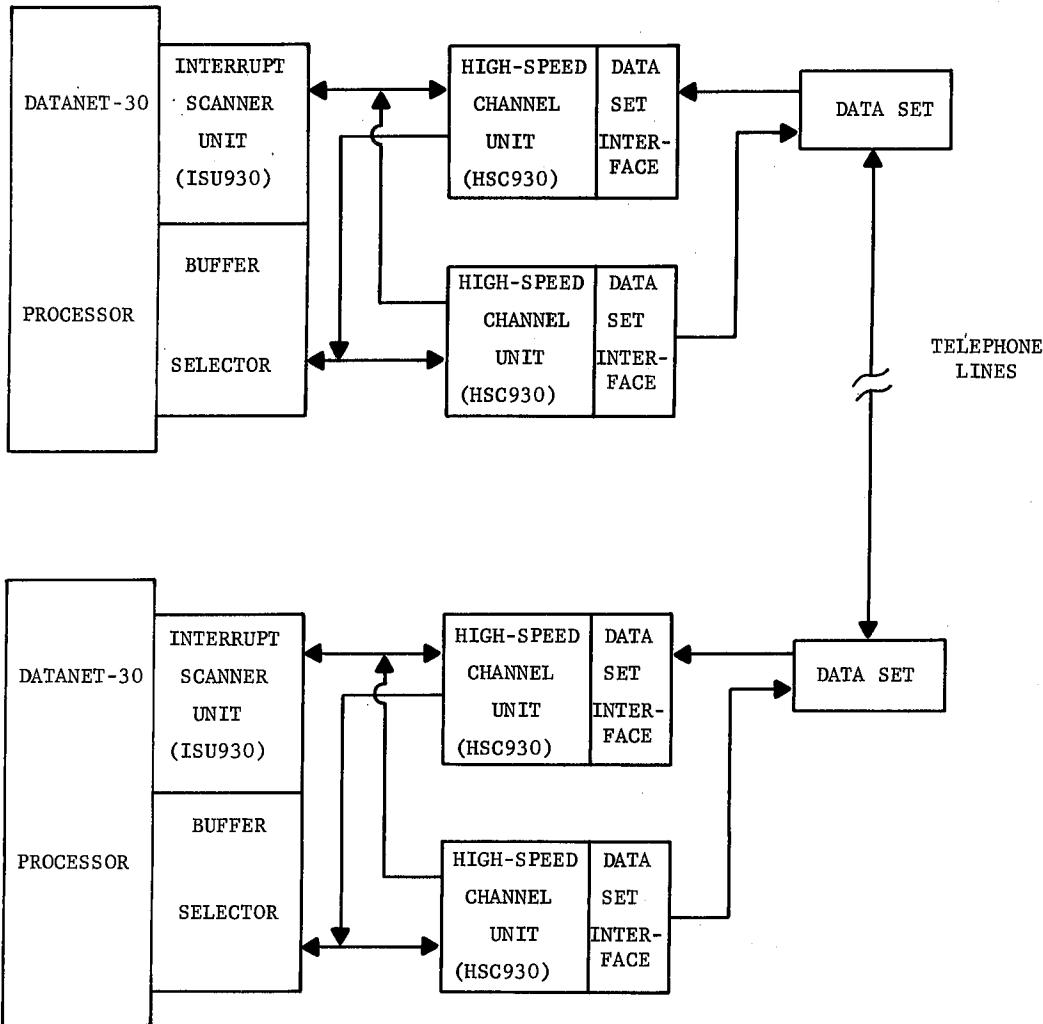


Figure N-2. Full-Duplex High-Speed Channel Unit with Telephone Company Data Sets and an Interrupt Scanner Unit.

## DATA TRANSFER

The HSC930 can transfer a maximum of 1024 words or 2048 characters (two characters per word) between the DATANET-30 memory and the remote station. Starting addresses in multiples of  $200_8$  ( $128_{10}$ ) may be specified within the 16k DATANET-30 memory by setting of switches.

The HSC930 automatically accesses memory sequentially through the Interrupt Scanner Unit at a maximum rate of 1 out of 40 memory cycles<sup>#</sup> and packs the characters two per word when placing information in the DATANET-30 memory. (See Figure N-3 for the Internal Code format for the Receive Mode.) When taking information from the DATANET-30 memory, two characters per word are transferred from the memory and transmitted serially. (See Figure N-4 for the Internal Code format for the Transmit Mode.)

---

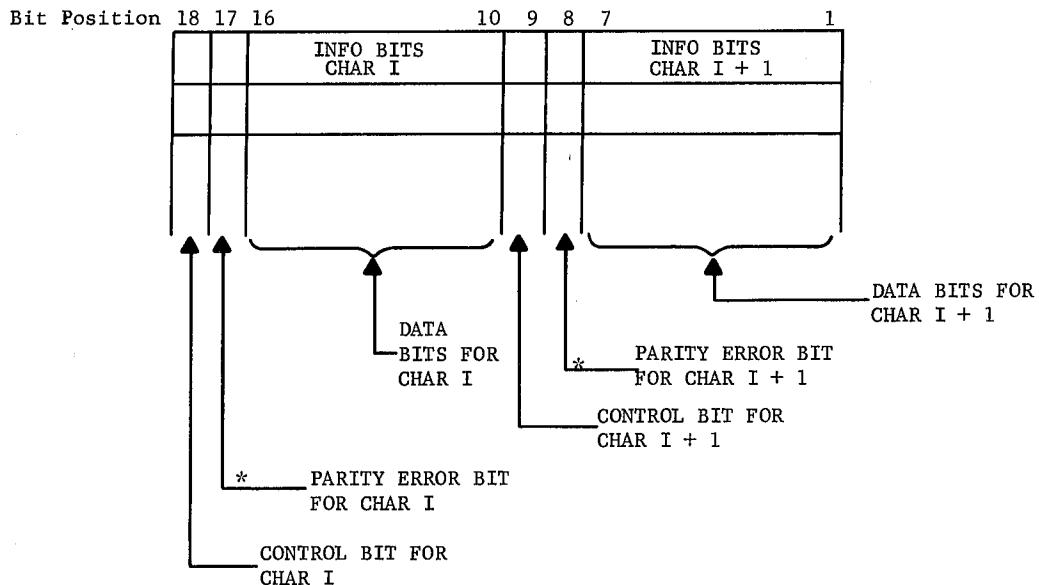
<sup>#</sup> The maximum memory access rate for the 6+1 code at 50 kilobits per second will be 1 out of 20 memory cycles.

The following character transfer rates may be serviced:

<u>Data Set #</u>	<u>Character Format Bit/Char.</u>	<u>Trans. Speed Kilobits/Sec.</u>	<u>Char./Sec.</u>
201A	6 + 1	2.0	286
201A	7 + 1	2.0	250
201A	8	2.0	250
201B	6 + 1	2.4	343
201B	7 + 1	2.4	300
201B	8	2.4	300
301B	6 + 1	40.8	5830
301B	7 + 1	40.8	5100
301B	8	40.8	5100
303B	6 + 1	19.2	2790
303B	7 + 1	19.2	2400
303B	8	19.2	2400
303C	6 + 1	50.0	7200
303C	7 + 1	50.0	6250
303C	8	50.0	6250

## CHECKING FEATURES

The HSC930 has several parity checking and generating features built into the hardware. They are listed and explained below. In addition, it has an overflow indicator to indicate when the assigned memory block is exceeded.



\*NOTE: Bits 8 and 17 are used to store data when receiving an 8-bit code with no character parity.

Figure N-3. Internal Code Format (Receive Mode)

#While only Bell System synchronous Data Sets are shown, equivalent Data Sets produced by other manufacturers may be substituted.

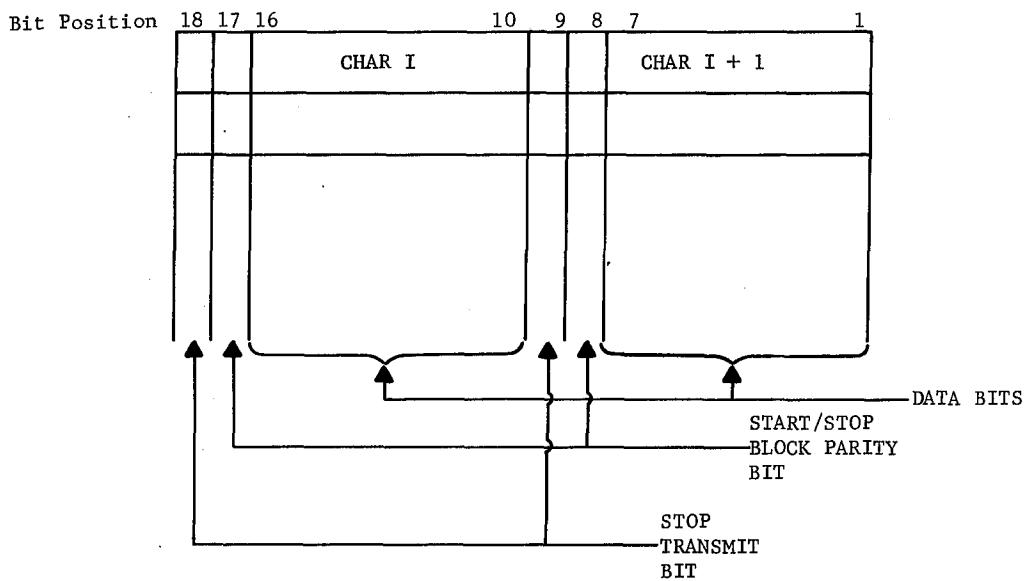


Figure N-4. Internal Code Format (Transmit Mode)

#### **Character Parity Checking\***

Each character is checked for odd parity as it is received from the line. If the character parity is incorrect, either bits 1 and 2, or bits 17 and 18 are set (depending on whether the erroneous character is the least significant or most significant character of the word, respectively) prior to transferring the erroneous word to memory. Also, the error status is set and can be sensed by the DATANET-30 buffer selector external status line 7 (Command level 1 - NES 7).

#### **Block Parity\***

As information is received from the line, block parity is accumulated on the message on a character-by-character basis. After the termination character (ETX or ETB) is received, the line adapter compares the 8-bit Block Check Character received from the line with the Block Check Character accumulated in the block check register. If the two do not compare, the check character is flagged (see paragraph above for Character Parity Checking) prior to being stored in memory. Also, the error flip-flop is set, and the error condition indication is made available to the DATANET-30 via external status line 7. (Command level 1 - NES 7.)

#### **Character Parity Generation\***

When the HSC930 is conditioned to the transmit mode, character parity is generated automatically. Odd character parity is generated on each character (including the block check character) immediately prior to transmission of the character. The parity bit is transmitted as the last bit of each character.

---

\* The block parity and character parity checking and generation features described above are used only with the 6 + 1 and 7 + 1 character formats. Software is responsible for character and block checking in the 8-bit format.

## **Block Check Character Generation\***

When the HSC930 is conditioned to the transmit mode, a horizontal check character is generated on a bit-by-bit basis as the message is transmitted. When a start bit is recognized (bit 8 or 17 on), the accumulation of the 8-bit Block Check Character starts on the character following the SOH or STX Control Character. The block check accumulation continues until a stop bit (bit 8 or 17 on) is detected. This character is included in the BCC, and then the block check is transmitted after the ETX or ETB control character.

## **Address Overflow**

When the last memory location of the assigned memory block has been accessed, the line adapter senses an attempt to access memory one more time. If the additional attempt is made, the address overflow flip-flop and the error flip-flop are set, and both status indications are made available to the DATANET-30 via external status lines 7 and 8. (Command level 1 - NES 8.)

## **BLOCK DIAGRAM**

The HSC930 contains a shift register, one 16-bit data buffer, an address counter, a bit counter, a block check register, control character recognition logic, mode control and data transfer control logic.

The HSC930 has three interfaces (shown in Figure N-5):

1. Data Set
2. Buffer Selector
3. Interrupt Scanner

The Data Set Interface provides the necessary logic and circuitry to allow the HSC930 to operate with any one of several data subsets.

The Buffer Selector Interface presents subset and line adapter status (NES lines) to the DATANET-30 program and accepts control signals (DEF lines) from the DATANET-30 program.

Any buffer channel address from 1 to 127 may be assigned to the line adapter. When the C-register (in the DATANET-30) contains the assigned address, the buffer selector instructions apply to the High-Speed Channel.

The Interrupt Scanner operates through the DATANET-30 interrupt interface. It is used to transfer data under memory interrupt control between the HSC930 and the DATANET-30 memory locations specified by the address counter and the address selection switches.

---

\* The block parity and character parity checking and generation features described above are used only with the 6 + 1 and 7 + 1 character formats. Software is responsible for character and block checking in the 8-bit format.

## Address Counter

The address counter is a 10-bit binary counter that is reset by the Set Transmit Mode command or by the Set Receive Mode command with the memory address control flip-flop reset. The combination of the 10-bit register and the address selection switches provides a 14-bit address register which holds the address of the memory location where data is to be removed from or stored in. The address counter is incremented by one immediately following each memory access. When the address counter and the address selection switches combined reach a count of all ones (lower 10 bits only), the line adapter checks for an attempt to access beyond that location. One more access attempt results in an address overflow error condition.

## Buffer Register

The buffer register temporarily stores 16 bits of data as it flows from memory to the shift register and from the shift register to the DATANET-30 memory. When a full 16-bit word is received in the shift register, it is automatically transferred from the shift register to the data buffer. When a full word has been transferred from the shift register to the line, another word is automatically transferred from the data buffer to the shift register. When the data buffer is filled on Receive or emptied on Transmit, a memory access request is automatically sent to the Interrupt Scanner.

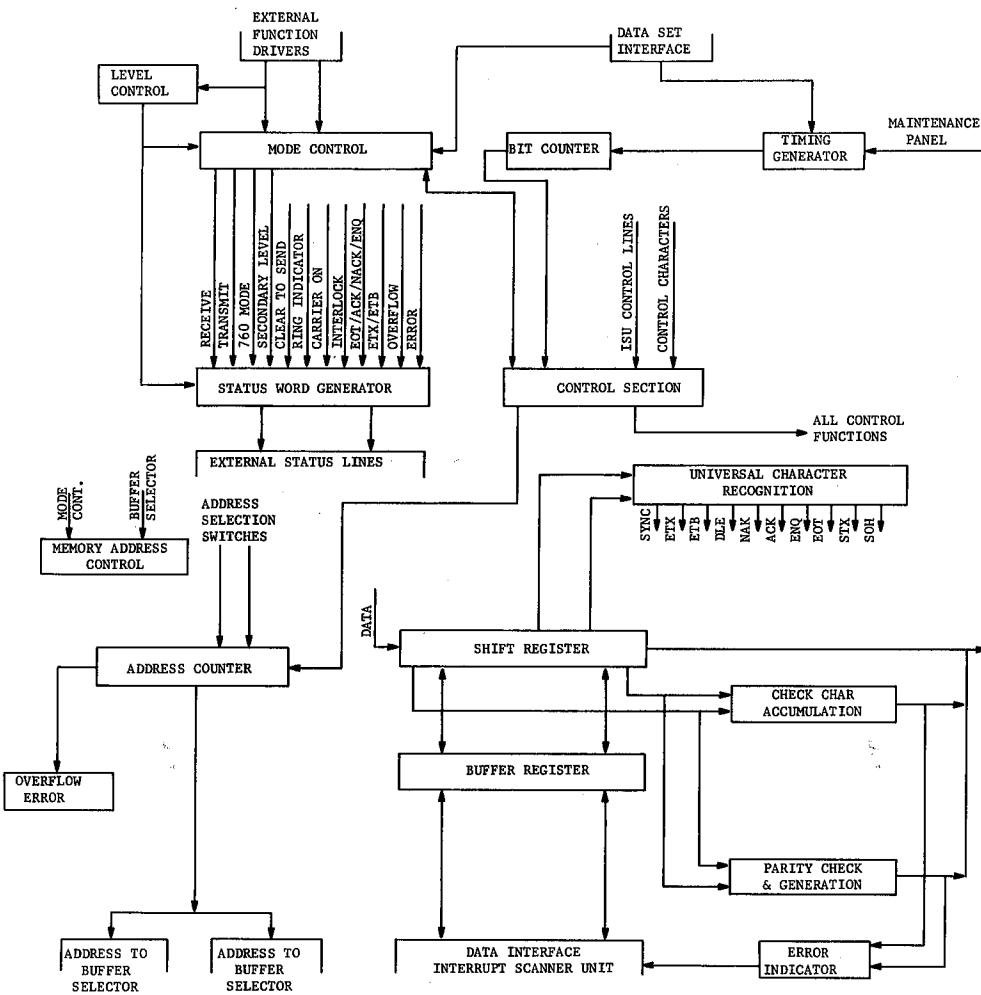


Figure N-5. Block Diagram

## **Universal Character Recognition**

This logic provides the necessary gating to decode the control characters used in the acceptable message formats. Line control characters which are decoded for the 7 + 1 (ASCII) code are SYN, ACK, NAK, ETX, ETB, STX, SOH, DLE, ENQ, and EOT. For 6 + 1 and 8 + 0 applications, equivalent unique control characters must be selected from those used with the 7 + 1 formats and plugged accordingly.

## **Shift Register**

The shift register is a 16-bit shift register that is implemented to receive 16 bits of serial information from the line or transmit 16 bits in serial form to the line.

## **INSTRUCTION REPERTOIRE**

The instructions are classified under buffer selector instructions for the DATANET-30. They are the Drive External Function (DEF), the External Status Lines (NES), and the Receive Data Lines (RDL) instructions.

The rules for using the DEF, NES, and RDL instructions are the same as those for using other DATANET-30 buffer selector instructions.

The Command Level flip-flop is used to expand the DEF and NES instructions from a maximum of 10 to a maximum of 18 usable function drivers and from a maximum of 10 to a maximum of 19 usable status lines. (Two drivers and one status line are used with the command level function.)

### **Drive External Function Instructions**

The Drive External Function instructions are as follows.

#### **COMMAND LEVEL 1 (ALL MAJOR FUNCTIONS)**

- DEF 1      Not to be used.
- DEF 2      Space reserved.
- DEF 3      Set Idle Mode. This line resets both Transmit and Receive Mode control flip-flops and leaves the line adapter in the Idle mode.
- DEF 4      Set Transmit Mode. The Transmit mode flip-flop when set, issues the Request to Send signal to the Data Set and provides the necessary control signals within the line adapter to accomplish transfer of data from the DATANET-30 memory to the Data Set.
- DEF 5      Set Receive Mode. This driver will set the Receive mode flip-flop, reset the Transmit Mode and Request to Send flip-flops, and may (depending upon the condition of the memory address control flip-flop) reset the address counter. The Receive mode flip-flop provides the necessary control signals to accomplish transfer of data from the Data Set to the DATANET-30 memory.

- DEF 6      Reset Line Status Indicators. This line resets the Address Overflow, the Any Error, the EOT, and the ETX/ETB flip-flops. Mode control flip-flops are unaffected. This command will not affect the reception or transmission of messages.
- DEF 7      Set Data Terminal Ready. This command is issued in response to the ring indication from the Data Set. It sends the Data Terminal Ready status to the Data Set (answers phone).
- DEF 8      Reset Data Terminal Ready. This command removes Data Terminal Ready status from the Data Set input (hangs up phone).
- DEF 9      Set Command Level 2. The command level flip-flop, when set, changes the functions of external function drivers and meanings of external status lines to those listed under Command Level 2.
- DEF 10     Reset Command Level 2. The command level flip-flop, when reset, changes the functions of external function drivers and meaning of external status lines to those listed under Command Level 1.

#### COMMAND LEVEL 2 (ADDED FUNCTIONS)

- DEF 1      Not to be used.
- DEF 2      Set DATANET-760 Mode. This mode, when set, allows the line adapter to recognize the EOT following the Block Check Character and not preceded by synchronizing characters. It will also suppress status indications for ACK, NAK, and ENQ characters as described below under External Status Lines Instructions (Command Level 1, NES 6).
- DEF 3      Reset DATANET-760 Mode. With the 760 mode reset in the line adapter, no line signal will be recognized unless preceded by at least two consecutive valid synchronizing (SYN) characters.
- DEF 4      Set Request to Send. The Request to Send flip-flop sends the Request to Send signal to the Data Set. The flip-flop does not provide control signals within the line adapter. The line adapter must be placed in the Transmit mode (Command Level 1 and a DEF 4 instruction) in order to accomplish any transfer of data.
- DEF 7      Set Memory Address Control Flip-Flop. The memory address control flip-flop, when set, allows the memory address counter to reset when the mode is changed from Transmit to Receive. The address contained in the address counter would be reset to the beginning block address.
- DEF 8      Reset Memory Address Control Flip-Flop. The memory address control flip-flop, when reset, prevents the address counter from being reset when the mode changes from Transmit to Receive. Characters received will be stored in the next consecutive memory address.
- DEF 9      Set Command Level 2. The command level flip-flop, when set, changes the functions of external function drivers and meaning of external status lines to those listed under Command Level 2.
- DEF 10     Reset Command Level 2. The command level flip-flop, when reset, changes the functions of external function drivers and meaning of external status lines to those listed under Command Level 1.

## **External Status Lines Instructions**

The External Status Lines Instructions are as follows.

### **COMMAND LEVEL 1 (ALL MAJOR FUNCTIONS)**

- |       |   |
|-------|---|
| NES 1 | Test Ring Indication. A one indicates that the Data Set is sending the ring indication. This line is used to inform the program of an incoming call.  |
| NES 2 | Test Carrier Detect. This line is used to test the condition of the carrier line in 201A3 and 301B Data Sets. It tests the AGC line in 303A-type Data Sets.   |
| NES 3 | Test Interlock. This line informs the operating program of the condition of the Data Set. A one indicates that power is on in the Data Set, and that the Data Set is operable.  |
| NES 4 | Test Transmit Mode. This line informs the operating program of the condition of the Transmit mode control flip-flop. A one indicates that the Transmit mode control flip-flop is set. The line adapter may or may not be in the process of transmitting data.   |
| NES 5 | Test Receive Mode. This line informs the operating program of the condition of the Receive mode control flip-flop. A one indicates that the Receive mode flip-flop is set. The line adapter may or may not be receiving information.  |
| NES 6 | Test EOT. This line indicates that an EOT, ACK, NAK, or ENQ has been received. The software determines which of these four characters actually caused the EOT indication to be set. The EOT indication will be set upon receipt of the ending character and will be reset when the DEF 6 instruction is issued by the operating program.  |
| NES 7 | Test Any Error Condition. This line provides an indication to the DATANET-30 program that an error condition exists as a result of the receipt or transmission of the last message. In the Receive mode, the error condition may indicate: <ul style="list-style-type: none"><li>• incorrect character parity on any character within the message</li><li>• incorrect block parity; i.e., the computed Block Check Character did not compare with that received from the line</li><li>• address overflow, or that the line adapter attempted to address memory beyond the limits of the established block length.</li></ul> |

In the Transmit mode, the error line would indicate the address overflow condition.

NOTE: The only error condition checked when using the 8 + 0 code is the address overflow.

- |       |   |
|-------|---|
| NES 8 | Test Address Overflow. This line provides an indication to the DATANET-30 program that the line adapter has attempted to address memory beyond the limits of the established block length. This allows the program to determine the cause of the error indication on ESL 7 (Command Level 1 - NES 7). |
|-------|---|

- NES 9      Test Block Ending Status. This line enables the operating program to detect the receipt of either the ETX or ETB ending control character. A one indicates receipt of ETX or ETB. The line will remain set until a DEF 6 instruction is issued to reset the status.
- NES 10     Test Command Level 2. This line allows the program to determine which command level is set in the line adapter. A one indicates that the line adapter is conditioned to recognize commands in Level 2.

#### COMMAND LEVEL 2 (ADDED FUNCTIONS)

- NES 1      Test Clear to Send. This line informs the operating program that the Data Set has been properly conditioned to the Transmit mode and is ready to begin transmitting data.
- NES 2      Test DATANET-760 Mode. This line indicates to the program that the line adapter has been conditioned to recognize the EOT character following a Block Check Character and not preceded by synchronizing characters. (See Drive External Function instructions, Command Level 2, DEF 2.)
- NES 10     Test Command Level 2. This line allows the program to determine which command level is set in the line adapter. A one indicates that the line adapter is conditioned to recognize commands in Level 2.

Those DEF and NES instructions which are not defined above should not be used by software. While it is expected that the performance would be independent of control level in the initial versions, incompatibilities will result due to future modifications.

#### **Receive Data Lines Instructions**

- TRA R,\*  
(\* may be A, B, or Z)      Bits 1-14 of the Receive data lines contain the information from the address counter and address selection switches. When the operating program desires this information, a register transfer instruction from R to any register may be used to obtain the information.

#### **CONTROLS AND INDICATORS**

##### **Address Selection Switches**

Address selection switches are provided on the maintenance panel to allow the user to select the starting location of the message area. The appropriate message area size is selected by the use of patch-plug wiring on each HSC930.

##### **Mode Selection Switches**

Switches are provided on the maintenance panel to allow the user to select either Receive Only or Transmit Only for full-duplex operation. With the Receive Only switch thrown, the Receive mode control flip-flop is held set. With the Transmit Only switch thrown, the Transmit mode control is allowed to set or reset under program control. The Receive mode flip-flop is not allowed to set.

## SAMPLE PROGRAM

A sample program is shown in Figure N-6.

<u>LOC</u>	<u>INST</u>	<u>SYMBOL</u>	<u>OPR OPERAND X</u>	<u>REMARKS</u>
	00010		LOC 10	
00010	200000	ADD	INB 0	
00011	000000	MADD	OCT 0	
00012	000000	ERR1	IND 0	
00013	000014		IND *+1	
00014	000100		HLT 64	
				* SUBROUTINE FOR REPORT OF DATA SET NOT READY
00016	000000	ERR2	IND 0	
00017	000020		IND *+1	
00020	000101		HLT 66	
				* SUBROUTINE FOR REPORT LEVEL 2 NOT SET
00022	000000	ERR3	IND 0	
00023	000024		IND *+1	
00024	000102		HLT 66	
				* SUBROUTINE FOR REPORT LEVEL 1 NOT SET
00026	000000	ERR4	IND 0	
00027	000030		IND *+1	
00030	000103		HLT 67	
				* SUBROUTINE FOR REPORT RECEIVE MODE NOT SET
00031	300032	MAP	INC *+1	7 + 1 ASCII MAP
00032	026026		OCT 026026	SYNC, SYNC
00033	026026		OCT 026026	SYNC, SYNC
00034	000202		OCT 000202	NUL, STX
00035	141140		OCT 141140	DATA, DATA
00036	143142		OCT 143142	DATA, DATA
00037	125052		OCT 125052	DATA, DATA
00040	174603		OCT 174603	BCC, ETX
00041	000000		OCT 0	
				*
				* PROGRAM STARTS
				*
	00100		LOC 100	
00100	011020	START	PIC 16	PLACE ADDRESS IN C
00101	026400		DEF 9	SET LEVEL 2
00102	023000		NES 0	TEST LEVEL 2 SET
00103	130105		BNZ *+2	YES
00104	110016		BRS ERR2	NO, REPORT LEVEL 2 NOT SET
00105	026100		DEF 7	SET MAC F/F
00106	027000		DEF 0	SET LEVEL 1
00107	023000		NES 0	TEST LEVEL 1 SET
00110	120112		BZE *+2	YES
00111	110022		BRS ERR3	NO, REPORT LEVEL 1 NOT SET
00112	026020		DEF 5	SET RECEIVE MODE
00113	022020		NES 5	TEST RECEIVE MODE SET
00114	130116		BNZ *+2	YES
00115	110026		BRS ERR4	NO, REPORT RECEIVE MODE NOT SET
00116	026004		DEF 3	SET IDLE MODE
00117	022004		NES 3	TEST DATA SET READY
00120	130122		BNZ *+2	YES
00121	110012		BRS ERR1	NO, REPORT DATA SET NOT READY
00122	060050		TRA R,A	TRANSFER BUFFER MEMORY
00123	500011		STA MADD	ADDRESS TO A-REGISTER
				* PROGRAM CONTINUES
00100			END START	

Figure N-6. Sample Program



## DOCUMENT REVIEW SHEET

TITLE: HSC930 High-Speed Channel for DATANET-30

CPB #: 1412

FROM:

Name: \_\_\_\_\_

Position: \_\_\_\_\_

Address: \_\_\_\_\_  
\_\_\_\_\_

Comments concerning this publication are solicited for use in improving future editions. Please provide any recommended additions, deletions, corrections, or other information you deem necessary for improving this manual. The following space is provided for your comments.

COMMENTS: \_\_\_\_\_

Please cut along this line

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

NO POSTAGE NECESSARY IF MAILED IN U.S.A.  
Fold on two lines shown on reverse  
side, staple, and mail.

STAPLE

FOLD

---

FIRST CLASS  
PERMIT, No. 4332  
PHOENIX, ARIZONA

**BUSINESS REPLY MAIL**  
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

**POSTAGE WILL BE PAID BY**

**GENERAL ELECTRIC COMPANY**  
**COMPUTER EQUIPMENT DEPARTMENT**  
**13430 NORTH BLACK CANYON HIGHWAY**  
**PHOENIX, ARIZONA - 85029**

**ATTENTION:** DOCUMENT STANDARDS AND PUBLICATIONS B-84

---

FOLD

## APPENDIX C

### CIU931 EXTERNAL DRIVE AND STATUS LINES

#### Drive Line

DEF 1	Reset Receive Data Flag
DEF 2	Reset Transmit Data Flag
DEF 3-4	Not Used
DEF 5	Reset Terminate
DEF 6	Reset Status
DEF 8	Write Initiate
DEF 9	Read Response
DEF 0	End Data Transfer

#### External Function

#### Status Line

1

#### External Status

##### Data Flag

Indicates that the CIU931 is ready for another word.

2

Not Used

3-6

##### Major Status

7-8

Not Used

9

##### Special Interrupt

Indicates that the GE-625/635 system is ready to send.

0

##### Terminate

Indicates a data transfer has been terminated.



## APPENDIX D

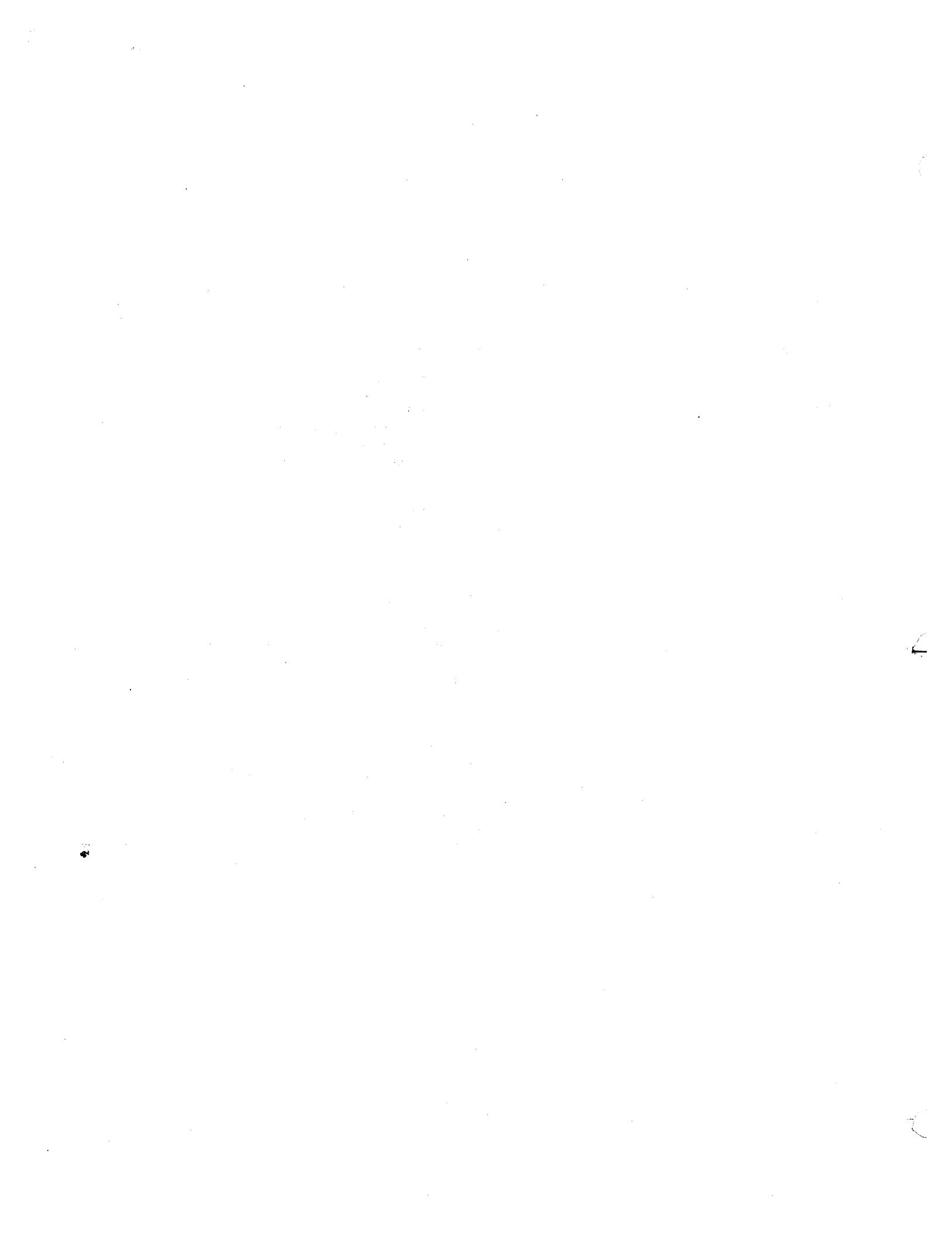
### CBC931 EXTERNAL DRIVE AND STATUS LINES

#### Drive Line

1	Reset Receive Flag
2	Reset Transmit Flag
3	Set Receive Mode
4	Turns carrier signal off.
5	Set Transmit Mode
6-8	Turns carrier signal on.
9	Reset Buffer
0	Not Used
	Answer Incoming Call
	Disconnect Call

#### Status Line

1	Receive Flag Set
2	Data Register contains a new character.
3	Transmit Flag Set
4	Data Register is ready for a new character.
5	Call in Progress Indicator
6	Phone is off the hook.
7	Ring Indicator
8-0	Phone is ringing.
	Interlock
	Indicates subset is in the data mode.
	Carrier On
	Carrier signal is on.
	Clear to Send
	Not used



## DOCUMENT REVIEW SHEET

TITLE: DATANET-30 Programming Reference Manual  
CPB #: 1019A

Name: \_\_\_\_\_

Position: \_\_\_\_\_

Address: \_\_\_\_\_  
\_\_\_\_\_

### CHECK ONE:

- Additional information would be helpful on following subjects.
- Errors indicated and pages where errors occur.
- Usefulness of manual could be improved as noted.

My comments are: \_\_\_\_\_

Please cut along this line

STAPLE

STAPLE

FOLD

---

FIRST CLASS  
PERMIT, No. 4332  
PHOENIX, ARIZONA

**BUSINESS REPLY MAIL**  
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

GENERAL ELECTRIC COMPUTER DEPARTMENT  
13430 NORTH BLACK CANYON HIGHWAY  
PHOENIX, ARIZONA - 85001

ATTENTION: Program Documentation  
Systems and Processors Operation

---

FOLD

