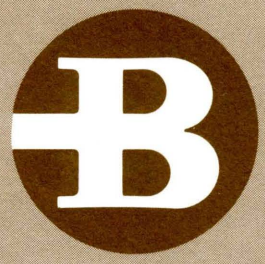


Burroughs

SERIES L/TC

SYSTEM SOFTWARE OPERATION GUIDE

Burroughs

SERIES L/TC

SYSTEM SOFTWARE

OPERATION GUIDE

Burroughs Corporation
Detroit, Michigan 48232

Burroughs Corporation
Detroit, Michigan 48232

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

* An asterisk denotes the I.D. number of a Utility Routine to be used on a 40 track Series L System. Their operating and functional characteristics are the same as their counterpart 32 track system routine.

INTRODUCTION

The purpose of this manual is to consolidate all information pertaining to the software of the Series L/TC with the exception of the assemblers and compilers which are separate manuals. This manual is presently organized into these sections: Section One; contains utilities that are generally used with all systems. Section Two; utility routines that are used with systems having paper tape capabilities. Section Three; consists of those utilities that must be used with systems having card I/O capabilities. Section Four; routines used to qualify various TC networks. Section Five; detailed discussion of machine language coding and interpretation.

Each routine is itself divided into several topics: A General Description whose purpose is to give the type of results to be expected and present any inherent limitations, a detailed discussion of the Operating Instructions, and a section for special considerations. Such information as firmware compatibility and dependency, and the memory area used by each utility is included in the individual write-ups.

Several of the utility routines have two I. D. numbers associated with them. One is to identify 32 track utilities and the other, marked with an asterisk, identifies utilities to be used on a 40 track system. The two versions must be used with the system they were designed for. They are not interchangeable.

Reference is made in this manual to normal program load procedures. This actually implies using the loader device to load the utility routine.

When using the loader device, depress the Load key (PKA 2) after loading the tape into the cartridge. Turn the Memory Load Switch on to give power to the memory loader. This will cause the tape to be read and stored in memory. If the program tape fails to read properly, an alarm will sound and the computer returns to the Ready Mode. Turn the Memory Load Switch to OFF to stop the loading, and start the procedure over again. If the alarm sounds after repeated loading attempts, the program tape may have to be replaced because of damage or wear, or service attention may be required. When the tape has been properly loaded, turn the Memory Load Switch to OFF and depress the Reset key to return the computer to the Ready Mode.

The contents of this manual replace any and all previous publications concerned with the aforementioned software of the Series L/TC. This includes Provisional Release Letters.

NOTE: Throughout this manual, the last two digits of the normal 10-digit program identification number have been omitted to avoid any reference to revision levels. Subsequent revision levels should be used as they become available. Refer to the Group II Software Catalog for the complete 10-digit identification number.

SECTION 1

GENERAL ROUTINES

MEMORY MODIFY AND SELECTIVE PROGRAM START

UTILITY ROUTINE I. D. No. 1-1001-060
UTILITY ROUTINE I. D. No. 1-1101-013*

GENERAL DESCRIPTION

The Memory Modify routine provides for the following:

- Programs, print masks, and data can be entered into memory through the keyboard one word at a time.
- Alphanumeric data may be entered by indexing the corresponding key on the typewriter keyboard. Hexadecimal entries are not required.
- Program execution can begin at any selected memory location.
- Printing the contents of any specified memory word.

The entries and/or changes are left in memory and subsequently may be punched from memory to produce a new or corrected object program.

Memory Modify resides in the utility track and is compatible with all firmware sets. It is loaded into memory using the normal Memory Load routine.

OPERATING INSTRUCTIONS:

From the Ready mode depress PKA 3 (Utility Key). The carrier will position to 10 on the print scale and the Numeric Keyboard will be enabled.

WORD AND MODE SELECTION

Index (in decimal) the word desired (O:N). Terminate the entry with one of the following:

OCK 1 – Memory Alteration

OCK 2 – Alphanumeric Data Entry

OCK 3 – Selective Program Start

OCK 4 – Memory Word Listing

After depressing one of the above OCK's, the memory word number indexed will be printed using red ribbon. What follows the printing of the word number is dependent on the mode selection.

MEMORY ALTERATION

When Memory Alteration is selected by terminating the word entry with OCK 1, the contents of the selected word will be printed in black ribbon on the same line as the word number using the following format:

(Word)		(Contents)			
156		EB10	A650	D440	3026
	Digit Position	1111 5432	11 1098	7654	3210
	Syllable	3	2	1	0
	Instruction	POS 16	NK 5,0	PN 4,0	TRM 38

Notice that the system prints the syllables left to right beginning with syllable 3 and ending with 0. This will differ from the object program listing which generally prints each syllable on successive lines starting with syllable 0 and reading down to syllable 3.

Each digit position of a word is represented in hexadecimal machine language code. Refer to Section Five for Operation Codes, Parameter Field Bounds and Hex Conversion Tables.

The carrier is positioned to 48 and the numeric keyboard is enabled. The contents of the selected word may be a Program Word, a Print Mask, or a Numeric Word. The alterations to memory are made accordingly.

Program Word Alteration

Program word alteration is entered in the Series L in Hexadecimal machine language.

1. At position 48 index the highest digit position to be altered. Depress any OCK. The digit position selected will be printed and the Alpha Keyboard will be enabled.

2. Enter the change to the position selected as a hexadecimal digit on the Alpha Keyboard. Valid characters are: 0-9, A, B, C, D, E, F.

   If an invalid character is depressed, the alarm sounds, the error light illuminates and additional typing is prevented. To eliminate the error state and re-initiate the keyboard, depress the Reset Key twice. A valid entry decreases the position counter by one, thus a second entry would change the next lowest digit position value. In this manner, consecutive descending digit positions may be altered.

3. If an error is made in an entry or if changes to be made are not in consecutive descending order, terminate with OCK 3. The paper will be vertically spaced and the program will return to step 1 for entry of another digit position. When the last change has been made, terminate with OCK 1.

4. Termination of the entry with OCK 1 causes the paper to advance one line; the modified word now stored in memory prints directly under the original word selected; and the system returns to Word and Mode Selection.

   EXAMPLE 1:

Word	Syl 3	Syl 2	Syl 1	Syl 0	Digit Position	Hex Digit
15	0333	6E66	E666	C444	3	9
	0333	6E66	E666	9444		

EXAMPLE 2:

Word	Syl 3	Syl 2	Syl 1	Syl 0	Digit Position	Hex Digits
32	2075	7873	2075	2896	11	740BD811EB4B
	2075	740B	D811	EB4B		

Print Mask Alteration

Print mask alterations follow the same steps as the previously described "Program Word Alterations". Alterations of memory words containing print masks are entered in the Series L hexadecimal machine language. Refer to the appendix for the hexadecimal interpretation of the print mask characters.

Numeric Word Alteration

Numeric word alterations follow the same steps as those described in "Program Word Alterations". The hexadecimal values displayed in digit positions 0 thru 14 of the numeric memory word are identical to their decimal equivalents.

Hex 0 = decimal 0

Hex 1 = decimal 1

⋮ ⋮

Hex 9 = decimal 9

Digit position 15 contains the hexadecimal mask character. Refer to appendix to obtain the correct entry for digit position 15.

ALPHANUMERIC DATA ENTRY

When Alphanumeric Data Entry is selected by terminating the word entry with OCK 2, the Alpha keyboard will be enabled immediately after printout of the word number.

NOTE: Do not select the Alphanumeric Data Entry option to alter the contents of the Data Comm Buffer. Any alteration to this area of memory must be made using the procedures outlined under PROGRAM WORD ALTERATION.

1. Alphanumeric data is entered on the Alpha keyboard and stored 8 characters to a word beginning with the memory word indexed. A maximum of 128 characters can be entered.

2. Depression of any OCK will return the system to Word and Mode Selection.

SELECTIVE PROGRAM START

When Selective Program Start is selected by terminating the word entry with OCK 3, the numeric keyboard is enabled following the printout of the word number.

1. Enter the syllable number (0, 1, 2, 3) of the instruction desired on the Numeric Keyboard.

2. Depression of OCK starts the following:
   a. Printing of the syllable number.

b. Restoring the Index Registers, Keyboard Register, Print Register, and PK Registers to their original value upon entering the memory modify routine.

c. Branching to word and syllable number entered to begin program execution at that point.

NOTE: The Accumulator and Accumulator Flags will contain arbitrary data at this point.

MEMORY WORD LISTING

When Memory Word List is selected by terminating the word number entry with OCK 4, the contents of the memory word designated will printout in the format previously discussed. The paper will be advanced 2 vertical spaces and the system will return to Word and Mode Selection.

TRACE I, FULL TRACE

UTILITY ROUTINE I.D. No. 1-1001-057

UTILITY ROUTINE I.D. No. 1-1101-016*

GENERAL DESCRIPTION

The Trace I Utility Routine is intended as a tool for the programmer. It permits the macro program to be executed in its normal step by step sequence and has the ability to provide a printed record of the contents of the accumulator and certain registers along with a print-out of each macro instruction executed.

Four types of output are available in Trace I. The difference is in the amount of information that is printed.

1. Full Trace Routine provides for the printing of the instruction itself, the word and syllable location of the instruction, the contents of the index registers, the contents of the flag registers, and the contents of the accumulator.

2. A second option will print the word and syllable location of the instruction being executed, the instruction itself, and the contents of the accumulator.

3. A third option will print only the word and syllable location of the instruction and the instruction itself.

4. A fourth option will print the word and syllable location of the instruction, the instruction itself, and the contents of the Pointer Register. This option is only of value when a Data Communications program is being traced.

In all cases, all printing will be in hexadecimal format. The trace print-out will begin at position 81 on the scale and will occur prior to the actual execution of the current instruction. In other words, the trace routine will show the results brought about by the execution of the instruction on the previous line. All printing will be in RED RIBBON.

The Trace I Routine is compatible with all firmware sets and resides in the Utility Track; it cannot be resident in memory with another Utility Routine that would occupy the same space.

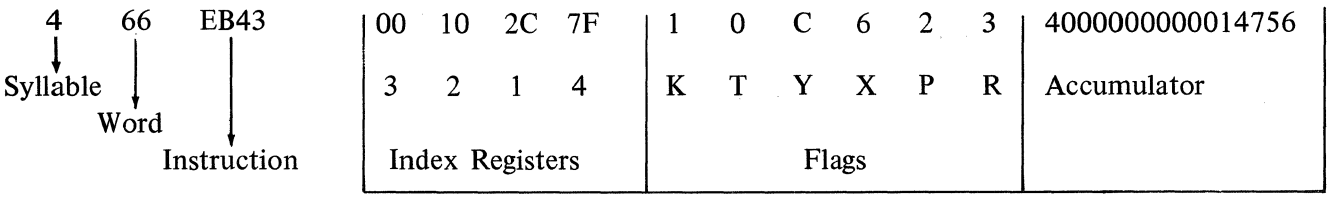
OPERATING INSTRUCTIONS:

The Trace program must be loaded using the normal Program Load procedures. The execution of the trace function begins after starting the object program by indexing a shifted numeric (1-9) at any numeric or alpha keyboard entry position in the object program. The shifted numeric digit may be entered before, between or after the normal entry. The Trace program may also be called into use while in the Ready mode by indexing a shifted numeric (1-9) prior to depressing the Start key (PKA 1). This method will enable the object program to be traced starting with the instruction in syllable 0 of word 0.

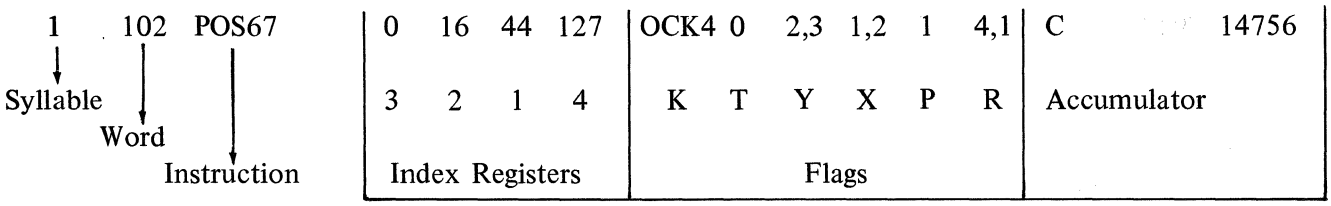
TRACE I

SHIFTED 1 (FULL TRACE EXCEPT POINTER REGISTER)

A shifted numeric 1 will call in that portion of the Trace I Utility that prints the word and syllable location of the instruction, the instruction itself, the contents of the Index Registers, the Flag Registers, and the content of the Accumulator including the flags in the sign position. The print-out will take the following format:



Translated into decimal format and mnemonics we have the following:



Syllable numbers 0, 1, 2 and 3 are printed as 0, 4, 8 and C respectively for words 0:255.

Syllable numbers 0, 1, 2 and 3 are printed as 1, 5, 9 and D respectively when 256 must be added to the word location printed.

Syllable numbers 0, 1, 2, and 3 are printed as 2, 6, A and E respectively when 512 must be added to the word location printed.

Flag Group	Hexadecimal Code 8	4	2	1
K (OCK's)	3	2	1	4
T	U	I	L	0
X,Y,R,P	3	2	1	5
Accum	M	C	S	–

The Trace Program can be stopped by indexing a shifted 0 at any keyboard entry position.

SHIFTED 2 (INSTRUCTION, WORD AND SYLLABLE, AND ACCUMULATOR)

A shifted numeric 2 will call that portion of Trace I that prints only the word and syllable location of the instruction being executed, the instruction itself, and the contents of the accumulator. The print-out takes the following format:

4	2B	ED01	0000000000000462
Syllable	Word	Instruction	Accumulator

Using the decimal translation rules discussed under the Shift 1 option, we have:

1	43	AL 1	462
Syllable	Word	Instruction	Accumulator

The Trace Program can be stopped by indexing a shifted 0 at any keyboard entry position.

SHIFTED 8 (INSTRUCTION, WORD AND SYLLABLE)

A shifted numeric 3,4,5,6 or 8 will call in that portion of the Trace I Utility that prints only the word and syllable location of the instruction and the instruction itself. The print-out will take the following format:

5	8C	80FD
Syllable	Word	Instruction

Using the decimal translation rules discussed under the Shift 1 option, we have:

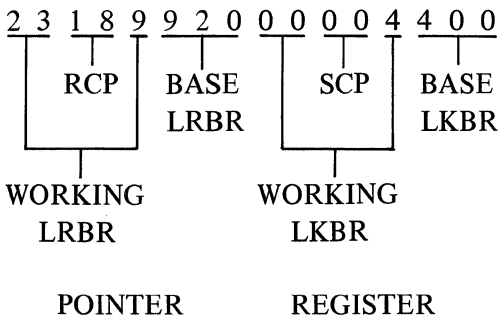
1	396	ADM 256
Syllable	Word	Instruction

The Trace Program can be stopped by indexing a shifted 0 at any keyboard entry position.

SHIFTED 9 (INSTRUCTION, WORD AND SYLLABLE AND POINTER REGISTERS)

A shifted numeric 7 or 9 will call in that portion of the utility that prints the word and syllable location of the instruction, the instruction itself, and the contents of the pointer register. This variation of Trace I would only be of value for a Data Comm Firmware set. The format follows:

5	8C	1D20
Syllable	Word	Instruction



Translated into decimal format and mnemonics as discussed under the Shift 1 option, we have:

1	396	PAB 32		(See Below)
Syllable	Word	Instruction	Pointer	Register

WORKING LRBR: Consists of two values; the one to the left of the RCP reflects the word currently in use and the one to the right indicates which Block the word is in. The maximum word value would be hexadecimal FF or decimal 255. The Block number could be one of three values: 8 indicating Block 0, 9 indicating that the word is in Block 1 or 4 indicating the Receive Buffer. In the format illustrated, hexadecimal 23 translates to decimal 35 while a block number of 9 indicates that 256 must be added to our word value of 35 since the word resides in Block 1. Thus we have word 296 in Block 1.

RCP: Hex 18 = decimal 24. The value of these two digits specifies the current character position within the Receive Buffer or within the working area.

BASE LRBR: Consists of two values: The first digit indicates the memory Block our area resides in and the remaining two digits specify the starting word of the area. The possible Block number values are the same as those discussed for the Working LRBR. The maximum word value would be hexadecimal FF or decimal 255. In the illustration, 920 indicates that the beginning word of our area is in Block 1 (9 = Block 1) and that 256 must therefore be added to our word value. Hexadecimal 20 translates to decimal 32; we add 256 and thus determine that the beginning of the area is at word 288 in Block 1.

WORKING LKBR: Consists of two values; the digits to the left of the SCP reflect the word currently being accessed and the digit to the right of the SCP indicates which Block the word is in. The maximum value for the word would be hexadecimal FF or decimal 256. The block number could be one of three values: 0 indicating Block 0, 1 indicating that the word is in Block 1 or 4 indicating the Transmit Buffer. In the format illustrated, after translating the word and block value we find that the word currently being accessed is word 0 in the Transmit buffer.

SCP: The value of these two hexadecimal digits specifies the current character position within the Transmit Buffer or within the working area.

BASE LKBR: Consists of two values: the first digit indicates the memory Block our area resides in and the remaining two digits specify the starting word of our area. The possible Block number values are the same as those discussed for the Working LKBR. The maximum word value would be hexadecimal FF or decimal 255. In the format shown, translating hexadecimal 400 would place the beginning of our area in the Transmit Buffer (4 = Transmit Buffer) at word 0.

The Trace Program can be stopped by indexing a shifted 0 at any keyboard entry position.

OTHER CONSIDERATIONS:

The depression of the READY button will cause the Trace Program to be stopped and the system to return to the Ready condition. Subsequently, if the Reset key was then depressed the program would return to the instruction it had just left, however the Trace Program would no longer be active.

Caution should be exercised that the Utility key is not depressed from the Ready mode while TRACE is residing in the Utility track. The machine would, in this case, attempt to execute a micro-instruction as a macro-instruction. The machine language happens to correspond to an XTKM 162. The alpha keyboard would therefore be enabled and anything indexed on the keyboard would be entered into memory, starting at the location which would presently be loaded in the Keyboard Base Register. Obviously, this could easily damage the object program or even firmware if memory is unprotected. It is therefore strongly recommended that TRACE be removed from the Utility track after use. Do not leave it in the machine.

TRACE II, COMBINED MEMORY MODIFY AND TRACE

UTILITY ROUTINE I. D. No. 1-1001-058

UTILITY ROUTINE I. D. No. 1-1101-017*

GENERAL DESCRIPTION

This Combined Memory Modify and Trace Routine provides the programmer with a very minimal Memory Modify along with a minimal trace routine.

The only capability of the normal Memory Modify Routine incorporated into Trace II is that of program word alteration. It does not permit the selective program start feature, nor does it permit the entry of alpha information.

The Trace portion of Trace II permits the macro program to be executed in its normal step by step sequence and will provide a printed record of only the location (word and syllable) of the instruction being executed and the instruction itself. Printing will be in hexadecimal format.

The Trace print-out will begin at position 81 on the scale and will occur prior to the actual execution of the printed instruction. In other words, the trace portion will show the results brought about by the execution of the instruction on the previous line. Printing will be in RED RIBBONS.

The Trace II Routine is compatible with all firmware sets and resides in the Utility Track; it cannot be resident in memory with another Utility Routine that would occupy the same space.

OPERATING INSTRUCTIONS:

Trace II is loaded using the normal Program Load procedures.

MEMORY MODIFY

The Memory Modify portion of this utility routine is entered in the same manner as is the normal Memory Modify Routine. From the Ready mode, depress PKA 3. The operating instructions are identical to those described under PROGRAM WORD ALTERATION of the Memory Modify and Selective Program Start Utility with the exception that the function associated with OCK 3 is not recognized.

The entries and/or changes made are left in memory and subsequently may be punched from memory to produce a new or corrected object program.

TRACE

CAUTION: Trace II can not be used to trace the REM instruction as it will cause the instruction to execute incorrectly, thus producing an erroneous value in the accumulator. To properly trace the REM instruction you must use Trace I.

The execution of the trace function begins, after starting the object program, by indexing any shifted numeric (except 0) at any Keyboard entry instruction in the object program. The shifted numeric digit may be entered before, between or after the normal entry. The trace portion may also be called into use while in the Ready mode by indexing a shifted numeric (except 0) prior to depressing the Start key (PKA 1). This method will enable the object program to be traced beginning with the instruction in syllable 0 of word 0.

As mentioned, the trace print-out will be the location (word and syllable) of the instruction and the instruction itself. Printing will be in hexadecimal format as shown:

5	8C	80FD
Syllable	Word	Instruction

Translated into decimal and mnemonics we have:

1	396	ADM 253
Syllable	Word	Instruction

Syllable numbers 0, 1, 2 and 3 are printed as 0, 4, 8 and C respectively for word 0:255. They are printed as 1, 5, 9 and D respectively when 256 must be added to the word location printed. The printing is 2, 6, A and E respectively when 512 must be added to the word location printed.

The trace program can be halted by indexing a shifted 0 at any keyboard entry position.

OTHER CONSIDERATIONS:

The depression of the READY button will cause the trace program to halt and the system to return to the Ready condition. Subsequently, if the Reset key was then depressed the program would return to the instruction it had just left, however the trace program would no longer be active.

TRACE III, LOCATION (IN DECIMAL) AND INSTRUCTION (IN HEXADECIMAL)

UTILITY ROUTINE I. D. No. 1-1001-059

UTILITY ROUTINE I. D. No. 1-1101-018*

GENERAL DESCRIPTION:

Trace III permits the macro program to be executed in its normal step by step sequence and in addition will provide for the printing of the location of the instruction being executed plus the instruction itself.

The difference between this trace and the limited trace option in Trace I and the tracking ability of Trace II is that the location (word and syllable) of the instruction being executed is printed in decimal format rather than hexadecimal. The instruction itself is printed in hexadecimal format.

The trace print-out will begin at position 81 on the print scale and will occur prior to the actual execution of the printed instruction. In other words, the results shown are brought about by the execution of the instruction on the previous line. Printing will be in RED RIBBON.

The Trace III Utility is compatible with all firmware sets and resides in the Utility Track; it cannot be resident in memory with another routine that would occupy the same space.

OPERATING INSTRUCTIONS:

Trace III is loaded using the normal Program Load procedures.

CAUTION: Trace III can not be used to trace the REM instruction as it will cause the instruction to execute incorrectly, thus producing an erroneous value in the accumulator. To properly trace the REM instruction you must use Trace I.

The execution of the trace function begins, after starting the object program, by indexing any shifted numeric (except 0) at any Keyboard entry instruction in the object program. The shifted numeric digit may be entered before, between or after the normal entry. Trace III may also be called into use while in the Ready mode by indexing a shifted numeric (except 0) prior to depressing the Start key (PKA 1). This method will enable the object program to be traced beginning with the instruction in syllable 0 of word 0.

As mentioned, the trace print-out will be the location of the instruction and the instruction itself. Printing will be in the following format:

1	396	80FD
Syllable	Word	Instruction

Since the location (word and syllable) is in decimal format, we read it as syllable 1 of word 396. The instruction is in hexadecimal format and hence translates to an ADM 253 instruction.

The trace program can be halted by indexing a shifted 0 at any keyboard entry position.

OTHER CONSIDERATIONS:

The depression of the READY button will cause the trace program to halt and the system to return to the Ready condition. Subsequently, if the Reset key was then depressed the program would return to the instruction it had just left, however the trace program would no longer be active.

MEMORY LIST

SELECTED MEMORY LIST

UTILITY ROUTINE I. D. No. 1-1001-016

UTILITY ROUTINE I. D. No. 1-1101-007*

GENERAL DESCRIPTION:

This Memory List Routine permits the operator to index a beginning and an ending word number and have that amount of memory printed out as a memory listing. The advantage of listing memory with this routine as opposed to using Memory Modify lies in the fact that multiple words are printed with no intervening keyboard halts. This is much faster than being limited to printing the contents of only a single word per keyboard entry.

The Memory List routine also permits the operator to type a documentation message before memory listing is performed.

This routine is compatible with all firmware sets and resides in the Utility track. It cannot be resident in memory with another Utility Routine that would occupy the same space. When used with Data Comm firmware the utility must be reloaded after executing a MACRO program.

OPERATING INSTRUCTIONS:

The Memory Listing routine is loaded into memory using the normal Memory Load procedures.

From the ready mode depress PKA 3 to start the Utility routine. The program will then halt at an Alpha keyboard instruction. Up to 145 characters may now be entered as a documentation message. Use any OCK to terminate the instruction whether or not an entry was made.

A Numeric keyboard instruction will be immediately enabled which allows the operator to enter the beginning word number. This number will not be printed. Terminate the entry with any OCK. A second Numeric keyboard instruction will now be enabled to allow entry of the ending word number. Again the number will not be printed. Terminate the entry with any OCK.

NOTE: If a beginning or ending word number greater than 511 is entered or if the ending word number is smaller than the beginning word number, an error exists. The program will sound the alarm and again enable the Numeric keyboard. Repeat the procedure described starting with the entry of a beginning word number. No error indicator light will be on and it is not necessary to use the Reset Key.

At the termination of the ending word entry, printing will begin in the following format:

012	: 0311 303C 782C 4581 :	803A 030F 383C 303A :	3831 7408 D4E0 383A :	014
Beginning Word Number	Word 12	Word 13	Word 14	Ending Word Number

On each line, the beginning word number for that line will be printed. The memory words will then be printed as four syllables (3, 2, 1 and 0) with a space between each syllable. Each line will contain a maximum of six words of memory. The ending word number will print only at the end of the memory listing.

At the completion of the selected area of memory, the program will return to the Numeric Keyboard instruction for selection of another area. If no other selections are to be made, return to the Ready mode by depressing the READY button.

SECTION 2
PAPER TAPE SYSTEM ROUTINES

MEMORY PUNCH NO. 1

UTILITY ROUTINE I.D. No. 1-1001-050
UTILITY ROUTINE I.D. No. 1-1101-014*

GENERAL DESCRIPTION

This Memory Punch Utility Routine permits the punching of selected words of memory into paper tape. Any type of Data which is stored in the words selected will be punched: Program Instructions, numeric or alpha constant, accumulations, firmware instructions, etc. The output tape may subsequently be used as input to load memory with the punched data.

Only the beginning word number is punched in a sequence of words. This provides an output tape approximately 20% shorter than a Memory Punch 2 output tape (which includes a word address with each word); however, if a portion of the tape becomes destroyed, the entire tape is then useless.

The paper tape output is in the format illustrated below. Each 16 digit word is compressed into 8 frames of tape. Each frame contains a "lower digit" (channels 1-4) and an "upper digit" (channels 5-8). This type of punch format is referred to as "compact code" or "compact Hexadecimal code". Most object program tapes will be punched in this format.

In the diagram below, "1" represents a punch (bit on) and "0" represents no punch (bit off) in the tape channel indicated.

Tape Channel Number	8 7 6 5 4 TF 3 2 1	
Bit Value of Tape Channels	8 4 2 1 8 4 2 1 0 0 0 0 0 0 0 0	
Start Code	1 0 1 0 (Block No.)	Frame 1
Word Number (0-255)	(Word Number)	Frame 2
Digit Positions of Word	1 0	Frame 3
	3 2	Frame 4
	5 4	Frame 5
	7 6	Frame 6
	9 8	Frame 7
	11 10	Frame 8
	13 12	Frame 9
	15 14	Frame 10
Stop Code	1 1 1 0 (Parity)	Frame 11
		etc.

The diagram shown represents a word of program as punched into paper tape. Only the first word in a sequence of words will contain the Start Code-Block Number frame and the Word Number frame. The Stop Code-Parity frame will be punched with every word. The stop code with the very last word in a sequence is "0 1 0 1". Parity (4 bits, as indicated above) is arrived at by exclusively OR'ing each four bit grouping as shown above with the exception of the Start Code and the Stop Code. Refer to Appendix B for a further illustration.

MEMORY PUNCH I

This Service Routine resides in the Utility Track and is compatible with all firmware sets; it can not be in memory along with any other routine which occupies the same memory space.

OPERATING INSTRUCTIONS:

1. Load Utility routine using normal load procedures.

2. Turn the Tape Perforator on. Be sure that a supply of blank tape is in the perforator. Install a roll journal in the roll journal holder and around platen.

3. From the READY MODE, depress PKA 3. (Utility Key). The machine prints "FR WD" (from word) on the journal.

4. When "FR WD" has printed, index the Memory Location of the starting word on the Numeric Keyboard. Depress any OCK. The starting word number will print on the journal.

5. The machine will then print "TO WORD" on the journal. Index the location of the "TO WORD" (This word is included in the Punch string.) and depress one of the following OCK's depending on the result desired.

OCK 2, 3, 4	Approximately 10 inches of leader tape will be punched out (sprocket only) preceding the punch string.
OCK 1	No leader tape will be punched.

   The machine will print the ending word number on the journal. Tape will be punched starting with the "FR WD", to and including the "TO WD". When punching is completed, the platen will space and "FR WD" will again print. Additional groups of memory words may be selected for punching, following the procedure as described above.

   EXAMPLE: FR WD 0 TO WD 325

6. When last punching sequence has been finished, blank trailing tape may be punched out (sprocket holes only) by indexing a "FR WD" number and a "TO WD" number so that the "FR WD" number is higher than the "TO WD" number (the reverse of normal). Then the depression of OCK 2, 3, or 4 (as in E above) will cause about 10 inches of sprocket holes only to be punched, but no word sequence will be punched.

7. When finally completed, depress the READY BUTTON to return the machine to the READY MODE.

MEMORY PUNCH NO. 2

UTILITY ROUTINE I. D. No. 1-1001-051

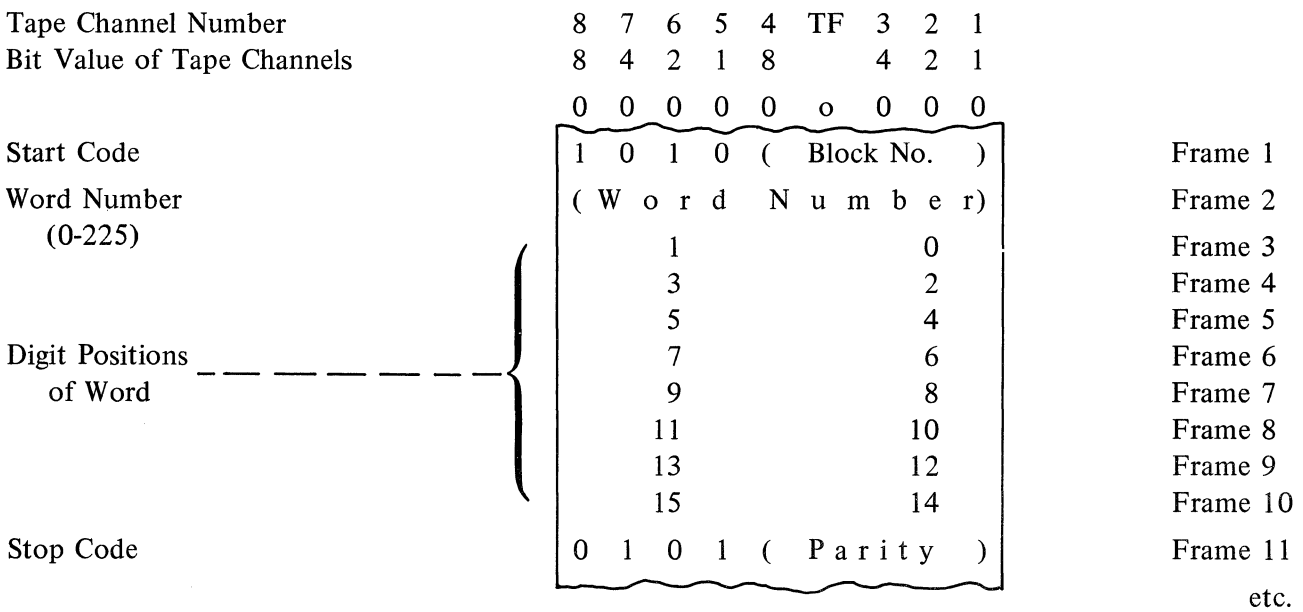
GENERAL DESCRIPTION

The Memory Punch Utility Routine permits the punching of selected words of memory into paper tape. The block number and word number is punched with every word of memory. Thus, if a portion of a tape becomes torn or destroyed, the removal of that portion permits the remaining portions to still be used, since each word is identified. This results in a slightly longer output tape (approximately 20%) than Memory Punch 1 (which punches only the beginning word number), but the recovery convenience cited above generally recommends Memory Punch 2.

Any type of data stored in the words selected will be punched: Program instructions, numeric or alpha constants, accumulations, firmware instructions, etc.

The paper tape output is in the format illustrated below. Each 16 digit word is compressed into 8 frames of tape. Each frame contains a "lower digit" (channels 1-4) and an upper digit (channels 5-8). This type of punch format is referred to as "compact code" or "compact Hexadecimal code". Most object program tapes will be punched in this format.

In the diagram below, "1" represents a punch (bit on) and "0" represents no punch (bit off) in the tape channel indicated.



The diagram shown represents a word of program as punched into paper tape. Every word will be preceded by the Start Code and Block Number frame and the Word Number frame. The Stop Code shown is different from the stop code under Punch Memory Number 1. However, the Parity is arrived at in the same way. (I. E., by exclusively or'ing each four bit group as shown above with the exception of the Start code and the Stop code.) Refer to Appendix B for a further illustration.

This routine resides in the Utility Track and is compatible with all non-datacom firmware sets. It cannot be resident in memory along with other Utility Routines that occupy the same memory space.

MEMORY PUNCH 2

OPERATING INSTRUCTIONS:

1. Load this Utility Program using normal program load procedures.

2. Turn Tape Perforator on. Be sure that a supply of blank tape is in the perforator. Install a roll journal in the roll journal holder and around the platen.

3. Depress PKA 3. The machine will print "FR WD" on the journal (From Word).

4. When "FR WD" has printed, index the Memory Location of the starting word on the Numeric Keyboard. Depress any OCK. The starting word number will print on the journal.

5. The machine will print "TO WD" on the journal (To word). Index the Memory Location of the "TO WD" on the Numeric Keyboard (i.e., the number indexed here will be the last word punched in this sequence.) Depress the following OCK's depending on the result desired:

   OCK 2, 3 or 4 — Approximately 10 inches of leader tape will be punched (sprocket holes only)

   OCK 1 — No leader tape will be punched.

   The ending word number will print on the journal. Words of memory will be punched beginning with the "FR WD" number, to and including the "TO WD" number. When the last word has been punched, the form will be spaced and "FR WD" will again print. Additional sequences may now be selected for punching following the procedures described in 4.

   EXAMPLE: FR WD 0 TO WD 292

6. When the last punching sequence has been finished, trailing tape may be punched (sprocket holes only) by indexing a "FR WD" number and a "TO WD" number such that the "FR WD" number is higher than the "TO WD" number (the reverse of normal). Then the depression of OCK 2, 3 or 4 (5. above) will cause about 10 inches of sprocket holes only to be punched. Because the word numbers were indexed in the reverse fashion, no words of memory will be punched.

7. When all punching is completed, depress the READY button to return the machine to the READY MODE.

FANCY PUNCH

UTILITY ROUTINE I.D. No. 1-1001-052

GENERAL DESCRIPTION

The Fancy Punch Utility Routine provides a mean to punch keyboard indexed characters into paper tape hole patterns that visually resemble printed characters. Its intended use is primarily for punching an identification in program or data tapes. This routine includes 4 functions.

FANCY PUNCH	Permits the operator to type to memory any alphanumeric characters and cause those characters to be punched out into paper tape in a graphic format.
TAPE DUPE	Permits duplicating the tape on the Tape Reader into a new output tape. Verify is not provided in this routine. If critical data is to be duplicated, the Multiple Paper Tape Copy, Generate, and Check Utility Routine should be used. (I. D. No. 1-1001-029-00)
RESET CODE	Permits a single code (Reset) to be punched into the output tape. When a program tape is loading through the Memory Loader device, a Reset code at the end of the tape automatically returns the system to the Ready Mode.
5 FT. LEADER	Permits 5 feet of leader tape (sprocket holes only) to be punched into the output tape.

This routine resides in words 0 through 191 and the Utility Track and is compatible with all firmware sets.

OPERATING INSTRUCTIONS

The Fancy Punch Routine must be loaded into memory using the normal Memory Load Procedures. From the Ready mode, the depression of the START PK will cause the program to stop at a Keyboard instruction with four (4) Program Keys enabled. Each PK will select a different routine.

NOTE: Use of Paper Tape Reader Load: The last word of Fancy Punch loads into the first word of the Utility Track and destroys Reader Load Utility. Thus, Reader Load must be reloaded (thru memory loader) before attempting to use it again.

PKA 1 – TAPE DUPE	To use this portion of the program, both the paper tape reader and paper tape punch must be turned on. The Tape that is to be duped is placed in the tape reader and then PKA 1 depressed. The program will dupe the tape code for code. The program is halted by depressing any Alpha Key. This will cause the program to return to the Keyboard Instruction with the same four (4) PK's enabled.
PKA 2 – RESET CODE	Depression of this key will cause a Reset code to be punched into the output tape. The Reset code will cause the Paper Tape Memory Load routine to return to the READY mode when it is read. After punching this single code the program will return to the Keyboard instruction with the same four (4) PK's enabled.

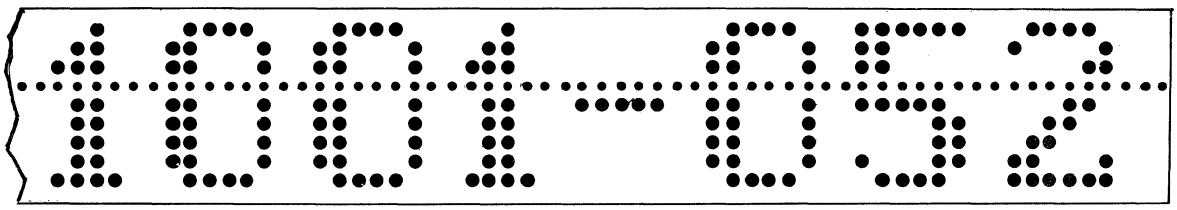
FANCY PUNCH

PKA 3 – FANCY PUNCH

Depression of this PK will cause the program to branch to a routine and stop on a Type to Memory instruction.

The operator may type into memory any character from the Alpha Keyboard. Depression of any OCK will terminate this instruction, enable the same four PK's, and will cause each character typed into memory to be punched out into the output paper tape in a graphic format. An example is shown below, where each circle represents a hole punched into paper tape.

EXAMPLE: A keyboard entry of 1001-052 would produce



PKA 4 – 5 FT LEADER

Depression of this PK will cause the program selected to punch out 5 feet of leader tape (sprocket holes only). When finished the program will return to the same Keyboard instruction with these same four (4) PK's enabled.

PAPER TAPE DUP

UTILITY ROUTINE I. D. No. 1-1001-061

GENERAL DESCRIPTION

The Paper Tape Dup Utility Routine permits the duplication of paper tape. Any code set or format may be used. The tape is duplicated frame by frame without Parity Checking.

This routine resides in the Utility Track and is compatible with all 32 track firmware sets. It cannot be resident in memory along with any other Utility Routines occupying the same memory area.

OPERATING INSTRUCTIONS

1. Load this Utility program using normal program load procedures.
2. Turn the Tape Perforator on. Be sure that a supply of blank tape is in the perforator.
3. Turn the Tape Reader on. Load the paper tape to be duplicated in the Tape Reader and depress the Read Key.
4. From the Ready Mode, Depress PKA 3 (Utility Key). The tape will be duplicated as long as paper tape is present in the Reader and Perforator. Depress the Ready button to exit the routine.

MULTIPLE PAPER TAPE COPY, GENERATE, AND CHECK

UTILITY ROUTINE I.D. No. 1-1001-029

GENERAL DESCRIPTION

The purpose of this utility routine is to reproduce any type of punched paper tape, using a Series L/TC equipped with a paper tape reader and punch, and to guarantee that the output tape is an exact copy of the master tape. The paper tape may be punched in any code set, and may contain object program, symbolic program, transaction data, or any other data, and it may be of any length. Any desired number of copies may be produced using this utility.

This routine consists of the following six functions:

1. "PKA 1 GENERATE CK TOTAL" reads the tape, generates a Check Total, and stores that total in memory. Every bit of every frame is included in the computation of this Check Total.

2. "PKA 2 SINGLE COPY-VERIFY" reads the tape, punches out an exact copy, re-generates the Check Total and verifies it against the Check Total stored in memory.

3. "PKA 3 MULTI COPY-VERIFY" permits the operator to index the number of copies to be punched, and then will read the tape, punch an exact copy, re-generate the Check Total and verify it against the Check Total stored in memory, count each copy and stop after the required number of copies are reproduced.

4. "PKA 4 READ & VERIFY" reads the input tape, re-generates the Check Total and verifies it against the Check Total stored in memory.

5. "PKA 6 INDEX CK TOTAL" permits the operator to index the Check Total to be used in the above verifications.

6. "PKA 7 PUNCH END CODE" punches the end code that is used to sense the end of the tape.

OPERATING INSTRUCTIONS

Load the Utility Program using normal loading procedures.

From the Ready Mode, depress PKA 1 "Start". Then, depress one of the following PK's depending on the type of tape to be copied and/or verified.

PKA 5 Select PKA 5 if an object program tape (compact hexadecimal format) is to be copied and/or verified.

PKA 8 Select PKA 8 if any tape other than an object tape in compact hexadecimal format is to be copied and/or verified.

After selecting the kind of tape to be copied and verified, the following six functions will be available.

PKA 1 Generate CK Total

1. Insert the master tape in the Tape Reader at any point in the leader tape and depress the Read Key. Depress PKA 1.
2. If the tape does not contain an end code, release the tape at any point in the trailer tape, then depress the Ready Button to print Check Total.
3. If the tape contains an end code, it will stop and print the Check Total automatically.

PKA 2 Single Copy-Verify

1. Insert the master tape in the Tape Reader at any point in the leader tape and depress the Read Key. Depress PKA 2.
2. If the tape does not contain an end code, release the tape at any point in the trailer tape and depress the Ready Button to verify the input tape.
3. If the tape contains an end code, the tape will stop and verify when the end code is read.
4. Verification that the generated Check Total agrees with the Check Total in memory prints as follows:

   "PKA 2 SINGLE COPY-VERIFY OK"

   If the input tape does not verify, both numbers print:

   "CK TOT XXXXXXXX"
   "INCORRECT TOT XXXXXXXX"

PKA 3 Multi Copy-Verify

1. Insert the tape coming from the punch into the Tape Reader at any point in the leader tape and depress the Read Key. Depress PKA 3 and index the number of copies desired. If the reverse entry key is depressed, system power is turned off when the required number of copies are reproduced. The entry is terminated by depressing any OCK.
2. The input tape will be read, an exact copy will be reproduced and a Check Total is generated to be verified against the Check Total in memory.
3. When the end code is read, the input tape is verified against the Check Total and if it corresponds the system prints the number of the input tape and "OK" as illustrated below. The System then continues to the next tape until the required number of tapes are reproduced.

CAUTION: If the end code provided by the Routine "PKA 7 PUNCH END CODE" is not punched in the first tape, the copy function will not terminate and an invalid verify will occur.

"TAPE NO 01 OK"

The following sequence is used to automatically reproduce and verify multiple copies of a master tape.

a. Use the "PKA 1 GENERATE CK TOTAL" routine or the "PKA 6 INDEX CK TOTAL"

routine to store the Check Total in memory to be used for verification.

b. Use the "PKA 2 SINGLE COPY-VERIFY" routine to reproduce the first tape and to verify the master tape. If the end code is not punched in the master tape then use the "PKA 7 PUNCH END CODE" routine to punch the end code in the first tape.

c. Insert the first tape in the Tape Reader at any point in the leader tape without cutting it off of the punch and depress the Read Key. Depress PKA 3 ("PKA 3 MULTI COPY-VERIFY"), then index the number of copies required; use the rev rse entry key if the power is to be turned off when the reproducing is completed. The entry is terminated by depressing any OCK. Insure that the tape travels from the punch to the reader in a path that will not let it become tangled.

   The system will halt when the desired number of tapes have been punched. As each tape is verified, the system will print a notice to the operator as illustrated in 3. Every tape punched except the last is verified.

d. To verify the last tape, cut the tape off at the punch and use the "PKA 4 READ VERIFY" routine.

If the input tape does not agree with the total in memory, both Check Totals will print, the reproducing will be discontinued and the power will be turned off if the reverse entry key was used when indexing the number of tapes required.

If a program parity error occurs or if the punch or reader runs out of tape or if a punch or reader malfunction occurs, the reproducing will be discontinued and the power will be turned off if the reverse entry key was used.

PKA 4 Read & Verify

1. Insert the tape in the Tape Reader, starting at any point in the leader tape and depress the Read Key. Depress PKA 4.

2. If the tape does not contain an end code, release the tape at any point in the trailer tape then depress the Ready button to verify the input tape.

3. If the tape contains an end code, it will stop and verify automatically.

PKA 6 Index CK Total

Index on the numeric keyboard the Check Total to be used for verification, and depress any OCK. This Check Total can now be used for verification of input tapes.

PKA 7 Punch End Code

PKA 7 should be depressed after the master tape is reproduced. The end code that is used to sense the end of the tape will then be punched into the new tape. If this end code is not included on the tape being reproduced, the copy function will not terminate automatically and an invalid verify may occur.

Other Considerations

This Utility Routine resides in words 0 to 127 of main memory, and the Utility Track. It is compatible with all 32 track firmware sets.

TAPE TO TAPE, PRINT AND/OR PUNCH

UTILITY ROUTINE I. D. No. 1-1001-019
UTILITY ROUTINE I. D. No. 1-1101-009*

This routine causes a Paper Tape to be read and permits the options of printing a listing of the contents of the input tape and/or punching a new paper tape.

The Input and/or Output tape may be in various code sets by using firmware with paper tape code translation as long as the proper code table is loaded with the firmware (Paper Tape Code sets must meet the criteria of a consistent parity scheme – odd or even – and must not exceed the character capacity of the code table). If I/O code translation firmware is not used, the I/O code set is USASCII.

The program occupies main memory words 0 through 127 and the Utility Track.

A paper tape Reader Style A 581 and a paper tape Punch Style A 562 are required.

Tape in 5,6,7 or 8 channel configuration may be utilized. Paper Tape I/O Firmware, in which both the Y and K flags are affected by the USASCII column 0 and 1 codes (respectively), should be used. The following features are provided by this utility routine:

1. Allows a description of the input and/or output tape which is being translated.
2. Any Options that are selected will be documented.
3. Single or double spacing may be selected.
4. Option of printing one record per line if a Field Identifier code sets a specified "K" flag, otherwise the standard print line is from 103 to 118 characters.
5. A Field Identifier code affecting "Y" or "K" flags may be designated to punch any code pattern desired.
6. Reads or punches any paper tape from 11/16 inch to 1 inch for which a translation table has been created.

OPERATING INSTRUCTIONS

1. The Program is loaded into memory using the normal Memory Load Procedures. From the READY mode depress PKA 1 "START". The following will print:

   TAPE/TAPE/PRINT
   I/P TB =

   The Program will stop at an Alpha Keyboard instruction for an entry of a description (up to 24 alpha characters) following = above. This entry is for documentation purposes to indicate the input code set being used and/or the type of tape being read. The entry is terminated by depressing any OCK. Then the following will print.

   O/P TB =

2. The Program will then stop at another Alpha Keyboard instruction for an entry of a description (up to 24 alpha characters) following = above, to indicate the output code set being used. The entry is terminated by depressing any OCK.

3. After the options are listed, select a PK based on the following:

PKA 2 – Punch Only –

a. In order for this utility to properly translate a tape punched in any code set, the operator must tell the L/TC what K and Y flag bit configurations to expect from the input tape, and what bit combination is to be punched in the output tape from these K & Y flag configurations. Refer to Appendix E for the appropriate values.

Enter the K Flag bit configuration set by the input code, as a single hexadecimal digit, when requested by the message "KFG TC =".

The bit configuration of the K Flag field identifier code to be interpreted is determined by its row location in column 1 on the USASCII chart. Up to 15 K Flag configurations may be entered.

When the message "OUTPUT TC =" is printed, enter the upper 4 bits and then the lower 4 bits (on two separate enabled Alpha Keyboard Instructions) of the output code you wish to be punched as a Field Identifier Code, using any OCK.

To terminate "K" flag entries, depress any OCK with no entry on message "KFLG TC =". See Example.

b. Enter the bit configuration of the Y flag (as a single hexadecimal digit) when requested by message "YFLG TC =".

The bit configuration of the Y Flag field identifier code to be interpreted is determined by its row location in column 0 on the USASCII chart. Up to 15 Y Flag configurations may be entered. Y Flag bit configurations can not be interpreted (punched into output tape) if firmware which does not affect Y flags is used.

When the message "OUTPUT TC =" is printed, enter the upper 4 bits and then the lower 4 bits (on two separate enabled Alpha Keyboard Instructions) of the output code you wish to be punched as a Field Identifier code, using any OCK. Do this after message "OUTPUT TC =" is printed.

To terminate "Y" flag entries depress any OCK with no entry on message "YFLG TC =".

Refer to example illustrating K flags, which is an identical procedure.

c. Code 00 (NUL) will be punched for any Y or K flag configuration set which has not been identified as an output code in steps a and b.

d. All other codes punched are governed by the output translation tables (when used) or by their USASCII Paper Tape Code equivalent.

PKA 3 – Print Only –

a. After message "OPTION-FORMS =" prints, double spacing may be selected by any numeric entry. For standard single spacing simply terminate enabled keyboard with any OCK.

b. Message "OPTION-PRT RECD, TC =" will print and by entering a "K" flag bit combination as a single hexadecimal digit using any OCK, a new line of print will start each time that code is encountered. Only one such code can be designated to start a "new line".

c. The option of spacing (escaping) the printer head once after each recognizable Field Identifier code may be obtained by making any numeric entry after message "SPACE=" is printed. If the spacing option is not selected, the data fields will not be separated on the print-out.

d. Printing will be in USASCII graphics and starts at position 10 on the print scale.

e. An asterisk will print for each recognizable Field Identifier code (Y or K flag combination).

f. A space will print for each input code translated to a non-graphic.

PKA 4 – Print and Punch

The options available under PKA 4 are the same as those discussed under PKA 2 – PUNCH ONLY and PKA 3 – PRINT ONLY.

PKA 5 – EOJ

a. When paper tape media is not present, select PKA 5: Message "EOJ" will print and machine will return to READY mode.

EXAMPLE:

Suppose a BCL INPUT tape has been read and documented and an USASCII output tape is to be obtained with the following BCL codes.

BCL " > " interpreted as an USASCII "DC1" (K Flag 4), and a 1,1 is to be punched in the output tape.

BCL " < " interpreted as an USASCII "DC4" (K Flag 2), and a 1,4 is to be punched in the output tape.

BCL " ≠ " interpreted as an USASCII "FS" (K Flags 3 & 2), and a 9,C is to be punched in the output tape.

No Y Flag field identifier codes are used.

To accomplish these requirements the following entries must be made after initial Documentation.

a. Select PKA 4.

b. After "KFLG TC =" prints, the entry for "DC 1" would be "1" ("DC 1" is located in Column 1 Row 1 of the USASCII Table). After "O/T TC =" prints, the entry would be 1,1 (USASCII paper tape value for "DC 1").

c. After "KFLG TC =" prints, the entry for "DC 4" would be "4". ("DC 4" is located in Column 1 Row 4 of the USASCII Table). After "O/T TC" prints the entry would be 1,4 (USASCII Paper Tape value for DC 4).

d. After "KFLG TC =" prints, the entry for "FS" would be "C". ("FS" is located in Column 1 Row 12 of the USASCII table). After "O/T TC =" prints the entry would be 9,C (USASCII Paper Tape value for "FS").

e. After "KFLG TC =" prints, no entry is made and any OCK may be depressed.

f. After "YFLG TC =" prints, no entry is made and any OCK may be depressed.

The print-out for the preceding entries would look as follows:

```
OPTION
PKA 2 – PUNCH
PKA 3 – PRINT
PKA 4 – PRINT & PUNCH
PKA 5 – **EOJ**

PKA 4 – SELECTED
KFLG TC=1 O/T TC=11
KFLG TC=4 O/T TC=14
KFLG TC=C O/T TC=9C
KFLG TC=
YFLG TC=
```

6, 7 & 8 CHANNEL PAPER TAPE CODE TABLE GENERATOR

UTILITY ROUTINE I. D. No. 1-1001-028

UTILITY ROUTINE I. D. No. 1-1101-010*

GENERAL DESCRIPTION:

This utility routine generates input and output code translation tables for 6, 7 and 8 channel paper tape code sets. The resulting code tables are used only in conjunction with paper tape I/O firmware which incorporates table look-up code translation capability (Firmware sets 2-1022-001 and 2-1002-001 fall into this catagory.). 5 Channel teletype code generation is not included in this routine.

The Code Table Generator provides for the automatic generation of four standard code sets (see Charts A and B on the last page), allows for manual alteration of an automatically generated table or provides for manually creating a completely non-standard table.

When two different standard tables (from Charts A & B) are required for input and output, the two tables may be generated simultaneously.

EXAMPLE: Input table in Standard BCL code & output in Standard USASCII code.

When either the input table or output table is non-standard or partially standard and the other table is standard, each table must be generated in a separate pass of the generator program.

EXAMPLE: BCL Input Tape & Friden Output Tape.

When both input and output non-standard or partially standard tables are required, they must correspond exactly if they are both to be constructed during the same pass of the generator program. If they do not correspond for every code, each table must be generated in a separate pass.

The completed tables can be punched into paper tape, in object program tape format, or they can be stored directly in the proper area of memory as required by the particular firmware set being used.

This utility requires words 0 to 383 of memory, and will operate with any paper tape I/O firmware which provides this amount of user memory. However, care should be taken to insure that the tables which are generated by this program do not overlay part of firmware or part of this utility program. It is recommended that this utility be used with the firmware for which the translator tables are being generated or with a firmware set which provides for 512 words of user memory.

OPERATING INSTRUCTIONS:

Load the Utility program tape by any of the standard program load procedures. Insert a journal roll which accommodates the 5-1/2" print area (from scale 10 to 65). From the Ready Mode, depress PKA 1 – START.

Before the actual table can be generated, the operator must specify certain options which determine the type of table to be generated.

A. Table Definition

1. When "TRACK" prints, a track number must be entered. Terminate the entry using any OCK. The track number in macro memory which may be entered is 02 through 15, and must be in accordance with the requirements of the selected firmware set with which the table will be used. (Consult the Group II Software Library Catalog for the Program Abstract of the selected firmware set to determine the track location of the code table.)

   If a non valid track number is entered, "ERROR" is printed in the remarks area of the print form, and the system will halt on a keyboard instruction waiting for a valid track number entry.

2. When "PUNCH" prints, an entry of Y or N (Yes or No) must be made. Terminate the entry using any OCK. If the table is to be punched at end of job enter Y, otherwise enter N. If an entry has not been made or some character other than Y or N has been entered, "ERROR" will print in the remarks area of the form, and the system will halt on a keyboard instruction waiting for a valid entry.

3. When "STNDRD" prints, an entry of Y or N (Yes or No) must be made using any OCK. If a standard table is desired, enter Y. If a partially standard table is desired, enter a Y. The options available to make the necessary changes to the standard table will be entered in the section labeled "GENERATION OF TABLES".

   If a standard or partially standard table is not required, enter N. If an entry has not been made or some character other than Y or N has been entered, "ERROR" will be printed in the remarks area, and the system will halt on a keyboard instruction waiting for a valid entry.

4. When "INPUT" prints, an entry must be made to indicate if an input table is required. The entry depends upon the selection of standard or non-standard in step 3 above.

   Standard Input Table Entries:

   A numeric entry of 0 to 4 must be made using any OCK, to indicate if an input table is required. (See Chart A for meaning of codes). If the entry is not valid "ERROR" will print and the system halts on a keyboard instruction waiting for a valid entry.

   Non-Standard Input Table Entries:

   An entry of Y or N (Yes or No) must be made using any OCK. If the table is to be used to convert the Non-Standard tape code to internal USASCII code enter Y. An entry of N informs the system that an input table is not to be generated. If an entry has not been made, or if a character other than Y or N has been entered, "ERROR" will print in the remarks area of the form, and the system will halt on a keyboard instruction waiting for a valid entry.

5. When "OUTPUT" prints, an entry must be made to indicate if an output table is required. The entry depends upon the selection of standard or non-standard tables in step 3.

Standard Output Table Entries:

A numeric entry of 0 to 4 must be made, using any OCK, to indicate if any output table is required, and if so, which table is required. (See Chart B) If the entry is not valid "ERROR" will print, and system halts on a keyboard instruction waiting for a valid entry.

Non-Standard Output Table Entries:

An entry of Y or N (Yes or No) must be made using any OCK. If the table is to be used to convert internal USASCII to a non-standard output tape code the entry is Y. An entry of N results in the output table not being generated. If the entry is some character other than Y or N, "ERROR" will print in the remarks area of the form, and the system will halt on a keyboard instruction waiting for a valid entry.

If an entry is valid, a check is made to determine if an entry was made for Input, Output or both. (It is possible to enter a valid N entry for both Input and Output and have an invalid situation where neither an input nor an output table has been specified): If an N entry has been made for both Input and Output, "ERROR" is printed in the remarks area of the form, and the Input/Output sequence is restarted.

6. If a non-standard table has been selected, "EVN PR" (Even Parity) prints and the type of parity scheme must be entered.

   An entry of Y or N (Yes or No) must be made using any OCK. If the table to be generated is to provide even parity, e. g. for a paper tape, enter Y, otherwise enter N. If an entry has not been made, or some character other than Y or N has been entered, "ERROR" will print in the remarks area, and the system will halt on a keyboard instruction waiting for a valid entry.

B. Generation of Tables

If valid entries have been made in the previous sections, the following PK's will now be enabled:

PKA 5 – The USASCII character desired internally can be entered as a keyboard charcter, if it is represented by one of the 64 graphic character keys.

PKA 6 – The USASCII character desired internally must be entered in hexadecimal format, if it is not represented by one of the 64 graphic character keys.

PKA 7 – If the required table is complete.

At this time either PKA 5, PKA 6 or PKA 7 must be depressed.

CAUTION: Once PKA 6 has been selected, it is impossible to return later and select PKA 5.

PKA 5 – If PKA 5 is selected, "KEYBOARD" prints.
The routine enters the mode in which it asks first for the "ASCII:" code and then for the corresponding "TAPE CD:"

"ASCII:"
The desired internal code is entered first, as one of the 64 USASCII graphic characters. Any entry from the alpha keyboard is allowed. The character is indexed and entered with any OCK.

"TAPE CD:"

Entry is now made for the paper tape code which is to be interpreted during input as the above listed USASCII internal characters, or during output to be produced from the above listed USASCII internal character. The entry is made as 2 hexadecimal digits (0 to 9 & A to F) on the alpha keyboard, which represents the binary value of the 8 tape channels (see examples). If an invalid alpha character is indexed, the error indicator is turned on and the RESET Key must be depressed to correct the error condition.

When constructing an Input Table, the entry of the same tape code more than once will replace the prior entry for that tape code, thus permitting correction when an error has been recognized. Likewise, when constructing an Output Table, the entry of the same internal code more than once will replace the prior entry for that internal code.

If the entry is valid, PKA 6 and PKA 7 are enabled. To continue in the Keyboard mode with the next ASCII code entry, depress any OCK. Otherwise, depress PKA 6 or 7 for the following functions.

PKA 6 – Provides for describing the desired internal code in hexadecimal format if it is not one of the 64 graphic characters. The sequence is identical to the operation of PKA 5 except that the USASCII characters desired internally must also be entered as 2 hexadecimal digits (only 0-9 and A-F). The error sequence is identical to that previously defined.

PKA 7 – This is the End-of-Job indicator.
If PKA 7 is selected, the program will store the table into memory, print each table word (with word number) and if the punch option was exercised, punch the table into paper tape.

Each non-standard table will be constructed so that any code not entered to the generator will, if read, cause a parity error to be generated by the firmware and a question mark (?) to be inserted in place of the code.

Example 1:

Suppose in an input table, a BCL tape code of 2,A (BCL ≠ expressed hexadecimally) is to be interpreted as a USASCII "___" (Underline – Col. 5, Row F).

The tape code, as it appears on the input tape including parity, would have the following format:

Channels:	8	7	6	5	4	o	3	2	1
Bit Values:	8	4	2	1	8		4	2	1
Tape Code:			●		●	o		●	

A punched hole in the tape is considered as bit on (1) while no hole in the tape is considered as a bit off (0).

The above tape code, expressed in 8 bits will then appear:

Channels:	8	7	6	5	4	3	2	1
Bit Values:	8	4	2	1	8	4	2	1
	0	0	1	0	1	0	1	0
			2				A	

Thus, the operator makes the following entry:

if: (PKA 5) "ASCII:" ——— "TAPE CD:" 2 A

if: (PKA 6) "ASCII:" 5 F "TAPE CD:" 2 A

Result if Input Table: Tape code of 2 A is translated to internal character "—" during input.

Result if Output Table: Internal character "—" is translated to tape code 2 A during output.

NOTE:

A. The manner in which Tape Code references a position in the Input Table and the resulting stored value for internal character is determined as follows:

The upper 4 bits (5, 6, 7 and 8) of the tape code are used as the word address in the table while bits 2, 3, and 4 are used to locate the character position in the word.

Channels:	8	7	6	5	4	3	2	1
Bit Values:	8	4	2	1	4	2	1	
	0	0	1	0	1	0	1	0
	Table Word Address				Character Address			Table Parity

The low order bit (b1) is used in parity checking in the tape input firmware. If the low order bit of the tape code is on (1) then the high order bit (8) of the code stored in the table must also be on (1); If the tape bit is off (0) then the table code high order bit must also be off (0). If these conditions are not true, the input firmware will cause the Invalid Code flag to be set.

According to the example given above, the word address would be "2" while the character address in the word would be "5".

The desired internal USASCII code entered on the Alpha Keyboard would be stored in the spot in the table selected by the tape code. In the example above, a hexadecimal value of 5,F (USASCII "—" Underline) would be placed in word 2 character 5 of the input table. In bit format it would be as follows:

Character bit Positions:	8	7	6	5	4	3	2	1
Bit Value:	8	4	2	1	8	4	2	1
"—"	0	1	0	1	1	1	1	1
			5				F	

Since the low order tape bit is off, bit 8 (8) of the code placed in the table will also be off. Hence in this example, the table code would be that as shown directly above this paragraph.

Suppose in an input table, a BCL tape code of 1,F (BCL " ≥ " expressed hexadecimally) is to be interpreted as an USASCII " ' " (apostrophe).

The tape code, as it appears on the input table including parity, would have the following format:

Channels:	8	7	6	5	4	o	3	2	1
Bit Values:	8	4	2	1	8		4	2	1
Tape Code				●	●	o	●	●	●

The above tape code expressed in 8 bits will then appear:

Channels:	8	7	6	5	4	3	2	1
Bit Values:	8	4	2	1	4	2	1	
	0	0	0	1	1	1	1	1
	Table Word Address				Character Address			Table Parity

In this example, the word address would be "1" and the character address in the word would be "7" (1 is not included in the character address).

In this example the value of A,7 (USASCII " ' " 2,7 plus 8 would be turned on in the value due to tape Channel 1 being on, making the internal code 10,7 or A,7) would be stored in character 7 word 1 of the input table, expressed hexadecimally.

In bit format it would be as follows:

Character Bit Positions:	8	7	6	5	4	3	2	1
Bit Value:	8	4	2	1	8	4	2	1
	1	0	1	0	0	1	1	1
			A				7	

In the translation of each character, after firmware has checked tape channel 1 with bit 8 in the proper input table character position, bit 8 is of no further use and is disregarded. Bits 1 to 7 only are used internally. In the above example, the 8 bit would be disregarded after parity check, making the actual translation 2,7 (USASCII " ' " expressed hexadecimally.)

B. The manner in which the Internal code references a position in the Output Table and the resulting stored value for punching is determined as follows:

If an output table is also called for in the same pass in which the input table is being created, tape codes that are interpreted to certain USASCII internal codes in the input table will have the process automatically reversed in the output table.

As an example, if the input table causes a tape code of 2,A (BCL " ≠ " expressed Hexadecimally) to be interpreted as an internal USASCII "—", then in the output table, the internal USASCII "—" will cause a 2,A (hexadecimally) to be punched into the output tape.

The value of the internal low order bit is inserted in the high order bit position. The 4 high order bit positions (8, 7, 6, & 5) determine the word address which the character references. Bits 4, 3, & 2 determine the character address in the word. An internal USASCII "—" (5,F expressed hexadecimally) would reference word 13 character 7 of an output table.

Internal "—" (5,F) to punch BCL " ≠ " (2,A)

Internal Code:	Upper Digit	Lower Digit
	0 1 0 1	1 1 1 1
	(5)	(F)
Insert bit 1 in bit 8	1 1 0 1	1 1 1 1
Bit Values	8 4 2 1	4 2 1
	Word (13)	Character (7)

The value placed in a character position in the table determines the tape code to be punched. The binary value of the upper digit punches channels 8, 7, 6, 5, the value of the lower digit punches channels 4, 3, 2, 1.

Character 7 of word 13 of the output table would contain 2,A.

If specific changes are to be made in the output table, exclusive of any input table function, then the output table must be generated in a separate pass from any input table generation.

C. Incorporating Input/Output Tables with Firmware and/or Object Programs

There are three methods in which the input and/or output tables generated by this routine can be incorporated into an object program or incorporated into firmware.

1. Combine tables with firmware.
   Load the L/TC 500 with the firmware set which is to be used with the tables, then load the I/O tables. Next, load the Memory Punch 2 Routine into the Utility Track and punch out a new firmware set which includes the I/O Table. Then have a Field Engineer Memory Protect the track containing the I/O tables as well as the firmware tracks.

2. Keep the I/O tables on a separate tape and have the Field Engineer Memory protect the tables after they are loaded.

The two methods above are recommended when all applicational programs utilize the same I/O Tables.

3. If more than one set of I/O Tables is required by the applicational programs, the I/O tables can be combined with the application program using the memory Punch 2 Utility Routine. In this case, the track containing the I/O Tables must remain unprotected as does the user area of memory.

CHART A

Standard Input Table

Code	Table
0	None required, reader not used
1	USASCII to USASCII
2	BCL to USASCII
3	NCR (Britain) to USASCII
4	ICT i 500 to USASCII

CHART B

Standard Output Table

Code	Table
0	None required, punch not used
1	USASCII to USASCII
2	USASCII to BCL
3	USASCII to NCR (Britain)
4	USASCII to ICT i 500

Refer to Appendix F for details of above tables.

MEMORY LOAD – PAPER TAPE READER

UTILITY ROUTINE I.D. No. 1-1001-055

UTILITY ROUTINE I.D. No. 1-1101-003*

GENERAL DESCRIPTION

The Reader Memory Load Utility Routine permits the loading of an object program paper tape with the Paper Tape Reader. The program tape must be in compact hexadecimal format, as illustrated in Memory Punch 1 and Memory Punch 2.

This routine permits the object program to be loaded at the rate of approximately 40 codes per second whereas the regular memory load routine will load the program at the rate of 15 1/2 characters per second.

This routine resides in the Utility Track and is compatible with all Firmware sets.

OPERATING INSTRUCTIONS

1. The Utility Program is loaded using the normal memory load procedures.
2. Turn the Tape Reader on and load the object program paper tape to be read. Depress Read Key on Tape Reader. From the Ready Mode, depress PKA 3 (Utility Key). The tape will read and the object program loaded into memory. The loading will stop when the tape has been completely read.
3. Depress the ready button to exit the routine.

MEMORY LOAD BOOTSTRAP – PAPER TAPE READER

UTILITY ROUTINE I. D. No. 1-1001-056

The Paper Tape Reader Load Bootstrap Utility program is provided so that firmware may be loaded into a Series L/TC through the Paper Tape Reader rather than through the memory loader. The advantage becomes the difference between the Memory Loader reading speed (15 1/2 characters per second) and the Paper Tape Reader speed (40 characters per second).

This routine is primarily intended for use by Field Engineering Personnel and therefore presumes that the user is familiar with normal Memory Loader Firmware loading procedures.

This utility must be reloaded for each use.

The Paper Tape Reader Load Bootstrap program resides in the Utility Track.

OPERATING INSTRUCTIONS

It is extremely important that the following steps are followed exactly to insure that the firmware will be loaded correctly.

1. The appropriate tracks of memory must be unprotected (read/write) and Series L must be put in the "Tape to Q" mode (FE Jumper on pads C/D on FE1 circuit card). This should be done while the machine is in an "off" condition. Then, turn on the system and allow 40 seconds before implementing the next step.

2. Turn the Memory Loader switch to "on" and feed the leader of the Reader Load Bootstrap tape into the Memory Loader. Before the first code is read from the tape, turn the Memory Load switch to "off", depress the "Initialize" button once and then turn the memory load switch "on". After the bootstrap has loaded, the system is in an idle state – with the error light and the option D Indicator Lights on.

3. At this point, the system "clock" is stopped to allow the operator to put the machine in the "Normal" mode (change FE jumper from pads C/D to A/B on FE1 circuit card).

   If an FE2 circuit card is not used, it is necessary to hold the "Program Halt" button depressed while the machine is put in the "Normal" mode.

   If an FE2 circuit card is used, the "Program Halt" button must be depressed after the machine is put in the "Normal" mode.

4. Place the firmware tape on the paper tape reader and Depress the READ Key. The firmware tape should now start loading into memory. As an indication that the bootstrap routine is functioning properly, the PK indicator lights should be counting in a binary fashion as each word of memory is loaded (both word number and block number). If the error light indicator turns on while loading the firmware tape, an error has occurred and the entire loading sequence must be restarted. After the firmware tape has loaded, depression of the program halt button will return the System to ready mode.

MEMORY LOAD & TRANSLATE – B5500/B300 ASSEMBLY OUTPUT

UTILITY ROUTINE I. D. No. 1-1001-014

UTILITY ROUTINE I. D. No. 1-1101-004*

GENERAL DESCRIPTION

If the option of punched paper tape object code is specified for the Series L/TC Assembler IV (B5500 Version) and Assembly V (B300 Version), it is punched in USASCII paper tape code. In this format, the object program must be read into Series L/TC memory using this routine.

This routine will convert the USASCII format code into compact hexadecimal and load the converted code into memory and/or punch an output paper tape in compact hexadecimal format (which can be used subsequently to load the object code).

Style A 581 Paper Tape Reader and Style A 562 Paper Tape Punch are required.

This Utility is compatible with any firmware set having paper tape I/O capabilities.

OPERATING INSTRUCTIONS

1. Load the utility program, via the standard memory load procedure. When the memory load is complete, depress reset to return the system to the READY mode. This will enable PKA 1, 2 and 3.

2. Mount the Assembler IV/V USASCII output tape on the paper tape reader and depress the Read Key.

3. From the Ready Mode, depress PKA 3. The program will halt on a keyboard instruction, and the following options may be exercised:

   OCK 1 & 2 – Conversion, memory load and punched paper tape output.

   OCK 3 – Conversion and memory load only.

   OCK 4 – Conversion and punched paper tape output only.

4. To exit the routine, depress the Program Halt button.

CAUTION: If forms are left in the forms handler during the execution of this utility program, special consideration should be given to the forms alignment. This utility program will cause the platen to space up several spaces upon closing the handler after returning to READY mode.

Refer to APPENDIX B for the USASCII object program tape format.

SECTION 3

80 COLUMN CARD SYSTEM ROUTINES

MEMORY LOAD – CARD READER

UTILITY ROUTINE I. D. No. 1-1001-054 (Non Data Comm Firmware)

UTILITY ROUTINE I. D. No. 1-1001-064 (Data Comm Firmware)

GENERAL DESCRIPTION:

This Memory Load Utility Routine permits the loading of object programs from 80 Column Cards.

This routine must be resident in the Utility Track to permit use of the LCD instruction.

A "Hash Total" of all cards read is stored in the accumulator. This "Hash Total" and the Program Identification (ID) may be printed out at the end of the load routine if the operator desires.

Program Cards are loaded into memory at the rate of approximately 100 cards per minute. From 1 to 8 words of object program may be stored on each card. Refer to Appendix D for the program card format.

This Routine resides in the Utility Track and cannot be resident in memory along with any other Utility Routine that would also occupy the same space.

OPERATING INSTRUCTIONS

1. The Card Memory Load routine must be loaded into memory using the normal Memory Load Procedures.

2. From the READY mode, depress PKA 3. If the program deck has already been placed in the Card Reader supply hopper and the RESTART button depressed, the cards will be read and loaded into memory. When the last card is read and the supply hopper is empty, the Reader Condition Indicator on the Series L will be turned on. The operator then has the following choices:

   a. If more cards are to be loaded, place them in the hopper and depress RESTART Key.

   b. If the Program ID and Hash Total are to be printed, depress any key on either keyboard. The Hash Total will print followed by the Program ID and then the machine will return the READY mode.

   c. Depress the READY BUTTON and the machine will return to the READY mode immediately without printing either the Hash Total or the Program ID.

NOTE: 1) This routine destroys the contents of the Accumulator.

2) The program ID is stored in word 01 during loading. Thus, if program data must be loaded in word 01, that program card must be placed as the last card in the deck.

MEMORY LOAD AND TRANSLATE BCL – CARD READER (B5500/B300 ASSEMBLER OUTPUT)

UTILITY ROUTINE I. D. No. 1-1001-021

UTILITY ROUTINE I. D. No. 1-1101-005*

GENERAL DESCRIPTION:

This Utility Routine permits memory to be loaded from BCL object program cards created on a B5500 or B300, or permits these object program cards to be read and new cards punched in the Series L MEMORY LOAD FORMAT (compact hexadecimal). Refer to Appendix D for a description of the card formats.

A Card Reader Style A 595 is required. A Card Punch Style A 149 is required only if the translation from BCL to Series L hexadecimal card format is needed.

This Utility Routine occupies the Utility Track and cannot be resident in memory along with any other Utility Routine that would occupy the same memory space. Words 192 to 223 (Track 6 Block 0) of main memory are also occupied by this utility. If the Memory Load option is used, Track 6 will be overlayed by any program being loaded into this part of Memory. When this routine is used with Data Comm firmware it must be reloaded for each use.

OPERATING INSTRUCTIONS

1. Load the Utility Routine using the normal load procedure. From the Ready Mode, depress PKA 3. The Program will stop on a Numeric Keyboard instruction. Insert the BCL Card Deck in the Card Reader. The following choices are available.

   OCK 1, 2 or 3 – MEMORY LOAD

   OCK 4 – PUNCH IN L CARD FORMAT

   A card with a zero punched in card column 12 will cause the routine to halt and the machine to return to the READY MODE.

2. If the Card Punch is used, a card must be positioned under the Read Station as well as the Punch Station. The print switch should be in the off position. Set the Program Select switch to P1 or P2.

3. The card with the zero punch in row 12 will be created by the B 5500 or B 300 Assembler. If it is lost, it should be replaced before running this utility program. A manual return to the READY MODE may cause loss of data.

MEMORY PUNCH – 80 COLUMN CARD

UTILITY ROUTINE I. D. No. 1-1001-053

GENERAL DESCRIPTION:

This Memory Punch Utility Routine permits the punching of selected words of memory into 80 column cards. Any type of Data which is stored in the words selected will be punched: Program instructions, numeric or alpha constants, accumulations, firmware instructions, etc. The output cards may subsequently be used as input to load memory with punched data.

The Memory Punch Routine provides the ability to punch up to 6 Characters of Program Identification in each program card. Each program card is self loading in that the beginning word number, block number and number of words is contained on each card. Up to 8 words of program can be punched on each card. Refer to Appendix D for the exact Series L card format.

This routine resides in the Utility Track (Track 2 Block 2 beginning with word 578). It cannot be resident in memory along with any other Utility Routine that also would occupy the same memory space.

OPERATING INSTRUCTIONS

1. From the READY mode, load the Memory Punch-Card Utility Routine using the normal memory load procedures.

2. Turn Card Punch on. Be sure that a supply of blank cards is positioned in the Card Punch and that a blank card is in both the Punch and Read stations. Install a roll journal in the roll journal holder and around the platen. The Program Select Key should be set at P 1 or P 2.

3. Depress PKA 3. The machine will print "ID" (Program Identification) on the journal. Enter up to 6 Alpha characters for the Program Identification. Any OCK may be used for termination.

4. The system will then print "~~FR WD~~" on the journal and halt at a numeric keyboard instruction. Index the memory location of the first word in this sequence to be punched on the Numeric Keyboard. Use any OCK. The starting word number will print on the journal.

5. The system will then print "~~TO WD~~" on the journal and halt at another numeric keyboard instruction. Index the last word of memory that is to be punched in this sequence. Depress any OCK. The last word number will print on the journal.

   Program Punch Cards will be created according to the card format described previously. The Program ID will be punched into card columns 1 through 6 of each card.

6. When the last word in the sequence just entered has been punched, the journal will be spaced and the program will halt at a numeric keyboard instruction. A new sequence may not be punched (Step 4).

7. When all punching is completed, depress the READY button to return the machine to the READY mode.

CARD TO CARD

CARD TO CARD PRINT AND/OR PUNCH

UTILITY ROUTINE I. D. No. 1-1001-020

UTILITY ROUTINE I. D. No. 1-1101-006*

GENERAL DESCRIPTION:

This Utility Routine permits a deck of cards, punched in BCL Code, to be read with the options of printing a listing of the contents of the cards and/or punching a new card deck. Refer to Appendix D for the exact BCL card format.

A deck of EBCDIC punched cards may be used if the card codes are the same as BCL card codes or if the I/O code translation tables have been modified prior to implementing this Utility Program. (Any such modification of the I/O code table must be within the confines of the table structure – Refer to GP 300 Card I/O Instructions and code table description.)

This Utility may also be used to reproduce Series L object program decks (compact hexadecimal format); however a listing of these cards is not usable.

A Card Reader Style A 595 and a Card Punch Style A 149 are required.

This Utility Routine occupies words 0-56 of main memory and therefore must be used with card I/O Firmware sets which provide this amount of user memory.

OPERATING INSTRUCTIONS

1. If a listing is desired, either a roll journal or continuous marginally punched forms providing a print line width of 10 inches must be provided. The first print position will be Position 10 on the print scale.

2. After loading the program, use PKA 1 (START) from the READY MODE. The program will print out the following:

   CARD/CARD/PRINT

   The form will then space twice and the following will print:

   PKA 1 – REPRODUCE

   PKA 2 – LIST

   PKA 3 – LIST/REPRODUCE

   If PKA 2 or 3 is selected (calling for a Listing) the following will also print:

   OCK 1 – SINGLE SPACE

   OCK 2 – DOUBLE SPACE

   The form will then be spaced to a new page, and listing and/or reproducing will begin.

3. The last card in the deck must have "END" punched in columns 1-3. When this "END" card is read, the program will print the number of cards that were processed and then return to the READY state. If the "END" card is not present, the program can be returned to the READY state but the number of cards processed will not be printed.

SYMBOLIC CARD PUNCH

UTILITY ROUTINE I. D. No. 1-1001-027

UTILITY ROUTINE I. D. No. 1-1101-008*

GENERAL DESCRIPTION

The Symbolic Card Punch Utility program permits the punching of an assembler symbolic card deck, and utilizes a Series L and an A 149 Card Punch. The Symbolic Card Format punched corresponds to the card format described by the Burroughs Assembler Coding Form (Series L). Diagnostic functions are included to insure the validity of most entries.

As an option, 80 column cards may be punched in free form.

This program resides in words 0-400 of main memory and therefore must be used with card I/O firmware sets which provide this amount of user memory.

OPERATING INSTRUCTIONS

Load this Utility Routine using the normal loading procedures. The Card Punch must be turned on and a card registered in the Read Station and the Punch Station. Set the Program Switch on the Card Punch to P 1 or P 2. From the READY mode, depress PKA 1 (Start PK). The program will stop at an Alpha Keyboard entry position.

A. Program ID

Enter 0-6 characters for the program ID and terminate with any OCK. The ID is printed on the journal beginning with Position 70 and is punched into the first output card and then automatically duped into each succeeding output card. The program will halt on a Numeric Keyboard instruction.

B. Sequence No

"Sequence No." is the start of the main program and the point to which the program will return following the completion of each card.

The following options are available:

PKA 3 – Will print a heading on the journal showing each column from 1 thorugh 80 in the following format:

1. . . . . .10 . . . . . 20 . . . . . (etc) . . . . . 80 . .

After the heading is printed, the journal is spaced and the program returns to step B. for another selection.

PKA 2 – This PK selects a routine that enables an Alpha KB instruction to provide entry of 80 columns of free format. After entering from 1 to 80 characters, termination with OCK 1, 2 or 3 will cause the card to be punched. Characters indexed are printed under the appropriate card column number in the heading (printed from use of PKA 3 described above). This portion of the program causes a blank card to be released following the punching of the free form card. This free form routine may be selected at any time, even between punching of symbolic cards, without having any effect on the symbolic program.

Initializing Sequence Numbers

The system will provide a starting sequence number of 10 if another number (maximum length of 5 digits) is not indexed. The sequence number will be right justified and printed under card columns 11-15 of the heading. The number will be punched into the card right justified also in columns 11-15. Following sequence numbers will be in increments of 10. Terminate the entry using OCK 1, 2 or 3.

Changing Sequence Number

If a change in sequence number is desired, index the new sequence number and terminate with OCK 1, 2 or 3. New sequencing will then be in increments of 10 from this new base.

OCK 4 – Use of OCK 4 in terminating a sequence number entry causes "ERROR" to print to the left of the sequence number; The forms advance and the program returns to Step B.

OCK 1 – Termination of the sequence number entry with OCK 1, 2, or 3 causes the program
2 & 3 to advance to an ALPHA keyboard instruction for Label entry.

C. Label

The Alpha Keyboard is enabled and the printer is positioned under column 16 of the heading. From 0 to 6 Alpha characters may be entered. The Label entry is terminated by one of the following OCK's.

OCK 1 – Allows the program to continue to the MNEMONIC entry described below. If an invalid first character is entered (based on label rules), a re-try is permitted.

OCK 4 – Causes "ERROR" to print to the left of the sequence number. The form advances and the program returns to Step B.

D. Mnemonic

The Alpha Keyboard is enabled and the printer is positioned under column 22 of the heading. From 1-5 characters may be entered representing a Mnemonic instruction. The Mnemonic entry is terminated using one of the following OCK's:

OCK 1 – Mnemonic entered will be verified. Invalid mnemonic will cause word "ERROR" to
2 & 3 be printed adjacent to last character of mnemonic, form will space and return for re-entry of mnemonic. Type of mnemonic will determine next entry which is usually the A PARAMETER FIELD. Exceptions to this are the following Mnemonics which will cause the program to stop in the FIELD LENGTH Entry position.

PKA, PKB, NUM: ALF: MASK

OCK 4 – Causes "ERROR" to print to the left of the sequence number; The form advances and the program returns to Step B.

E. Field Length

The Numeric Keyboard is enabled for entry of 1 or 2 digits representing field length. The Field Length entry is terminated using one of the following OCK's:

OCK 1 – Use of OCK 1, 2 or 3 will cause the program to advance to the A PARAMETER
2 & 3 FIELD entry.

OCK 4 – Causes "ERROR" to print to the left of the sequence number, the form advances and the program returns to Step B.

F. Parameters

Either the Alpha Keyboard or the Numeric Keyboard will be enabled depending on the Mnemonic entered in step D.

1. LABEL REQUIRED

Label Entry – The operator must make an entry of from 1 to 6 characters. If no entry is made, an error condition exists. Label rules apply. The A PARAMETER entry is terminated by one of the following OCK's.

OCK 1 – Allows an INCREMENT entry if mnemonic permits, otherwise
2 & 3 the program advances to REMARKS.

OCK 4 – Causes "ERROR" to print to the left of the sequence number, the form advances and the program returns to Step B.

Numeric Entry – If a character is not entered, the use of OCK 1, 2 or 3 allows a numeric entry based on the mnemonic. After the Numeric Entry, the use of OCK 1, 2 or 3 will allow a 2nd Numeric entry if permitted. If a 2nd Numeric entry is not permitted the program advances to REMARKS.

OCK 4 – Causes "ERROR" to print to the left of the sequence number, the form advances and the program returns to Step B.

2. NO LABEL PERMITTED

Numeric Entry – Enter Numeric Entry. OCK 1, 2, or 3 – will stop for a numeric 2nd Numeric entry if permitted. If a 2nd numeric entry is not permitted the program advances to REMARKS.

OCK 4 – Causes "ERROR" to print to the left of the sequence number, the form advances and the program returns to Step B.

Alpha Entry – Same as the numeric entries except that no diagnostics are provided and a different keyboard is enabled.

3. In all cases, the print head will be positioned under the proper column number if the heading option in Step B. was used to print column numbers.

4. Certain Mnemonics are exceptions insofar as the A Parameter field is concerned. These are:

NUM – The A Parameter has two entries. The first entry is an Alpha Keyboard entry for sign position flags (-SCM).

OCK 1 – The system advances to the second entry position for a Numeric Keyboard
2 & 3 Entry.

OCK 4 – Causes "ERROR" to print to the left of the sequence numbers, the form advances and the program returns to Step B.

The Second entry is a Numeric Keyboard entry permitting from 1-15 digits to be entered. The Second field must have at least 1 digit; therefore, if NUM is to have a sign entry, the field length must be a minimum of 2.

OCK 1 – The system advances to REMARKS
2 & 3

OCK 4 – Causes "ERROR" to print to the left of the sequence number, forms will space and the program will return to Numeric KB described under Step B.

ALF – Enter 1-24 characters for a single symbolic card. The maximum field length on the ALF mnemonic is 99.

OCK 1 – The system advances to REMARKS.
2 & 3

OCK 4 – Causes "ERROR" to print to the left of the sequence number, forms will space and the program will return to Numeric KB described under Step B.

PKA 5 – Terminating the ALF entry in this manner will stop any further entry. In this case the number of characters entered must be equal to or greater than the Field Length entry. (Maximum number of characters must be entered for each card, otherwise an incorrect assembly will occur). If Field Length is exceeded, "ERROR" will print to left of sequence number and card will be punched.

After entering the remarks for a given symbolic line, the program will return and print the continuation code "CC" for another line until the ALF entry has been satisfied.

MASK – Same as ALF, except that the A parameter Field entry may not be terminated with PKA 5.

There are no diagnostics on ALF, MASK and the Alpha portion of the NUM.

G. Remarks

The Alpha Keyboard is enabled for entry of from 0-25 characters.

OCK 1 – Card will be punched and program will return to Numeric Keyboard entry under Step B.
2 & 3 (SEQUENCE NO.)

OCK 4 – Causes "ERROR" to print to the left of the sequence number, forms will space and the program will return to Numeric KB described under Step B.

OBJECT CARD TO OBJECT PAPER TAPE CONVERSION

UTILITY ROUTINE I. D. No. 1-1001-065

GENERAL DESCRIPTION:

The purpose of this routine is to convert 80 column object program cards into paper tape in standard object code tape format. Input cards may be in either BCL object code (B 5500/B 300 output) or compact hexadecimal code (L 2000/B 3500/B 300 output). Refer to Appendix D & E for the Card Formats.

The card to tape conversion program is compatible with the following firmware sets:

2-1004-001	GP 300 with Card I/O
2-1005-002	GP 300 with Card In, Tape Out
2-1024-001	GP 300 with Data Comm & Card I/O.

OPERATING INSTRUCTIONS:

The card to tape conversion program is loaded into memory using the normal Memory Load procedures.

Load the 80 column object cards into the card reader. Be sure to check the tape supply in the paper tape punch. Turn on both the card reader and the tape punch. From the Ready mode, depress the Start Key (PKA 1).

1. The program will read the first card and print "CD/PPT" followed by the six character program identification contained in the first card.
2. The program will next interrogate column 12 and determine which card format is present;
   a. A zero or a blank in column 12 indicates a B 3500 format.
   b. A non-blank in column 12 indicates a B 5500 format.
3. A message is printed describing the format of the cards (B 3500 or B 5500). The system will then halt.
4. If the format printed is correct, depress OCK 1 or 2 to start the program. If the format is correct but the cards are not to be printed, depress OCK 3. However, if the format is not correct, depress OCK 4. This causes the system to assume the opposite format. Execution of the program will begin again with Step 3.
5. As each card is processed, the contents will be printed unless OCK 3 was selected in Step 4.

SECTION 4
NETWORK QUALIFICATION ROUTINES

IBM 2260 CONTROL PROGRAM

I. D. No. 7-2000-003

The purpose of this program is to provide a means of testing a slave TC loaded with the IBM 2260 line discipline firmware prior to interfacing it to an IBM 360 computer. This control program will both poll and select the slave TC and can process messages received from the slave in one of three ways.

1. The message is printed and transmitted back to the 2260 TC.
2. The message is not printed but is transmitted back.
3. The message is printed but not transmitted back.

The first three characters of the message received by the Central TC determine how it shall be processed. The three characters received are compared against constants loaded in the program to determine which of the three processing routines will be used.

The Control Program in the Central TC allows the user to enter the two digit address of the slave terminal and is designed to operate with the slave in a 4 wire environment (leased line or direct connect).

SYSTEM REQUIREMENTS:

The following outlines the software programs, Datacom Processor firmware sets, and Main Memory firmware sets required for both the Central TC and the 2260 look-alike TC.

Central TC

MAIN MEMORY:	Any data communications firmware set.
DCP MEMORY:	Central TC Controller (CTCC) I. D. No. 2-1049-008
CTCC ADD ON:	I. D. No. 2-1070-054
SOFTWARE PROGRAM:	2260 Control Program I. D. No. 7-2000-003

IMPORTANT: Track 4 of Datacom Processor memory (Block 4) must remain unprotected while using the CTCC firmware.

2260 TC

MAIN MEMORY:	Any datacom firmware
DCP MEMORY:	2260 Line Control Firmware I. D. No. 2-1049-003
SOFTWARE PROGRAM:	Plymouth 4PK Handshake I. D. No. 1-1001-063

OPERATING INSTRUCTIONS

A. 2260 TC

Refer to the documentation of the 4PK Handshake program for detailed operating instructions. The following parameters should be specified:

1. Four (4) wire mode.

2. The same address must be entered for both the Send and Receive address.

NOTE: Make certain this address is used in the Central TC program also.

3. Set the Print Back code to ABC.

4. Set the Non-Print code to DEF.

B. Central TC

1. From the Ready mode, depress the Start key (PKA 1).

2. The system prints "ENTER 2 DIGIT ADDRESS". From the Alpha keyboard, enter the same two (2) character address specified in the 4PK Handshake program of 2260 TC. Terminate with any OCK.

   RECEIVING MESSAGES – The system is always capable of receiving messages. The first three characters of the message received are compared to constants loaded in the Control Program. This determines how the message is to be processed. The constants loaded in the program are:

   a. ABC – The message is printed and transmitted back to the slave unit.
   b. DEF – The message is transmitted back.

   SENDING MESSAGES – Enter from the Alpha keyboard the message to be sent. The depression of OCK 4 causes an advance to a new line. OCK's 1, 2, or 3 signify the end of the message. The message is then transferred to the data comm processor for transmission.

   To return the system to the Ready mode, first depress the Reset key and then the Ready button. A new address to control a different slave terminal may then be entered by again depressing the Start key (PKA 1).

Error Messages

The 2260 Control Program prints the following error messages.

1. INVALID ERROR DIGIT – This indicates a meaningless error flag pattern.

2. NO RESPONSE FROM REMOTE – Printed when no response is received from either a poll or a select.

3. STRANGE RESPONSE – Printed when a strange response is received from either a poll or a select.

4. NAK LIMIT REACHED – The Central TC has tried to select the slave a total of ten (10) times and each time has received a negative reply.

5. PARITY ERROR DETECTED – Persistent parity error occurs between the Central TC and the terminal.

After printing the error message, the program resets the trouble indicator and tries the operation again.

POINT TO POINT HANDSHAKE

I. D. No. 7-2000-002

The Point to Point Handshake program is a routine designed to exercise a TC in a real time Point to Point environment. Its purpose is to provide both systems representatives and field engineers a tool for checking out a Point to Point Data Comm Network.

This Handshake program will communicate with any other unit utilizing the same Point to Point line discipline, (See Series L/TC Equipment Manual for discussion of Point to Point line discipline.) i.e., communication can be from one TC to another TC or from TC to a CPU. The communication link can be via switched lines, leased lines or direct connect.

Messages to be transmitted can be entered through the typewriter keyboard or, the operator may choose one of the following automatic transmission tests.

1. INCREMENT TEST – When in this mode, the TC will initiate transmission with a message of ABCO and wait for a response from the other unit. When a message is received in this mode it will be incremented by one character and transmitted back to the other unit.

2. CONSTANT TRANSMIT – In this mode the TC will continuously transmit the message stored in its send buffer.

Messages received will be processed by the Handshake program in one of the following ways.

1. The message is printed and transmitted back to the other unit.

2. The message is transmitted back to the other unit without printing.

3. The message is printed but not transmitted back to the other unit.

4. The message is neither printed nor transmitted back – Constant Receive Mode.

The first three characters of the message received determine how it is processed. The Handshake Program compares the first three characters of the message to constants loaded in the program. The following constants are loaded by the Program.

1. ABC – The message is printed and transmitted back to the other unit.

2. DEF – The message is transmitted back to the other unit.

3. END – Any test other than constant receive is terminated.

While the above constants are loaded by the program, they may be changed by the user if desired (see PKA 3 – Change Constants).

The Point to Point Handshake Program is compatible with any main memory data comm firmware set. This Handshake requires Point to Point DCP Firmware set No. 2-1044-004.

PT - PT HANDSHAKE

OPERATING INSTRUCTIONS

A. Initialization Routines

From the Ready mode depress the start PK. A program identification message ("Point to Point Handshake") prints out. Following the I. D. message, the program will ask for the following set up parameters; Line Mode, Timeout Limit, Idle Line Timeout, and NAK/No Response Limit. (A detailed discussion of the parameters required in a Point to Point environment are covered in the Series L/TC Equipment Manual):

1. The program displays the following message "2 WIRE MODE? OCK 1, 2 = Yes - OCK 3,4 = No". The program then halts on an alpha keyboard instruction and waits for the operator to select the appropriate OCK. The function of the OCK's are as follows:

   OCK 1 = 2 wire mode; the program continues the set up operation.

   OCK 2 = 2 wire mode; the program bypasses the setup operation and proceeds to the main line program (See Section B below.).

   OCK 3 = 4 wire; the program continues the setup operation.

   OCK 4 = 4 wire; the bypasses the setup operation and proceeds to the mainline program (See Section B below.).

   The mode selected is displayed and the Program will proceed to the appropriate routine.

2. The program will then display the message "AUTO ANSWER? OCK 1,2 = Yes, OCK 3,4 = No. If Auto Answer hardware is used the No response time out value is controlled automatically by firmware. Therefore, if Auto Answer hardware is used the routine for entering the parameters (step 3 below) necessary to generate the no response time out value is skipped.

   Select OCK 1 or 2 if Auto Answer hardware is used.

   Select OCK 3 or 4 if Auto Answer hardware is not used.

   The selection made is displayed and the program proceeds to the next step.

3. The next two parameters asked for by the program are used in generating the no response time out value. The message "TIMEX? OCK 1,2 = Yes - OCK 3,4 = No" is asking if this unit will have the lessor time out value of the two units. Depression of OCK 1 or 2 is an affirmative answer. OCK 3 or 4 is a negative answer. The selection made is displayed on the journal. The message "INDEX MØDEM DELAY IN MILS" will then print and the program will halt on a numeric keyboard instruction. Enter the modem turn around time in milliseconds on the numeric keyboard and terminate with any OCK. The turn around time of Burroughs and Bell modems are as follows:

Burroughs - 211, 212, & 713	– 10 milliseconds
Bell - 202C	– 200 milliseconds
Bell - 202D	– 50 milliseconds
Direct Connect	– 0 milliseconds

The program displays the entry and proceeds to the next step.

4. The Program then displays the message "IDLELINE TIME OUT IN MINUTES?" and halts on a numeric keyboard instruction. The maximum entry is 255 minutes. The minimum entry is 0. An entry of zero will set the program to never declare an idle line time out. Enter the desired value on the numeric keyboard and terminate the entry with any OCK. The program displays the entry and proceeds to the next step.

5. The program then displays the message "SET NAK/NO RESPONSE LIMIT TO" and halts on a numeric keyboard instruction. The maximum entry is 255 and the minimum entry is 0. Enter the desired value and terminate the entry with any OCK. The entry is displayed on the journal and the initialization phase of the program is complete.

B. Mainline Routines

1. PK SELECT

When the initialization phase of the program is complete the system will halt on an Alpha keyboard instruction (TKM 130) with nine PK's enabled. Every routine in the program provides an exit back to this routine so that another test or function may be initiated. The functions of the enabled PK's are as follows:

PKA 1	Transmit Message
PKA 2	Listen Loop
PKA 3	Change Message Codes
PKA 4	Selective Change of Control Register
PKA 5	Constant Transmit
PKA 6	Constant Receive
PKA 7	Increment Test
PKB 2	End of Test
PKB 4	Disconnect

2. PKA 1 TRANSMIT MESSAGE:

Depression of PKA 1 will cause the message which has been entered thru the Alpha keyboard to be transferred to the data comm processor for transmission. After the message has been transferred, the program automatically branches to the listen loop.

3. PKA 2 LISTEN LOOP

In this routine, the program is constantly checking for a message received, a data comm error condition or an operator interrupt.

When a message is received the program proceeds to a handling routine to determine the manner in which the message is to be processed. Upon completion of the necessary processing, the program will return to the listen loop.

If a data comm error condition is recognized the type of error is displayed on the journal and the program will return to the listen loop.

When an operator interrupt condition occurs (depression of any alpha or numeric key) the program will go to the PK SELECT routine (Step B 1).

4. PKA 3 CHANGE CODE PATTERN:

Standard 3 character alpha code patterns are loaded automatically by the program. These codes may be selectively changed by the operator if desired.

Upon depression of PKA 3, the program will halt on a numeric keyboard entry. Index one of the following numeric keys to change the desired code pattern and terminate the entry with any OCK.

NUMERIC 1 – Index a 1 on the numeric keyboard to change the print back pattern. After the entry of a 1, the program will display the message "SET PRINT BACK PATTERN TO" and halt on an alpha keyboard instruction. Enter the desired 3 character code and terminate with any OCK. The program will return to the PK SELECT Routine (B 1).

NUMERIC 2 – Index a 2 on the numeric keyboard to change the non print back pattern. After the entry of a 2, the program will display the message "SET NON PRINT BACK PATTERN TO" and halt on an alpha keyboard instruction. Enter the desired 3 character code and terminate with any OCK. The program will return to the PK SELECT Routine (B 1).

5. PKA 4 CHANGE CONTROL REGISTER

This routine provides the operator with the ability to selectively change the various entries in the control register. Upon depressing PKA 4, the following operating instructions are displayed on the journal.

"OCK 1 = Change Time Our Value"
(See Section A-3 for operating instructions.)

"OCK 2 = Change Idle Line Time Out"
(See Section A-4 for operating instructions.)

"OCK 3 = Change NAK/NO Response Limit"
(See Section A-5 for operating instructions.)

After the desired change has been completed the program will return to the PK SELECT Routine.

6. PKA 5 CONSTANT TRANSMIT

This routine will cause a standard message to be constantly transmitted by the TC. When the program recognizes that a message has been sent, it will immediately set its transmit ready flag to transmit again. A count, which is displayed when the routine is terminated, is accumulated for each successful transmission. The depression of any key will cause the routine to terminate and the transmission count to be displayed. After the routine is terminated, the program will go to the LISTEN ROUTINE. When running this routine to another TC also using the Point to Point Handshake, the other TC should be in a constant receive mode (PKA 6).

7. PKA 6 CONSTANT RECEIVE

This routine is identical to the constant transmit routine except it will constantly look for a message received. When a message is received it is counted and the program is immediately ready to receive another message. The depression of any key causes the routine to terminate and the count to be displayed. After the routine is terminated, the program will go to the LISTEN ROUTINE.

8. PKA 7 INCREMENT TEST

Depression of PKA 7 causes the program to initiate the increment test routine. The first three characters of the message will be the print back code which is currently in memory. The increment test starts with a message of "ABCO". The receiving unit, which must be in a listen mode, will transmit the message back to the initiating unit. The incrementing unit will then add one character to the message string and send it back to the "listen" unit. Either unit may terminate the test by depressing PKB 2. The test will stop automatically when it has incremented up to 255 characters.

9. PKB 2 END OF TEST

The depression of PKB 2 will cause any test in progress (other than CONSTANT TRANSMIT or CONSTANT RECEIVE) to be terminated. When the routine has been successfully terminated the message "END OF TEST" will be displayed on the journal of both units. Either unit in the network may terminate a test, regardless of which unit initiated the Test.

Once a test has been successfully ended, either unit may initiate a new test.

10. PKB 4 DISCONNECT

The depression of PKB 4 will cause a DLE-EOT message to be transmitted. If auto answer hardware is used on the TC, the line will be automatically disconnected and a message is displayed on the journal. If Auto Answer hardware is not in use only the message will be displayed. The line will not be disconnected.

SECTION 5
MACHINE LANGUAGE CODING

MACHINE LANGUAGE CODING

During the debugging process, any print-out of the program instructions, flag settings, contents of Index Registers, etc., is in the form of Hexadecimal or Machine Language Code. To allow a Programmer to quickly determine exactly what has printed in Programing Language, a series of Tables has been provided to enable an easy conversion of Machine Language Code to Programmer's Language and vice versa.

Table A Lists all the instructions in alphabetical order by instruction and gives the equivalent Machine Language code; Table B lists the same instructions but in Machine Language order; Table C is used for converting decimal values to hexadecimal and vice versa.

HOW TO USE TABLE A:

Table A lists all instructions of GP 300 in Alphabetical order and shows their equivalent Machine Language Code. The first two characters represent the "OP" code (type of instruction), as well as the syllable of the word being addressed, if applicable. The lower OP code also specifies whether the instruction deals with numeric values 0:255, 256:511, or 512:767.

In most cases, the last two characters of the instruction expressed in machine language code must be determined from additional information, depending on the type of instruction. The information that is associated with a particular instruction is footnoted. Certain instructions such as "ALARM", "STOP", "CC", and several others are shown with their complete makeup in Hexadecimal Code as no references need to be made.

HOW TO USE TABLE B:

In all the Trace Routines, the print-out of the instructions and result of the instructions is in the form of Machine Language Code, and it is desirable to convert this to Programmer's Language. Table B is used identically to Table A except in reverse. For example, let's assume the following print-out occurs:

80DE

Refer to Table B and scan the OP Code column until you come to the "80" instruction. It shows this to be an "ADM" instruction. The second characters of the print-out ("DE") are converted to their equivalent decimal value by scanning the Parameter Field columns in Table C until you come to "DE". There are several possible conversions for the characters "DE". The determining factor is the lower OP code of the instruction. Since the print-out has a lower OP value of 0, use the corresponding decimal equivalent of 222 for the characters "DE". Therefore, the instruction must be "ADM 222". If the print-out had been 81DE, the decimal equivalent shown for the character "DE" with a lower OP value of 1 would be used. This would give an "ADM 478" instruction. In all other respects then, the reference to Table B works identically to Table A, except you work backwards to arrive at the answer.

HOW TO USE TABLE C:

Table C will be most often used to convert the decimal values of the programed instruction to their corresponding hexadecimal values and to convert the 3rd and 4th digit position of the machine language print-out to its decimal value. The first two digit positions of machine code print-out are translated using Table B.

MACHINE CODING

For each hexadecimal machine code value in Table C, several decimal values are possible. The decimal value chosen will depend upon which block of memory the lower OP Code of the instruction references. See example under "How to Use Table B".

EXAMPLE: Write the machine code for the instruction "TRA 314".

Step 1. From Table A, TRA becomes 38.

Step 2. From Table C, the decimal value 314 converts to hexadecimal 3A.

Step 3. From Table C, the decimal value of 314 also shows that we must add 1 to the OP lower when converting to hexadecimal. Thus our OP value of 38 for the TRA instruction becomes 39.

The machine code for "TRA 314" is 393A.

OPERATION CODE AND PARAMETER TO MACHINE CODE

TABLE A

INSTRUCTION	OP CODE		PARAMETER		SEE FOOTNOTE
	U	L	U	L	#
ADA	8	8	0:15	0:15	1
ADIR	5	4	0:15	0:15	2
ADK	8	F	0:14	0:9	
ADM	8	0	0:15	0:15	1
AL	E	D	0:15	0:15	
ALARM	0	9	8	0	
ALR	E	F	0:15	0:15	
ALTO	E	9	0:15	1:15	
AR	E	E	0:15	0:15	
ARTO	E	A	0:15	1:15	
ALTP	E	5	0	0	
BRU	7	0	0:15	0:15	3
CC	E	C	0	0	
CHG	6	6	0:15	0:15	
CLA	8	E	0:14	0:9	
CLM	D	8	0:15	0:15	1
CPA	D	A	0:15	0:15	1
DIR	5	C	0:15	0:15	2
DIV	9	A	0:15	0:15	1
DUP	E	1	0:5	0:15	
EAM	A	9	0:9	0:15	8
EX	4				4,5
EXE	4				4,5
EXL	6		0:15	0:15	4
EXZ	4				4,5

TABLE A - Continued

INSTRUCTION	OP CODE		PARAMETER		SEE FOOTNOTE
	U	L	U	L	#
IIR	5	8	0:15	0:15	2
INK	9	E	0:14	0:15	
IRCP	1	A	0:15	0:15	
LIR	5	0	0:15	0:15	2
LKBR	F	0	0:15	0:15	1
LLCR	E	0	0:15	0:15	
LLLR	E	4	0:15	0:15	
LOD	6	4			5
LPF	3	4	A	D	
LPKR	F	C	0:15	0:15	1
LPNR	F	8	0:15	0:15	1
LPR	3	1	8	A	
LRA	3	4	B	1	
LRBR	1	8	0:15	0:15	1
LRCR	E	2	0:15	0:15	
LRLR	E	6	0:15	0:15	
LSA	3	4	B	2	
LSN	3	4	A	7	
LSR	6	4	2	0:15	
LTN	3	4	A	6	
LXC	0	6	0:15	0:15	
MOD	6	0	0	0	2
MUL	8	A	0:15	0:15	1
MULR	8	C	0:15	0:15	1
NK	A	6	0:15	0:15	
NKCM	A	2	0:15	0:15	
NKR	A	4	0:15	0:15	
NKRCM	A	0	0:15	0:15	
NOP	0	8	0	0	

TABLE A - Continued

INSTRUCTION	OP CODE U	OP CODE L	PARAMETER U	PARAMETER L	SEE FOOTNOTE #
OC	E	8	0:15	0:15	
OFF	0	9	1	0	
PA	C	8	0:15	0:15	1
PAB	1	D	0:9	0:15	8
PBA	B	C	0	0:15	
PC	C	0			7
PC+	C	4			7
PC-	C	5			7
PCP	C	1			7
PKA	F	6	0:15	0:15	6
PKB	F	7	0:15	0:15	6
PN	D	4	0:14	0:15	
PNS+	D	0	0:14	0:15	
PNS-	D	1	0:14	0:15	
POS	E	B	0:9	0:15	8
RCD	C	C	0	0	
RCP	1	C	0:15	0:15	
REAM	B	9	0:9	0:15	8
REL	0	1	0	0	
REM	3	A	4	1	1
RND	B	2	0	2	
RNK	B	0	0:15	0:15	
RPF	3	C	A	D	
RPR	3	9	8	A	
RR	0	7	0	0	
RRA	3	C	B	1	
RSA	3	C	B	2	
RSN	3	C	A	7	
RST	6	5			5
RTH	3	C	A	0	

TABLE A - Continued

INSTRUCTION	OP	CODE	PARAMETER		SEE FOOTNOTE
	U	L	U	L	#
RTK	B	C	0:9	0:15	8
RTKM	B	D	0:9	0:15	8
RTN	3	C	A	6	
RXEAM	B	B	0:9	0:15	8
RXTK	B	E	0:9	0:15	8
RXTKM	B	F	0:9	0:15	8
SCP	1	4	0:15	0:15	
SET	6	7		0:15	5
SK	4				4,5
SKE	4				4,5
SKL	6		0:15	0:15	4
SKZ	4				4,5
SKP	E	3	0:5	0:15	
SLRO	0	2	0:14	0:14	
SLROS	0	3	0:15	0:15	
SRJ	2	0	0:15	0:15	3
SRR	0	4	0	0:3	9
STOP	0	0	0	0	
SUA	9	8	0:15	0:15	1
SUK	9	F	0:14	0:9	
SUM	9	0	0:15	0:15	1
TAIR	9	C	0	0:3	
TK	A	C	0:9	0:15	8
TKM	A	D	0:9	0:15	8
TRA	3	8	0:15	0:15	1
TRAB	1	5	0	0:15	
TRB	1	E	0	1:15	
TRBA	1	B	0	1:16	
TRCA	B	8	0	0:15	

TABLE A - Continued

INSTRUCTION	OP	CODE	PARAMETER		SEE FOOTNOTE
	U	L	U	L	#
TRCB	1	6	0:15	0:15	
TRCM	B	9	0	0:15	
TRF	1	7	0:15	0:15	
TRM	3	0	0:15	0:15	1
TSB	1	F	0	1:15	
XA	C	6	0:15	0:15	1
XB	0	C	0:15	0:15	
XBA	E	A	0	0:15	
XC	C	2			7
XEAM	A	B	0:9	0:15	8
XMOD	0	A	0	0	
XN	D	7	0:14	0:15	
XPA	C	A	0:15	0:15	1
XPBA	B	E	0	0:15	
XPN	D	6	0:14	0:15	
XPNS+	D	2	0:14	0:15	
XPNS-	D	3	0:14	0:15	
XTK	A	E	0:9	0:15	8
XTKM	A	F	0:9	0:15	8

OPERATION CODE AND PARAMETER TO MACHINE CODE
(DATA COMMUNICATIONS ADDITION)

TABLE A - Continued

INSTRUCTION	OP CODE		PARAMETER		SEE FOOTNOTE
	U	L	U	L	#
IRCP	1	A	0:15	0:15	11
LKBR	F	0	0:15	0:15	1,10
LPF	3	4	A	D	
LPR	3	2	4	A	
LRA	3	4	B	1	
LRBR	1	8	0:15	0:15	1,10
LSA	3	4	B	2	
LSN	3	4	A	7	
LTN	3	4	A	6	
OFF					
PAB	1	D	0:9	0:15	8
RCP	1	C	0:15	0:15	11
RPF	3	C	A	D	
RPR	3	A	4	A	
RRA	3	C	B	1	
RSA	3	C	B	2	
RSN	3	C	A	7	
RTH	3	C	A	0	
RTN	3	C	A	6	
SCP	1	4	0:15	0:15	11
TKM					
TRAB	1	5		0:15	15

TABLE A - Continued

INSTRUCTION	OP	CODE	PARAMETER		SEE FOOTNOTE
	U	L	U	L	#
TRB	1	E		1:15	12
TRBA	1	B		1:16	13
TRCB	1	6	0:15	0:15	16
TRF	1	7	0:15	0:15	14
TSB	1	F		1:15	12

MACHINE CODING
TABLE B

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

NOTE: An asterisk denotes an instruction as being restricted in use to 40 track systems only.

TABLE B

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
0	0	STOP	0	0	
0	1	REL	0	0	
0	2	SLRO	0:14	0:14	
0	3	SLROS	0:15	0:15	
0	4	SRR	0	0	9
0	4	SRR	0	1	9
0	4	SRR	0	2	9
0	4	SRR	0	3	9
0	6	LXC	0:15	0:15	
0	7	RR	0	0	
0	8	NOP	0	0	
0	9	ALARM	8	0	
0	9	OFF	1	0	
0	A	XMOD	0	0	
0	C	XB	0:15	0:15	
1	4	SCP	0:15	0:15	
1	5	TRAB		0:15	
1	6	TRCB	0:15	0:15	
1	7	TRF	0:15	0:15	
1	8	LRBR	0:15	0:15	1
1	9	LRBR	0:15	0:15	1
1	A	IRCP	0:15	0:15	
* 1	A	PA	0:15	0:15	1
* 1	B	PA	0:15	0:15	1
1	B	TRBA	0:1	0:15	
* 1	C	PA	0:15	0:15	1
1	C	RCP	0:15	0:15	
1	D	PAB	0:9	0:15	8
1	E	TRB	0	1:15	

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

TABLE B - Continued

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
1	F	TSB	0	1:15	
2	0	SRJ	0:15	0:15	3
2	1	SRJ	0:15	0:15	3
* 2	2	SRJ	0:15	0:15	3
2	4	SRJ	0:15	0:15	3
2	5	SRJ	0:15	0:15	3
* 2	6	SRJ	0:15	0:15	3
2	8	SRJ	0:15	0:15	3
2	9	SRJ	0:15	0:15	3
* 2	A	SRJ	0:15	0:15	3
2	C	SRJ	0:15	0:15	3
2	D	SRJ	0:15	0:15	3
* 2	E	SRJ	0:15	0:15	3
3	0	TRM	0:15	0:15	1
3	1	TRM	0:15	0:15	1
3	2	LPR	4	A	
* 3	2	TRM	0:15	0:15	1
3	4	LTF	A	4	
3	4	LTN	A	6	
3	4	LSN	A	7	
3	4	LPF	A	D	
3	4	LRA	B	1	
3	4	LSA	B	2	
3	8	TRA	0:15	0:15	1
3	9	TRA	0:15	0:15	1
3	A	RPR	4	A	
3	A	REM	4	1	
* 3	A	TRA	0:15	0:15	1

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

TABLE B - Continued

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
* 3	B	REM	4	1	1
3	C	RTH	A	0	
3	C	RTF	A	4	
3	C	RTN	A	6	
3	C	RSN	A	7	
3	C	RPF	A	D	
3	C	RRA	B	1	
3	C	RSA	B	2	
4	0	SK			4,5
4	0	SKZ	D	0	4,5
4	1	SK			4,5
4	1	SKZ	D	0	4,5
4	2	SK			4,5
4	2	SKZ	D	0	4,5
4	3	SK			4,5
4	3	SKZ	D	0	4,5
4	4	EX			4,5
4	4	EXZ	D	0	4,5
4	5	EX			4,5
4	5	EXZ	D	0	4,5
4	6	EX			4,5
4	6	EXZ	D	0	4,5
4	7	EX			4,5
4	7	EXZ	D	0	4,5
4	8	SKE			4,5
4	9	SKE			4,5
4	A	SKE			4,5
4	B	SKE			4,5
4	C	EXE			4,5
4	D	EXE			4,5

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

TABLE B - Continued

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
4	E	EXE			4,5
4	F	EXE			4,5
5	0	LIR	0:15	0:15	2
5	1	LIR	0:15	0:15	2
5	2	LIR	0:15	0:15	2
5	3	LIR	0:15	0:15	2
5	4	ADIR	0:15	0:15	2
5	5	ADIR	0:15	0:15	2
5	6	ADIR	0:15	0:15	2
5	7	ADIR	0:15	0:15	2
5	8	IIR	0:15	0:15	2
5	9	IIR	0:15	0:15	2
5	A	IIR	0:15	0:15	2
5	B	IIR	0:15	0:15	2
5	C	DIR	0:15	0:15	2
5	D	DIR	0:15	0:15	2
5	E	DIR	0:15	0:15	2
5	F	DIR	0:15	0:15	2
6	0	MOD	0	0	2
6	1	MOD	0	0	2
6	2	MOD	0	0	2
6	3	MOD	0	0	2
6	4	LOD			5
6	4	LSR	2	0:15	
6	5	RST			5
6	6	CHG			5
6	7	SET			5
6	8	SKL	0:15	0:15	4
6	9	SKL	0:15	0:15	4
6	A	SKL	0:15	0:15	4

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

TABLE B - Continued

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
6	B	SKL	0:15	0: 15	4
6	C	EXL	0:15	0:15	4
6	D	EXL	0:15	0:15	4
6	E	EXL	0:15	0:15	4
6	F	EXL	0:15	0:15	4
7	0	BRU	0:15	0:15	3
7	1	BRU	0:15	0:15	3
* 7	2	BRU	0:15	0:15	3
7	4	BRU	0:15	0:15	3
7	5	BRU	0:15	0:15	3
* 7	6	BRU	0:15	0:15	3
7	8	BRU	0:15	0:15	3
7	9	BRU	0:15	0:15	3
* 7	A	BRU	0:15	0:15	3
7	C	BRU	0:15	0:15	3
7	D	BRU	0:15	0:15	3
* 7	E	BRU	0:15	0:15	3
8	0	ADM	0:15	0:15	1
8	1	ADM	0:15	0:15	1
* 8	2	ADM	0:15	0:15	1
8	8	ADA	0:15	0:15	1
8	9	ADA	0:15	0:15	1
8	A	MUL	0:15	0:15	1
8	B	MUL	0:15	0:15	1
8	C	MULR	0:15	0:15	1
8	D	MULR	0:15	0:15	1
8	E	CLA	0:14	0:9	
8	F	ADK	0:14	0:9	
9	0	SUM	0:15	0:15	1
9	1	SUM	0:15	0:15	1

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

TABLE B - Continued

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
*9	2	SUM	0:15	0:15	1
9	8	SUA	0:15	0:15	1
9	9	SUA	0:15	0:15	1
9	A	DIV	0:15	0:15	1
9	B	DIV	0:15	0:15	1
9	C	TAIR	0	0	
9	C	TAIR	0	1	
9	C	TAIR	0	2	
9	C	TAIR	0	3	
9	E	INK	0:14	0:15	
9	F	SUK	0:14	0:9	
A	0	NKRCM	0:15	0:15	
A	2	NKCM	0:15	0:15	
A	4	NKR	0:15	0:15	
A	6	NK	0:15	0:15	
A	9	EAM	0:9	0:15	8
A	B	XEAM	0:9	0:15	8
A	C	TK	0:9	0:15	8
A	D	TKM	0:9	0:15	8
A	E	XTK	0:9	0:15	8
A	F	XTKM	0:9	0:15	8
B	0	RNK	0:15	0:15	
B	2	CDC	2:15	0:9	
B	3	CDV	2:15	0:9	
B	8	TRCA	0	0:15	
B	9	TRCM	0	0:15	
B	9	REAM	0:9	0:15	8
B	A	XBA	0	0:15	
B	B	RXEAM	0:9	0:15	8
B	C	PBA	0	0:15	

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

TABLE B - Continued

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
B	C	RTK	0:9	0:15	8
B	D	RTKM	0:9	0:15	8
B	E	RXTK	0:9	0:15	8
B	E	XPBA	0	0:15	
B	F	RXTKM	0:9	0:15	8
C	0	PC			7
C	1	PCP			7
C	2	XC			7
C	4	PC+			7
C	5	PC-			7
C	6	XA	0:15	0:15	1
C	7	XA	0:15	0:15	1
C	8	PA	0:15	0:15	1
C	9	PA	0:15	0:15	1
C	A	XPA	0:15	0:15	1
C	B	XPA	0:15	0:15	1
*C	C	XPA	0:15	0:15	1
C	C	RCD	0	0	
C	D	LCD	0:15	0:15	
D	0	PNS+	0:14	0:15	
D	1	PNS-	0:14	0:15	
D	2	XPNS+	0:14	0:15	
D	3	XPNS-	0:14	0:15	
D	4	PN	0:14	0:15	
D	6	XPN	0:14	0:15	
D	7	XN	0:14	0:15	
D	8	CLM	0:15	0:15	1
D	9	CLM	0:15	0:15	1
D	A	CPA	0:15	0:15	1
D	B	CPA	0:15	0:15	1

NUMERIC HEXADECIMAL LIST FOR PROGRAM INSTRUCTIONS

TABLE B - Continued

OP	CODE	INSTRUCTION	PARAMETER		SEE FOOTNOTE
U	L		U	L	#
D	C	LCFR	0:15	0:15	1
D	D	LCFR	0:15	0:15	1
* D	E	LCFR	0:15	0:15	1
E	0	LLCR	0:15	0:15	
E	1	DUP	0:5	0:15	
E	2	LRCR	0:15	0:15	
E	3	SKP	0:5	0:15	
E	4	LLLR	0:15	0:15	
E	5	ALTP	0	0	
E	6	LRLR	0:15	0:15	
E	8	OC	0:15	0:15	
E	9	ALTO	0:15	1:15	
E	A	ARTO	0:15	1:15	
E	A	XBA	0	0:15	
E	B	POS	0:9	0:15	8
E	C	CC	0	0	
E	D	AL	0:15	0:15	
E	E	AR	0:15	0:15	
E	F	ALR	0:15	0:15	
F	0	LKBR	0:15	0:15	1
F	1	LKBR	0:15	0:15	1
* F	2	LKBR	0:15	0:15	1
F	6	PKA			6
F	7	PKB			6
F	8	LPNR	0:15	0:15	1
F	9	LPNR	0:15	0:15	1
* F	A	LPNR	0:15	0:15	1
F	C	LPKR	0:15	0:15	1
F	D	LPKR	0:15	0:15	1
* F	E	LPKR	0:15	0:15	1

FOOTNOTES

FOOTNOTE (Statements preceded by an asterisk apply to only extended memory machines.)

1. For word number 256:511 add 1 to the OP code lower.

   *For word number 512:767 add 2 to the OP code lower except for the following instructions which are restricted to referencing words 0 to 511 of user memory:

   ADA
   SUA
   MUL
   MULR
   DIV
   CLM
   CPA
   XA

   *The PA instruction for the 40 track system has a new OP code of 1A. For word number 256:511 add 1 to the OP code lower (1 A+1=1 B). For word number 512:767 add 2 to the OP code lower (1 A+2=1 C).

   *The REM instruction for the 40 track system has a new code of 3B41.

2. Modify OP code lower as follows:

INDEX REG. NO.	4	1	2	3
ADD TO OP LOWER	0	1	2	3

3. Modify OP code lower as follows:

Replace

Syllable	Word Number	~~Add to~~ OP Lower
0	0:255	0
1	0:255	4
2	0:255	8
3	0:255	C
0	256:511	1
1	256:511	5
2	256:511	9
3	256:511	D
*0	512:767	2
*1	512:767	6
*2	512:767	A
*3	512:767	E

4. OP code lower is derived from table below.

INSTRUCTION	# OF INSTRUCTIONS TO BE EXECUTED OR SKIPPED	OP LOWER
SK	4	0
or	1	1
SKZ	2	2
	3	3
EX	4	4
or	1	5
EXZ	2	6
	3	7
SKE	4	8
	1	9
	2	A
	3	B
EXE	4	C
	1	D
	2	E
	3	F
SKL	4	8
	1	9
	2	A
	3	B
EXL	4	C
	1	D
	2	E
	3	F

5. Parameter Upper Digit is determined as follows:

FLAG TYPE		PARAMETER UPPER
PUNCH	P	1
READ	R	0
	X	4
	Y	5
TEST	T	8
OCK'S	K	9
ACCUM	A	C
KEYBOARD	B (Buffer)	3
DATA COMM	D (Buffer)	A
SKZ		D
EKZ		D

Parameter Lower Digit is determined as follows:

FLAG GROUP	MACHINE LANGUAGE
A T P, R, X, Y, K, B, D	CODE VALUE
- 0 4	1
S L 1	2
C I 2	4
M U 3	8

EXAMPLE: If test flag L is being tested, the lower digit parameter is 2. If test flag L and U are being tested, the lower digit parameter is the sum of the representative code values 2 and 8 = 10. Since the hexadecimal representative of 10 is A, the Parameter lower digit is A.

6. Program Key Parameter Designation is determined as follows for Upper and Lower Digit Parameters.

	UPPER				LOWER			
PROGRAM KEY	8	7	6	5	4	3	2	1
WEIGHT	8	4	2	1	8	4	2	1

EXAMPLE: To determine upper and lower parameter values for keys 7 and 4. Key 7 value is 4, key 4 value is 8. The upper and lower parameter digits are then 4, 8.

EXAMPLE: To determine Upper and Lower parameter values for keys 8 and 5. Key 8 value is 8 and 5 value is 1. Both are within the Upper parameter giving an Upper parameter digit of 9 (8 + 1) and Lower parameter digit of 0.

7. PC character codes are determined from the following chart:

TC 500 CHARACTER SETS

The USASCII and Commerical character sets for the TC 500 are listed below in their collating sequence in ascending order. Each character set consists of 64 graphic characters, the Space code, and the End of Alpha code. The USASCII character set consists of the USASCII characters in columns 2, 3, 4, and 5 of the USASCII table, plus End of Alpha (NUL) and Overline. Those Commercial characters that differ from the USASCII characters are shown in parentheses.

The internal or machine language code for each character is given; this code consists of two hexadecimal digits which correspond to the column and row number of the character in the USASCII table (A = row 10, B = 11, C = 12, D = 13, E = 14, F = 15). In addition, the decimal value of each character is given as required when using Index Registers for modification.

Character	Internal Code	Indexing Value	Character	Internal Code	Indexing Value	Character	Internal Code	Indexing Value	Character	Internal Code	Indexing Value
End of Alpha (NUL)	0 0	0									
Space	2 0	32	0	3 0	48	@	4 0	64	P	5 0	80
!	2 1	33	1	3 1	49	A	4 1	65	Q	5 1	81
”	2 2	34	2	3 2	50	B	4 2	66	R	5 2	82
#	2 3	35	3	3 3	51	C	4 3	67	S	5 3	83
$	2 4	36	4	3 4	52	D	4 4	68	T	5 4	84
%	2 5	37	5	3 5	53	E	4 5	69	U	5 5	85
&	2 6	38	6	3 6	54	F	4 6	70	V	5 6	86
’	2 7	39	7	3 7	55	G	4 7	71	W	5 7	87
(	2 8	40	8	3 8	56	H	4 8	72	X	5 8	88
)	2 9	41	9	3 9	57	I	4 9	73	Y	5 9	89
*	2 A	42	:	3 A	58	J	4 A	74	Z	5 A	90
+	2 B	43	;	3 B	59	K	4 B	75	[(¾)	5 B	91
,	2 C	44	<(½)	3 C	60	L	4 C	76	\(¢)	5 C	92
-	2 D	45	=	3 D	61	M	4 D	77	](CR)	5 D	93
.	2 E	46	>(¼)	3 E	62	N	4 E	78	^ (°)	5 E	94
/	2 F	47	?	3 F	63	O	4 F	79	–	5 F	95
									~(◇)	7 E	126
									DEL	7 F	127

8.	PARAMETER LIMIT	0:150			
9.	SUBROUTINE RETURN LEVEL	1	2	3	4
	MACHINE CODE	0	1	2	3

FOOTNOTES

(DATA COMMUNCIATIONS ADDITION)

FOOTNOTE

10. Use upper and lower parameter of 0 (zero) to indicate data communication processor send or receive buffer.

11. Parameter limit 1:255

12. Parameter limit 1:15

13. Parameter limit 0:16

14. Parameter limit 0:255

15. Parameter limit 0:15

16. Column number (stick number) from USASCII table is upper parameter. Row number (level number) from USASCII table is the lower parameter.

DECIMAL TO HEXADECIMAL CONVERSION TABLE

TABLE C

DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD	
	L	U	L		L	U	L		L	U	L		L	U	L
0	0	0	0	32	0	2	0	64	0	4	0	96	0	6	0
1	0	0	1	33	0	2	1	65	0	4	1	97	0	6	1
2	0	0	2	34	0	2	2	66	0	4	2	98	0	6	2
3	0	0	3	35	0	2	3	67	0	4	3	99	0	6	3
4	0	0	4	36	0	2	4	69	0	4	4	100	0	6	4
5	0	0	5	37	0	2	5	69	0	4	5	101	0	6	5
6	0	0	6	38	0	2	6	70	0	4	6	102	0	6	6
7	0	0	7	39	0	2	7	71	0	4	7	103	0	6	7
8	0	0	8	40	0	2	8	72	0	4	8	104	0	6	8
9	0	0	9	41	0	2	9	73	0	4	9	105	0	6	9
10	0	0	A	42	0	2	A	74	0	4	A	106	0	6	A
11	0	0	B	43	0	2	B	75	0	4	B	107	0	6	B
12	0	0	C	44	0	2	C	76	0	4	C	108	0	6	C
13	0	0	D	45	0	2	D	77	0	4	D	109	0	6	D
14	0	0	E	46	0	2	E	78	0	4	E	110	0	6	E
15	0	0	F	47	0	2	F	79	0	4	F	111	0	6	F
16	0	1	0	48	0	3	0	80	0	5	0	112	0	7	0
17	0	1	1	49	0	3	1	81	0	5	1	113	0	7	1
18	0	1	2	50	0	3	2	82	0	5	2	114	0	7	2
19	0	1	3	51	0	3	3	83	0	5	3	115	0	7	3
20	0	1	4	52	0	3	4	84	0	5	4	116	0	7	4
21	0	1	5	53	0	3	5	85	0	5	5	117	0	7	5
22	0	1	6	54	0	3	6	86	0	5	6	118	0	7	6
23	0	1	7	55	0	3	7	87	0	5	7	119	0	7	7
24	0	1	8	56	0	3	8	88	0	5	8	120	0	7	8
25	0	1	9	57	0	3	9	89	0	5	9	121	0	7	9
26	0	1	A	58	0	3	A	90	0	5	A	122	0	7	A
27	0	1	B	59	0	3	B	91	0	5	B	123	0	7	B
28	0	1	C	60	0	3	C	92	0	5	C	124	0	7	C
29	0	1	D	61	0	3	D	93	0	5	D	125	0	7	D
30	0	1	E	62	0	3	E	94	0	5	E	126	0	7	E
31	0	1	F	63	0	3	F	95	0	5	F	127	0	7	F

TABLE C Cont'd.

DECIMAL TO HEXADECIMAL CONVERSION TABLE

DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD	
	L	U	L		L	U	L		L	U	L		L	U	L
128	0	8	0	160	0	A	0	192	0	C	0	224	0	E	0
129	0	8	1	161	0	A	1	193	0	C	1	225	0	E	1
130	0	8	2	162	0	A	2	194	0	C	2	226	0	E	2
131	0	8	3	163	0	A	3	195	0	C	3	227	0	E	3
132	0	8	4	164	0	A	4	196	0	C	4	228	0	E	4
133	0	8	5	165	0	A	5	197	0	C	5	229	0	E	5
134	0	8	6	166	0	A	6	198	0	C	6	230	0	E	6
135	0	8	7	167	0	A	7	199	0	C	7	231	0	E	7
136	0	8	8	168	0	A	8	200	0	C	8	232	0	E	8
137	0	8	9	169	0	A	9	201	0	C	9	233	0	E	9
138	0	8	A	170	0	A	A	202	0	C	A	234	0	E	A
139	0	8	B	171	0	A	B	203	0	C	B	235	0	E	B
140	0	8	C	172	0	A	C	204	0	C	C	236	0	E	C
141	0	8	D	173	0	A	D	205	0	C	D	237	0	E	D
142	0	8	E	174	0	A	E	206	0	C	E	238	0	E	E
143	0	8	F	175	0	A	F	207	0	C	F	239	0	E	F
144	0	9	0	176	0	B	0	208	0	D	0	240	0	F	0
145	0	9	1	177	0	B	1	209	0	D	1	241	0	F	1
146	0	9	2	178	0	B	2	210	0	D	2	242	0	F	2
147	0	9	3	179	0	B	3	211	0	D	3	243	0	F	3
148	0	9	4	180	0	B	4	212	0	D	4	244	0	F	4
149	0	9	5	181	0	B	5	213	0	D	5	245	0	F	5
150	0	9	6	182	0	B	6	214	0	D	6	246	0	F	6
151	0	9	7	183	0	B	7	215	0	D	7	247	0	F	7
152	0	9	8	184	0	B	8	216	0	D	8	248	0	F	8
153	0	9	9	185	0	B	9	217	0	D	9	249	0	F	9
154	0	9	A	186	0	B	A	218	0	D	A	250	0	F	A
155	0	9	B	187	0	B	B	219	0	D	B	251	0	F	B
156	0	9	C	188	0	B	C	220	0	D	C	252	0	F	C
157	0	9	D	189	0	B	D	221	0	D	D	253	0	F	D
158	0	9	E	190	0	B	E	222	0	D	E	254	0	F	E
159	0	9	F	191	0	B	F	223	0	D	F	255	0	F	F

TABLE C Cont'd.

DECIMAL TO HEXADECIMAL CONVERSION TABLE

DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD	
	L	U	L		L	U	L		L	U	L		L	U	L
256	1	0	0	288	1	2	0	320	1	4	0	352	1	6	0
257	1	0	1	289	1	2	1	321	1	4	1	353	1	6	1
258	1	0	2	290	1	2	2	322	1	4	2	354	1	6	2
259	1	0	3	291	1	2	3	323	1	4	3	355	1	6	3
260	1	0	4	292	1	2	4	324	1	4	4	356	1	6	4
261	1	0	5	293	1	2	5	325	1	4	5	357	1	6	5
262	1	0	6	294	1	2	6	326	1	4	6	358	1	6	6
263	1	0	7	295	1	2	7	327	1	4	7	359	1	6	7
264	1	0	8	296	1	2	8	328	1	4	8	360	1	6	8
265	1	0	9	297	1	2	9	329	1	4	9	361	1	6	9
266	1	0	A	298	1	2	A	330	1	4	A	362	1	6	A
267	1	0	B	299	1	2	B	331	1	4	B	363	1	6	B
268	1	0	C	300	1	2	C	332	1	4	C	364	1	6	C
269	1	0	D	301	1	2	D	333	1	4	D	365	1	6	D
270	1	0	E	302	1	2	E	334	1	4	E	366	1	6	E
271	1	0	F	303	1	2	F	335	1	4	F	367	1	6	F
272	1	1	0	304	1	3	0	336	1	5	0	368	1	7	0
273	1	1	1	305	1	3	1	337	1	5	1	369	1	7	1
274	1	1	2	306	1	3	2	338	1	5	2	370	1	7	2
275	1	1	3	307	1	3	3	339	1	5	3	371	1	7	3
276	1	1	4	308	1	3	4	340	1	5	4	372	1	7	4
277	1	1	5	309	1	3	5	341	1	5	5	373	1	7	5
278	1	1	6	310	1	3	6	342	1	5	6	374	1	7	6
279	1	1	7	311	1	3	7	343	1	5	7	375	1	7	7
280	1	1	8	312	1	3	8	344	1	5	8	376	1	7	8
281	1	1	9	313	1	3	9	345	1	5	9	377	1	7	9
282	1	1	A	314	1	3	A	346	1	5	A	378	1	7	A
283	1	1	B	315	1	3	B	347	1	5	B	379	1	7	B
284	1	1	C	316	1	3	C	348	1	5	C	380	1	7	C
285	1	1	D	317	1	3	D	349	1	5	D	381	1	7	D
286	1	1	E	318	1	3	E	350	1	5	E	382	1	7	E
287	1	1	F	319	1	3	F	351	1	5	F	383	1	7	F

TABLE C Cont'd.

DECIMAL TO HEXADECIMAL CONVERSION TABLE

DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD	
	L	U	L		L	U	L		L	U	L		L	U	L
384	1	8	0	416	1	A	0	448	1	C	0	480	1	E	0
385	1	8	1	417	1	A	1	449	1	C	1	481	1	E	1
386	1	8	2	418	1	A	2	450	1	C	2	482	1	E	2
387	1	8	3	419	1	A	3	451	1	C	3	483	1	E	3
388	1	8	4	420	1	A	4	452	1	C	4	484	1	E	4
389	1	8	5	421	1	A	5	453	1	C	5	485	1	E	5
390	1	8	6	422	1	A	6	454	1	C	6	486	1	E	6
391	1	8	7	423	1	A	7	455	1	C	7	487	1	E	7
392	1	8	8	424	1	A	8	456	1	C	8	488	1	E	8
393	1	8	9	425	1	A	9	457	1	C	9	489	1	E	9
394	1	8	A	426	1	A	A	458	1	C	A	490	1	E	A
395	1	8	B	427	1	A	B	459	1	C	B	491	1	E	B
396	1	8	C	428	1	A	C	460	1	C	C	492	1	E	C
397	1	8	D	429	1	A	D	461	1	C	D	493	1	E	D
398	1	8	E	430	1	A	E	462	1	C	E	494	1	E	E
399	1	8	F	431	1	A	F	463	1	C	F	495	1	E	F
400	1	9	0	432	1	B	0	464	1	D	0	496	1	F	0
401	1	9	1	433	1	B	1	465	1	D	1	497	1	F	1
402	1	9	2	434	1	B	2	466	1	D	2	498	1	F	2
403	1	9	3	435	1	B	3	467	1	D	3	499	1	F	3
404	1	9	4	436	1	B	4	468	1	D	4	500	1	F	4
405	1	9	5	437	1	B	5	469	1	D	5	501	1	F	5
406	1	9	6	438	1	B	6	470	1	D	6	502	1	F	6
407	1	9	7	439	1	B	7	471	1	D	7	503	1	F	7
408	1	9	8	440	1	B	8	472	1	D	8	504	1	F	8
409	1	9	9	441	1	B	9	473	1	D	9	505	1	F	9
410	1	9	A	442	1	B	A	474	1	D	A	506	1	F	A
411	1	9	B	443	1	B	B	475	1	D	B	507	1	F	B
412	1	9	C	444	1	B	C	476	1	D	C	508	1	F	C
413	1	9	D	445	1	B	D	477	1	D	D	509	1	F	D
414	1	9	E	446	1	B	E	478	1	D	D	510	1	F	E
415	1	9	F	447	1	B	F	479	1	D	F	511	1	F	F

TABLE C Cont'd.

DECIMAL TO HEXADECIMAL CONVERSION TABLE

DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD	
	L	U	L		L	U	L		L	U	L		L	U	L
512	2	0	0	544	2	2	0	576	2	4	0	608	2	6	0
513	2	0	1	545	2	2	1	577	2	4	1	609	2	6	1
514	2	0	2	546	2	2	2	578	2	4	2	610	2	6	2
515	2	0	3	547	2	2	3	579	2	4	3	611	2	6	3
516	2	0	4	548	2	2	4	580	2	4	4	612	2	6	4
517	2	0	5	549	2	2	5	581	2	4	5	613	2	6	5
518	2	0	6	550	2	2	6	582	2	4	6	614	2	6	6
519	2	0	7	551	2	2	7	583	2	4	7	615	2	6	7
520	2	0	8	552	2	2	8	584	2	4	8	616	2	6	8
521	2	0	9	553	2	2	9	585	2	4	9	617	2	6	9
522	2	0	A	554	2	2	A	586	2	4	A	618	2	6	A
523	2	0	B	555	2	2	B	587	2	4	B	619	2	6	B
524	2	0	C	556	2	2	C	588	2	4	C	620	2	6	C
525	2	0	D	557	2	2	D	589	2	4	D	621	2	6	D
526	2	0	E	558	2	2	E	590	2	4	E	622	2	6	E
527	2	0	F	559	2	2	F	591	2	4	F	623	2	6	F
528	2	1	0	560	2	3	0	592	2	5	0	624	2	7	0
529	2	1	1	561	2	3	1	593	2	5	1	625	2	7	1
530	2	1	2	562	2	3	2	594	2	5	2	626	2	7	2
531	2	1	3	563	2	3	3	595	2	5	3	627	2	7	3
532	2	1	4	564	2	3	4	596	2	5	4	628	2	7	4
533	2	1	5	565	2	3	5	597	2	5	5	629	2	7	5
534	2	1	6	566	2	3	6	598	2	5	6	630	2	7	6
535	2	1	7	567	2	3	7	599	2	5	7	631	2	7	7
536	2	1	8	568	2	3	8	600	2	5	8	632	2	7	8
537	2	1	9	569	2	3	9	601	2	5	9	633	2	7	9
538	2	1	A	570	2	3	A	602	2	5	A	634	2	7	A
539	2	1	B	571	2	3	B	603	2	5	B	635	2	7	B
540	2	1	C	572	2	3	C	604	2	5	C	636	2	7	C
541	2	1	D	573	2	3	D	605	2	5	D	637	2	7	D
542	2	1	E	574	2	3	E	606	2	5	E	638	2	7	E
543	2	1	F	575	2	3	F	607	2	5	F	639	2	7	F

TABLE C Cont'd.

DECIMAL TO HEXADECIMAL CONVERSION TABLE

DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD		DEC. EQUIV.	ADD TO OP	PARAMETER FIELD	
	L	U	L		L	U	L		L	U	L		L	U	L
640	2	8	0	672	2	A	0	704	2	C	0	736	2	E	0
641	2	8	1	673	2	A	1	705	2	C	1	737	2	E	1
642	2	8	2	674	2	A	2	706	2	C	2	738	2	E	2
643	2	8	3	675	2	A	3	707	2	C	3	739	2	E	3
644	2	8	4	676	2	A	4	708	2	C	4	740	2	E	4
645	2	8	5	677	2	A	5	709	2	C	5	741	2	E	5
646	2	8	6	678	2	A	6	710	2	C	6	742	2	E	6
647	2	8	7	679	2	A	7	711	2	C	7	743	2	E	7
648	2	8	8	680	2	A	8	712	2	C	8	744	2	E	8
649	2	8	9	681	2	A	9	713	2	C	9	745	2	E	9
650	2	8	A	682	2	A	A	714	2	C	A	746	2	E	A
651	2	8	B	683	2	A	B	715	2	C	B	747	2	E	B
652	2	8	C	684	2	A	C	716	2	C	C	748	2	E	C
653	2	8	D	685	2	A	D	717	2	C	D	749	2	E	D
654	2	8	E	686	2	A	E	718	2	C	E	750	2	E	E
655	2	8	F	687	2	A	F	719	2	C	F	751	2	E	F
656	2	9	0	688	2	B	0	720	2	D	0	752	2	F	0
657	2	9	1	689	2	B	1	721	2	D	1	753	2	F	1
658	2	9	2	690	2	B	2	722	2	D	2	754	2	F	2
659	2	9	3	691	2	B	3	723	2	D	3	755	2	F	3
660	2	9	4	692	2	B	4	724	2	D	4	756	2	F	4
661	2	9	5	693	2	B	5	725	2	D	5	757	2	F	5
662	2	9	6	694	2	B	6	726	2	D	6	758	2	F	6
663	2	9	7	695	2	B	7	727	2	D	7	759	2	F	7
664	2	9	8	696	2	B	8	728	2	D	8	760	2	F	8
665	2	9	9	697	2	B	9	729	2	D	9	761	2	F	9
666	2	9	A	698	2	B	A	730	2	D	A	762	2	F	A
667	2	9	B	699	2	B	B	731	2	D	B	763	2	F	B
668	2	9	C	700	2	B	C	732	2	D	C	764	2	F	C
669	2	9	D	701	2	B	D	733	2	D	D	765	2	F	D
670	2	9	E	702	2	B	E	734	2	D	E	766	2	F	E
671	2	9	F	703	2	B	F	735	2	D	F	767	2	F	F

MASK WORD

Alteration of print masks is done in hexidecimal machine language.

Mask Position 0:14

HEX CODE	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
MASK CHAR.	C	E	S	I	X		Z	D	.C	.D	Z:	D:	.X		Z,	D,

Mask Position 15

HEX CODE	0	1	2	3	8	9	A	B
MASK CHAR.		+	P	+P	F	+F	PF	PF+
SAFEGUARD	NO	NO	NO	NO	YES	YES	YES	YES
SUPPRESS PUNCTUATION	NO	YES	NO	YES	NO	YES	NO	YES
PUNCH	NO	NO	YES	YES	NO	NO	YES	YES

NUMERIC DATA WORD

The codes for decimal digit are directly equivalent to the Hexadecimal Codes (0:9).

CODES FOR FLAG POSITION 15

HEX CODES	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
FLAGS -	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
S	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
C	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
M	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

USASCII OBJECT PROGRAM TAPE FORMAT

B 5500/B 300 OUTPUT

Each program word in tape consists of 21 frames punched in USASCII code.

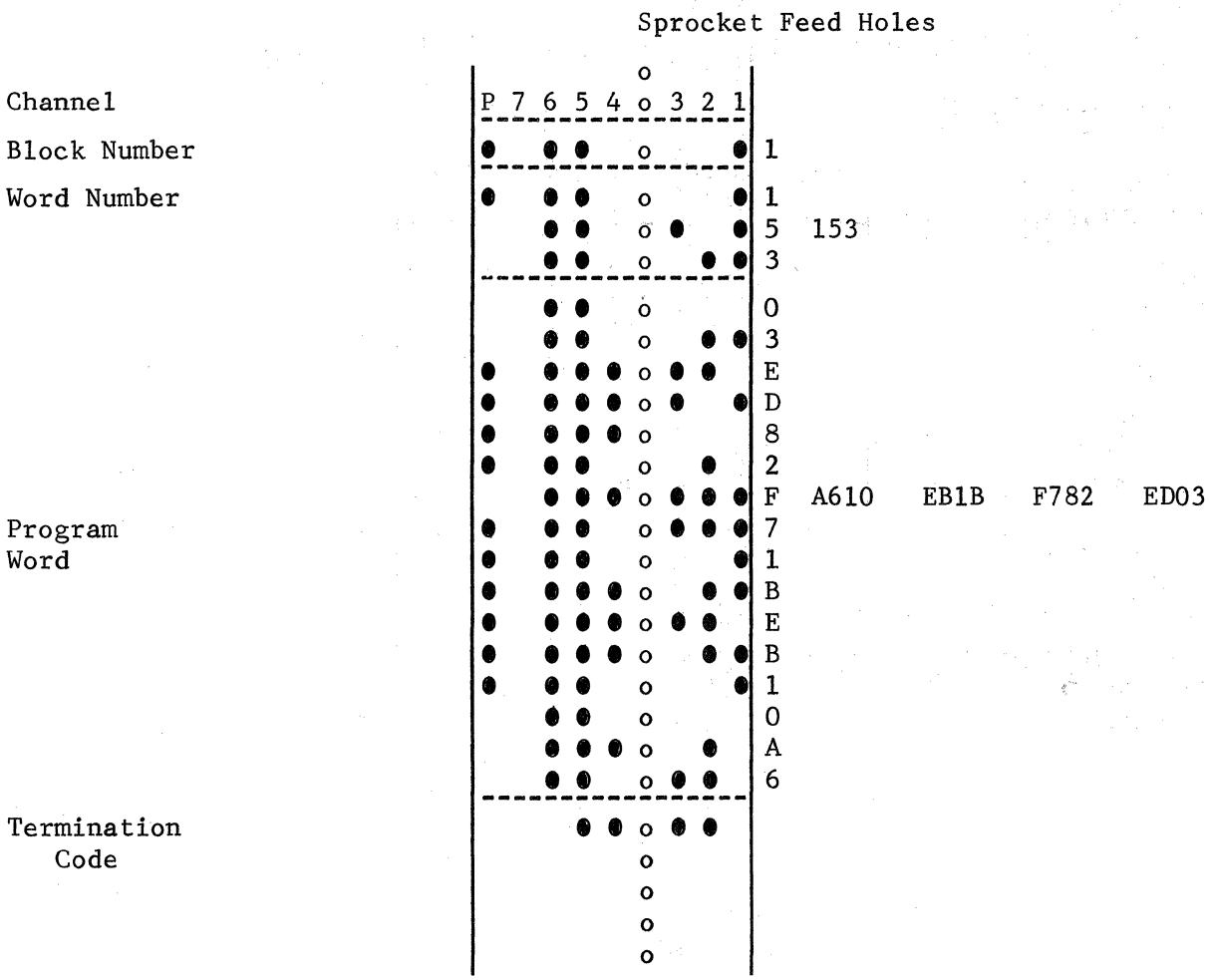
Tape Frame	
1	Block No. (0 to 3)
2-4	Word No. (000 to 255)
5-20	16 Digits:

Frame	Digit Position
5	1
6	0
7	3
8	2
9	5
10	4
11	7
12	6
13	9
14	8
15	11
16	10
17	13
18	12
19	15
20	14

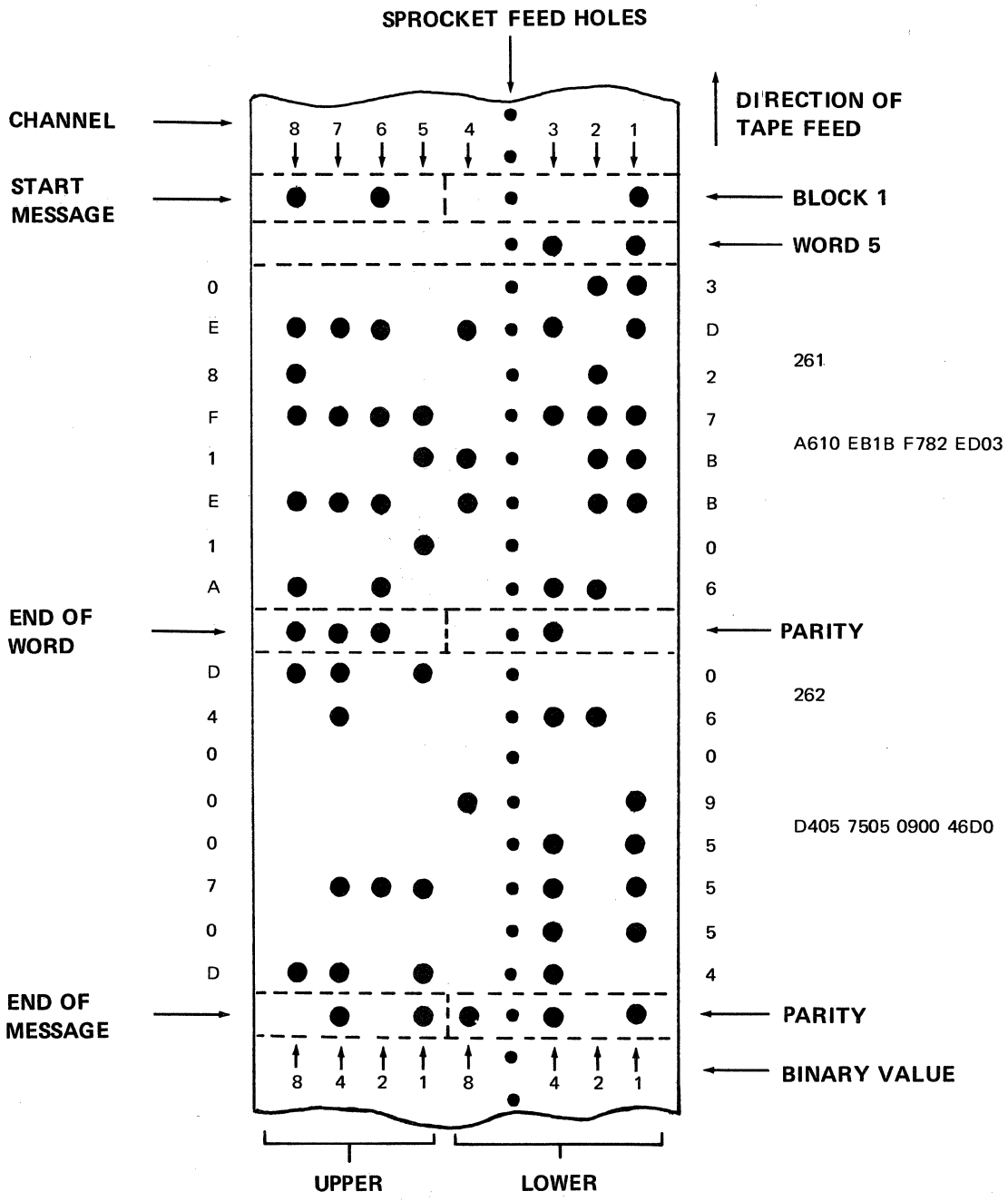
Hexadecimal digit Value expressed as code from Column 3 of USASCII table:

Hexadecimal	USASCII
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	:
B	;
C	<
D	=
E	>
F	?

21 Termination Code (1,E)


Sprocket Feed Holes
Channel
P 7 6 5 4 o 3 2 1
Block Number
Word Number
153
Program Word
A610 EB1B F782 ED03
Termination Code

SERIES L PUNCH PAPER TAPE COMPACT OBJECT CODE



261	0	ED03	1046 0
	1	F782	1047 0
	2	EB1B	1048 0
	3	A610	1049 0
262	0	46DO	1050 0
	1	0900	1051 0
	2	7505	1052 0
	3	D405	1053 0

INDEX NO.

AL	3	
PKB	28	
POS	28	
NK	1	0
EXZ	2	
ALARM		
BRU	INDEX NO.	
PN	0	5

FORMAT OF INPUT CARDS

COLUMNS		CONTENTS
B 5500 FORMAT		
1-6		Program identification
7-11		Unused
12	**	Number of words in Card (decimal)
13-16		Start Address (decimal)
17-32	*	Load Word 1
33-48	*	Load Word 2
49-64	*	Load Word 3
65-80	*	Load Word 4
B 3500 FORMAT		
1-6		Program identification
7-8		Unused
9		Start (hexadecimal)
10		Number of words in card (hexadecimal)
11-15		Unused
16		Block number
17-24	***	Load Word 1
25-32	***	Load Word 2
33-40	***	Load Word 3
41-48	***	Load Word 4
49-56	***	Load Word 5
57-64	***	Load Word 6
65-72	***	Load Word 7
73-80	***	Load Word 8

** A Zero in Column 12 indicates End of File.

* Format of Load Words:

Digits	15 - 12	11-8	7-4	3-0
Syllables	Syl 3	Syl 2	Syl 1	Syl 0
	OPPR	OPPR	OPPR	OPPR

*** Format of Load Word (card is punched binary) is the same as for B 5500.

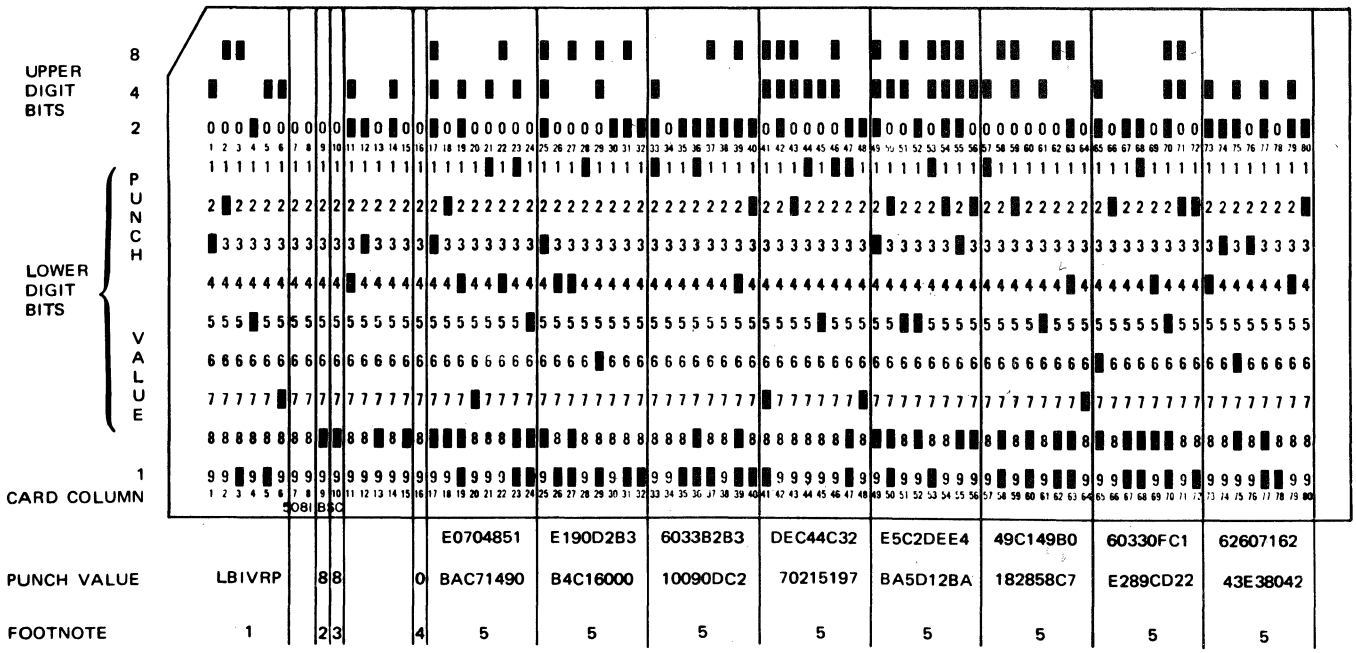
THE SERIES L PROGRAM PUNCH CARD FORMAT (COMPACT HEXADECIMAL)

Card-Column	
1 - 6	Program Identification (Alpha-in BCL Code)
9	Beginning Word Number (hexadecimal value for word 0 to 255)
10	Number of Program Words on Card (Decimal 1 to 8)
11 - 15	No significance
16	Block Number Block 0 = blank card column Block 1 = decimal 1
17 - 24 25 - 32 33 - 40 41 - 48 49 - 56 57 - 64 65 - 72 73 - 80	Up to 8 program words Each word occupies 8 card columns. Each card column contains binary value for 2 of the 16 hexadecimal digits in a word.

Hardware will cause the 12 bit representation on an 80 Column Card to be compressed into 8 bits in the Card Read Area during input, and conversely will cause the 8 bit representation in memory to be expanded into 12 bits on the output punch card in the following manner.

8 bit code representation								Card Column Punching	
Upper Digit Bits				Lower Digit Bits					
8	4	2	1	8	4	2	1		
X								12	Upper
	X							11	Digit
		X						0	Bits
							X	1	
						X		2	
						X	X	3	
					X			4	Lower
					X		X	5	Digit
					X	X		6	Bits
					X	X	X	7	
				X				8	
			X					9	Upper Digit Bits

A Sample Output Card would appear as follows:



				E0704851	E190D2B3	6033B2B3	DEC44C32	E5C2DEE4	49C149B0	60330FC1	62607162
PUNCH VALUE	LBIVRP	8	8	0	BAC71490	B4C16000	10090DC2	70215197	BA5D12BA	182858C7	E289CD22	43E38042
FOOTNOTE	1	2	3	4	5	5	5	5	5	5	5	5

FOOTNOTES

1. Program I. D. (Alpha BCL Code)
2. Beginning Word Number (Hexadecimal 0-255)
3. Number of Program Words on card (Decimal 1-8)
4. Block Number (Decimal 0-4)
5. 12 Program Words 1-8 (Compact Hexadecimal-2 Hexadecimal digits per card column)

Word No.	Syllable	3	2	1	0
1		EB0A	7C07	4184	5910
2		EB14	9C01	D620	B030
3		6100	3039	B02D	BC32
4		D7E0	C241	45C1	3927
5		EB5A	C52D	D1E2	EB4A
6		4198	C218	4598	BC07
7		6E02	3839	0CFD	C212
8		6423	6E03	7810	6422

BCL OBJECT PROGRAM CARD FORMAT

Card Column	
1-6	PROGRAM I. D.
12	Number of words in card (1 to 4)
13-16	Beginning Word Number (0000 to 1023)
17-32	1st word (digit 15 in cc 17 to digit 0 in cc 32)
33-48	2nd word (digit 15 in cc 33 to digit 0 in cc 48)
49-64	3rd word (digit 15 in cc 49 to digit 0 in cc 64)
65-80	4th word (digit 15 in cc 65 to digit 0 in cc 80)

Row Number	USASCII COLUMN 0 FIELD IDENTIFIER CODES						USASCII COLUMN 1 FIELD IDENTIFIER CODES						Row Number
			FLAG PATTERN SET BY CODE*						FLAG PATTERN SET BY CODE*				
	CODE	PAPER TAPE VALUE	Y FLAG NUMBER 3	2	1	4	CODE	PAPER TAPE VALUE	K FLAG NUMBER 3	2	1	4	
0	NUL	0, 0	0	0	0	0	DLE	9, 0	0	0	0	0	0
1	SOH	8, 1	0	0	0	1	DC1	1, 1	0	0	0	1	1
2	STX	8, 2	0	0	1	0	DC2	1, 2	0	0	1	0	2
3	ETX	0, 3	0	0	1	1	DC3	9, 3	0	0	1	1	3
4	EOT	8, 4	0	1	0	0	DC4	1, 4	0	1	0	0	4
5	ENQ	0, 5	0	1	0	1	NAK	9, 5	0	1	0	1	5
6	ACK	0, 6	0	1	1	0	SYN	9, 6	0	1	1	0	6
7	BEL	8, 7	0	1	1	1	ETB	1, 7	0	1	1	1	7
8	BS	8, 8	1	0	0	0	CAN	1, 8	1	0	0	0	8
9	HT	0, 9	1	0	0	1	EM	9, 9	1	0	0	1	9
A	IF	0, A	1	0	1	0	SUB	9, A	1	0	1	0	A
B	VT	8, B	1	0	1	1	ESC	1, B	1	0	1	1	B
C	FF	0, D	1	1	0	0	FS	9, C	1	1	0	0	C
D	CR	8, D	1	1	0	1	GS	1, D	1	1	0	1	D
E	SO	8, E	1	1	1	0	RS	1, E	1	1	1	0	E
F	SI	0, F	1	1	1	1	US	9, F	1	1	1	1	F

* 0 = flag is reset
1 = flag is set

ASCII 8 CHANNEL

PAPER TAPE INPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER

		7		6		5		4		3		2		1		0	
480	TAPE CODE	SI		FF		LF		HT		ACK		ENQ		ETX		NUL	
	TAPE IMAGE	0000	1111	0000	1100	0000	1010	0000	1001	0000	0110	0000	0101	0000	0011	0000	0000
	L2000 CODE / INTERNAL CODE	NO FLAG	8,0	NO FLAG	0,0	NO FLAG	0,0	NO FLAG	8,0	NO FLAG	0,0	NO FLAG	8,0	NO FLAG	8,0	NO FLAG	0,0
481	TAPE CODE	RS		GS		ESC		CAN		ETB		DC4		DC2		DC1	
	TAPE IMAGE	0001	1110	0001	1101	0001	1011	0001	1000	0001	0111	0001	0100	0001	0010	0001	0001
	L2000 CODE / INTERNAL CODE	K 123	1,E	K 234	K 9,D	K 134	9,B	K 3	1,8	K 124	9,7	K 2	1,4	K 1	1,2	K 4	9,1
482	TAPE CODE	.		-		+		(		'		$		"		!	
	TAPE IMAGE	0010	1110	0010	1101	0010	1011	0010	1000	0010	0111	0010	0100	0010	0010	0010	0001
	L2000 CODE / INTERNAL CODE	.	2,E	-	A,D	+	A,B	(	2,8	'	A,7	$	2,4	"	2,2	!	A,1
483	TAPE CODE	?		<		:		9		6		5		3		0	
	TAPE IMAGE	0011	1111	0011	1100	0011	1010	0011	1001	0011	0110	0011	0101	0011	0011	0011	0000
	L2000 CODE / INTERNAL CODE	?	B,F	1/2	3,C	:	3,A	9	B,9	6	3,6	5	B,5	3	B,3	0	3,0
484	TAPE CODE	N		M		K		H		G		D		B		A	
	TAPE IMAGE	0100	1110	0100	1101	0100	1011	0100	1000	0100	0111	0100	0100	0100	0010	0100	0001
	L2000 COE / INTERNAL CODE	N	4,E	M	C,D	K	C,B	H	4,8	G	C,7	D	4,4	B	4,2	A	C,1
485	TAPE CODE	—		\		Z		Y		V		U		S		P	
	TAPE IMAGE	0101	1111	0101	1100	0101	1010	0101	1001	0101	0110	0101	0101	0101	0011	0101	0000
	L2000 CODE / INTERNAL CODE	—	D,F	¢	5,C	Z	5,A	Y	D,9	V	5,6	U	D,5	S	D,3	P	5,0
486	TAPE CODE	o		l		j		i		f		e		c			
	TAPE IMAGE	0110	1111	0110	1100	0110	1010	0110	1001	0110	0110	0110	0101	0110	0011	0110	0000
	L2000 CODE / INTERNAL CODE	o	E,F	l	6,C	j	6,A	i	E,9	f	6,6	e	E,5	c	E,3		6,0
487	TAPE CODE	~		}		{		x		w		t		r		q	
	TAPE IMAGE	0111	1110	0111	1101	0111	1011	0111	1000	0111	0111	0111	0100	0111	0010	0111	0001
	L2000 CODE / INTERNAL CODE	◇	7,E	}	F,D	{	F,B	x	7,1	w	F,7	t	7,4	r	7,2	q	F,1
488	TAPE CODE	SO		CR		VT		BS		BEL		EØT		STX		SOH	
	TAPE IMAGE	1000	1110	1000	1101	1000	1011	1000	1000	1000	0111	1000	0100	1000	0010	1000	0001
	L2000 CODE / INTERNAL CODE	NO FLAG	0,0	NO FLAG	8,0	NO FLAG	8,0	NO FLAG	0,0	NO FLAG	8,0	NO FLAG	0,0	NO FLAG	0,0	NO FLAG	8,0
489	TAPE CODE	US		FS		SS		EM		SYN		NAK		DC3		DLE	
	TAPE IMAGE	1001	1111	1001	1100	1001	1010	1001	1001	1001	0110	1001	0101	1001	0011	1001	0000
	L2000 CODE / INTERNAL CODE	K 1234	9,F	K 23	1,C	K 13	1,A	K 34	9,9	K 12	1,6	K 24	9,5	K 14	9,3	K -	1,0
490	TAPE CODE	/		,		*		)		&		%		#		SPACE	
	TAPE IMAGE	1010	1111	1010	1110	1010	1010	1010	1001	1010	0110	1010	0101	1010	0011	1010	0000
	L2000 CODE / INTERNAL CODE	/	A,F	,	2,C	*	2,A	)	A,9	&	2,6	%	A,5	#	A,3	SPACE	2,0
491	TAPE CODE	>		=		;		8		7		4		2		1	
	TAPE IMAGE	1011	1110	1011	1101	1011	1011	1011	1000	1011	0111	1011	0100	1011	0010	1011	0001
	L2000 CODE / INTERNAL CODE	1/4	3,E	=	B,D	;	B,B	8	3,8	7	B,7	4	3,4	2	3,2	1	B,1
492	TAPE CODE	O		L		J		I		F		E		C		@	
	TAPE IMAGE	1100	1111	1100	1100	1100	1010	1100	1001	1100	0110	1100	0101	1100	0011	1100	0000
	L2000 CODE / INTERNAL CODE	Q	C,F	L	4,C	J	4,A	I	C,9	F	4,6	E	C,5	C	C,3	@	4,0
493	TAPE CODE	^		]		[		X		W		T		R		Q	
	TAPE IMAGE	1101	1110	1101	1101	1101	1011	1101	1000	1101	0111	1101	0100	1101	0010	1101	0001
	L2000 CODE / INTERNAL CODE	○ ^	5,E	CR	D,D	3/4	D,B	X	5,8	W	D,7	T	5,4	R	5,2	Q	D,1
494	TAPE CODE	n		m		k		h		g		d		b		a	
	TAPE IMAGE	1110	1110	1110	1101	1110	1011	1110	1000	1110	0111	1110	0100	1110	0010	1110	0001
	L2000 CODE / INTERNAL CODE	n	6,E	m	E,D	k	E,B	h	6,8	g	E,7	d	6,4	b	6,2	a	E,1
495	TAPE CODE	DEL		/		z		y		v		u		s		p	
	TAPE IMAGE	1111	1111	1111	1100	1111	1010	1111	1001	1111	0110	1111	0101	1111	0011	1111	0000
	L2000 CODE / INTERNAL CODE	IGNORE	F,F	/	7,C	z	7,A	y	F,9	v	7,6	u	F,5	s	F,3	p	7,0

Internal Values for Field Identifier and Ignore Codes

Hexadecimal Value	8 Bit Character Upper	Lower	Function
7, F or F, F	X111	1111	Ignore
0, 0:F or 8, 0:F	X000	0:15	No Flag or Y Flag Field Identifier Codes
1, 0:F or 9, 0:F	X001	0:15	K Flag Field Identifier Code

X = 0 if low order bit channel 1 of tape code is 0
1 if low order bit channel 1 of tape code is 1

In the table shown above all ASCII codes from columns 6 & 7 are coded to be interpreted as the exact ASCII codes; therefore, the printer would not print these characters nor would it escape. These characters would, however, be stored in memory under a Read to Memory instruction.

For K Field Identifier codes, the low order 4 bits of the table character position will represent the flag pattern set. For No Flag Field Identifier codes the lower 4 bits are meaningless.

An invalid code situation may be created by reversing bit 8 of any internal code in the table.

ASCII 8 CHANNEL

PAPER TAPE OUTPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER		7		6		5		4		3		2		1		0	
496	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE / TAPE CODE																
497	L2000 CODE / INTERNAL CODE	1,E	1,E	1,C	9,C	1,A	9,A	1,8	1,8	1,6	9,6	1,4	1,4	1,2	1,2	1,0	9,0
	TAPE IMAGE / TAPE CODE	0001 1110 RS		1001 1100 FS		1001 1010 SS		0001 1000 CAN		1001 0110 SYN		0001 0100 DC4		0001 0010 DC2		1001 0000 DLE	
498	L2000 CODE / INTERNAL CODE	.	2,E	,	A,C	*	A,A	(	2,8	&	A,6	$	2,4	"	2,2	SPACE	A,0
	TAPE IMAGE / TAPE CODE	0010 1110 .		1010 1100 ,		1010 1010 *		0010 1000 (		1010 0110 &		0010 0100 $		0010 0010 "		1010 0000 SPACE	
499	L2000 CODE / INTERNAL CODE	> 1/4	B,E	< 1/2	3,C	:	3,A	8	B,8	6	3,6	4	B,4	2	B,2	0	3,0
	TAPE IMAGE / TAPE CODE	1011 1110 >		0011 1100 <		0011 1010 :		1011 1000 8		0011 0110 6		1011 0100 4		1011 0010 2		0011 0000 0	
500	L2000 CODE / INTERNAL CODE	N	4,E	L	C,C	J	C,A	H	4,8	F	C,6	D	4,4	B	4,2	@	C,0
	TAPE IMAGE / TAPE CODE	0100 1110 N		1100 1100 L		1100 1010 J		0100 1000 H		1100 0110 F		0100 0100 D		0100 0010 B		1100 0000 @	
501	L2000 CODE / INTERNAL CODE	^ °	D,E	\ ¢	5,C	Z	5,A	X	D,8	V	5,6	T	D,4	R	D,2	P	5,0
	TAPE IMAGE / TAPE CODE	1101 1110 ^		0101 1100 \		0101 1010 Z		1101 1000 X		0101 0110 V		1101 0100 T		1101 0010 R		0101 0000 P	
502	L2000 CODE / INTERNAL CODE	n	E,E	l	6,C	j	6,A	h	E,8	f	6,6	d	E,4	b	E,2	\	6,0
	TAPE IMAGE / TAPE CODE	1110 1110 n		0110 1100 l		0110 1010 j		1110 1000 h		0110 0110 f		1110 0100 d		1110 0010 b		0110 0000 \	
503	L2000 CODE / INTERNAL CODE	~ ◇	7,E	¦	F,C	z	F,A	x	7,8	v	F,6	t	7,4	r	7,2	p	F,0
	TAPE IMAGE / TAPE CODE	0111 1110 ~		1111 1100 ¦		1111 1010 z		0111 1000 x		1111 0110 v		0111 0100 t		0111 0010 r		1111 0000 p	
504	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE / TAPE CODE																
505	L2000 CODE / INTERNAL CODE	1,F	9,F	1,D	1,D	1,B	1,B	1,9	9,9	1,7	1,7	1,5	9,5	1,3	9,3	1,1	1,1
	TAPE IMAGE / TAPE CODE	1001 1111 US		0001 1101 GS		0001 1011 ESC		1001 1001 EM		0001 0111 ETB		1001 0101 NAK		1001 0011 DC3		0001 0001 DC1	
506	L2000 CODE / INTERNAL CODE	/	A,F	-	2,D	+	2,B	)	A,9	'	2,7	%	A,5	#	A,3	!	2,1
	TAPE IMAGE / TAPE CODE	1010 1111 /		0010 1101 -		0010 1011 +		1010 1001)		0010 0111 '		1010 0101 %		1010 0011 #		0010 0001 !	
507	L2000 CODE / INTERNAL CODE	?	3,F	=	B,D	;	B,B	9	3,9	7	B,7	5	3,5	3	3,3	1	B,1
	TAPE IMAGE / TAPE CODE	0011 1111 ?		1011 1101 =		1011 1011 ;		0011 1001 9		1011 0111 7		0011 0101 5		0011 0011 3		1011 0001 1	
508	L2000 CODE / INTERNAL CODE	O	C,F	M	4,D	K	4,B	I	C,9	G	4,7	E	C,5	C	C,3	A	4,1
	TAPE IMAGE / TAPE CODE	1100 1111 O		0100 1101 M		0100 1011 K		1100 1001 I		0100 0111 G		1100 0101 E		1100 0011 C		0100 0001 A	
509	L2000 CODE / INTERNAL CODE	_	5,F	CR]	D,D	[3/4	D,B	Y	5,9	W	D,7	U	5,5	S	5,3	Q	D,1
	TAPE IMAGE / TAPE CODE	0101 1111 _		1101 1101]		1101 1101 [		0101 1001 Y		1101 0111 W		0101 0101 U		0101 0011 S		1101 0001 Q	
510	L2000 CODE / INTERNAL CODE	o	6,F	m	E,D	k	E,B	i	6,9	g	E,7	e	6,5	c	6,3	a	E,1
	TAPE IMAGE / TAPE CODE	0110 1111 o		1110 1101 m		1110 1011 k		0110 1001 i		1110 0111 g		0110 0101 e		0110 0011 c		1110 0001 a	
511	L2000 CODE / INTERNAL CODE	DEL	F,F	}	7,D	{	7,B	y	F,9	w	7,7	u	F,5	s	F,3	q	7,1
	TAPE IMAGE / TAPE CODE	1111 1111 DEL		0111 1101 }		0111 1011 {		1111 1001 y		0111 0111 w		1111 0101 u		1111 0011 s		0111 0001 q	

Words 497, 502, 503 (except character 7), 505, 510 and 511 contain internal codes that may not be entered through the keyboard. They may get into memory from a Tape Input instruction. Under a Punch Alpha instruction, the correct output code will be punched, however, no printing will occur nor will the printer escape. The two blank words are not used nor are they accessible with any internal code combinations.

Any ASCII codes may be punched with the XC (Punch Code) instruction or the table may be changed so that certain internal codes will punch any of the ASCII codes (refer to Table A).

ASCII/HEXADECIMAL EQUIVALENCE TABLE A.

NUL	0, 0	DLE	9, 0	SP	A, 0	0	3, 0	@	C, 0	P	5, 0	`	6, 0	p	F, 0
SOH	8, 1	DC1	1, 1	!	2, 1	1	B, 1	A	4, 1	Q	D, 1	a	E, 1	q	7, 1
STX	8, 2	DC2	1, 2	"	2, 2	2	B, 2	B	4, 2	R	D, 2	b	E, 2	r	7, 2
ETX	0, 3	DC3	9, 3	#	A, 3	3	3, 3	C	C, 3	S	5, 3	c	6, 3	s	F, 3
EOT	8, 4	DC4	1, 4	$	2, 4	4	B, 4	D	4, 4	T	D, 4	d	E, 4	t	7, 4
EN	0, 5	NAK	9, 5	%	A, 5	5	3, 5	E	C, 5	U	5, 5	e	6, 5	u	F, 5
ACK	0, 6	SYN	9, 6	&	9, 6	6	3, 6	F	C, 6	V	5, 6	f	6, 6	v	F, 6
BEL	8, 7	ETB	1, 7	'	2, 7	7	B, 7	G	4, 7	W	D, 7	g	E, 7	w	7, 7
BS	8, 8	CAN	1, 8	(	2, 8	8	B, 8	H	4. 8	X	D, 8	h	E, 8	x	7, 8
HT	0, 9	EM	9, 9	)	A, 9	9	3, 9	I	C, 9	Y	5, 9	i	6, 9	y	F, 9
LF	0, A	SUB	9, A	*	A, A	:	3, A	J	C, A	Z	5, A	j	6, A	z	F, A
VT	8, B	ESC	1, B	+	2, B	;	B, B	K	4, B	[	D, B	k	E, B	{	7, B
FF	0, C	FS	9, C	,	A, C	<	3, C	L	C, C	\	5, C	l	6, C	¦	F, C
CR	8, D	GS	1, D	-	2, D	=	B, D	M	4, D	]	D, D	m	E, D	}	7, D
SO	8, E	RS	1, E	.	2, E	>	B, E	N	4, E	^	D, E	n	E, E	~	7, E
SI	0, F	US	9, F	/	A, F	?	3, F	O	C, F	_	5, F	o	6, F	DEL	F, F

PAPER TAPE INPUT CODE TABLE

WORD NUMBER

		CHARACTER POSITION 7		6		5		4		3		2		1		0	
480	TAPE CODE	>		?		#		8		7		4		2		1	
	TAPE IMAGE	0000	1110	0000	1101	0000	1011	0000	1000	0000	0111	0000	0100	0000	0010	0000	0001
	L2000/INTERNAL CODE/CODE	1/4 >	3,E	?	B,F	#	A,3	8	3,8	7	B,7	4	3,4	2	3,2	1	B,1
481	TAPE CODE	≥		@		:		9		6		5		3		SPACE	
	TAPE IMAGE	0001	1111	0001	1100	0001	1010	0001	1001	0001	0110	0001	0101	0001	0011	0001	0000
	L2000/INTERNAL CODE/CODE	K2	9,4	@	4,0	:	3,A	9	B,9	6	3,6	5	B,5	3	B,3	SPACE	2,0
482	TAPE CODE	"		%		≠		Z		W		V		T		O	
	TAPE IMAGE	0010	1111	0010	1100	0010	1010	0010	1001	0010	0110	0010	0101	0010	0011	0010	0000
	L2000/INTERNAL CODE/CODE	"	A,2	%	2,5	K4	1,1	Z	D,A	W	5,7	V	D,6	T	D,4	O	3,0
483	TAPE CODE	]		=		,		Y		X		U		S		/	
	TAPE IMAGE	0011	1110	0011	1101	0011	1011	0011	1000	0011	0111	0011	0100	0011	0010	0011	0001
	L2000/INTERNAL CODE/CODE	]CR	5,D	=	B,D	,	A,C	Y	5,9	X	D,8	U	5,5	S	5,3	/	A,F
484	TAPE CODE	≤		*		X		R		Ø		N		L		-	
	TAPE IMAGE	0100	1111	0100	1100	0100	1010	0100	1001	0100	0110	0100	0101	0100	0011	0100	0000
	L2000/INTERNAL COE/CODE	K3	9,8	*	2,A	?	B,F	R	D,2	Ø	4,F	N	C,E	L	C,C	-	2,D
485	TAPE CODE	;		)		$		Q		P		M		K		J	
	TAPE IMAGE	0101	1110	0101	1101	0101	1011	0101	1000	0101	0111	0101	0100	0101	0010	0101	0001
	L2000/INTERNAL CODE/CODE	;	3,B	)	A,9	$	A,4	Q	5,1	P	D,0	M	4,D	K	4,B	J	C,A
486	TAPE CODE			(		.		H		G		D		B		A	
	TAPE IMAGE	0110	1110	0110	1101	0110	1011	0110	1000	0110	0111	0110	0100	0110	0010	0110	0001
	L2000/INTERNAL CODE/CODE	< 1/2	3,C	(	A,8	.	A,E	H	4,8	G	C,7	D	4,4	B	4,2	A	C,1
487	TAPE CODE	TAPE FEED				+		I		F		E		C		&	
	TAPE IMAGE	0111	1111	0111	1100	0111	1010	0111	1001	0111	0110	0111	0101	0111	0011	0111	0000
	L2000/INTERNAL CODE/CODE	IGNORE	F,F	⌐ 3/4	5,B	+	2,B	I	C,9	F	4,6	E	C,5	C	C,3	&	2,6
488	TAPE CODE															EL	
	TAPE IMAGE	1000	1111	1000	1100	1000	1010	1000	1001	1000	0110	1000	0101	1000	0011	1000	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	K1	1,2
489	TAPE CODE																
	TAPE IMAGE	1001	1110	1001	1101	1001	1011	1001	1000	1001	0111	1001	0100	1001	0010	1001	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
490	TAPE CODE																
	TAPE IMAGE	1010	1110	1010	1101	1010	1011	1010	1000	1010	0111	1010	0100	1010	0010	1010	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
491	TAPE CODE																
	TAPE IMAGE	1011	1111	1011	1100	1011	1010	1011	1001	1011	0110	1011	0101	1011	0011	1011	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
492	TAPE CODE																
	TAPE IMAGE	1100	1110	1100	1101	1100	1011	1100	1000	1100	0111	1100	0100	1100	0010	1100	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
493	TAPE CODE																
	TAPE IMAGE	1101	1111	1101	1100	1101	1010	1101	1001	1101	0110	1101	0101	1101	0011	1101	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
494	TAPE CODE																
	TAPE IMAGE	1110	1111	1110	1100	1110	1010	1110	1001	1110	0110	1110	0101	1110	0011	1110	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
495	TAPE CODE																
	TAPE IMAGE	1111	1110	1111	1101	1111	1011	1111	1000	1111	0111	1111	0100	1111	0010	1111	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F

Internal Values for Field Identifier and Ignore Codes

Hexadecimal Value	8-Bit Character Upper	Lower	Function
7,F or F,F	X111	1111	Ignore
0,0:F or 8,0:F	X000	0:15	Y Field Identifier Codes
1,0:F or 9,0:F	X001	0:15	K Field Identifier Codes

X = 0 if low order bit (channel 1) of tape code is 0
1 if low order bit (channel 1) of tape code is 1

In the above table the following BCL tape codes are used as field identifier codes and set the indicated K Flags

Tape	Image	Code	K Flag Set	K Flag Reset	Table Value
1000	0000	EL	K 1	K 2, 3, 4	1, 2
0001	1111	≥	K 2	K 1, 3, 4	9, 4
0100	1111	≤	K 3	K 1, 2, 4	9, 8
0010	1010	≠	K 4	K 1, 2, 3	1, 1

For K field identifier codes, the low order 4 bits of the table character positions will represent the flag pattern set. For those firmware sets that set Y flag field identifier codes, the low order 4 bits of the table character position represent the Y flag pattern set. For those firmware sets where no Y flags are desired, the low order 4 bits are meaningless.

An Invalid code situation may be created by reversing bit 8 of any internal code in the table.

The lower half of the table contain tape codes not found in BCL tape. They have been coded as "?" with the parity bit reversed so as to cause the Invalid Flag to be set.

BCL 8 CHANNEL

PAPER TAPE OUTPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER		7		6		5		4		3		2		1		0	
496	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE / TAPE CODE																
497	L2000 CODE / INTERNAL CODE	1,E	B,F	1,C	B,F	1,A	B,F	1,8	B,F	1,6	B,F	1,4	B,F	1,2	B,F	1,0	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
498	L2000 CODE / INTERNAL CODE	.	6,B	,	3,B	*	4,C	(	6,D	&	7,0	$	5,B	"	2,F	Sp	1,0
	TAPE IMAGE / TAPE CODE	0110	1011	0011	1011	0100	1100	0110	1101	0111	0000	0101	1011	0010	1111	0001	0000
		.		,		*		(		&		$		"		Space	
499	L2000 CODE / INTERNAL CODE	1/4 >	0,E	1/2 <	6,E	:	1,A	8	0,8	6	1,6	4	0,4	2	0,2	0	2,0
	TAPE IMAGE / TAPE CODE	0000	1101	0110	1101	0001	1010	0000	1000	0001	0110	0000	0100	0000	0010	0010	0000
		>		<		:		8		6		4		2		0	
500	L2000 CODE / INTERNAL CODE	N	4,5	L	4,3	J	5,1	H	6,8	F	7,6	D	6,4	B	6,2	@	1,C
	TAPE IMAGE / TAPE CODE	0100	0100	0100	0011	0101	0001	0110	1000	0111	0110	0110	0100	0110	0010	0001	1100
		N		L		J		H		F		D		B		@	
501	L2000 CODE / INTERNAL CODE	^ ○	B,F	\ ¢	B,F	Z	2,9	X	3,7	V	2,5	T	2,3	R	4,9	P	5,7
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	0010	1001	0011	0111	0010	0101	0010	0011	0100	1001	0101	0111
						Z		X		V		T		R		P	
502	L2000 CODE / INTERNAL CODE	n	B,F	l	B,F	j	B,F	h	B,F	f	B,F	d	B,F	b	B,F	\	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
503	L2000 CODE / INTERNAL CODE	~ ◇	B,F	/	B,F	z	B,F	x	B,F	v	B,F	t	B,F	r	B,F	p	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
504	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE / TAPE CODE																
505	L2000 CODE / INTERNAL CODE	1,F	B,F	1,D	B,F	1,B	B,F	1,9	B,F	1,7	B,F	1,5	B,F	1,3	B,F	1,1	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
506	L2000 CODE / INTERNAL CODE	/	3,1	-	4,0	+	7,A	)	5,D	'	B,F	%	2,C	#	0,B	!	B,F
	TAPE IMAGE / TAPE CODE	0011	0001	0100	0000	0111	1010	0101	1101	1011	1111	0010	1100	0000	1011	1011	1111
		/		-		+		)		'		%		#		!	
507	L2000 CODE / INTERNAL CODE	?	0,D	=	3,D	;	5,E	9	1,9	7	0,7	5	1,5	3	1,3	1	0,1
	TAPE IMAGE / TAPE CODE	0000	1101	0011	1101	0101	1110	0001	1001	0000	0111	0001	0101	0001	0011	0000	0001
		?		=		;		9		7		5		3		1	
508	L2000 CODE / INTERNAL CODE	ø	4,6	M	5,4	K	5,2	I	7,9	G	6,7	E	7,5	C	7,3	A	6,1
	TAPE IMAGE / TAPE CODE	0100	0110	0101	0100	0101	0010	0111	1001	0110	0111	0111	0101	0111	0011	0110	0001
		ø		M		K		I		G		E		C		A	
509	L2000 CODE / INTERNAL CODE	_	B,F	CR ⌋	3,E	3/4 ⌊	7,C	Y	3,8	W	2,6	U	3,4	S	3,2	Q	5,8
	TAPE IMAGE / TAPE CODE	1011	1111	0011	1110	0111	1100	0011	1000	0010	0110	0011	0100	0011	0010	0101	1000
				⌋		⌊		Y		W		U		S		Q	
510	L2000 CODE / INTERNAL CODE	o	B,F	m	B,F	k	B,F	i	B,F	g	B,F	e	B,F	c	B,F	a	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
511	L2000 CODE / INTERNAL CODE	DEL	B,F	}	B,F	{	B,F	y	B,F	v	B,F	u	B,F	s	B,F	q	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111

Words 497, 502, 503 (except character 7),505, 510 and 511 contain internal codes that may not be entered through the keyboard. They may get into memory from a Tape Input instruction. Under a Punch Alpha instruction, they will reference the output table, printing will not occur nor will the printer escape. Each of these positions in the table will output a tape code of B,F (Hexadecimal). With BCL Table Look-up on input, such a code will reference a position in the table that will cause the incoming code to be interpreted as a "?" and will set the Invalid Code Flag.

The "L" graphic characters shown in table A below do not have a BCL Tape Code couterpart and are coded as B,F (hexadecimal).

The BCL tape codes shown in table B below are not coded for punching in this output table. They may be punched with an XC (Punch Code) instruction or the table may be changed so that certain internal codes will punch any of these codes. (refer to Table C)

The two blank words are not used nor are they accessible with any internal code combinations.

A. Graphic with no BCL Counterpart

^ ○	\ ¢
!	'
_	~ ◇

B. BCL Tape Codes not coded in Output Table

x
≠
≥
≤

BCL/Hexadecimal Equivalence Table C.

A	6, 1	V	2, 5	)	5, D
B	6, 2	W	2, 6	?	0, D
C	7, 3	X	3, 7	TF	7, F
D	6, 4	Y	3, 8	≥	1, F
E	7, 5	Z	2, 9	≤	4, F
F	7, 6	1	0, 1	⌋	3, E
G	6, 7	2	0, 2	←	8, 0
H	6, 8	3	1, 3	Space	1, 0
I	7, 9	4	0, 4	/	3, 1
J	5, 1	5	1, 5	$	5, B
K	5, 2	6	1, 6	,	3, B
L	4, 3	7	0, 7	.	6, B
M	5, 4	8	0, 8	@	1, C
N	4, 5	9	1, 9	%	2, C
O	4, 6	0	2, 0	*	4, C
P	5, 7	-	4, 0	⌊	7, C
Q	5, 8	&	7, 0	#	0, B
R	4, 9	:	1, A	+	7, A
S	3, 2	x	4, A	<	6, E
T	2, 3	≠	2, A	;	5, E
U	3, 4	=	3, D	>	0, E
		(	6, D	"	2, F

PAPER TAPE INPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER		7		6		5		4		3		2		1		0	
480	TAPE CODE	EC 1		PI 7		#		8		7		4		2		1	
	TAPE IMAGE	0000	1110	0000	1101	0000	1011	0000	1000	0000	0111	0000	0100	0000	0010	0000	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	#	A,3	8	3,8	7	B,7	4	3,4	2	3,2	1	B,1
481	TAPE CODE	Dup		@		PI 1		9		6		5		3		Space	
	TAPE IMAGE	0001	1111	0001	1100	0001	1010	0001	1001	0001	0110	0001	0101	0001	0011	0001	0000
	L2000 CODE / INTERNAL CODE	K2	9,4	@	4,0	?	B,F	9	B,9	6	3,6	5	B,5	3	B,3	Sp	2,0
482	TAPE CODE	EC 2		%		PI 3		Z		W		V		T		O	
	TAPE IMAGE	0010	1111	0010	1100	0010	1010	0010	1001	0010	0110	0010	0101	0010	0011	0010	0000
	L2000 CODE / INTERNAL CODE	?	3,F	%	2,5	K4	1,1	Z	D,A	W	5,7	V	D,6	T	D,4	O	3,0
483	TAPE CODE	Skip		PI 4		,		Y		X		U		S		/	
	TAPE IMAGE	0011	1110	0011	1101	0011	1011	0011	1000	0011	0111	0011	0100	0011	0010	0011	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	,	A,C	Y	5,9	X	D,8	U	5,5	S	5,3	/	A,F
484	TAPE CODE	ERR		*		PI 2		R		Ø		N		L		-	
	TAPE IMAGE	0100	1111	0100	1100	0100	1010	0100	1001	0100	0110	0100	0101	0100	0011	0100	0000
	L2000 COE / INTERNAL CODE	K3	9,8	*	2,A	?	B,F	R	D,2	Ø	4,F	N	C,E	L	C,C	-	2,D
485	TAPE CODE	CR		PI 6		$		Q		P		M		K		J	
	TAPE IMAGE	0101	1110	0101	1101	0101	1011	0101	1000	0101	0111	0101	0100	0101	0010	0101	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	$	A,4	Q	5,1	P	D,0	M	4,D	K	4,B	J	C,A
486	TAPE CODE	Sp 2		PI 5		.		H		G		D		B		A	
	TAPE IMAGE	0110	1110	0110	1101	0110	1011	0110	1000	0110	0111	0110	0100	0110	0010	0110	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	.	A,E	H	4,8	G	C,7	D	4,4	B	4,2	A	C,1
487	TAPE CODE	Tape Feed				Sp 1		I		F		E		C		&	
	TAPE IMAGE	0111	1111	0111	1100	0111	1010	0111	1001	0111	0110	0111	0101	0111	0011	0111	0000
	L2000 CODE / INTERNAL CODE	Ig-nore	F,F	?	B,F	?	B,F	I	C,9	F	4,6	E	C,5	C	C,3	&	2,6
488	TAPE CODE																
	TAPE IMAGE	1000	1111	1000	1100	1000	1010	1000	1001	1000	0110	1000	0101	1000	0011	1000	0000
	L2000 CODE / INTERNAL CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	K1	1,2
489	TAPE CODE																
	TAPE IMAGE	1001	1110	1001	1101	1001	0111	1001	1000	1001	0111	1001	0100	1001	0010	1001	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
490	TAPE CODE																
	TAPE IMAGE	1010	1110	1010	1101	1010	1011	1010	1000	1010	0111	1010	0100	1010	0010	1010	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
491	TAPE CODE																
	TAPE IMAGE	1011	1111	1011	1100	1011	1010	1011	1001	1011	0110	1011	0101	1011	0011	1011	0000
	L2000 CODE / INTERNAL CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
492	TAPE CODE																
	TAPE IMAGE	1100	1110	1100	1101	1100	1011	1100	1000	1100	0111	1100	0100	1100	0010	1100	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
493	TAPE CODE																
	TAPE IMAGE	1101	1111	1101	1100	1101	1010	1101	1001	1101	0110	1101	0101	1101	0011	1101	0000
	L2000 CODE / INTERNAL CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
494	TAPE CODE																
	TAPE IMAGE	1110	1111	1110	1100	1110	1010	1110	1001	1110	0110	1110	0101	1110	0011	1110	0000
	L2000 CODE / INTERNAL CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
495	TAPE CODE																
	TAPE IMAGE	1111	1110	1111	1101	1111	1011	1111	1000	1111	0111	1111	0100	1111	0010	1111	0001
	L2000 CODE / INTERNAL CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F

Internal Values for Field Identifier and Ignore Codes

Hexadecimal Value	8 Bit Character Upper	Lower	Function
7,F or F,F	X111	1111	Ignore
0,0:F or 8,0:F	X000	0:15	Y Field Identifier Codes
1,0:F or 9,0:F	X001	0:15	K Field Identifier Codes

X = 0 if low order bit of tape code is 0
1 if low order bit of tape code is 1

In the above table the following IBM tape codes are used as field identifier codes and set the indicated K Flags

Tape Image	Code	K Flag Set	K Flags Reset	Table Value
1000 0000	EL	K 1	K 2,3,4	1,2
0001 1111	Dup	K 2	K 1,3,4	9,4
0100 1111	Err	K 3	K 1,2,4	9,8
0010 1010	PI 3	K 4	K 1,2,3	1,1

For K field identifier codes, the low order 4 bits of the table character position will represent the flag pattern set. For those firmware sets that set Y flag field identifier codes, the low order 4 bits of the table character position will represent the Y flag pattern set. For those firmware sets where no Y flags are desired, the low order 4 bits are meaningless.

An Invalid code situation may be created by reversing bit 8 of any internal code in the table. The lower half of the table contains tape codes not found in IBM tape. They have been coded as "?" with the parity bit reversed so as to cause the Invalid Flag to be set. In addition, certain other codes in the upper half of the table are IBM tape codes that have no ASCII internal counterparts. These too have been coded as "?" with the parity bit reversed so as to cause the Invalid Flag to be set.

IBM 8 CHANNEL

PAPER TAPE OUTPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER		7		6		5		4		3		2		1		0	
496	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE / TAPE CODE																
497	L2000 CODE / INTERNAL CODE	1,E	B,F	1,C	B,F	1,A	B,F	1,8	B,F	1,6	B,F	1,4	B,F	1,2	B,F	1,0	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
498	L2000 CODE / INTERNAL CODE	.	6,B	,	3,B	*	4,C	(	B,F	&	7,0	$	5,B	"	B,F	Sp	1,0
	TAPE IMAGE / TAPE CODE	0110	1011	0011	1011	0100	1100	1011	1111	0111	0000	0101	1011	1011	1111	0001	0000
		.		,		*				&		$				Space	
499	L2000 CODE / INTERNAL CODE	¼ >	B,F	½ <	B,F	:	B,F	8	0,8	6	1,6	4	0,4	2	0,2	0	2,0
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	0000	1000	0001	0110	0000	0100	0000	0010	0010	0000
								8		6		4		2		0	
500	L2000 CODE / INTERNAL CODE	N	4,5	L	4,3	J	5,1	H	6,8	F	7,6	D	6,4	B	6,2	@	1,C
	TAPE IMAGE / TAPE CODE	0100	0101	0100	0011	0101	0001	0110	1000	0111	0110	0110	0100	0110	0010	0001	1100
		N		L		J		H		F		D		B		@	
501	L2000 CODE / INTERNAL CODE	^ °	B,F	\ ¢	B,F	Z	2,9	X	3,7	V	2,5	T	2,3	R	4,9	P	5,7
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	0010	1001	0011	0111	0010	0101	0010	0011	0100	1001	0101	0111
						Z		X		V		T		R		P	
502	L2000 CODE / INTERNAL CODE	n	B,F	l	B,F	j	B,F	h	B,F	f	B,F	d	B,F	b	B,F	\	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
503	L2000 CODE / INTERNAL CODE	~ ◇	B,F	¦	B,F	z	B,F	x	B,F	v	B,F	t	B,F	r	B,F	p	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
504	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE / TAPE CODE																
505	L2000 CODE / INTERNAL CODE	1,F	B,F	1,D	B,F	1,B	B,F	1,9	B,F	1,7	B,F	1,5	B,F	1,3	B,F	1,1	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
506	L2000 CODE / INTERNAL CODE	/	3,1	-	4,0	+	B,F	)	B,F	'	B,F	%	2,C	#	0,B	!	B,F
	TAPE IMAGE / TAPE CODE	0011	0001	0100	0000	1011	1111	1011	1111	1011	1111	0010	1100	0000	1011	1011	1111
		/		-								%		#			
507	L2000 CODE / INTERNAL CODE	?	B,F	=	B,F	;	B,F	9	1,9	7	0,7	5	1,5	3	1,3	1	0,1
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	0001	1001	0000	0111	0001	0101	0001	0011	0000	0001
								9		7		5		3		1	
508	L2000 CODE / INTERNAL CODE	Ø	4,6	M	5,4	K	5,2	I	7,9	G	6,7	E	7,5	C	7,3	A	6,1
	TAPE IMAGE / TAPE CODE	0100	0110	0101	0100	0101	0010	0111	1001	0110	0111	0111	0101	0111	0011	0110	0001
		Ø		M		K		I		G		E		C		A	
509	L2000 CODE / INTERNAL CODE	_	B,F	CR]	5,F	3/4 [	B,F	Y	3,8	W	2,6	U	3,4	S	3,2	Q	5,8
	TAPE IMAGE / TAPE CODE	1011	1111	0101	1111	1011	1111	0011	1000	0010	0110	0011	0100	0011	0010	0101	1000
				CR				Y		W		U		S		Q	
510	L2000 CODE / INTERNAL CODE	o	B,F	m	B,F	k	B,F	i	B,F	g	B,F	e	B,F	c	B,F	a	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
511	L2000 CODE / INTERNAL CODE	DEL	B,F	}	B,F	{	B,F	y	B,F	v	B,F	u	B,F	s	B,F	q	B,F
	TAPE IMAGE / TAPE CODE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111

Words 497, 502, 503 (except char. 7), 505, 510, and 511 contain internal codes that may not be entered through the keyboard. They may get into memory from a Tape Input instruction. Under a Punch Alpha instruction, they will reference the output table, printing will not occur nor will the printer escape. Each of these spots in the table will output a tape code of B,F (hexadecimal) for an invalid "?". With an IBM Table Look-up on input, such a code will reference a spot in the table that will cause the incoming code to be interpreted as a "?" and will set the Invalid Code Flag.

The "L" graphic characters shown in Table A below do not have an IBM Tape Code counterpart and are coded as B,F (hexadecimal) for invalid "?".

The IBM tape codes shown in Table B below are not coded for punching in this output table. They may be punched with an XC (Punch Code) instruction or the table may be changed so that certain internal codes will punch these codes, (refer to Table C).

The two blank words are not used nor are they accessible with any internal code combinations.

A.

Graphics with no IBM counterpart			
(	+	"	)
> ¼	:	< ½	!
;	?	^ °	=
\ ¢	'	_	~ ◇

B.

IBM Tape Codes not coded in Output Table			
PI 1	PI 5	Dup	Sp 1
PI 2	PI 6	Skip	Sp 2
PI 3	PI 7	Err	Ec 1
PI 4	EL	¤	Ec 2

IBM/Hexadecimal Equivalence Table C.

A	6,1	V	2,5	PI 6	5,D
B	6,2	W	2,6	PI 7	0,D
C	7,3	X	3,7	TF	7,F
D	6,4	Y	3,8	Dup	1,F
E	7,5	Z	2,9	Err	4,F
F	7,6	1	0,1	Skp	3,E
G	6,7	2	0,2	El	8,0
H	6,8	3	1,3	Sp	1,0
I	7,9	4	0,4	/	3,1
J	5,1	5	1,5	$	5,B
K	5,2	6	1,6	,	3,B
L	4,3	7	0,7	.	6,B
M	5,4	8	0,8	@	1,C
N	4,5	9	1,9	%	2,C
O	4,6	0	2,0	*	4,C
P	5,7	&	7,0	¤	7,C
Q	5,8	PI 1	1,A	#	0,B
R	4,9	PI 2	4,A	Sp 1	7,A
S	3,2	PI 3	2,A	Sp 2	6,E
T	2,3	PI 4	3,D	CR	5,F
U	3,4	PI 5	6,D	EC 1	0,E
		-	4,0	EC 2	2,F

FRIDEN 8 CHANNEL

PAPER TAPE INPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER		7		6		5		4		3		2		1		0	
480	TAPE CODE	F9 (AUX 2)		F8 (AUX 1)		F1 (STOP)		8		7		4		2		1	
	TAPE IMAGE	0000	1110	0000	1101	0000	1011	0000	1000	0000	0111	0000	0100	0000	0010	0000	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	8	3,8	7	B,7	4	3,4	2	3,2	1	B,1
481	TAPE CODE	F10 (AUX 3)		F2 (AUX Sp)		+ (PI 1)		9		6		5		3		SPACE	
	TAPE IMAGE	0001	1111	0001	1100	0001	1010	0001	1001	0001	0110	0001	0101	0001	0011	0001	0000
	L2000/INTERNAL CODE/CODE	K2	9,4	?	B,F	?	B,F	9	B,9	6	3,6	5	B,5	3	B,3	Sp	2,0
482	TAPE CODE	F6 (PUN OFF)		F3 (AUX 0)		B Sp (PI 3)		Z		W		V		T		O	
	TAPE IMAGE	0010	1111	0010	1100	0010	1010	0010	1001	0010	0110	0010	0101	0010	0011	0010	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	K4	1,1	Z	D,A	W	5,7	V	D,6	T	D,4	O	3,0
483	TAPE CODE	TAB		F13 (PI 4)		,		Y		X		U		S		/	
	TAPE IMAGE	0011	1110	0011	1101	0011	1011	0011	1000	0011	0111	0011	0100	0011	0010	0011	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	.	A,C	Y	5,9	X	D,8	U	5,5	S	5,3	/	A,F
484	TAPE CODE	F11 (AUX L)		F4 (ON)		(PI 2)		R		O		N		L		-	
	TAPE IMAGE	0100	1111	0100	1100	0100	1010	0100	1001	0100	0110	0100	0101	0100	0011	0100	0000
	L2000/INTERNAL COE/CODE	K3	9,8	?	B,F	?	B,F	R	D,2	O	4,F	N	C,E	L	C,C	-	2,D
485	TAPE CODE	F7 (FCON)		F12		% _		Q		P		M		K		J	
	TAPE IMAGE	0101	1110	0101	1101	0101	1011	0101	1000	0101	0111	0101	0100	0101	0010	0101	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	_	5,B	Q	5,1	P	D,O	M	4,D	K	4,B	J	C,A
486	TAPE CODE	F5 (ON 2)		½ ¼ (AUX A)		.		H		G		D		B		A	
	TAPE IMAGE	0110	1110	0110	1101	0110	1011	0110	1000	0110	0111	0110	0100	0110	0010	0110	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	.	A,E	H	4,8	G	C,7	D	4,4	B	4,2	A	C,1
487	TAPE CODE	TAPE FEED		UPPER CASE		LOWER CASE		I		F		E		C		& ;	
	TAPE IMAGE	0111	1111	0111	1100	0111	1010	0111	1001	0111	0110	0111	0101	0111	0011	0111	0000
	L2000/INTERNAL CODE/CODE	Ignore	F,F	?	B,F	?	B,F	I	C,9	F	4,6	E	C,5	C	C,3	&	7,0
488	TAPE CODE															CARR RET	
	TAPE IMAGE	1000	1111	1000	1100	1000	1010	1000	1001	1000	0110	1000	0101	1000	0011	1000	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	K1	1,2
489	TAPE CODE																
	TAPE IMAGE	1001	1110	1001	1101	1001	0111	1001	1000	1001	0111	1001	0100	1001	0010	1001	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
490	TAPE CODE																
	TAPE IMAGE	1010	1110	1010	1101	1010	1011	1011	1000	1010	0111	1010	0100	1010	0010	1010	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
491	TAPE CODE																
	TAPE IMAGE	1011	1111	1011	1100	1011	1010	1011	1001	1011	0110	1011	0101	1011	0011	1011	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
492	TAPE CODE																
	TAPE IMAGE	1100	1110	1100	1101	1100	1011	1100	1000	1100	0111	1100	0100	1100	0010	1100	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F
493	TAPE CODE																
	TAPE IMAGE	1101	1111	1101	1100	1101	1010	1101	1001	1101	0110	1101	0101	1101	0011	1101	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
494	TAPE CODE																
	TAPE IMAGE	1110	1111	1110	1100	1110	1010	1110	1001	1110	0110	1110	0101	1110	0011	1110	0000
	L2000/INTERNAL CODE/CODE	?	3,F	?	B,F	?	B,F	?	3,F	?	B,F	?	3,F	?	3,F	?	B,F
495	TAPE CODE																
	TAPE IMAGE	1111	1110	1111	1101	1111	1011	1111	1000	1111	0111	1111	0100	1111	0010	1111	0001
	L2000/INTERNAL CODE/CODE	?	B,F	?	3,F	?	3,F	?	B,F	?	3,F	?	B,F	?	B,F	?	3,F

Internal Values for Field Identifier and Ignore Codes

Hexadecimal Value	8 Bit Character Upper	Lower	Function
7,F or F,F	X111	1111	Ignore
0,0:F or 8,0:F	X000	0:15	No Flag or Y Flag Field Identifier Codes
1,0:F or 9,0:F	X001	0:15	K Flag Field Identifier Codes

X = 0 if low order bit (channel 1) of tape code is 0
X = 1 if low order bit (channel 1) of tape code is 1

In the above table the following Friden tape codes are used as field identifier codes and set the indicated K flag patterns

Tape	Image	Code	K Flag Set	K Flags Reset	Table Value
1000	0000	CR	OCK 1	OCK 2,3,4	1,2
0001	1111	F10	OCK 2	OCK 1,3,4	9,4
0100	1111	F11	OCK 3	OCK 1,2,4	9,8
0010	1010	BSp	OCK 4	OCK 1,2,3	1,1

Other Friden Functional Codes are coded to be ignored.

For K field identifier codes, the low order 4 bits of the table character position represent the flag pattern set. For those firmware sets that set Y flag field identifier codes, the low order 4 bits of the table character position will represent the Y flag pattern set. For those firmware sets where no Y flags are desired, the lower order 4 bits are meaningless.

An invalid code situation may be created by reversing bit 8 of any internal code in the table. The lower half of the table contains tape codes not found in Friden tape. They have been coded as "?" with the parity bit reversed so as to cause the Invalid Flag to be set. In addition, certain codes that fall in the upper half of the table are tape codes that have no ASCII internal counterparts and have been coded as "?" with the parity bit reversed so as to cause the Invalid Flag to be set.

FRIDEN 8 CHANNEL

PAPER TAPE OUTPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER		7		6		5		4		3		2		1		0	
496	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE																
	TAPE CODE																
497	L2000 CODE / INTERNAL CODE	1,E	F,F	1,C	F,F	1,A	F,F	1,8	F,F	1,6	F,F	1,4	F,F	1,2	F,F	1,0	F,F
	TAPE IMAGE	1111	1111	1111	1111	1111	1111	1111	1111	1111	1111	1111	1111	1111	1111	1111	1111
	TAPE CODE	TAPE	FEED	TAPE	FEED	TAPE	FEED	TAPE	FEED	TAPE	FEED	TAPE	FEED	TAPE	FEED	TAPE	FEED
498	L2000 CODE / INTERNAL CODE	.	6,B	,	3,B	*	0,8	(	1,9	&	7,0	$	5,B	"	4,0	SPACE	1,0
	TAPE IMAGE	0110	1011	0011	1011	0000	1000	0001	1001	0111	0000	0101	1011	0100	0000	0001	0000
	TAPE CODE	.		,		*		(		&		$		"		SPACE	
499	L2000 CODE / INTERNAL CODE	1/4	6,D	1/2	6,D	:	3,1	8	0,8	6	1,6	4	0,4	2	0,2	0	2,0
	TAPE IMAGE	0110	1101	0110	1101	0011	0001	0000	1000	0001	0110	0000	0100	0000	0010	0010	0000
	TAPE CODE	1/4		1/2		:		8		6		4		2		0	
500	L2000 CODE / INTERNAL CODE	N	4,5	L	4,3	J	5,1	H	6,8	F	7,6	D	6,4	B	6,2	@	0,2
	TAPE IMAGE	0100	0101	0100	0011	0101	0001	0110	1000	0111	0110	0110	0100	0110	0010	0000	0010
	TAPE CODE	N		L		J		H		F		D		B		@	
501	L2000 CODE / INTERNAL CODE	^ ○	B,F	¢	1,6	Z	2,9	X	3,7	V	2,5	T	2,3	R	4,9	P	5,7
	TAPE IMAGE	1011	1111	0001	0110	0010	1001	0011	0111	0010	0101	0010	0011	0100	1001	0101	0111
	TAPE CODE			¢		Z		X		V		T		R		P	
502	L2000 CODE / INTERNAL CODE	n	4,5	1	4,3	j	5,1	h	6,8	f	7,6	d	6,4	b	6,2	\	B,F
	TAPE IMAGE	0100	0101	0100	0011	0101	0001	0110	1000	0111	0110	0110	0100	0110	0010	1011	1111
	TAPE CODE	n		1		j		h		f		d		b			
503	L2000 CODE / INTERNAL CODE	~ ◇	B,F		B,F	z	2,9	x	3,7	v	2,5	t	2,3	r	4,9	p	5,7
	TAPE IMAGE	1011	1111	1011	1111	0010	1001	0011	0111	0010	0101	0010	0011	0100	1001	0101	0111
	TAPE CODE					z		x		v		t		r		p	
504	L2000 CODE / INTERNAL CODE																
	TAPE IMAGE																
	TAPE CODE																
505	L2000 CODE / INTERNAL CODE		B,F	1,D	B,F	1,B	B,F	1,9	B,F	1,7	B,F	1,5	B,F	1,3	B,F	1,1	B,F
	TAPE IMAGE	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111	1011	1111
	TAPE CODE																
506	L2000 CODE / INTERNAL CODE	/	3,1	-	4,0	+	1,A	)	2,0	'	4,A	%	5,B	#	1,3	!	0,1
	TAPE IMAGE	0011	0001	0100	0000	0001	1010	0010	0000	0100	1010	0101	1011	0001	0011	0000	0001
	TAPE CODE	/		-		+		)		'		%		#		!	
507	L2000 CODE / INTERNAL CODE	?	0,7	=	7,A	;	7,0	9	1,9	7	0,7	5	1,5	3	1,3	1	0,1
	TAPE IMAGE	0000	0111	0111	1010	0111	0000	0001	1001	0000	0111	0001	0101	0001	0011	0000	0001
	TAPE CODE	?		=		;		9		7		5		3		1	
508	L2000 CODE / INTERNAL CODE	O	4,6	M	5,4	K	5,2	I	7,9	G	6,7	E	7,5	C	7,3	A	6,1
	TAPE IMAGE	0100	0110	0101	0100	0101	0010	0111	1001	0110	0111	0111	0101	0111	0011	0110	0001
	TAPE CODE	O		M		K		I		G		E		C		A	
509	L2000 CODE / INTERNAL CODE	_	5,B	CR	B,F	3/4	7,C	Y	3,8	W	2,6	U	3,4	S	3,2	Q	5,8
	TAPE IMAGE	0101	1011	1011	1111	0111	1100	0011	1000	0010	0110	0011	0100	0011	0010	0101	1000
	TAPE CODE	_				UPPER	CASE	Y		W		U		S		Q	
510	L2000 CODE / INTERNAL CODE	o	4,6	m	5,4	k	5,2	i	7,9	g	6,7	e	7,5	c	7,3	a	6,1
	TAPE IMAGE	0100	0110	0101	0100	0101	0010	0111	1001	0110	0111	0111	0101	0111	0011	0110	0001
	TAPE CODE	o		m		k		i		g		e		c		a	
511	L2000 CODE / INTERNAL CODE	DEL	7,F	}	B,F	{	B,F	y	3,8	v	2,5	u	3,4	s	3,2	q	5,8
	TAPE IMAGE	0111	1111	1011	1111	1011	1111	0011	1000	0010	0101	0011	0100	0011	0010	0101	1000
	TAPE CODE							y		v		u		s		q	

Words 497, 502, 503 (except char. 7), 505, 510, and 511 contain internal codes that may not be entered through the keyboard. They may get into memory from a Tape Read instruction. Under a Punch Alpha instruction they will reference the output table, printing will not occur nor will the printer escape. Where a comparable Friden code exists, it is coded to be punched; where no comparable character exists a tape frame with all holes punched (hexidecimal B,F) will be punched. With Friden table look-up on input such a code will be interpreted as a "?" with the Invalid Code flag set.

The two blank words are not used nor are they accessible with any internal code combinations.

Table A below shows 3 keys and internal code combinations that will punch the indicated Friden codes. Friden functional codes shown in Table B below are not coded for punching. They may be punched with an XC (Punch Code) instruction or the table may be changed so that certain internal codes will punch any of these codes (refer to Table C).

A.

Key/Internal Code			Friden Code
~	◇	7,E	Back Space
	=	3,D	Lower Case
[	3/4	5,B	Upper Case

B.

Friden functional codes not coded in output table	
CR	F6
TAB	F7
F1 Stop	F8
F2	F9
F3 Pr Res	F10
F4	F11
F5	F12
	F13

Friden/Hexadecimal Equivalence Table C.

CR		2,A	H	h	6,8	F4		4,C
Space		1,0	I	i	7,9	F5		6,E
Tab		3,E	J	j	5,1	F6		2,E
Bk Sp		2,A	K	k	5,2	F7		5,E
0	)	2,0	L	l	4,3	F8		0,D
1	!	0,1	M	m	5,4	F9		0,E
2	@	0,2	N	n	4,5	F10		1,F
3	#	1,3	O	o	4,6	F11		4,F
4	$	0,4	P	p	5,7	F12		5,D
5		1,5	Q	q	5,8	F13		3,D
6	¢	1,6	R	r	4,9	Upper Cs		7,C
7	?	0,7	S	s	3,2	Lower Cs		7,A
8	*	0,8	T	t	2,3	.	.	6,B
9	(	1,9	U	u	3,4	,	,	3,B
A	a	6,1	V	v	2,5	-	"	4,0
B	b	6,2	W	w	2,6	/	:	3,1
C	c	7,3	X	x	3,7	%	-	5,B
D	d	6,4	Y	y	3,8	&	;	7,0
E	e	7,5	Z	z	2,9	+		1,A
F	f	7,6	F1 Stop		0,B	½	¼	6,D
G	g	6,7	F2		1,C		'	4,A
			F3 PrRes		2,C	Tape Feed		7,F

5 CHANNEL TELETYPE (BAUDOT)

PAPER TAPE INPUT CODE TABLE

CHARACTER POSITION

WORD NUMBER		7	6	5	4	3	2	1	0	
496	TAPE CODE / TAPE IMAGE	C 01 110	I 01 100	R 01 010	Line Feed 01 000	N 00 110	Space 00 100	Carr. Ret. 00 010	Blank 00 000	LETTERS SHIFT
	L2000 CODE / INTERNAL CODE	C 4,3	I 4,9	R 5,2	K 1 8,2	N 4,E	Sp A,0	K 2 8,4	Ignore 9,0	
497	TAPE CODE / TAPE IMAGE	K 11 110	U 11 100	J 11 010	A 11 000	F 10 110	S 10 100	D 10 010	E 10 000	
	L2000 CODE / INTERNAL CODE	K 4,B	U 5,5	J 4,A	A 4,1	F 4,6	S 5,3	D 4,4	E 4,5	
498	TAPE CODE / TAPE IMAGE	V 01 111	P 01 101	G 01 011	L 01 001	M 00 111	H 00 101	Ø 00 011	T 00 001	
	L2000 CODE / INTERNAL CODE	V 5,6	P 5,0	G 4,7	L 4,C	M 4,D	H 4,8	Ø 4,F	T 5,4	
499	TAPE CODE / TAPE IMAGE	Ltrs 11 111	Q 11 101	Figs 11 011	W 11 001	X 10 111	Y 10 101	B 10 011	Z 10 001	
	L2000 CODE / INTERNAL CODE	0,0	Q 5,1	0,0	W 5,7	X 5,8	Y 5,9	B 4,2	Z 5,A	
500	TAPE CODE / TAPE IMAGE	: 01 110	8 01 100	4 01 010	Line Feed 01 000	, 00 110	Space 00 100	Carr. Ret. 00 010	Blank 00 000	FIGURES SHIFT
	L2000 CODE / INTERNAL CODE	: 3,A	8 3,8	4 3,4	K 1 8,2	, 2,C	Sp A,0	K 2 8,4	Ignore 9,0	
501	TAPE CODE / TAPE IMAGE	(11 110	7 11 100	' 11 010	- 11 000	! 10 110	Bell 10 100	$ 10 010	3 10 000	
	L2000 CODE / INTERNAL CODE	(2,8	7 3,7	' 2,7	- 2,D	! 2,1	Ignore 9,0	$ 2,4	3 3,3	
502	TAPE CODE / TAPE IMAGE	; 01 111	0 01 101	& 01 011	) 01 001	. 00 111	# 00 101	9 00 011	5 00 001	
	L2000 CODE / INTERNAL CODE	; 3,B	0 3,0	& 2,6	) 2,9	. 2,E	# 2,3	9 3,9	5 3,5	
503	TAPE CODE / TAPE IMAGE	Ltrs 11 111	1 11 101	Figs 11 011	2 11 001	/ 10 111	6 10 101	? 10 011	" 10 001	
	L2000 CODE / INTERNAL CODE	0,0	1 3,1	0,0	2 3,2	/ 2,F	6 3,6	? 3,F	" 2,2	

This Chart illustrates an input table for a standard interpretation of 5 Channel Teletype code into USASCII Code. The positions shown in the table for LTRS and FIGS are never referenced and are meaningless, thus are leaded with zeros.

The Line Feed and Carriage Return Tape Codes are interpreted as Field Identifier Codes and set OCK flags 1 and 2 respectively. LF sets K1, resets K2,3,4; CR sets K2, resets K1,3,4.

The Space Tape Code is interpreted in both locations in the table to be a Space Code and to reset the status bit to the LTRS Shift. Thus, subsequent codes will reference the LTRS section of the table until the next FIGS code.

Field Identifier Codes:

An input code may be interpreted as a field identifier code (termination code) by placing an appropriate 2 digit hexadecimal value in that position of the table. This will terminate the input instruction and set an OCK flag pattern in accordance with the lower digit (4 bits). Y Flags are not affected by any input codes in 5 Channel Tape Firmware.

Internal Table Character Value		OCK Flags Set/Reset by Field Identifier Codes 1 = Set, 0 = Reset			
		OCK Flag Number			
Upper	Lower	3	2	1	4
8 or C	0	0	0	0	0
8 or C	1	0	0	0	1
8 or C	2	0	0	1	0
8 or C	3	0	0	1	1
8 or C	4	0	1	0	0
8 or C	5	0	1	0	1
8 or C	6	0	1	1	0
8 or C	7	0	1	1	1
8 or C	8	1	0	0	0
8 or C	9	1	0	0	1
8 or C	A	1	0	1	0
8 or C	B	1	0	1	1
8 or C	C	1	1	0	0
8 or C	D	1	1	0	1
8 or C	E	1	1	1	0
8 or C	F	1	1	1	1

Ignore Code:

An input code will be ignored if a table internal character value is one of the following:

Upper	Lower	
9	X	"X" can be any value
D	X	

Space Code:

An input code may be interpreted as a SPACE Code only, or as a SPACE Code which also resets the shift mode to LTRS Shift, by using a table internal character value as follows:

Upper	Lower	
A	0	Space, reset at LTRS Shift
2	0	USASCII Space Code, does not affect Shift Mode

5 CHANNEL TELETYPE (BAUDOT)
PAPER TAPE OUTPUT CODE TABLE

WORD NUMBER

		CHARACTER POSITION													
		7		6		5		4		3		2		1		0	
504	L2000 / INTERNAL CODE	/	3,7[f]	-	3,8[f]	+	6,0	)	2,9[f]	'	3,A[f]	%	6,0	#	2,5[f]	!	3,6[f]
	TAPE CODE	/		-		Blank		)		'		Blank		#		!	
	TAPE IMAGE	10	111	11	000			01	001	11	010			00	101	10	110
505	L2000 / INTERNAL CODE	?	3,3[f]	=	6,0	;	2,F[f]	9	2,3[f]	7	3,C[f]	5	2,1[f]	3	3,0[f]	1	3,D[f]
	TAPE CODE	?		Blank		;		9		7		5		3		1	
	TAPE IMAGE	10	011			01	111	00	011	11	100	00	001	10	000	11	101
506	L2000 / INTERNAL CODE	Ø	0,3[1]	M	0,7[1]	K	1,E[1]	I	0,C[1]	G	0,B[1]	E	1,0[1]	C	0,E[1]	A	1,8[1]
	TAPE CODE	Ø		M		K		I		G		E		C		A	
	TAPE IMAGE	00	011	00	111	11	110	01	100	01	011	10	000	01	110	11	000
507	L2000 / INTERNAL CODE	_	6,0	CR	6,0	3/4	6,0	Y	1,5[1]	W	1,9[1]	U	1,C[1]	S	1,4[1]	Q	1,D[1]
	TAPE CODE	Blank		Blank		Blank		Y		W		U		S		Q	
	TAPE IMAGE							10	101	11	001	11	100	10	100	11	101
508	L2000 / INTERNAL CODE	.	2,7[f]	,	2,6[f]	*	6,0	(	3,E[f]	&	2,B[f]	$	3,2[f]	"	3,1[f]	Sp	4,4
	TAPE CODE	.		,		Blank		(		&		$		"		Space	
	TAPE IMAGE	00	111	00	110			11	110	01	011	10	010	10	001	00	100
509	L2000 / INTERNAL CODE	1/4	6,0	1/2	6,0	:	2,E[f]	8	2,C[f]	6	3,5[f]	4	2,A[f]	2	3,9[f]	O	2,D[f]
	TAPE CODE	Blank		Blank		:		8		6		4		2		O	
	TAPE IMAGE					01	110	01	100	10	101	01	010	11	001	01	101
510	L2000 / INTERNAL CODE	N	0,6[1]	L	0,9[1]	J	1,A[1]	H	0,5[1]	F	1,6[1]	D	1,2[1]	B	1,3[1]	@	6,0
	TAPE CODE	N		L		J		H		F		D		B		Blank	
	TAPE IMAGE	00	110	01	001	11	010	00	101	10	110	10	010	10	011		
511	L2000 / INTERNAL CODE		6,0	¢	6,0	Z	1,1[1]	X	1,7[1]	V	0,F[1]	T	0,1[1]	R	0,A[1]	P	0,D[1]
	TAPE CODE	Blank		Blank		Z		X		V		T		R		P	
	TAPE IMAGE					10	001	10	111	01	111	00	001	01	010	01	101

Values in Output Table:

The values placed in the character positions of the output table specify the pattern of holes to be punched, whether the output code is in the LTRS or FIGS shift, and provide for certain special conditions. The 8-bit value in each character position functions as follows:

Character Position in Output Table:

Upper Digit				Lower Digit				
8	4	2	1	8	4	2	1	← binary value of each bit
0	X	X	b	b	b	b	b	
				1	2	3	4	5, ← Channel Number

Represents 5-bit pattern to be punched

Designates if character is LTRS or FIGS shift:
- 0 if LTRS shift - designated "L" in table
- 1 if FIGS shift - designated "F" in table

Special Conditions:

0 if code is not subject to special conditions.
1 for special case codes in both LTRS and FIGS shift as described below:

These conditions are needed for all control codes since they are neither LTRS or FIGS characters, and since they may or may not be required to positively shift to LTRS mode. This avoids punching many unneeded LTRS and FIGS codes.

a: 0 1 0 b b b b b Punches the 5-bit code designated by "b", reverts to LTRS shift punch mode but does not punch LTRS code into tape.

b: 0 1 b b b b b Punches the 5-bit code designated by "b", a shift code is not punched and punch instruction remains in same shift.

Special condition "a" is used in the above standard output table for the Space code. Input to the L will automatically shift to Letters when a Space code is read, (assuming standard table interpretation of space), as do some other devices that use 5 channel tape. This tends to eliminate unneeded code since letters more often follow space codes then figures.

Special Condition "b" is used in the above standard output table for the Series L characters for which no 5 channel tape code exists. Blank codes are punched for each such character.

Series L Characters for which 5 Channel Codes do not exist:
The following Series L characters do not have counterpart characters in the 5 channel code set. These have been coded to punch blanks (sprocket holes) during output. A Blank tape code is interpreted to be ignored during input:

\+ % = - CR 3/4 * ¼ ½ @ ¢ ^

1047818

6-70

Printed in U.S. America